National Technical University of Athens

School of Electrical and
Computer Engineering

Communication, Electronic and Information
Engineering
Network Management and
Optimal Design Laboratory

Dynamic Resource Orchestration and Management of IoT-based Cyber-Physical Systems

Dissertation submitted for the degree of Doctor of Philosophy

of

Dimitrios Spatharakis

Athens
October, 2023

National Technical University of Athens
School of Electrical and Computer Engineering
Communication, Electronic and Information Engineering
Network Management and Optimal Design Laboratory

Dynamic Resource Orchestration and Management of IoT-based Cyber-Physical Systems

Dissertation submitted for the degree of Doctor of Philosophy

of

Dimitrios Spatharakis

Advisory Committee: Symeon Papavassiliou
Efstathios Sykas
Ioanna Roussaki

Approved by the seven-member committee on 6th October 2023.

....	....	....
S. Papavassiliou	E. Sykas	I. Roussaki
Professor NTUA	Professor NTUA	Assoc. Professor NTUA
....	....	....
G. Matsopoulos	V. Karyotis	N. Athanasopoulos
Professor NTUA	Assoc. Professor IU	Senior Lecturer QUB
	....	
	D. Soudris	
	Professor NTUA	

Athens, October 2023

....
Σπαθαράκης Δημήτριος,
Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών

Περίληψη

Με την έλευση των ασύρματων δικτύων πέμπτης γενιάς (5G) και του Διαδικτύου των Πραγμάτων (IoT), ένα κρίσιμο μέρος της ενεργοποίησης των σύγχρονων τεχνολογιών είναι η βελτιστοποίηση της απόδοσης των μηχανισμών Μεταφόρτωσης Υπολογιστικών Διεργασιών και δυναμικής κατανομής πόρων. Το τρέχον υπολογιστικό παράδειγμα παροχής πρόσθετων πόρων σε συσκευές είναι το δικτύο Υπολογισμών Πολλαπλής Πρόσβασης στα Άκρα Multi-Accessed Edge Computing (MEC), το οποίο χαρακτηρίζεται από την περιορισμένη διαθεσιμότητα πόρων. Ως εκ τούτου, οι συσκευές μαζί με τους διακομιστές edge σχηματίζουν ένα Κυβερνο-Φυσικό Σύστημα (Cyber-Physical System) το οποίο πρέπει να σχεδιαστεί ώστε να ταιριάζει στις ανάγκες κάθε εφαρμογής και σεναρίου. Έτσι, σε αυτή τη διατριβή, αντιμετωπίζουμε το πρόβλημα της υλοποίησης αποτελεσματικών στρατηγικών μεταφόρτωσης υπολογιστικών διεργασιών που ωφελούν τις συσκευές και, αφετέρου, τον προγραμματισμό εργασιών και τη δυναμική κατανομή πόρων για βέλτιστη χρήση των πόρων, επιτυγχάνοντας παράλληλα υψηλή ποιότητα εξυπηρέτησης και εμπειρίας χρήστη. Επίσης σε αυτή τη διδακτορική διατριβή, εξετάσαμε το ενδεχόμενο να σχεδιάσουμε πλαίσια και να συν-σχεδιάσουμε αυτές τις δύο θεμελιώδεις πτυχές καθώς είναι στενά συνδεδεμένες.

Συγκεκριμένα, σε αυτή την εργασία, καταβάλλεται προσπάθεια να αντιμετωπιστούν συνδυαστικά οι ακόλουθες κρίσιμες ερευνητικές προκλήσεις: (i) βελτιστοποίηση της απόφασης μεταφόρτωσης υπολογιστικών διεργασιών λαμβάνοντας υπόψη πτυχές που σχετίζονται με τη συσκευή (π.χ. ισχύς σήματος, θέση συσκευών), (ii) βελτιστοποίηση της διαχείρισης πόρων με από κοινού σχεδιασμό του προγραμματισμού των εργασιών και ενός μηχανισμού δυναμικής κατανομής πόρων, ο οποιός εγγυάται ορισμένα κριτήρια απόδοσης ενώ ταυτόχρονα οι πόροι χρησιμοποιούνται βέλτιστα, (iii) τον καθορισμό του κατάλληλου αριθμού πόρων για κάθε εφαρμογή, (iv) ο σχεδιασμός πρακτικών υπολογιστικών στρατηγικών μεταφόρτωσης, για την ελάφρυνση του υπολογιστικού φόρτου από τις συσκευές με ταυτόχρονη εξεύρεση ισορροπίας μεταξύ του χρόνου απόκρισης και της ποιότητας της εφαρμογής και, τέλος, (v) ο σχεδιασμός ελεγκτών κλειστού βρόχου λαμβάνοντας επίσης υπόψη την τρέχουσα κατάσταση των συσκευών, το κόστος επικοινωνίας και τους διαθέσιμους πόρους στο δικτύο Υπολογισμών Πολλαπλής Πρόσβασης.

Για το σκοπό αυτό, σχεδιάζουμε μια ολιστική αρχιτεκτονική για τη βελτιστοποίηση της απόφασης μεταφόρτωσης των χρηστών, δηλαδή, οι χρήστες μεταφορτώνουν το αίτημά τους εάν το κόστος επικοινωνίας είναι χαμηλό. Τα αιτήματα μιας εφαρμογής μπορεί επίσης να απορριφθούν από τον διακομιστή ακμών εάν το συνολικό πλήθος αιτημάτων, υπερβαίνει την τρέχουσα χωρητικότητα των πόρων για τη συγκεκριμένη εφαρμογή. Στη συνέχεια, μια κεντρική οντότητα επιλύει από κοινού το πρόβλημα προγραμματισμού εργασιών και δυναμικής κατανομής πόρων διατηρώντας ένα συγκεκριμένο επίπεδο εξυπηρέτησης για την εφαρμογή, σε δυναμικό φορτίο.

Στη συνέχεια, σε μια παρόμοια περιβάλλον, βελτιώνουμε τον προγραμματισμό εργασιών και τον μηχανισμό δυναμικής κατανομής πόρων χρησιμοποιώντας μοντελοποίηση από τη θεωρία ουρών για να προσαρμόσουμε δυναμικά τον αριθμό των πόρων για μια εφαρμογή. Η λύση που προτείνουμε παρέχει εγγυήσεις σταθερότητας και απόδοσης για το πρόβλημα διαχείρισης πόρων. Αυτό επιτυγχάνεται με τη βέλτιστη κατανομή των εισερχόμενων εργασιών και τη μοντελοποίηση απόδοσης των πόρων. Πιο συγκεκριμένα, χρησιμοποιήσαμε αλγόριθμους μηχανικής μάθησης για να αντιστοιχίσουμε μετρικές αποδόσεις μαζί με θεωρητικά υπολογισμένες τιμές για τον ρυθμό επεξεργασίας των πόρων.

Επιπλέον, προτείνεται ένας νέος μηχανισμός μεταφόρτωσης για ρομποτικές εφαρμογές. Στο πλαίσιο της Βιομηχανίας 4.0 για Κυβερνο-Φυσικά Συστήματα, οι εφαρμογές βασίζονται σε κινητούς ρομποτικές συσκευές που εκτελούν πολλές σύνθετες εργασίες που έχουν αυστηρές απαιτήσεις ασφάλειας και χρόνου. Σε ένα τέτοιο περιβάλλον, το μοντέλο παροχής υπηρεσιών Υπολογισμών Πολλαπλής Πρόσβασης στα Άκρα, επιτρέπει στις ρομποτικές συσκευές να μεταφορτώνουν τις υπολογιστικά βαριές διεργασίες τους. Συγκεκριμένα, σχεδιάζονται πρακτικές στρατηγικές μεταφόρτωσης για εφαρμογές σχεδιασμού διαδρομής και εντοπισμού θέσεις των ρομπότ. Η απόφαση μεταφόρτωσης βασίζεται στην αβεβαιότητα της θέσης του ρομπότ, στη διαθεσιμότητα πόρων του διακομιστή των πόρων, στην ποιότητα της σύνδεσης δικτύου και στη δυσκολία της υπολογισμένης διαδρομής.

Παρομοίως, εστιάζοντας στις εγγυήσεις σταθερότητας και στη σύγκλιση του συστήματος, εισάγουμε ένα πλαίσιο μεταφόρτωσης εκτίμησης θέσης βάσει συνόλου, για τη πλοήγηση μονόκυκλου ρομπότ από μία θέση προς μία άλλη. Το ρομπότ υπόκειται σε αβεβαιότητες μοντελοποίησης και μέτρησης και το σύνολο εκτίμησης υπολογίζεται χρησιμοποιώντας τεχνικές υπερπροσέγγισης που μετριάζουν τους πρόσθετους υπολογισμούς. Ένας μηχανισμός ελέγχου που βασίζεται σε σετ μεταγωγής παρέχει ακριβή πλοήγηση και ενεργοποιεί πιο ακριβείς αλγόριθμους εκτίμησης όταν χρειάζεται. Για να εξασφαλιστεί η σύγκλιση του συστήματος και να βελτιστοποιηθεί η χρήση απομακρυσμένων πόρων, έχει σχεδιαστεί ένας μηχανισμός μεταφόρτωσης, ο οποίος λαμβάνει υπόψη τόσο τις δυναμικές συνθήκες δικτύου όσο και τους διαθέσιμους υπολογιστικούς πόρους στην άκρη του δικτύου.

Τέλος, μελετάμε τον προγραμματισμό πολλών αιτημάτων για αναγνώριση εικόνας σε πραγματικό χρόνο με βοήθεια στα άκρα του δικτύου. Η ποιότητα της αναγνώρισης εικόνας και η συνολική καθυστέρηση του συστήματος είναι προφανώς ανταγωνιστικές μετρήσεις. Έτσι, διαμορφώνουμε ένα κοινό πρόβλημα βελτιστοποίησης για να μεγιστοποιήσουμε την ποιότητα της διαδικασίας αναγνώρισης, ελαχιστοποιώντας ταυτόχρονα τη συνολική καθυστέρηση για τη μαζική επεξεργασία των αιτημάτων χρησιμοποιώντας πόρους που χρησιμοποιούν κάρτες γραφικών. Για να αντιμετωπίσουμε την υπολογιστική πολυπλοκότητα του προβλήματος βελτιστοποίησης, υπολογίζουμε πρώτα τις βέλτιστες πολιτικές συμπίεσης για τα αιτήματα που στέλνουν οι χρήστες για να ελαχιστοποιήσουμε τον χρόνο μετάδοσης. Εξετάζοντας προσεκτικά τα αποτελέσματα του προβλήματος συμπίεσης, συμπαιρένουμε ότι η συμπίεση των εργασιών με τέτοιο τρόπο ώστε να φτάνουν ταυτόχρονα για απομακρυσμένη επεξεργασία αυξάνει σημαντικά την απόδοση της ταύτοχρονης επεξεργασίας από τη μεριά των απομακρυσμένων πόρων.

Συνοψίζοντας, σε αυτή τη διατριβή, μελετάμε το πρόβλημα της δυναμικής ενορχήστρωσης και διαχείρισης των Κυβερνοφυσικών Συστημάτων. Η τεράστια πρόοδος των σύγχρονων εφαρμογών μαζί με τις αυστηρές απαιτήσεις για συνεχώς διευρυνόμενους πόρους απαιτούν νέες εξελιγμένες προσεγγίσεις για παροχής υψηλού ποιότητα υπηρεσιών και ποιότητα εμπειρίας στους χρήστες. Αν και πολλές νέες αρχιτεκτονικές έχουν προκύψει τα τελευταία χρόνια, υπάρχουν ακόμα πολλές προκλήσεις που πρέπει να αντιμετωπιστούν για την παροχή απρόσκοπτων λειτουργιών και εκτέλεσης των πολύπλοκων αλγορίθμων που μπορεί να έχει ένα Κυβερνοφυσικό Σύστημα. Οι δύο κύριες προκλήσεις είναι οι στρατηγικές μεταφόρτωσης εργασιών και οι συνοδευτικοί μηχανισμοί διαχείρισης πόρων.

Λέξεις Κλειδιά: Μεταφόρτωση Υπολογιστικών Διεργασιών, Προγραμματισμός εργασιών, Δυναμική κατανομή πόρων, Διαχείριση πόρων, Θεωρία ελέγχου, Διαδίκτυο των πραγμάτων, Θεωρία Συστημάτων Αναμονής

Abstract

With the advent of fifth-generation (5G) wireless networks and the Internet of Things (IoT), a crucial part of enabling modern technologies is optimizing the performance of computational offloading and resource allocation mechanisms. The current paradigm in providing additional resources to devices is Multi-Accessed Edge Computing (MEC), which is characterized by the limited availability of resources. Hence, the devices along with the edge servers form a Cyber-Physical System (CPS), which must be carefully designed to fit each application's and scenario's needs. Thus, in this dissertation, we tackle the problem of realizing efficient offloading strategies that benefit the devices and on the other hand, the task scheduling and dynamic resource allocation for optimal utilization of the edge layer's resources, while achieving high-quality of service and experience. We considered designing holistic frameworks and co-designing these two fundamental aspects as they are tightly coupled. Specifically, an effort is made to address the following crucial research challenges: (i) optimizing the offloading decision taking into consideration device-specific aspects (e.g., signal strength, the position of devices), (ii) optimizing the resource management of resources by jointly designing task scheduling and dynamic resource allocation mechanism that guarantees certain performance criteria while the resources are optimally utilized, (iii) the performance modeling of dynamic workloads and application to assist in deciding the correct number of deployed resources for each application, (iv) the design of practical computation offloading strategies in the concept of approximate computing to alleviate the computational burden from the devices while finding balance between duration and quality of operations, and finally, (v) the design of closed-loop controllers taking also into account the current state of devices, the communication overhead and the available resource on the edge side.

To this end, we design a holistic architecture to optimize the offloading decision of users, i.e., users offloading their requests if the communication overhead is low. The offloaded requests may also be rejected by the edge server if the total amount exceeds the current capacity of the deployed resources utilizing performance modeling for the specific application. Then a central entity jointly solves the task scheduling and dynamic resource allocation problem keeping a certain QoS level for the application under a dynamic workload.

Then, in a similar setting, we enhance the task scheduling and dynamic resource allocation mechanism using modeling from queuing theory to dynamically adjust the number of deployed resources for a computing-intensive application. The framework allows for stability and performance guarantees for the resource management problem. This is achieved by optimally distributing the incoming tasks and the performance modeling of resources. More specifically, we utilized machine learning algorithms to map theoretically computed values for the processing rate of the deployed resources along with various monitoring KPIs.

Additionally, a novel switching offloading mechanism for robotic applications is proposed. In the context of Industry 4.0, applications rely on mobile robotic agents that execute many complex tasks that have strict safety and time requirements. Under this setting, the Edge Computing service delivery model allows the robotic agents to offload their computationally intensive tasks. In particular, practical offloading strategies for mobile robot path planning and localization tasks are designed. The offloading decision is based on the uncertainty of the robot's pose, the resource availability of the edge server, the quality of the network connection, and the difficulty of path planning.

Similarly, focusing on the stability guarantees and the convergence of the system, we introduce a set-based estimation offloading framework, for the specific case of the navigation of a unicycle robot towards a target position. The robot is subject to modeling and measurement uncertainties, and the estimation set is calculated using overapproximation techniques that alleviate additional computations. A switching set-based control mechanism provides accurate navigation and triggers more precise estimation algorithms when needed. To guarantee the convergence of the system and optimize the utilization of remote resources, a utility-based offloading mechanism is designed, which takes into account both the dynamic network conditions and the available computing resources at the network edge.

Finally, we study the case of real-time edge-assisted inference and batch scheduling. We find out that the quality of the Edge-assisted inference process and the overall latency of the system are competing metrics. So, we formulate a joint optimization problem to maximize the quality of inference while minimizing the overall latency for the GPU-enabled batch processing of inference applications. To deal with the computational complexity of the optimization problem, we first compute the optimal compression policies for the inference tasks to minimize the transmission time. By carefully examining the results of the compression problem, we identify that compressing the tasks in such a way to arrive simultaneously for remote processing significantly increases the performance of batch processing.

In summary, in this thesis, we study the problem of dynamic resource orchestration and management of IoT-based Cyber-Physical Systems. The immense advancement of modern applications along with the stringent requirements for ever-expanding resources requires new sophisticated approaches while providing high QoS and QoE to end-users. Although many new architectural concepts have arisen in the past few years, there are still many challenges to be addressed to provide seamless operations and execution of the complex algorithms a CPS may have. The two major challenges are the task offloading strategies and the accompanied resource management mechanisms.

Keywords: Computational Offloading, Task Scheduling, Dynamic Resource Allocation, Resource Management, Control Theory, Internet of Things, Approximate Computing, Queuing Theory

Abstract

Εκτεταμένη Περίληψη

Παρά τις αξιοσημείωτες βελτιώσεις του υλικού και των υπολογιστικών δυνατοτήτων των συσκευών τα τελευταία χρόνια, οι σύγχρονες εφαρμογές απαιτούν τεράστιους χώρους αποθήκευσης και δυνατότητες επεξεργασίας πολύπλοκων διεργασιών σε πραγματικό χρόνο. Στις περισσότερες περιπτώσεις, οι συσκευές του Διαδικτύου των Πραγμάτων (ΔτΠ) είναι συσκευές χαμηλής ενεργειακής κατανάλωσης (π.χ. αισθητήρες) με περιορισμένες υπολογιστικές δυνατότητες και ενεργειακούς πόρους (μπαταρία) και δεν μπορούν να εγγυηθούν την απαιτούμενη υψηλή απόδοση για τις εφαρμογές. Ακόμη και οι ισχυρές συσκευές (π.χ. έξυπνα κινητά, ρομπότ) δεν μπορούν να εγγυηθούν την απαιτούμενη υψηλή απόδοση ή/και την εκπλήρωση των χρονικών περιορισμών, για σύνθετες εφαρμογές με ανάγκες για γρήγορη απόκριση ή για μεγάλη ακρίβεια. Ως αποτέλεσμα, η μεταφόρτωση εφαρμογών με απαιτητικές υπολογιστικά διαεργασίες και με μεγάλες ενεργειακές ανάγκες, σε μια υπολογιστική υποδομή με παραπάνω δυνατότητες, για περαιτέρω επεξεργασία είναι ο βασικός παράγοντας για την υλοποίηση της νέας εποχής των Τεχνολογιών Πληροφορικής και Επικοινωνιών (ΤΠΕ). Αυτή η λύση, επιτρέπει τη βελτίωση της εμπειρίας του χρήστη παρέχοντας χαμηλότερο χρόνο απόκρισης για τις εφαρμογές, αυθημένη αξιοπιστία και βελτιωμένη ενεργειακή απόδοση για τις συσκευές που τροφοδοτούνται με μπαταρία. Σε ένα τυπικό σενάριο μεταφόρτωσης, τα δεδομένα μεταδίδονται μέσω ασύρματων συνδέσεων, (π.χ., κινητής τηλεφωνίας ή ασύρματων δικτύων) και η ποιότητα της ασύρματης σύνδεσης εξαρτάται σε μεγάλο βαθμό από την ισχύ του σήματος, τις παρεμβολές, την εγκατάλειψη πακέτων και άλλες παραμέτρους που σχετίζονται με το ασύρματο περιβάλλον, οι οποίες πρέπει να λαμβάνονται υπόψη την απόφαση μεταφόρτωσης. Έτσι, είναι σημαντικό η υποδομή η οποία είναι αφιερωμένη στο να δέχεται διεργασίες από τις υπολογιστικά περιορισμένες συσκευές να βρίσκεται σε κοντινή απόσταση στο δίκτυο με αυτές.

Προκλήσεις και κίνητρα:
Η μεταφορά της αποκεντρωμένης υπολογιστικής υποδομής στην άκρη του δικτύου φέρνει διάφορα πλεονεκτήματα (π.χ. επικοινωνία χαμηλής καθυστέρησης, επεκτασιμότητα), αλλά φέρνει επίσης νέες προκλήσεις. Επιπλέον, η βέλτιστη υποστήριξη των Κυβερνο-Φυσικών Συστημάτων (ΚΦΣ) εξαρτάται σε μεγάλο βαθμό από την έγκαιρη παροχή του σωστού τμήματος δεδομένων στη σωστή υπολογιστική οντότητα. Αυτό εισάγει νέες προκλήσεις έρευνας και σχεδιασμού σχετικά με την επεξεργασία δεδομένων, τις αποφάσεις μεταφόρτωσης, της αρχιτεκτονικής, και την δυναμική κατανομή πόρων. Προφανώς, υπάρχει ανάγκη να διερευνηθούν οι βασικές απαιτήσεις και οι πιθανές ευκαιρίες για να καταστεί δυνατή η υποστήριξη του ΚΦΣ στο όραμα της Υπολογιστικής στα Άκρα του Δικτύου. Η σημαντική πρόοδος στις πρόσφατες τεχνολογικές τάσεις και το ενδιαφέρον για έρευνα αιχμής τόσο από τον ακαδημαϊκό χώρο όσο και από τη βιομηχανία, παρακινούν τους ερευνητές να προ-

σεγγίσουν και να προσπαθήσουν να λύσουν ενδιαφέροντα και προκλητικά προβλήματα. Στη διατριβή μας, θέλαμε να εκμεταλλευτούμε αυτή την ευκαιρία και να προτείνουμε λύσεις σε ορισμένα σύγχρονα και ενδιαφέροντα προβλήματα λαμβάνοντας υπόψη τις ακόλουθες ερευνητικές προκλήσεις:

- **Αποφάσεις μεταφόρτωσης που αφορούν τις συσκευές:** Τα δίκτυα ΔτΠ χαρακτηρίζονται κυρίως ως ταχέως μεταβαλλόμενα δίκτυα πρόσβασης. Επομένως, οι συνθήκες δυναμικού δικτύου είναι ένα πολύ δύσκολο πρόβλημα στο πλαίσιο της μεταφόρτωσης διεργασιών. Απαιτείται ένας μηχανισμός λήψης απόφασης για το πρόβλημα του προγραμματισμού των εργασιών και συγκεκριμένα για το εάν η εργασία πρέπει να εκτελεστεί τοπικά (στη συσκευή) ή να μεταφορτωθεί σε κάποια απομακρυσμένη υποδομή. Μια εσφαλμένη απόφαση μεταφόρτωσης μπορεί να οδηγήσει σε χειροτέρευση της απόδοσης της εφαρμογής. Βελτιστοποιώντας τη συνολική απόδοση, η μεταφόρτωση είναι επωφελής μόνο εάν είναι διαθέσιμη μια ασύρματη συνδεσιμότητα η οποία είναι αξιόπιστη και με χαμηλή καθυστέρηση. Εκτός από το δίκτυο, η επιτυχία της μεταφόρτωσης και κατά συνέπεια η απόδοση των εφαρμογών εξαρτάται από πολλές άλλες παραμέτρους που αφορούν την συσκευή, π.χ. την κινητικότητα του χρήστη και την υπολειπόμενη ενέργεια της συσκευής. Η απόφαση προγραμματισμού των εργασιών σχετίζεται επίσης με την καθυστέρηση εκτέλεσης εργασιών, την καθυστέρηση μετάδοσης και την κατανάλωση ενέργειας της συσκευής. Ως αποτέλεσμα, η απόφαση για την μεταφόρτωση ενός αιτήματος εφαρμογής ή όχι είναι η πρώτη πρόκληση που πρέπει να αντιμετωπιστεί.
- **Βελτιστοποίηση διαχείρισης πόρων**: Αυτό αποτελεί το κύριο κίνητρο πίσω από αυτή τη διατριβή, καθώς η διαχείριση των πόρων είναι η βασική πρόκληση που πρέπει να αντιμετωπιστεί στο πρότυπο της Υπολογιστικής στα Άκρα του Δικτύου, όπου, φυσικά, οι πόροι δεν είναι άφθονοι. Έτσι, πρέπει κανείς να αντιστοιχίσει κατάλληλα τον φόρτο εργασίας των εφαρμογών στις κατάλληλες υποδομές με τους αντίστοιχους πόρους με βάση τις απαιτήσεις και την διαθεσιμότητα. Στην εποχή των τεχνικών εικονικοποίησης, είναι πολυ διαδεδομένο για τις εφαρμογές του ΔτΠ να εκτελούνται σε εικονικές μηχανές (VM) ή κοντέινερ στους εξυπηρετητές στα Άκρα του Δικτύου (ΑτΔ). Δύο κύριες αποφάσεις προκύπτουν από τους παρόχους υποδομής που επηρεάζουν άμεσα τον χρόνο απόκρισης των εφαρμογών, και συγκεκριμένα: (i) τους υπολογιστικούς πόρους που θα εκχωρηθούν στις εικονικοποιημένες οντότητες και (ii) τον αριθμό των αντίγραφων (ρέπλικων) για κάθε εικονικοποιημένη οντότητα. Επιπλέον, καθώς πολλές εφαρμογές ή/και συσκευές ενδέχεται να αναζητήσουν πρόσθετους υπολογιστικούς πόρους στον ίδιο εξυπηρετητη, είναι πολύ σημαντική η αποτελεσματική και δυναμική εκχώρηση των πόρων των φιλοξενούμενων εφαρμογών. Η ταυτόχρονη κάλυψη των απαιτήσεων Ποιότητας της Υπηρεσίας (ΠτΥ) και των χρηστών και των παρόχων των υποδομών, είναι μια μεγάλη πρόκληση, κυρίως λόγω ενός συνόλου ανταγωνιστικών Βασικών Δεικτών Απόδοσης (ΒΔΑ). Για παράδειγμα,

μια στρατηγική διαχείρισης πόρων πρέπει να εγγυάται τις απαιτήσεις απόδοσης των εφαρμογών, ενώ ταυτόχρονα να ελαχιστοποιεί το κόστος των αναπτυσσόμενων πόρων. Επιπλέον, είναι σύνηθες οι σύγχρονες εφαρμογές να είναι υπολογιστικά απαιτητικές και να απαιτούν σημαντικό αριθμό πόρων για την εύριθμή λειτουργία τους. Ως εκ τούτου, οι πάροχοι των υποδομών συνήθως χρησιμοποιούν προληπτικές ή και αντιδραστικές λύσεις για να αντιμετωπίσουν τη δυναμική φύση του φόρτου εργασίας και την προσαρμογή των παρεχόμενων πόρων, ενώ παρακολουθούν διάφορους ΒΔΑ. Από την άλλη πλευρά, η αποτελεσματική αξιοποίηση των διαθέσιμων πόρων είναι μια σημαντική πρόκληση. Είναι ζωτικής σημασίας να βελτιστοποιηθεί η χρήση των πόρων για να επιτευχθεί καλύτερη απόδοση για όλες τις αναπτυγμένες εφαρμογές. Η βελτιστοποίηση της χρήσης πόρων είναι δύσκολη, καθώς πρέπει να ληφθούν υπόψη οι διαφορετικές απαιτήσεις των εφαρμογών, οι κυμαινόμενες απαιτήσεις φόρτου εργασίας και οι πεπερασμένοι πόροι της υποδομής. Σε όλη τη βιβλιογραφία, οι ερευνητές χρησιμοποιούν μοντέλα από τη θεωρίατων συστημάτων αναμονής για τη μοντελοποίηση φορητών συσκευών και εξυπηρετητών ΑτΔ, μαζί με τεχνικές βελτιστοποίησης για να αποφασίσουν τη βέλτιστη πολιτική μεταφόρτωσης. Δυστυχώς, σε ορισμένες περιπτώσεις, υπάρχει ένα σημαντικό μειονέκτημα που μπορεί να οδηγήσει σε γενική επιδείνωση της απόδοσης: η στατική μοντελοποίηση των πόρων των εξυπηρετητών ΑτΔ που οδηγεί σε αχρείαστη παροχή ή υποπαροχή πόρων και σε μειωμένη χρησιμοποίηση ή αυξημένη χρησιμοποιήση αντίστοιχα. Τέλος, ακολουθώντας τη διαδικασία παροχής πόρων, υιοθετείται ένας αλγόριθμος προγραμματισμού (ή εξισορρόπηση φορτίου) για την επεξεργασία και τη βέλτιστη αντιστοίχιση των εισερχόμενων εργασιών/εφαρμογών (φόρτος εργασίας) στους εικονικοποιημένους πόρους. Για παράδειγμα, εάν μια εικονική μηχανή χρησιμοποιείται υπερβολικά, τότε θα πρέπει να αποκλειστεί προσωρινά από τη δεξαμενή πόρων που αφιερώνονται στην εκτέλεση των εργασιών των εφαρμογών. Ωστόσο, ο προγραμματισμός εργασιών περιλαμβάνει επίσης διάφορες προκλήσεις όπως η ετερογένεια των αιτημάτων και των πόρων και η αβεβαιότητα του αριθμού των εισερχόμενων αιτημάτων, τα οποία δεν μπορούν να επιλυθούν με τη χρήση παραδοσιακών προσεγγίσεων διαχείρισης πόρων. Ως εκ τούτου, θα πρέπει να δοθεί μεγάλη έμφαση και προτεραιότητα σε αυτά για να γίνουν οι υπηρεσίες πιο αξιόπιστες και με μεγαλύτερη απόδοση. Με άλλα λόγια, η εξυπηρέτηση του φόρτου εργασίας θα πρέπει να καταλαμβάνει την ελάχιστη ποσότητα πόρων που θα χρησιμοποιηθούν βέλτιστα για τη διατήρηση ενός επιθυμητού επιπέδου ΠτΥ.

- **Μοντελοποίηση της απόδοσης εφαρμογών με δυναμικό φόρτο αιτημάτων:** Όπως αναφέρθηκε προηγουμένως, η ενασχόληση με τη δυναμική φύση του φόρτου εργασίας των εφαρμογών είναι το κλειδί για την επίτευξη υψηλών επιδόσεων του συστήματος. Στη βιβλιογραφία, τα περισσότερα από τα υπάρχοντα μοντέλα για την αποτύπωση των απαιτήσεων σε πόρους εφαρμογών ΔτΠ είναι εμπειρικά και συνήθως ασχολούνται με μια συγκεκριμένη μέτρηση απόδοσης, όπως ο χρόνος απόκρισης και η κατανάλωση ενέργειας. Η αντιμετώπιση του δυναμικού φόρτου εργασίας, των ποικίλων απαιτήσεων των διαφορετικών εφαρμογών ΔτΠ και της δυσκολίας μοντελοποίησης της συμπεριφοράς και της κατάστασης των πόρων, είναι πολύ σημαντικές περιοχές έρευνας. Ως αποτέλεσμα, κάποιος πρέπει να σχεδιάσει μοντέλα απόδοσης πολλαπλών εισόδων πολλαπλών εξόδων που μπορούν να εφαρμοστούν εύκολα σε ένα μεγάλο σύνολο εφαρμογών για να αποτυπωθεί μια τέτοια δυναμική. Με αυτόν τον τρόπο μπορούν να δημιουργηθούν νέοι ελεγκτές για τη ρύθμιση των προαναφερθέντων ΒΔΑ. Για την επίτευξη του χαρακτηρισμού του φόρτου εργασίας και την πρόβλεψη απόδοσης, το πρώτο βήμα είναι η προσπάθεια πρόβληψης του αριθμού των εισερχόμενων αιτημάτων. Αυτή η πρόβλεψη μπορεί να συνδυαστεί με έναν μηχανισμό κατανομής πόρων, καθώς η ποσότητα των πόρων που απαιτούνται για την εκτέλεση της εργασίας είναι ευθέως ανάλογη με την ποσότητα της κίνησης της εφαρμογής. Σε αυτόν τον βαθμό, ένας βασικός παράγοντας για την κλιμάκωση των πόρων σύμφωνα με την κίνηση του φόρτου εργασίας είναι ο εντοπισμός του μέγιστου αριθμού αιτημάτων που μπορούν να επεξεργαστούν οι παρεχόμενοι πόροι για ένα συγκεκριμένο χρονικό διάστημα.

- **Σχεδιασμός πρακτικών στρατηγικών μεταφόρτωσης διεργασιών βάση προσεγγιστικής υπολογιστικής:** Τα τελευταία χρόνια αναδύονται πολύπλοκες εφαρμογές, που στοχεύουν μεταξύ άλλων να την Τεχνητή Νουμοσύνη σε συσκευές ΔτΠ, φορητές συσκευές και ρομπότ. Ειδικά, οι εφαρμογές της Βιομηχανίας 4.0, οι οποίες βασίζονται ειδικά σε ρομπότ που εκτελούν πολλές σύνθετες εργασίες που έχουν αυστηρές απαιτήσεις ασφάλειας και χρόνου, π.χ. αλγόριθμους που σχετίζονται με την επεξεργασία εικόνας, τον προγραμματισμό μονοματιού για την κίνηση, τον εντοπισμό θέσης, τη χαρτογράφηση του χώρου και την αυτόνομη εκμάθηση για όλα τα παραπάνω. Αυτοί οι αλγόριθμοι είναι υπολογιστικά απαιτητικοί και για τους επεξεργαστές αλλά και για τη μνήμη και οι πόροι των ρομπότ και των συσκευών ΔτΠ δεν επαρκούν πάντα για τη δημιουργία αποτελεσματικών, ασφαλών και αυτόνομων λειτουργιών. Κατά συνέπεια, προκύπτουν νέα προβλήματα και πρέπει κανείς να αποφασίσει προσεκτικά τις στρατηγικές ανάπτυξης μεταφόρτωσης διεργασιών, π.χ. πού να τοποθετήσετε φόρτο εργασίας, τις πολιτικές σύνδεσης, πότε να χρησιμοποιήσετε τους εξυπηρετητές ΑτΔ, η ετερογένεια, η ακρίβεια πολύπλοκων αλγορίθμων, πώς να εξοικονομήσετε χρόνο απόκρισης με αποδεκτή απώλεια στην ακρίβεια ενός σύνθετου αλγορίθμου. Επιπλέον, όσο και αν φαίνεται κατάλληλο, η χρήση αποκλειστικά απομακρυσμένων υπολογιστικών πόρων μπορεί να μην είναι αρκετή. Ένας αριθμός ανεπιθύμητων φαινομένων δυνητικά λαμβάνουν χώρα κατά τη μετάδοση και την επεξεργασία των πληροφοριών, όπως η καθυστέρηση δικτύου, η μεταβλητή ΠτΥ και ο χρόνος διακοπής λειτουργίας λόγω διάφορων φαινομένων. Για αυτούς τους λόγους, για παράδειγμα, οι αυτόνομες κινητές συσκευές (π.χ. ρομπότ, μη επανδρωμένα οχήματα) έχουν συχνά κάποια ικανότητα για τοπική επεξεργασία όταν στοχεύουν σε αποκρίσεις χαμηλής καθυστέρησης και σε περιόδους όπου η πρόσβαση στο δίκτυο δεν είναι διαθέσιμη ή αναξιόπιστη. Κατά συνέπεια, μια σημαντική πρόκληση, από την άποψη του σχεδιασμού, της εκτίμησης και της βελτιστοποίησης του δικτύου είναι ο συνδυασμός τοπικών γρήγορων αλλά πιθανώς περιορισμένης ακρίβειας αλγορίθμων και απομακρυσμένων, ακριβών αλλά υπολογιστικά απαιτητικών αλγορίθμων με αποτελεσματικό τρόπο.

Συνεισφορές της Διατριβής

Αυτή η διατριβή προσπαθεί να αντιμετωπίσει μερικά από τα προαναφερθέντα προβλήματα που προκύπτουν στην Υπολογιστική στα άκρα του Δικτύου και την υπολογιστική μεταφόρτωση των διεργασιών των ΚΦΣ. Εστιάσαμε στη βελτιστοποίηση της απόδοσης των ΚΦΣ με τη βέλτιστη χρήση των πολιτικών μεταφόρτωσης και τον προσεκτικό σχεδιασμό στρατηγικών διαχείρισης πόρων. Οι συνεισφορές στα παραπάνω θέματα μπορούν να συνοψιστούν στα ακόλουθα:

- **Μοντελοποίηση της δυναμικής απόδοσης:** Οι μεθοδολογίες για την αντιμετώπιση του προβλήματος της δυναμικής διαχείρισης πόρων απαιτούν εξελιγμένες αφαιρέσεις μοντελοποίησης της απόδοσης για να αποτυπώσουν τις διάφορες δυναμικές των απαιτήσεων της εφαρμογής, την κατάσταση των πόρων και τις απαιτήσεις των χρηστών. Μια σημαντική καινοτομία αυτής της διατριβής είναι η εισαγωγή του όρου προφίλ πόρων για την επίτευξη όλων των παραπάνω. Τα προφίλ πόρων εκφράζουν τη σχέση μεταξύ των κατανεμημένων πόρων και του μέγιστου αριθμού των εισερχόμενων αιτημάτων μιας συγκεκριμένης εφαρμογής που μπορούν να εξυπηρετήσουν οι πόροι διατηρώντας ένα συγκεκριμένο επίπεδο ΠτΥ. Το πλεονέκτημα του σχεδιασμού προφίλ πόρων είναι τριπλό. Πρώτον, ένα προφίλ πόρων είναι στην πραγματικότητα ένα εφικτό σημείο ισορροπίας και διευκολύνει τη δυναμική διαδικασία κατανομής πόρων παρέχοντας γνώση σχετικά με τον αριθμό των διαφορετικών καταστάσεων (π.χ. με την έννοια του αριθμού των πυρήνων του επεξεργαστή) που απαιτούνται για τους εικονικοποιημένους πόρους. Δεύτερον, ο συνδυασμός αυτών των προφίλ πόρων με κατάλληλους ελεγκτές επιτρέπει τον σχεδιασμό αλγορίθμων για εφαρμογές κρίσιμες για το χρόνο ή την ακρίβεια. Τρίτον, τα προφίλ πόρων βελτιστοποιούν την ομαλή λειτουργεία των πόρων που χρησιμοποιεί μία εφαρμογή μειώνοντας τις περιπτώσεις που οι πόροι, είτε δεν χρησιμοποιούνται επαρκώς, είτε δεν αρκούν για να πετύχουν την αναζητούμενη απόδοση της εφαρμογής. Σε αυτή την διατριβή, πειραματιστήκαμε με δύο διαφορετικές μεθοδολογίες μοντελοποίησης για την εξαγωγή των προφίλ πόρων. Από τη μία πλευρά, χρησιμοποιήσαμε τη Θεωρία Συστημάτων, η οποία δίνει την δυνατότητα να περιλαμβάνει διάφορες μετρήσεις απόδοσης (δηλαδή μεταβλητές κατάστασης) και τους πόρους ως παραμέτρους ελέγχου (μεταβλητές εισόδου) και να περιγράφει τη σχέση τους υπό διάφορες συνθήκες λειτουργίας και εγγυήσεις ΠτΥ. Από την άλλη πλευρά, χρησιμοποιήσαμε απλούς αλγόριθμους μηχανικής μάθησης, που αξιοποιούν πληροφορίες από (i) τον προγραμματισμό της κατανομής των πόρων, (ii) τους υπο παρακολούθηση ΒΔΑ και (iii) τον αλγόριθμο πρόβλεψης φόρτου εργασίας για την εκτίμηση του βασικού αριθμού αντιγράφων για το δυναμικό αριθμό αιτημάτων μια εφαρμογής. Ο βασικός παράγοντας και για τις δύο προσεγγίσεις είναι η μοντελοποίηση του φόρτου εργασίας, δηλαδή η εκτίμηση του αριθμού των εισερχόμενων αιτημάτων για την εφαρμογή αξιοποιώντας μεθόδους πρόβλεψης χρονοσειρών.

- **Κοινή διαχείριση πόρων και προγραμματισμός εργασιών στην υποδομή:** Η μεγιστοποίηση της απόδοσης των σύγχρονων εφαρμογών απαιτεί έγκαιρη ενορχήστρωση των εικονικών πόρων στην υποδομή στα άκρα του δικτύου. Ωστόσο, η προληπτική ανάπτυξη πόρων για την ικανοποίηση συγκεκριμένων απαιτήσεων των εφαρμογών κάτω από δυναμικό φόρτο εργασίας είναι εξαιρετικά δύσκολο πρόβλημα. Για το σκοπό αυτό, μια άλλη σημαντική καινοτομία αυτής της διατριβής είναι ότι τα θεμελιώδη προβλήματα του προγραμματισμού των εργασιών και της αυτόματης κλιμάκωσης των πόρων αντιμετωπίζονται από κοινού. Κατά συνέπεια, μαζί με την απόφαση μεταφόρτωσης, προτείνεται ένας δυναμικός μηχανισμός κατανομής πόρων και ελέγχου αποδοχής. Αυτός ο μηχανισμός είναι υπεύθυνος για τη διανομή των μεταφορτωμένων αιτημάτων μεταξύ των αντιγράφων της εφαρμογής στους εικονοποιημένους πόρους, παράλληλα με τον έλεγχο αποδοχής (δηλ. αποδοχή ή απόρριψη των μεταφορτωμένων αιτημάτων ανάλογα με τη διαθεσιμότητα πόρων) και τη δυναμική κατανομή πόρων για κάθε εφαρμογή (δηλαδή, αποφασίζοντας τον αριθμό των αντιγράφων για κάθε εφαρμογή και προφίλ πόρων). Επίσης, ένας προβλεπτής του φόρτου εργασίας και η δυναμική μοντελοποίηση των πόρων και της συνολικής κατάστασης του δικτύου/εξυπηρετητών παρέχουν τη βάση πάνω στην οποία λειτουργεί ο αλγόριθμος. Ο προτεινόμενος αλγόριθμος έχει σχεδιαστεί ώστε να είναι πρακτικός και να εγγυάται προδιαγραφές, όπως προσαρμοστικότητα σε γρήγορες αλλαγές στον φόρτο εργασίας και στη διαθεσιμότητα των πόρων. Αυτός ο πρώτος μηχανισμός στη συνέχεια εμπλουτίζεται με τις εγγυήσεις σταθερότητας που προέρχονται από τη θεωρία ουρών για την κατασκευή ενός αποκεντρωμένου μηχανισμού για το πρόβλημα του προγραμματισμού εργασιών και κατανομής πόρων, προσπαθώντας να φτιάξουμε μια ολιστική και κλιμακούμενη αρχιτεκτονική.

- **Ένας διακοπτικός μηχανισμός μεταφόρτωσης των εργασιών:** Στα επόμενα κεφάλαια της διατριβής ασχολούμαστε με τη σχεδίαση βέλτιστων πολιτικών μεταφόρτωσης για ΚΦΣ. Αρχικά, δίνεται έμφαση στην καλή ποιότητα του ασύρματου δικτύου επικοινωνίας για χρήστες κινητών. Σχεδιάζουμε ένα μηχανισμό δύο βημάτων για την υποστήριξη και την υλοποίηση της απόφασης μεταφόρτωσης, λαμβάνοντας υπόψη τόσο τις πληροφορίες που αφορούν τις συσκευές όσο και τη διαθεσιμότητα των πόρων της υποδομής στα Άκρα του Δικτύου. Πιο συγκεκριμένα, οι χρήστες μεταφορτώνουν τα αιτήματά τους, λαμβάνοντας υπόψη τη θέση τους και την ισχύ του σήματος για να ελαχιστοποιηθεί η χρονική επιβάρυνση της επικοινωνίας. Επίσης, η υποδομή στα Άκρα του Δικτύου μπορεί να απορρίψει ένα αίτημα (δηλαδή, ο χρήστης πρέπει να το εκτελέσει τοπικά) εάν οι διαθέσιμοι πόροι δεν επαρκούν για να διατηρήσουν ένα συγκεκριμένο επίπεδο ΠτΥ. Σε συνέχεια των επόμενων κεφαλαίων της διατριβής, υλοποιείται ένας παρόμοιος διακοπτικός μηχανισμός μεταφόρτωσης για να βοηθήσει στη βελτιστοποίηση της μεταφόρτωσης αλγορίθμων απαιτητικών υπολογιστικά με έμφαση σε σενάρια ρομποτικής. Στόχος είναι να μετριάσουμε την υπολογιστική πίεση που απαιτεί ο υπολογοσμός τέτοιων αλγορίθμων στο ρομπότ, το οποίο έχει συνήθως περιορισμένους πόρους. Υλοποιούμε λοιπόν, διαφορετικές εκδόσεις των αλγορίθμων ακολουθώντας την ιδέα της προσεγγιστικής υπολογογιστικής, δηλαδή, θεωρούμε (i) μια έκδοση που εκτελείται τοπικά (στη συσκευή), και η οποία είναι γρήγορη αλλά δεν είναι επαρκώς ακριβής και (ii) μία έκδοση που εκτελείται απομακρυσμένα (στην υποδομή στα Άκρα του Δικτύου), η οποία είναι ακριβής μεν αλλά ακριβή από άποψη πόρων ή/και χρόνου. Σε αυτό το πλαίσιο, υιοθετούμε έναν διακοπτικό μηχανισμό μεταφόρτωσης που λαμβάνουν υπόψη την αβεβαιότητα των τοπικών αλγορίθμων. Έτσι είμαστε σε θέση να δημιουργήσουμε ένα μηχανισμό που βοηθά ένα κινητό ρομπότ σε μια απλή αποστολή πλοήγησης, δηλαδή να φτάσει σε ένα τελικό σημείο από μια αρχική θέση. Η απομακρυσμένη εκτέλεση αυτών των αλγορίθμων εκτελείται μόνο σε περίπτωση που η μεταφόρτωση είναι επωφελής, δηλαδή όταν η επιβάρυνση της επικοινωνίας είναι χαμηλή και οι διαθέσιμοι πόροι της υποδομής επαρκούν για την εκτέλεση των εργασιών. Επιπλέον, λαμβάνουμε υπόψιν την ανάγκη για ακριβέστερη πλοήγηση του κινητού ρομπότ, την εγγύτητα σε εμπόδια και τη δυσκολία της πλοήγησης, ώστε να διασφαλίζεται η ασφαλής πλοήγηση. Κατά συνέπεια, αυτός ο μηχανισμός διευκολύνει τη βέλτιστη μεταφόρτωση για ΚΦΣ.

- **Πλαίσιο εκτίμησης και ελέγχου για ρομποτικά ΚΦΣ:** Σεβόμενοι την ανάγκη για εγγυήσεις σταθερότητας και την ανάγκη για ελεγκτές κλειστού βρόχου για τα ΚΦΣ, επεκτείνουμε τη μοντελοποίηση για την πλοήγηση ενός κινητού ρομπότ. Σχεδιάζουμε ένα νέο πλαίσιο εκτίμησης της θέσης του ρομπότ καθώς και ελέγχου της κίνησης, το οποίο αντιμετωπίζει από κοινού το πρόβλημα της σύγκλισης στην επιθυμητή τροχιά πλοήγησης με την αποτελεσματική χρήσης των πόρων επικοινωνίας και διαθέσιμων πόρων στην απομακρυσμένη υποδομή. Χρησιμοποιώντας μια συνάρτηση που ενσωματώνει τη δυναμική του ρομποτ, της επικοινωνίας και τους υπολογιστικούς πόρους, κατασκευάζουμε ένα πλαίσιο για να χειριστούμε δύο θεμελιώδη προβλήματα: α.) τη σύνθεση ελεγκτών που υπαγορεύουν την κίνηση του ρομπότ σε αυτό το πρόβλημα πλοήγησης και β.) μια στρατηγική μεταφόρτωσης για την αντιστάθμιση της αβεβαιότητας των τοπικών τεχνικών εκτίμησης θέσης με τις πιο ακριβείς απομακρυσμένες. Αυτός το πλαίσιο λειτουργίας βρίσκει την ισορροπία μεταξύ της ακρίβειας πλοήγησης και της διάρκειας της συγκεκριμένης αποστολής. Το πιο σημαντικό είναι ότι το προτεινόμενο πλαίσιο εγγυάται τη σύγκλιση με τη θέση στόχο, ανεξάρτητα από τις διάφορες παραμέτρους που επιλέγονται, σε αντίθεση με τα συστήματα περιοδικής μεταφόρτωσης. Η όλη προσέγγιση προσαρμόζεται εύκολα στις ανάγκες των διαφορετικών χαρακτηριστικών κάθε πιθανού σεναρίου αποστολής.

- **Αξιολόγηση προτεινόμενων αλγορίθμων με γνωστές Αρχιτεκτονικές Ενορχήστρωσης και Πρότυπα:** Αξιολόγηση προτεινόμενων πλαισίων με Σχετικά με Αρχιτεκτονικές και Πρότυπα: Παρουσιάζονται εκτεταμένα αριθμητικά αποτελέσματα, τα οποία προκύπτουν μέσω πραγματικών πειραματισμών ή προσομοιώσεων προκειμένου να αποτυπωθεί η αποτελεσματικότητα και η αποδοτικότητα των προτεινόμενων πλαισίων. Ειδικά, για κάθε πραγματικό πειραματισμό που θα παρουσιαστεί σε αυτή τη διατριβή, επιλέξαμε να ακολουθήσουμε τις ευρέως υιοθετημένες λύσεις, δηλαδή τα πρότυπα του Ευρωπαϊκού Ινστιτούτου Τηλεπικοινωνιακών Προτύπων (ETSI) και να τα εφαρμόσουμε με εμπορικά ή ανοιχτού κώδικα εργαλεία ενορχήστρωσης πόρων, που επιτρέπουν επεκτασιμότητα, διαλειτουργικότητα και διαφανής ανάπτυξη των εφαρμογών σε ετερογενείς τεχνολογίες υλικού και λογισμικού, π.χ. ευρέως χρησιμοποιούμενα εργαλεία λογισμικού όπως OpenStack, Kubernetes.

Κεφάλαιο 2

Στο Κεφάλαιο 2 παρουσιάζονται οι ορισμοί των βασικών στοιχείων από τους τομείς της Θεωρίας Συστημάτων Αυτόματου Ελέγχου και της Θεωρίας Συστημάτων Αναμονής. Η γνώση του βασικού μαθηματικού υποστρώματος που δίνεται περιεκτικά στο κεφάλαιο αυτό, είναι κρίσιμη για την κατανόηση των προβλημάτων αλλά και των αλγορίθμων που παρουσιάζονται στην συνέχεια στην προτεινόμενη διατριβή.

Κεφάλαιο 3

Στο κεφάλαιο 3, προτείνεται μια αρχιτεκτονική Υπολογιστικής Παρυφών (ΥΠ) δύο επιπέδων για να προσφέρει υπολογιστικούς πόρους για την απομακρυσμένη εκτέλεση μιάς εφαρμογής με βάση τη θέση του χρήστη. Στο επίπεδο της συσκευής, εκτελείται μια αρχική απόφαση μεταφόρτωσης λαμβάνοντας υπόψη την εκτιμώμενη θέση και την ποιότητα της ασύρματης σύνδεσης κάθε χρήστη. Στο επίπεδο της απομακρυσμένης υποδομής, προτείνεται ένας μηχανισμός δημιουργίας προφίλ πόρων που αντιστοιχίζει τον εισερχόμενο φόρτο εργασίας σε υπολογιστικούς πόρους υπό συγκεκριμένες απαιτήσεις απόδοσης της εφαρμογής. Αντιμετωπίζοντας το δυναμικό φόρτο εργασίας, ένας μηχανισμός κλιμάκωσης λαμβάνει ταυτόχρονα την απόφαση μεταφόρτωσης και κατανέμει μόνο τους απαραίτητους πόρους με βάση τα προφίλ πόρων και την εκτίμηση μιας τεχνικής πρόβλεψης φόρτου εργασίας. Για την αξιολόγηση της προτεινόμενης αρχιτεκτονικής, υλοποιήθηκε ένα έξυπνο σενάριο τουριστικής εφαρμογής σε μια πραγματική μεγάλης κλίμακας υποδομής 5G. Η προτεινόμενη αρχιτεκτονική έχει σχεδιαστεί ιδιαίτερα για να επιταχύνει την εκτέλεση μιας υπηρεσίας αναγνώρισης αντικειμένων για χρήστες κινητών. Στο συγκεκριμένο σενάριο έξυπνης πόλης που εξετάζεται σε αυτήν την εργασία, οι επισκέπτες μιας πολυσύχναστης τουριστικής περιοχής χρησιμοποιούν τις συσκευές τους που είναι εξοπλισμένες με κάμερα για να τραβήξουν στιγμιότυπα ή βίντεο μικρής διάρκειας ενός Σημείου Ενδιαφέροντος (*π.χ.*, εκθέματα ένα μουσείο) προκειμένου να λάβουν πρόσθετες πληροφορίες σχετικά με αυτά. Δεδομένου ότι η αναγνώριση εικόνας είναι μια εργασία απαιτητική υπολογιστικά και ενεργοβόρα, μια υποδομή ΥΠ προσφέρει τους επιπλέον υπολογιστικούς πόρους για την κάλυψη των αυστηρών απαιτήσεων χρόνου. Επιπλέον, υλοποιείται και αξιολογείται μια τεχνική εντοπισμού θέσης των χρηστών. Υποθέτουμε ότι οι χρήστες δραστηριοποιούνται σε μια πολυσύχναστη τουριστική περιοχή. Ως εκ τούτου, το κοινό εύρος ζώνης και κυρίως η παρεμβολή των σημάτων παίζει σημαντικό ρόλο στην επιβάρυνση της μετάδοσης των μεταφορτωμένων αιτημάτων. Ως αποτέλεσμα, για να αντιμετωπιστεί αυτή η δυναμική συμπεριφορά, η απόφαση μεταφόρτωσης ενός αιτήματος χρηστών λαμβάνει υπόψη τη θέση των χρηστών και επιπλέον την διαθεσιμότητα πόρων των εξυπηρετητών ΥΠ. Οι κύριες συνεισφορές αυτού του κεφαλαίου συνοψίζονται ως εξής:

- Ένας μηχανισμός δημιουργίας προφίλ πόρων αντιστοιχίζει τις απαιτήσεις απόδοσης της εφαρμογής σε υπολογιστικούς πόρους στην πλευρά της υποδοπής ΥΠ. Αυτός ο μηχανισμός επιτρέπει τον σχεδιασμό και την κατανομή των πόρων της ΥΠ προκειμένου να ανταποκριθεί στη μεταβαλλόμενη ζήτηση μεταφόρτωσης. Το προφίλ πόρων βασίζεται σε γραμμικά μοντέλα από τη Θεωρία Συστημάτων, τα οποία αποτυπώνουν αποτελεσματικά τη δυναμική του συστήματος.
- Επινοήθηκε ένας μηχανισμός δύο σταδίων για την υποστήριξη και την υλοποίηση της απόφασης μεταφόρτωσης, λαμβάνοντας υπόψη τόσο τις πληροφορίες που αφορούν τους χρήστες όσο και τη διαθεσιμότητα των πόρων της υποδομής ΥΠ. Προς αυτή την κατεύθυνση, η απόφαση μεταφόρτωσης γίνεται βασικός παράγοντας για τη διατήρηση των αποτελεσμάτων υψηλών επιδόσεων, προς όφελος τόσο του παρόχου υποδομής όσο και των χρηστών.
- Παρουσιάζεται μια αρχιτεκτονική ΥΠ που υποστηρίζει την ανάπτυξη, την ενορχήστρωση και τη διαχείριση της εφαρμογής με βάση τη θέση του χρήστη. Ακολουθώντας τις βασικές αρχές των οδηγιών για 5G δίκτυα, παρουσιάζεται μια αρχιτεκτονική, χρησιμοποιώντας εργαλεία λογισμικού τελευταίας τεχνολογίας για την ενορχήστρωση και τη διαχείριση της υποδομής. Πραγματοποιείται αξιολόγηση για να επαληθευτεί η εγκυρότητα και η εφαρμοσιμότητα της προτεινόμενης μεθοδολογίας. Επίσης, ο προτεινόμενος μηχανισμός δημιουργίας προφίλ πόρων και η προσέγγιση εκτίμησης του φόρτου εργασίας συγκρίνονται με καθιερωμένες αντίστοιχες μεθοδολογίες στη βιβλιογραφία.

Με βάση τα αναλυτικά αποτελέσματα της πειραματικής αξιολόγησης, η μέθοδος εκτίμησης τοποθεσίας αποδείχθηκε πολύ αξιόπιστη για το έξυπνο τουριστικό σενάριο που εξετάζεται. Με αυτήν την εκτίμηση, η απόφαση έξυπνης μεταφόρτωσης δύο βημάτων είχε κυρίαρχο ρόλο στην απόδοση του συνολικού συστήματος και της αρχιτεκτονικής συνολικά. Ο μηχανισμός κλιμάκωσης και τα προφίλ πόρων απέδωσαν αξιοσημείωτα κέρδη απόδοσης σε σύγκριση με σχετικές ερευνητικές εργασίες στη βιβλιογραφία, όχι μόνο όσον αφορά την ικανοποίηση των κριτηρίων ΠτΥ των χρηστών αλλά και την αποτελεσματική χρησιμοποίηση των πόρων.

Κεφάλαιο 4

Σε αυτό το κεφάλαιο επεκτείνουμε την εργασία του Κεφαλαίου 3 και σε αντίθεση με τη μονόπλευρη φιλοσοφία του κλασικού αλγορίθμου αυτοκλιμάκωσης (HPA) του Κυβερνήτη που λαμβάνει υπόψη μόνο τις μετρήσεις απόδοσης, προτείνουμε μια ευέλικτη αρχιτεκτονική για τη διαχείριση πόρων. Αν και αξιολογούμε την προτεινόμενη αρχιτεκτονική με μια διαδικτυακή εφαρμογή, η λύση μας μπορεί να προσαρμοστεί στις ανάγκες οποιασδήποτε εφαρμογής. Για το πρόβλημα προγραμματισμού εργασιών, προτείνεται ένα σχήμα που βασίζεται στην πολλαπλασιαστική μείωση της προσθετικής αύξησης (AIMD). Εκμεταλλευόμενοι τις εγγυήσεις σταθερότητας αυτής της νέας λύσης προγραμματισμού εργασιών, ανακατευθύνουμε δυναμικά τα εισερχόμενα αιτήματα προς τις ρέπλικες της εικονοποιημένης εφαρμογής. Όπως και πριν, για να αντιμετωπίσουμε δυναμικούς φόρτους εργασίας, ένας μηχανισμός πρόβλεψης μας επιτρέπει να εκτιμήσουμε τον αριθμό των εισερχόμενων αιτημάτων. Επιπλέον, εισάγεται ένας μηχανισμός προφίλ εφαρμογών βάσει μηχανικής μάθησης για την αντιμετώπιση της κλιμάκωσης, συν-σχεδιάζοντας τις θεωρητικά υπολογισμένες μετρικές παροχής υπηρεσιών που λαμβάνονται από τον αλγόριθμο AIMD, με τις τρέχουσες μετρήσεις απόδοσης. Η προτεινόμενη λύση συγκρίνεται με τις σύγχρονες τεχνικές αυτόματης κλιμάκωσης κάτω από ένα ρεαλιστικό φόρτο εργασίας σε μια μικρή υποδομή και αναλύεται η σύγκριση μεταξύ της χρήσης πόρων και των παραβιάσεων ΠτΥ. Οι βασικές συνεισφορές αυτού του κεφαλαίου συνοψίζονται ως εξής:

- Μια ολιστική κλιμακούμενη αρχιτεκτονική για την αντιμετώπιση του κοινού προβλήματος του προγραμματισμού εργασιών και της προληπτικής ενορχήστροσωσης πόρων σε μία υποδομή ΥΠ. Το προτεινόμενο σχήμα AIMD που ενεργοποιείται αν συμβεί κάποιο συγκεκριμένο συμβάν διευκολύνει την προληπτική κλιμάκωση των πόρων. Αυτός ο μηχανισμός επιτρέπει την αποκεντρωμένη ενορχήστρωση πόρων στην άκρη του δικτύου.
- Εισάγεται μια διαφορετική μοντελοποίηση προφίλ εφαρμογής που αξιοποιεί πληροφορίες από (α) τη λύση προγραμματισμού και κατανομής πόρων, (β) τα παρακολουθούμενα ΒΔΑ και (γ) τον αλγόριθμο πρόβλεψης φόρτου εργασίας για την εκτίμηση του αριθμού αντιγράφων της εικονοποιημένης εφαρμογής. Συλλέγοντας ένα σύνολο διακριτών προφίλ πόρων, μπορούμε να βελτιστοποιήσουμε τη χρήση των πόρων χωρίς να παραβιάσουμε τη συνολική σταθερότητα του συστήματος. Στη συνέχεια, η απόφαση κλιμάκωσης εκτελείται χρησιμοποιώντας μια απλή τεχνική μηχανικής μάθησης που μας επιτρέπει να ενσωματώσουμε επίσης τις μετρήσεις απόδοσης. Ο προτεινό-

μενος σχεδιασμός είναι επεκτάσιμος και εύκολα τροποποιήσιμος.

Η δικιά μας λύση μας υπερτερεί άλλων λύσεων που χρησιμοποιούνται συνήθως για αυτόματη κλιμάκωση, καθώς οι μέσοι πόροι της χρήσης των διαθέσιμων επεξεργαστών είναι τουλάχιστον 7% λιγότερο, έχοντας μόνο μια μικρή αύξηση στις παραβιάσεις ΠτΥ.

Κεφάλαιο 5

Το Κεφάλαιο 5 εστιάζει στην ανάπτυξη ενός εναλλακτικού διακοπτικού μηχανισμού μεταφόρτωσης υπολογιστικών διεργασιών στα άκρα του δικτύου, για εφαρμογές της Βιομηχανίας 4.0. Οι εφαρμογές αυτές, απευθύνονται σε ρομπότ τα οποία εκτελούν περίπλοκες διεργασίες, οι οποίες παρουσιάζουν αυστηρές απαιτήσεις τόσο σε χρονική απόκριση όσο και σε ασφάλεια. Σε αυτό το πλαίσιο, η μεταφόρτωση των διεργασιών στα άκρα του δικτύου επιτρέπει στα ρομπότ να ελαφρύνουν τον υπολογιστικό τους φόρτο, αναθέτοντας την εκτέλεση των παραπάνω διεργασιών σε μία ισχυρή υπολογιστική υποδομή σε κοντινή απόσταση. Σε αυτό το κεφάλαιο, λοιπόν, προτείνεται ένας διακοπτικός μηχανισμός μεταφόρτωσης διεργασιών, ενώ σχεδιάζονται ευκαιριακές στρατηγικές μεταφόρτωσης για εφαρμογές που αφορούν στον προγραμματισμό της πορείας και τον εντοπισμό της θέσης των ρομπότ. Η απόφαση για τη μεταφόρτωση λαμβάνεται βάσει της αβεβαιότητας ως προς την τρέχουσα θέση του ρομπότ και την διαθεσιμότητα υπολογιστικών και δικτυακών πόρων στα άκρα του δικτύου, την δεδομένη στιγμή. Το διακοπτικό αυτό σύστημα υλοποιείται και αξιολογείται χρησιμοποιώντας ένα πραγματικό ρομπότ σε μια πραγματική υποδομή στα άκρα του δικτύου· κατά την αξιολόγηση τονίζεται το αντιστάθμισμα ανάμεσα στο χρόνο ολοκλήρωσης των διεργασιών και την επιτυχή έκβαση της αποστολής τους.

Αναλυτικά, το σενάριο που παρουσιάζεται σε αυτό το κεφάλαιο περιγράφει ένα ρομπότ εξοπλισμένο με αισθητήρες και υπολογιστικούς και δικτυακούς πόρους, το οποίο επιχειρεί να φτάσει από ένα αρχικό σε ένα τελικό σημείο, εντός ενός εργοστασιακού χώρου, ανάμεσα σε εμπόδια. Αυτή η λειτουργικότητα είναι βασική για την υλοποίηση εφαρμογών που αφορούν στον εφοδιασμό και την αποθήκευση εμπορευμάτων. Ένα σύνηθες πρόβλημα που αντιμετωπίζεται σε τέτοιου είδους σενάρια είναι η αβεβαιότητα γύρω από την ακριβή «στάση» (θέση και προσανατολισμό) ενός ρομπότ, η οποία αυξάνεται με τον χρόνο κατά την κίνηση, λόγω συσσωρευόμενων ανακριβειών των αισθητήρων, ολισθημάτων των τροχών και αστοχιών στο υλικό. Συνεπώς, η ανάγκη για μια ακριβή, δυναμικά ρυθμιζόμενη τεχνική εντοπισμού θέσης είναι εμφανής. Οι βασικές συνεισφορές αυτού του κεφαλαίου συνοψίζονται, λοιπόν, ως εξής:

- σχεδίαση και υλοποίηση ενός πρωτότυπου μηχανισμού μεταφόρτωσης υπολογιστικών διεργασιών για ρομποτικές εφαρμογές, ο οποίος χρησιμοποιεί μια υπολογιστική υποδομή στα άκρα του δικτύου ενός βιομηχανικού χώρου, για να βελτιώσει την ακρίβεια του εντοπισμού θέσης και της πορείας του ρομπότ.
- σχεδίαση και υλοποίηση ενός αλγορίθμου απόφασης μεταφόρτωσης υπολογιστικών

διεργασιών, ο οποίος λαμβάνει υπόψιν την δυναμική φύση των κινήσεων του ρομπότ και αντιμετωπίζει την αβεβαιότητά που προκαλούνε στον ακριβή εντοπισμό της θέσης τους στο χρόνο.

- σχεδίαση και υλοποίηση καινοτόμων αλγορίθμων εντοπισμού θέσης και προσανατολισμού, οι οποίοι επιτυγχάνουν υψηλή ακρίβεια χρησιμοποιώντας το απλούστερο σύστημα καμερών και τον ελάχιστο αριθμό εντοπισμένων διακριτών σημείων στο περιβάλλον.

Σχετικά με την τεχνική εντοπισμού θέσης, στην εικόνα 5.12 φαίνεται ότι παρόλο που η απόκλιση από την πραγματική θέση αυξάνεται όσο αυξάνεται η απόσταση του ρομπότ από τα ορόσημα, η ακρίβεια της δεν πέφτει ποτέ κάτω από 93%. Σχετικά με τον σχεδιασμό της πορείας του ρομπότ, ο αλγόριθμος μας εγγυάται την εγγύτητα μιας υπολογισμένης τροχιάς με την πραγματικά βέλτιστη, η οποία βρίσκεται εντός αποδεκτών πλαισίων για την εφαρμογή. Προχωρώντας στο βασικό κομμάτι της αξιολόγησης, αυτό του διακοπτικού μηχανισμού μεταφόρτωσης υπολογιστικών διεργασιών, γίνεται η σύγκρισή του με τις δύο ακραίες αλλά απλοϊκές προσεγγίσεις εκτέλεσης των υπολογιστικών διεργασιών: α) μόνο τοπική εκτέλεση και β) μόνο υπολογιστική μεταφόρτωση. Από τα αποτελέσματα, γίνεται εμφανής η υπεροχή του προτεινόμενου μηχανισμού, τόσο στον μέσο χρόνο εκτέλεσης όσο και στο ποσοστό επιτυχούς έκβασης των αποστολών που κλήθηκε να υλοποιήσει το ρομπότ. Αναλυτικότερα, στη μόνο τοπική εκτέλεση της σχεδίασης διαδρομής, ο χρόνος εκτέλεσης του πειράματος είναι γραμμικώς ανάλογος των βημάτων που αποφασίζει ο A$*$ αλγόριθμος που εκτελείται, ενώ η αβεβαιότητα στον εντοπισμό της θέσης του ρομπότ δεν σταματά ποτέ να αυξάνεται (αφού το σύστημα καμερών δεν αξιοποιείται ποτέ. Συνεπώς δεν υπάρχει κάποια εγγύηση σχετικά με την επιτυχή περάτωση του πειράματος. Στην περίπτωση β), της μεταφορτωμένης εκτέλεσης μόνο, από τη μία έχουμε την εγγύηση για την επιτυχή περάτωση του πειράματος, αφού το σύστημα καμερών χρησιμοποιείται για τον εντοπισμό της θέσης του ρομπότ κάθε φορά, έχουμε όμως αυξημένο μέσο χρόνο εκτέλεσης της αποστολής, λόγω ακριβώς αυτής της επιλογής: η υπολογιστικά πιο ακριβή τεχνική εντοπισμού θέσης χρησιμοποιείται άκριτα, κάθε φορά που χρειάζεται εντοπισμός, ακόμα και όταν η αβεβαιότητα σχετικά με την στάση είναι ελάχιστη και δυνητικά το ρομπότ θα μπορούσε να κινηθεί για κάποιο χρονικό διάστημα και με τα αποτελέσματα του τοπικού εντοπισμού θέσης. Η σύγκριση αυτή καταλήγει με τα αποτελέσματα του διακοπτικού μηχανισμού μεταφόρτωσης διεργασιών στα άκρα του δικτύου, ο οποίος τα καταφέρνει σημαντικά καλύτερα σε σχέση με τις άλλες δύο προσεγγίσεις, παρέχοντας τόσο εγγυήσεις για την επιτυχή περάτωση του πειράματος, όσο και χαμηλότερους μέσους χρόνους εκτέλεσης.

Κεφάλαιο 6

Αυτό το κεφάλαιο εστιάζει ξανά στην περίπτωση των ρομποτικών ΚΦΣ. Το κινητό ρομπότ που εξετάζεται είναι ένας μονόκυκλο, που υπόκειται σε αβεβαιότητες μοντελοποίησης και μέτρησης. Στο προτεινόμενο σενάριο, το ρομπότ πρέπει να λύσει το πρόβλημα του εντοπισμού της θέσης του, πρόβλημα και στη συνέχεια το πρόβλημα παρακολούθησης και υλοποίησης μιας συγκεκριμένης τροχιάς. Κάτω από αυτήν τη ρύθμιση, είναι διαθέσιμος ένας μηχανισμός μεταφόρτωσης για τη μετάδοση των δεδομένων εντοπισμού θέσης για τη διαδικασία εντοπισμού σε μια υποδομή ΥΠ για περαιτέρω επεξεργασία. Κύριος στόχος είναι να παρέχουμε ντετερμινιστικές εγγυήσεις για την εκτιμώμενη θέση του ρομπότ και στη συνέχεια να εγγυηθούμε τη σύγκλιση προς ένα σημείο-στόχο. Η ενσωμάτωση μεθόδων εκτίμησης βάσει συνόλου στον σχεδιασμό του ελεγκτή για την παροχή πλοήγησης σε πραγματικό χρόνο, είναι αρκετά δύσκολη.

Σε ένα έξυπνο εργοστασιακό περιβάλλον, τα ρομπότ είναι εξοπλισμένα με αισθητήρες που μπορούν να παρέχουν εκτιμήσεις της στάσης του ρομπότ (δηλαδή, δισδιάστατη θέση και προσανατολισμό). Αν και η χρήση ενσωματωμένης τοπικής εκτίμησης θέσης βάσει αισθητήρων αποφέρει γρήγορα αποτελέσματα, είναι γνωστό ότι είναι επιρρεπής σε συσσωρευτικά σφάλματα, ειδικά όταν πρόκειται για μακριές τροχιές. Έτσι, απαιτούνται συνήθως πιο εξελιγμένες αλλά υπολογιστικά απαιτητικές τεχνικές για να αυξηθεί η ακρίβεια εντοπισμού. Με στόχο την ανάλυση των παραπάνω και την πρόταση νέων στρατηγικών συνσχεδιασμού ελέγχου που εγγυώνται τη σωστή συμπεριφορά ενός τέτοιου συστήματος, αυτό το κεφάλαιο παρουσιάζει μια μεθοδολογία συν-σχεδιασμού ελέγχου για πλοήγηση κινητών ρομπότ. Το ρομπότ, εξοπλισμένο με κάμερες και αισθητήρες οδομετρίας, πλοηγείται από μια θέση εκκίνησης σε μια θέση στόχο, για να ολοκληρώσει μια δεδομένη αποστολή (π.χ. μια αυτοματοποιημένη αποθήκευση/ανάκτηση). Σε αυτό το περίπλοκο σενάριο, διερευνάται η θεμελιώδης αναζήτηση ισορροπιας μεταξύ απόδοσης και καταναλωμένων πόρων, μαζί με τις συνθήκες που εγγυώνται τη σύγκλιση του συστήματος. Συνολικά, επεκτείνουμε τη μοντελοποίηση του Κεφαλαίου 5 και παρέχουμε εγγυήσεις κλειστού βρόχου για τη σύγκλιση του ρομπότ και τη βέλτιστη χρήση των πόρων. Οι κύριες συνεισφορές της εργασίας μας που τη διαφοροποιούν από την υπόλοιπη βιβλιογραφία συνοψίζονται ως εξής:

1. Εισάγεται ένας συν-σχεδιασμός για ΚΦΣ όπου ένα φορητό ρομπότ τύπου μονόκυκλου χρησιμοποιεί τόσο τους τοπικούς πόρους όσο και απομακρυσμένους για να εκτελέσει τις υπολογιστικά απαιτητικές εργασίες μιας εφαρμογής της Βιομηχανίας 4.0 που απαιτεί πλοήγηση σε εργοστασιακό χώρο. Δύο θεμελιώδη προβλήματα αντιμετωπίζονται από κοινού: (i) η σύνθεση ελεγκτών που υπαγορεύουν την κίνηση του μονοκύκλου ρομπότ σε αυτό το πρόβλημα σχεδιασμού διαδρομής και (ii) μια στρατηγική μεταφόρτωσης για την αντιστάθμιση της αβεβαιότητας των τοπικών τεχνικών εκτίμησης με τις πιο ακριβείς απομακρυσμένες.

2. Οι νέοι ελεγκτές έχουν σχεδιαστεί για να ικανοποιούν τον θεμελιώδη στόχο της αποστολής, δηλαδή να διασφαλίζουν τη σύγκλιση της πλοήγησης του κινητού ρομπότ σε ένα σύνολο στόχων. Για τη δυναμική του ρομπότ θεωρείται ένα κινηματικό μοντέλο μονοκύκλου, ενώ η κίνησή του αναλύεται σε δύο μέρη, δηλαδή σε περιστροφικό και μεταφορικό. Σημειώνουμε ότι, παρόλο που υπάρχουν έργα στη βιβλιογραφία

που παρέχουν κομψούς στιβαρούς ελεγκτές παρουσία αβεβαιοτήτων, η προτεινόμενη μέθοδος μας επιτρέπει να αντιμετωπίσουμε τρεις διακριτές προκλήσεις, συγκεκριμένα, (i) αβεβαιότητες/διαταραχές που επηρεάζουν τη δυναμική της κατάστασης, (ii) αποτελεσματική εφαρμογή εκτίμησης βάσει συνόλου χρησιμοποιώντας μετρήσεις οδομετρίας, (iii) εγγύηση σύγκλισης του συστήματος κλειστού βρόχου σε ένα σημείο αναφοράς. Για να μετριαστεί περαιτέρω η υπολογιστική πίεση από το ρομποτ, χρησιμοποιούνται κατά προσέγγιση υπολογισμοί για την τοπική εκτίμηση της στάσης του ρομπότ. Στη συνέχεια, εφαρμόζονται μηχανισμοί ελέγχου σταθεροποίησης της ανάδρασης κατάστασης και στις δύο κινήσεις, για να εξασφαλιστεί η σύγκλιση στην περιοχή στόχο.

3. Μια διαδικασία λήψης αποφάσεων που βασίζεται σε μια συνάρτηση είναι κατάλληλα διαμορφωμένη για να αναλάβει στρατηγική μεταφόρτωσης. Αυτή η στρατηγική υπαγορεύει ποιες από τις δύο διαφορετικές διαθέσιμες τεχνικές εντοπισμού θα χρησιμοποιηθούν. Εκτός από την ποιότητα της πλοήγησης, η οποία αποκτάται από τις εξόδους των ελεγκτών, αυτή η διαδικασία λαμβάνει επίσης υπόψη την διαθεσιμότητα των δικτυακών πόρων και των διαθέσιμων πόρων της υποδομής.

Εκτελείται μια σειρά πειραμάτων για την αξιολόγηση της απόδοσης του προτεινόμενου ΚΦΣ όσον αφορά την ακρίβεια πλοήγησης και τη διάρκεια της αποστολής. Η αξιολόγηση δείχνει ότι η επιθυμητή λύση σχετικά με την προτίμηση των δύο αλγορίθμων εντοπισμού είναι μια μικτή στρατηγική που χρησιμοποιεί και τους δύο. Η χρήση αποκλειστικά τοπικά παραγόμενης εκτίμησης δεν αρκεί για να παρέχει αρκετά υψηλή ακρίβεια, ενώ η συνεχής αναζήτηση ακριβέστερης εκτίμησης από τον εξ αποστάσεως εκτελούμενο αλγόριθμο προσθέτει σημαντικό κόστος στη διάρκεια της αποστολής. Επιπλέον, μια λεπτομερής συγκριτική αξιολόγηση με εναλλακτικά σχήματα μεταφόρτωσης καταδεικνύει τα οφέλη του πλαισίου μας, καθώς και την προσαρμοστικότητά του στις συγκεκριμένες απαιτήσεις της εφαρμογής. Η αξιολόγηση απόδοσης της προτεινόμενης τεχνικής υποδηλώνει ότι η λύση μας ξεπερνά τα άλλα τυπικά χρησιμοποιούμενα σχήματα μεταφόρτωσης και είναι εύκολα προσαρμόσιμη στις ανάγκες διαφορετικών χαρακτηριστικών εφαρμογής. Το πιο σημαντικό είναι ότι το προτεινόμενο πλαίσιο εγγυάται τη σύγκλιση με τη θέση στόχο ανεξάρτητα από τις διάφορες παραμέτρους που επιλέγονται, σε αντίθεση με τα συστήματα περιοδικής μεταφόρτωσης.

Κεφάλαιο 7

Αυτό το κεφάλαιο εστιάζει στην περίπτωση εφαρμογών αναγνώρισης προτύπων σε πραγματικό χρόνο (δηλαδή, ανίχνευσης εικόνας) που αναπτύσσονται σε διάφορα σενάρια. Υποθέτουμε ότι οι διεργασίες αναγνώρισης προτύπων (π.χ. εικόνες ή ροή βίντεο) που δημιουργούνται από συσκευές χαμηλής κατανάλωσης υποβάλλονται σε επεξεργασία είτε τοπικά είτε μεταφορτώνονται, χρησιμοποιώντας ασύρματη σύνδεση για περαιτέρω επεξεργασία. Οι μεταφορτωμένες διεργασίες συμπιέζονται πρώτα στις συσκευές για να μειωθεί ο χρόνος μετάδοσης. Αντίστοιχα, όταν μια εργασία φτάνει σε μια υποδομή ΥΠ, αποκωδικοποιείται και στη συνέχει εκτελείται η αναγνώριση εικόνας. Έτσι, διερευνούμε την ισορροπία μεταξύ της ελαχιστοποίησης του χρόνου μετάδοσης (δηλαδή, της συμπίεσης του μεγέθους της διεργασίας) και της απώλειας ακρίβειας του συμπεράσματος. Επιπλέον, θεωρούμε ότι κάθε υποδομή έχει μια διαθέσιμη μονάδα κάρτας γραφικών που βοηθά με τους υπολογισμούς. Από την πλευρά της υποδομής, υποθέτουμε ότι οι διεργασίες εκτελούνται παράλληλα, δηλ. σε μια παρτίδα. Η ομαδική επεξεργασία των διεργασιών μειώνει δραστικά τη συνολική καθυστέρηση σε σύγκριση με τη διαδοχική εκτέλεση των εργασιών. Επιπλέον, ο χρόνος λειτουργίας της κάρτας γραφικών μειώνεται, ελαχιστοποιώντας έτσι το κόστος και την ενέργεια από την πλευρά της υποδομής. Ωστόσο, για να βελτιστοποιηθεί η επεξεργασία κατά παρτίδες, οι εργασίες που έχουν μεταφορτωθεί πρέπει να φτάνουν στην υποδομή ταυτόχρονα για να μειωθεί η συνολική καθυστέρηση. Συνολικά, στοχεύουμε σε μια στρατηγική μεταφόρτωσης διεργασιών για προγραμματισμό παρτίδας που (α) διευκολύνει τη συνεργασία μεταξύ συσκευών και υποδομής για τη μείωση της μέσης καθυστέρησης και (β) διατηρεί μια ορισμένη ποιότητα της αναγνώρισης εικόνας. Οι συνεισφορές μας συνοψίζονται ως εξής:

- Διατυπώνουμε ένα κοινό πρόβλημα βελτιστοποίησης για να μεγιστοποιήσουμε τη μέση ακρίβεια και ταυτόχρονα να ελαχιστοποιήσουμε τον μέσο χρόνο απόκρισης. Εισάγουμε ένα πλαίσιο για να προσεγγίσουμε τη λύση του προβλήματος βελτιστοποίησης χωρίζοντάς το σε δύο υποπροβλήματα: (α) να υπολογίσουμε μια βέλτιστη συμπίεση των διεργασιών και (β) να αποφασίσουμε μια πολιτική μεταφόρτωσης των διεργασιών για τον προγραμματισμό των παρτίδων.
- Ακολουθώντας το παράδειγμα της προσεγγιστικής υπολογιστικής, εξετάζουμε πάλι δύο εκδοχές του ίδιου αλγορίθμου, συγκεκριμένα: (i) μία υπολογιστικά ελαφριά εκδοχή στις συσκευές (ii) και μία εκδοχή που δίνει πολύ καλά αποτελέσματα στην απομακρυσμένη υποδομή με δυνατότητα αξιοποίησης της κάρτας γραφικών και παράλληλου υπολογισμού. Αρχικά υπολογίζουμε μια βέλτιστη στρατηγική συμπίεσης. Η λύση στο πρόβλημα της συμπίεσης εισάγει μερικά σημαντικά αποτελέσματα που υποδηλώνουν ότι ο προγραμματισμός παρτίδας βελτιστοποιείται όταν οι εργασίες φτάνουν ταυτόχρονα, ελαχιστοποιώντας έτσι τη συνολική καθυστέρηση της εφαρμογής.

Πραγματοποιούμε μια εκτενή αξιολόγηση του προτεινόμενου πλαισίου μέσω μοντελοποίησης και προσομοίωσης. Τα αποτελέσματα δείχνουν ότι η προτεινόμενη τεχνική υπερέχει όλων των άλλων μεθόδων αναφοράς, έχοντας μια σχεδόν βέλτιστη λύση και επιτυγχάνοντας ελάχιστο συνολικό κόστος. Συγκεκριμένα η ανώτερη απόδοση της προτεινόμενης προσέγγισης αποδεικνύεται σε συγκριτικά πειράματα, καθώς το συνολικό κόστος είναι τουλάχιστον 50% λιγότερο από όλες τις άλλες τεχνικές αναφοράς και βελτιώνεται ακόμη περισσότερο καθώς αυξάνεται ο αριθμός των διεργασιών. Τέλος, διερευνούμε την επίδραση όλων των σημαντικών παραμέτρων στη συνολική απόδοση του πλαισίου σε μια σειρά πειραμάτων.

Κεφάλαιο 8
Στο τελευταίο κεφάλαιο της διατριβής εξάγουμε συμπεράσματα αναφορικά με όλη την δουλειά μας και προτείνουμε μελλοντικές κατευθύνσεις. Τα συμπεράσματα στα οποία καταλήξαμε καθ' όλη τη διάρκεια της διατριβής συνοψίζονται παρακάτω:

- Οι Θεωρητικές Προσεγγίσεις Αυτόματου Ελέγχου μπορούν να παρέχουν ισχυρά εργαλεία για την υποστήριξη της διαχείρισης πόρων και της μεταφόρτωσης στα ΚΦΣ. Λόγω της ανάγκης για αποφάσεις σε πραγματικό χρόνο στον προγραμματισμό εργασιών και τις αποφάσεις μεταφόρτωσης. Παρέχοντας μια ακριβή κατανόηση της συμπεριφοράς του συστήματος, αυτές οι προσεγγίσεις προσφέρουν πολυάριθμα οφέλη όσον αφορά την κατανομή πόρων και την μεταφόρτωση. Μέσω της μαθηματικής μοντελοποίησης και της ανάλυσης της δυναμικής του συστήματος, αυτές οι προσεγγίσεις μπορούν να εντοπίσουν τομείς αναποτελεσματικότητας στη χρήση πόρων και να σχεδιάσουν αλγόριθμους ελέγχου για τη ρύθμιση της συμπεριφοράς του συστήματος. Αυτοί οι αλγόριθμοι μπορούν να προσαρμόσουν δυναμικά την κατανομή πόρων με βάση την τρέχουσα κατάσταση και τους στόχους απόδοσης του συστήματος, οδηγώντας σε βελτιωμένη απόδοση, και σταθερότητα.
- Ο προγραμματισμός πόρων και η δυναμική κατανομή πόρων είναι βασικές προκλήσεις που πρέπει να αντιμετωπιστούν από κοινού. Η μεγιστοποίηση της απόδοσης των σύγχρονων εφαρμογών απαιτεί έγκαιρη διαχείριση πόρων των εικονικών πόρων. Ωστόσο, η προληπτική ανάπτυξη πόρων για την ικανοποίηση συγκεκριμένων απαιτήσεων εφαρμογών που υπόκεινται σε δυναμικό φόρτο εργασίας εισερχόμενων αιτημάτων είναι εξαιρετικά δύσκολη. Για παράδειγμα, εάν ο προγραμματισμός πόρων δεν εφαρμοστεί σωστά, μπορεί να οδηγήσει σε συμφόρηση στην επεξεργασία, προκαλώντας καθυστερήσεις και μειωμένη απόδοση του συστήματος.
- Η στρατηγική διακόπτικων συστημάτων απόφασης της μεταφόρτωσης μπορεί να είναι επωφελής τόσο για την απόδοση της εφαρμογής όσο και για τη χρήση των πόρων.
- Η κατάλληλη αντιστοίχιση των πόρων σε έναν δυναμικό φόρτο εργασίας και η ενσωμάτωση της δυναμικής και των διαφόρων κριτηρίων απόδοσης ήταν μια βασική πρόκληση σε αυτή τη διατριβή. Αξιοποιώντας τους Μηχανισμούς Προφίλ Εφαρμογών, τα προβλήματα κατανομής πόρων και προγραμματισμού πόρων μπορούν να ωφεληθούν πολύ παρέχοντας απλούστερες, ωστόσο, σταθερές λύσεις. Ως εκ τούτου, απαιτούνται αποτελεσματικοί και αποδοτικοί αλγόριθμοι χαμηλής πολυπλοκότητας, καθώς η φάση εκπαίδευσης του προφίλ εφαρμογής μπορεί να χρειαστεί πολύ χρόνο ή πολλά πειράματα.
- Η ενοποίηση ολιστικών αρχιτεκτονικών με πλαίσια πραγματικού κόσμου δημιουργεί νέα προβλήματα και προκλήσεις. Πολλές από τις ερευνητικές προκλήσεις που

συναντήθηκαν σε αυτή τη διατριβή ανακαλύφθηκαν κατά την προσπάθεια ενσωμάτωσης των προτεινόμενων μηχανισμών με πλαίσια τελευταίας τεχνολογίας όπως το Kubernetes και το OpenStack. Η προσπάθεια οικοδόμησης ολιστικών αρχιτεκτονικών συμβατών με αυτά τα πλαίσια και ευθυγραμμισμένες με τα τρέχοντα πρότυπα και τις ερευνητικές τάσεις του Ακαδημαϊκού χώρου και της βιομηχανίας ήταν η κινητήρια δύναμη σε όλη αυτή τη διατριβή.

Ολοκληρώνοντας αυτή τη διατριβή, αυτή η τελευταία ενότητα αποσαφηνίζει ορισμένες από τις πιθανές μελλοντικές κατευθύνσεις έρευνας που μπορούν να ακολουθηθούν με βάση τα αποτελέσματα της εργασίας που παρουσιάζεται και τις προκλήσεις που αντιμετωπίζει. Μια ενδιαφέρουσα κατεύθυνση είναι η εξέταση διαφόρων ΒΔΑ από την πλευρά της υποδομής για το πρόβλημα της δυναμικής κατανομής πόρων. Ως εκ τούτου, το πρόβλημα βελτιστοποίησης που καθορίζει τον αριθμό των αναπτυγμένων πόρων που εισάγονται στο Κεφάλαιο 3, θα μπορούσε να βελτιωθεί ώστε να επικεντρωθεί στη βελτιστοποίηση ισχύος ελαχιστοποιώντας τον αριθμό των ενεργών εξυπηρετητών. Μια άλλη ενδιαφέρουσα εργασία είναι η συμπερίληψη μη ντετερμινιστικών/στοχαστικών προσεγγίσεων που θα μπορούσαν να αξιολογηθούν για σκοπούς εκτίμησης κινητικότητας χρηστών και εκτίμησης του φόρτου μιάς εφαρμογής. Αυτό μπορεί να επιτευχθεί αξιοποιώντας τις τεχνικές Μηχανικής Μάθησης και συνδυάζοντάς τες με πιο λεπτομερή μοντελοποίηση της παρεμβολής σήματος μεταξύ των χρηστών, προκειμένου να προσφέρουμε τελικά μια πιο αποτελεσματική και αποδοτική απόφαση μεταφόρτωσης.

Λέξεις Κλειδιά: Μεταφόρτωση Υπολογιστικών Διεργασιών, Προγραμματισμός εργασιών, Δυναμική κατανομή πόρων, Διαχείριση πόρων, Θεωρία ελέγχου, Διαδίκτυο των πραγμάτων, Θεωρία Συστημάτων Αναμονής

Contents

Acknowledgements

Η παρούσα εργασία είναι το αποτέλεσμα της διδακτορικής μου διατριβής στο εργαστήριο NETMODE τα τελευταία 5 χρόνια, στο πλαίσιο ενός ανοιχτού και δημόσιου πανεπιστημίου. Στα παρακάτω κεφάλαια δεν φαίνεται επαρκώς πόσο σημαντικοί ήταν κάποιοι άνθρωποι που συνάντησα σε αυτή την διαδρομή. Οπότε εδώ θα προσπαθήσω να τους ευχαριστήσω, δυστυχώς χωρίς να φτάνουν τα λόγια.

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα Καθηγητή μου κ. Συμεών Παπαβασιλείου, για την ευκαιρία και την εμπιστοσύνη που μου έδειξε επιτρέποντάς μου να γίνω μέρος του εργαστηρίου NETMODE. Πέρα από την εξαιρετική συνεργασία, την καθοριστική καθοδήγηση και την επιστημονική του κατάρτιση όποτε χρειαζόταν, θέλω ιδιαίτερα να επισημάνω τη στήριξη που έδειξε σε όλους μας όλα αυτά τα χρόνια. Η κουβέντα, η συζήτηση για προβληματισμούς αλλά και το χτίσιμο μιας σχέσης εμπιστοσύνης, ήταν όλα καθοριστικά για την προσωπική μου πορεία εν γένει αλλά κυρίως για την αφοσίωση σε αυτή την πολλές φορές επίπονη πορεία της διατριβής.

Δεν θα μπορούσα εδώ να μην αναφερθώ στον Καθηγητή Νίκο Αθανασόπουλο, που υπήρξε για μένα ένας πραγματικός δάσκαλος. Πέρα από την εξαιρετική συνεργασία και την συμμετοχή του στις περισσότερες δουλειές αυτής της διατριβής, τις καθόλου δεδομένες ευκαιρίες που μου έδωσε, τις γνώσεις του και σίγουρα την υπομονή του και την αφοσίωση σου, θέλω να πω κάτι άλλο. Ο Νίκος είναι ένας άνθρωπος κόσμημα για την ακαδημαϊκή κοινότητα και το πανεπιστήμιο, που με την όρεξή του, τον ρομαντισμό για την έρευνα και την αφοσίωση για την επιστήμη, εμπνέει όποιον έχει την τύχη να συνεργαστεί μαζί του.

Ένας καφές στα νέα κτίρια Ηλεκτρολόγων μετά από ένα χρόνο διπλωματικής με τον Δημήτρη Δεχουνιώτη ήταν αρκετός για την απόφαση να ξεκινήσω το διδακτορικό μου. Ό,τι και να γράψω σε αυτές τις γραμμές είναι πολύ λίγο. Έχω την τύχη να θεωρώ τον Δημήτρη φίλο μου μετά από αυτά τα χρόνια. Πέρα από την εξαιρετική συνεργασία, την εμπιστοσύνη, το πείραγμα, τα τσιγάρα, και την υπομονή του - κρίσιμα όλα - θα ήθελα να τονίσω ότι η όρεξη μου πηγαίνοντας στο γραφείο ήταν απόλυτα συνδεδεμένη με το ότι μοιραζόμουν την καθημερινότητά μου με αυτόν τον σπουδαίο άνθρωπο. Ο Δημήτρης δεν έχει ανάγκη να ανταγωνιστεί κανέναν και τον ευχαριστώ που μου έδειξε ένα δρόμο συναδελφικό και παρεΐστικο. Μοιράζει απλόχερα στους ανθρώπους δίπλα του και τον ευχαριστώ ειλικρινά που "με έκανε άνθρωπο" (και όχι μόνο επαγγελματικά) όπως λέει και ο ίδιος και κάτι ξέρει.

Πολλά έχουν αλλάξει τα τελευταία χρόνια στο εργαστήριο. Αλλά ένα ιδιαίτερο ευχαριστώ θέλω να πω στους φίλους και συνεργάτες Γιάννη Δημολίτσα και Μάρω Διαμαντή γιατί ξεκινήσαμε και γεράσαμε παρέα, και τελικά μοιραστήκαμε τόσα πράγματα. Ιδιαίτερα με τον Γιάννη και την εξαιρετική συνεργασία σε πολλές δουλειές, τον ευχαριστώ για την υπομονή του και την κατανόηση του και τις κουβέντες ψυχολογικής στήριξης εκατέρωθεν. Και στους δύο εύχομαι η πορεία που θα ακολουθήσουν να συνεχίσει να είναι το ίδιο εντυπωσιακή όσο ήταν ως τώρα.

Στα μέλη του εργαστηρίου που είχα την τύχη να γνωρίσω, τον Βασίλη Καρυώτη για την βοήθεια του όλα αυτά τα χρόνια, την αφοσίωση του στην επιστήμη και στο ταλέντο του στην καλωδίωση, τον Γιώργο Μήτση που έγινε κολλητός μου και ελπίζω να με βλέπει για πάντα σαν 25 χρονών, τον Μάριο Αυγέρη γιατί είναι καταπληκτικός ερευνητής και με έμαθε πάρα πολλά πράγματα, τον Χρήστο Πελέκη για όλη την συνεργασία μας, τον Αδάμ Παυλίδη γιατί είναι ο Αδάμ, τους νεαρούς Νίκους για την υπομονή τους

και την κοροϊδία τους (γιατί μου έλειπε πολύ), και όλους τους άλλους συναδέλφους, το προσωπικό και τους πυλώνες (Γιώτα Κουάκη) του NETMODE, χρωστάω ένα μεγάλο ευχαριστώ για τα πράγματα που μου μάθανε, για τη βοήθειά τους και για την κοινή μας πορεία.

Σε αυτό το σημείο ας μου επιτραπεί να ευχαριστήσω προσωπικά και κάποιους σημαντικούς φίλους/ες που που έχω την τύχη να έχω, για την συμπαράσταση τους και την καταπληκτική τους παρέα, που πέρασαν όλη τη διαδρομή του διδακτορικού μαζί μου, χωρίς αυτούς είμαι σίγουρος ότι αυτή η διατριβή θα ήταν ανέφικτο να ολοκληρωθεί. Τον σπουδαίο και αθεράπευτα δοτικό Λευτέρη Πουλακάκη, την καταπληκτική και πάντα γεμάτη εκπλήξεις Ειρήνη Καρβούνη την σταθερή αξία και (δύσκολα κανείς να περιγράψει) υποστηρικτική Βαλεντίνη Καϊσίδου και την Κατερίνα Καινούργιου που εκτιμώ φοβερά που αντέχει να μεγαλώνουμε παρέα. Θα ήθελα να ευχαριστήσω επίσης μέσα από την ψυχή μου, τον Ορέστη Λάγκα, τον Παναγιώτη Τουμάση και τον Κωστή Παπαζαφειρόπουλο. Το πόσο σημαντικό για μένα ήταν να έχω κοινή πορεία στο διδακτορικό όλα αυτά τα χρόνια με αυτούς τους φανταστικούς ανθρώπους είναι κάτι που ελπίζω να ξέρουν και να μην ξεχνάνε. Και αν στη ζωή μερικές φορές τα πράγματα μοιάζουν "orthogonal", κάποιες σχέσεις -οι τόσο καθοριστικές για τη ζωή μας- θα είναι πάντα εκεί να μετράμε βήματα πλάι-πλάι. Στα καταπληκτικά αδέρφια μου Κωστή, Χρήστο, Βάσω αλλά και τον Χάρη, χρωστάω ειλικρινά ευχαριστώ για την στήριξη τους την αγάπη τους όλα αυτά τα χρόνια που τα περνάμε μαζί, μεγαλώνουμε παρέα και είμαστε πάντα εκεί ο ένας για τον άλλον. Να ξέρουν ότι καμία ευχαριστία δεν είναι αρκετή να περιγράψει το τι είναι για μένα. Στους γονείς μου, οφείλω ένα τεράστιο ευχαριστώ για την αγάπη τους την στήριξη τους (υλική μα και συναισθηματική), για όλα αυτά που μου έχουν προσφέρει απλόχερα, την εμπιστοσύνη τους και τη χαρά τους. Για οτιδήποτε δεν διαλέγεις σε αυτή τη ζωή μάλλον πρέπει να αισθάνεσαι τυχερός κάπου κάπου. Και τέλος με πολλά σημαντικά ονόματα να λείπουν και σας ζητώ να με συγχωρέσετε γι' αυτό, θέλω να ευχαριστήσω όλους όσους όλα αυτά τα χρόνια σταθήκαμε πλάι. Που μοιραστήκαμε, ζήσαμε και γελάσαμε, ήπιαμε ποτά και καπνίσαμε σε μπαρ ή καλοκαιρινές διακοπές, ερωτευτήκαμε, περάσαμε καλά ή και άσχημα, αγκαλιαστήκαμε ένα πρωί έξω από το εφετείο, που κλάψαμε μαζί στις μεγάλες προσωπικές (ή και όχι) ήττες και γιορτάσαμε νίκες και χαρές. Ίσως η ολοκλήρωση μιας διατριβής δεν εξαρτάται άμεσα από αυτά όπως λέει ο αγέραστος Κωσταντίνος Σωτηρόπουλος, αλλά η ζωή ενός ΥΔ σίγουρα.

Δημήτρης Σπαθαράκης

The philosophers have only interpreted the world, in various ways.
The point, however, is to change it. - Karl Marx, Theses on Feuerbach - Thesis XI

Structure

The thesis is organized in the following structure.

In Chapter 1, a general introduction on the topics concerning the thesis is presented. We discuss the environment and the research challenges that motivated our work and exhibit the main contributions of this thesis. In Chapter 2, some basic mathematical background is presented. Additional information is provided in the main part of the thesis whenever required. The following chapters are the main chapters of the thesis. Each chapter presents a specific problem with their general setting, along with their related work found in the literature. The proposed framework is discussed, the modeling is thoroughly analyzed, and the mathematical formulation of the problem and the proposed solution are introduced. Finally, a thorough evaluation of the proposed framework is presented. In particular, in Chapter 3, we introduce a holistic framework to assist in the offloading decision of a location-based service that is considered computing-intensive. Users offload their requests to a nearby edge server only when beneficial, i.e., when the communication overhead is minimized. Then, a resource scheduling and dynamic resource allocation mechanism is introduced to tackle with the varying workload of requests, keeping a certain QoS for the response time of requests. In Chapter 4, we extend the previous setting by focusing on formal modeling utilizing concepts from Queuing Theory. The task scheduling algorithm and accompanying resource allocation mechanism provide stability guarantees for the offloaded requests. The evaluation is performed in Kubernetes and we utilized machine learning algorithms to map theoretically-computed values for the processing rate of the deployed resources along with various monitoring KPIs. In Chapter 5, we consider a robotic scenario as part of an Industrial Internet of Things application. A robot follows the concept of approximate computing having two options for executing computing-intensive algorithms, i.e., localization and path planning tasks, in a simple case of moving from one waypoint to another. The locally-implemented algorithms are fast but prune to errors, while the remote ones are precise but computationally intensive. Hence, a switching offloading strategy is realized to optimize the performance of the agent and find a balance between mission duration and accuracy. In Chapter 6, we extend the previous work providing stability and closed-loop guarantees for the robot. A utility-based offloading strategy is realized that can be easily tweaked to assist in various mission scenarios. The overall framework is designed to be applicable to various design options. The set-estimation analysis is presented thoroughly and acts as a hard constraint in the offloading policy, providing guarantees for the safe navigation of the mobile agent. In Chapter 7, we focus on the case of real-time inference and rely on the Approximate Computing paradigm. An interesting trade-off arises as the quality of the Edge-assisted inference process and the overall latency of the system are competing metrics. We formulate a joint optimization problem to maximize the quality of inference while minimizing the overall latency for the GPU-enabled batch processing of inference applications. Finally, in Chapter 8, we present the conclusions drawn from our observations, as well as some propositions for current and future extensions of our work and future directions of the related research.

Chapter 1

Introduction

The proliferation of fifth-generation (5G) networks, the ever-expanding need for mobile or cloud applications, and the Internet of Things (IoT) paradigm bring a new era in Information and Communication Technology (ICT). Despite the remarkable improvements in hardware advances and computing capabilities of the devices in recent years, modern applications require huge amounts of storage, and more often real-time processing capabilities. In most cases, the IoT devices are low-power devices (e.g., sensors, actuators) with limited computing capabilities, and insufficient battery resources and can not guarantee the high performance required by the application [1]. Even, powerful devices (e.g. smartphones, robots) cannot guarantee the required high performance and/or fulfillment of time constraints, for time-critical and mission-critical IoT-enabled applications [2]. As a result, offloading compute-intensive and energy-intensive tasks to a more resilient computing infrastructure for further processing is a key enabler to the realization of the new era of ICT. This solution, called task offloading [3], allows for enhancing the user's experience by providing lower latency, better reliability, and improved energy efficiency for battery-powered devices. In a typical offloading scenario, data are transmitted through wireless links, i.e., cellular or WiFi and the quality of the wireless connection heavily depends on signal strength, interference, packet dropouts, and other parameters related to the wireless environment, which must be considered in the offloading decision. The computation offloading aims to save time and energy on the end devices' side.

However, as Fig. 1.1 illustrates the global projection of Internet users. The Compound Annual Growth Rate (CAGR) is expected to grow 6 percent, making the total number of users an immense number of 5.3 billion by 2023, [4]. As studies show [5], [6], in a few years, one of the most anticipated challenges will be the huge growth in data produced and devices connected since the majority of the devices (sensors, wearables, etc.) will be interconnected. Moreover, new types of applications, in the context of massive Machine to Machine (M2M) [7], will be available to support every aspect of everyday life, significantly affecting the way we perceive the world. Accordingly, the global growth of devices is expected to grow to almost 30 billion in 2023 [4]. These connected devices will

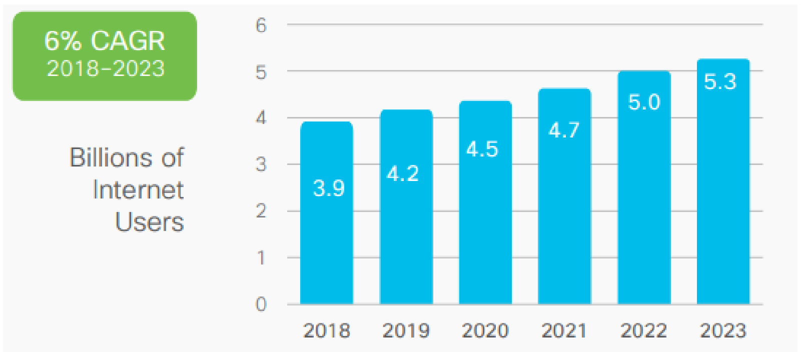


Figure 1.1: Global Internet user growth [4].

constitute the Internet of Things (IoT) and potentially generate a massive amount of data.

Naturally, a well-tested solution of where to seek increased availability of computational resources lies in the Cloud. Cloud Computing (CC) a centralized computing model, has shown its great power of infinite computing capability and on-demand resource provisioning [8], [9]. Therefore, to augment the power of mobile devices, the concept of Mobile Cloud Computing (MCC) was introduced [10]. MCC is still the state-of-practice service delivery paradigm that can extend the resource capabilities of the end devices. Hence, it is common to offload computation-intensive tasks of resource-intensive applications to a centralized Cloud Computing infrastructure speeding up application execution and saving on energy consumption. In current implementations of IoT applications, most data that needs storage, analysis, and decision-making is sent to the data centers in the Cloud [11]. Generally, public cloud vendors have built large data centers in various parts of the world. These large-scale, commodity-computer data centers have enough computing resources to serve a very large number of users.

However, for this vision to become a reality, new demands arise from these large-scale infrastructures, such as the ability to manage and orchestrate massive amounts of data and devices, while enabling automated instantiation and communication of the corresponding services [12]. On the other hand, as time-sensitive and location-aware applications emerge (such as patient monitoring, real-time manufacturing, self-driving cars, drones, augmented reality (AR), or virtual reality (VR)), the cloud will not be able to satisfy the ultra-low latency requirements of these applications, deliver location-aware services, or scale to the magnitude of the data that these applications produce, as cloud services are not able to directly access local contextual information, such as precise user location, local network conditions, or even information about users' mobility behavior [13]. Therefore, the CC paradigm falls short to address all emerging scenarios due to its inherent limitations [14]. The benefits of the powerful computing resources that are available at a cloud data center are counterbalanced by the increased network delay for sending data over the public Internet. This is exactly how the term Edge Computing (EC) was coined [15]. Coinciding with some similar concepts such as "fog computing" [16] or "mist computing" [17], EC pushes the computing from the centralized cloud towards the data source (e.g., mobile devices). Hence, lately,

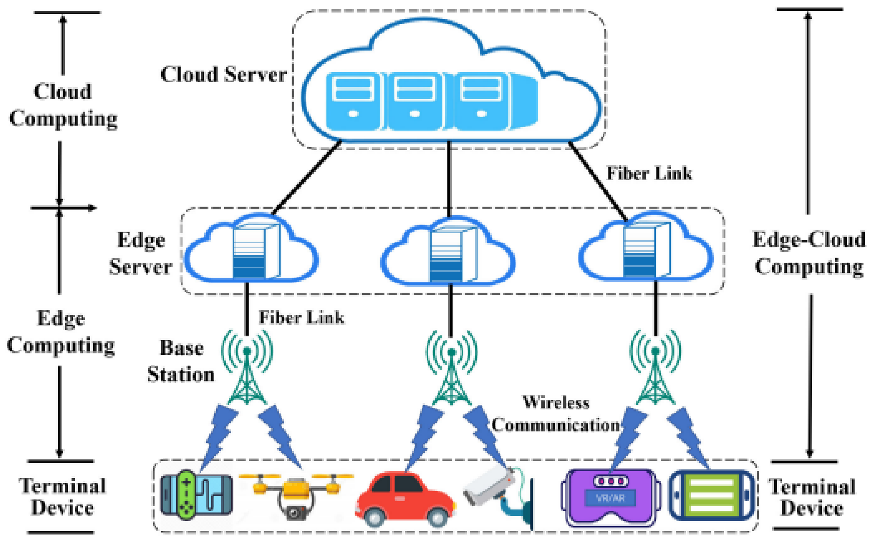


Figure 1.2: Edge-Cloud computing architecture.

the computing paradigm that has drawn a lot of attention from researchers, and the Industry is the concept of Multi-Access Edge Computing (MEC) [18]. The core idea is to introduce a more distributed infrastructure that will enhance performance and assist applications by bringing Cloud-like capabilities closer to the end devices, at the edge of the network [19]. This new infrastructural layer between the end devices and the Cloud is able to reduce the increased communication delay of the network, while keeping the computation power of servers to support applications with real-time requirements. Therefore, end devices are now capable of offloading their resource-intensive tasks to a proximate server of the Edge infrastructure, thus minimizing the overall execution time.

Fig 1.2 shows the general architectural concept of EC. MEC deploys edge servers between the device layer and the Cloud layer for providing distributed networking, computing, and storage resources. Oftentimes, these edge servers are placed near a collection of selected base stations, that is, every edge server should be co-located with a specific base station. Moreover, the MEC and MCC servers are commonly interconnected with high-efficiency/high-throughput fiber connections [20]. In such an environment, a device can offload complicated tasks to an edge server via a given base station within its proximity. The key difference between Cloud and Edge data centers is that the latter has a finite amount of computing resources. As a result, edge servers may suffer from overloaded workloads as the number of application requests is explosively growing. Therefore fine-grained resource management solutions are required to meet the strict constraints of the deployed time- and mission-critical applications. Another crucial point is the automatic orchestration of virtualized applications both in terms of computing and network resources [21]. To this extent, Network Function Virtualization (NFV) [22] and Software-Defined Networking (SDN) [23] are the key enablers of the prominent 5G infrastructure facilitating the decoupling of software deployment from hardware. By enabling 5G network slicing [24], NFV and SDN automatically orchestrate, instantiate, and manage the computing and network resources of virtual networks over a single physical infrastructure.

The synergetic combination of recent technological improvements redefines the way people live in various societal domains. In the smart computing context, sensor networks, Edge Computing, IoT, and big data analytics are properly orchestrated to provide assisting applications for human daily activities such as education, health, and transportation. Moreover, there are numerous indicative scenarios of the new breeds of latency-sensitive applications that require the joint consideration of IoT and MEC technologies. Indicative ones include smart homes and smart cities, healthcare, autonomous vehicles, augmented reality and virtual reality applications, retail, wearable IoT, IoT in mechanized agriculture, and the Industrial Internet of Things (IIoT).

The MEC paradigm is currently more relevant than ever, especially in the context of (i) the much-anticipated Industry 4.0 revolution [25] and (ii) the realization of Smart Things [26]. For the first case, IIoT introduces the novel concepts of Edge/Fog [27] or Cloud Robotics [28]. Both concepts can be defined as the architecture that distributes computing, storage, and networking functions at the Edge/Cloud continuum in a federated manner[29],[30], i.e., where robots and automation systems rely on data or code from a proximate infrastructure to support their operation. For the second case, we provide two indicative use cases. Smart Cities [31], [32] envision a world where sensors, devices, and citizens will seamlessly interact to assist in human activities [33]. As smart applications are mainly based on portable devices, the location awareness of people and devices is one key ingredient for enabling these services. Moreover, Smart Vehicles [34] allow the development of vehicular applications, e.g., planning, navigation, and compression of data, which will provide safe and comfortable services for drivers. Hence, the evolution of Location Based Services (LBS) [35],[36] and their integration with MEC architecture is expected to play a significant role in both scenarios of smart things [37].

Moreover, the convergence of Artificial Intelligence (AI) and IoT can redefine the way industries, businesses, and economies function. AI-enabled IoT creates intelligent machines that simulate smart behavior and supports decision-making with little or no human interference. Naturally, Edge Intelligence is a new term that defines the confluence of AI and Edge Computing [38]. Bringing intelligence to devices (i.e., IIoT, Smart Things) is crucial in the emergence of new applications.

These novel technologies facilitate the formulation of Cyber-Physical Systems [39] (CPS) that require automated decisions in the sense-compute-actuate cycle. CPS are systems where software and hardware components are seamlessly integrated towards performing well-defined tasks. CPS are characterized by seamlessly interweaving the physical world of infrastructure objects and the virtual world of information processing [40]. As a research field, CPS are about the intersection, not the union, of the physical and the virtualized worlds. Separately designing, analyzing, modeling, and understanding the physical and computational components and then connecting them together is not enough. To properly understand and design the optimal behavior of a CPS it is crucial to enable the integration of different components including computation, networking, and physical processes. Moreover, CPS must execute complex algorithms that usually are computing-intensive and can not be executed in resource-constrained devices.

To this end, the paradigm of approximate computing is introduced and widely used in CPS in recent years [41]. Approximate computing follows the basic principle of reducing the number of computations that an application must perform, thus reducing the applications' execution time, however producing a potential but acceptable loss in accuracy. Approximate computing is applied at different levels, including hardware acceleration (e.g., CPUs, GPUs, FPGAs, TDUs, etc.), software, algorithms, and even programming languages.

1.1 Challenges and Motivation

As discussed previously, bringing the decentralized computing infrastructure to the edge of the network brings various advantages (e.g., low latency communication, scalability), but it also brings new challenges. Moreover, the optimal support of CPS depends heavily on the provisioning of the right piece of data to the right computing entity in a timely manner [24],[42]. This introduces new research and design challenges concerning data processing, offloading decisions, architecture aspects, resource allocation, and controllers' design [43]. Apparently, there is a need to investigate the key requirements and potential opportunities for enabling the vision of edge computing to support CPS. Thus, it can be concluded that in the emerging edge computing paradigm, several problems arise, and innovative research is needed to address them. The significant advances in recent technology trends, and interest in state-of-the-art research from both Academia and Industry, motivate researchers for approaching and trying to solve interesting and challenging problems. In this thesis, we wanted to take this opportunity and propose solutions to some modern and interesting problems by taking into account the following subjects:

- **Device-Specific Offloading Aspects:** This is one key challenge in realizing the optimal support of CPS in the edge computing ecosystem. To begin with, IoT networks are dominantly characterized as rapidly varying access networks. Therefore, the dynamic network conditions are a very challenging problem in the context of task offloading [44]. Estimating the behavior of the network and predicting all of the phenomena (e.g., noise, interference, fading, and signal deterioration) that heavily impact the overall throughput and communication delay adds an extra level of complexity. When new tasks are generated by a device, a decision mechanism is required for the task partitioning problem namely; to decide whether the task should be executed locally or be offloaded to a remote infrastructure. An incorrect offloading decision may result in performance bottlenecks in the application. Optimizing the overall performance, offloading is beneficial only if reliable and low-latency wireless connectivity is available. Apart from the network, the success of the computation offloading, and consequently the performance of IoT-enabled applications, depends on many contextual parameters, e.g., the user's mobility, and residual energy of the device. This partition decision of the tasks is also associated with the task execution delay, the transmission delay, and the energy consumption of the device. As a result, deciding whether to offload an application request or not is the first challenge to address as it defines the core of the offloading problem and the accompanied resource allocation on the edge.

- **Optimizing Resource Management:** This constitutes the main motivation behind this thesis, as resource management is the core challenge that has to be tackled in the edge computing paradigm, where, naturally, resources are not abundant. In the decentralized edge computing paradigm one must appropriately map the applications' workloads to suitable host nodes based on their resource requirements. Also, as said before the available resources on the edge servers are limited compared to the cloud. Thus, resource management arises as an important concern in the emerging computing paradigms [16].

  In the age of virtualization techniques, it is typical for IoT workloads to run on Virtual Machines (VMs) [45] or Containers [46] on the edge servers. Two main decisions arise by the infrastructure providers that directly impact the response time of the applications, namely; (i) the computing resources to assign to the virtualized resources and (ii) the number of instances (replicas) for each virtualized entity. Furthermore, as multiple applications [47] and/or devices may seek additional computing resources in the same edge datacenter, it is of high importance the efficient and dynamic assignment of the resources of the hosted applications. It is a major challenge to meet viral QoS requirements primarily due to a set of competing Key Performance Indicators (KPIs). For instance, a resource management strategy must guarantee the performance requirements of the applications, while at the same time minimizing the cost of the deployed resources [48]. Moreover, it is common for modern applications to be computationally intensive and require a significant amount of resources for task execution. This imposes the design of dynamic resource allocation strategies for the underlying edge and cloud infrastructure, which in turn facilitates the deployment of industrial applications as network services that can be reconfigured on demand. Hence, orchestrator providers usually realize proactive and reactive solutions to address the dynamic nature of the workload and the adjustment of the provisioned resources, while monitoring various KPIs.

  On the other hand, efficiently utilizing the available resources is an important challenge. It is crucial to optimize resource utilization to gain better performance for all the deployed applications [49]. The optimization of resource utilization is a multi-objective task and very exciting to tackle as one must consider the diverse requirements of applications, the fluctuating workload demands, and the finite resources of the infrastructure. Throughout the literature, researchers utilize queuing theory to model mobile devices and edge servers, along with an optimization technique for deciding the optimal offloading policy [39]. Unfortunately, in some cases, there is a major drawback that can lead to overall performance deterioration: the static modeling of the edge servers' resources which leads to over-provisioning or under-provisioning of resources or to under-utilization and over-utilization respectively [50]. Finally, following the resource provisioning process, a scheduling algorithm (or load balancing) is adopted for processing and optimally mapping the incoming tasks/applications (workload) to the virtualized resources. For instance, if a VM becomes overutilized then it should be temporarily excluded from the pool of resources dedicated to executing the applications' tasks. resources are temporarily excluded from any future allocation. However, task scheduling also includes diverse challenges such as heterogeneity of requests and resources, and uncertainty of the number of incoming requests, which cannot be resolved by using traditional resource management approaches [51]. Hence, a large focus, and priority should be drawn to these to make services more reliable. In other words, serving the workload should take up the minimal amount of resources that will be well utilized to maintain a desired QoS level.

- **Performance Modeling of Dynamic Workloads and Applications:** As said before dealing with the dynamic nature of the applications' workloads is key to achieving high system performance [52]. In the literature, most of the existing models for capturing the requirements in resources of IoT applications are empirical and most commonly highlight a specific performance metric such as response time, and energy consumption [53],[51]. Dealing with the dynamic workload, the various requirements of different IoT applications, and the unmodelled behavior of resources' state, are not trivial tasks to handle. Hence, relying on these simple may lead to either performance degradation or not optimal utilization of resources. As a result, one must design multi-input multi-output performance models easily applicable to a large subset of applications to capture such dynamics. In such a way novel controllers can be realized for regulating the aforementioned KPIs. To achieve the highly-anticipated workload characterization and performance prediction the first step is to try to predict the number of incoming requests in the edge infrastructure [54]. This prediction can be combined with a resource allocation mechanism since the amount of resources required for the task execution is directly proportional to the amount of traffic (i.e., the request rate). To this extent, a key enabler of scaling resources according to the workload traffic is identifying the maximum number of requests that the provisioned resources can process.

- **Designing Practical Approximate Computation Offloading Strategies:** In recent years complex applications are emerging, targeting among others to bring Edge Intelligence to IoT devices, mobile agents, and robotics [55],[56]. Especially, Industry 4.0 applications especially rely on mobile robotic agents that execute many complex tasks that have strict safety and time requirements, e.g., algorithms related to image processing, planning, localization, mapping, and autonomous learning. These algorithms are computing- and memory-intensive and the resources of robots and IoT devices are not sufficient to build efficient, safe, and autonomous robot operations [57]. Consequently, new problems arise and one must carefully decide deployment strategies - where to place a workload, connection policies

- when to use the edge nodes, heterogeneity - how to deal with different types of nodes, the accuracy of complex algorithms - how much to save response time while the loss in accuracy of a complex algorithm is acceptable. The execution of compute-intensive components of IoT applications in edge computing infrastructures involves complex application partitioning and application quality selection at different granularity levels and component migration to the edge server node. Furthermore, suitable as it may seem, solely utilizing remote computational resources might not be enough; a number of unwanted phenomena potentially take place in the transmission and processing of the information, such as network latency, variable QoS, and downtime. For these reasons, for example, autonomous mobile agents (e.g., robots, unmanned vehicles) often have some capacity for local processing when targeting low-latency responses and during periods where network access is unavailable or unreliable. Consequently, a major challenge, from a control design, estimation, and network optimization point of view is to combine local fast but possibly accuracy-limited algorithms and remote, precise but computationally intensive ones in an efficient way.

- **Co-Design Closed Loop Controllers in the context of 3C:** Finally, having said all that, a key challenge in designing controllers for the optimal behavior of CPS is to provide some guarantees on closed-loop performance [43]. Especially, in modern manufacturing, the current trend is to remove IIoT (e.g., mobile robots, actuators, etc.) from confined spaces and allow them to roam freely around the production floor, improving collaboration and collaboration with humans, thus increasing productivity and efficiency. Open challenges in this area include the development of adaptive multi-robot/machine controllers, the detection, modeling, prediction, and forecasting of human-robot interactions, and the design of distributed control and planning algorithms that provide flexible and safe working environments. Because most IIoT applications such as warehouse robotics, telesurgery, industrial automation, and real-time data processing, are mission-critical and require real-time communication and processing to function effectively in a closed loop, the IIoT requires the joint design of control, computing, and communication (3C). However, most of the existing IIoT research focuses on just one aspect of the 3Cs. Such an independent design approach, ignoring the close interaction between the 3Cs, has poor overall and system performance, and consumes significant wireless and computational resources. To achieve safe reliable and high-performance control with low latency in industrial environments the emerging approach of co-design is mandatory. Towards successful closed-loop operations, the co-design of Control, Communication, and Computing (3C) is required to provide reliable, low-latency, and high-performance control. Thus, an approach is to design a local controller that takes feedback from the remote controller. Then, one must derive the conditions under which such a hybrid scheme might offer stability assurances and be beneficial. To this end, the controller synthesis should provide formal closed-loop guarantees and stabilize the overall performance leading to safe and optimal behaviors.

1.2 Contributions

This thesis tries to tackle some of the aforementioned problems that arise in edge computing and computational offloading of CPS. Our main focus lay on optimizing the performance of CPS optimally utilizing offloading policies and carefully designing resource management strategies. The contributions to the above topics can be summarized in the following:

1. **Modeling the Performance Dynamics:** New-generation resource management solutions require sophisticated performance modeling abstractions to capture the various dynamics of the application requirements, resources' state, and users' demands. A major novelty of this thesis is the introduction of resource profiles (or flavors) to achieve all of the above. Resource profiles express the relationship between the allocated resources and the maximum request of incoming requests of a specific application that the resources can serve while keeping a certain QoS level. The advantage of the resource profiles design is three-fold. First, a resource profile is actually a feasible operating point and facilitates the dynamic resource allocation process by giving knowledge regarding the number of instances required for the virtualized resources. Secondly, the combination of these resource profiles with controllers can enable fine-grained scaling approaches for time- or mission-critical applications. Third, we optimize the performance of the underlying resources solving the aspect of under/over-provisioning while paving the way for tackling resource heterogeneity, as will be shown in the following sections. In this thesis, we experimented with two different modeling methodologies to extract the resource profiles. On the one hand, we used System Theory, which has the capacity to include several performance metrics (i.e., state variables) and resources as control parameters (input variables) and describe their relationship under various operating conditions and QoS guarantees. On the other hand, we used simple ML algorithms that leverage information from (i) the scheduling and resource allocation solution, (ii) the monitored KPIs, and (iii) the workload prediction algorithm to estimate the essential number of replicas for meeting the workload demand. The key enabler to both approaches is modeling the workload, i.e., estimating the number of incoming requests for the application leveraging time-series forecasting methods. The proposed designs aim to be easily re-configurable solutions that tackle both the proactive and reactive aspects of the resource management problem.

2. **Joint Resource Management and Task Scheduling in EC:** Maximizing the performance of modern applications requires timely resource management of the virtualized resources in EC. However, proactively deploying resources for meeting specific application requirements subject to a dynamic workload profile of incoming requests is extremely challenging. To this end, another major novelty of this thesis is that the fundamental problems of task scheduling and resource autoscaling are jointly addressed. Consequently, together with the offloading decision, a dynamic resource allocation and admission control mechanism is proposed in this thesis, which is called Task Scheduling and Scaling Mechanism. This mechanism is responsible for the distribution of offloaded requests among the application

instances, alongside the admission control (i.e., accepting or rejecting the offloaded requests according to resource availability) and dynamic resource allocation for each application (i.e., deciding the number of replicas for each resource profile). Again, a workload profile estimator and the dynamic modeling of the resources and overall status of the network/servers provide the foundation upon which the resource allocation algorithm works. Specifically, the Task Scheduling and Scaling Mechanism is introduced for the resource management of the constrained resources of an edge datacenter, utilizing the aforementioned resource profiles to dynamically adjust the number of deployed resources and assign the incoming requests of an application to them while keeping certain QoS levels. The proposed algorithm is designed to be practicable, and guarantee feasibility and performance specifications, such as adaptability to rapid changes in the workload and resource availability. This preliminary mechanism is then enriched by exploiting stability guarantees derived from queuing theory to provide a decentralized mechanism for the task scheduling and resource allocation problem, trying to build a holistic scalable architecture. The proposed event-triggered scheme facilitates the proactive scaling of resources subject to a dynamic workload profile via a novel, intuitively conceived locally identifiable triggering condition. This research addresses the so far untouched challenge, of designing controllers that address a mixture of these unwanted phenomena by changing in an event-based nature the provisioning of the resources to the control algorithm, taking also into consideration various KPIs in both proactive and reactive manner.

3. **A Switching Offloading Mechanism in the context of 3C:** In all the settings of the next chapters, we assume that devices seek additional computing resources at an edge server. As a result, in this thesis, we design optimal offloading policies for CPS. Initially, the focus is placed on pure wireless communication network quality for mobile users. A two-step mechanism to support and realize the offloading decision is devised, considering both users' contextual information and the availability of the edge resources. More specifically, mobile users offload their LBS requests taking into consideration their position and the signal strength of the connection with a wireless access point to minimize the communication overhead. Also, the edge infrastructure may reject a request (i.e., the user has to execute it locally) if the available resources are not sufficient to keep a certain QoS level. Following in the next chapters of the thesis, a similar switching offloading mechanism is realized to assist in optimizing the offloading of computing-intensive algorithms in IIoT, i.e., a mobile robot. To further alleviate the computational strain from the resource-constrained robot, we implement two different versions of these algorithms following the approximate computing paradigm namely; (i) a local (on the device), fast must not sufficiently accurate, and (ii) a remote (on the edge server), precise but expensive in terms of resources and/or time. Hence, by adopting switching offloading policies which consider the uncertainty of the local's algorithms we are able to build a framework that assists a mobile robot in a simple navigation mission, i.e., reaching a waypoint from a starting position. Remotely executing these computing-intensive tasks is

only performed in case offloading is beneficial i.e., when the communication overhead is low and the available resources of the edge server are adequate to perform the tasks. Moreover, embracing the concept of 3C, the switching offloading mechanisms also consider input from a control perspective namely; the need for a more accurate estimation of the mobile robot, the proximity to obstacles, and the difficulty of the navigation, to guarantee safe navigation. Consequently, this mechanism facilitates optimal offloading for CPS.

4. **A Resource-Aware Estimation and Control Framework for Edge Robotics:** Respecting the need for stability guarantees and closed-loop controllers for CPS, we extend the modeling for the navigation of a mobile robot. A novel resource-aware estimation and control framework for edge robotics is realized, which jointly tackles the problem of convergence in the trajectory navigation of a unicycle robot, and the problem of efficiently using communication and computing resources is presented. Utilizing a utility function that incorporates the dynamics from respective control, communication, and available computing resources, we build a framework to handle two fundamental problems: a) the synthesis of controllers that dictate the motion of the unicycle robot in this path planning problem and b) an offloading strategy to compensate the uncertainty of the local estimation techniques with the more accurate remote ones while finding the balance between navigation accuracy and mission duration. More importantly, the proposed framework guarantees convergence to the target position independent of the various parameters chosen, in contrast to the periodic offloading schemes. The whole approach is easily adjustable to the needs of different mission characteristics.

5. **Evaluation of proposed frameworks with Relevant to Architectures and Standards:** Extensive numerical results are presented, which are obtained via real experimentation or simulations in order to capture the effectiveness and efficiency of the proposed frameworks. Especially, for every real experimentation that will be presented in this thesis, we chose to follow the wide-adopted solutions, namely the European Telecommunications Standards Institute (ETSI) MEC standards [58] and implement them with commercial or open-source resource orchestration tools, that enable scalability, interoperability [59], and transparent development of the applications [60] over heterogeneous hardware and software technologies [61], i.e., NFV orchestration and widely used software tools (*e.g.,* OpenStack [62], Kubernetes [63]).

In the following, Chapter 2 introduces some preliminary notions and concepts pertaining to the adopted scientific methodologies by this thesis. Then, each chapter focuses on one of the aforementioned settings and use case scenarios, presenting the related work in this respective field and introducing the developed solutions together with extended numerical evaluations of their effectiveness and efficiency.

Chapter 2

Background

In this chapter, the essential mathematical background necessary to understand the methods that are used in the following work will be presented. Any additional information required will be provided in the main part of the thesis, whenever it is needed.

2.1 Basic Definitions of Modeling and Control Theory

2.1.1 Linear Time Invariant State Space Models

A state space model of a system is the mathematical description of the relationship between *the cause and the effect* or *the inputs and the outputs* of the system [64]. In this thesis, discrete time state space models are solely utilised. The general form of a discrete time state space model is

$$x(k+1) = f(x(k), u(k)), f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n, \tag{2.1}$$

where $x(k) \in \mathbb{R}^n, u(k) \in \mathbb{R}^m$ are the state and the input vector respectively and $k \in \mathbb{N}$. The most widely used state space models are the linear time invariant (LTI) state space models, where the function $f(x(k), u(k))$ of (2.1) is linearly dependent on $x(k)$ and $u(k)$,

$$x(k+1) = Ax(k) + Bu(k), \tag{2.2}$$

$$y(k) = Cx(k) + Du(k). \tag{2.3}$$

Here, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are constant, time invariant matrices that describe the system's dynamics and $y(k) \in \mathbb{R}$ is the system's output vector.

2.1.2 Stability

In modern control theory, the notion of stability is strongly connected with the dynamic behavior of a system. Many different kinds of stability are defined, i.e., input-output stability or stability of an equilibrium point. In general, a stable system means that the state variables of the system are driven to a specific equilibrium point or inside a desired area and they remain there despite any insisting or momentary disturbances. In this chapter, the focus is put on the stability of an equilibrium point (x_{eq}, u_{eq}). An equilibrium point[1] satisfies the equality

$$x_{eq} = Ax_{eq} + Bu_{eq}. \tag{2.4}$$

The most general type of equilibrium point stability is Lyapunov stability, which guarantees that the system trajectories will remain close to x_{eq}, if they start from a neighborhood in the equilibrium point's vicinity [65].

Furthermore, in the following sections, asymptotic stability is adopted, together with the set-theoretic notions of stability analysis and control design problem, which identify and characterize subsets of the state space, containing the desired equilibrium state with special properties: positively invariant sets are introduced here. In the forthcoming paragraphs, without loss of generality, all the necessary definitions are given assuming that $x_{eq} = 0$.

Definition 1. *A sphere B_s with radius $s > 0$ and the origin as its center, is denoted as*

$$B_s = \{x \in \mathbb{R}^n : \|x\| \leq s\}, \tag{2.5}$$

where $\|\cdot\|$ is any possible norm of vector x.

Definition 2. *Assuming a discrete-time system of the following form*

$$x(k+1) = f(x(k)), \tag{2.6}$$

then the equilibrium point $x_{eq} = 0$ is locally Lyapunov stable, if and only if $\forall \epsilon > 0, \exists \delta = \delta(\epsilon) > 0$. Then,

$$x_0 \in B_{\delta(\epsilon)} \Rightarrow x(t; x_0) \in B_\epsilon, \forall t \geq 0. \tag{2.7}$$

Definition 3. *The zero equilibrium point of* (2.6) *is contractive in a region $D \subset \mathbb{R}^n$ if $\forall x_0 \in D$ and*

$$\lim_{t \to \infty} x(t; x_0) = 0. \tag{2.8}$$

The region D is called Domain of Attraction (DoA) of the equilibrium point.

[1]It should be noted here that, for the rest of this thesis, the terms "equilibrium point", "operating point" and "resource profile" will be used interchangeably.

Definition 4. *The zero equilibrium point of* (2.6) *is asymptotically stable if and only if it is Lyapunov stable and contractive,*

$$\lim_{t\to\infty} x(t;x_0) = 0, x_0 \in B_{\delta(\epsilon)} \subseteq D, \tag{2.9}$$

where D is DoA.

In the following definitions, the essential results of the Lyapunov theory (second or direct Lyapunov method), which connect the stability property with a specific type of functions, are presented.

Definition 5. *Assuming a continuous function $V(x), V : D \to \mathbb{R}$, where D contains the origin. Then V(x) is positive (semi)definite in D if*

$$V(x) > (\geq) 0, \forall x \in D\backslash\{0\}, \tag{2.10}$$

$$V(0) = 0. \tag{2.11}$$

Definition 6. *Function $V(x)$ is negative (semi)definite if $-V(x)$ is positive (semi)definite.*

Definition 7. *For discrete time systems* (2.6)*, the total difference of function $V(x), V : \mathbb{R}^n \to \mathbb{R}$, respectively to system* (2.6) *is*

$$\Delta V(x)_{(2.6)} = V(f(x)) - V(x). \tag{2.12}$$

Then, the Lyapunov theorem for discrete time system is formulated as,

Theorem 1. *([64, 65]) Assuming a positive definite function $V(x), V : D \to \mathbb{R}$, then*

- *If the total difference $\Delta V(x)_{(2.6)}$ of* (2.12) *is negative semidefinite $\forall x \in D$, the system is locally Lyapunov stable.*
- *If the total difference $\Delta V(x)_{(2.6)}$ of* (2.12) *is negative definite $\forall x \in D$, the system is locally asymptotically stable.*

The function $V(x)$, which satisfies the above theorem, is called Lyapunov Function (LF). From the previous analysis, the stability problem is equal to finding a positive definite function that is non-increasing or decreasing along the trajectories of the system (2.6). Finding an LF, allows for defining sets with special properties respective to the equilibrium point. For example, if there exists an LF $V(x)$ that guarantees the stability or asymptotic stability in a region D and the sets $R(V;\gamma) = \{x \in \mathcal{R}^n : V(x) \leq \gamma\} \subseteq D$ are close and contain the zero equilibrium point, then these sets consist an estimation of the DoA. The essential definitions follow.

The most important benefits of the Lyapunov theory are the characterization of the stability of the equilibrium point and the possibility of defining sets with interesting properties. For example, if an LF can be found, which ensures the asymptotic stability of the equilibrium point of a constrained system $x(t) \in S_x \subset \mathbb{R}^n$, then any trajectory that begins from a point inside $R(V;\gamma) \subseteq S_x$ will be driven to the equilibrium point, without violating the system constraints.

2.2 Basic Definitions of Queuing Theory

We define a request as an individual demand for computing resources provided by a computing node. A computing node is defined as the physical (or virtual) computing environment, consisting of hardware, software, and network resources, where a request is executed. A queue is defined as the waiting mechanism where a request arrives at a node and is temporarily put on hold until it is selected for service from among other requests that are waiting. Here, we consider queues consistent with the First Come First Served (FCFS) selection policy.

A queuing system [66], [67] is defined as the dynamic relationship that is developed between a stream of request arrivals and a flow of request departures from a computing device, respectively, in the presence of a queue. From a mathematical perspective, a queue acts as an integrator of the difference between arrival and departure rates. A simple queuing system is consistent with the following notation. Denoting by $X(t)$ and $Y(t)$ the arrivals and departures from a queue, respectively, in the interval $[0,t]$, the number of queued requests at time $t, Q(t)$, is defined as the difference between arrivals and departures in intervals $[0,t]$. Note that exact knowledge of $X(t)$ and $Y(t)$ is difficult in real applications.

Furthermore, in the following chapters, we consider a set of n computing nodes represented by a multi-queue system, where each node is modeled by a queue combined with a computing node. We assume that a workload λ from an application arrives at the system and the central batch queue. The workload then must be scheduled or dispatched to n nodes according to an admission control policy $u_i(t), i = 1,\ldots,n$, with each node representing a virtualized computing unit. We denote the number of queued requests that have not yet been admitted at time t by $\delta(t)$, and the number of admitted requests waiting to be selected for service by the i^{th} node at time t by $w_i(t)$. The service rate (processing) of the i^{th} node is denoted by $\gamma_i, i = 1,\ldots,n$. The described system is depicted in Fig. 2.1.

2.3 Basic Definitions of Set-Theoretic Estimation

In chapter 6, we investigate the problem of Set-Theoretic Estimation for mobile robots. We first need to find the Reachable Sets. Consider a dynamic system of the form

$$x(t) = f(x(t), u(t)) \tag{2.13}$$

or of the form

$$x(t+1) = f(x(t), u(t)) \tag{2.14}$$

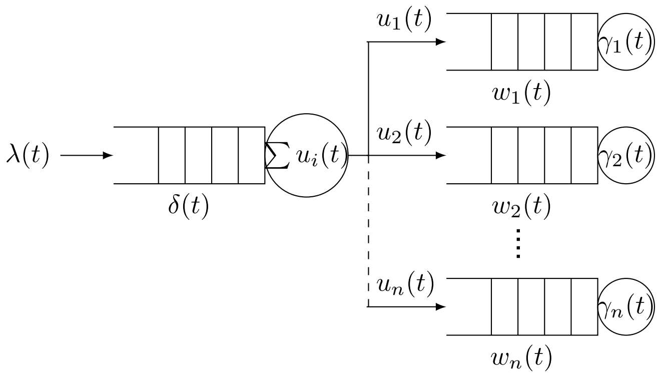


Figure 2.1: A multi-queue system.

where $u(t) \in U$. The following classical definition of reachability set is reported.

Definition 8. *Reachability set. Given the set P, the reachability set $R_T(P)$ from P in time $T < +$ is the set of all vectors x for which there exists $x(0) \in P$ and $u(\mathring{u}) \in U$ such that $x(T) = x$.*

Reachable sets are useful to describe the effects of a bounded disturbance on a dynamical system or to describe the range of effectiveness of a bounded control. Unfortunately, the computation of reachable sets is, in general, very hard even in the discrete-time case.

Now, let us formulate the problem of Set Estimation in a general framework. Here we mainly consider discrete-time systems, although the concepts we present apply to continuous-time as well.

$$x(t+1) = f(x(t), u(t), d(t)) \tag{2.15}$$

$$y(t) = g(x(t), w(t) \tag{2.16}$$

where we assume $d(t) \in D$ and $w(t) \in W$, with D and W are convex and compact subset of $\mathcal{R}^n$. The control $u(t) \in U$ is always assumed to be known in the present context, for example, determined by a respective control algorithm.

Together with the information given by equations (2.15)-(2.16) we assume to have a priori information regarding the initial state:

$$x(0) \in \hat{X}_0. \tag{2.17}$$

Now we introduce two operators from set to set. Given a set $X \subset \mathcal{R}^n$, define the set of all reachable states in one step for all possible $d(t) \in D$ given the control action $u(t) \in U$, according to equation (2.15).

$$Reach[X, D](u) = z = f(x, u, d), x \in X, d \in D. \tag{2.18}$$

Given a guess about the state in t, the previous equation propagates this information at $t + 1$. Clearly, in this way, the information about the actual state spreads. However, by means of (2.16), we can exclude a portion of the new guess region that is not compatible with measurements. Let us introduce the set of all the states compatible with measurements as:

$$Comp[W](y) = x : g(x, w) = y, \text{for some } w \in W. \tag{2.19}$$

Besides, using (2.18),(2.19) we can describe, the estimation set. Let us formally define the concept.

Definition 9. *Estimation region: The set $\hat{X}(t)$ is an estimation region (set) at time t given the information* (2.15),(2.16) *and* (2.17) *over a prescribed horizon* $0, 1, \ldots, t$, *if the condition* $x(t) \in \hat{X}(t)$ *is assured for all* $w \in W$ *and* $d \in D$.

Now we present a common procedure to compute a Non-conservative state-membership estimation. For the dynamic system described by (2.18),(2.19), given the initial estimation (2.19) let $\hat{X}(k|k)$ be the set of estimated states at step k that are output compatible:

$$\hat{X}(k|k) = \hat{X}(k) \cap Comp[W](y(k))$$

. Then set $t = k$ and perform the following steps.

1. Given the current value of the control $u(t)$, propagate this set forward

$$\hat{Z}(t+1|k) := Reach[\hat{X}(t|k), D](u(t)).$$

2. Compute the new estimation set as the intersection with the compatible set

$$\hat{X}(t+1|k) = Z\hat{Z}(t+1|k) \cap Comp[W](y(t+1)).$$

3. Set $t = t + 1$ and go to Step 1.

The difficulty of iterating the method is apparent as we may encounter non-convex sets and computations.

2.4 Semidefinite Relaxation

In this section, we will introduce the semidefinite relaxation (SDR) technique. In particular, this technique can be applied to many nonconvex quadratically constrained quadratic programs (QCQPs). Let us write the problem of interest, namely, the real-valued homogeneous QCQP: The QCQP Problem is defined as follows:

$$\min_{\mathbf{x} \in \mathcal{R}^n} \mathbf{x^T C x} \tag{2.20a}$$

$$\text{s.t.} \quad \mathbf{x^T A_i x} \triangleq_i b_i \quad i = 1, \ldots, m \tag{2.20b}$$

where $\trianglelefteq_i$, represent either " $\leq$ "," $\geq$ " or " $=$ " and the matrices $\mathbf{C}, \mathbf{A_1}, \ldots, \mathbf{A_m} \in \mathcal{S}^n$ are all positive semidefinite, where $\mathcal{S}^n$ denotes the set of all real symmetric $n \times n$ matrices and $b_i \in \mathcal{R}$.

A crucial first step in deriving an SDR of 2.20 is to observe that

$$\mathbf{x^T C x} = \mathrm{Tr}(\mathbf{x^T C x}) = \mathrm{Tr}(\mathbf{C x x^T}) \tag{2.21}$$

$$\mathbf{x^T A_i x} = \mathrm{Tr}(\mathbf{x^T A_i x}) = \mathrm{Tr}(\mathbf{A_i x x^T}) \tag{2.22}$$

We define a new variable $\mathbf{X} = \mathbf{x x^T}$, where X is symmetric and noting that $\mathbf{X} = \mathbf{x x^T}$ is equivalent to $\mathbf{X}$ being a rank one PSD matrix, we obtain the equivalent to (2.20) formulation:

$$\min_{\mathbf{X} \in \mathcal{S}^n} \ \mathrm{Tr}(\mathbf{CX}) \tag{2.23a}$$

$$\text{s.t.} \quad \mathrm{Tr}(\mathbf{A_i X}) \trianglelefteq_i b_i \quad i = 1, \ldots, m. \tag{2.23b}$$

$$\mathbf{X} \geq 0 \tag{2.23c}$$

$$rank(\mathbf{X}) = 1. \tag{2.23d}$$

The Tr() operator is the sum of elements of the main diagonal. The Rank operator refers to the number of linearly independent rows or columns of a matrix.

Then, if we drop the $rank(\mathbf{X}) = 1$ constraint then we get the relaxed problem:

$$\min_{\mathbf{X} \in \mathcal{S}^n} \ \mathrm{Tr}(\mathbf{CX}) \tag{2.24a}$$

$$\text{s.t.} \quad \mathrm{Tr}(\mathbf{A_i X}) \trianglelefteq_i b_i \quad i = 1, \ldots, m. \tag{2.24b}$$

$$\mathbf{X} \geq 0. \tag{2.24c}$$

Using convex optimization (e.g., Matlab CVX) we can obtain a solution.

Let $\mathbf{X}^\star$ be the optimal solution to (2.24) then if $\mathbf{X}^\star$ is of rank one then we have the optimal solution for (2.20). Otherwise, we need a vector $\tilde{\mathbf{x}}$ that is feasible for (2.20).

Let $\mathbf{X} \in \mathcal{S}^n$ be an arbitrary symmetric positive semidefinite matrix. Consider a random vector $\xi \in \mathcal{R}^n$ drawn by the Gaussian distribution $\xi \sim \mathcal{N}(0, \mathbf{X})$. Then we want to optimize the expected value:

$$\min_{\mathbf{X} \in \mathcal{S}^n, \mathbf{X} \geq 0} E_{\xi \sim \mathcal{N}(0,\mathbf{X})} \{\xi^T \mathbf{C} \xi\} \tag{2.25a}$$

$$\text{s.t.} \quad E_{\xi \sim \mathcal{N}} \{\xi^T \mathbf{A_i} \xi\}, \quad i = 1, \ldots, m, \tag{2.25b}$$

this stochastic interpretation can generate approximate solutions to (2.24). Given a solution $\mathbf{X}$ we can generate $\xi \sim \mathcal{N}(0, \mathbf{X})$, λ solutions and construct a QCQP-feasible point with a mapping function, e.g., $sgn(\xi_\lambda)$. Then from the computed feasible solutions, we choose the best.

This thesis will also provide more explanations on modeling and control theory-related concepts whenever needed.

Chapter 3

A Scalable Edge Computing Architecture Enabling Smart Offloading for Location-based Services

3.1 General Setting

As mentioned in Chapter 1, the ever-increasing amount of data produced, along with the emergence of next-generation computationally intensive applications, requires new service delivery models. Such models should capitalize on the Edge Computing (EC) paradigm for supporting the data offloading process, by considering the user's contextual information in the offloading decision along with the infrastructure resource allocation operations, towards meeting the stringent performance specifications. Despite the plethora of research works that have adopted and exploited the MEC paradigm, there are still important challenges to be addressed, pertinent to the realization of Location-based Services for a Smart City environment.

In this chapter, a two-level Edge Computing architecture is proposed to offer computing resources for the remote execution of an LBS. At the Device layer, an initial offloading decision is performed taking into consideration the estimated position and quality of the wireless connection of each user. At the Edge layer, a resource profiling mechanism maps the incoming workload to EC computing resources under specific performance requirements of the LBS. Dealing with the dynamic workload, a scaling mechanism simultaneously takes the offloading decision and allocates only the necessary resources based on the resource profiles and the estimation of a workload prediction technique. For the evaluation of the proposed architecture, a smart touristic application scenario was realized on a real large-scale 5G testbed, following the principles of NFV orchestration.

3.2 Related Work

This section presents the most relative studies of the literature to our proposed EC solution for LBS. The following studies are classified based on three interdependent pillars; (i) Edge Computing, (ii) Computation Offloading, and (iii) Location Estimation.

(i) Edge Computing

Puliafito et al. [68] presented a detailed survey for Fog Computing platforms for IoT-based applications. In [69], a network slicing orchestration system was proposed for federated data centers. The orchestration is based on Open Source MANO [70], which is aligned with ETSI NFV standard. Based on G/G/m queuing model, a heuristic scaling algorithm regulated the mean overall processing time of the incoming slice orchestration requests. However, no further details are given on how the network slice placement problem is addressed. MESON [71] focuses on communication between slices deployed on edge data centers. The cross-slice communication aims to alleviate the congestion of the core network, minimize the utilization of cloud resources and meet the requirement of time-sensitive LBS.

Leivadeas et al. [72] proposed a meta-heuristic approach for the placement of network slices in cloud and edge infrastructure aiming at the minimization of the deployment cost and the end-to-end network delay. This approach considered that the resource demands of the slices are static and it can be applied only for their initial placement. Sonmez et al. [73] proposed a fuzzy workload orchestrator for various EC scenarios. For each offloaded request, a set of fuzzy rules decided the destination computational unit within an hierarchical EC architecture. However, the authors empirically defined the fuzzy rules sets, which might not be applicable for services with different workload characteristics. In [74], the authors addressed jointly the problems of network selection and service placement for MEC infrastructure. To reduce the complexity of the general problem, they decomposed it into a series of sub-problems and solved them in an iterative fashion. However, the proposed QoS model included only network parameters ignoring the processing delay of the service.

Zenith et al. [75] proposed resource sharing contracts between edge infrastructure providers and service providers. Specifically, an auction-based contract establishment and resource allocation mechanism was introduced, focusing on ensuring the utility-maximization for both entities and the latency constraints. However, the authors did not provide any details on how the tasks were distributed among the containers of the established contracts. Wang et.al. [76] studied the problem of VM placement along with the workload assignment for mobile cloud applications in MEC. A MILP method was designed to minimize hardware consumption in order to deploy VMs, while satisfying diverse latency requirements of different applications. In [77], the authors proposed a novel solution to determine both the optimal number of the VMs to spawn, as well as their placement. For VM spawning, a Mixed Integer Linear Program (MILP) was formulated, while a game theoretic approach was employed (i.e., Coalition Formation Games), in order to treat the latter problem.

(ii) Computation Offloading

The offloading decision is critical for guaranteeing the high QoS of LBS. In the literature, most of the proposed studies focus on a single or group of performance criteria, i.e., latency or energy, to decide whether the offloading is profitable or not. This paragraph illustrates the most recent relative studies to our work. The interested reader can refer to [78] for a comprehensive detailed review of older studies on computation offloading. In [79] a mobility-aware offloading mechanism (referred to as MAGA) was proposed, based on frequent moving patterns of the users and a genetic algorithm. MAGA mechanism focused on partial task offloading to EC servers. However, the dynamic resource allocation of these resources is not considered. Tang et al. [80] and Apostolopoulos et al. [81] proposed the adoption of Prospect Theory to design user behavior-aware MEC offloading frameworks. In both works, the communication and the computation models for the users were provided and Prospect Theory-based offloading games are formulated taking into account, user risks, as well as weighting and framing effects. In [82], the authors proposed a task offloading framework for Software Defined Ultra-Dense Networks that focused on the reduction of the task duration and the energy saving. The offloading problem was split in two sub-problems;(i) task placement problem and (ii) resource allocation. Lyu et al. [83] presented a collaborative Cloud-MEC-IoT architecture and proposed a request modelling scheme and an admission control framework to address the scalability problem of these platforms. Although the authors considered heterogeneous edge resources, the computation model was not dynamic.

(iii) Location Estimation

In the context of indoor localization, many approaches can be found in the literature that addresses different scenarios and accuracy requirements. In [84], the authors proposed a mechanism that takes advantage of the information by the WiFi signals and calculates the position of a moving user with the fingerprinting technique. The mean estimation error of this mechanism is about $0.85m$, showing the advantages of the weighted K-Nearest Neighbor algorithm. Kang et al. [85] proposed a method to track pedestrian movement in an indoor environment under the existence of sensing inaccuracy. Similar to our work, they used a smartphone equipped with accelerometer, magnetometer, and gyroscope sensors and exploited the characteristics of human motion to estimate the position of a user. The proposed method reduces the influence of the cumulative errors of the dead reckoning technique. In [86], exploiting Machine Learning techniques, a tracking system was proposed, estimating again the position of a user, who moves in an indoor place and interacts with a wireless sensor network. The classification of speed and heading of the trajectory of the pedestrian was implemented offline supervised by Neural Networks. A different approach to handle the challenges of indoor localization in a robotic scenario in the Industry 4.0 context was presented in [87]. The single vision-based method is landmark assisted, exploiting natural features of the landmarks and the core principles of projective geometry, for the self-localization of an indoor autonomous mobile robot agent. The results of the proposed algorithm showed a significant accuracy in close range. However, as most of the vision-based methodologies, this method is heavily dependent on ambient conditions (e.g., light conditions). In a similar vein, authors in [88], realized a novel unsupervised learning framework requiring video frames, to extract pose estimation and introduce a Simultaneous Localization and Mapping (SLAM) system, that could be also applicable in an indoor environment.

Most of the aforementioned studies usually proposed solutions addressing only one of the above research problems in isolation, ignoring the rest dominant parameters, thus failing to allow the design of a holistic solution that achieves both high performance of LBS and optimal utilization of EC resources. To exactly fill this gap, in this article we propose an EC architecture for LBS that considers the user's contextual information on the offloading decision and the resource allocation of the edge resources. Towards this direction, the proposed framework can meet the performance requirements of both LBS and EC infrastructure.

3.3 Contributions and Outline

In this Chapter, we develop a two-layer architecture to support smart offloading of a location based service from IoT or mobile devices to an EC infrastructure. The proposed architecture has been particularly designed to accelerate the execution of an object identification service for mobile users. However, it is generic and applicable to several types of EC environments and smart city applications. In the particular smart city scenario under consideration in this work, the visitors of a crowded touristic area use their camera-equipped devices to take snapshots or short-duration videos of a Point of Interest (PoI) (e.g., exhibits of a museum) in order to receive additional information about them. Since image recognition is a computational-intensive and energy-consuming task, a cluster of proximate edge servers offer the essential computing resources to meet the strict time requirements of massive users. Furthermore, a localization technique is realized and evaluated. We assume that users are active in a crowded touristic area; hence, the shared bandwidth and foremost the interference of signals plays an important role in the overhead of transmission of the offloaded requests. As a result, to address this dynamic behavior, the offloading decision of a users' request takes into consideration the users' position in addition to the available resources of the edge servers at each time. The main contributions of this chapter are summarized, as follows:

- A resource profiling mechanism interprets the performance requirements of the application to computing resources on the EC side. This mechanism enables the modular design and allocation of the EC resources in order to meet the varying offloading demand. The resource profiling of the LBS is based on linear models from System Theory, which effectively capture the system dynamics of the service, and determines the essential resources for serving various amounts of offloaded requests.

- A two-step mechanism to support and realize the offloading decision is devised, considering both users' contextual information and the availability of the edge resources. Towards this direction, the offloading decision becomes a key-enabler for maintaining high performance results, benefiting both the infrastructure provider and the users.

- An Edge Computing architecture to support the deployment, orchestration and management of LBS is introduced. Following the principles of 5G directives, an NFV-based deployment of the architecture is presented, using state-of-the-art software tools for the orchestration and management of the infrastructure.
- A real-world evaluation over an open 5G infrastructure is conducted to verify the validity and applicability of the proposed methodology. Also, the proposed resource profiling mechanism and the workload estimation approach are compared against well-established counterpart methodologies in the literature [89, 90].

3.4 System Architecture

As mentioned before, in the smart city scenario under consideration, visitors use an LBS to retrieve information about PoIs in a crowded touristic area. The LBS can be executed either on the user's mobile device or on the EC servers. Accordingly, the proposed EC architecture consists of two layers: namely the Device layer and the Edge layer.

At the Device layer, portable IoT devices, e.g., Raspberry Pis and mobile phones, equipped with sensors and camera modules provide location information and media content in order to identify PoIs e.g., museum exhibits) and retrieve information about them through an image recognition service. Since image processing is a computationally-intensive task, the users forward their requests to a cluster of edge servers for further processing in order to save energy and get the response in a timely manner. At the Edge layer, a control mechanism is responsible for scaling the resources of the instances of the virtualized image processing service and distributing the admitted requests among them. However, as specific may the above scenario seems, as generic and versatile the proposed architecture is, and it can be applied in any smart city application. To further elaborate on the generic essence of this work, we have identified the following use-case requirements and challenges.

- Every smart city application has specific dominant parameters that affect its performance. Thus, it is essential to build a profile of the application to include all important features. In this work, we focus on the user's location and propose a generic resource profiling of the application. The profiling mechanism provides accurate performance modeling under varying conditions and application requirements and facilitates the design of effective resource allocation strategies. It is an interpretation mechanism that translates the application's QoS requirements to EC computing resources and facilitate the interoperability in the case of heterogeneous resources.
- A successful offloading decision of mMTC applications must consider both the user's contextual information and the availability of EC resources. With this capacity, the user should decide whether the offloading is beneficial based on the location and the network/communication

parameters (e.g., signal strength). Especially in crowded places, communication related aspects such as the transmission time of a request, may present a significant overhead in the overall response time. Additionally, for applications of multiple users, the EC infrastructure provider should accurately estimate the dynamic workload demand and admit the number of requests that can be served with a guaranteed level of performance. Hence, we propose a two-step offloading decision that takes into account both the user and the provider's operating conditions and perspectives.

- The focus of this chapter is placed on the challenging scenarios where the edge computing infrastructure is receiving a great amount of incoming workload of offloaded tasks. This workload also fluctuates during the day, for example following the traffic of visitors at the place of interest. Hence, the scalability of EC infrastructure is a prerequisite for meeting the performance requirements and the optimal resource utilization. Our proposed dynamic resource allocation mechanism leverages the application's resource profile and allocates only the necessary resources in order to maximize the number of offloaded requests with response time guarantee.

In this study, the response time is defined as the sum of the transmission and processing time to serve an offloaded request. Figure 3.1 illustrates an overview of all levels of the proposed architecture. At the bottom level, the mobile device estimates the location of the user and originally decides the local or remote execution of the generated request. At the upper level, the Controller component realizes the intelligent functionalities of the proposed architecture, which includes the resource profiling, the workload prediction and the scaling mechanism along with the necessary monitoring service. Furthermore, the Virtual Machines (VMs) with specific computing resources host the image recognition service. The proposed application is considered to be stateless with stringent performance requirements. In the following sections, the individual components of the architecture are described in more detail along with the intelligence of the mechanisms provided at each layer.

3.4.1 Device Layer

At the Device layer, two fundamental mechanisms co-exist, providing the user's contextual information and the initial offloading decision.

3.4.1.1 Location Estimator

To estimate the user's position, a step detection algorithm has been developed. The users' motion is measured by inertial measurement unit (IMU) sensors and the estimation is performed by dead reckoning technique. This mechanism is classified in the category of Inertial based Systems, where there is no need for external information by the physical infrastructure, and the position of the user is estimated relative to a known starting point. This location estimation method is applicable in

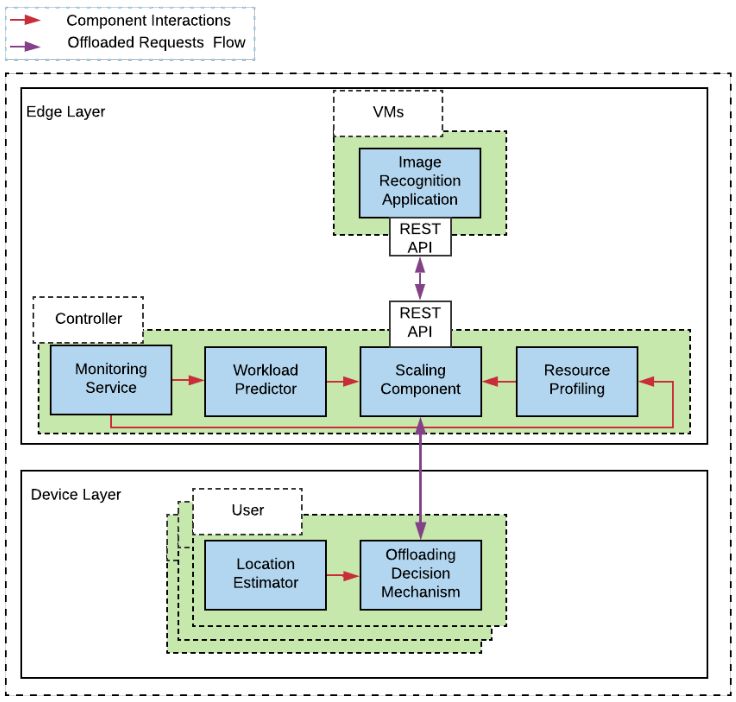


Figure 3.1: An overview of the EC architecture for LBS.

any end-device equipped with the appropriate IMU sensor. As thoroughly examined in [91], a step detection algorithm is applied using an accelerometer based on the mechanics of human motion, while the heading of the user is calculated by a gyroscope sensor. Consequently, given a starting position, the step and heading mechanism grants users the capability to locate themselves in an indoor environment and share this information with the infrastructure for an optimal offloading decision. Hence, in the scope of this work, the Location Estimator plays a significant role in the workflow of the offloaded requests, as shown in the next subsection.

3.4.1.2 Offloading Decision Mechanism

One of the main objectives of this chapter is the development of a smart dynamic offloading mechanism that is based on contextual information and aims to satisfy both the QoS requirements of the users and optimal utilization of resources of the underlying infrastructure. Therefore, we propose a two-step offloading mechanism, involving both the Device and Edge layer of the architecture. More specifically, whenever a new request is generated, the offloading decision is based on two parameters; (a) the position of the user, and (b) the current measurement of the wireless signal strength. On the one hand, we assume that the signal coverage is fluctuating in the territory of this scenario. As a result, one's position affects the quality of the wireless connection. Moreover, to facilitate automated offloading, the touristic area is divided into sections where offloading is permitted based on offline metrics about the quality of the connection. Subsequently, a map is constructed based on these metrics, hereinafter referred to as “offloading map”. On the other hand, in crowded areas, signal interference is high due to the shared medium of the wireless connection among the users. Considering that the transmission power of the antenna is bounded, the shared bandwidth results in a maximum number of users, who can offload their requests with satisfying transmission rate. Hence, the measured signal strength is adopted as an indication of the quality of the wireless transmission rate, which heavily affects the performance of the system.

At the Device Layer, we assume a predefined signal strength threshold that the user must satisfy in order to offload a request. If both the position and signal strength constraints are met, then the device offloads its request for further processing at the Edge layer, where the final offloading decision is taken. As a result, the user’s offloading decision is performed in both a proactive and reactive manner, exploiting contextual information of the user’s status in terms of position and connection quality. In case of a rejected request during this offloading decision phase, the image recognition service is executed locally on the mobile device.

3.4.2 Edge Layer

The workload generated by the mobile users is directed to the Controller of the Edge layer through a Wireless Access Point. As mentioned previously, the Controller is responsible for the allocation of resources to the deployed VMs, which incorporates the second step of the offloading decision, and the load balancing of the requests to the VMs. Initially, the monitoring service collects the necessary data for the incoming requests, the performance of running VMs and the utilization of the EC resources. The monitoring data is used by the rest components of the Controller. In particular, the Resource Profiling component provides an offline training tool based on System Theory to extract specific resource profiles of running VMs. A resource profile interprets the QoS requirements of the service (*i.e.,* response time) to EC computing resources (*i.e.,* vCPUs, memory). These profiles enable the Scaling Component (SC) of our approach. To better manage and orchestrate the EC infrastructure, time is divided into discrete time intervals. At the beginning of a time interval, based on Kalman Filtering approach and monitoring data, the Workload Predictor provides an estimation of the expected number of requests within the next time interval. This estimation and the computed resource profiles are used by the SC to determine the number and resources of running VMs that host the object recognition application. Hence, to guarantee the QoS of the the users, the SC accepts only up to a limit of incoming requests at each interval, derived from the profiles of the operating VMs, while rejecting those exceeding it. Thus, the SC takes the final offloading decision considering the overall performance of the EC infrastructure. Leveraging this mechanism, the SC assures that for each request that is accepted for remote execution, the users’ QoS criteria are met, and for the remaining requests, users execute the object-recognition service locally on their devices. Last but not least, within each time interval, the SC distributes each accepted request to the corresponding operating VMs, via a REST-API operating at the latter.

The activated VMs host the image recognition-based applications, which are trained off-line and identify the candidate PoIs.

3.4.2.1 Resource Profiling Component

In general, the resource profiling mechanism quantifies the relation between the application's requirements, the incoming requests and the computing resources at the EC side. Towards this direction, the Resource Profiling component build various resource profiles of an application that correspond to different operating conditions. In our scenario, the image processing service is CPU-intensive, thus, for the rest of the chapter, the resource the profiling mechanism refers to vCPUs. To extract the resource profiles, we adopt switching linear systems that can describe the performance of an application under different operational conditions. For different numbers of allocated vCPUs to a VM and varying incoming requests, we define a discrete linear system of the following form,

$$x(t+1) = \alpha x(t) + \beta u(t)$$

where $x(t)$, named as state variable, is the average response time in time interval t and $u(t)$, named as input variable, is the admitted requests in time interval t. In such a way, we describe the performance of the image recognition service under dynamic workload conditions. The Recursive Least Square algorithm [92] is exploited to calculate an estimation for the parameters α and β. The main objective of the Resource Profiling component is to maximize the offloaded requests and optimally allocate them the available computing resources. Towards this objective, for each linear system and given the desired average response time x_e, we extract feasible resource profiles, which maximizes the admitted requests and satisfies input constraints, by solving the following optimization problem,

$$\max_{x_m, x_M, x_e, u_m, u_M} \quad u_e \tag{3.1a}$$

subject to

$$x_e = a x_e + b u_e \tag{3.1b}$$

$$x_m \leq x_e \leq x_M \tag{3.1c}$$

$$u_m \leq u_e \leq u_M. \tag{3.1d}$$

First of all, the resource profile must be an equilibrium point of the linear system as defined by the first constraint. The second constraint implies that there is a minimum (u_m) and the maximum value (u_M) for the offloaded requests. Similarly, the last constraint refers to the state variables and means that the average response time varies between the minimum (x_m) and the maximum values (x_M). In such a way, the resource profiles actually translate the QoS requirements of the object recognition-based service to EC computing resources. Within the scope of this work, the resource profiles actually correspond to a VM with specific computing resources that serves a predefined number (u_e) of offloaded requests with a predefined average response time per interval (x_e). The value of x_e can be selected as a percentage of the maximum acceptable response time, which is implied by a Service Level Agreement (SLA) and it represents the strictness of the underlying QoS constraints. In our use case, the average response time of the resource profiles are set for demonstration only purposes, at the half of the maximum acceptable response time by the user.

The benefits of above resource profiles' design are manifold. Initially, it is a generic process that is applicable for any type of IoT-based application and edge/cloud infrastructure. The utilization of switched linear systems allows to model any operating condition and the design of the resource profiles according to the specific QoS requirements of the LBS. Contrary to queue models [90, 93] that are valid only in steady state, the linear systems can capture transient phenomena and do not provide only static information such as mean service and arrival rate. Additionally, although out of the scope of this chapter, the linear systems enable the design of feedback controllers, which can guarantee valuable system properties such as stability and reachability apart from the satisfaction of the QoS constraints. The latter cannot be accomplished with queue models. Finally, resource profiles enables the interoperability of heterogeneous infrastructure. In the case of edge-to-edge or edge-to-cloud collaboration, the computation of resource profiles for the different types of hardware simplifies the load distribution between sites, because the number of exchanged requests can be automatically translated into resources of each site through the resource profiles.

3.4.2.2 Workload Predictor

Due to the mobility of the users, the amount of offloaded requests changes significantly. In order to enable optimal scaling and resource allocation, an accurate workload prediction methodology is necessary to estimate future incoming requests. For this purpose, we adopt Kalman Filtering [94], which is a well-known estimation methodology for dynamic systems. We further assume that the offloaded requests can be modeled as a system with process and measurement uncertainties, as follows,

$$L(t+1) = L(t) + w(t) \tag{3.2}$$

$$Z(t) = L(t) + v(t) \tag{3.3}$$

where L is the number of offloaded requests of the image recognition service at time interval t. At interval $t+1$, the value of the workload is defined as the sum of the current value and the process noise w, which models the generation of requests by the users. The covariance matrix of the process noise is defined as Q. The variable Z denotes the measurement of the offloaded request and equals the actual number of requests L plus a measurement noise value $v(t)$, which follows a normal distribution with zero mean, while R is its covariance matrix. The estimation of the request at the next time interval $\hat{L}^-$ can be computed by applying the following Kalman filter equations.

$$\hat{L}^{-}(k) = \hat{L}(k-1) \tag{3.4a}$$
$$P^{-}(k) = P^{-}(k-1) + Q \tag{3.4b}$$
$$K(k) = \frac{P^{-}(k)}{P^{-}(k) + R} \tag{3.4c}$$
$$\hat{L}(k) = \hat{L}^{-}(k) + K(k)(Z(k) - \hat{L}^{-}(k)) \tag{3.4d}$$
$$P(k) = (1 - K(k))P^{-}(k) \tag{3.4e}$$

The term $\hat{L}^{-}$ is the update state value based on the measured value of Z and the extrapolated state value of the previous step. The term K is called Kalman gain and is a number between zero and one which expresses the importance of the measurement. The term P is the error covariance. In every step, we use the first equation to estimate the forthcoming requests for the next interval based on the previous extrapolated values. The estimation of the incoming request is used by the SC for the scaling decision.

3.4.2.3 Scaling Component

Based on the resource profiles of the application and the estimation of the offloaded requests, the SC aims to meet the dynamic workload demands and avoid over- or under-provisioning of EC resources. We formulate a mixed integer linear optimization problem (MILP), which minimizes the amount of the allocated EC resources. We define n binary variables $c_i, i = 1, \ldots, n$ that correspond to the n different resource profiles of a VM. Furthermore, for each c_i, we define the weight $w_i, i = 1, \ldots, n$, which is proportional to the number of the vCPUs, c, dedicated to a specific VM flavor. The optimization problem is formulated as follows:

$$\min_{c_i}\{\sum_{i=1}^{n}(w_i c_i)\} \tag{3.5a}$$

subject to:

$$\sum_{i=1}^{n}(c_i u_i) \geqslant \hat{L}^{-},\ i = 1, \ldots, n \tag{3.5b}$$
$$c_i = 0, 1,\ \forall i \in [0, \ldots, n], \tag{3.5c}$$
$$\sum_{i=1}^{n} c_i \leq TV \tag{3.5d}$$

where the first constraint indicates that the estimated offloaded requests $\hat{L}^{-}$ must be served by the activated VMs in the following time interval. The second constraint indicates that a VM with

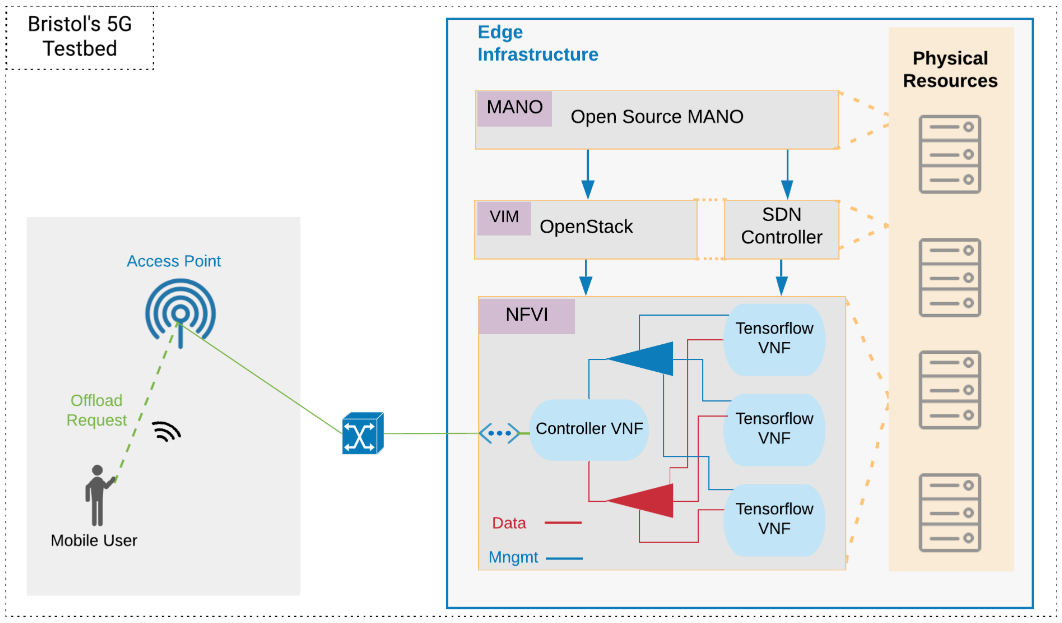


Figure 3.2: System Orchestration overview.

specific flavor can be either running or closed. This constraint implies that multiple instances of a VM with specific resource profile are not allowed in the same server. The last constraints guarantees that the resources of the running VMs do not exceed the total amount of available EC resources (TV) in terms of vCPUs. As a result, solving this optimization problem at each time interval, the SC ensures that the predicted workload will be served based on the corresponding resource demands with the minimum allocated resources. Nevertheless the overall cost of the flavors to be implemented is minimized. It is worth mentioning that the resource profiling mechanism allows to solve the problem for both cases of homogeneous and heterogeneous resources.

3.5 System Orchestration and Implementation

The aforementioned architecture was implemented in Bristol's University 5G testbed [95], which provides NFV orchestration and management solutions, bearing in mind that a real-world experience evaluation is of imperative importance when testing a system architecture. The System Orchestration overview is illustrated in Figure 3.2. Open Source MANO[1] (OSM) operates as an NFV management and orchestration (MANO) tool, in the context of the NFV MANO reference architecture, as defined by the ETSI NFV Standard. Thus, independently of the underlying Virtual Infrastructure Managers (VIM), OSM allows performing resource and service orchestration. In the

[1]https://osm.etsi.org/

proposed orchestration mechanism, OSM interacts with Openstack[2], which operates as the VIM. Openstack is a well-known open-source software for realizing Infrastructure as a Service (IaaS), structured by a stack of software tools used to build and manage cloud computing platforms for public and private clouds. Moreover, it is responsible for controlling and managing the Network Function Virtualized Infrastructure (NFVI) resources.

The proposed system orchestration has a twofold aim; to meet the system architecture requirements, and to provide a scalable virtualization solution, enabling the basic aspects of a next-generation LBS-enabled architecture. Hence, the respective services are deployed as Virtual Network Functions (VNFs). Subsequently, two different VNFs have been deployed; namely the Controller, and the LBS-enabled object recognition application. The Controller VNF implements the intelligence mechanisms of the proposed Edge architecture as thoroughly presented in Section 3.4.2, while the Object Recognition application is used for the identification of PoIs in the Bristol's Millennium Square. Both VNFs are managed and orchestrated by OSM as depicted in Figure 3.2. Similar to the system's architecture, the NFV implementation follows a top-down design. At the top layer lies the Controller VNF, which dictates the decisions needed for the scaling of the operating application-VNFs and the load balancing of the incoming workload, while, at the bottom layer, three VNFs are facilitating with the object recognition application. Moreover, the Network Service (NS) of the architecture is defined, to deploy an automated service chain between those VNFs. Furthermore, the NS is broken down into two parts, namely; data and management network, for the connection of VNFs, following the standards of NFV and SDN .

In this section, we present the necessary procedure for the realization of the system orchestration. Initially, we created the appropriate system images comprised of an operating system and the corresponding services. The functionalities of the Controller are deployed as the first system image, while the second one involves the object recognition service alongside with the REST-API service. These two images have been onboarded in the Openstack of Bristol's 5G testbed. Furthermore, the appropriate VNF Descriptor (VNFD) files for both images are realized. These descriptors contain all the necessary information for the VIM to instantiate Virtual Deployment Units (VDUs), which are the VMs that host the network functions. To be more specific, these descriptors define the flavors of virtual resources (vCPUs, RAM and storage) and virtual network connection points between the VDUs. Similarly, the Network Service Descriptor (NSD) specifies the Network Service (NS), which is the top-level structure that includes all the VNFs and implements the network topology of the experiment, associating respectively the connection points defined in VNFDs. Those descriptors have to be onboarded in OSM too. After this onboarding phase, OSM instantiates the NS and consequently the VNFs. Additionally, OSM dictates Openstack to realize the corresponding VDUs and the virtual network of the experiment. In our case, four VDUs are instantiated in Bristol's Openstack compute node. As explained in Section 3.4.2.1, three VDUs of different flavors were deployed hosting the object recognition service. Based on them, the SC decides at the beginning of each interval the running VDUs and directs the incoming traffic of

[2]https://www.openstack.org/

offloaded requests accordingly, solving the optimization problem in Section 3.4.2.3.

3.6 Experimental Evaluation

In this section, the experimental set-up and corresponding results of the proposed EC framework realization and deployment are presented and discussed in detail. The proposed framework was deployed in a real large-scale experimental facility of University of Bristol 5G testbed [95], that exploited OpenStack and OSM for NFV orchestration. To highlight the validity of the proposed location-aware EC architecture, a smart touristic application for crowded indoor areas was deployed. The considered experimental set-up provided realistic conditions to evaluate the proposed EC architecture for LBS from the perspective of: (i) location estimation accuracy, (ii) success of the offloading decision and the application performance, and (iii) effectiveness of the scaling mechanism, (iv) a comparative evaluation with well established studies [89, 90]. Before proceeding with the illustration and analysis of the the obtained results in terms of the aforementioned directions, a detailed description of the experimental set-up and considered scenario is provided.

3.6.1 Experimental Set-up and Scenario

In the experimental scenario under consideration, the visitors of Millennium Square of Bristol University testbed were scattered in a crowded touristic area, where several PoIs exist. Leveraging an object-recognition service, the visitors were able to retrieve interesting information about these PoIs. Google-powered TensorFlow [96] framework was deployed as the object-recognition service. Tensorflow is an open-source machine learning platform that can be easily retrained to classify the PoIs. For the retraining phase, a data set containing real pictures of the PoIs was used to build an object detection model for our LBS. At the Device layer, the visitors were emulated by Raspberry Pi devices equipped with the IMU sensor SparkFun MPU-9250 IMU Breakout[3], which contains a 3-axis gyroscope, a 3-axis accelerometer, and a 3-axis magnetometer. The Pi devices offloaded requests to the proximate edge computing infrastructure using a wireless local area network (WLAN) via an AC400 Nokia access point. Additionally, a laptop generated a large number of requests in order to create a dynamic realistic workload. As explained in Section 3.5, the Edge layer of the proposed architecture deployment was achieved exploiting the OpenStack and OSM for NFV orchestration, of the utilized tedbed facility. A powerful Dell PowerEdge T630 server with 32 vCPUs, 64GB RAM and 1TB storage was used to deploy all the necessary VDUs. Initially, in order to compute various resource profiles of the application we created a measurement data set and inferred three different types of VDUs. Table 3.1 provides the computing resources and the number of requests to be served per time interval for each resource profile. For every flavor, the average response time is set to 5*sec*. The control time interval was set to 30 *sec*. At the beginning of each interval, a workload estimation was computed according to Section 3.4.2.2 and subsequently the SC determined the combination of the running VDUs.

[3]https://www.sparkfun.com/products/13762

Table 3.1: VDU flavors based on the Resource Profiling.

Flavor	vCPUs	RAM	Requests served per Interval	Average Response Time
Small	2	2GB	3	5sec
Medium	4	4GB	14	5sec
Large	8	8GB	27	5sec

Table 3.2: Maximum error of estimation at each point of the trajectory of the user.

Point	Maximum Error of Estimations (m)
O(0,0)	0
A(10,0)	0.08
B(10,10)	0.1131
C(10,20)	0.1789
D(18,20)	0.394
E(18,30)	0.4861

3.6.2 Evaluation of User's Location Estimation

The purpose of this experiment is to demonstrate the accuracy of the location estimation technique. Figure 3.3 presents a sketch of Millennium Square topology. The wireless access point was located at point A(10,0) and enabled the users to offload their requests to the EC infrastructure. Furthermore, Figure 3.3 illustrates the "offloading" map of the Millennium Square. In the green-colored area, the wireless signal strength was high and the users were encouraged to offload requests, while the signal strength was poor in the white-colored areas. The strength of the wireless signal was measured using the Wavemon software[4], and the areas were divided by setting the threshold for a good quality signal to the value of $-70dB$. With respect to the differentiation in terms of signal strength in these areas, please note that in the specific square in the "white areas," physical obstacles exist (e.g., metallic constructions), which cause high interference resulting in wireless "dead zones". However, these metrics were performed by a single user while in a real-world scenario, the users' signal interference may heavily affect signal strength. To address this issue the offloading decision considers both the location-based classification of a user according to the "offloading map" and the contemporary signal strength of a user to specify his transmission capability in an online manner. This procedure is fundamental for the proposed architecture since the object recognition service is CPU-intensive. In such a way, we ensure that the transmission time is negligible relative to the processing time of an offloaded request.

In order to evaluate the location estimation accuracy, the user repeatedly (*i.e.,* five times) followed a path of six already known points, starting from point O and finishing at point E, while holding a Raspberry PI device mounted with the aforesaid IMU sensor. In Figure 3.3, for each point, the real position is marked with a star, while the cross symbol refers to the estimated position. At each point, the drawn circle represents the maximum estimation error. Additionally,

[4]https://github.com/uoaerg/wavemon

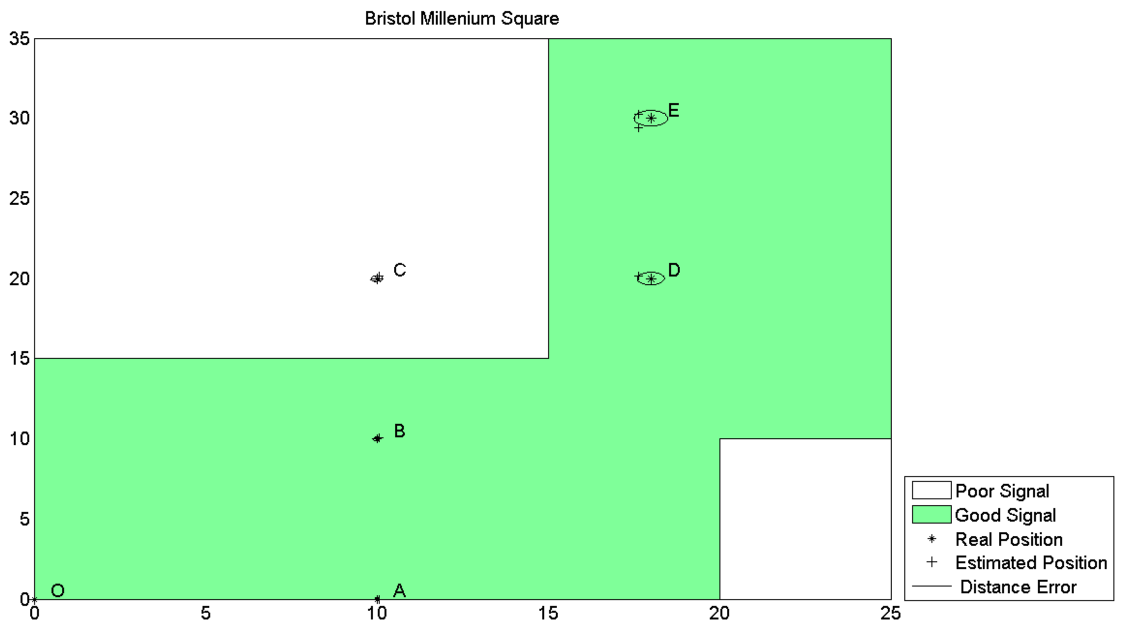


Figure 3.3: The “offloading map” and the user trajectory scenario at the Millennium Square of Bristol’s 5G testbed.

Table 3.2 summarizes the estimation of maximum error at each predefined point. It is obvious that the location estimation approach is very precise and the error is less than 0.5 meters in all cases, which is acceptable in our scenario. It is also worth mentioning that the estimation error is cumulative, thus, the error becomes higher at the last point.

3.6.3 Evaluation of Offloading Decision

Based on the experimental setup described previously, an experiment of one hour duration, was conducted to validate the success of the two-step offloading decision mechanism and the performance of the LBS on the edge infrastructure. A set of photos captured in Millennium Square were used to generate requests, whose inter-arrival time follows a Poisson distribution. As depicted in Figure 3.4, the black-colored line represents the magnitude of requests that were offloaded by the mobile devices based on the first step of the offloading mechanism. At the beginning of each time interval, the estimation of these requests (green-colored line) was computed by the Kalman Filter and was forwarded for further processing by the EC infrastructure. The blue-colored line corresponds to the number of executed requests on the deployed VDUs after the final offloading decision on the EC side, while the red-colored line represents the requests that were rejected by the EC servers and were executed locally on the users’ devices. Two useful remarks can be discussed here. First, the workload predictor can accurately follow the fluctuation of requests. It fails only when a sudden change occurs (e.g., at 200sec). Therefore, the number of rejected requests is high only at these intervals, and until the prediction is properly and timely adapted, as shown in this figure. Secondly, 91.37% of total requests were successfully offloaded and executed on the EC side, while only 8.63% were executed locally on the devices, which evinces the success of the two-step offloading strategy.

The proposed workload predictor is compared with an autoregressive integrated moving average (ARIMA) model [89] which is widely used for time-series forecasting. For comparison purposes, two well-known accuracy metrics are used, namely the percentage average error (PAE) and the Best Fit Rate (BFR) [97], which are defined respecitvely,

$$PAE = 100|\frac{E[y_k - \hat{y}_k]}{E[y_k]}|$$
$$BFR = 100\% max(1 - \sqrt{\frac{E[(y_k - \hat{y}_k)^2]}{E[(y_k - \overline{y}_k)^2]}}, 0)$$

The lower the PAE score is the more accurate the predictor is. However the PAE metric is sensitive to the values of large absolute error. Thus, we utilize the BFR metric which is more robust to the effect of small set of values. Contrary to PAE, high score of BFR indicates precise estimation.Table 3.3 shows the score of Kalman-based and ARIMA-based workload predictors. As it shown, the PAE score is very low for both predictors and ARIMA-based predictor is slightly better than Kalman-based. On the other hand, the proposed workload predictors achieves significantly higher BFR score than the ARIMA-based. These remarks indicate that both predictors provide successful estimation, which is acceptable for the scaling and offloading decisions.

The key performance indicator of the object recognition service is the response time of the requests. The execution time of a request by the mobile device varies between 30 to 60 sec. Following the design principles of Section 3.4.2.1, the extracted resource profiles were selected to have an average response time of 5sec, half of the maximum acceptance value by the users. As it is shown in Figure 3.5, the average response time of the offloaded requests throughout the experiment is 4.75sec (below the aforesaid threshold), subdivided into processing time (with an average of 4.02sec) and transmission time (with an average of 0.73sec). The average response time of the offloaded requests is at least five times lower than the local execution, which indicates that the resource profiling models effectively the performance of the application and motivates its necessity for properly meeting the QoS requirements. This also demonstrates that the second step of the offloading decision on the EC side becomes more of a requirement rather than a desire. Finally, the low transmission time implies that the first step of the offloading decision benefits the fulfillment of the user's requirement.

Table 3.3: Comparison of the Workload Predictors.

Accuracy Metric	Kalman-based Predictor (%)	ARIMA-based Predictor (%)
PAE	3.72	0.25
BFR	73.81	63.38

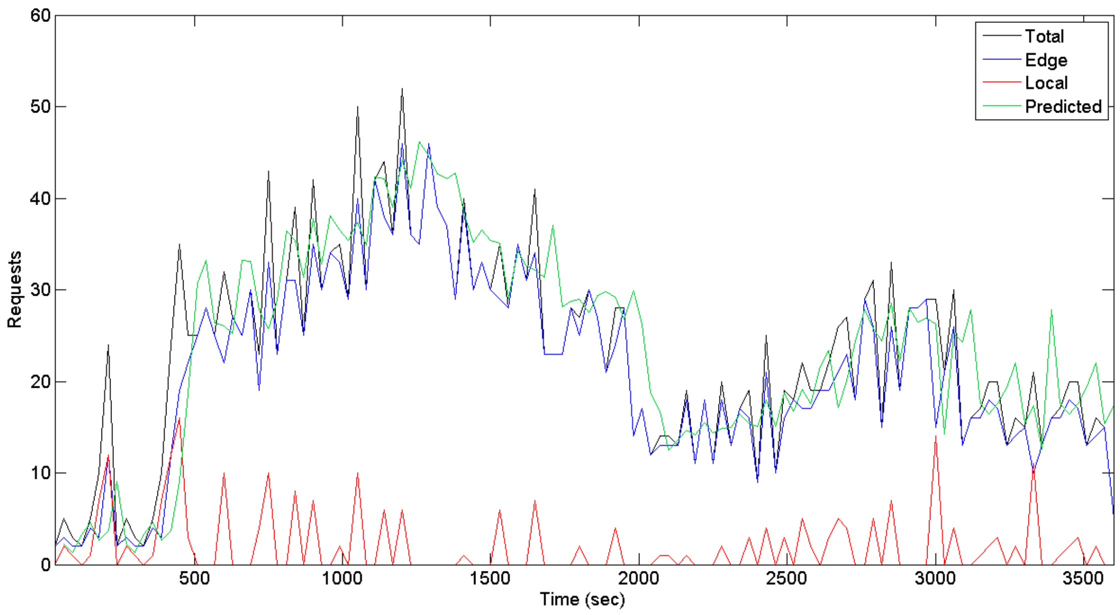


Figure 3.4: Number of total, predicted, remotely and locally executed requests per time interval

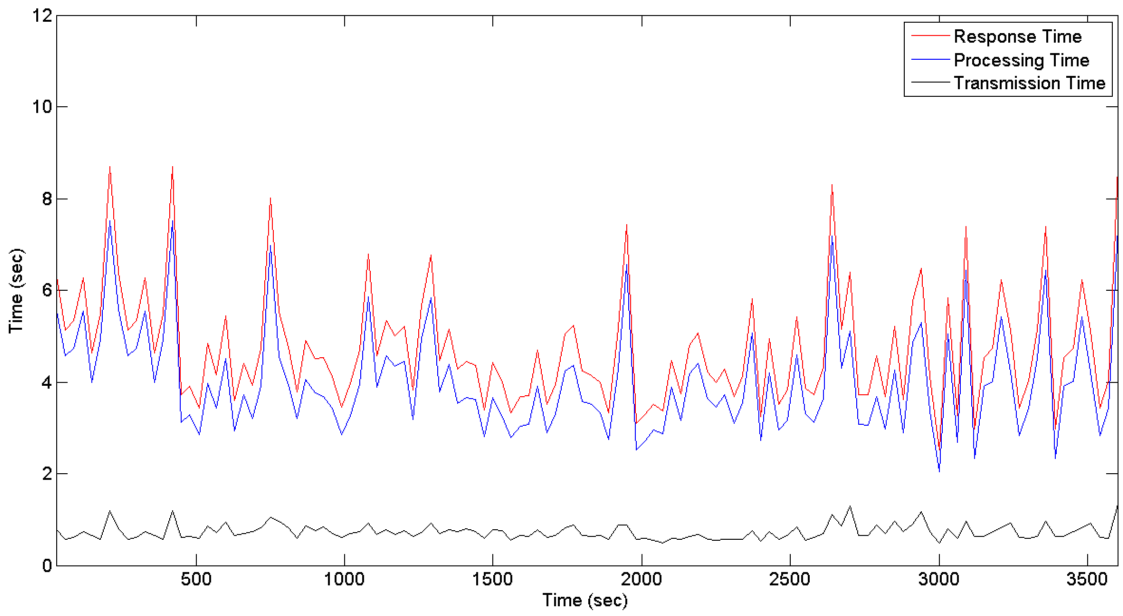


Figure 3.5: Average response, processing and transmission time of offloaded requests.

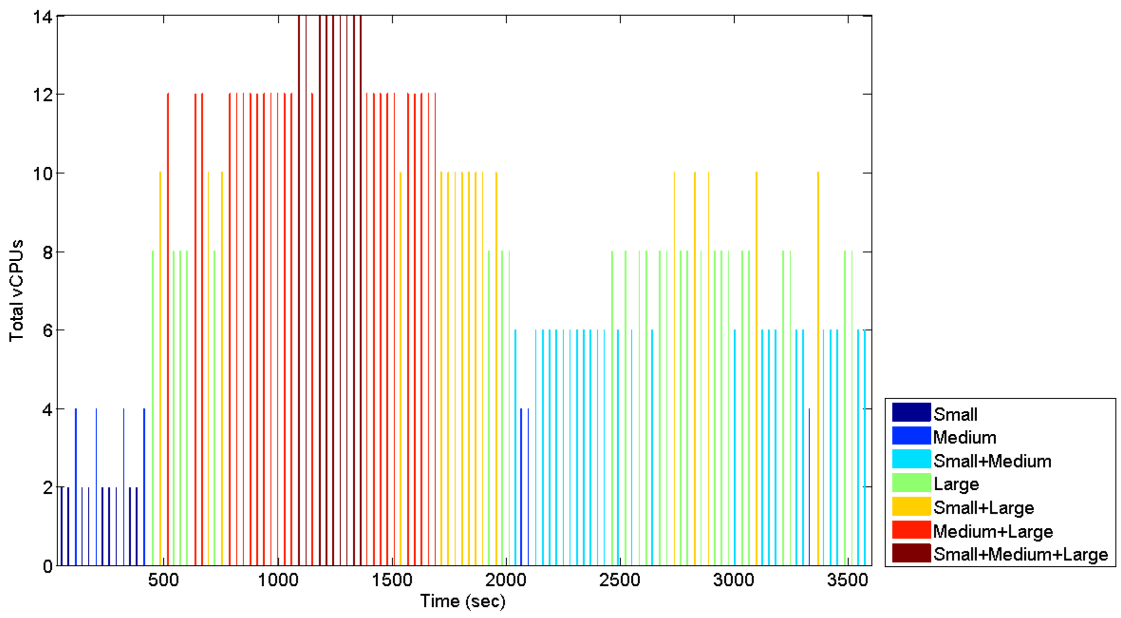


Figure 3.6: VDU flavors selected to operate at each interval according to the scaling mechanism.

3.6.4 Evaluation of Scaling Mechanism

The evaluation of the scaling mechanism was conducted based on the one-hour experiment described in the previous subsection. In particular, Figure 3.6 depicts the selected combination of operating VDUs at each time interval. Each combination of operating flavors is illustrated with a distinct color. Having in mind the prediction of requests in Figure 3.4 and corresponding trend, at the first intervals of the experiment only a few visitors offloaded their requests on the EC infrastructure. This workload was handled by a small or medium flavor VDUs. At the 600*th* simulation second, more visitors swarmed in the Square interested in PoIs, resulting in the gradual increase of the workload, which peaked at 50 requests per interval at the 1200*th* second. During this period, the full functionality of the scaling mechanism is clearly demonstrated. Specifically, to deal with the rising traffic, the SC determined the appropriate combination of active VDUs in order to avoid performance degradation. Moreover, even when the maximum number of requests occurred, the average response time remained below $4secs$. When the number of users began to decrease again, the scaling mechanism adapted timely and released resources from the operating VDUs, to follow the course. After the 2000*th* second of the experiment, the amount of requests fluctuates between 20 and 30 per interval. As it is shown, the scaling mechanism efficiently adapted the resources of running VDUs towards the workload fluctuation. The success of this mechanism is further strengthened by the performance of each VDU, as explained in detail below.

In Figure 3.7, for each VDU, the average response time of the offloaded requests is depicted in the respective subfigure. The information derived from these subfigures complements the operation

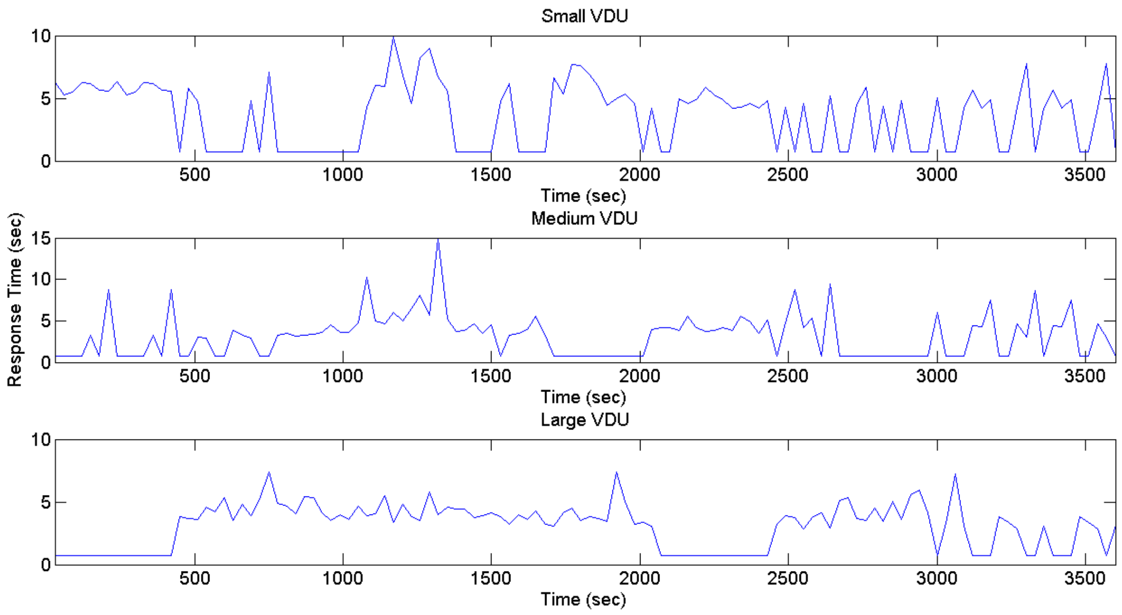
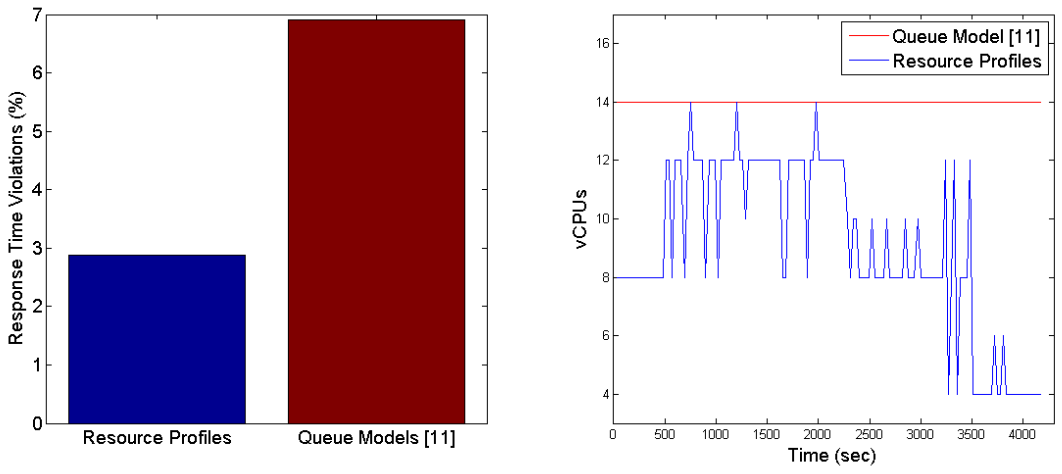


Figure 3.7: Processing time of requests at each VDU per time interval

of the scaling mechanism, and elucidates the periods of time that each VDU is active or not. During the whole experiment, there is only one time interval that the user's maximum acceptable response time is violated in the medium VDU. This implies that the scaling decision based on the resource profiles and the estimation of requests can guarantee the QoS requirements and the success of the final offloading decision.

3.6.5 Comparison of the Scaling Mechanism

In this section, we present a comparative evaluation of the proposed framework with the work presented in [90]. Similarly to the proposed EC architecture, this work presented a set of interconnected clusters forming a wireless metropolitan area network. Each cluster of computers; namely cloudlet, has a static resource provisioning to serve the offloaded requests. The main goal of this study is to redirect part of the incoming traffic of the overutilized cloudlets toward the underutilized ones. In [90], the average response time for each cloudlet is calculated by a queueing model. Utilizing the M/M/c queue model, the maximum offloaded workload served at each cloudlet is bounded by the corresponding service rate capabilities. The rest is considered rejected and redirected back to the user to perform local execution of the request. The service rate of a cloudlet is defined as the average number of requests that can be executed throughout a time interval. Following, the results of a comparative evaluation between the two methods, are presented. Moreover, to ensure an unbiased comparison, we deployed, the method presented in [90] in the same manner as the system orchestration and experiment setup presented in Sections 3.5,3.6 and we used the exact same workload of requests for both setups. Following the design, in subsection 3.4.2.1 the thresh-



(a) QoS Violations. (b) Utilization of Resources.

Figure 3.8: Comparative Experiment

old of the QoS violations is set at 10*sec*. In one hour of experimentation, as depicted in Figure 3.8a the resource profiles mechanism reported only 2.88% QoS violations contrary to 6.9% of the queuing theory modeling proposed in [90]. The static placement of the three flavors discussed in Section 3.6.1 has a total of 14 vCPUs. On the other hand, our approach utilized 35% less or an average of 9 vCPUs, throughout the experiment, as shown in Figure 3.8b. More specifically when the offloaded workload can be served the medium flavor VDU, as in the end of this experiment, the static placement results in a waste of resources that could be used for another application. Finally, to achieve all of the above, our framework had to reject 9.3% of the incoming requests, while in [90] 1.21% was rejected. To sum up, a dynamic resource allocation mechanism along with the resource profiles, are able to guarantee the QoS requirements while the resources utilized are minimized, sacrificing a small percent of requests rejected.

3.7 Summary

This chapter focused on the introduction, design, and experimental evaluation of a scalable two-tier Edge Computing architecture to realize location-based services in a smart city context. The scenario that motivated this work and was used for validation purposes, considers several users located in the vicinity of a tourist place, moving around looking for PoIs, and holding portable devices. Exploiting media content they opt to get additional information for the PoIs through an object-recognition service. As such, their requests are offloaded via Wi-Fi to an EC infrastructure in order to reduce power consumption and improve response time.

In particular, we proposed and evaluated a location estimation technique to assist with a smart offloading decision at the Device Layer, using contextual information of the users' state (*i.e.*, position and wireless signal strength). Furthermore, a mechanism for scaling and allocating the resources of the underlying infrastructure is realized to support and accompany the realization of the final offloading decision at the Edge layer, which depends on resource availability. This EC architecture is applicable in various MEC environments and mMTC application scenarios. The overall approach was implemented and evaluated in a real 5G testbed exploiting NFV orchestration and widely used software tools (e.g., OpenStack, OSM) for the execution of the experiment. Based on the obtained detailed experimental evaluation results, the location estimation method proved very reliable for the smart touristic scenario under consideration. With this valuable estimation, the two-steps smart offloading decision had a dominant role in the performance of the overall system and architecture. The scaling mechanism and the resource profiles yielded remarkable performance gains when compared to relevant well-established research works in the literature, not only in terms of meeting the user QoS criteria but also in eliminating the under-utilization of resources. The results of this Chapter are also presented in [98].

Chapter 4

Distributed Resource Autoscaling in Kubernetes Edge Clusters

4.1 General Setting

Maximizing the performance of modern applications requires timely resource management of the virtualized resources. However, proactively deploying resources for meeting specific application requirements subject to a dynamic workload profile of incoming requests is extremely challenging. To this end, the fundamental problems of (i) the scheduling of the incoming requests and (ii) the appropriate instantiation of resources for hosting the application must be jointly addressed. Kubernetes [63] is a state-of-the-art resource orchestration platform for containerized applications. The Horizontal Pod Autoscaler (HPA)[99] is widely used to implement the scaling of resources. The HPA's functionality mainly relies on stabilizing the utilization of the deployed resources towards meeting a target value of a single or a set of performance metrics, e.g., CPU utilization. Therefore, the HPA scales horizontally the number of deployed containers to meet these targets. However, 5G applications require more sophisticated scaling techniques to incorporate the dynamic nature of the incoming workload and the complex operation of modern applications [100]. Meeting vital QoS requirements via dynamic resource scaling is a major challenge primarily due to a set of competing Key Performance Indicators (KPIs), such as the utilization of deployed resources and the response time of the underlying application. A key enabler of scaling resources according to the workload traffic is the identification of the maximum number of requests that can be processed by the provisioned resources. Moreover, many recent Machine Learning (ML) based studies investigate extensively the challenging problem of resource management [101]. Nevertheless, the proposed ML-based solutions usually have a very challenging training phase and their scalability is limited [102].

In this chapter we extend the work of Chapter 3 and contrary to the one-sided philosophy of the HPA that takes into account only performance metrics, we propose a versatile architecture for resource management in Kubernetes Edge Clusters (KEC). We consider a scenario where users offload their requests to a KEC for further processing. Although we evaluate the proposed architecture with a web application, our solution can be tailored to the needs of any use case. An Additive Increase Multiplicative Decrease (AIMD) based scheme is proposed for the task scheduling problem. Our approach to scheduling is inspired by the AIMD algorithm, a celebrated method for congestion avoidance in the context of network management [103]. The proposed AIMD-based algorithm has recently been introduced in [104] and provides a stabilizing decentralized solution for parallel task scheduling and resource allocation of distributed computing systems. Exploiting the stability guarantees of this novel AIMD-like task scheduling solution, we dynamically redirect the incoming requests toward the containerized application. To cope with dynamic workloads, a prediction mechanism allows us to estimate the number of incoming requests. Additionally, a Machine Learning-based (ML) Application Profiling mechanism is introduced to address the scaling, by co-designing the theoretically-computed service rates obtained from the AIMD algorithm with the current performance metrics. The proposed solution is compared with the state-of-the-art autoscaling techniques under a realistic dataset in a small edge infrastructure and the trade-off between resource utilization and QoS violations is analyzed.

4.2 Related Work

In this Section, we review some of the most notable approaches in the literature for addressing the challenging research problems of task scheduling and resource management.

Baresi et al. [105], introduce a discrete feedback controller for cloud applications. The proposed controller assigns CPU cores to VMs or containers to meet the required average response time under the time-varying request rate of incoming requests. The proposed framework suits well with the concept of web microservices and outperforms Amazon's autoscaling controller. However, the load balancing in this work is considered static. Similarly in [98], the authors rely on VM flavors to build a scaling and load balancing mechanism to serve the incoming workload by adapting the number of replicas for each VM flavor. They formulate an optimization problem to determine the scaling decision while guaranteeing the QoS of the application and taking into consideration the available Edge Cluster resources. In both the previous works, the incoming requests are load-balanced with a static scheduling policy. Authors in [106] propose a solution for resource management in the cloud by clustering workload to meet the QoS requirements. They incorporate performance metrics using a decision tree (K-means algorithm) to determine the scaling decision. They utilize an imperialist competitive algorithm that assigns the workload to the closest cluster and then the resource provisioning is dictated by a decision tree algorithm. Therein, unlike our work, the scaling decision tries to balance the CPU utilization among the deployed resources without having guarantees for the QoS level.

Additionally, various studies in the literature propose auto-scaling approaches based on Machine Learning techniques. Iqbal et al., [107] tackle the issue of multi-tier application auto-scaling based on a supervised learning method. Specifically, they nominate, among others, the Random Decision Forest classifier to identify the appropriate resource provisioning scheme. The method features a specific workload, predicted via polynomial regression and the arrival request rate on the last k intervals on varying VMs that host web-based applications. Authors in [108], introduce an auto-scaler engine for Kubernetes that takes into account additional parameters compared to the default HPA. They integrate a parameter that sets a trade-off between over-provisioning and under-provisioning of the allocated resources for a precise workload, while they propose an ML-based forecasting approach for predicting the request arrival rate on Web server applications deployed on Kubernetes. Four distinct ML methods are implemented and compete for the scaling decision depending on their performance during a specific time window. Rossi et al., [109] present a Reinforcement Learning approach towards both horizontal and vertical scaling for container-based applications. Two distinct action models are proposed, where one acts either for horizontal or vertical scaling, while the other can perform two-dimensional scaling (both horizontal and vertical). However, the cost function used in the models does not contain any information regarding the incoming workload. Compared to the discussed ML-based approaches, in this chapter we provide an Application Profiling Mechanism, which uses an ML-based classifier trained with features gathered from workload estimation and monitoring performance metrics, taking also account of the theoretical values of the service rates obtained form the AIMD mechanism. Moreover, most of the aforementioned works provide centralized solutions for the resource management of resources. Moreover, in most cases, the task scheduling is decoupled from the scaling decision. In addition, determining the resource scaling decision typically falls in the order of seconds, which also adds significant overhead to the overall performance and fails to adapt to rapid changes in the workload.

4.3 Contributions & Outline

The key contributions of this chapter are summarized as follows:

- A holistic scalable architecture to tackle the joint problem of task scheduling and proactive resource management in KEC. We exploit an AIMD-like solution for the scheduling problem of incoming requests. The proposed event-triggered AIMD scheme facilitates proactive scaling of resources subject to a dynamic workload profile via a novel, intuitively conceived triggering locally identifiable condition. This mechanism enables decentralized resource orchestration at the network edge.

- An Application Profiling Modeling component is introduced that leverages information from (a) the scheduling and resource allocation solution, (b) the monitored KPIs, and (c) the workload prediction algorithm to estimate the essential number of replicas for meeting the workload demand. By collecting a set of distinct resource profiles, all associated with a stabilising resource allocation solution obtained from the AIMD mechanism, we can optimize resource utilization without violating the overall system stability. Then, the scaling decision is performed using a simple ML technique that allows us to also incorporate the performance metrics. The proposed design is scalable and easily re-configurable solution for the autoscaling problem in KECs.

- The proposed architecture is evaluated in a small-scale KEC using a real dataset of a touristic web application. Our numerical results illustrate how to efficiently trade-off between allocated resources and application performance, highlighting that the proposed framework significantly outperforms various configurations of well-known autoscaling solutions in terms of the utilization of resources.

4.4 System Architecture

In this Section, we present the core components of the proposed architecture deployed in the KEC. We assume that IoT devices generate workload for a specific application that arrives at the edge layer. We select Kubernetes as the Resource Orchestrator of the virtualized resources. Therefore, the architecture relies on widely used open-source state-of-the-art software tools for resource management and monitoring. An example application could be but is not limited to a computationally intensive algorithm, such as image classification. We should note that in most cases, Kubernetes is mainly used for web applications that need to perform rapid scaling decisions according to the ingress traffic. The proposed architecture operates in the same layer as the Resource Orchestrator and has three main components, namely i) the Load and resource Controller, ii) the Application Profiling Modeling, and iii) the Workload Estimator, as Figure 4.1 depicts. These components are responsible for incorporating the dynamics of the resource allocation mechanisms with the ones of the time-varying workload. In particular, we attempt to propose a holistic solution that simultaneously optimizes well-defined metrics related to the infrastructure (computing resources), and stabilizes the performance of the offloaded application, leading to safe, predictable, and optimal behaviors. The main functionality of each component can be summarized as follows:

1. **Load and Resource Controller:** the objectives of this component are (i) task scheduling and (ii) resource autoscaling. Regarding the first objective, this component is responsible for dynamic task scheduling of the offloaded requests based on the output of the AIMD algorithm. Secondly, using information from the Application Performance Modeling component, this component dictates the resource allocation strategy, aiming at guaranteeing the application's performance requirement in terms of QoS metrics and concurrently guaranteeing various system properties, such as stability.

2. **Workload Modeling:** this component is responsible for estimating the number of incoming requests for the containerized application leveraging a time-series forecasting method. The output of this component is used to take proactively scaling decisions.

3. **Application Performance Modeling:** this component aims at constructing representative dynamic models that can be subsequently utilized for dynamic resource allocation operation

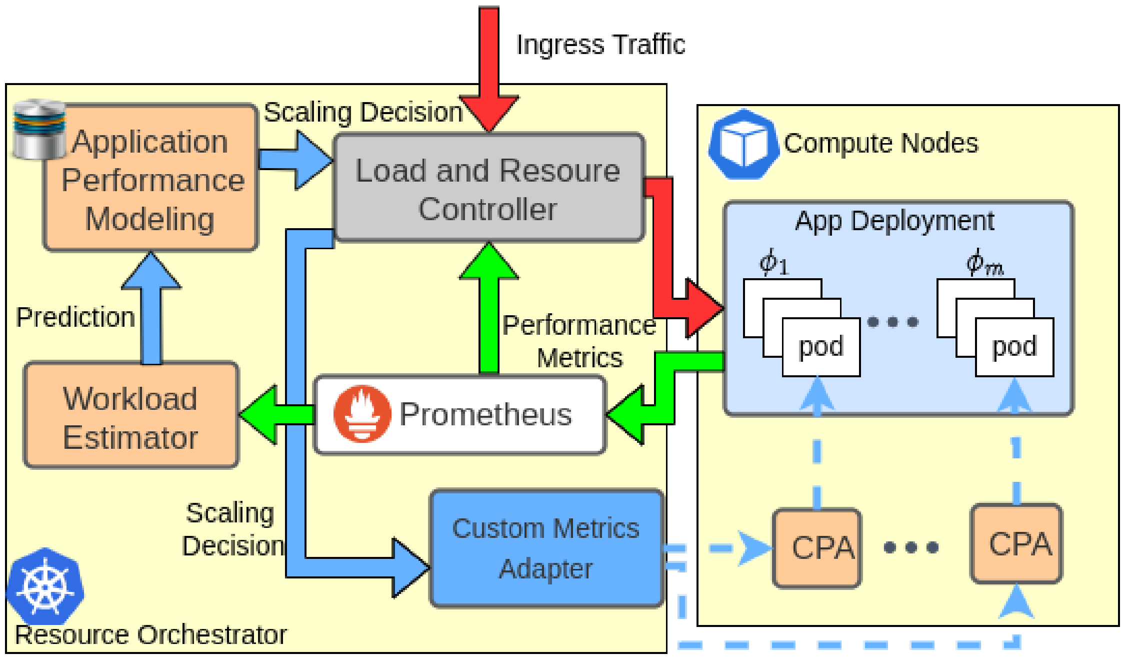


Figure 4.1: The proposed System Architecture for KEC.

of KECs. Based on ML, various models are extracted to sufficiently express the dynamic relationship between output (QoS) and control variables, respectively, i.e., allocated CPU resources, admitted workload, and active requests.

Kubernetes is used in this work as a resource orchestration platform. In this ecosystem, (i) a *pod* is the smallest deployment unit of computing resources for a containerized application, (ii) a *service* is an abstract way to declare applications as a network service, and (iii) *deployment* is a declarative way for the instantiation, configuration, and scaling of the pods that serve an application. Typically, multiple containers co-exist in the same application pod, managed as the same entity. Without loss of generality, we assume that each pod hosts exactly one container. Next, we define the application Resource Profiles:

Theorem 2. *An application Resource Profile φ_i is the relationship between the allocated resources of a pod and the maximum request rate of incoming requests that the pod serves while keeping a certain QoS level.*

For example, we assume that one container allocated with 1 vCPU and 1 GB of RAM, can serve on average 2 requests per second without having QoS violations. We let m be the number of the different resource profiles for a specific application. Therefore, a Kubernetes service and deployment are realized for all resource profiles; φ_i, $i = 1, \dots, m$. We define App Deployment as the abstract way to refer to the network and the computing resources of each resource profile. Then, each App Deployment has different available resource limits for the deployed pods. In the context of Kubernetes, the resource limits are used to enforce that the instantiated containers of the pod will operate in predefined regions in terms of CPU and memory. Moreover, for each App Deployment, r_i denotes the number of identical pod replicas that are running. The upper replica limit is considered to be the same for all deployments and expressed as r_{max}. To identify the resource profiles, extensive offline experimentation has been carried out involving three different configurations regarding (i) the incoming request rate, (ii) the resource limits, and (iii) the number of replicas. The aim was to empirically compute the maximum request rate before QoS violations occur for different sets of resource limits and replicas.

Concerning the monitoring system, we rely on Prometheus [110] for collecting (i) the workload-, (ii) performance- and (iii) application-, related metrics. The metrics are updated at each time slot. Moreover, we utilized the Custom Pod Autoscaler (CPA)[111] as the controller that scales the App Deployments. For each resource profile, a CPA instance is realized to scale in/out the number of replicas, which varies from zero to r_{max}. Contrary to CPA, HPA does not support zero replicas. Hence, CPA enforces the scaling decision of the Load and Resource Controller, checking for changes in the desired number of replicas at each time slot.

The core intelligence of the proposed framework is provided by the Load and Resource Controller. The incoming requests are load-balanced between the App Deployments according to the output of the scheduling algorithm, which is deployed on this component. The Workload Estimator takes input from Prometheus to predict the incoming workload. At each time slot, given this prediction, the performance metrics, and the output of the scheduling algorithm, the Application Performance Modeling component computes the desired number of replicas for each App Deployment using an ML model. Then, the Load and Resource Controller exposes the scaling decision for each App Deployment, via the Custom Metrics Adapter to be fed to the CPA's Controller unit. The complete architecture of the proposed solution is illustrated in Figure 4.1. In the following Sections, the above three components are analyzed in detail.

4.5 Load and Resource Controller

This Section provides all the necessary information for the operations of the Load and Resource Controller. Initially, an overview of the AIMD-like task scheduling algorithm is described. Then, the actual implementation of the algorithm for a KEC is presented.

4.5.1 AIMD-like Task Scheduling Algorithm

As Figure 4.2 shows, a central task queue $\delta(t)$ is defined as an aggregation point for the incoming requests with an arrival rate $\lambda(t)$ and then served in a First Come First Served policy. An event is generated whenever this queue empties, which is guaranteed to happen since the admission rates $u_i(t)$ are piecewise increasing functions of time. At the time of the event, i.e., when the queue empties, the admission rates drop instantaneously. Each request is processed via a multi-queue

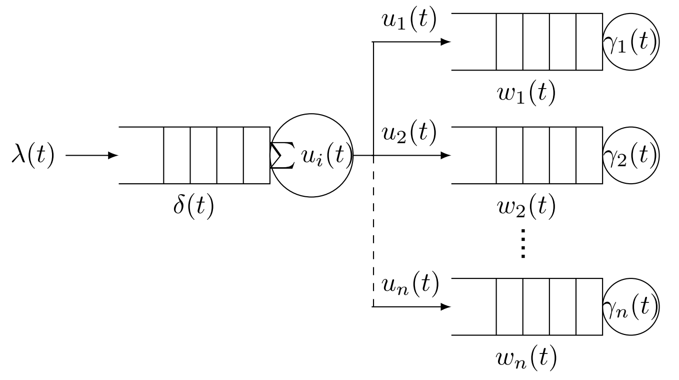


Figure 4.2: A multi-queue system with AIMD policy.

system with multiple processing nodes. Each processing node $i = 1, \ldots, n$ has a service rate $\gamma_i(t)$ and the queued requests at each node are denoted with $w_i(t)$.

The proposed algorithm has two phases, namely, (a) the Additive Increase (AI) phase and (b) the Multiplicative Decrease (MD) phase. During the first phase, the admission rates towards the processing units continuously increase aiming to admit all of the requests and drain the central queue. Inevitably, an event is triggered, i.e., $\delta(t_k) = 0$, where k is the number of events. At this instant, the scheduling algorithm enters the MD phase, which enforces a rapid decrease in the admission rates, and then reenters the first phase immediately.

The admission rate towards the i^{th} processing node is defined as:

$$u_i(t) = \beta_i u_i(t_k) + a_i(t - t_k), i = 1, \ldots, n, \tag{4.1}$$

where t is the current time, t_k is the time of the k^{th} last event and $\alpha_i > 0, 0 < \beta_i < 1,\ i = 1, \ldots, n$ are tuning AIMD parameters. Specifically, the α_i is the growth rate of the admission rate of node i and β_i is the multiplicative decrease factor of the admission rate when an event is generated. Thus, a request arriving at the aggregation point stalls at the central task queue until it is redirected to a processing node. The rate of redirecting requests from the central task queue to the processing nodes at each time is $\sum_i^n u_i(t)$. It is easy to show that the inter-event period between two events is defined as:

$$T(k) = \max(0, \frac{\lambda(k) - \sum_{i=1}^n \beta_i u_i(k)}{\sum_{i=1}^n \frac{1}{2} a_i}), \tag{4.2}$$

where $u_i(k)$ is the admission rate towards the i^{th} node at time $t = t_k$. Under the scheduling solution (4.1)-(4.2), results from [104] indicate that if we consider the following decentralized resource allocation policy for the service rate of each node at the k^{th} event:

$$\gamma_i(t_k) = \beta_i u_i(t_k) + \sqrt{2a_i w_i(t_k)}, \tag{4.3}$$

then, the overall systems is stable, in the sense that performance metrics, such as execution and response times, are bounded as more requests are added to the system. QoS requirements can be attained by an efficient tuning of the AIMD parameters, see e.g., [112]. Moreover, individual service rates are adjusted only when an event occurs via the decentralised feedback rule (4.3) which requires only local information, namely, the local admission rate u_i and queue length w_i. To sum up, requests arriving at the first queue are not served instantaneously by a processing node having a slight increase in the total response time, however, they are placed in subsequent local queues in a way guaranteeing the stability of the overall system. We should note that the AIMD parameters can be selected individually for each processing node and remain constant. Thus, Eq. (4.3) is a decentralized feedback resource allocation controller, scalable and locally configurable. Moreover, in [104] it is shown that $T(k)$ also converges after a few steps of the algorithm. For the interested reader, a survey on AIMD-like algorithms is presented in [113].

4.5.2 Integration with Kubernetes

In this section, we provide information about the integration in Kubernetes of the Load and Resource Controller component that exploits the AIMD-like task scheduling policy and resource allocation.

To develop the AIMD algorithm several open-source software tools and technologies are used. In what follows we briefly explain the functionalities, as we believe it is important for understanding the rest of the chapter. In Figure 4.3 the main components are presented. The ingress traffic for a specific application hits the exposed service of a Flask Rest API. This API is the aggregation point that also performs the task scheduling policy. To implement the dynamic task scheduling policy of the AIMD algorithm, we selected Python Celery, which is a software that suits well with Flask and the main functionality is to create workers to implement asynchronous tasks. The workers communicate via redis, a messaging broker, monitoring several events, e.g., the arrival of a request, completion of a task, etc. The Queue Worker implements the central task queue $\delta(t)$. Celery can implement a rate limit for each worker, thus, the Queue Worker redirects tasks with a rate of $\sum_i^n u_i(t)$ at each time slot. Moreover, the Beat Worker is triggered every time slot to update the admission rates and enforce the new rate limit to the Queue Worker. Hence, the requests may hold in the aggregation point before being redirected to the processing nodes due to the rate limit. We should note that the rate limit is a common practice at the production level to enforce billing policies for the incoming traffic. The dynamic task scheduling policy serves as a stabilizer for the processing nodes giving them time to process the requests and instantiate new resources accordingly. All configuration and monitoring of these procedures are stored in a Postgresql relational database.

As mentioned above, we consider $\varphi_m, i = 1, \dots m$, distinct resource profiles with different processing capabilities as the processing nodes. For each resource profile, a separate Celery Worker is created to redirect the HTTP traffic from the Queue Worker to the corresponding Application Pod that serves the virtualized instance of the application. We consider that if a request is redirected by a Celery Worker towards the i^{th} application pod then it belongs to the w_i queue. Subsequently,

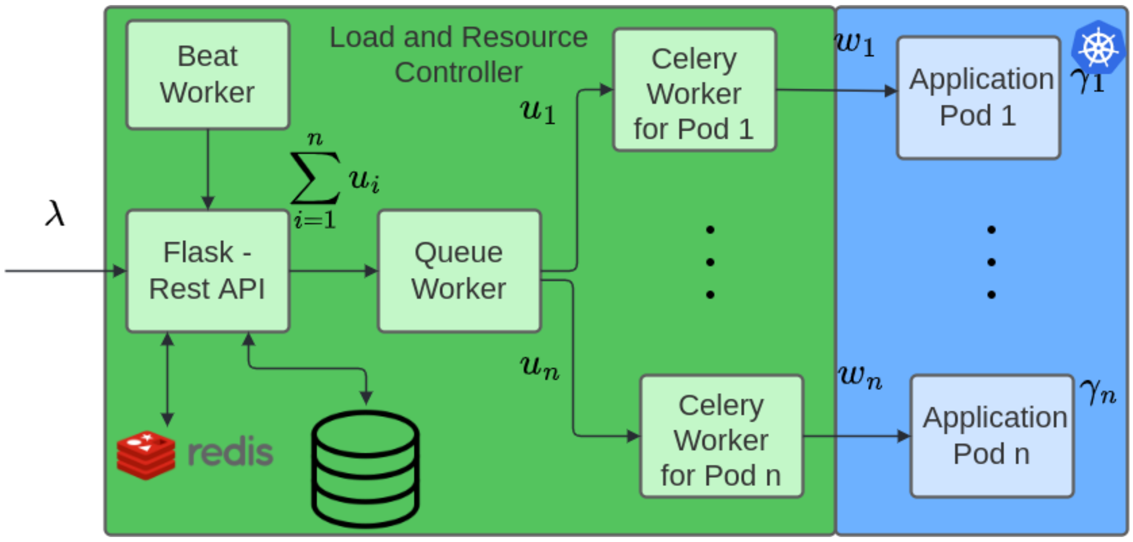


Figure 4.3: AIMD integration with Kubernetes.

we dynamically load balance the incoming workload on the instantiated pods.

The proposed architecture is scalable and operates in a distributed manner as for each computing node, no knowledge of the state of other nodes is needed. We should also mention that in the case of multiple applications is also possible having multiple aggregation points for the incoming workload. However, a missing capability is to interpret, and subsequently enforce the computed theoretical service rates to each processing node, in other words, decide the number of replicas for each resource profile φ_i to guarantee at least a service rate of $\gamma_i(k)$. In the next Section, we provide a solution to this challenging problem.

In the following Sections, we present the last two components of the proposed system architecture i.e., (a) the Workload Estimator, and (b) the Application Profiling Modeling.

4.6 Workload Estimator

To timely adjust the service rate of the deployed resources, it is of high importance to predict the admission rates. As the AIMD algorithm dynamically changes the admitted workload $u_i(t)$ towards the pods, the Workload Estimator tries to estimate the number of requests that will be admitted since the next event under the dynamic workload. This is important as continuously applying different service rates to the pods, i.e., scaling out/in the number of replicas may lead to performance deterioration. Hence, we deploy an autoregressive integrated moving average (ARIMA) model [114]. Such predictive techniques are very common for time-series forecasting. The model is trained using offline information and retrains online as the workload varies. This information is then used from the Application Profiling Modeling component to deploy the desirable number of replicas for each resource profile.

4.7 Application Profiling Modeling

The Application Profiling Modeling is based on machine learning (ML) techniques, which essentially involve the classification of distinct combinations of computing resources concerning the application's service rate. The classification reflects the number of replicas r_i of a resource profile φ_i, leveraging various metrics retrieved from the core components of the proposed framework. Taking into consideration the workflow followed by the Load and Resource Controller, separate ML models-classifiers are implemented for each resource profile. Subsequently, the training process of distinct models demands separate data that characterize the respective resource profile φ_i. Thus, hereby, a description of the data collection process is initially given.

Data Generation: The appropriate scaling decision is directly related to the satisfaction of the QoS constraints. Considering the discussed architecture, several features can be taken into account to characterize the performance of a pod and of course, the dynamic conditions regarding the pod's resource utilization, the service and admission rates occurred by the task scheduling solution. To acquire the data that contain mandatory metrics regarding the scaling decision, we conducted offline experiments, utilizing the components of the proposed architecture for isolated resource profiles. In detail, per experiment, only one resource profile is used for the corresponding application. Different configuration schemes of (i) the incoming request rate, (ii) the resource limits, and (iii) the number of replicas were applied to identify settings that meet the QoS constraints for the underlying application. Subsequently, the collection of data, is categorized in the components of the architecture as depicted in Table 4.1. The data are filtered before the processing phase to contain only valid training samples in terms of QoS guarantees based on the average response time of the application occurring from node i. The average response time of the requests in the last inter-event period in node i is denoted as art_i. Moreover, we utilize the ARIMA-based Workload Predictor to get an estimation regarding the admission rate for each resource profile, denoted by u_i^{pred}. This feature contributes to the Application Profiling Modeling component to consider a characterization of the workload in a specific time window. Specifically, u_i^{pred} is determined by a specific range of previously gathered u_i^{real} values.

Dataset Pre-processing: The collected data from the aforementioned experiments are raw and therefore not suitable for training machine learning models. The retrieved features are on a different scale. To achieve faster training and improved classification accuracy, we perform feature scaling with normalization by utilizing the *MinMaxScaler*[1]. Normalization of the features leads to values that range between 0 and 1, and, so each feature contributes equally to the classification process, especially to distance-based classification algorithms. Depending on the configuration of each experiment, the distribution of samples across the known classes (i.e., number of replicas) might be biased or skewed. Data imbalances can affect classification predictions when they are not properly managed. Thus, balancing the dataset is mandatory before proceeding. One way to address this issue is to generate new samples in the class which is under-represented. The most naive strategy

[1] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

Table 4.1: Dataset's Features Specifications.

Feature	Component	Specification
γ_i	AIMD Algorithm	The service rate of each node.
w_i	AIMD Algorithm	The queued requests at each node.
u_i	AIMD Algorithm	The admission rate to each node.
u_i^{real}	Performance Metric	The actual admission rate to each node during a time window.
u_i^{pred}	Predicted Metric	The admission rate predicted by the Workload Estimator for each node.
art_i	Performance Metric	The average response time obtained by each node.
r_i	Performance Metric	The number of replicas for the underlying resource profile φ_i.

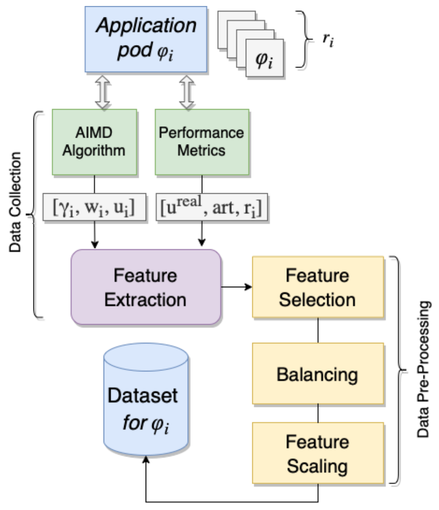


Figure 4.4: Data Generation Workflow for the Application Profiling Modeling.

is to generate new samples by randomly sampling with the replacement of the currently available samples. The *RandomOverSampler* offers such a scheme, and we utilized this technique from the sklearn python library. Another aspect of the pre-processing phase is the feature importance computation, as well as the correlation between them. So, we calculate the feature importance using Random Forest Algorithm [115], to identify the features that reflect more information regarding our classification problem while reducing the dataset's dimensions. Nevertheless, the dimensionality reduction of the feature space will lead to a faster training process in general and specifically faster prediction. In our case, the selected features after the normalization are the (i) u_i^{real}, (ii) γ_i, (iii) w_i, and (iv) u_i^{pred}.

ML-based Application Profiling: The Application Profiling Modeling component undertakes the process of providing the CPA with the value of r_i, at each event, leveraging the aforementioned selected features, i.e., w_i, γ_i, u_i^{real}, u_i^{pred}. Considering the fact that the sample points of the

Table 4.2: Classifiers' Accuracy on Test Dataset

Classifiers	**Accuracy**		
	Small	**Medium**	**Large**
Support Vector Machine	0.51	0.55	0.66
Logistic Regression	0.44	0.57	0.60
Random Forest	0.75	0.87	0.83
k-Nearest Neighbors	0.72	0.80	0.79
Random Forest - optimized	0.76	0.88	0.87
k-Nearest Neighbors - optimized	0.76	0.90	0.87

dataset are labeled, as the class index is the number of replicas r_i, we attempt to train and evaluate several supervised learning algorithms to produce the profiles for each φ_i. In precise, the following algorithms were trained and evaluated: Logistic Regression (LR) [116], k-Nearest Neighbors (kNN) [117], Random Forest (RF) [115] and Support Vector Machines (SVM) [118].

We perform benchmarks to nominate the algorithm that will be used in the Application Profiling mechanism. As Table 4.3 shows, three different resource profiles are used. For each resource profiles, we generate a training and an evaluation dataset. Relying on the aforementioned pre-processing techniques, dataset preparation is performed, prior to the training phase. The ratio between training and test sample points is 70% to 30% of the initial dataset. Initially, the ML-algorithms are trained with their default hyperparameter settings. The performance of the algorithms is evaluated based on their accuracy score on the test set. Table 4.2 illustrates the accuracy of the ML algorithms for each resource profile. In general, the kNN and RF methods achieve the highest accuracy score compared to the rest classifiers. Thus, we perform optimization for these two algorithms via hyperparameter tuning, using grid search technique, where multiple hyperparameter configurations are tested. This procedure significantly increased the accuracy of the optimized kNN and RF algorithms. However, generating a large dataset to further increase or test the accuracy of the proposed ML models is very challenging. In particular, in our case, the performance metrics heavily rely on the infrastructure capabilities, the type of requests and the application. On the other hand, working with small-scale datasets may lead to overfitting the ML models. As a result, we select the kNN algorithm as it is an instance-based learning algorithm with fast training phase, and it involves only storing feature vectors and class labels of the training samples. Also, it suits multi-class problems like our and its non-parametric nature is appropriate for small datasets since no assumptions on the data distribution is required. We note that the proposed scaling decision runs in the order of ms, so the overall performance is not disrupted. More details on kNN hyperparameter configuration are given in Section 4.8.

4.8 Experimental Evaluation

This Section presents the experiment setup and the experimental evaluation of the proposed architecture with other autoscaling schemes for KECs. We deployed a Kubernetes cluster using 3

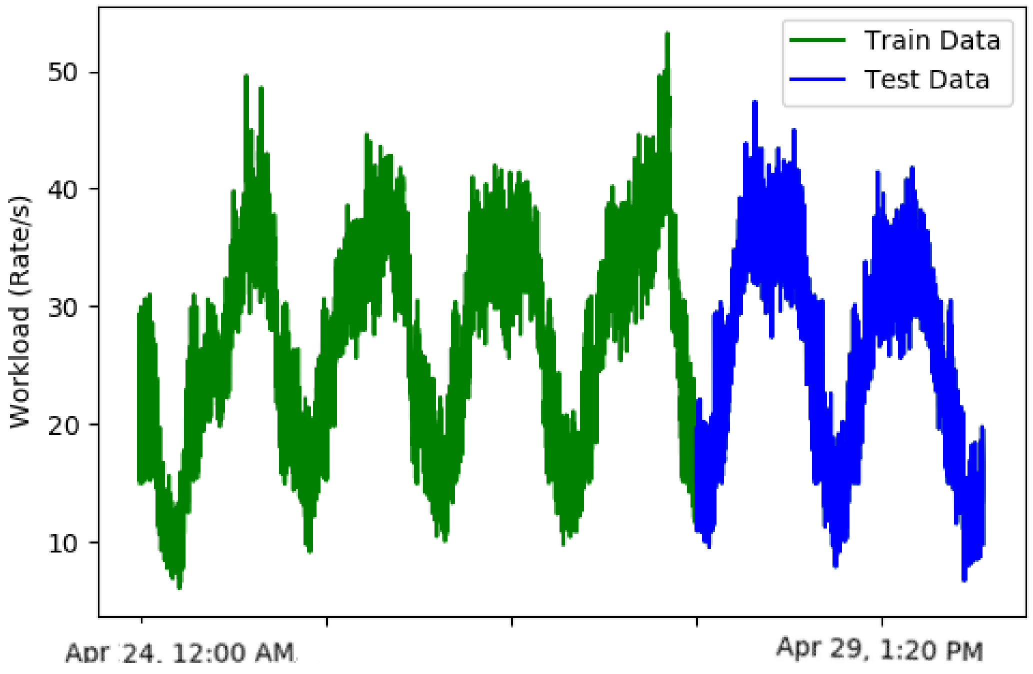


Figure 4.5: Workload Trace used for training and experimentation.

Table 4.3: Resource Profile Setting.

Resource Profiles	Small	Medium	Large
CPU cores	1	2	4
RAM (GB)	2	4	8
$[\alpha_i, \beta_i]$	[0.4,0.5]	[1,0.5]	[2.4,0.5]
Max Req Rate (req/s)	1.5	3.6	7.8
i	1	2	3

Openstack VMs in two Intel Xeon Silver 4110 servers. We consider three different resource profiles, $m = 3$, namely small, medium, and large, following the production standards, see, e.g., Azure[2]. The maximum number of replicas selected is $r_{max} = 4$. The resource limits and AIMD parameters for each resource profile are presented in Table 4.3. Also, in the same Table, the maximum request rate that a pod can serve before noticing QoS violations is presented. The selected application is a simple image classification application that uses OpenCV. We assume that if a request takes three times more than expected to be processed then we have a QoS violation, and the request is rejected by enforcing a connection timeout.

Every time slot of 0.1 sec, we scrape the Prometheus metrics and the Beat Worker operates. For the offline training of the ML-based Application Profiles, we used a workload trace acquired

[2]https://azure.microsoft.com/en-us/pricing/calculator/

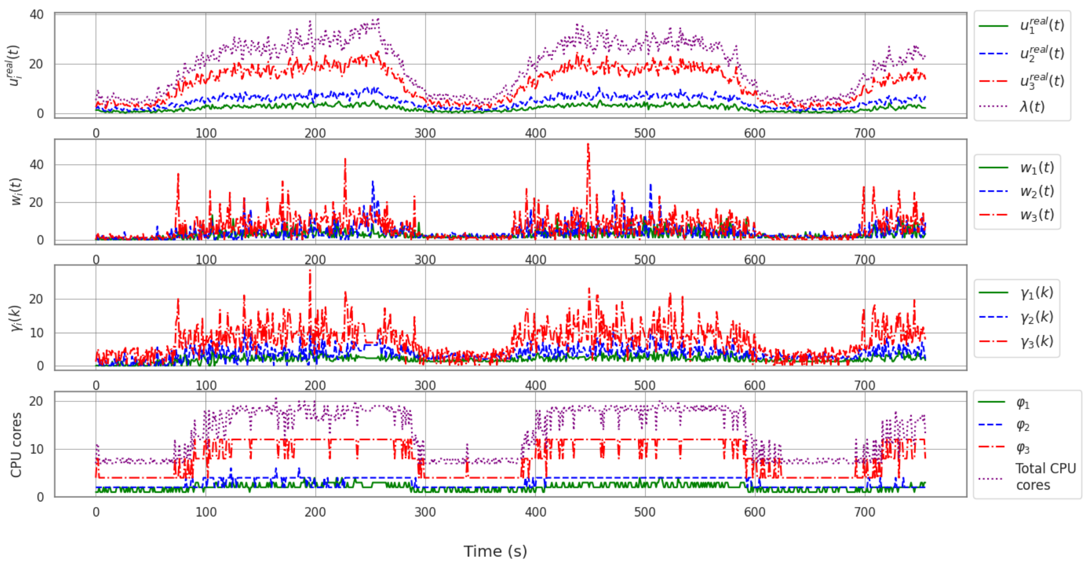


Figure 4.6: The performance of the proposed DRA method.

from Ferryhopper website[3], which provides ferry booking services around Europe. Figure 4.5 presents the per minute distribution of HTTP request rate spanning in six days. The data of the first four days were used to train the Workload Estimator and the Application Profiling Modeling components, i.e., the ARIMA model and the ML-based Application Profiles. The data of the last day is used for the evaluation part. The ARIMA model was trained using the autoarima package, which yielded a model of order (3,1,1).

Regarding the kNN models, used in the application profiling mechanism a hyperparameter tuning is performed, using 10-fold cross-validation [119], which is a typical ML modeling optimization technique. The basic hyperparameters of kNN are (i) the k parameter, which determines the number of neighbors that are taken into account to classify a new sample, (ii) the distance measurement between the sample points (e.g. euclidean), and (iii) the weight for the distance. After the optimization, the hyperparameters configuration is: (1) for the small resource profile: $k = 2$, $distance{=}euclidean$ and $weight{=}1/distance$, (ii) for the medium: $k = 1$, $distance{=}euclidean$ and $weight{=}uniform$ and (iii) for the big $k = 16$, $distance{=}manhattan$ and $weight{=}\ 1/distance$.

We compare the proposed architecture, hereinafter Distributed Resource Autoscaling (DRA), with four different setups, namely (i) modified HPA (m-HPA), (ii) S-HPA (iii) M-HPA, and (iv) L-HPA. The m-HPA, on the one hand, utilizes the three resource profiles by load balancing the incoming requests with a constant ratio. On the other hand, the resource scaling decision is dictated by an HPA instance for each resource profile by targeting 70% of CPU utilization. The last three setups deploy only one resource profile at each time, i.e., only small (S-HPA), only medium (M-HPA), or solely large (L-HPA) resource profiles, respectively. Moreover, the scaling is, again, performed by HPA by targeting 70% of CPU utilization for the deployed replicas. These setups

[3]https://www.ferryhopper.com/

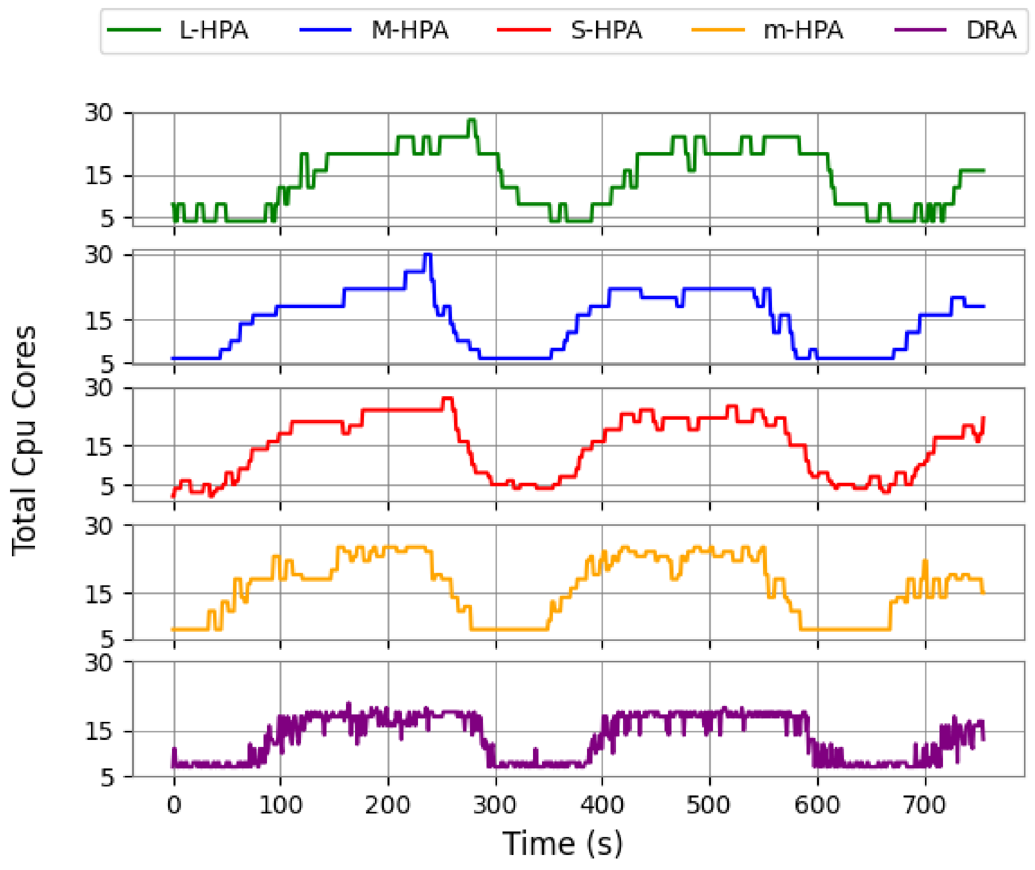


Figure 4.7: The Total CPU cores utilized for each method throughout the experiment.

are all evaluated against a seventy-minutes workload taken from the test data of the Ferryhopper trace. All HPA instances operate every 1sec. The experiments were conducted ten times for each method, and all the results are averaged.

In Figure 4.6, we illustrate the performance of the proposed DRA method. The first diagram at the top shows the incoming request rate $\lambda(t)$ and the admission rates $u_i^{real}(t)$, $i = 1, 2, 3$, towards the small, medium, and large resource profile, respectively. at each time slot. As expected, $\lambda(t)$ is distributed according to the AIMD-based scheduling solution Eq. (4.1), while the workload share of each resource profile depends on the ratio $\frac{\alpha_i}{\sum_{i=1}^{n} \alpha_i}$. In particular, the average admission rates of the small, medium and large resource profile, respectively, for the entire experiment, are calculated as 12%, 31% and 57%, which are consistent with the ratio $\frac{\alpha_i}{\sum_{i=1}^{3} \alpha_i}$. The lengths of individual queues of the application pods denoted by $w_i(t)$ are depicted in the second diagram. Evidently, when the workload is peaked, individual queues grow reasonably indicating an increase in local backlog. The AIMD-based service rates $\gamma_i(k)$ for each resource profile, as obtained from Eq. (4.3), are shown in the third diagram from the top. Recall that $\gamma_i(k)$ is a strictly increasing function of $w_i(t)$ as shown in Eq. (4.3). This coupling is clearly demonstrated in the second and

Table 4.4: Results for the five experiments.

Setup	**Total QoS Violations**	**Average CPU cores**
DRA	1.17%	13.1
m-HPA	0.52%	16.2
S-HPA	3.64%	14.8
M-HPA	0.93%	14.7
L-HPA	0.2%	14.2

third diagrams. The ML-based scaling decision is shown at the bottom diagram. The number of deployed CPU cores for each resource profile, is illustrated at each time slot, and the scaling decision takes into account the behavior of all previously shown metrics. The rest features of the ML-based profiles are omitted as they do not add any value to the discussion of the results.

In Figure 4.7, the utilized CPU cores needed to serve the incoming requests at each time slot are illustrated, for all methods. The HPA-based solutions, i.e., S-HPA, M-HPA, and L-HPA utilize on average 14.8, 14.6, and 14.1 CPU cores respectively, throughout the experiment. The M-HPA and L-HPA provide negligible QoS violations, i.e., 0.2% and 0.9% respectively, however on the peak of the workload both need at least 28 CPU cores to operate. This occurs because of the lack of a task scheduling algorithm that leads to instantiating under-utilized cores. Nevertheless, for the S-HPA solution, which intuitively could be the most fine-grained scaling solution, we can notice that it has on average the same performance in terms of CPU cores, however, leading to a significant increase in lost requests, namely, 3.64% of the total. This is reasonable as the small resource profile can serve only 2.5 requests per second as shown in Table 4.3. For the m-HPA, we can assume that under the assistance of the task scheduling policy, the total QoS violations are minimized, however, the average CPU cores are maximized, leading to 16.1 CPU cores. As one can notice, our method outperforms all other setups, utilizing on average 12.6 CPU cores to handle the incoming varying workload, having only 1.8% total QoS violations. Hence, the provided solution outperforms all others, having at least 8% fewer CPU cores utilized throughout the experiment. It is evident, that the task scheduling mechanism is key to handle the time-varying workload and optimizing the utilization of the deployed resources. Table 4.4 summarizes the results of the experiments.

4.9 Summary

This chapter presents a scalable architecture for resource management in KEC. The proposed method provides a solution to the task scheduling problem. Also, based on the theoretical results from the proposed AIMD-like algorithm and various performance metrics, we introduce an ML-based Application Profiling mechanism that decides the number of replicas for the different resource profiles to proactively and in a decentralized manner, scale the deployed resources to serve the incoming workload. Our framework outperforms other commonly used solutions for autoscaling, as the average CPU resources are at least 7% less, having only a slight increase in QoS violations. The results of this Chapter are also presented in [120].

Chapter 5

A switching offloading mechanism for path planning and localization in robotic applications

5.1 General Setting

For optimizing offloading the tradeoffs between computing and communication resources must be investigated with a focus on control design, estimation, and implementation [55], [121]. Recently, a specific effort is placed on exploring the offloading opportunities of the decision-making and monitoring algorithms (control and estimation) in the path planning problem for autonomous agents [29]. To this end, a control-based, computation offloading mechanism for robotic applications in Edge Computing ecosystems are a convenient way to optimize offloading. Also, following the paradigm of approximate computing, one may decide to reduce the computations of the specific algorithm having as a return a slight decrease in the overall accuracy or quality.

In this chapter, we propose a computation offloading mechanism for robotic applications. In particular, we realize an IoT-enabled localization and path planning framework and verify the expected gains of computation offloading by utilizing a real Edge Computing setting. To achieve this, we design and implement local and remote localization and path planning controllers, followed by a scheduling mechanism. The offloading mechanisms are treated as switches, leading to different dynamics of the resulting closed-loop system. Specifically, the algorithms involved in the localization process are decided to run remotely, rather than locally, when the uncertainty of the robot's pose is high and at the same time the network and computing resources status at the Edge is favorable. On the other hand, path planning is offloaded when the robot navigates in a part of a map where better planning strategies can be achieved through involved algorithms that can only be executed remotely. These switches compose a switching system that is adaptive and can operate under different scenarios and applications. This architecture perspective, which constitutes the main contribution of this chapter, offers our framework a degree of contextual awareness; that is the ability to sense and dynamically adapt to the robot's environment, implicitly enhancing to an extent the robustness of its operation, as well as improving the QoS of the supported applications.

5.2 Related Work

Computation offloading in current and next-generation networks is becoming increasingly important due to the proliferation of the Internet of Things (IoT) real-world applications [122]. These applications introduce a vast number of low-capability, low-energy devices to the networking ecosystem, which regularly need to perform computationally intensive and/or energy-hungry tasks. However, when latency and energy consumption minimization are required, the limited resources of the IoT devices prove inadequate [123]. For example, in Industry 4.0 and especially in collaborative robotics, where humans and robots work together in dynamic environments, computationally heavy algorithms enable IoT devices in sensing and actuating[124]. Consequently, large amount of information has to be processed and complex algorithms need to be executed in real-time.

The increasing availability of networking in the Edge and Cloud supports new approaches, where processing is performed remotely, with access to extensive computing and memory resources. In this direction, Edge Computing (EC) alongside Fog Computing (FC)[125] constitutes a particularly prominent way of dealing with the aforementioned shortcomings of IoT devices. FC offers an attractive alternative providing low-latency and high energy-efficient operation, while maximizing system performance. This paradigm is currently more relevant than ever, especially in the context of the much-anticipated Industry 4.0 revolution [126] and Industrial IoT (IIoT), where Fog Robotics (FR) is introduced. FR can be defined as the architecture that distributes computing, storage and networking functions at the Edge/Cloud continuum in a federated manner[127], i.e., where robots and automation systems rely on data or code from a network to support their operation.

Suitable as it may seem, solely utilizing remote computational resources is not enough; a number of unwanted phenomena potentially take place in the transmission and processing of the information, such as network latency, variable Quality of Service (QoS), and downtime. For these reasons, autonomous mobile robots often have some capacity for local processing when targeting low-latency responses, and during periods where network access is unavailable or unreliable. Consequently, a major challenge, from a control design, estimation, and network optimization point of view is to combine local and remote resources in an efficient way.

Open challenges in this area throughout the literature are concerned with developing adaptive multi-robot/machine control, capturing, modeling, predicting, and anticipating the agent's interactions, and designing distributed control and path planning algorithms that deliver flexible and safe working environments. Approaches similar to ours include [128], where gesture-based semaphore mirroring with a humanoid robot is split to remotely and locally executed functionality; [129], in which the authors identify a three-layered environment (Robot, Edge, and Cloud) to overcome the challenges of network limits in a Deep Robot Learning application and [130] where Dew Robotics is introduced; this concept posits that critical computations are executed locally so that the robot can always react properly, while less critical tasks are moved to the Fog and Cloud, so to exploit the larger availability in computing, storage, and power supply. However, none of the aforementioned offloading decision schemes addresses the dynamic nature of the robot's environment.

5.3 Contributions and Outline

The scenario addressed in this chapter involves a mobile robot equipped with sensing, computing, and wireless communication capabilities, which makes its way from a starting position to a target position in an operating ground (e.g., a factory floor), navigating through obstacles. This functionality is a key component to realizing autonomous robotic navigation in Industry 4.0 use cases, e.g., warehousing and logistic robots which automate the process of storing and moving supply chain goods. Tracking the robot's location is essential for robust and safe trajectory planning. However, a common problem in such a scenario is that the uncertainty in estimating the exact pose (i.e., position and orientation) grows over time in motion, due to inaccuracies in sensing, wheel slips, and hardware failures. Thus, the importance of an accurate, dynamically adjusted localization technique is evident. The key contributions of this work that differentiate it from the rest of the literature are summarized as follows:

- A novel computation offloading mechanism for robotic applications that utilizes an Edge Computing setting to improve the accuracy of both the robot's localization and trajectory.
- An offloading scheme, based on switched systems, that addresses the dynamic nature of the robot's movement and deals with the unpredictability in its exact pose over time.
- An innovative position and orientation estimation component that achieves high precision while using the simplest camera system and the minimum amount of identified natural landmarks.

5.4 Architecture Overview

The scenario addressed in this work involves a mobile robot equipped with sensing, computing, and wireless communication capabilities, which makes its way from a starting position to a target position in an operating ground (e.g. a factory floor), navigating through obstacles. This functionality is a key component to realizing autonomous robotic navigation in Industry 4.0 use cases, e.g. warehousing and logistic robots which automate the process of storing and moving supply chain goods. Tracking the robot location is essential for a robust and safe trajectory planning. However, a common problem in such a scenario is that the uncertainty in estimating the exact pose (i.e., position and orientation) grows over time in motion, due to inaccuracies in sensing, wheel

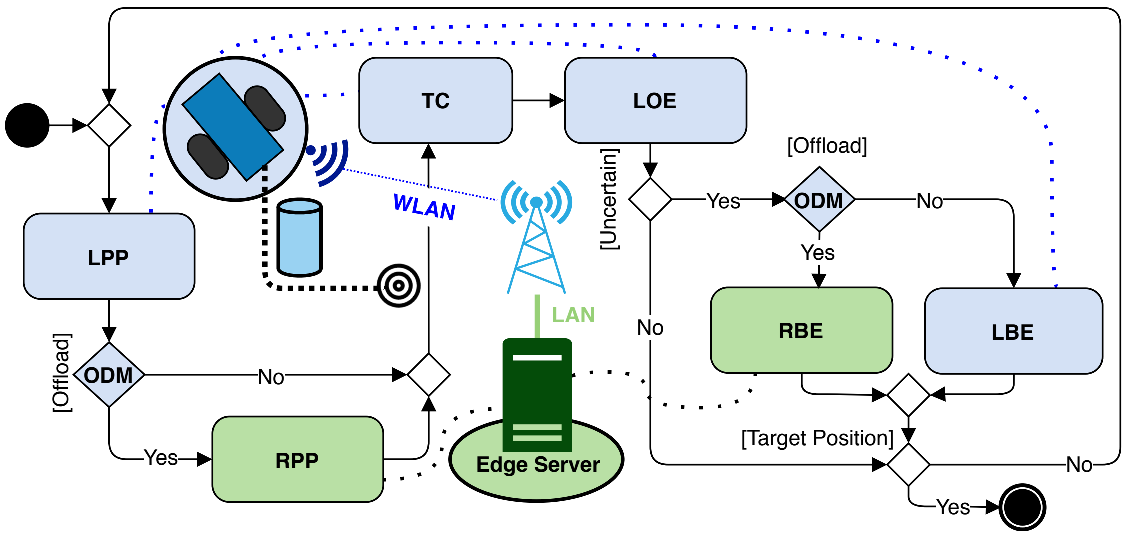


Figure 5.1: Architecture Overview. The locally executed components are highlighted with blue color, while the remotely executed ones with green.

slips, hardware failures, etc., [131]. Thus, the importance of an accurate, dynamically adjusted localization technique is evident.

In our case self-localization through landmark assisted pose estimation is implemented; the robots are equipped with a camera module, while in their proximity unique cylindrical beacons are used as landmarks to assist in the pose estimation process. In the computationally demanding involved algorithms, two offloading opportunities are revealed in, namely, pose estimation and path planning. To this purpose, a small-scale network infrastructure is set up, connecting the robot to a wireless LAN (WLAN) through an Access Point located within the robots' network range, which in turn connects via a wired connection (LAN) to a server in the robot's proximity, the Edge Server.

Locally, the intangible assets include the (i) the Tracking Controller (TC), (ii) the Local Odometry-Based Estimator (LOE), (iii) the Local Beacon-Based Estimator (LBE), (iv) the Local Path Planner (LPP) and (v) the Offloading Decision Mechanism (ODM) components, all located within the robot; component (i) is responsible for carrying out movement-related decisions, (ii), (iii) and (iv) are the locally executed pose estimation and path planning applications respectively and (v) encompasses the intelligence of our switching system by monitoring the offloading-related metrics and realizing the offloading decisions. On the remote side, containerized counterparts of the path planning and pose estimation applications are co-hosted on the Edge Server; these are namely (vi) the Remote Beacon-Based Estimator (RBE) and (vii) the Remote Path Planner (RPP) which are able to receive offloaded requests from the robot. A more detailed discussion on these components follows in Sections 5.5, 5.6 and 5.7.

In order to outline the sequence of interactions between the main components of the architecture, we showcase a representative scenario in which our solution applies successfully. Fig. 5.1

depicts an overview of this scenario. Without loss of generality, we assume that only one robot operates in the field. Also, its starting pose, the operating space dimensions and the obstacles' and beacons' positions and shapes are considered known a priori.

A typical activity flow of our scenario, initiates with Local Path Planner component calculating locally a trajectory from the starting position to the target position. This triggers the ODM for the first time; should a quick analysis on the projected trajectory indicate room for significant refinement of the selected path, the Remote Path Planner is invoked. This analysis is based on the trajectory curvature and the degree in which the more elegant remote component is potentially able to smooth it around obstacles; Section 5.7.3 provides more insight on this process. Eventually, the resulted trajectory dictates the *intermediate positions* the robot needs to reach. In order to sequentially perform the transition to the each of them, the Tracking Controller component is invoked.

After reaching the next position of its trajectory, an uncertainty indicator of the pose estimation is calculated; this indicator is a scalar that grows with time and actually accumulates the error between the estimated and the reference pose after each move, as explained thoroughly Section 5.7.1. Here, the second decision occurs; if this indicator measures bellow a predefined threshold, the robot continues to move based on the feedback coming from the Tracking Controller's monitoring process, i.e., the Local Odometry-Based Estimator, which leverages the robot's photoelectric sensors (encoders) attached to each wheel to measure the wheels' angular velocities during a period of time. Else, it invokes the more precise, but computationally heavy, Beacon-Based Pose Estimator, leveraging information coming from the beacons in the environment. That triggers the ODM once again; the Edge Server is queried to provide an estimation on the duration of the potentially offloaded pose estimation task. As described by the mathematical modelling in Section 5.7.2.1, this duration is proportional to the availability of the computational resources. Based on this estimated duration, a decision is made on whether to offload the pose estimation task to the Remote Beacon-Based Estimator, or execute it locally. The flow ends with the robot checking if the target position is reached. If not, it reverts to first step.

It is worth highlighting that the tracking controller, as well as the path planning and pose estimation, are aperiodic. The position of the robot on the operating ground is defined by the state vector $\boldsymbol{x^i} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^\top$. The robot has to move towards the next reference position $\boldsymbol{x^i_{ref}} = [x_{1,\text{ref}}(t_i) \quad x_{2,\text{ref}}(t_i)]^\top$, generated by the path planning algorithms, to approach the target position. Fig. 5.2 gives a brief insight on the timing sequence in which the rest of the sections will refer to. Let subscript i correspond to the step during which the robot reaches the next reference position in k_i actuation steps, while simultaneously tracking its pose. In particular, at time t_i^0 the robot is in the position x^i. When the next reference position x^{i+1}_{ref} is close, the uncertainty about the current estimation is calculated. Thus, the time duration T_i^1 corresponds to the time spent for localization. When the local odometry-based estimator is used, this time is equal to zero, while the beacon-based estimation algorithm is time-consuming. The time duration T_i^2 corresponds to the path planning algorithm running time either remote or local, which generates the next reference

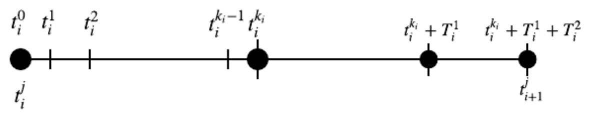


Figure 5.2: The timing sequence in the proposed scenario.

position. Similarly, the time to execute the local path planning algorithm is equal to zero.

5.5 System Dynamics

5.5.1 Robot dynamics

The differential drive robot used in this study has two wheels that can turn at different rates, allowing motion by changing the orientation and the position (x_1, x_2) either separately or simultaneously. For the robot dynamics, the 2D coordinates, i.e., position, and the orientation of the robot are denoted by the state variables z_1, z_2 and z_3. Hence we consider $\boldsymbol{z} = \begin{bmatrix} z_1 & z_2 & z_3 \end{bmatrix}^\top = \begin{bmatrix} \boldsymbol{x}^\top & \theta \end{bmatrix}^\top$. The robot is controlled by the angular velocities w_R and w_L, accounting for the right and left wheel respectively. The robot dynamics is defined by the following continuous time system, based on the work in [132], using the aforesaid state-space representation. Specifically, we have for any $t \geq 0$,

$$\dot{z}_1(t) = \frac{r}{2}(w_L(t) + w_R(t)) \cos z_3(t), \tag{5.1}$$

$$\dot{z}_2(t) = \frac{r}{2}(w_L(t) + w_R(t)) \sin z_3(t), \tag{5.2}$$

$$\dot{z}_3(t) = \frac{r}{l}(w_L(t) - w_R(t)), \tag{5.3}$$

where l, r are the distance between the two wheels and the radius of each wheel respectively. The odometry measurements $\tilde{w}_L(t_i^j)$,$\tilde{w}_R(t_i^j)$ are taken at each time instant t_i^j, $i = 0, 1, ...,$ $j = 0, \ldots, k_i$ of the timing sequence introduced in Section 5.4. The corresponding discretized system using Euler forward method is:

$$\tilde{z}_1(t_i^{j+1}) = \frac{r}{2}(\tilde{w}_L(t_i^j) + \tilde{w}_R(t_i^j)) \cos \tilde{z}_3(t_i^j)(t_i^{j+1} - t_i^j) + \tilde{z}_1(t_i^j), \tag{5.4}$$

$$\tilde{z}_2(t_i^{j+1}) = \frac{r}{2}(\tilde{w}_L(t_i^j) + \tilde{w}_R(t_i^j)) \sin \tilde{z}_3(t_i^j)(t_i^{j+1} - t_i^j) + \tilde{z}_2(t_i^j), \tag{5.5}$$

$$\tilde{z}_3(t_i^{j+1}) = \frac{r}{l}(\tilde{w}_L(t_i^j) - \tilde{w}_R(t_i^j))(t_i^{j+1} - t_i^j) + \tilde{z}_3(t_i^j). \tag{5.6}$$

5.5.2 Tracking controller

As previously mentioned, the robot moves towards the next reference position $\boldsymbol{x_{ref}^i}$ to reach the target position. For this actuation phase, given the specific robot dynamics, we propose a tracking controller executed locally on the robot, by fixing the control inputs w_L, w_R to be either equal or opposite. Therefore, the control input is w, while $|w| = |w_L| = |w_R|$. As a result, we restrict the motion of the robot to a straight line, i.e., "translational motion", or a rotation around the center of the wheel axle, i.e., "rotational motion", respectively. This control structure is chosen as it is

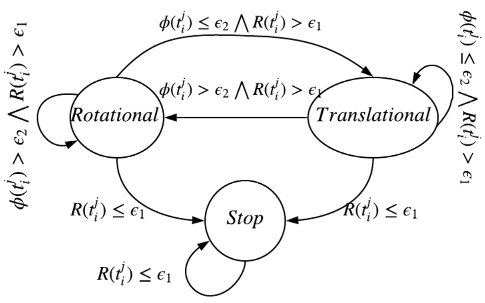

$\phi(t_i^j) \le \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1$
$\phi(t_i^j) > \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1$
$\phi(t_i^j) > \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1$
$\phi(t_i^j) \le \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1$
Rotational
Translational
Stop
$R(t_i^j) \le \epsilon_1$
$R(t_i^j) \le \epsilon_1$
$R(t_i^j) \le \epsilon_1$

Figure 5.3: The hybrid automaton of our system.

efficient for tracking purposes, leading to a simple structure of the closed-loop system. Specifically, the closed-loop dynamics for the translational and rotational motion are

$$S_1^{Tran}: \begin{cases} \dot{z}_1(t) = \frac{r}{2}(w(t))\cos z_3(t), \\ \dot{z}_2(t) = \frac{r}{2}(w(t))\sin z_3(t), \\ \dot{z}_3(t) = 0 \end{cases} \tag{5.7}$$

$$S_2^{Rot}: \begin{cases} \dot{z}_1(t) = 0, \\ \dot{z}_2(t) = 0, \\ \dot{z}_3(t) = \frac{r}{l}(w(t)), \end{cases} \tag{5.8}$$

where S_1^{tran} is used for the translational motion and S_2^{rot} when the robot needs to rotate. Let $R(t_i^j) = \left\| \begin{bmatrix} \tilde{z}_1(t_i^j) \\ \tilde{z}_2(t_i^j) \end{bmatrix} - \begin{bmatrix} z_{1,\text{ref}}(t_i) \\ z_{2,\text{ref}}(t_i) \end{bmatrix} \right\|_2$ be the distance between the robot's current estimation and the reference position and let $\phi(t_i^j) = \tilde{z}_3(t_i^j) - \tan^{-1}\left(\frac{\tilde{z}_2(t_i^j) - z_{2,\text{ref}}(t_i)}{\tilde{z}_1(t_i^j) - z_{1,\text{ref}}(t_i)}\right)$ be the angle between the robot's current estimation of orientation and the line connecting the robot and the reference position. Here, $\tilde{z}$ accounts for the estimation of its current pose calculated by Equations (5.4) – (5.6) at the time period of the actuation $t = t_i^j$, $j = 0, 1, \ldots, k_i$.

The closed-loop system with the tracking controller can be modeled by a discrete-event systems, see, e.g., [133], as shown in Fig. 5.3, where the control input can be calculated as follows:

$$w(t_i^j) = \begin{cases} L_1 R(t_i^j), & \phi(t_i^j) \le \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1, \text{Translational}, \\ L_2 \phi(t_i^j), & \phi(t_i^j) > \epsilon_2 \bigwedge R(t_i^j) > \epsilon_1, \text{Rotational}, \\ 0, & R(t_i^j) \le \epsilon_1, \qquad\qquad \text{Stop}. \end{cases}$$

The quantities ϵ_1, ϵ_2 are positive constants, while the gains L_1, L_2 are constant control parameters.

The reference position is reached when the estimation of its position is close, and in particular is inside a ball of radius ϵ_1 close to the reference, i.e., centered at $\mathbf{B}(\boldsymbol{x^i_{ref}}, z(t_i^j)) = \{z \in \mathcal{R}^3 : \|z - \tilde{z}(t_i^j)\| \leq \epsilon_1\}$. The effect of the uncertainty is taken into account explicitly in the offloading decision that follows.

5.6 Localization and Path Planning

In what follows, we present the algorithms chosen for localization and path planning, with a varying degree of complexity and accuracy, that are implemented locally and remotely accordingly.

5.6.1 Localization

The localization problem is equivalent to the pose estimation problem in our setting. Two algorithms of different complexity are implemented, namely, (i) an odometry-based one, and (ii) a camera-based estimation. The first estimation algorithm is light enough to run efficiently on the robotic platform. Roughly, the robot's on-board wheel encoder readings are fed to the motion model (5.4) – (5.6). While this is a lightweight and fairly accurate localization technique when it comes to short trajectories, odometry is known to be prone to accumulative errors [134].

The second localization technique is the computationally heavier beacon-based estimator. Details on the technical parts of the algorithm and its software and hardware implementation can be found in [135]. Roughly, the technique is based on a bilateration method using principles of projective geometry. Distance calculation is based on feature extraction from pictures depicting the landmarks, with the localization algorithm relying on a minimum of two strategically positioned landmarks. To address this requirement, the attached camera scans the area in front of the robot, capturing pictures and analyzing them until two landmarks are detected. Hence, computationally intensive, real-time image processing is required to achieve highly accurate results. Relevant works include [136] and [137].

5.6.2 Path Planning

Many works exist in the literature addressing the path planning problem; realistic robot navigation and smooth trajectory planning is a major challenge [138], [139]. Planning algorithms generate a trajectory consisting of intermediate reference positions to reach the final target position. In this work, we select and adapt graph-based methods of varying complexity, see, e.g., [132, Chapter 8]. As a result, the algorithms described below, take as input a graph that represents the real-space grid space along with the target positions, the obstacles, and the starting position. This grid has a predefined cell size, that depends on the length of the robot. Each cell corresponds to a possible reference position. In our case, the obstacles are rectangular-shaped, in the sense of simplicity, however, arbitrarily shaped obstacles could also be included.

On the one hand, a lightweight implementation of the $A^\star$ algorithm [140] acts as the Local Path Planner. Similar to [141], four directions of movement are allowed in the grid. The cells containing obstacles are not connected with the neighboring cells. The $A^\star$ algorithm returns a sequence of positions to reach the target position, according to a heuristic cost function; in our case this is the Manhattan Distance. The implementation is suitable for a robot with minimal computational resources providing a solid and quick solution, however the generated trajectory is not smooth.

The computationally intensive algorithm acting as the Remote Path Planner is deployed on the Edge Server. Similar to [142], the main process of the proposed algorithm is to locate a possible move towards a node that is closer to the target given the aforesaid graph. To this purpose, a multiple sources single destination problem is solved, utilising Dijkstra's shortest path algorithm, which calculates a path from each node towards the target position, offline. These precalculated paths, along with the total cost to reach the desired destination, are stored in a database on the server's startup. When the Remote Path Planner is invoked, given the current location of the robot, a neighbor pruning is performed similarly to [143]. A node of the graph is considered to be a neighbor of the current position if (i) the distance between them is less than twice the specified cell size and (ii) no obstacle is in the line of sight of the current position to that node. Consequently, to retrieve the set of possible neighbors, it is sufficient to search for the avoidance of line clipping (intersection) between the line connecting the current position to each of the adjacent cells and the set of obstacles present in the real-space grid. The optimal path is chosen by comparing all possible neighbors. In particular, the cost to reach each one of them from the current position is added to the cost from each neighbor to reach the desired target. In this way, the algorithm allows "shortcuts' to the neighboring nodes, while any-angle trajectories are feasible.

5.7 Switching System

In this section, we present the switching mechanisms that are realizing the Offloading Decision Mechanism of our framework. We assume that starting from a position $\boldsymbol{x_0} = [x_1(0) \quad x_2(0)]^\top$, the closed-loop system converges asymptotically to a reference position $\boldsymbol{x_{ref}} = [x_{1,\mathrm{ref}}(t_i) \quad x_{2,\mathrm{ref}}(t_i)]^\top$ when exact measurements are available, i.e., when $\tilde{\boldsymbol{z}}(t) = \boldsymbol{z}(t)$. We identify two offloading opportunities related to the pose estimation and the path planning problem. In Fig. 5.4 the proposed switching system is presented. In particular, switches $\mathcal{S}_1$ and $\mathcal{S}_2$ relate to the estimation procedure, and switch $\mathcal{S}_3$ concerns the path planning part.

5.7.1 Sensor Selection (Switch 1)

The measurements of the onboard sensors are imperfect, thus the pose estimation error is accumulated. When the error becomes too large, the more precise, yet more computationally intensive remote localization algorithm is invoked. In order to decide when to offload, we introduce the

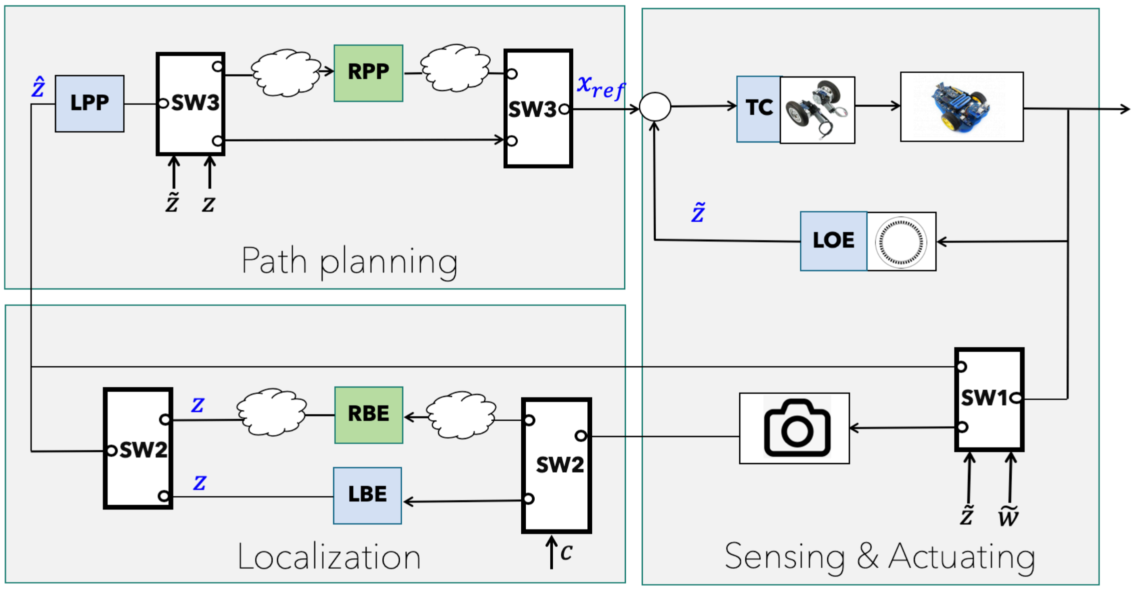


Figure 5.4: The block diagram of the switching system. Component abbreviations and colors follow the pattern introduced in Section 5.4.

variable $\delta(\cdot)$ that describes the uncertainty in estimation. We set

$$\delta(t_i^{j+1}) = \delta(t_i^j) + b_0 + b_1\tilde{\delta}(t_i^j),$$

$j = 1, \ldots, k_i$, $i \in \mathcal{N}$, where $\tilde{\delta}$ is the deviation between the measurements of the states $\tilde{z}$, computed by the Equations (5.4) – (5.6) and the model-based estimations $\check{\boldsymbol{z}}$, i.e.

$$\tilde{\delta}(t_i^j) = \left\|\check{\boldsymbol{z}}(t_i^j) - \tilde{\boldsymbol{z}}(t_i^j)\right\|_2,$$

where $\check{\boldsymbol{z}}(t_i^j)$ consists of:

$$\begin{aligned}
\check{z}_1(t_i^{j+1}) &= \frac{r}{2}(w_L(t_i^j) + w_R(t_i^j)) \cos \check{z}_3(t_i^j)(t_i^{j+1} - t_i^j) + \check{z}_1(t_i^j),\\
\check{z}_2(t_i^{j+1}) &= \frac{r}{2}(w_L(t_i^j) + w_R(t_i^j)) \sin \check{z}_3(t_i^j)(t_i^{j+1} - t_i^j) + \check{z}_2(t_i^j),\\
\check{z}_3(t_i^{j+1}) &= \frac{r}{l}(w_L(t_i^j) - w_R(t_i^j))(t_i^{j+1} - t_i^j) + \check{z}_3(t_i^j),
\end{aligned}$$

which are the model-based estimation of the dynamics at time instants t_i^j, $j = 1, \ldots, k_i$ and w_L,w_R are the outputs of the tracking controller. At the time t_0^0, the model-based estimation is equal to a known initial position, i.e., $\check{\boldsymbol{z}}_1(t_0^0) = \check{\boldsymbol{z}}_1^0$. As a result, δ linearly depends on the deviation and gets bigger as the robot actuates, especially when the actual motion of the robot differs from what the model dictates.

The offloading mechanism, aiming to reset the uncertainty, is triggered when δ becomes too large, namely larger than a prespecified threshold $\delta^\star$, i.e.,

$$S_1(t_i^{k_i}) = \begin{cases} \text{OFF}, & \text{if } \delta(t_i^{k_i}) \leq \delta^\star, \\ \text{ON}, & \text{else}, \end{cases}$$

where k_i refers to the time instant, when the robot's position, calculated by Equations (5.4) and (5.5), is close to the next reference position $\boldsymbol{x}_{\mathbf{ref},\boldsymbol{k}}$. Moreover, ON corresponds to using the beacon-based localization and OFF to proceeding based on the local odometry estimation. In the scope of this work, we assume that the uncertainty becomes equal to zero when the beacon-based localization is used. Hence, when $S_1(t_i^{k_i}) = \text{ON}$, then $\delta(t_{i+1}^0) = 0$, which means we get a valid measurement of the states $\boldsymbol{z}$. Otherwise, $\delta(t_{i+1}^0) = \delta(t_i^{k_j})$.

5.7.2 Estimation Offloading (Switch 2)

Switch S_2 decides whether the localization algorithm will be executed locally on the microcontroller mounted on the robot, or remotely on the Edge Server. Although the execution of such a computationally heavy algorithm on a battery-powered IoT device is energy-consuming, it may be preferable in some cases as offloading might result to larger response times due to lack of available resources on the remote server and network congestion.

5.7.2.1 Resource modelling and estimation

We assume that the resources of the localization service on the Edge Server are managed by the resource orchestrator of the infrastructure provider and we can only estimate the allocated resources through measurements. Thus, we model the resource allocation strategy on the Edge Server as a linear dynamical system subject to process and measurements uncertainty disturbances

$$\begin{aligned} c((k+1)T_s) &= c(kT_s) + w(kT_s), \\ z(kT_s) &= c(kT_s) + v(kT_s), \end{aligned}$$

where c accounts for the virtual CPU cores of the container, z is the measurement of c and T_s is a constant sampling time. The terms w, v are the process and measurement noise respectively, both following a normal distribution. Based on previous measurements, we compute a current estimation of the virtual CPU cores allocated to the container, $\hat{c}$, by applying a Kalman Filter [144], which is a computationally light prediction method.

5.7.2.2 Processing time estimation

Having acquired the estimation of the available remote virtual CPU cores $\hat{c}$, the estimated processing time of the beacon-based localization algorithm can be calculated. To this purpose, the processing time, t_p is modeled as a linear relationship of the available resources, $t_p = a\hat{c} + b$. The coefficients a,b are calculated using the least squares fitting method, on a set of pairs $(t_p, \hat{c})$ produced offline while experimenting with a dataset of pictures. Moreover, we consider the wireless network induced delay t_{net} to be constant as a standard network delay in a WLAN network.

5.7.2.3 Localization Offloading

The processing time is related directly to the CPU availability. The local beacon-based localization has an average time t_{loc} to be executed based on the robot's resources. Hence, Switch S_2 is formulated as:

$$S_2(t_i^{k_i}) = \begin{cases} \text{ON}, & \text{if } t_p + t_{net} \leq t_{loc}, \\ \text{OFF}, & \text{else}, \end{cases}$$

where k_i refers to the time instant that the robot must decide whether to offload or not the beacon-based localization algorithm. Moreover, ON corresponds to the remote execution of the self-localization algorithm and OFF to the local execution.

5.7.3 Path Planning Offloading (Switch 3)

Two path planning algorithms are implemented. By default, the computationally light $A^\star$ algorithm presented in Section 5.6.2,provides a reference trajectory on the robot. However, whenever a prediction cost indicates a possible amelioration by choosing a more refined path, the remote path planning algorithm is invoked. Both algorithms take as input the current estimation of the position and the reference position and generate a reference trajectory.

The offloading decision for the path planning depends on a cost consisting of two parts; the first part estimates the closeness of the generated reference trajectory to obstacles and the second part evaluates the curvature of the trajectory. Both terms follow theoretical aspects from standard works, e.g., [145]. We define the function $D(\boldsymbol{x})$ that quantifies the "density" of obstacles according to the estimation of the current position $\hat{\boldsymbol{x}}$, either computed by the beacon-based localization or the local odometry measurements.

$$D(\boldsymbol{x}) = \sum_{\hat{\boldsymbol{x}}_{\mathbf{obs}} \in \mathcal{X}_{\text{obs}}} \exp\left(-\|\boldsymbol{x} - \boldsymbol{x}_{\mathbf{obs}}\|\right),$$

and $\boldsymbol{\mathcal{X}}_{\mathbf{obs}}$ is the set of positions that correspond to the centers of the cells that are unreachable, e.g., occupied by an obstacle.

Let $\{\check{\boldsymbol{x}}(i)\}_{i=1,\dots,M}$ be the part of the path sequence consisting of the first M positions, generated by the local path planning algorithm.

The local path planning algorithm takes as input the current position estimation $\hat{\boldsymbol{x}}(t_i^{k_i})$ at $t = T_i^{k_i} + T_i^1$ and creates a reference trajectory sequence $\{\check{\boldsymbol{x}}(i)\}_{i=0,1,\dots,M}$, with $\check{\boldsymbol{x}}(0) = \hat{\boldsymbol{x}}(t_i^{k_i} + T_i^1)$.

We define:

$$J_{\text{local}}(\hat{\boldsymbol{x}}(t_i^{k_i} + T_i^1)) = \\ \sum_{i=0}^{M-1} \left(\left\| \check{\boldsymbol{x}}(i+1) - \check{\boldsymbol{x}}(i) \right\| \right) - \left\| \check{\boldsymbol{x}}(M) - \check{\boldsymbol{x}}(0) \right\|,$$

as a cost describing the curvature of the reference local trajectory. The offloading strategy can be formulated as:

$$S_3(t_i^{k_i} + T_i^1) = \\ \begin{cases} \text{OFF}, \text{ if } D(\hat{\boldsymbol{x}}(t_i^{k_i} + T_i^1)) - J_{\text{local}}(\hat{\boldsymbol{x}}(t_i^{k_i} + T_i^1)) \leq J^\star, \\ \text{ON}, \text{ else}, \end{cases}$$

where $t_i^{k_i} + T_i^1$ indicates the time instant after the actuation and pose estimation. The constant $J^\star$ accounts for the degree of difficulty of the next moves in terms of proximity to obstacles and curvature of the trajectory. When S_3 in ON, the remote path planning provides the next step to reach the target position. Otherwise, the robot relies on the local path planning trajectory. It should be mentioned that, contrary to Switch 2, here, we do not include the CPU availability in the offloading decision, as we noticed that the remote path planner chosen is mainly memory intensive.

5.8 Experiments and Evaluation

The experiments were conducted in an operating space of 2.5×2.5 meters, divided by 25×25 cells, with a cell size of 10×10cm. The robot chosen was the commercially available AlphaBot[1], equipped with a Raspberry Pi 3 device as the control unit. The length of the AlphaBot is 22cm and the radius of each wheel is 6.6cm. The coloured beacons were placed at the periphery of the grid for the localization procedure described in Section 5.6. The rectangular-shaped obstacles were placed as depicted with grey colour in Fig. 5.5. The map is considered known. The Access Point used was a MikroTik wireless SOHO AP, providing up to 100Mbs LAN connection, Single Band (2.4GHz). The Edge Server deployed on the NETMODE, testbed part of Fed4FIRE[2] initiative, was equipped an Intel Atom CPU, up to 1Gbit Ethernet port and 8GB of RAM. The services provided by the edge server were deployed as Docker containers. For each Docker container, one can set constraints, to limit a given container's access to the host machine's CPU cores, by provisioning a percentage of them as the virtual cores of the containers. Thus, containers can be assigned with partial virtual CPUs using decimal values. Using a collection of pictures from the actual experimentation room, from different positions and viewing angles, a dataset was created

[1]https://www.waveshare.com/wiki/AlphaBot
[2]https://www.fed4fire.eu/testbeds/netmode/

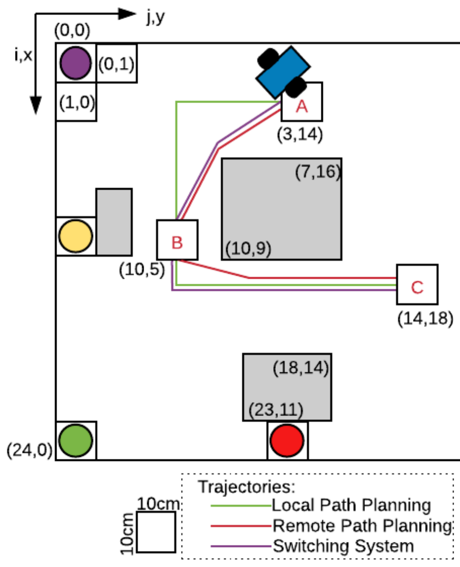


Figure 5.5: The experiment setup and the trajectories produced by the three experiments.

Average Time per picture (sec), t_p	Virtual Allocated Cores, $\hat{c}$
2.41	0.25
1.06	0.5
0.56	0.75
0.39	1
0.30	1.25
0.26	1.5

Table 5.1: The average time for remote beacon-based estimation per virtual allocated core to the container.

to estimate the time duration of the remote beacon-based localization. In Table 5.1, the values of the set of pairs $(t_p, \hat{c})$, introduced in Section 5.7.2, are presented. Using the least squares fitting method we calculated the coefficients $a = -1.34$ and $b = 1.675$. Hence, the estimated processing time of the remote beacon-based localization is given by $t_p = -1.34\hat{c} + 1.675$. Provisioning over 1.5 cores resulted in similar computation time, thus, the maximum CPU allocation was set to that value. In our experiments, the allocated cores of the containerized application were updated every 10sec, following a Normal Distribution with a mean value of 0.75 and 0.5 variance. The following values were used for the aforesaid constant values: $b_0 = 1$; $b_1 = 0.2$; $e_1 = 5\text{cm}$ $e_2 = 5°$, $L_1 = 0.2$, $L_2 = 0.6$, $\delta^\star = 6$ and $J^\star = 3$. Finally, the average network delay of the WLAN was empirically measured to $t_{net} = 1\text{sec}$ per offloaded picture and the average time for each picture to be processed locally on the AlphaBot was $t_{loc} = 3\text{sec}$.

Three experiments were conducted, namely, local only execution, remote only execution and the proposed switching offloading scheme. In Table 5.2 the average completion time and the average

Experiment	Average completion time (sec)	Success Rate
Local Only Execution	61	40%
Remote Only Execution	105	100%
Switching System	90	100%

Table 5.2: The average completion time and success rate of 10 experiments for each setting.

success rate for 10 experiments of each setting is presented. For the rest of the evaluation, we will present the results of the best trials for each setting. Moreover, in Fig. 5.5 the reference trajectories of these trials for the three experiments, are illustrated, with green colour for local only execution, red colour for remote only execution and purple colour for the switching system. As outlined in Section 5.6, the local $A^\star$ algorithm allows only four directions of movement, while the remote path planner allows any-angle movements. For better visualization, we uploaded timelapse videos[3] from the conducted trials for each setting. In these experiments, the starting position for the AlphaBot was the already known position $A(3,14)$, while the desired target reference positions were $B(10,5)$ and $C(14,18)$ in sequence. The scale of uncertainty is illustrated as a percentage of $\delta^\star$, i.e., $\delta/\delta^\star$, which is the predefined quantity for Switch 1 to be ON.

5.8.1 Experiment A - Local Only Execution

In the first experiment Switches 1 and 3 were ON, throughout the experiment and Switch 2 was never used. This setting results to a fast, although not precise navigation with $\delta/\delta^\star$ growing monotonically. The average duration was 61 seconds as the main time consuming process was the actuation. The amount of successful trials was low. Consequently, without a more sophisticated localization algorithm and a more precise path planning technique there is no guarantee the target reference position is reached.

5.8.2 Experiment B - Remote Only Execution

In the second experiment, whenever the uncertainty about AlphaBot's pose grew over the predefined threshold $\delta^\star$, beacon-based localization was invoked (Switches 1 and 2 ON) on the Edge Server. Moreover, the reference trajectory was always generated by the remote path planning algorithm (Switch 3 ON). In this setting, the robot always reached the target positions, as shown in Table 5.2, although the completion time was heavily affected, as shown in Fig. 5.6. Beacon-based localization was executed twice during this experiment and, as a result, $\delta/\delta^\star$ became equal to 0. The setup of the particular experiment underlines the importance of a slower but more precise navigation.

[3]https://github.com/Dspatharakis/alphabot-ppl/tree/master/timelapsed-videos

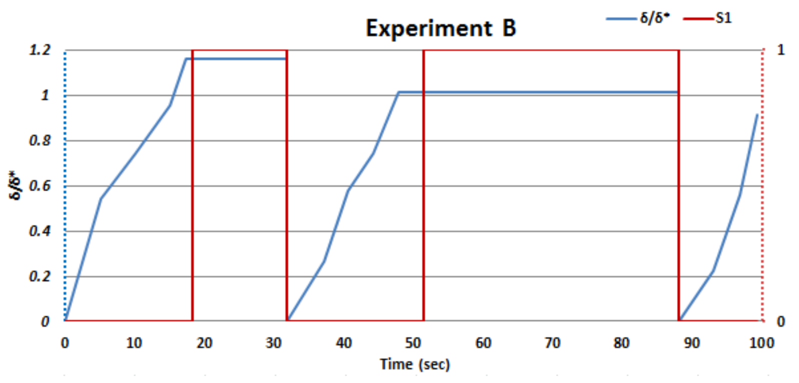


Figure 5.6: Experiment B - Remote Only Execution.

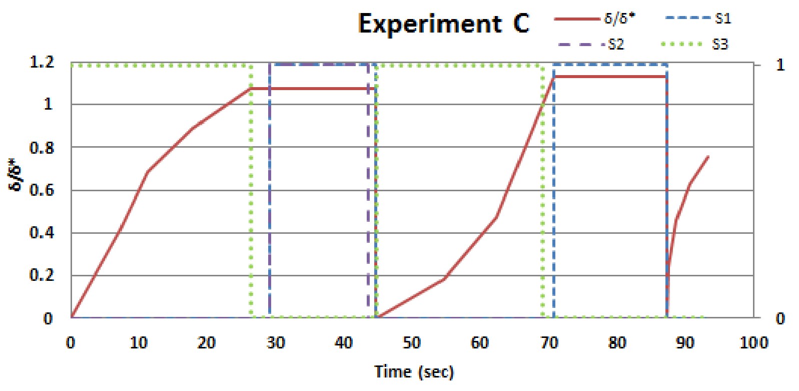


Figure 5.7: Experiment C - Switching System.

5.8.3 Experiment C - Switching System

As described in Section 5.7.3, Switch 3 decides which path planning algorithm solution the AlphaBot will use to generate the next reference position. When, the curvature function of the trajectory calculated by the $A^\star$ algorithm and the obstacle density function exceeded the threshold value $J^\star$, the remote path planning solution was selected; e.g., from the beginning of the experiment until the 25th sec of the simulation and from the 43rd sec till the 67th sec, as illustrated with green dashed line in Fig. 5.7. In the same figure, with red solid line, $\delta/\delta^\star$ is depicted. Two times during the experiment the more precise beacon-based estimation was invoked to reset $\delta/\delta^\star$. The first estimation attempt, at the 25th sec of the experiment, was executed on the Edge Server, because S2 was ON. The second one, at the 71st sec of the experiment, was executed locally, as S2 dictated (OFF), because the estimation of the CPU availability of the Edge Server, provided by the Kalman Filter, along with the network delay for each picture, at that time, would have provided worse results than the local execution. This setup provided a very precise and robust navigation for the robot, leading to a very high success rate of the experiments, achieving a balance between execution time and trajectory accuracy.

5.9 Summary

In this study, we introduced a switching offloading mechanism for localization and path planning applications of mobile robots. The offloading decision for localization is based on pose uncertainty and the availability of edge resources, while the offloading decision for path planning depends on the difficulty of the trajectory. The proposed framework achieves more precise navigation than the case of exclusive local execution of the applications, without paying the price of a slower execution time, like in the case of remote only execution of the algorithms. Also, it is modular and applicable to various scenarios, applications and objectives under the robot's dynamic environment. Our future work will focus on extending the proposed mechanism to more sophisticated control algorithms, providing theoretical guarantees for stability and convergence of the proposed robot's dynamics. Furthermore, we plan to develop more precise estimation and planning algorithms in multi-robot scenarios and more sophisticated control algorithms in the co-design setting that will take into account the available resources on the infrastructure side. The results of this Chapter are also presented in [146].

Chapter 6

Resource-aware Estimation and Control for Edge Robotics: a Set-based Approach

6.1 General Setting

This chapter focuses again on the case of Edge Robotics [147], which is widely used in 5G industrial verticals. Following the current trend in service delivery, Edge Robotics leverages the computing capabilities of Edge Computing to achieve low-latency communication [146]. In this chapter, the mobile robot considered is a unicycle, subject to modeling and measurement uncertainties. In the proposed scenario, the robot must solve a state estimation, also called localization, problem and subsequently a trajectory tracking problem. Under this setting, an offloading mechanism is available for transmitting the sensing data of the localization procedure to an edge server for further processing. Following recent works in the literature, [148], a set-based estimation approach is considered, as our main concern is to provide deterministic guarantees on the robot's estimated pose and subsequently guarantee convergence towards a target waypoint. Although existing elegant solutions for control of unicycle robots exist, e.g., [149], [150], incorporating set-based estimation methods into the controller design to provide real-time navigation, is quite challenging. The proposed algorithm aims to provide a set-based estimation offloading mechanism in the context of Edge Robotics. Under this complex scenario, the fundamental trade-off between performance and consumed resources is investigated, along with the conditions that guarantee the system's convergence. All in all, we extend the modeling of Chapter 5, and provide closed-loop guarantees for the convergence of the robot and optimal resource utilization.

6.2 Related Work

This section presents the most recent related studies in the literature that discuss the deployment of robotic applications, such as path planning, localization and obstacle avoidance. Depending on whether local or remote resources are utilized, these studies are classified into two groups; (i) the onboard sensor-based approaches and (ii) the offloading-based ones.

Due to limited computing resources, the onboard sensor-based approaches aim to provide lightweight solutions for robotic applications. Bajcsy et al. [151] proposed a safe navigation framework for autonomous vehicles moving in a priory unknown static environments under the assumption that the sensors work perfectly within their ranges. This framework was based on Hamilton Jacobi reachability analysis. Due to the computationally expensive nature of this analysis, the authors proposed an algorithm that uses only new measurements to update the safety set and can be executed in an online fashion. On the other hand, the authors in [152] focused on the path planning of autonomous vehicles that are able to maneuver on the road. The path planning problem was formulated as a nonlinear optimization problem and two Model Predictive Control solutions were designed for the lane selection and collision avoidance problems respectively. Miller et al. [153] proposed a controller synthesis algorithm for path planning in dynamic and partially unknown environments. The proposed vehicle system consisted of three subsystems: (i) a perception subsystem that provided a free space prediction around the vehicle, (ii) a plant-controller subsystem that guided the vehicle to specific waypoints and (iii) the planner subsystem that produced safe reference trajectories using Mixed Integer Linear Programming. Finally, the authors in [154] proposed a multi-robot collaborative localization framework, where followers assisted the self-localization of the leaders in a time-varying measurement topology. A binomial regulation function was used to describe the loss or resurgence of an observation. Then, a centralized extended Kalman filter was implemented for estimation purposes.

The offloading-based studies leverage the network and computing capabilities of edge servers to execute remotely navigation or localization algorithms. The authors in [155], similar to this work, presented a two-layer architecture for the realization of a visual-based Simultaneous Localization And Mapping application (SLAM) for tracking. They proposed a lightweight version of the computationally-intensive visual SLAM algorithm that is more suitable for resource-constrained mobile devices. On the other hand, a more precise variation of the visual SLAM is deployed on an edge server in proximity. The mobile devices transfer keyframes of a video for further processing at the edge side, only when necessary, i.e., when the feature points between two consecutive images are not similar. The offloading decision also considers the network conditions.

Chinchali et al. [156] proposed network offloading for Cloud Robotics. The offloading problem was formulated as a Markov Decision Process (MDP), where an autonomous system updated the offloading decision at every time interval. Then, Deep Reinforcement Learning (DRL) was used for solving the offloading problem taking into account the diverse network conditions and the trade-off between local and remote computation. On the other hand, the authors in [157] focused on data representation for task-centric communication rather than addressing the decision making problem of when to offload. Based on DRL, a robot encoder compressed and transmitted concise representations instead of raw data, while a server decoder generated a reconstructed estimation of the raw sensory input. Then, one of the pre-trained task modules was used to predict object locations and classes. Spatharakis et al. [146] proposed a switching offloading mechanism for robot path planning and localization. In that work, both services could be executed either locally, on the mobile robot, or remotely on an edge server. The offloading problem was formulated as a switching model that balanced the trade-off between navigation accuracy and mission duration. The offloading decision took into account both the robot pose uncertainties and the resource availability at the server side. In a different manner, the authors in [158] proposed a symbiotic robotic network for task offloading in the factory floor. Based on their vicinity, the robots formed clusters where members could offload tasks to each other. Additionally, a reward-based feedback task offloading mechanism was proposed to support delay-sensitive applications. Based on these rewards, each node had a social repute score which was used to select the appropriate node to offload the tasks and for the election of the cluster head.

6.3 Contributions and Outline

In a smart factory environment, the robots are equipped with sensors that can provide estimations of the robot's pose (i.e., two-dimensional location and orientation) [159]. Although employing on-board sensor-based localization yields quick results, they are known to be prone to accumulative errors, especially when it comes to long trajectories [160]. Thus, more sophisticated yet computationally intensive techniques are usually required to increase the localization accuracy. However, as discussed in [147], mobile robotic agents have limited computing capabilities. To overcome these limitations, an Edge Computing infrastructure in the robots' proximity can be leveraged to undertake the computationally intensive localization tasks. The communication between the robots and the servers can be performed through wireless access points located also on the operating floor, while 5G connectivity can be applied as well if available.

Aiming to analyze the above trade-offs and propose novel control co-design strategies that guarantee the correct behavior of such a system, this chapter presents a control co-design methodology for mobile robot navigation. Specifically, we consider a unicycle operating on a smart factory floor in an Industry 4.0 application, able to move independently without following predefined trajectories. The robot, equipped with cameras and odometry sensors, navigates from a starting to a target position, to complete a given mission (e.g., an automated storage/retrieval). In this context, the trade-off between the accuracy of the navigation and the mission duration is investigated, according to the mission's characteristics.

The main contributions of our work that differentiate it from the rest of the literature, are summarized as follows:

1. A 3C co-design for CPS is introduced where a unicycle-type mobile robot utilize both on-board and remote resources to execute the computationally intensive tasks of an Industry 4.0

application that requires navigation in a factory floor. Two fundamental problems are jointly tackled: (i) the synthesis of controllers that dictate the motion of the unicycle robot in this path planning problem and (ii) an offloading strategy to compensate the uncertainty of the local estimation techniques with the more accurate remote ones, while finding the balance between navigation accuracy and mission duration. In the context of event-triggered control and following well-acknowledged works, e.g., [161] and [162], we introduce a framework that incorporates the network and computation availability together with stability-preserving conditions. To the best of our knowledge, this is the first work that introduces such a solution.

2. Novel controllers are designed to satisfy the mission's hard constraint, i.e., to ensure the convergence of the mobile robot's navigation to a target set. A unicycle kinematic model is assumed for the robot's dynamics, while its movement is broken down into two parts, i.e., rotational and translational. We note that, although there are works in the literature providing elegant robust controllers in the presence of uncertainties, e.g., [150],[163], our proposed method allows us to deal with three distinct challenges, namely, (i) uncertainties/disturbances acting on the dynamics of the third state, (ii) efficiently applying set-based estimation using odometry measurements, (iii) guaranteeing convergence of the closed-loop system to a target waypoint. To further alleviate the computational strain from the resource-constrained platform, approximate computations are employed to locally estimate the robot's pose. Subsequently, stabilizing state feedback control mechanisms are applied to both movements, to guarantee convergence to the target region.

3. A utility function-based decision making process is properly formulated to undertake the computational offloading strategy. This strategy dictates which of the two different available localization techniques are used; an error-prone, for example, odometry-based localization, executed locally on the robot and the accurate vision-based localization method that requires a significant amount of computing resources and is executed on the edge server. Apart from the navigation's quality, which is acquired by the controllers' outputs, this process also takes into consideration the networking and edge computing resource availability to regulate the trade-off between navigation accuracy and mission duration, based on the performance requirements of the deployed application.

4. A series of experiments are performed to evaluate the performance of the proposed CPS in terms of navigation precision and mission duration. The evaluation indicates that the desirable solution concerning the preference of the two localization algorithms is a mixed strategy employing both of them. Using exclusively a locally produced estimation, is not sufficient to provide high enough accuracy, while constantly seeking for a more precise estimation from the remotely executed algorithm adds significant overhead in mission duration. Furthermore, a detailed comparative evaluation with alternative offloading schemes demonstrates our framework's benefits, as well as its adaptability to the application's specific requirements.

6.4 Problem Definition

Formally introducing the overall problem, let $x \in \mathbb{R}^3$ be the pose (states) and $u \in \mathbb{R}^2$ the control actions. Let us denote by $\mathcal{X}_t \subset \mathbb{R}^3$ the estimation set for $x(t)$ for each time instant t. Specifically, $\mathcal{X}_t$ contains all the possible states the robot can reach starting from an initial state $x(0) = x_0 \in \mathbb{R}^3$, with an initial uncertainty $\mathcal{X}(0) = \mathcal{X}_0 \in \mathbb{R}^3$. The problem of estimating a conservative approximation of $\mathcal{X}_t$ is a rather challenging computational problem for general dynamics and constrained environments [164, Chapter 10]. In our setting, we assume that two localization algorithms are available; (i) a fast, locally-executed one, providing an unreliable estimation while moving and (ii) a time-consuming, however more precise one, executed remotely on an edge server. Furthermore, we consider we have the option of invoking either of the two algorithms above; in fact, the challenge addressed in this chapter is to provide a stabilizing tracking controller and the corresponding offloading decision mechanism. We let $\kappa(t)$ be a switching signal that denotes which estimation algorithm is used at each time instant. Then, the twofold problem lies on the specification of:

a. the robot's respective control actions $u(t)$, under the current estimation set, such that the robot reaches and remains after finite time ϵ-close to the target position $x^\star$, i.e., there exists a t_f such that $||x(t) - x^\star|| \leq \epsilon$, for all $t \geq t_f$ and

b. the offloading control action $O(t) = h(q_o)$, where q_o denotes a utility function that incorporates the current navigation quality, network conditions and the edge computing resources availability.

Under this setting, a CPS is formulated by the mobile robot and the edge infrastructure, which collaborate to satisfy the requirements of a navigation application.

In the context of this chapter, the target position is dictated by an external path planning algorithm, for example as in [146], under a smart industry application. Moreover, this consideration allows us to ignore the case of obstacles located in the factory floor, as the path planning algorithm provides a safe trajectory. Table 6.1 summarizes the key notation used throughout the chapter.

6.5 Dynamics and Approximation Analysis

In this section, we introduce our proposed method to estimate the robot's pose under specific robot dynamics and uncertainties, a problem which is considered to be challenging throughout the literature [164].

6.5.1 System Dynamics

As mentioned before, for the robot's movement, we consider the unicycle kinematic model, also equivalent via an affine transformation to the differential drive dynamics. This model assumes

Table 6.1: Summary of the key notation.

Symbol	**Interpretation**
$x_1(t), x_2(t)$	Robot's position at time t
$x_3(t)$	Robot's orientation at time t
$u_1(t), u_2(t)$	Control actions at time t
$x^\star$	Target position
T	Time period - sampling time
$\sigma(t)$	Model switching signal
$\kappa(t)$	Offloading switching signal
$\delta(t)$	Heading error - disturbance at time t
$\mathcal{D}$	Heading error domain
$y_1^\star(t), y_2^\star(t)$	Measurements at time t
$w_1(t), w_2(t)$	Robot sensors' errors at time t
$\mathcal{W}$	Measurement errors domain
$\mathcal{X}_o$	Initial Estimation Set
$\mathcal{X}_t$	Estimation set at time t
$\mathcal{X}_t^{1:2}$	Estimation set of position at time t
$\mathcal{X}_t^3$	Estimation set of orientation at time t
$\hat{\mathcal{X}}_t$	Overapproximated Estimation set at time t
$Vol(\hat{\mathcal{X}}_t)$	Volume of $\hat{\mathcal{X}}_t$ set at time t
$\mathcal{Z}_t$	One-step Reachable set at time t
$\hat{\mathcal{Z}}_t$	Overapproximated One-step Reachable set at time t
$\mathcal{C}_t$	Compatibility set at time t
$d(\cdot,\cdot)$	Euclidean distance between two points
$V(\cdot,\cdot)$	Distance from a set to a point
$\varPhi(\cdot,\cdot)$	Orientation-target orientation incline
M	Set of states where translation motion is allowed
ϵ	Acceptable distance from target position
$\mathcal{G}$	Goal set
$x^{repr}(t)$	Representative point of $\hat{\mathcal{X}}_t$ at time t
$e_{ul}(d)$	Transmission time of offloaded task at time t
$e_{comp}(t)$	Computation time of offloaded task at time t
$q_o(t)$	Utility function value at time t
$O(t)$	Offloading strategy at time t
c_1, c_2	Coefficients of the Utility function

that the robot has two wheels that can rotate at different rates, allowing motion by changing the orientation and the position either separately or simultaneously. More specifically, the unicycle kinematics are given as follows, with the respective state space representation

$$\begin{cases} \dot{x}_1(t) &= u_1 \cos x_3(t), \\ \dot{x}_2(t) &= u_1 \sin x_3(t), \\ \dot{x}_3(t) &= u_2, \end{cases} \tag{6.1}$$

where x_1, x_2 refer to the position and x_3 to the orientation of the robot. We consider the control actions $(u_1(t), u_2(t))$ to be piecewise constant since the implementation of the control action is digital and the sampling time T is considered also constant. Moreover, as noted before, in this chapter, a path planning algorithm has already provided the target position the robot has to reach to complete its mission.

One celebrated approach in the bibliography [165, p. 96] is to manipulate the unicycle model by breaking down and discretizing the motion to two parts, i.e., rotate/adjust the orientation of the robot ("rotational motion") and move forward towards the target ("translational motion").

Hence, two subsystems are defined. The signal $\sigma(t) : \mathbb{R} \to \{0, 1\}$, switches between a translation (S_1, $\sigma(t) = 0$) and a rotation (S_2, $\sigma(t) = 1$), respectively. The systems with the respective state space representation are defined as:

$$S_1 : \begin{cases} \dot{x}_1(t) &= u_1(t) \cos(x_3(t)), \\ \dot{x}_2(t) &= u_1(t) \sin(x_3(t)), \\ \dot{x}_3(t) &= \delta(t), \end{cases} \tag{6.2}$$

$$S_2 : \begin{cases} \dot{x}_1(t) &= 0, \\ \dot{x}_2(t) &= 0, \\ \dot{x}_3(t) &= u_2(t) + \delta(t). \end{cases} \tag{6.3}$$

We assume that the effect the rotational motion has on the robot's position is negligible, while a bounded, however unknown, heading non-zero error $\delta(t) \in \mathcal{D} = [\delta_{min}, \delta_{max}]$ is applied to both motions. We should note here that, for the simplified case of zero disturbance, we utilize directly the dynamics derived by integrating both parts of Systems (6.2), (6.3), as in this case the x_3 state is now a constant. The modeling of the controllers and the proof of convergence remain the same. At each time step, the robot must decide whether to rotate or move forward to a straight line, by choosing the corresponding model to reach to a target position $x^\star$. Further analysis on the respective controllers is presented in Section 6.6. Regardless, the estimation set $\mathcal{X}_t$ for the three states grows from the initial estimation $\mathcal{X}_0$, as the robot moves. As in [164, Chapter 10], we propagate this set forward, given the control action $u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$ with $u_i(t) \in \mathcal{U}_i \subset \mathbb{R}$, $i = 1, 2$. Then, for the selected piecewise-constant control action and for an unknown but constant disturbance $\delta \in \mathcal{D}$, we solve the two subsystems analytically over one time period T

$$S_1 : \begin{cases} x_1(t+1) = x_1(t) + \frac{u_1(t)}{\delta(t)}[\sin(x_3(t+1)) - \sin(x_3(t))] \\ x_2(t+1) = x_2(t) + \frac{u_1(t)}{\delta(t)}[\cos(x_3(t)) - \cos(x_3(t+1))] \\ x_3(t+1) = x_3(t) + T\delta(t), \end{cases} \tag{6.4}$$

$$S_2 : \begin{cases} x_1(t+1) &= x_1(t), \\ x_2(t+1) &= x_2(t), \\ x_3(t+1) &= x_3(t) + T(u_2 + \delta(t)), \end{cases} \tag{6.5}$$

where T is the constant sampling time. We define the following mapping functions g_1, g_2 which consist of the dynamics of eq. (6.4),(6.5) respectively

$$g_1(x,u,\delta) = \begin{bmatrix} x_1 + \frac{u_1}{\delta}[\sin(x_3 + T\delta) - \sin(x_3)] \\ x_2 + \frac{u_1}{\delta}[\cos(x_3) - \cos(x_3 + T\delta)] \end{bmatrix},$$
$$g_2(x,u,\delta) = x_3 + T(u_2 + \delta)$$

The one-step reachable set $\mathcal{Z}_{t+1}$ is defined next; that is, the set of states that the robot can reach from the estimation set $\mathcal{X}_t$. To reduce complexity, the calculation of the one-step reachable set $\mathcal{Z}_{t+1}$, is decoupled by first computing the one-step reachable set for each state and then calculating the Cartesian product of these sets $\mathcal{Z}_{t+1} = Z^{1:2}_{t+1} \times Z^3_{t+1}$, where:

$$\mathcal{Z}^3_{t+1} = \left\{ z \in \mathbb{R} : \left(\exists x_3 \in \mathcal{X}^3_t, \exists \delta \in \mathcal{D} : z_3 = \begin{cases} x_3 + T\delta, & \text{if } \sigma(t) = 0 \\ g_2(x_3, u, \delta), & \text{if } \sigma(t) = 1. \end{cases} \right) \right\}. \tag{6.6}$$

and $\mathcal{Z}^{1:2}_{t+1}$ is computed as the Cartesian product of the propagated states $z_1(t), z_2(t)$, as follows

$$\mathcal{Z}^{1:2}_{t+1} = \left\{ z \in \mathbb{R}^2 : \left(\exists \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathcal{X}^{1:2}_t, \exists x_3 \in \mathcal{X}^3_t, \exists \delta \in \mathcal{D} : z = \begin{cases} g_1(x_1, u(t), \delta) & \text{, if } \sigma(t) = 0 \\ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} & \text{, if } \sigma(t) = 1 \end{cases} \right) \right\}. \tag{6.7}$$

Moreover, as the robot moves, the onboard sensors provide a local pose estimation, $y^\star(t+1)$. Typically, these measurements come from odometry calculations and concern distance travelled and changes in orientation [166, Chapter 5]. Hence, in the context of this chapter, we consider that the local estimation is of the following form:

$$\begin{bmatrix} y^\star_1(t+1) \\ y^\star_2(t+1) \end{bmatrix} = \begin{bmatrix} \|x_{1:2}(t+1) - x_{1:2}(t)\|_2^2 + w_1(t) \\ x_3(t+1) - x_3(t) + w_2(t) \end{bmatrix}, \tag{6.8}$$

where $w = \begin{bmatrix} w_1(t) & w_2(t) \end{bmatrix}^\top$ are the sensors' errors for the distance and the shift in orientation measurements respectively, that are unknown, however, bounded, i.e., $w \in \mathcal{W} \subset \mathbb{R}^2$. Then, the compatibility set, $\mathcal{C}_{t+1}$, which includes all the states compatible with the current measurements, is introduced as,

$$\mathcal{C}_{t+1} = \left\{ v \in \mathbb{R}^3 : \left(\exists w \in \mathcal{W}, \exists x \in \mathcal{X}_t \right) : \begin{bmatrix} y_1^\star(t+1) - \|v_{1:2} - x_{1:2}\|_2^2 \\ y_2^\star(t+1) - v_3 + x_3 \end{bmatrix} \in \mathcal{W} \right\}. \tag{6.9}$$

Finally, to complete the calculation of the estimation set, we compute the intersection set of $\mathcal{C}_{t+1}$ with $\mathcal{Z}_{t+1}$, $\mathcal{X}_{t+1}$, which comprises of the output compatible states

$$\mathcal{X}_{t+1} = \mathcal{C}_{t+1} \cap \mathcal{Z}_{t+1}. \tag{6.10}$$

6.5.2 Approximation Analysis

The exact calculations for the one-step reachable set and the compatibility set are challenging and usually computationally intensive [164, Chapter 10]. To alleviate the computational strain from the resource-constrained robot, we employ approximate computations, thus managing to provide fast computations of the estimation set. Specifically, we employ parallelotope overapproximations for the $\mathcal{Z}_{t+1}$ set, computed by eq. (6.6),(6.7) and for the $\mathcal{C}_{t+1}$ set computed by eq. (6.9).

6.5.2.1 Approximation of the one-step reachable set

For the problem of computing the one-step reachable set for each state (eq. (6.6),(6.7)), a Taylor Model (TM) approximation is invoked, as in [167]. TMs are used to represent flowpipes, i.e., a set of states reachable by continuous dynamics from an initial set within a given time interval. As a result, they can be used to provide guaranteed enclosures to the solutions of ordinary differential equations, often involving non-linear functions, such as the model into consideration. The interested reader may refer to [168] for further analysis of a TM flowpipe construction for non-linear hybrid systems. In the scope of this chapter, given the non-linear continuous system defined in eq. (6.2),(6.3) and the current estimation set $\mathcal{X}_t$, acting as the initial set at each time, we compute the TM flowpipes for each state variable, such that a polynomial overapproximation is computed, given the respective control action $u(t)$. In this way, each overapproximated state variable lies on an interval. Next, to acquire a guaranteed overapproximation of eq. (6.4), the Cartesian product of the three intervals of the states is computed. As a result, $\mathcal{Z}_{t+1}$ is overapproximated by a parallelotope in the 3-D space, hereinafter, denoted as $\hat{\mathcal{Z}}_{t+1}$ and computed by

$$\hat{\mathcal{Z}}_{t+1} = \hat{\mathcal{Z}}_{t+1}^1 \times \hat{\mathcal{Z}}_{t+1}^2 \times \hat{\mathcal{Z}}_{t+1}^3 = [z_1^{\min}(t+1), z_1^{\max}(t+1)] \times [z_2^{\min}(t+1), z_2^{\max}(t+1)] \times [z_3^{\min}(t+1), z_3^{\max}(t+1)]. \tag{6.11}$$

The reachable sets for the remaining states are computed similarly. We should note here that different approximation methods for the calculation of the one-step reachable set can be chosen, using for example a Bernstein polynomial basis [169],[170]. Moreover, the exact calculation of the one-step reachable set is known [171] or can be approximated [172], however for recursive calculations, the computation time rises. For our work particularly, we selected the TM approximation due to its popularity and the fact that, as the initial set becomes smaller or the order of the polynomial used for approximation becomes larger [168, Chapter 3.3], the overapproximation becomes more accurate. On the other hand, shorter sampling time results in more accurate approximations, increasing, however, the complexity. The proposed technique is computationally light and complexity-preserving. Hence, it is suitable for calculating the reachable set repeatedly and providing real-time navigation to the resource-constrained mobile robot. The calculation of the estimation set (eq. (6.10)), i.e., the intersection of the reachable set with the compatibility set, is in principle a non-convex problem. As a result, using a fixed-complexity polytope as a template for the reachable set, alleviates the complexity and reduces computation time, as presented in the next subsection.

6.5.2.2 Approximation of the estimation set

As stated in [164, Chapter 10], even more challenging is the calculation of the output compatible states. These states are defined by eq. (6.9). Hence, an interval for where each state lies can be found. Again, a parallelotope approximation $\hat{\mathcal{X}}$ is used to provide an overapproximation of the $\mathcal{X}$ set. Specifically, we aim to further reduce the overapproximation acquired with $\hat{\mathcal{Z}}_{t+1}$, by investigating the compatibility with the acquired measurements $y^\star(t+1)$. Thus, we formulate two optimization problems for each state variable, to find the respective maximum and minimum values of this interval. As a result, the calculation of the interval approximation of the output compatible set can be achieved by solving generally nonlinear optimization problems. Specifically, to find the maximum attainable value for x_1, given a measurement $y^* \in \mathbb{R}^2$, we solve the following:

$$\begin{aligned} \underset{v_1,v_2,w_1,x_1,x_2}{\text{maximize}} \quad & v_1 && (6.12a)\\ \text{subject to} \quad & y_1^* - \|v_{1:2} - x_{1:2}\|_2^2 \in \mathcal{W}^1, && (6.12b)\\ & (x_1, x_2) \in \hat{\mathcal{X}}_t^{1:2}, && (6.12c)\\ & (v_1, v_2) \in \hat{\mathcal{Z}}_{t+1}^{1:2}. && (6.12d) \end{aligned}$$

From the solution of the above problem, the maximum output-compatible value $u_1^{\max}$ for the x_1 state can be obtained. Similarly, to find the minimum output-compatible value for each state, an equivalent optimization problem with different cost functions and the same constraint set, is solved. All such problems are quadratically constrained linear programs. It should be noted that, the output compatible values for x_1, x_2 states must be positive, as we consider that the operating ground corresponds to the first quadrant of a Cartesian plane. Since $\mathcal{W}_1 = [w_1^{\min}, w_1^{\max}]$, with $w_1^{\min} \in \mathbb{R}$, $w_1^{\max} > 0$ being an interval, relation (6.12b) is equivalent to:

$$\|v_{1:2} - x_{1:2}\|_2^2 \leq y_1^* - w_1^{\min}, \quad (6.13)$$

$$\|v_{1:2} - x_{1:2}\|_2^2 \geq y_1^* - w_1^{\max}. \quad (6.14)$$

Ineq. (6.13) contributes with a convex constraint and (6.14) with a concave constraint respectively, thus, the total constraint set of the optimization problem is non-convex. As a result, the problem cannot be easily solved with efficient convex interior-point methods. Nevertheless, an optimal solution can be found by exploiting the other interval constraints.

First, the solutions for the maximization problem without constraint (6.14) and the minimization problem without constraint (6.13), are presented. Subsequently, using Proposition 1, we showcase that the constraints for each problem are redundant. For example, regarding Problem (6.12a), let us focus on constraint (6.13): The general solution of the inequality is: $-\sqrt{y_1 - w_1^{\min} - (v_2 - x_2)^2} + x_1 \leq v_1 \leq \sqrt{y_1 - w_1^{\min} - (v_2 - x_2)^2} + x_1$, therefore the maximum positive output-compatible value $v_1^{\max}$ is attained in

$$v_1^{\max} = \min\left\{v_1^{\max}, x_1^{\max} + \sqrt{y_1^\star - w_1^{\min} - \min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}\right\}, \tag{6.15}$$

where $\min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ can be easily calculated, as both v_2, x_2 have known values in intervals. Analogously, for the minimization of v_1, (6.14) has the following general solution: $v_1 \in (-\infty, -\sqrt{y_1 - w_1^{\max} - (v_2 - x_2)^2} + x_1] \cup [\sqrt{y_1 - w_1^{\max} - (v_2 - x_2)^2} + x_1, \infty)$ and therefore the minimum positive output-compatible value is given by:

$$u_1^{\min} = \max\left\{v_1^{\min}, x_1^{\min} + \sqrt{y_1^\star - w_1^{\max} - \max_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}\right\}, \tag{6.16}$$

where, similarly, $\max\limits_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ can be easily calculated. We should note here that, in the case where the quantities under the square root are negative for both problems, then they become infeasible and, thus, the maximum and minimum value of v_1 becomes equal to the corresponding value obtained by the one-step reachable set.

Proposition 1. *Let $v_1^{\max}$ be a feasible solution of the maximization problem* (6.12a) *without constraint* (6.14) *contributing to the problem and, similarly, let $v_1^{\min}$ be a feasible solution for the respective minimization problem without* (6.13) *respectively. If $v_1^{\min} \leq v_1^{\max}$, then the two constraints are redundant for solving the respective optimization problems.*

Proof. Let $v_1^{\max}, v_1^{\min} \in \hat{\mathcal{Z}}_{t+1}^{1:2}$ be the two feasible solutions of the maximization problem (6.12a) without constraint (6.14) and the corresponding minimization problem without constraint (6.13), respectively. Then, we want to show that $v_1^{\min} \leq v_1^{\max}$. By construction of the TM overapproximation, we know that $x_1^{\min} \leq x_1^{\max}$. As a result, proving $v_1^{\min} \leq v_1^{\max}$ is equivalent to showing that $x_1^{\min} + \sqrt{y_1^\star - w_1^{\max} - \max\limits_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2} \leq x_1^{\max} + \sqrt{y_1^\star - w_1^{\min} - \min\limits_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}$. This is also equivalent of proving that $y_1^\star - w_1^{\max} - \max\limits_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2 \leq y_1^\star - w_1^{\min} - \min\limits_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ which is true as both $v_2, x_2, w_1^{\max}, y_1$ are positive. As a result, the two solutions do not intersect and the respective absent constraints do not contribute to the feasible solutions. □

Hence, the solution of a non-convex optimization problem reduces to a simple calculation for each

Table 6.2: A numerical example of the proposed technique.

i	$\hat{\mathcal{X}}_t^i$	$\hat{\mathcal{Z}}_{t+1}^i$	$\hat{\mathcal{X}}_{t+1}^i$	$u_1(t)$	$y_1^\star$	$y_2^\star$
1	[5.42, 5.64]	[5.62, 5.84]	[5.62, 5.83]	0.38	0.15	0.1
2	[5.67, 6.00]	[5.98, 6.32]	[5.98, 6.20]			
3	[0.98, 1.02]	[0.97, 1.03]	[0.97, 1.02]			

state variable. The minimum and maximum output-compatible values of the x_2 state are attained similarly.

On the other hand, for the output-compatible values of the x_3 state, it suffices to solve two similar linear programs to find the minimum and maximum admissible values. In order to find a maximum value $u_3^{\max}$, the following optimization problem must be solved, derived by eq. (6.9)

$$\begin{aligned} \underset{v_3, x_3, w_2}{\text{maximize}} \quad & v_3 && (6.17a)\\ \text{subject to} \quad & v_3 - x_3 \leq y_2^\star - w_2^{\min}, && (6.17b)\\ & v_3 - x_3 \geq y_2^\star - w_2^{\max}, && (6.17c)\\ & w_2 \in \mathcal{W}^2, && (6.17d)\\ & x_3 \in \hat{\mathcal{X}}_t^3, && (6.17e)\\ & v_3 \in \hat{\mathcal{Z}}_{t+1}^3, && (6.17f) \end{aligned}$$

where $\mathcal{W}_2 = [w_2^{\min}, w_2^{\max}]$, with $w_2^{\min} \in \mathbb{R}$, $w_2^{\max} > 0$. The minimum output-compatible value $u_3^{\max}$ is attained by the respective minimization problem subject to the same constraints. Finally, having calculated the output-compatible intervals for the three states, e.g., $\hat{\mathcal{X}}_{t+1}^1 \in [u_1^{\min}, u_1^{\max}]$, the overapproximated estimation set $\hat{\mathcal{X}}_{t+1}$ is computed as the Cartesian product of the three sets:

$$\hat{\mathcal{X}}_{t+1} = \hat{\mathcal{X}}_{t+1}^1 \times \hat{\mathcal{X}}_{t+1}^2 \times \hat{\mathcal{X}}_{t+1}^3 = [u_1^{\min}, u_1^{\max}] \times [u_2^{\min}, u_2^{\max}] \times [u_3^{\min}, u_3^{\max}]. \tag{6.18}$$

Thus, in the context of this chapter, the estimation set is a parallelotope in the 3-D space. As it will be thoroughly examined in the following section, the low complexity of the approximation of the estimation set leads to simple calculations for deriving controllers for each motion. Moreover, the volume of the estimation set in the 3-D space is introduced:

$$\mathrm{Vol}(\hat{\mathcal{X}}_{t+1}) = (u_1^{\max} - u_1^{\min})(u_2^{\max} - u_2^{\min})(u_3^{\max} - u_3^{\min}). \tag{6.19}$$

In Fig. 6.1, an example of our approximation is demonstrated, for the experiment setting introduced in Section 6.8. The blue-dashed line denotes $\hat{\mathcal{X}}_t$, while the red-dashed line the overapproximation using the TM technique. Finally, the next estimation set, $\hat{\mathcal{X}}_{t+1}$, is drawn with a green line. Additionally, for the specific example of translational motion, Table 6.2 presents the actual overapproximation of our method in numbers. For the calculation of these sets, the following values are used $w_1^{\min} = 0.38$, $w_1^{\max} = -0.22$, $w_2^{\min} = -0.008$, $w_2^{\max} = 0.002$ and $\delta = 0.1$. More details regarding the measurement errors and the disturbance follow in Section 6.8.

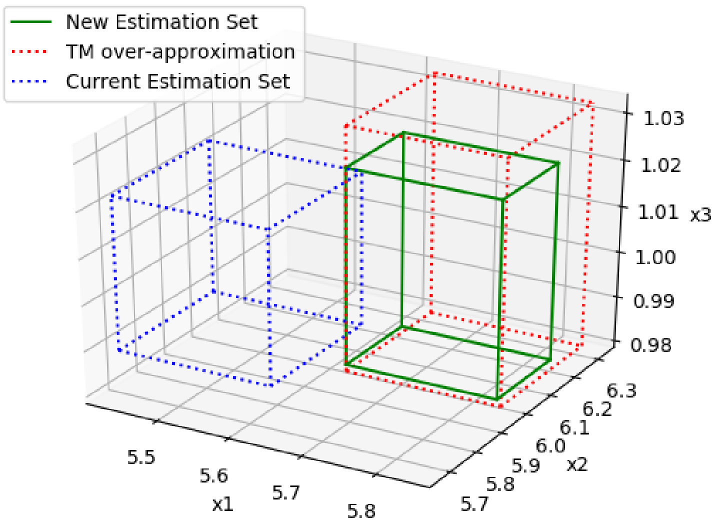


Figure 6.1: An example of the overapproximation for the calculation of the next estimation set, $\hat{\mathcal{X}}_{t+1}$.

We can notice that the conservative approximation of the estimation set, while taking into account the measurements, affects significantly the result, especially in this nonlinear setting. To sum up, the challenging calculation of the estimation set is handled with overapproximation techniques due to the TM properties and our approach of computing the output-compatible states.

6.6 Control Design

In the previous section, the solution for calculating a conservative approximation of the estimation set was agnostic to the control actions. In this section, we propose a stabilizing state feedback control mechanism, that guarantees convergence to the target region. Let $d(x(t), x^\star) = \|x^\star_{1:2} - x_{1:2}(t)\|_2$ be the distance between two points, specifically between a specific point and the target. The distance between a point and a set is defined as:

$$V(\hat{\mathcal{X}}_t, x^\star) = \max_{x \in \hat{\mathcal{X}}_t} d(x, x^\star). \tag{6.20}$$

Moreover, let

$$\Phi(\hat{\mathcal{X}}_t, x^\star) = \Big\{ \varphi \in \mathbb{R} : (\exists x \in \hat{\mathcal{X}}_t : \varphi = x_3 - \tan^{-1}\left(\frac{x_2 - x_2^\star}{x_1 - x_1^\star}\right)) \Big\}, \tag{6.21}$$

be the set of angles between the robot's current estimation of orientation and the line connecting current robot's position and the target position for $\hat{\mathcal{X}}_t$, simply put, the set of angles of incline towards the target. Finally, let $\mathcal{M}$ be the set of states where translation is performed,

$$\mathcal{M} = \Big\{x \in \mathbb{R}^3 : \Big(\exists u_1 \in \mathbb{R} : l|u_1| < 2V(\hat{\mathcal{X}}_t, x^\star)\cos(\delta_M)\Big)\Big\}, \tag{6.22}$$

where $\delta_m = \max\{|\delta_{min}|, \delta_{max}\}$,

$$l = \sqrt{2(\delta_m{}^{-2}(1 - \cos(T\delta_m))},$$
$$\delta_M = \max\Big\{\Big|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\Big|, \Big|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\Big|\Big\}$$

The reasoning behind the choice of the $\mathcal{M}$ set is presented in Subsection 6.6.4. In order to select between performing either translational or rotational motion and reduce the overall computational complexity, only the center of the convex hull of the estimation set is investigated in terms of whether it belongs in the M set. Hereinafter, the center point of the current estimation set, $\hat{\mathcal{X}}_t$, is referred to as the *representative point*, $x^{repr}(t)$. As the estimation set is the Cartesian product of three intervals of the states, the coordinates of the representative point are computed straightforwardly:

$$x^{repr}(t) = (x_1^{repr}(t), x_2^{repr}(t), x_3^{repr}(t)), \tag{6.23}$$

where $x_i^{repr}(t) = \frac{1}{2}(x_i^{\max}(t) + x_i^{\min}(t))$, $i \in [1, 2, 3]$, is the center of the interval of each state $x_i \in \hat{\mathcal{X}}_t^i$. The translational motion is selected if $x^{repr} \in \mathcal{M}$ for any u_1, otherwise the rotational motion is selected. Moreover, we consider that the robot has reached the target position when $\hat{\mathcal{X}}_t \subseteq \mathcal{G}$, with

$$\mathcal{G} = \{x \in \mathbb{R}^3 : V(\hat{\mathcal{X}}_t, x^\star) \leq \epsilon\}, \tag{6.24}$$

where ϵ is an acceptable distance from the target position, for the mission to be assumed successful. Note that the robot's orientation when the goal set is reached is not considered important. Since in the proposed modeling the motion is considered either strictly translational or rotational, one control input in each case is nonzero. The overall control strategy that includes the state-dependent switching mechanism can be described by a directed graph of Fig. 6.2, which illustrates the control automaton. It is noted that the closed-loop system can be described by a linear hybrid automaton with non-convex guard conditions. Nevertheless, we do not use this formalism in order to simplify exposition, especially since the convergence proofs that follow are simple enough not to necessitate the adoption of this powerful modeling approach. Independently of the selected estimation technique, the robot moves as described in the control automaton; specifically (i) decides the motion (signal $\sigma(t)$) according to (6.22) for $x^{repr}(t)$, (ii) computes and moves according to the selected control action, (iii) acquires an estimation of its pose using an estimation technique and finally (iv) follows the procedure introduced in the previous section to compute the new estimation set and repeats the process. When the goal set is reached the robot stops. A complete algorithm of the proposed technique is presented in detail in Subsection 6.7.4.

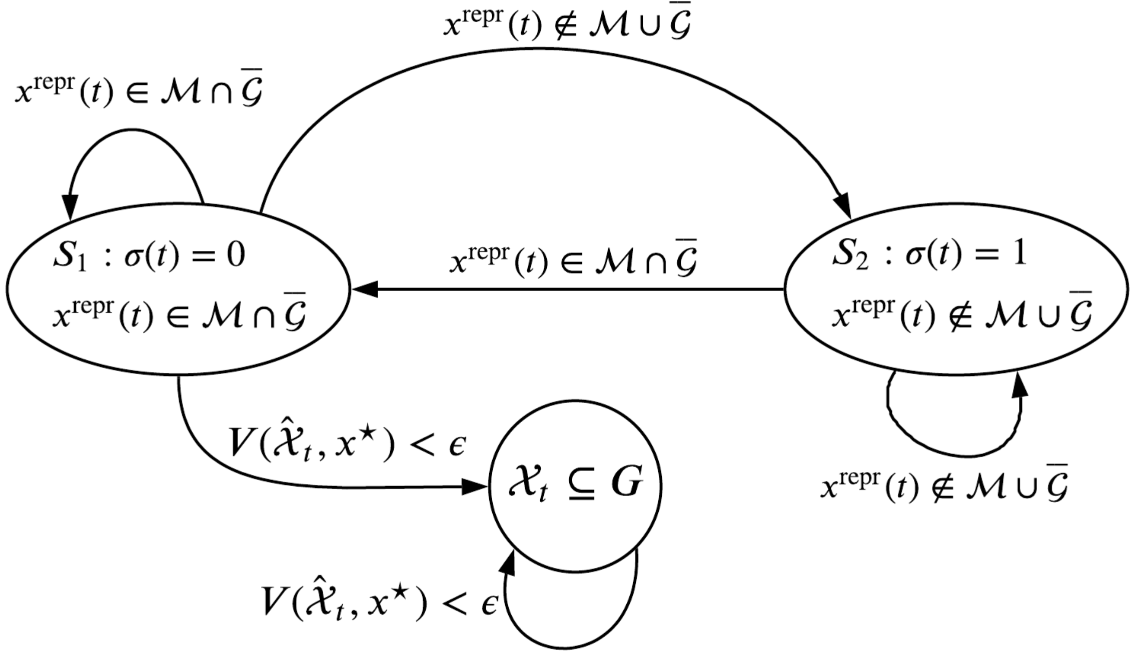

$x^{\text{repr}}(t) \notin \mathcal{M} \cup \overline{\mathcal{G}}$
$x^{\text{repr}}(t) \in \mathcal{M} \cap \overline{\mathcal{G}}$
$S_1 : \sigma(t) = 0$
$x^{\text{repr}}(t) \in \mathcal{M} \cap \overline{\mathcal{G}}$
$x^{\text{repr}}(t) \in \mathcal{M} \cap \overline{\mathcal{G}}$
$S_2 : \sigma(t) = 1$
$x^{\text{repr}}(t) \notin \mathcal{M} \cup \overline{\mathcal{G}}$
$V(\hat{\mathcal{X}}_t, x^\star) < \epsilon$
$\mathcal{X}_t \subseteq G$
$x^{\text{repr}}(t) \notin \mathcal{M} \cup \overline{\mathcal{G}}$
$V(\hat{\mathcal{X}}_t, x^\star) < \epsilon$

Figure 6.2: The control automaton for the robot's motion.

6.6.1 Set Controller for Translational Motion

In this subsection, we compute a translational control action that decreases the distance to the target set, i.e.: $V(\hat{\mathcal{X}}_{t+1}, x^\star) < V(\hat{\mathcal{X}}_t, x^\star)$. The control action $u_1(t)$ is computed by the solution of the following optimization problem,

$$\underset{\lambda,\lambda_2}{\text{minimize}}\ \lambda \tag{6.25a}$$

$$\text{subject to} \quad \lambda^2 \geq 1 + (\lambda_2^2 - 2\lambda_2)\cos^2(\delta_M) \tag{6.25b}$$

$$\lambda \in (0,1) \tag{6.25c}$$

$$\lambda_2 \in (0,2), \tag{6.25d}$$

where $\delta_M = \max\{\left|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|, \left|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|\}$. Then, for the selected λ_2 that minimize λ, we select the following control action

$$l|u_1| = \lambda_2 d(x_{1:2}^{repr}, x^\star)\cos(T\delta_M), \tag{6.26}$$

where $l = \sqrt{2(\delta_m{}^{-2}(1-\cos(\delta_m))}$. In this chapter we consider only forward motion for the unicycle robot; as a result, the positive value of u_1 is selected. Next, for the selected control action $u_1(t)$, the approximation of the one-step reachable set $\hat{\mathcal{Z}}_t$ is computed using eq. (6.11). Also, as stated in Subsection 6.5.2.1, the approximated one-step reachable set $\hat{\mathcal{Z}}_t$, has guaranteed enclosures for the solution of eq. (6.4) by construction. Let us consider $\hat{\mathcal{Z}}_t^{1:2}$, omitting the orientation plane as the distance function denoted in eq. (6.20), is defined in the 2-D space. It should be noted here that the following condition for the selected control action is investigating using the reachable set approximation and without taking into account the compatibility set, as this presupposes a control action that is already applied to the robot. To this purpose, the value of eq. (6.20) can be deduced from the vertices of $\hat{\mathcal{Z}}_t^{1:2}$, since it is a parallelotope overapproximation. Thus, it suffices to investigate whether the distance from the vertices of $\hat{\mathcal{Z}}_t^{1:2}$ towards the target decreases for a given control action. To this purpose, let us denote the vertices of $\hat{\mathcal{Z}}_{t+1}$ as:

$$p_{ij}(t+1) = [z_1^i, z_2^j], \tag{6.27}$$

where $i = 1, 2$, $j = 1, 2$ indicate the four vertices of $\hat{\mathcal{Z}}_t^{1:2}$. For example, $i = 1$ denotes $z_1^{\min}(t+1)$ and similarly $j = 2$ denotes $z_2^{\max}(t+1)$, where $z_1^{\min}(t+1)$, $z_1^{\max}(t+1) \in \hat{\mathcal{Z}}_{t+1}^1$ and $z_2^{\min}(t+1)$, $z_2^{\max}(t+1) \in \hat{\mathcal{Z}}_{t+1}^2$ as in eq. (6.11). Consequently,

$$V(\hat{\mathcal{X}}_{t+1}, x^\star) = \max_{i,j\in[1,2]} d(p_{ij}(t+1), x^\star). \tag{6.28}$$

As a result, given the selected control action $u_1(t)$, the following condition must be investigated to determine if the translational motion decreases the distance of the $\hat{\mathcal{Z}}_t^{1:2}$ set to the target position in one step

$$\max_{i,j\in[1,2]} d(p_{ij}(t+1), x^\star) \leq \lambda' V(\hat{\mathcal{X}}_t, x^\star), \tag{6.29}$$

where $\lambda' \in (0, 1)$. If this condition holds, then the robot implements the translational motion for $u_1(t)$, acquires the measurements and proceeds with the calculation of $\hat{\mathcal{X}}_{t+1}$, as described in Subsection 6.5.2.2. Otherwise, eq. (6.20) is not decreasing and a more precise estimation of the current location is required to approach the target position. Hence, each time the translational motion is allowed, it suffices to find the maximum $\lambda'_{max}(t) \in (0, 1)$ for all vertices to guarantee that the distance towards the target is decreasing.

6.6.2 Set Controller for Rotational Motion

When the representative point $x^{repr}(t+1)$ does not belong in $\mathcal{M}$, the robot performs a rotational motion. During this, the rotational controller acts to ensure that after one actuation step the translational motion will be allowed. This is essential as naturally the considered function of eq. (6.20) is not decreasing when the rotational motion is performed.

In this section, an interval of angles of incline $a \in [a_{\min}, a_{\max}]$ of the lines connecting each point in the estimation set to the target position, is specified. The purpose of this is to find, if exists, a rotational control action that shifts the robot's orientation, aiming to satisfy ineq. (6.22) for

$x^{repr}(t+1)$. Since the overapproximated estimation set $\hat{\mathcal{X}}_t$ is a parallelotope in the 3-D space, i.e., the states x_i, $i \in [1,2,3]$ lie in known intervals, then $a_{\min}$ and $a_{\max}$ can be specified by calculating the angles of incline for the lines that connect the vertices of the estimation set, in the 2-D space and the target position, as illustrated in Fig. 6.3. Consequently, the control action $u_2(t)$ is calculated by solving the following linear problem:

$$\begin{aligned}
\min_{u_2} \quad & \max_{\hat{\mathcal{X}}_t, a, \delta} (x_3 + \delta - u_2 - a) && \text{(6.30a)}\\
\text{subject to} \quad & x_3 \in \hat{\mathcal{X}}_t^3, && \text{(6.30b)}\\
& a \in [a_{\min}, a_{\max}], && \text{(6.30c)}\\
& \delta \in \mathcal{D}, && \text{(6.30d)}\\
& u_2 \in \mathcal{U}_2, && \text{(6.30e)}
\end{aligned}$$

where $a_{\min} = \min\{\tan^{-1}\left(\frac{x_{2,j}(t)-x_2^\star}{x_{1,i}(t)-x_1^\star}\right)\}$ and $a_{\max} = \max\{\tan^{-1}\left(\frac{x_{2,j}(t)-x_2^\star}{x_{1,i}(t)-x_1^\star}\right)\}$, where $i = 1,2$ and $j = 1,2$ indicate the four vertices of the estimation set in the 2-D space. For example, $i = 1$ indicates the minimum value of the interval of $x_1(t)$ and similarly $j = 2$ indicates the maximum value for $x_2(t)$. Next, for the selected control action $u_2(t)$, $\hat{\mathcal{Z}}_{t+1}$ is computed by, eq. (6.11) which also has guaranteed enclosures for the solution of eq. (6.6) by construction. Similarly to the previous subsection, the compatibility set in not taken into consideration. We define

$$z^{repr}(t+1) = (z_1^{repr}(t+1), z_2^{repr}(t+1), z_3^{repr}(t+1)), \tag{6.31}$$

where $z_i^{repr}(t+1) = \frac{1}{2}(z_i^{\max}(t+1) + z_i^{\min}(t+1))$, for $i \in [1,2,3]$, is the center of the interval of each state $z_i \in \hat{\mathcal{Z}}_{t+1}^i$. It should be noted that when a rotation is performed the robot's position (x_1, x_2) is not affected. As a result, the following condition is examined; whether the shift in orientation endeavors to satisfy ineq. (6.22) for the representative point $z^{repr}(t+1)$:

$$|u_1| < \frac{2}{l} d(z_{1:2}^{repr}(t+1), x^\star)\cos(\delta_M), \tag{6.32}$$

where $l = \sqrt{2(\delta_m{}^{-2}(1-\cos(T\delta_m)))}$ and $\delta_M = \max\{\left|T\delta_{min} + \Phi(z^{repr}(t+1), x^\star)\right|, \left|T\delta_{max} + \Phi(z^{repr}(t+1), x^\star)\right|\}$. If $z^{repr}(t+1)$ belongs in $\mathcal{M}$ then the robot performs the rotational motion for $u_2(t)$, acquires the measurements and proceeds with the calculation of $\hat{\mathcal{Z}}_{t+1}$, as described in Subsection 6.5.2.2.

To sum up, after the rotational motion, if $x^{repr}(t)$ belongs in $\mathcal{M}$ then the translational motion is allowed in the next step. Hence, the robot performs the translational motion for $u_1(t)$, acquires a measurement and proceeds with the calculation of $\hat{\mathcal{X}}_{t+1}$, as described in Subsection 6.5.2.2. Otherwise, a more precise estimation is necessary to proceed towards the target position. One must note that, after rotating with $u_2(t)$, if $z^{repr}(t+1)$ is in $\mathcal{M}$, then, due to the parallelotope overapproximation, it is certain that $x^{repr}(t+1)$ will be in $\mathcal{M}$, as described in Subsection 6.5.2.2. That is why translational motion is allowed in the next step.

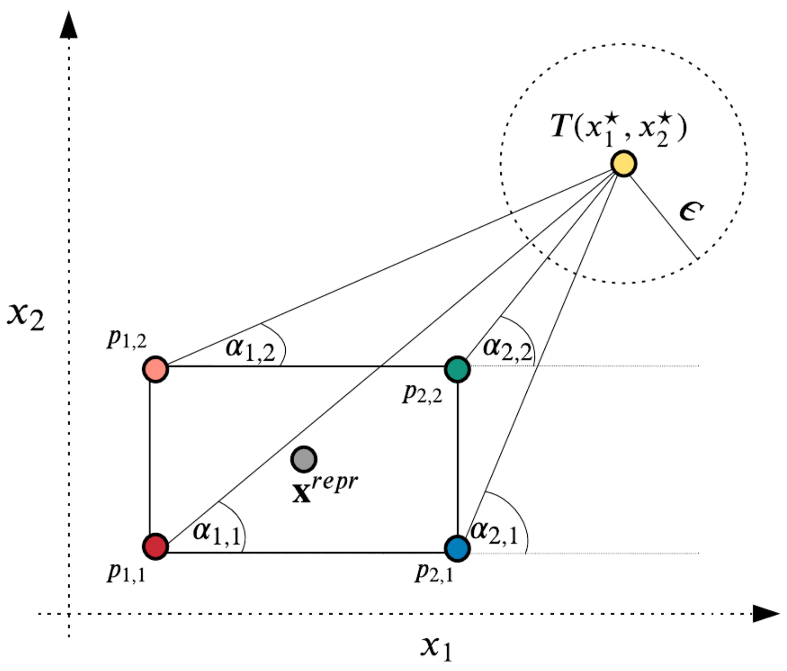


Figure 6.3: An illustrated example of the maximum minimum distance and slope between $\hat{\mathcal{X}}_t$ and the target position.

6.6.3 Remote Estimation Technique

It should be clear now that if conditions (6.29) and (6.32) are not satisfied for the translational and the rotational motion respectively, then the system can no longer converge to the target position and the switching signal $\kappa(t)$ triggers the precise estimation technique to assist in the navigation

$$\kappa(t) = \begin{cases} 1, & \text{if } \sigma(t) = 0 \text{ and (6.29) is not satisfied} \\ & \text{or} \\ & \text{if } \sigma(t) = 1 \text{ and (6.32) is not satisfied,} \\ 0, & \text{otherwise.} \end{cases} \tag{6.33}$$

Whenever $\kappa(t) = 1$, the localization algorithm described in [159] is invoked remotely. Briefly, this estimation technique relies on a bilateration method using principles of projective geometry. The robot's equipped camera captures images from the area and offloads them into the proximate edge server. There, the localization algorithm analyzes the images to detect landmarks and provide a highly precise estimation regarding the pose of the robot. Thus, this real-time image processing is resource-intensive and time-consuming, even when executed on an edge server. Hence, both the transmission overhead of the images via the access point and the remote processing overhead must be considered to find the right balance between navigation accuracy and mission duration.

In the context of this chapter, the estimation computed by this technique is considered accurate, without measurement errors. Consequently, after invoking the remote estimation technique, the exact pose of the robot is considered known. This allows us to compute a fine-grained control approach and prove the convergence of the proposed technique. We should note here that, the choice of the selected remote estimation technique is made to showcase a general setting in edge robotics in which computationally intensive algorithms are not to be executed on the robot, but rather to be offloaded on an edge server. Many works exist in the bibliography such as [173],[174] that provide very precise estimations regarding the robot's pose. Moreover, such landmark-based techniques are broadly utilized for indoor localization in the context of Industry 4.0. [175]. It should be emphasized, that the scope of this chapter is to introduce an offloading strategy between different estimation techniques, seeking a trade-off between mission duration and accuracy and not seeking the best between different methods.

6.6.4 Convergence when Constantly Invoking Remote Estimation

In this section, we show that the proposed technique converges to the target position after finite time, when the remote estimation technique is exclusively invoked. Let us assume that at time $t = t^\star$ the remote estimation technique provides a precise estimation to the robot, thus the estimation set is minimized to a point $\mathcal{X}_{t^\star} = \{x^A\}$, as illustrated in Fig. 6.4. Then, it is straightforward to compute the $\mathcal{Z}_{t+1}$ set using eq. (6.4) and (6.5).

The robot initially aligns the orientation accordingly and then proceeds with the translational motion. Another benefit of the precise estimation is that the set-based controller, proposed in Problem (30), considers only one point in order to provide the control action $u_2^\star$. In this way, the robot manages to rotate, aligning to (AT), as illustrated in Fig. 6.4. Thus, the angle of incline towards the target is minimized, specifically $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$. As a result, the orientation of the robot $\mathcal{Z}^3_{t^\star+1}$ lies in the interval $\left[\tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{min}, \tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{max}\right]$.

Lemma 1. *Consider the Subsystem* S_2 (6.5) *and that the remote estimation technique provides an accurate estimation* x^A *at time* $t = t^\star$. *Let* u_2^* *be the control input obtained by the solution of Problem* (6.30) *applied to* (6.5) *at time* t^*. *If* $\left|(T+1)\delta_m\right| < \frac{\pi}{2}$, *then the translational motion is allowed in the next step, i.e., there exists a control action* $u_1^\star$ *that satisfies ineq.* (6.22) *for* $\mathcal{Z}_{t^\star+1}$.

Proof. After performing a rotational motion, the robot is still positioned at $x^A_{1:2}$, i.e., $\mathcal{Z}^{1:2}_{t^\star+1} = x^A_{1:2}$, and $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$. Moreover, for the translational motion to be allowed, it suffices that $\mathcal{Z}_{t^\star+1} \subseteq \mathcal{M}$. Subsequently, given that $V(\mathcal{Z}_{t^\star+1}, x^\star) = d(x^A, x^\star)$, from ineq. (6.22) we get:

$$l\,|u_1| < 2d(x^A, x^\star)\cos(\delta_M), \tag{6.34}$$

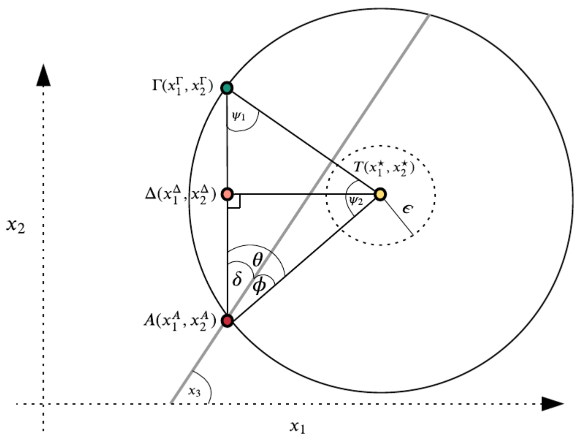

$\Gamma(x_1^\Gamma, x_2^\Gamma)$
ψ_1
$T(x_1^\star, x_2^\star)$
$\Delta(x_1^\Delta, x_2^\Delta)$
ψ_2
ϵ
x_2
θ
δ
ϕ
$A(x_1^A, x_2^A)$
x_3
x_1

Figure 6.4: Convergence example after the rotational motion.

where

$$l = \sqrt{2(\delta_m{}^{-2}(1-\cos(T\delta_m))}, \quad \delta_m = \max\{|\delta_{min}|\,, \delta_{max}\}$$
$$\delta_M = \max\Big\{\Big|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\Big|\,, \Big|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\Big|\Big\}$$

Under our assumption that $\big|(T+1)\delta_m\big| < \frac{\pi}{2}$ and since $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$, then l is a positive constant and $0 < \cos(\delta_M) < 1$. As a result, the translational motion is allowed at time $t = t^\star + 1$ for all $u_1^\star$ that satisfy ineq. (6.34) and therefore (6.22). $\square$

We note that typically the heading error lies in the fraction of a few degrees [176], thus, the condition of Lemma 1 is not particularly conservative since the sampling time T is small.

Proposition 2. *Suppose that the robot rotates at time $t^\star$. Then, there exists a control action $u_1^\star$ such that $V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star)$ for $t = t^\star + 1$, $\lambda \in (0,1)$.*

Proof. After rotational motion, by eq. (6.10) we know that $\mathcal{X}_{t^\star+1} \subseteq \mathcal{Z}_{t^\star+1}$, since the compatibility set is not included in this proof. By Lemma 1, at time $t = t^\star + 1$, the robot is positioned at $x_{1:2}^A$, hence the orientation $\mathcal{X}_t^3$ lies in the interval $\left[\tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{min}, \tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{max}\right]$. Moreover, it also holds that $\Phi(\mathcal{X}_t, x^\star) \subseteq \mathcal{D}$ and the translational motion is allowed for all $u_1^\star$ that satisfy ineq. (6.34). Also, from Subsystem (6.4), it occurs that the shift in the robot's orientation, produced by the heading error during the translational motion, is equal to $T\delta$. Then, Θ is the set of angles that combine $\Phi(\mathcal{X}_{t^\star+1}, x^\star)$ and this shifts in the orientation, as follows:

$$\Theta(\delta, \Phi(\mathcal{X}_t, x^\star)) = \Big\{\vartheta \in \mathbb{R} : \Big(\exists \delta \in \mathcal{D}, \exists \varphi \in \Phi(\mathcal{X}_{t^\star+1}, x^\star) : \vartheta = T\delta + \varphi\Big)\Big\}. \tag{6.35}$$

It should be noted that $\theta \in \Theta \subseteq [d_{min} + Td_{min}, d_{max} + Td_{max}]$, since rotational motion is applied first, i.e., $\Phi(\mathcal{X}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$ and d is bounded in $\mathcal{D}$.

Let us also define the circle $C(h, r)$, centered at the target position $x^\star$, $h = T(x_1^\star, x_2^\star)$ and its radius is the distance from the target position $r = (AT) = d(x_{1:2}^A, x^\star)$. If the robot after translational motion is located inside this circle then the distance in one step, is decreasing.

Let $x^{\Gamma}_{1:2}$ be a point in the circumference of circle C, as illustrated in Fig. 6.4. By definition, $(A\Gamma)$ chord's length, $d(x^A, x^{\Gamma})$, is equal to $2d(x^A, x^\star)\sin(\frac{\psi_2}{2})$. We assume that the robot traverses the $(A\Gamma)$ chord by performing a translational motion with a combined error $\theta \in \Theta$. Using the circle's identities, we get that $\psi_1 = \theta$, as $(A\Gamma T)$ is an isosceles triangle and, thus, $\psi_2 = \pi - 2\theta$. This means that for the $(A\Gamma)$ length we get that $d(x^A, x^{\Gamma}) = 2d(x^A, x^\star)\cos(\theta)$.

Next, suppose that $x^{\Delta}_{1:2}$ is a point on $(A\Gamma)$ that the robot can reach in one step, i.e., $x^{\Delta}_{1:2} \in Z^{1:2}_{t+1}$ for $t = t^\star + 1$. Using basic trigonometric identities, we get that the distance between Δ and the target point $x^\star$, i.e., $d(x^{\Delta}, x^\star)$, is minimized if Δ is a point on the perpendicular bisector of $(A\Gamma)$, i.e., when its distance from point A is equal to $\frac{d(x^A, x^{\Gamma})}{2} = d(x^A, x^\star)\cos(\theta)$. Now, let us assume the general case where

$$d(x^A, x^{\Delta}) \leq \lambda_2 d(x^A, x^\star)\cos(\theta), \tag{6.36}$$

where $\lambda_2 \in (0, 2)$. This parametrization guarantees that point Δ is inside the defined circle. The coordinates of x^{Δ}, can be computed by eq. (6.4): $x^{\Delta} = g_1(x^A, u_1, \delta)$. As a result, $d(x^A, x^{\Delta}) = \|x^{\Delta}_{1:2} - x^A_{1:2}\|_2$. Using eq. (6.4) and involving trigonometric identities ineq. (6.36) becomes:

$$\sqrt{(\frac{u_1}{\delta})^2[2 - 2\cos(T\delta)]} < \lambda_2 d(x^A, x^\star)\cos(\theta). \tag{6.37}$$

Satisfaction of ineq. (6.37), which is by assumption true, implies existence of a u_1 satisfying ineq. (6.34), for $\mathcal{X}_t = x^A$ and for $\theta = \delta_M$. Next, we show that ineq. (6.37) implies

$$V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star) \text{ for } t = t^\star + 1. \tag{6.38}$$

To this purpose, let us assume that, $V(\mathcal{X}_{t+1}, x^\star) = d(x^{\Delta}, x^\star)$, as point Δ lies arbitrarily in $\mathcal{X}_t$ for $t = t^\star + 1$ Exploiting the law of cosines for the $(A\Delta T)$ triangle we know that for $t = t^\star + 1$

$$V^2(\mathcal{X}_{t+1}, x^\star) = V^2(\mathcal{X}_t, x^\star) + d^2(x^A, x^{\Delta}) - 2V(\mathcal{X}_t, x^\star)d(x^A, x^{\Delta})\cos(\theta) \tag{6.39}$$

By replacing eq. (6.39) to eq. (6.38), then

$$V^2(\mathcal{X}_t, x^\star) + d^2(x^A, x^{\Delta}) - 2V(\mathcal{X}_t, x^\star)d(x^A, x^{\Delta})\cos(\theta) \leq \lambda^2 V^2(\mathcal{X}_t, x^\star) \tag{6.40}$$

By eq. (6.36), eq. (6.40) becomes:

$$V^2(\mathcal{X}_t, x^\star) + \lambda_2^2 V^2(\mathcal{X}_t, x^\star)\cos^2(\theta) - 2V^2(\mathcal{X}_t, x^\star)\lambda_2\cos^2(\theta) \leq \lambda^2 V^2(\mathcal{X}_t, x^\star), \text{ for } t = t^\star + 1. \tag{6.41}$$

And finally,

$$1 + \cos^2(\theta)(\lambda_2^2 - 2\lambda_2) < \lambda^2. \tag{6.42}$$

Ineq. (6.42) is satisfied for any $\lambda_2 \in (0, 2)$ and $\lambda \in (0, 1)$, thus, $V(\mathcal{X}_{t+1}, x^\star) < \lambda V(\mathcal{X}_t, x^\star)$, for $t = t^\star + 1$. $\square$

Theorem 3. *Consider the system* (6.2), (6.3)*. Then, by repeatably, invoking the remote estimation and performing first rotational motion and then translational motion, the robot converges to the target set* $\mathcal{G}$ *eq.* (6.24)*.*

Proof. By Lemma 1, after the remote estimation technique is invoked, the shift is orientation defined by problem (6.30a), guarantees that the translational motion is allowed in the next step, i.e.: $\mathcal{X}(t^\star + 1) \subseteq \mathcal{M}$. Furthermore, by Proposition 2, there is guarantee that $V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star)$ for $t = t^\star + 1$, namely the distance from the target position is decreasing. The robot repeats the following process when constantly invoking the remote estimation technique, namely (i) performing a rotational motion and (ii) proceeds after one time step to the translational motion. Consequently, for any initial condition $\mathcal{X}_0$ such that $V(\mathcal{X}_0, x^*) \leq c$, it follows that $V(\mathcal{X}_t) \leq \lambda^{\lfloor \frac{N}{2} \rfloor} V(\mathcal{X}_0, x^*) \leq \lambda^{\lfloor \frac{N}{2} \rfloor} c$, thus, the robot converges to $\mathcal{G}$ as defined in eq. (6.24), in at most $N^* \geq 2\lfloor \frac{\log \epsilon - \log c}{\log \lambda} \rfloor$ steps. □

Following this simple technique, i.e., initially rotating to fix the heading error and then proceed to translational motion, it is apparent that the convergence of the proposed technique is guaranteed in the case of the robot constantly invokes the remote estimation. As stated before, this computationally intensive method heavily affects the mission duration of the robot. Thus, while relying on the theoretical guarantee, next we need to find an offloading strategy to find a balance between accuracy and mission duration, whilst the dynamic conditions of the network and remote computation resources, are also considered.

6.7 Computational Offloading Decision Mechanism

In this section, the decision-making mechanism behind the offloading of the computationally intensive tasks of our framework is presented. This mechanism is based on a utility function decision-making procedure that quantitatively ranks the current conditions, in terms of their associated resource metrics (i.e., availability of networking and computing resources at the edge and quality of robot's navigation) and dynamically assigns tasks to be executed on the edge infrastructure. To get a reliable estimation of these conditions, network and computing resource profiles are developed, as described in the following subsections.

6.7.1 Network Profiling

In the following, mainly for demonstration purposes, we assume that the wireless access technique between the robot and the access point is based on IEEE 802.11g. In this network deployment, a common effect that occurs when a signal travels through a communication channel is its power level decreases as the distance increases. To estimate this propagation loss, the well-accepted Log-Distance Path Loss (LDPL) model is utilized [177]. The LDPL model applies to indoor environments with the presence of obstacles, having a propagation exponent that indicates whether the environment has more or fewer obstacles, impacting on the computed loss. The respective path-loss is calculated as follows:

$$PL(d)_{dB} = PL(d_0)_{dB} + 10nlog_{10}(\frac{d}{d_0}),\ d \geq d_0, \tag{6.43}$$

where $PL(d_0)_{dB}$ is the path-loss at a reference distance $d_0 = 1m$, n is the path-loss exponent (PLE), which depends on the presence of obstacles in the environment. To set the upper bounds of the channel capacity we also leverage the signal-to-noise-ratio (SNR) metric,

$$SNR(d) = P_{dB} - PL(d)_{dB} - N_{dB}, \tag{6.44}$$

where P_{dB} is the incoming signal to the access point and N_{dB} is a Gaussian noise. Then, the channel capacity C can be calculated using the Shannon–Hartley theorem,

$$C(d) = B\ log_2(1 + SNR(d)), \tag{6.45}$$

where B is the available WLAN bandwidth (in Hz), giving in this way an estimation of the tightest upper bound on the information rate of data (in bits per second) that can be communicated at an arbitrarily low error rate using SNR. Having this bound available, an estimation of the task transmission duration (in seconds) can be calculated as follows:

$$e_{ul}(d) = \frac{8\ m}{C(d)}, \tag{6.46}$$

where m is the size of the offloaded data in bytes.

6.7.2 Edge Computing Resources Profiling

Regarding the profiling of the computing resources at the edge, we assume that the allocation of the resources to the localization service on the server is managed by the resource orchestrator of the infrastructure provider. Hence, we can only estimate the amount of allocated resources through measurements. To this end, we model the resource allocation strategy on the edge server as a linear dynamical system, subject to process and measurement uncertainty disturbances which follow the standard Kalman Filtering estimation approach [178], a computationally light prediction method. Briefly, using previous CPU utilization measurements, allows for acquiring a current estimation of the virtual CPU cores allocated to the container, $\hat{c}(t)$, dedicated the remote estimation technique. Subsequently, this allows for estimating the expected computation time (in seconds) of the offloaded task to the edge server, $e_{comp}(t)$, by modeling it as a linear relationship of the available resources, specifically:

$$e_{comp}(t) = \alpha\ \hat{c}(t) + \beta. \tag{6.47}$$

The coefficients α, β are calculated using a least-squares fitting method on a set of pairs $(e_{comp}(t), \hat{c}(t))$ produced offline while experimenting with a dataset of pictures from the robot's camera. The interested reader may also refer to [146] for further information on this process, as it is based on this previous work.

6.7.3 Utility-based Offloading Strategy

After evaluating the current network and available edge computing resources, we apply the results in a utility function in order to quantify the trade-off between remote execution time and navigation performance. Naturally, this utility function incorporates a term, which assesses the quality of the navigation till the point of the offloading decision making, t, which in our case is expressed as a fraction (rational number) that is the quotient of the volume of the estimation set $\hat{\mathcal{X}}_t$, divided by the robot's remaining maximum distance of the estimation set towards the target position, $V(\hat{\mathcal{X}}_t, x^\star)$, as introduced in Section 6.6. The rationale behind this inclusion lies in the fact that the smaller the distance is between the robot and the target, the more precise the navigation has to be in order to approach it ϵ-close, thus the more preferable the offloading. Additionally, in Subsection 6.6.3, we identified a hard constraint for resorting to remote execution, expressed by the value of $\kappa(t)$ in eq. (6.33), which also has to be engaged in the offloading decision.

Therefore, taking the above into consideration, the utility function of edge computing processing for the robot can be defined as:

$$q_o(t) = c_1 \cdot \frac{Vol(\hat{\mathcal{X}}_t)}{V(\hat{\mathcal{X}}_t, x^\star)} - c_2 \cdot e_o(d,t) + \kappa(t) \cdot C, \tag{6.48}$$

where $e_o(d,t) = e_{ul}(d) + e_{comp}(t)$ denotes the total estimated duration of the remote execution. We should note here that, similarly to other studies [179], in the envisioned application, the duration of the transmission of the computation results is negligible, as the size of the computation results (plain text) is much smaller than the size of the input data (image or video). Thus, in our case, the total duration specifically involves the uplink transmission and the computation execution time. The $c_1, c_2 \geq 0, \in \mathbb{R}$ coefficients are carefully selected after thoroughly experimenting with different combinations in order to reflect the desired balance of mission duration and accuracy. $C \geq 0, \in \mathbb{R}$ represents a very large number.

By carefully examining eq. (6.48), one can notice that the system's utility increases as the navigation accuracy decreases, while decreases as the total offloading duration increases and is maximized when offloading is deemed necessary, i.e., $\kappa(t) = 1$. Leveraging the above, we define $O(t)$, a function that dictates whether the task is offloaded to the edge ($O(t) = 1$) or not ($O(t) = 0$), as follows:

$$O(t) = \begin{cases} 1, & \text{if } q_o(t) \geq J_{TH} \\ \\ 0, & \text{otherwise,} \end{cases} \tag{6.49}$$

where $J_{TH} \ll \mathcal{C}$ is a predefined threshold, the value of which, together with the values of c_1, c_2 and C, tunes the sensitivity of our mechanism in triggering the remote execution technique and depends on the mission's characteristics.

6.7.4 Convergence of the Proposed Technique and Core Algorithm

To summarize the whole operation of the proposed CPS, as introduced in Section 6.6, the robot's motion, dictated by the control automaton of Fig. 6.2, relies at first on local techniques for pose estimation. On the one hand, the robot performs a translational motion using the respective controller computed by eq. (6.25) and (6.26), only when it is guaranteed that the distance towards the target position for the whole estimation set decreases. On the other hand, the robot performs a rotation using the controller computed by problem (6.30a), only when it is guaranteed that in the next step the translational motion is allowed. In both cases, an overapproximation of the estimation set is calculated, as explained in Subsection 6.5.2.2. The offloading strategy switches to the remote estimation according to the system's utility, given by the utility function (6.48). This utility function dictates the robot to invoke the remote estimation whenever at least one of the two conditions for convergence, eq. (6.29) and (6.32), is not satisfied, or when the communication conditions and/or the available computing resources result in a fast computation for the precise remote estimation technique. In this case, given the precise estimation, by Theorem 3, the distance between the robot and the target position decreases. Thus, using a combination of the local and remote estimation technique, the robot eventually converges to the target position.

Corollary 1. *Consider the system* (6.2)*,* (6.3) *and the utility-based offloading strategy defined in* (6.48)*. Then, the robot converges to the target set $\mathcal{G}$ eq.* (6.24)*.*

Proof. The proof of Corollary 1 follows the same reasoning as in Theorem 3, with the exception that the convergence speed is $\hat{\lambda} = \max_t \lambda'(t)$, as given by eq. (6.29) and that the rotation is invoked at worst every two time instants. □

6.8 Performance Evaluation

In this section, we provide a detailed numerical performance evaluation of the proposed technique, through modeling and simulation of various scenarios, illustrating the operation, features and benefits of our approach. Specifically, in Subsection 6.8.1, the detailed configuration of the experimental setup is described. In Subsection 6.8.2, we focus on the control sequence benchmarking. To this end, we identify two metrics, based on which we evaluate the performance of our mechanism, namely the *a) navigation accuracy* and *b) mission duration* and different offloading strategies are compared. In Subsection 6.8.3, the utility function is evaluated for different application scenarios, indicating the easily adapted applicability of the proposed technique. The benchmarking is conducted by simulating the motion of a mobile wheeled robot in Python. For the calculation of $\hat{\mathcal{X}}$, we utilize Flow*, described in [180] and [181]. Flow* is a software used for calculating reachable sets

Parameter	**Value**
ϵ	$0.2\ m$
$\mathcal{D}$	$[0.1, 0.2]\ rad$
w_1	$[-0.2, 0.03]\ m$
w_2	$[-0.06, 0.04]\ rad$
$x^\star$	$(13, 14)$
$\mathcal{X}_0$	$(3, 2)$
$\mathcal{J}_{TH}$	10
$\mathcal{U}_1$	$[0, 5]\ m/s$
$\mathcal{U}_2$	$[0, 2\pi]\ rad/s$

Table 6.3: Simulation parameters.

in hybrid automata, together with other dedicated software (e.g., CORA[1]) [182]. The simulation code, alongside any related dataset used in this section, is publicly available[2].

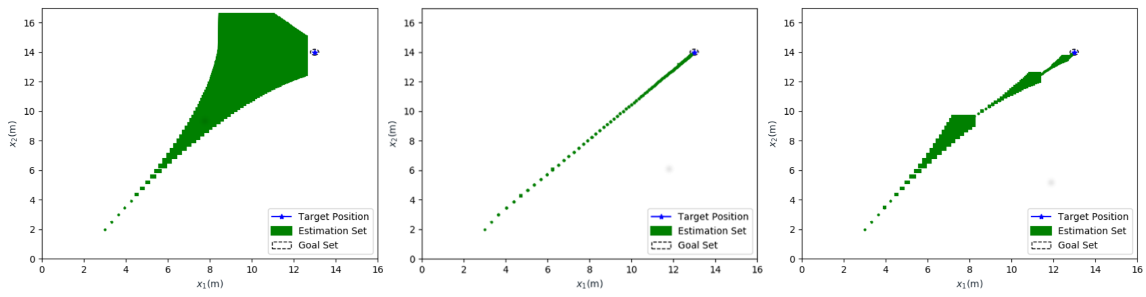
6.8.1 Experiment Setup

In order to have a realistic setting, for evaluating the described scenarios, we simulate a square, $20m \times 20m$ sized factory floor where five wireless access points are located. Four of them are placed in the corners of the floor and the last one is places in its center. As the simulated robot, an AlphaBot[3] is selected, thus, the local pose estimation is performed by using the robot's photo-electric sensors (encoders) attached to each wheel, which provide an estimation about the travelled distance and the shift in orientation. Moreover, the discretization time is set at $T = 0.1s$. The remaining key experiment parameters are listed in Table 6.3, unless otherwise explicitly stated. Additionally, regarding the mission, we consider that the robot starts from a known position $\mathcal{X}_0$ and stops when it reaches ϵ-close to the target position $x^\star$; for illustration purposes and in order to be able to average the following results over multiple repetitions of the experiment, the starting and final position are kept always the same.

Regarding the networking settings, we assume a signal of power P_{dB} for the uplink, which is proportional to the distance between the robot and the access point it is connected to and which has a maximum value of $P_{dB}^{max} = 24dB$. Moreover, we fix $PL(d_0)$ at $-20dBm$, based on the work of [177], which presents an access point with the same characteristics of ours and the same reference distance. The path-loss exponent n is set equal to 3.5, a value typical for a factory floor setting [183]. The size of offloaded data, in MB, follows a uniform distribution with a mean value of 0.075 and variance equal to 0.25. The Gaussian Noise N_{dB} is set equal to $-114dB$ while the bandwidth B allocated to the robot at any given time is set to $1MHz$.

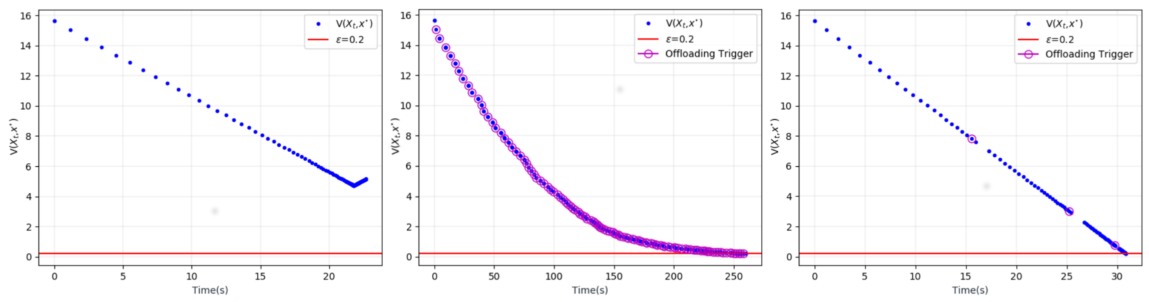
Finally, regarding the edge computing resources, as mentioned in subsection 6.7.2 and similar to [146], a least-squares fitting method is used to calculate the coefficients $\alpha = -1.34$ and $\beta = 3.675$

[1]https://swmath.org/software/25659
[2]https://github.com/Dspatharakis/Replan
[3]https://www.waveshare.com/wiki/AlphaBot



(a) Exclusively Local. (b) Exclusively Remote. (c) Balanced Utility-based.

Figure 6.5: Estimation set propagation for the three offloading schemes.



(a) Exclusively Local. (b) Exclusively Remote. (c) Balanced Utility-based.

Figure 6.6: Decrease of distance $V(\hat{\mathcal{X}}_t, x^\star)$ for the three offloading schemes.

for the computation of the remote estimation technique. Also, the allocated cores of the edge server $\hat{c}(t)$ are updated every $0.5s$, following a Normal Distribution with a mean value of 0.75 and variance equal to 0.5.

6.8.2 Control Sequence Benchmarking

In this scenario, we compare the performance of the following three computational offloading schemes, in terms of navigation accuracy and mission duration: *i) exclusively local estimation*, where the robot never invokes the remote estimation algorithm, *ii) exclusively remote estimation*, where the robot constantly invokes the estimation algorithm, *iii) utility-based offloading decision*, where our proposed mechanism comes into operation. For each of these three cases, a set of 35 experiments is conducted and the results are averaged per time slot for better illustration.

Fig. 6.5 depicts the changes in the estimation set volume (per second), as the robot moves and offloads tasks. The estimation set is depicted in 2-D, as the robot's position (and not its orientation) is sufficient to satisfy the termination condition (eq. (6.24)). Alongside this, Fig. 6.6 shows how quickly the robot reaches ϵ-close to the target in each case. As expected, in case *i)*, where only local estimations are used, although the robot moves rather quickly as shown in Fig. 6.6a, it consistently fails to approach the target as the localization error is accumulated in each

Offloading Scheme	**Average mission duration (s)**	**Average offloading triggers**	**Success Rate**
Exclusively Local	25	0	3%
Exclusively Remote	270	95	100%
Utility-based	33	3	100%
Time-triggered ($\mu = 5$)	95	26	100%
Time-triggered ($\mu = 10$)	58	11	100%
Time-triggered ($\mu = 15$)	48	7	100%
Time-triggered ($\mu = 35$)	35	3	100%
Time-triggered ($\mu = 50$)	38	3	95%

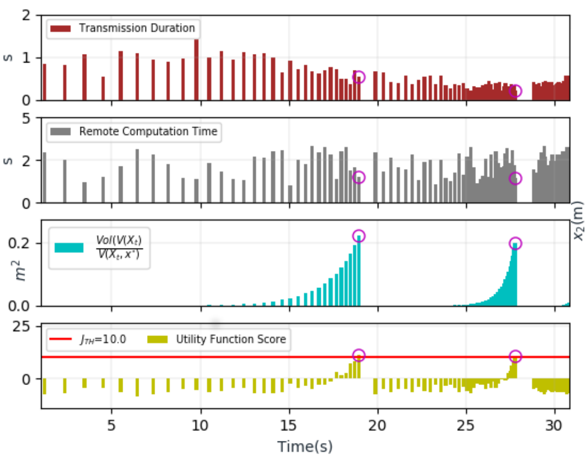
Table 6.4: Experiment metrics for each offloading scheme.

step. This is illustrated in Fig. 6.5a. There, the vast increase of the size of the estimation set, due to this accumulated error, indicates that the robot can not converge to the target position. It must be noted that, in the 35 repetitions of this setting, only once the robot managed to reach ϵ-close to the target, as shown in Table 6.4. On the other hand, case *ii)* showcases great precision in localization, which also reflects at the size of the volume of the estimation set, which is fixed at the minimum value (point) throughout the mission, as shown in Fig. 6.5b. However, this comes with a cost in mission duration which is significantly increased due to the robot uncritically invoking a time-consuming estimation technique, despite the networking conditions and available edge computing resources not being always favorable.

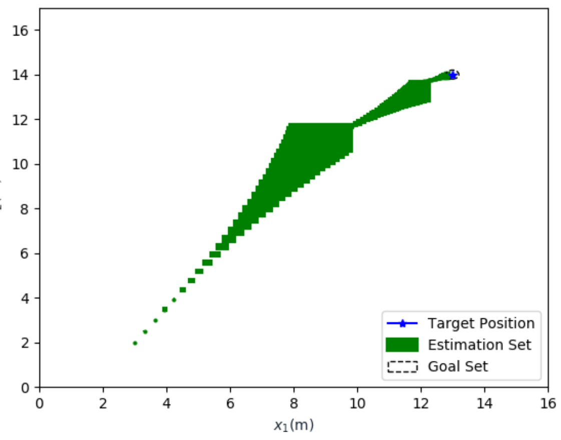
After the evaluation of the first two cases, it becomes evident that a balance between the investigated metrics, i.e., navigation accuracy and mission duration, is of paramount importance and our method, *iii*), manages to deliver one.

As illustrated in Fig. 6.5c, in the beginning, where the robot moves quickly using the local pose estimation, the estimation set increases over time. Then, the utility-based offloading strategy invokes the remote estimation technique three times, specifically at the $16th$, $25th$ and $29th$ second of the experiment. In these moments, a precise estimation of the robot's pose is provided, thus the estimation set is minimized to a point. After offloading, the robot moves again faster towards the target position, as long as the size of the estimation set allows it. In this setting, the average mission duration is kept to a low $33s$, which is only 10% longer in duration than the single successful try of case *i)*. We should note here that the coefficients of the utility function of eq. (6.48) were fixed to values that provided a balanced outcome between the evaluation metrics, in order to reflect a generic mission. In the next subsection, an evaluation on the different values and dynamics between these coefficients follows.

Finally, along with the three aforementioned offloading schemes we evaluate five different settings of a *time-triggered* remote estimation, a variation of case *iii)*, where the remote estimation is now triggered periodically every μ seconds. Although this time-triggered estimation provides,



(a) Utility function breakdown.



(b) Estimation set propagation.

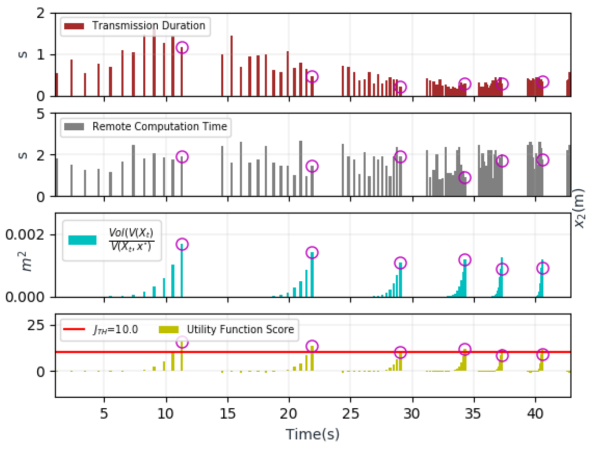
Figure 6.7: Time-critical mission.

in some cases, similar results to our case, it has no convergence guarantees. This is elucidated in Table 6.4, where the averaged results for the set of 35 experiments, for all the offloading schemes are presented. As one can notice, our technique is superior to all the other offloading strategies when it comes to providing a guaranteed convergence to the navigation target, while keeping the mission duration low.

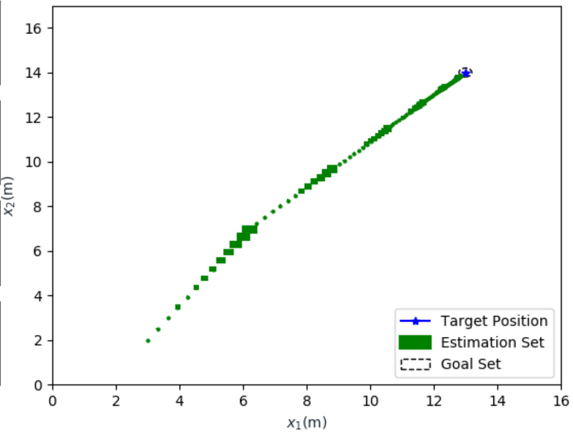
6.8.3 Offloading Strategy Benchmarking

Having evaluated the overall usefulness of the proposed mechanism, in this section, we investigate the effect of the coefficients of the utility function, on navigation accuracy and mission duration. To this end, we tweak the coefficients of eq. (6.48) to steer the offloading strategy towards benefiting one of the two metrics, depending on the mission requirements. In this context, we identify three mission-scenarios, differing on which term of the utility function gets promoted and the results this has on the mission metrics: *i) time-critical mission*, (e.g., rescue robots), *ii) navigation-critical mission*, (e.g., autonomous museum tour guides) and *iii)* sparse communication mission (e.g., space robotics). The first case differs from the exclusively local execution one of the previous subsection, as we guarantee a 100% mission success rate, i.e., the robot always reaches the target.

For case *i)*, we tweak the coefficients of the utility function to benefit the mission duration. Consequently, c_2, which is associated with the total duration of the remote execution of the precise estimation technique is promoted, while c_1, which is associated with the quality of the navigation, is demoted. Hence, the time-consuming remote estimation is invoked either when the total duration of remote execution is really low, compared to the local estimations, or when the estimation set has grown vastly, as shown in Fig 6.7a. In detail, the remote estimation technique is invoked two times in this experiment, when the volume of $\hat{\mathcal{X}}_t$ results in a great deterioration of the quality of the navigation, as shown in Fig. 6.7b and at the same time the total duration of remote execution



(a) Utility function breakdown.



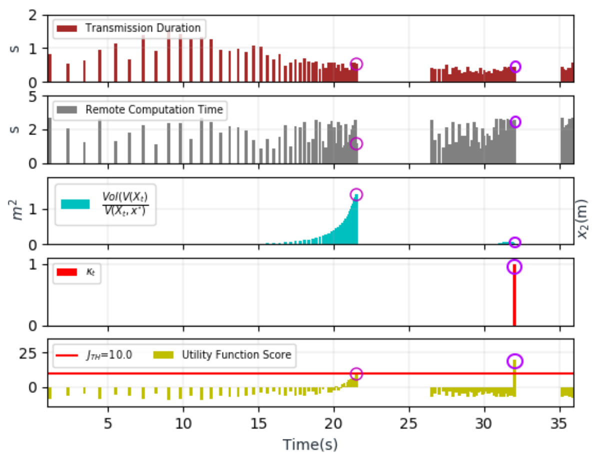
(b) Estimation set propagation.

Figure 6.8: Navigation-critical mission.

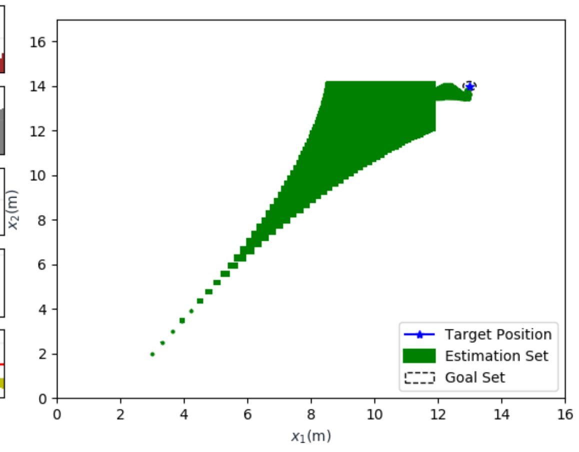
is minimized. The average mission duration for this setting was $33s$.

On the other hand, in case *ii)*, to address the need for a more fine-grained navigation, c_1 is promoted and c_2 is demoted, thus, the estimation set's volume is not allowed to grow uncontrollably. This results, as presented in Fig 6.8a, in invoking the remote estimation technique more frequently, i.e., 6 times during this experiment. In this case, the quality of navigation is more important than execution time; this is evident in the $1st$, $3rd$ and $6th$ offloading trigger, where the network conditions and available edge resources are not so preferable, but, still the robot chooses to offload in order to minimize the localization uncertainties. In addition, naturally, the volume of the estimation set, depicted in Fig. 6.8b, is relatively compared to the first case, resembling the case of exclusively offloading of Subsection 6.8.2. The average mission duration for this setting is 33% longer ($45s$) than case *i)*, as expected.

Finally, in case *iii)*, we minimize c_1, to simulate a scenario where the robot invokes the remote estimation algorithm only when it is an absolute necessity. This setting is a special case of case *i)* to indicate that the mission duration also depends on the increase of the estimation set. Therefore, as shown in Fig. 6.9a, in this case we notice only two offloading triggers, namely; the $1st$ due to the increase of the estimation set, at the $22nd$ second of the experiment and the $2nd$ due to the $\kappa(t)$ signal trigger at the $33rd$ second. As presented in Subsection 6.6.3, $\kappa(t)$ becomes equal to 1 either when the distance towards the target is not decreasing or the rotational controller fails to allow translational motion in the next time instant. As shown in the same figure, regardless of the current network and computing conditions, the robot seeks a more precise estimation to guarantee that the target is reached. A last note is that the average mission duration for this setting is 15% longer than time-critical experiment ($38s$). So, a conclusion that is drawn is that the bigger the estimation set, the more it deteriorates the quality of the navigation, under the proposed controller design.



(a) Utility function breakdown.



(b) Estimation set propagation.

Figure 6.9: Sparse communication mission.

To sum up, it is noticeable that promoting c_1 results in more precise yet time-consuming navigation, as the robot offloads more frequently. On the other hand, when promoting c_2, the robot offloads rarely, when absolutely necessary, resulting in quicker but less precise navigation. Moreover, in the special case, where c_1 is minimized, the robot offloads either when the estimation set grows vastly, or when the conditions that guarantee convergence are not satisfied. It should be noted that, while all three configurations manage to achieve a 100% success rate in reaching the target position, the first one is allowed to roam more freely, relying on its local navigation capabilities in favor of speed, the second one manages so in a more fine-grained trajectory while the third one invokes the remote estimation technique only when necessary. To sum up, the utility-based offloading decision framework manages to achieve the required balance among the various mission characteristics, making it suitable in the context of co-design of 3C for a CPS.

6.9 Summary

A novel resource-aware estimation and control framework for edge robotics, that jointly tackles the problem of convergence in trajectory navigation of a unicycle robot and the problem of efficiently using communication and computing resources, is presented. The conservative overapproximation techniques introduced alleviate additional computationally intensive tasks from the resource-constrained robot and provide a quick solution to the challenging problem of calculating the estimation set in the presence of modeling and measurement uncertainties. Moreover, we propose controllers that guarantee the robot reaches a target position after a finite number of steps. Finally, a utility-based offloading decision strategy is presented and thoroughly evaluated to highlight the need of finding a balance between two important metrics, namely navigation accuracy and mission duration. The performance evaluation of the proposed technique suggests that our solution outperforms other typically used offloading schemes and is easily adjustable to the needs of different application characteristics. More importantly, the proposed framework guarantees convergence to the target position independent of the various parameters chosen, in contrast to the periodic offloading schemes. The results of this Chapter are also presented in [184].

Chapter 7

A Task Offloading and Batch Scheduling Framework for Edge-assisted Inference

7.1 General Setting

The era of modern applications has witnessed the emergence of new computing paradigms that alleviate the computational burden of complex algorithms from the devices. There are several scenarios in which conducting the best quality computations provides disproportionate benefits in terms of performance and energy of the overall system [185]; a fact that paves the way for an adaptive computing paradigm, namely, *Approximate Computing* [41] or *Transprecision Computing Paradigm* [186]. The basic idea of this paradigm is to reduce the number of computations an application must perform, thus, reducing the execution time but producing a potential but acceptable loss in accuracy. Approximate computing is applied at various levels, including hardware acceleration (e.g., CPUs, GPUs, etc.) and software (e.g., model compression [38]).

This Chapter focuses on the case of real-time inference (i.e., image detection) applications deployed in various scenarios such as AR/VR, mobile robotics, and Smart City applications, among others [187]. We assume that inference tasks (e.g., images or video stream) generated by low-power and compute-constrained EDs are processed either locally or offloaded, using a wireless connection, to the ECs for further processing. The offloaded tasks are first compressed on the EDs to reduce the transmission time [188]. Accordingly, when an offloaded task arrives at an EC, it is decoded, and then the inference is executed. Thus, we investigate the trade-off between the minimization of transmission time (i.e., compression of task size) and the accuracy loss of the inference. Furthermore, we consider that each EC has an available GPU unit that assists with the computations. On the EC side, we assume that the offloaded tasks are executed in parallel, i.e., in a batch. The batch processing of tasks drastically decreases the overall latency compared to the sequential execution of the tasks [189]. Moreover, the time the GPU is operating is reduced, thus minimizing the cost and energy from the EC side. However, for the batch processing to be optimized, the offloaded tasks must arrive at the EC simultaneously to reduce the total latency [190].

7.2 Related Work

In this section, we present studies related to the task offloading problem and Edge-assisted real-time inference.

7.2.1 Task Offloading

Leveraging the Edge Computing capabilities, there are numerous works related to the task offloading problem subject to various metrics (e.g., latency, bandwidth utilization, energy). Here, we present the most recent studies on SDR-based task offloading for the binary case.

In [191], the authors propose the modeling of a single device that has a set of tasks to execute and a set of ECs that the tasks can be offloaded. They consider two different cases of resource allocation on the ECs, namely; fixed and elastic CPU frequency, and they solve the QCQP with SDR. The results indicate that SDR provides near-optimal solutions and outperforms all other baseline techniques. In Chen et. al. [192], the authors assume only one EC and a set of users. The task is executed either locally or remotely on the EC. They formulate the Multi-User Multi-Task Offloading algorithm that solves, using SDR, the binary offloading problem and then maps the solution to a feasible one using stochastic randomization. Mukherjee et. al. [193] design a computation offloading strategy to minimize a utility function defined as the weighted sum of every device's delay and energy consumption, in a UAV scenario. Then, they formulate a binary offloading problem that is solved with SDR as the problem is a Quadratically Constrained Linear Program. The SDR solution is then used to train a Deep Neural Network model to find a near-optimal offloading strategy. Finally, in [194], a joint task offloading and resource allocation problem for mobility scenarios is presented. The proposed framework utilizes Lyapunov optimization combined with Semi Definite Program (SDP) to optimize the offloading ratio, power, and CPU frequencies of the ECs.

7.2.2 Edge-Assisted Inference

This section briefly presents studies that tackle the problem of optimizing inference accuracy in an Edge-assisted offloading scenario.

In particular, in [195], the authors propose a framework where each device offloads an inference task taking into account the network quality. If the channel quality is adequate (in terms of signal strength) then the task is offloaded to the EC. Otherwise, the tasks (images) are compressed to reduce the transmission latency and manage to receive the result within the frame rate. Similarly, in [196] the decision for offloading is based on the network and the application's requirements; a problem is formulated that simultaneously decides the frame rate and frame resolution, the bit-rate of the video along with the model selected for the inference. The proposed framework focuses on executing a convolutional neural network (CNN) for real-time augmented reality applications. Moreover, the authors in [197] utilize a dynamic Region Of Interest (ROI) encoding technique that adjusts the encoding quality for each subframe of the original picture based on the latest inference. They introduce a real-time framework with dynamic offloading policies to minimize latency and maximize accuracy for an AR application. In [188], the authors address the issue of real-time object detection under poor communication conditions. They run a lightweight deep neural network model on the device to find candidate ROIs and then encode with higher quality the ROIs by controlling the compression ratio (i.e., bitrate). Then, a fine-grained offloading strategy is introduced to limit the frequency of offloaded images to the ECs. Specifically, offloading occurs only when the results of the local inference are inaccurate or significant changes in the motion of the objects are observed.

In the following we also highlight two research works that try to formulate optimization problems regarding the maximization of accuracy for Edge-assisted inference. Fresa et. al. [198] propose an approximation algorithm to maximize the total accuracy for inference tasks subject to latency constraints of the makespan. The authors use model selection and formulate an NP-hard problem that is linearized in order to achieve a feasible solution. Also, in [199], an NP-hard problem is formulated to investigate the trade-off between latency and accuracy in a 3-layered architecture. Tasks can be either executed locally or offloaded to the edge and the cloud layer. The joint optimization of compression and offloading is similar to ours, however, the offloading decision is approximated using a heuristic algorithm with a moving average technique. Moreover, the solution provided in [199] is not compared to the optimal solution. In difference with the above-mentioned works, we formulate an optimization algorithm that jointly minimizes the average latency and maximizes the accuracy of the inference process, by computing the optimal values for the compression of the offloaded tasks and employing them to compute an SDR-based task offloading decision for batch scheduling.

7.3 Contributions & Outline

Overall, we aim at a task-offloading strategy for batch scheduling that (a) facilitates the collaboration between the EDs and the ECs to reduce the average latency and (b) maintains a certain quality of the inference process. Our contributions are summarized as follows:

- A two-layer environment is introduced, where EDs produce inference tasks and seek additional computing resources from the ECs, in an Edge Intelligence scenario. We formulate a joint optimization problem to maximize the average accuracy and simultaneously minimize the average latency of Edge-assisted inference. We introduce a framework to approximate the solution of the optimization problem by breaking it into two sub-problems: (a) compute an optimal task compression and (b) decide a task offloading policy for batch scheduling.

- Following the Approximate Computing paradigm, we consider two inference models, namely: (i) a computationally light for the resource-constrained ED and (ii) a precise one executed on the GPU-enabled ECs. Thus, we investigate the trade-off between the expected quality of the inference and the transmission time of tasks. We first compute an optimal compression strategy subject to a task offloading policy. The solution to the compression problem introduces some important results which imply that the batch scheduling is optimized when the tasks arrive simultaneously at the EC, thus, minimizing the total latency of the inference process.

- The task offloading problem remains challenging as it is a mixed-integer linear program. We formulate the equivalent Quadratically Constrained Quadratic Problem (QCQP) since the binary offloading constraints are expressed as quadratic equations [200]. Therefore, we exploit the power of Semi Definite Relaxation (SDR) to achieve a near-optimal solution while maintaining polynomial complexity based on the results of the compression problem. We utilize a randomized method to obtain a feasible solution that converges close to the optimal solution with low polynomial complexity, as the results indicate. We propose an algorithm that iteratively solves the two subproblems until convergence.

- We perform an extensive evaluation of the proposed framework via modeling and simulation. The superior performance of the proposed approach is demonstrated in comparative experiments, as the total cost is at least 50% less than all other benchmark techniques, and improves even further as the number of tasks increases. Finally, we investigate the effect of all significant parameters on the overall performance of the framework in a series of experiments.

7.4 System Model and Problem Formulation

We consider a two-layer computing environment, involving a set of homogeneous (in terms of processing capabilities) EDs that produce tasks $\mathcal{N} = \{1, \ldots, N\}$ and a set of ECs $\mathcal{M} = \{1, \ldots M\}$ located in the proximity of the devices. Each task $i \in \mathcal{N}$ contains one or more images, or ROIs, of a similar size (in bytes), which is denoted by d_i and is bounded by a maximum data size; i.e., $d_i \leq d^{max}$. The images are used for real-time object detection algorithms e.g., using the newest version of YOLO software [201]. The EDs communicate with the ECs via wireless access points. A high-level illustration of the systems' overview is depicted in Fig 7.1. Each ED has to decide the task scheduling strategy, i.e., whether to execute the inference process locally or offload to an EC. Hence, we consider two binary offloading decision variables, x_i, for $i \in \mathcal{N}$ and y_{ij}, for

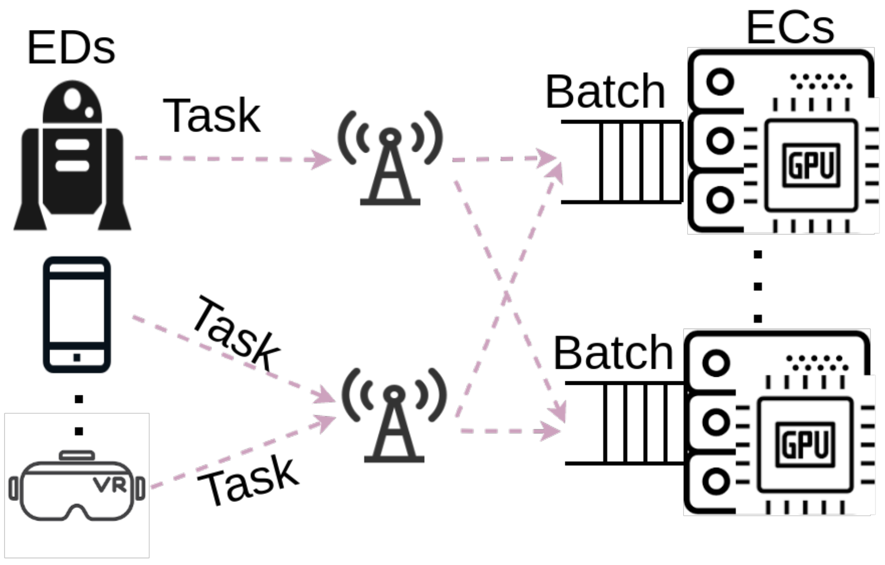


Figure 7.1: High-level concept of Inference Offloading for GPU-enabled ECs.

$i \in \mathcal{N}, j \in \mathcal{M}$, to denote the task scheduling:

$$x_i = \begin{cases} 1, \text{ the } i^{th} \text{ task is executed locally} \\ 0, \text{ the } i^{th} \text{ task is offloaded} \end{cases},$$

$$y_{ij} = \begin{cases} 1, \text{ the } i^{th} \text{ task is assigned to EC j} \\ 0, \text{ otherwise} \end{cases},$$

We denote the task scheduling matrix as

$$\mathbf{X} = \begin{bmatrix} x_1 & y_{11} & \cdots & y_{1M} \\ \vdots & \vdots & & \vdots \\ x_N & y_{N1} & \cdots & y_{NM}. \end{bmatrix}.$$

Observe that $\mathbf{X}$ is a $N \times (M+1)$-matrix. To assist the reader, a summary of the key notations used in the text is provided in Table 7.1.

In addition, we consider that each edge server is mounted with a GPU unit to assist with the image detection process. Following similar works (see [202]), we let the GPUs execute multiple tasks in parallel to reduce the total computation delay. We denote b_j, for $j \in \mathcal{M}$, the batch of parallel executed tasks, following [190]. Moreover, for each EC $j \in \mathcal{M}$, we consider a maximum batch size b_j^{max}. Let $\mathbf{b}^{\max} = [b_1^{max}, \ldots, b_M^{max}]$ be the $(M \times 1)$-matrix containing the maximum batch size of each GPU-enabled EC. We also denote $s_i, i \in \mathcal{N}$, the compression ratio of the data

Table 7.1: Summary of the key notation.

Symbol	**Interpretation**
N	number of EDs
i	index of an ED
M	number of ECs
j	index of an EC
x_i	local offloading decision
y_{ij}	remote offloading decision
$\mathbf{X}$	task scheduling matrix
d_i	data size for task i
R_{ij}	data rate between ED i and EC j
s_i	compression factor for task i
$g(s,k)$	expected accuracy related to compression
b_j^{max}	maximum batch size for EC j
L_{ij}^{tran}	transmission latency of task i offloaded to EC j
L_{ij}^{comp}	computation latency of task i offloaded to EC j
L_j^{wait}	waiting time at EC j
L_i	overall latency of task i
$A(\mathbf{X},\mathbf{s})$	average accuracy of all tasks
$T(\mathbf{X},\mathbf{s})$	average latency of all tasks
T_{max}	maximum allowed latency
A_{min}	minimum allowed accuracy
$\Psi(\mathbf{X},\mathbf{s})$	objective function
λ_T	scalar weight for latency
λ_A	scalar weight for accuracy
ϵ	solution convergence threshold

size of the tasks which are offloaded. We further assume that the data size of the tasks is known beforehand. As mentioned in the introductory section, in the case of offloading, the tasks are first compressed using the appropriate software (e.g., libjxl [203]) and then sent to a specific EC, to reduce the communication overhead. Following the Approximate Computing paradigm, we consider two different inference models to perform the image recognition algorithm, namely; (i) a computationally light but inaccurate model executed locally on the constrained ED and referred to as k_{loc}, and (ii) a precise model executed on the ECs, which is denoted by k. This practice is common when executing inference in computing-constrained EDs [204]. Therefore, in order to express the relationship between the expected accuracy of a machine learning model and the image compression of each task, we use the following function, following [205]:

$$g(s,k) = \alpha_k \exp(-\beta_k \exp(-\gamma_k s)) \,, \tag{7.1}$$

where k denotes the selected machine learning model and $s \in (0,1]$ is the compression ratio used for data compression (i.e., $s = 0.9$ means that the output size of the task is at 90% of the original size). This function is known in the literature as the *Gompertz function* and captures the idea that as the compression ratio increases there is a significant loss in the accuracy of the inference. It is easy to see that the concavity of the Gompertz function depends on the parameters α, β, γ. Hence, we choose the parameters α, β, γ in such a way that (a) the concavity of the function is guaranteed, and (b) the local and remote machine learning models produce different maximum accuracy. Finally, we denote by $\mathbf{s} = [s_1, \dots, s_N]$, the $(N \times 1)$-compression matrix. We now proceed with defining the accuracy of the system.

1. Accuracy: The average accuracy of all tasks is given by:

$$A(\mathbf{X}, \mathbf{s}) = \frac{1}{N} \sum_{i \in \mathcal{N}} \bigg(x_i \cdot g(1, k_{loc}) + \sum_{j \in \mathcal{M}} y_{ij} \cdot g(s_i, k) \bigg), \tag{7.2}$$

where k_{loc} is the local machine learning model used for the inference and k is the machine learning model used by the ECs. Clearly, in the case of local execution, compressing the data is redundant, and therefore the compression ratio is equal to 1.

Let R_{ij} denote the data rate to transmit the images from the i^{th} ED to the j^{th} EC. We assume that the latency of transferring the response from the EC to the ED is considered negligible as in [206]. Then, let $L_{ij}^{tran}, L_{ij}^{comp}$ be the transmission and computational latency, respectively, when the task i is executed on the EC j. More specifically, we define

$$L_{ij}^{tran} = \frac{s_i d_i}{R_{ij}} \quad \text{and} \quad L_{ij}^{comp} = f_k(b_j)\,, \tag{7.3}$$

where $f_k(b_j) = \xi_k b_j + \zeta_k$, where ξ_k, ζ_k are constants and $b_j = \sum_{i \in \mathcal{N}} y_{ij}$, for all $j \in \mathcal{M}$. Driven by [190], the function $f_k(b_j)$ is a linear approximation to the computing latency of the inference for a batch of tasks b_j, when executed on a GPU; as showcased also in the results of our prior work [189]. Furthermore, from our prior work, we know that small deviations in the task size have an insignificant effect on the latency of batch processing. In the function $f_k(b_j)$, apart from the computational latency, we also include the time needed to encode the task on the ED side and decode it on the EC side. More information regarding the linear approximation of the latency is presented in Sec. 7.6.

However, in this chapter, we consider parallel processing of the tasks at the EC. This means that each EC starts the processing of the batch when the last of the tasks arrive. As a result, let

$$L_j^{wait} = \max_{i \in \mathcal{N}} \left\{ y_{ij} \cdot \frac{s_i d_i}{R_{ij}} \right\} \tag{7.4}$$

be the waiting time of all tasks $i \in \mathcal{N}$ offloaded at EC j. Evidently, if a task, say i, is offloaded at the j^{th} EC satisfies $\frac{s_i d_i}{R_{ij}} = L_j^{wait}$, then the waiting time for the specific task is zero. Moreover, we know that $L_j^{wait} \geq L_{ij}^{tran}$ for all tasks offloaded at EC j. We define the *makespan* of task $i \in \mathcal{N}$, that corresponds to the transmission, waiting time, and computing latency at the j^{th} EC when at least one task is offloaded, as follows:

$$L_i = \max_{i \in \mathcal{N}} \left\{ L_{ij}^{tran}, L_j^{wait} \right\} + L_{ij}^{comp} = L_j^{wait} + L_{ij}^{comp} \tag{7.5}$$

Having defined the accuracy if the system, we now proceed with defining its latency.

2. Latency: The average latency of all tasks is given by:

$$T(\mathbf{X}, \mathbf{s}) = \frac{1}{N} \sum_{i \in \mathcal{N}} \big(x_i L_{loc}(d_i) + \sum_{j \in \mathcal{M}} y_{ij} \left(L_j^{wait} + L_{ij}^{comp} \right) \big), \tag{7.6}$$

where $L_{loc}(d_i) = f_{k_{loc}}(d_i) = \xi_{k_{loc}} d_i + \zeta_{k_{loc}}$, is the *latency of the local execution.* Observe that, similarly as before, we consider a linear function of the data size, d_i, for the local computing time. As already mentioned, the same processing capabilities are considered for all EDs.

We aim to minimize the tradeoff between the average latency of the tasks produced by the EDs and the average inference accuracy. To this end, we formulate the total cost as the weighted sum of the two metrics:

$$\Psi(\mathbf{X}, \mathbf{s}) = \lambda_T T(\mathbf{X}, \mathbf{s}) - \lambda_A A(\mathbf{X}, \mathbf{s}), \tag{7.7}$$

where $\lambda_T, \lambda_A \in \mathcal{R}^+$ are tunable positive parameters. We subtract the second term in the objective function aiming to maximize the average accuracy of the tasks. Therefore the joint minimization problem can be formulated as:

$$\mathcal{E}1 : \min_{\mathbf{X}, \mathbf{s}} \Psi(\mathbf{X}, \mathbf{s}) \tag{8}$$

$$\text{s.t. } T(\mathbf{X}, \mathbf{s}) \leq T_{max} \tag{7.8a}$$

$$A(\mathbf{X}, \mathbf{s}) \geq A_{min} \tag{7.8b}$$

$$0 \leq s_i \leq 1, \forall i \in \mathcal{N} \tag{7.8c}$$

$$\sum_{i \in \mathcal{N}} y_{ij} \leq b_j^{max}, \ \forall j \in \mathcal{M} \tag{7.8d}$$

$$x_i + \sum_{j \in \mathcal{M}} y_{ij} = 1, \ \forall i \in \mathcal{N} \tag{7.8e}$$

$$x_i, y_{ij} \in \{0, 1\}, \ \forall i \in \mathcal{N}, j \in \mathcal{M}. \tag{7.8f}$$

The constraint (7.8a) guarantees that the average completion of the inference of all tasks is upper bounded by T_{max}, while constraint (7.8b) ensures that the average accuracy for all tasks is lower bounded by A_{min}. Furthermore, (7.8c) ensures that the compression ratio of the data at the ED side is greater than 0 and at most equal to 1. Constraint (7.8d) ensures that the amount of offloaded tasks to EC j is upper bounded by b_j^{max} for all ECs. Constraint (7.8e) also secures that each task is executed either on the ED or offloaded to a specific EC. Finally, (7.8f) describes the binary character of offloading, i.e., the task is not split into several tasks. Problem $\mathcal{E}1$ is a mixed integer nonlinear problem that is well-known to be NP-Hard [207].

7.5 Accuracy Maximization and SDR Relaxation

In this section, we discuss an optimization framework to efficiently find a near-optimal solution to Problem (7.8).

7.5.1 Optimization of Inference Accuracy

We first aim to find an optimal solution to the compression ratio of the offloaded tasks given a feasible task scheduling matrix $\mathbf{X}$. More concretely, we first optimize over the values of $\mathbf{s}$ in Problem (7.8), while considering the values of $\mathbf{X}$ as fixed.

$$\mathcal{E}2 : \min_{\mathbf{s}} \Psi(\mathbf{X}, \mathbf{s}) \tag{9}$$

$$\text{s.t. } (7.8a), (7.8b), (7.8c) \tag{7.9a}$$

Note that Problem (7.9) can be equivalently written as follows:

$$\begin{aligned} \min_{\mathbf{s}} \ & \frac{\lambda_T}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} y_{ij} \cdot \max_{\ell \in \mathcal{N}} \left\{ y_{\ell j} \frac{s_\ell d_\ell}{R_{\ell j}} \right\} + \\ & - \frac{\lambda_A}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} y_{ij} \cdot g(s_i, k) + C_A + C_T \end{aligned} \tag{10}$$

$$\text{s.t. } \frac{1}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} y_{ij} \max_{\ell \in \mathcal{N}} \left\{ y_{\ell j} \frac{s_\ell d_\ell}{R_{\ell j}} \right\} + \frac{C_T}{\lambda_T} \le T_{\max} \tag{7.10a}$$

$$\frac{1}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} y_{ij} g(s_i, k) + \frac{C_A}{\lambda_A} \ge A_{\min} \tag{7.10b}$$

$$0 \le s_i \le 1, \forall i \in \mathcal{N}, \tag{7.10c}$$

where $C_T = \frac{\lambda_T}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} \left(x_i L_{loc}(d_i) + \xi_{k_j} \sum_{l \in \mathcal{M}} y_{lj} \right)$ and $C_A = -\frac{\lambda_A}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} \lambda_A g(1, k_{loc} j)$ are independent of $\mathbf{s}$, and may be regarded as constants. in this chapter, we choose the parameters α, β, γ in the Gompertz function in such a way to guarantee that we consider only the concave part of the function that is convex in the negative form. To solve problem (7.10), since a task scheduling matrix $\mathbf{X}$ is known beforehand, we substitute the max operator that indicates the maximum transmission latency of all tasks towards each EC, with respective linear constraints [208]. Moreover, C_T, C_A are constants subject to the task scheduling matrix and are omitted as they do not contribute to the solution. Hence the equivalent optimization problem is:

$$\min_{\mathbf{s}} \frac{1}{N}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}\big(\lambda_T t_j - \lambda_A g(s_i,k)\big) \tag{11}$$

$$\text{s.t.}\ \frac{1}{N}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}t_j + \frac{C_T}{\lambda_T} \le T_{max} \tag{7.11a}$$

$$\frac{1}{N}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} y_{ij}g(s_i,k) + \frac{C_A}{\lambda_A} \ge A_{min} \tag{7.11b}$$

$$0 \le s_i \le 1, \forall i \in \mathcal{N} \tag{7.11c}$$

$$y_{ij}\cdot\frac{s_i d_i}{R_{ij}} \le t_j, \forall i\in\mathcal{N}, \forall j\in\mathcal{M}. \tag{7.11d}$$

As a result problem (7.11) can be solved efficiently by a convex programming solver in polynomial time which provides an optimal solution for the compression matrix $\mathbf{s}$ and the maximum transmission latency t_j for all $j \in \mathcal{M}$. With respect to an optimal solution, we provide the following results. For each EC, say j, we define the following two sets of offloaded tasks:

a. $\mathcal{P}_j = \{l \in \{1,\dots N\} : s_l < 1, y_{ij} = 1\}$, which is the set of the tasks with feasible compression ratio and

b. $\mathcal{Q}_j = \{l \in \{1,\dots N\} : s_l = 1, y_{ij} = 1\}$, which is the set of tasks with maximum compression ratio.

Let $t_{\mathcal{P}_j} = \underset{i\in\mathcal{P}_j}{max} \frac{s_i d_i}{R_{ij}}$ and $t_{\mathcal{Q}_j} = \underset{i\in\mathcal{Q}_j}{max} \frac{d_i}{R_{ij}}$ be the maximum transmission times over $\mathcal{P}_j$ and $\mathcal{Q}_j$, respectively. It is evident from Problem (7.10) that an for optimal solution $t_j^\star$, $j \in \mathcal{M}$, of Problem (7.11) it holds that either $t_j^\star = t_{\mathcal{P}_j}$ or $t_j^\star = t_{\mathcal{Q}_j}$. Then we proceed with two results that provide a qualitative description of an optimal solution.

Lemma 2. *Let* $\mathbf{s}^\star = (s_1^\star,\dots,s_N^\star)$ *and* $t_j^\star, j \in \mathcal{M}$*, be an optimal solution to Problem* (7.11)*. Then, for every* $j \in \mathcal{M}$ *and every* $i_1, i_2 \in \mathcal{P}_j$*, it holds*

$$\frac{s_{i_1}^\star d_{i_1}}{R_{i_1 j}} = \frac{s_{i_2}^\star d_{i_2}}{R_{i_2 j}} = t_{\mathcal{P}_j}\,. \tag{7.12}$$

Proof. Fix a $j \in \mathcal{M}$. Suppose, towards arriving at a contradiction, that there exist $i_1, i_2 \in \mathcal{P}_j$ for which it holds

$$\frac{s_{i_1}^\star d_{i_1}}{R_{i_1,j}} > \frac{s_{i_2}^\star d_{i_2}}{R_{i_2,j}}\,. \tag{7.13}$$

Since $s_{i_2}^\star < 1$, we can find $\varepsilon > 0$ such that $\frac{s_{i_1}^\star d_{i_1}}{R_{i_1,j}} > \frac{(s_{i_2}^\star+\varepsilon)d_{i_2}}{R_{i_2,j}}$ and $s_{i_2}^\star + \varepsilon < 1$. Now let $s_{i_2}^+ = s_{i_2}^\star + \varepsilon$, and let $\mathbf{s}^+$ be the vector obtained from $\mathbf{s}$ after replacing the value $s_{i_2}^\star$ with $s_{i_2}^+$, and leaving all remaining values unchanged. Now observe that the objective function in Problem (7.11) decreases; indeed, note that $T(\mathbf{X},\mathbf{s}^+)$ remains the same and, since the Gompertz function is concave, $A(\mathbf{X},\mathbf{s}^+)$

increases. Moreover, the constraints are satisfied. This contradicts the optimality of $\mathbf{s}^\star$ and we conclude that the transmission rates for all tasks $i \in \mathcal{P}_j$ are equal. The result follows. $\square$

We now provide the analogous statement for $\mathcal{Q}_j$.

Lemma 3. *Let* $\mathbf{s}^\star = (s_1^\star, \dots, s_N^\star)$ *and* $t_j^\star, j \in \mathcal{M}$, *be an optimal solution to Problem* (7.11). *Suppose that for some* $j \in \mathcal{M}$ *it holds* $t_{\mathcal{Q}_j} = t_j^\star$. *Then it also holds*

$$t_{\mathcal{Q}_j} = t_{\mathcal{P}_j} = t_j^\star \tag{7.14}$$

Proof. The proof is similar to Lemma (2), so we only sketch it. Suppose, towards arriving at a contradiction, that $t_j^\star = t_{\mathcal{Q}_j} > t_{\mathcal{P}_j}$. Then from Lemma (2) we know that $\frac{s_{i_1}^\star d_{i_1}}{R_{i_1 j}} = \frac{s_{i_2}^\star d_{i_2}}{R_{i_2 j}} = t_{\mathcal{P}_j}, \forall i_1, i_2 \in \mathcal{P}_j$. Now, for every $i \in \mathcal{P}_j$ we can find $\varepsilon_i > 0$ such that, using the same reasoning as in Lemma (2), upon replacing $s_i^\star$ with $s_i^+ = s_i^\star + \varepsilon_i$, the accuracy increases while the latency remains unchanged. This contradicts the assumption that $s_1^\star, \dots s_N^\star$ and $t_j^\star$ are the optimal solutions, and the result follows. $\square$

Let us remark that a similar result is obtained in [209], where it is shown that the task offloading delay is equal for all tasks offloaded at a specific EC. This result is important for the proposed framework as we consider batch processing of the tasks in the ECs. Moreover, as said previously the batch processing starts when the last of the tasks arrives at the EC. Consequently, it is evident that in order to minimize the computing latency at EC j it is enough to minimize the waiting time L_j^{wait}. Hence, the performance of batch processing can be significantly increased by carefully compressing the offloaded tasks in such a way as to arrive simultaneously at each specific ECs.

7.5.2 SDR-based Offloading

Considering the optimal solution $s_1^\star, \dots s_N^\star$ and $t_1^\star \dots t_M^\star$ of the compression problem (7.9) let us formulate the offloading problem derived from Problem (7.8), by substituting the max operator with the respective $t_j^\star$ for $j \in \mathcal{M}$

$$\begin{aligned}
\mathcal{E}3 : \min_{\mathbf{x}} \frac{1}{N} \sum_{i \in \mathcal{N}} x_i \Big(\lambda_T d_i \xi_{k_{loc}} + \lambda_T \zeta_{k_{loc}} - \lambda_A g_i(1, k_{loc}) \Big) & \\
+ \frac{1}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} y_{ij} \Big(\lambda_T t_j^\star + \lambda_T \zeta_{k_j} - \lambda_A g(s_i^\star, k_j) \Big) & \\
+ \frac{\lambda_T}{N} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} \xi_{k_j} y_{ij} \sum_{l \in \mathcal{N}} y_{lj} & \qquad (15) \\
\text{s.t. } (7.8\text{a}), (7.8\text{b}), (7.8\text{d}), (7.8\text{e}), (7.8\text{f}) & \qquad (7.15\text{a})
\end{aligned}$$

Problem $\mathcal{E}3$ is a mixed-integer linear program still challenging to solve. We now proceed with transforming Problem $\mathcal{E}3$ to an equivalent QCQP problem. To begin with, we replace the binary constraints with the following quadratic constraints:

$$\begin{aligned} & x_i(1-x_i)=0,\ \forall i \in \mathcal{N} \\ & y_{ij}(1-y_{ij})=0,\ \forall i \in \mathcal{N}, j \in \mathcal{M}. \end{aligned} \tag{16}$$

We begin with some auxiliary notation. Let us consider the following $1 \times (NM+N)$-matrix: $\mathbf{z} = [x_1, \ldots, x_N, y_{11}, \ldots y_{1M}, \ldots, y_{N1}, \ldots, y_{NM}]^T$, and consider the following $1 \times (NM)$-matrices: $\boldsymbol{\delta} = [t_1^\star + \zeta_{k_1}, \ldots, t_1^\star + \zeta_{k_1}, \ldots, t_M^\star + \zeta_{k_M}, \ldots, t_M^\star + \zeta_{k_M}]^T$ and $\boldsymbol{\Delta} = [\lambda_T(t_1^\star + \zeta_{k_1}) - \lambda_A g(s_1, k_M), \ldots, \lambda_T(t_M^\star + \zeta_{k_M}) - \lambda_A g(s_N, k_M)]^T$. Moreover, we denote $\boldsymbol{\phi} = [\xi_{k_{loc}} d_1 + \zeta_{k_{loc}}, \ldots, \xi_{k_{loc}} d_N + \zeta_{k_{loc}}]^T$, and $\boldsymbol{\Phi} = [\lambda_T(\xi_{k_{loc}} d_1 + \zeta_{k_{loc}}) - \lambda_A g(1, k_{loc}), \ldots, \lambda_T(\xi_{k_{loc}} d_N + \zeta_{k_{loc}}) - \lambda_A g(1, k_{loc})]^T$ which are $1 \times N$-dimension matrices. Then Problem (7.15) can be written as:

$$\mathcal{E}3.2: \min_{\mathbf{z}} \ \mathbf{z}^{\mathbf{T}} \mathbf{A_0} \mathbf{z} + \mathbf{b_0^T} \mathbf{z} \tag{17}$$

$$\text{s.t. } \mathbf{z}^{\mathbf{T}} \mathbf{A_0} \mathbf{z} + \mathbf{c_0^T} \mathbf{z} \le N \cdot T_{max} \tag{7.17a}$$

$$-\mathbf{d_0^T} \mathbf{z} \le -N \cdot A_{min} \tag{7.17b}$$

$$\mathbf{A_1} \mathbf{z} = \mathbf{1}_{N \times 1} \tag{7.17c}$$

$$\mathbf{A_2} \mathbf{z} \le \mathbf{b}_{M \times 1}^{max} \tag{7.17d}$$

$$\mathbf{z}^{\mathbf{T}} \mathit{diag}(\mathbf{u}_p) \mathbf{z} - \mathbf{u}_p^T \mathbf{z} = 0,\ p = 1, \ldots, NM+N, \tag{7.17e}$$

where $\mathbf{A_3} = \left[\frac{\lambda_T}{N} \xi_{k_1} \ldots \frac{\lambda_T}{N} \xi_{k_M} \right]^T$,

$$\mathbf{A_0'} = \begin{bmatrix} \mathit{diag}(\mathbf{A_3})_{M \times M} \ldots \mathit{diag}(\mathbf{A_3})_{M \times M} \\ \vdots \\ \mathit{diag}(\mathbf{A_3})_{M \times M} \ldots \mathit{diag}(\mathbf{A_3})_{M \times M} \end{bmatrix}_{(NM) \times (NM)},$$

$$\mathbf{A_0} = \begin{bmatrix} \mathbf{0}_{N \times N} & \mathbf{0}_{N \times (NM)} \\ \mathbf{0}_{(NM) \times N} & \mathbf{A_0'} \end{bmatrix}, \quad \mathbf{b_0} = [\boldsymbol{\Phi}, \boldsymbol{\Delta}]^T,$$

$$\Gamma_\ell = \begin{bmatrix} \mathbf{0}_{\ell \times M} \\ \mathbf{1}_{1 \times M} \\ \mathbf{0}_{(N-\ell-1) \times M} \end{bmatrix}, \text{ for } \ell = 0, 1, \ldots, N-1,$$

$$\mathbf{A_1} = \begin{bmatrix} \mathbf{I}_{N \times N} & \Gamma_0 & \ldots & \Gamma_{N-1} \end{bmatrix}, \quad \mathbf{c_0} = [\boldsymbol{\phi}, \boldsymbol{\delta}]^T$$

$$\mathbf{A_2} = \begin{bmatrix} \mathbf{0}_{M \times N} & \mathbf{I}_{M \times M} & \ldots & \mathbf{I}_{M \times M} \end{bmatrix}_{M \times (NM+N)},$$

$\mathbf{d_0} = [g(1, k_{loc}), \ldots, g(1, k_{loc}), g(s_1, k_1), \ldots, g(s_N, k_M)]^T$,

and $\mathbf{u}_p$ is a $(NM + N) \times 1$ unit vector with the p_{th} entry being 1 and all others zero. Constraints (7.17a),(7.17b),(7.17c),(7.17d) are equivalent to the constraints (7.8a),(7.8b),(7.8d),(7.8e) respectively, while Constraints (7.17e) are equivalent to the binary constraints (7.16).

Let $\mathbf{W} = \begin{bmatrix} \mathbf{z} \\ 1 \end{bmatrix} \begin{bmatrix} \mathbf{z}^T 1 \end{bmatrix}$. Now we further transform the Problem (7.17) into homogeneous QCQP as follows:

$$\mathcal{E}3.3 : \min_{\mathbf{W}} Tr(\mathbf{B_0 W}) \tag{18}$$

$$\text{s.t. } Tr(\mathbf{B_1 W}) \leq N \cdot T_{max} \tag{7.18a}$$

$$- Tr(\mathbf{B_2 W}) \leq -N \cdot A_{min} \tag{7.18b}$$

$$Tr(\mathbf{H}_h \mathbf{W}) = 1, \;\; h = 1, \ldots, NM + N \tag{7.18c}$$

$$Tr(\mathbf{J}_j \mathbf{W}) \leq b_j^{max}, \;\; j = 1, \ldots, NM + N \tag{7.18d}$$

$$Tr(\mathbf{B_3 W}) = 0, \;\; p = 1, \ldots, NM + N, \tag{7.18e}$$

$$\mathbf{W} \geq 0, \quad rank(\mathbf{W}) = 1, \tag{7.18f}$$

where $Q_2 = (NM + N)$,

$$\mathbf{B_0} = \begin{bmatrix} \mathbf{A_0} & \frac{1}{2}\mathbf{b_0} \\ \frac{1}{2}\mathbf{b_0}^T & 0 \end{bmatrix}, \qquad \mathbf{B_1} = \begin{bmatrix} \mathbf{A_0} & \frac{1}{2}\mathbf{c_0} \\ \frac{1}{2}\mathbf{c_0}^T & 0 \end{bmatrix},$$

$$\mathbf{B_2} = \begin{bmatrix} \mathbf{0}_{Q_2 \times Q_2} & -\frac{1}{2}\mathbf{d_0} \\ -\frac{1}{2}\mathbf{d_0}^T & 0 \end{bmatrix}, \qquad \mathbf{H_h} = \begin{bmatrix} \mathbf{0}_{Q_2 \times Q_2} & \frac{1}{2}\mathbf{A_{1,h}} \\ \frac{1}{2}\mathbf{A_{1,h}}^T & 0 \end{bmatrix},$$

$$\mathbf{J_j} = \begin{bmatrix} \mathbf{0}_{Q_2 \times Q_2} & \frac{1}{2}\mathbf{A_{2,j}} \\ \frac{1}{2}\mathbf{A_{2,j}}^T & 0 \end{bmatrix}, \qquad \mathbf{B_3} = \begin{bmatrix} diag(\mathbf{u}_p) & -\frac{1}{2}\mathbf{u}_p \\ -\frac{1}{2}\mathbf{u}_p^T & 0 \end{bmatrix},$$

and $\mathbf{W} \geq 0$ indicates that the matrix is positive semi-definite (PSD) and $\mathbf{A_{1,h}}, \mathbf{A_{2,j}}$ are the h^{th} and j^{th} row vector of matrix $\mathbf{A_1}$ and $\mathbf{A_2}$ respectively. Following [210] by dropping the $rank(\mathbf{W}) = 1$ constraint which is the only non-convex constraint, Problem $\mathcal{E}3.3$ becomes the SDR of $\mathcal{E}3.2$ and therefore can be solved efficiently using a convex optimization toolbox. The worst case complexity is $\mathcal{O}(\max\{m, n\}^4 n^{\frac{1}{2}} log(\frac{1}{\epsilon}))$ where n is the dimension of the PSD matrix $\mathbf{W}$, m the number of constraints and ϵ the solution accuracy. What is still to be defined is how to obtain a feasible solution.

Let $\mathbf{W}^\star$ be the optimal solution of Problem (7.18) without the rank one constraint. If $rank(\mathbf{W}^\star) = 1$ then we can construct the optimal solution of Problem (7.15) by $\mathbf{W}^\star = \begin{bmatrix} \mathbf{z}^\star \\ 1 \end{bmatrix} \begin{bmatrix} \mathbf{z}^{\star T} 1 \end{bmatrix}$. We call $\mathbf{W}'$

Algorithm 1: SDR-based Offloading Algorithm.

```
 1: Input $N$, $M$, $T_{max}$, $A_{min}$, $\lambda_T$, $\lambda_A$,$\epsilon$, $t_j^\star$, $s_i^\star$, $d_i$, $b_j^{max}$
 2: Solve Problem (7.18) without the rank one constraint to get $\mathbf{W}^\star$.
 3: Extract the upper left $(NM+N) \times (NM+N)$ sub-matrix to get $\mathbf{W}'$.
 4: if $rank(\mathbf{W}^\star) = 1$ then
 5:    $\mathbf{z}^\star$ is the diagonal of $\mathbf{W}'$
 6:    Construct task scheduling matrix $\mathbf{X}^\star$ from $\mathbf{z}^\star$
 7: else
 8:    for $l = 1 \,\text{to}\, L$ do
 9:       Draw vector $\mu^{(l)} \sim \mathbf{N}(\mathbf{0}, \mathbf{W}')$
10:       Set $\mu^{(l)}$ to $\hat{\mathbf{x}}^{(\mathbf{l})}$ by sig function.
11:       Reshape $\hat{\mathbf{x}}^{(l)} \in [0,1]^{MN+N}$ into a candidate task scheduling matrix
          $\hat{\mathbf{X}}^{(l)} \in [0,1]^{N\times(NM+1)}$
12:       for $i = 1 \,\text{to}\, N$ do
13:          $\theta^\star \leftarrow \underset{j\in\mathcal{M}}{argmax} \hat{\mathbf{X}}_{ij}$
14:          $\bar{x}_{ik} \leftarrow 0, \forall k \in \mathcal{M}\ \{\theta^\star\}$ and $\bar{x}_{ik^\star} = 1$
15:       end for
16:    end for
17:    Chose $\mathbf{X}^\star$ by finding the minimum $\Psi(\bar{\mathbf{X}}, \mathbf{s}^\star)$ over L samples.
18: end if
```

the upper-left $(NM+N) \times (NM+N)$ sub-matrix of $\mathbf{W}^\star$. $\mathbf{W}'$ is also PSD [211]. As $z_i = \{0,1\}$, then $z_i \cdot z_i = z_i$. As a result, because $\mathbf{W}' = \mathbf{z}^\star \mathbf{z}^{\star T}$, we can extract the task scheduling matrix $\mathbf{X}^\star$ from $\mathbf{z}^\star$. On the other hand, if $rank(\mathbf{W}^\star) \neq 1$ we need to map the optimal solution of the SDR problem to a nearby feasible solution. For this purpose, we chose to employ a well-known but efficient Gaussian Randomization method [191]. Specifically, we generate L random $(MN+N)\times 1$ vectors $\mu \in \mathcal{R}^{NM+N}$, drawn by a Gaussian distribution with zero mean and covariance the PSD matrix $\mathbf{W}'$, i.e., $\mu^{(l)} \sim \mathbf{N}(\mathbf{0}, \mathbf{W}'), l = 1 \ldots L$. To maintain the binary characteristics (constraint (7.17e)) we map each vector $\mu(l) \in \mathcal{R}^{NM+N}$ into a new vector $\hat{\mathbf{x}}^{(l)} \in [0,1]^{MN+N}$ using the sigmoid function $sig(x) = \frac{1}{1+exp(-\omega x)}$, where $\omega \gg 1$. Then, we reshape $\hat{\mathbf{x}}^{(l)} \in [0,1]^{MN+N}$ into a candidate task scheduling matrix $\hat{\mathbf{X}}^{(l)} \in [0,1]^{N\times NM}$. The largest value for each ED's offloading combination of tasks is set to 1 and all others to 0. By this procedure a matrix $\bar{X}^{(l)}$ is constructed containing the binary constraints for each ED. We repeat the process seeking to find the minimum over L iterations. The best task scheduling matrix is chosen as the solution to the offloading problem $\mathcal{E}3$. In the next Section, we will show that near-optimal solutions can be achieved with a relatively (compared to the problem's state space) small number of L vectors. The randomization mapping has worst-case complexity $\mathcal{O}(LNM)$. We summarize the SDR-based Offloading algorithm in Algorithm 1.

Algorithm 2: SDR-based Offloading with Maximization of Inference Accuracy.

```
1: Input $N$, $M$, $T_{max}$, $A_{min}$, $\lambda_T$, $\lambda_A$, $\epsilon$, $d_i$, $b_j^{max}$
2: $\Psi_{old}(\mathbf{X}, \mathbf{s}) = +\infty$
3: while True do
4:     Solve Problem (7.9) to obtain $\mathbf{s}^\star, \mathbf{t}^\star$
5:     Given $\mathbf{s}^\star, \mathbf{t}^\star$ solve Problem (7.15) to obtain $\Psi(\mathbf{X}, \mathbf{s})$ with Algortihm 1.
6:     if $|\Psi_{old}(\mathbf{X}, \mathbf{s}) - \Psi(\mathbf{X}, \mathbf{s})| \leq \epsilon$ then
7:         break;
8:     else
9:         $\Psi_{old}(\mathbf{X}, \mathbf{s}) = \Psi(\mathbf{X}, \mathbf{s})$
10:    end if
11: end while
```

Table 7.2: Total inference time per batch size, in seconds, in linear and parallel processing with YOLOv5

Batch Size	Total Inference Time	
	Linear CPU	Parallel GPU
1	0.774	0.101
5	3.899	0.377
10	7.615	0.683
20	15.531	1.386
40	33.397	2.707

7.5.3 Overall Algorithm

In this section, we present an overall Algorithm to efficiently solve Problem (7.8). In summary, using an iterative alternating optimization [212], we solve the compression Problem (7.9) to find optimal compression ratios for all tasks and maximum transmission time for the offloading and then utilize these values to solve Problem (7.15) and acquire a near-optimal task scheduling matrix. We repeat the process until the solution $\Psi(\mathbf{X}, \mathbf{s})$ converges and can not be further improved by a small threshold ϵ. A summarized Algorithm is presented in Algorithm 2. The overall complexity of the algorithm is $\mathcal{O}(\max\{m, n\}^4 n^{\frac{1}{2}} log(\frac{1}{\varepsilon}) + LNM)$ where n is the dimension of the PSD matrix $\mathbf{W}$, m the number of constraints, L the number of Gaussian Samples and ε the solution accuracy.

7.6 Numerical Evaluation

In this section, we provide a systematic evaluation of the proposed framework. The parameters we chose are based on [199] and [189]. More specifically, we repeated the evaluation of the latter to extract the approximated linear functions regarding the inference pipeline, i.e., compress the image on the ED, decode and execute inference at the EC. We used the newest version of YOLO as the inference application [201]. The role of ED plays an Intel NUC equipped with Intel Core i5-1135G7 and 8 GiB CPU Intel Core i5-1135G7 RAM 8 GiB and Tiny YOLOv5. The GPU-

Table 7.3: Simulation parameters.

Parameter	**Value**
d_i	$\mathcal{U}(1,2)$ MB
R_{ij}	$\mathcal{U}(5,10)$ MB/s
ϵ	0.001
T_{max}	0.8 s
A_{min}	0.75
$\zeta_{k_{loc}}$	0.01
$\xi_{k_{loc}}$	0.76
ζ_k	0.03
ξ_k	0.06
Gompertz ED	$a = 0.6, b = 5, c = 5$
Gompertz EC	$a = 0.95, b = 5, c = 5$
λ_A	1
λ_T	1
L	300
b^{max}	$\mathcal{N}(\frac{N}{4}), \frac{N}{3})$

enabled server is a Lenovo IdeaCente5 equipped with an Nvidia GeForce GTX 1660 SUPER with 6 GiB memory GPU and 16 GiB RAM. We selected to deploy a YOLOv5[1] tiny model to the ED and x-large model on the ED. For the compression and decoding processes, we select the libjxl [203] library. Based on 200 images of 640 by 640 pixels from the Dota dataset [213], we measured the average accuracy of the inference using Yolov5, and we selected the Gompertz parameters as follows (a) for the ED k_{loc} model (tiny): $a = 0.6, b = 5, c = 5$ and for the GPU-enabled EC k model (x-large) $a = 0.95, b = 5, c = 5$. Moreover, using the thunder effort (a libjxl set of parameters for compression), with a quality factor of 100, we measured that the encoding speed is 22.83 MP/s or an average of 26 ms for the selected images. Then we measured the inference pipeline (except the transmission) in case of offloading a batch to the GPU-enabled ED, and the results are presented in Table 7.2. In the case of local execution, we consider that tasks are executed sequentially, and no encoding-decoding is required. Based on these results, we extract the approximated linear functions $f_k(b_j) = \xi_k b_j + \zeta_k$ to have an estimation of the average latency for a batch of offloaded inference tasks. For the local inference we compute that $\zeta_{k_{loc}} = 0.01$, $\xi_{k_{loc}} = 0.76$, while $\zeta_k = 0.03$, $\xi_k = 0.06$ for the remote inference.

Regarding the rest parameters, the precision of the solution is $\epsilon = 0.001$ while the threshold for latency and accuracy are $T_{max} = 0.8\ s$, $A_{min} = 0.75$. The weights of the objective function are $\lambda_A = \lambda_T = 1$ the number of Gaussian samples is $L = 300$ and task size and the data rates are drawn from a uniform distribution, i.e., $d_i \sim \mathcal{U}(1,2)$ MB and $R_{ij} \sim \mathcal{U}(5,10)$ MB/sec, respectively. The maximum batch size for EC j is drawn by a Gaussian distribution, i.e., $b_j^{max} \sim \mathcal{N}(\frac{N}{4}), \frac{N}{3})$. Table 7.3 presents a complete list of simulation parameters and represents all the experiments

[1]https://github.com/ultralytics/yolov5

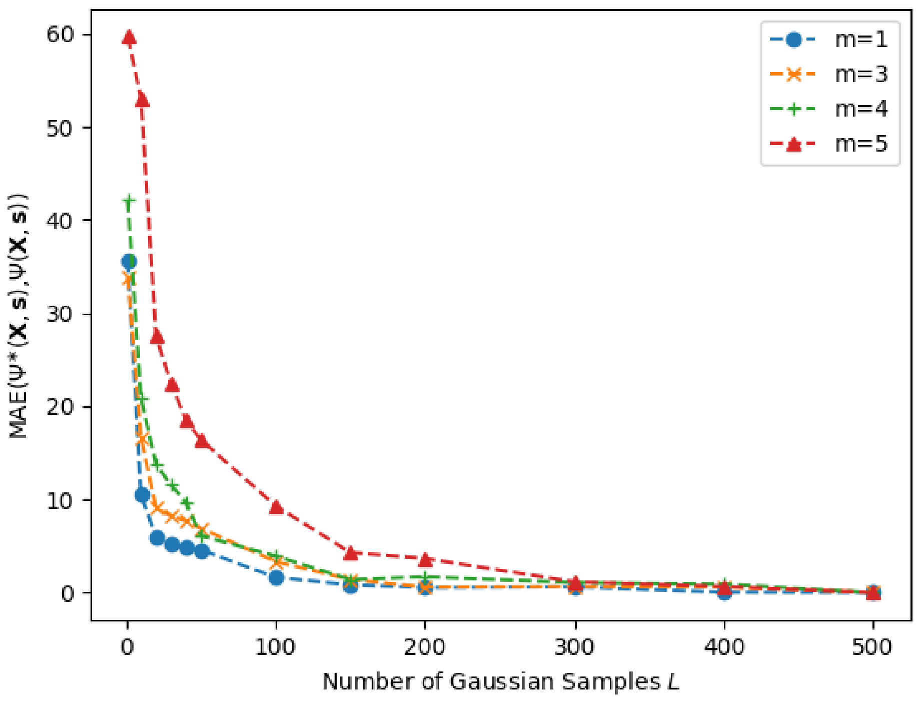


Figure 7.2: Convergence of SDR solution to optimal, for different ECs over the Number of Gaussian samples L, when N=10.

unless stated explicitly otherwise. The simulation code for the experiments is available here [2]. For all experiments to get unbiased results we generated the same seeds for the simulation parameters and averaged the results over 50 experiments.

7.6.1 Convergence of SDR-based Offloading to the Optimal Solution

First, we study the convergence of the SDR-based offloading algorithm to the optimal solution. We set the number of tasks $n = 10$ and use the default values for the rest parameters as shown in Table 7.3. Fig. 7.2 shows the Mean Average Error (MAE) between the SDR-based offloading and the optimal solution to the number of Gaussian Samples L. The optimal solution is acquired with brute force in each experiment with exponential running time. One can observe that the higher the number of samples, the better the solution provided by the SDR algorithm. Also, as the number of servers increases, the solution convergence speed declines, due to the fact that the number of total variables and constraints to the SDR problem rapidly increases. We chose $L = 300$ for the rest of the experiments, as to achieve a marginal difference in the MAE, a much larger L is required. In particular, to obtain an MAE score of 1.15, L needs to increase from 50 to 300 for $m = 5$, while to achieve an MAE of 0.01, L needs to increase from 300 to 500.

[2]https://github.com/Dspatharakis/approximate

7.6.2 Algorithm Comparison

Next, we study the performance of the proposed SDR-based Offloading with Optimized Compression solution (SDR-OC) compared to other baseline methods. Namely; (a) Local Processing (Local): all tasks executed on the EDs, (b) Remote Processing Random Compression (RemoteRC): all requests executed on the ECs with random compression, (c) Remote Processing Optimized Compression (RemoteOC): all requests executed on the ECs with optimized compression, (d) Random Processing Random Compression (RPRC): random offloading of tasks and random compression, (e) Random Processing Optimized Compression (RPOC): random selection for the tasks and optimized compression, (f) SDR-based Random Compression (SDR-RC): SDR-based offloading and random compression and (g) Convex-Relaxation Optimized Compression (ConvexOC). Specifically, the ConvexOC method refers to Problem (7.15) if we relax the binary constraint, i.e., $x_i, y_{ij} \in [0,1], \forall i \in \mathcal{N}, j \in \mathcal{M}$. With this relaxation Problem (7.15) becomes Convex and it is possible to find a solution with the interior point method with a complexity of $\mathcal{O}(v^{3.5}K^2)$, where v denotes the number of variables and K the number of the bits in the input [214]. However, the extracted solution of ConvexOC is a solution vector $\mathbf{x}^{\text{ConvexOC}}$, which is not feasible for the initial Problem. As a result, in this case, we repeat a similar randomization process as in the SDR-OC, to extract a feasible solution. Specifically, for the $\mathbf{x}^{\text{ConvexOC}}$ vector we execute steps [9-14] of Algorithm 1. For the methods Local, RemoteRC, RemoteOC, RPRC, and RPOC, in order to even obtain solutions it is necessary to let the thresholds $T_{max} = 0.9$ and $A_{min} = 0.6$ and $b_j^{max} = N, \forall j \in \mathcal{M}$. For the rest parameters and for the methods SDR-OC, SDR-RC, and ConvexOC, we select the values as in Table 7.3.

For this experiment, we set the number of servers $m = 5$. In Fig. 7.3 we compare the total cost $\Psi(\mathbf{X}, \mathbf{s})$ over the number of tasks N, for the same random seeds for all methods. We should mention that the lower the cost the better the solution. As expected, for all methods the total cost increases with the number of requests for fixed ECs. The proposed method (SDR-OC) outperforms all others achieving a significant reduction in the total cost. As the number of tasks increases the margin of reduction is increased. For example, for $N = 70$, the ratio of total cost between SDR-OC and ConvexOC is $\frac{\Psi^{\text{ConvexOC}}(\mathbf{X},\mathbf{s})}{\Psi^{\text{SDR-OC}}(\mathbf{X},\mathbf{s})} = \frac{0.192}{0.091} \approx 2.11$ and between SDR-OC and SDR-RC $\frac{\Psi^{\text{SDR-RC}}(\mathbf{X},\mathbf{s})}{\Psi^{\text{SDR-OC}}(\mathbf{X},\mathbf{s})} = \frac{0.339}{0.091} \approx 3.72$. Moreover, all methods with optimized compression provide better results than those with random compression, e.g., see RemoteRC when compared to RemoteOC or RPRC and RPOC. As a result, the optimization of the compression of tasks remarkably boosts the overall performance. We should also mention that although $\lambda_T = \lambda_A = 1$, the total cost seems to be more dependent on the offloading decision than the compression, e.g., see RPOC which provides in all cases, worse results than SDR-RC.

Another interesting remark is that for $N = 40$, we note the ratio of total cost between the RemoteOC method and our approach (SDR-OC) is $\frac{\Psi^{\text{RemoteOC}}(\mathbf{X},\mathbf{s})}{\Psi^{\text{SDR-OC}}(\mathbf{X},\mathbf{s})} = \frac{0.192}{0.091} \approx 2.01$. Let us consider that for $m = 5$, the 40 tasks are equally distributed among the ECs. Hence, the average computational for the RemoteOC method is approximately $L_{ij}^{\text{comp}} = f_k(8) \equiv 0.5(s)$. In addition, if we consider the mean values from the Uniform distribution for the data rates and the task size and set

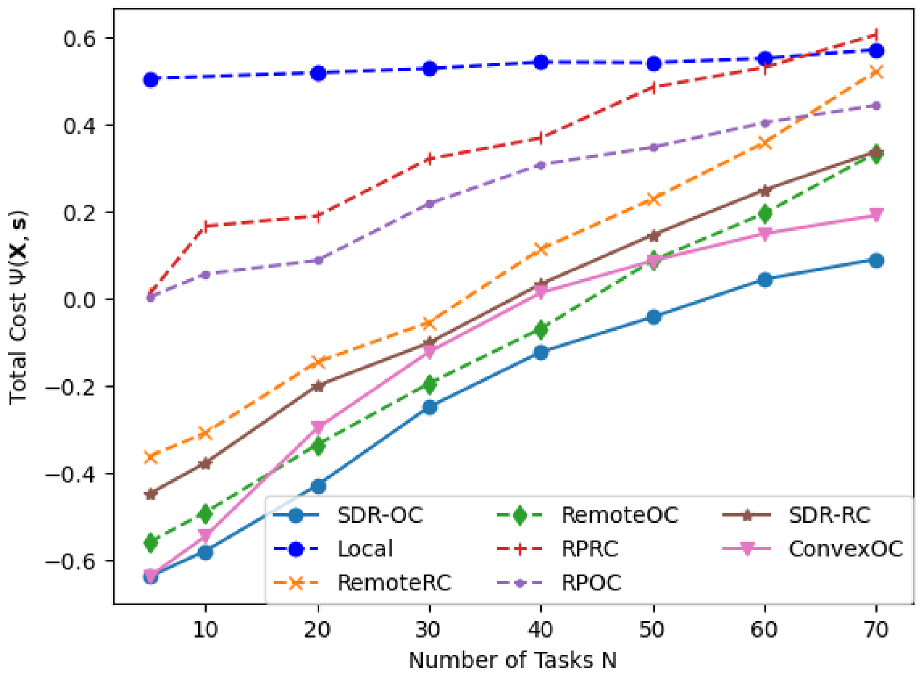


Figure 7.3: Comparison of the proposed method to other baseline methods.

compression equal to 1 for all tasks, then the average transmission time $L_{ij}^{\text{tran}} = \frac{1.5\text{MB}}{7.5\frac{\text{MB}}{\text{s}}} \approx 0.2$. As a result, the average latency for one task i offloaded to EC j is $L_i \equiv 0.7$, which is comparable with the local execution which is $L_{loc} \equiv 0.76$ as $\zeta_{k_{loc}} = 0.01$ and $\xi_{k_{loc}} = 0.76$. This implies, that for such a number of requests in the particular scenario, the remote execution of all tasks is optimized. As expected, the RemoteOC method performance declines as the number of tasks increases because the batch size execution is not as beneficial as executing some of the tasks locally. Finally, for $N = 70$, the RPRC method is worse even than Local, whose performance is almost static throughout the experiments. This, clearly shows that for non-optimized offloading and compression, the total performance deteriorates as the total system is stressed.

7.6.3 Maximum Batch Size effect on Total Cost

In this experiment, we study the effect of the maximum batch size on ECs on the performance of the solution. We select three different Gaussian Distributions to draw the b^{max} for each EC, namely, (a) small batch: $b^{max} \sim \mathcal{N}(\frac{N}{5}, \frac{N}{4})$, (b) medium batch: $b^{max} \sim \mathcal{N}(\frac{N}{4}, \frac{N}{3})$, (c) large batch: $b^{max} \sim \mathcal{N}(\frac{N}{3}, \frac{N}{2})$. We set the number of ECs $m = 3$ and repeat the experiment for $N = \{10, 20, 30, 40\}$. We should note that for $N = 40$, the ECs are considered stressed. In Fig 7.4, the Average Latency, Average Accuracy, and Percentage of Offloaded Tasks are depicted for each setting. For the small batch, the percentage of offloaded tasks is approximately 60% for all values of N. For the medium batch, the percentage of offloaded tasks is $76\%, 81\%, 75\%$ and $\%65$ for

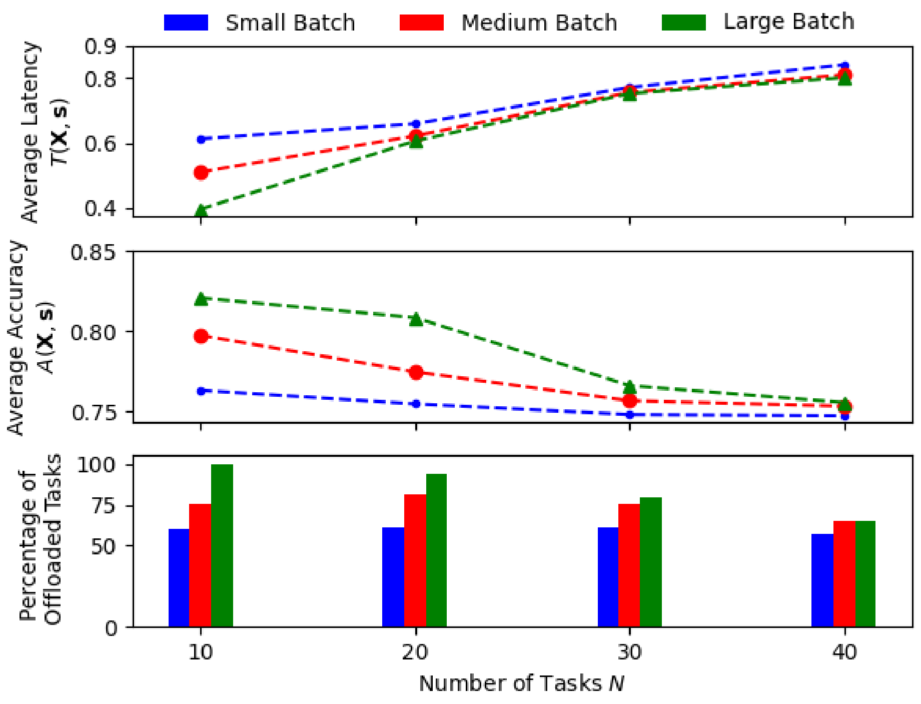


Figure 7.4: Average Latency, Accuracy, and Percentage of Offloaded Tasks for different maximum batch sizes for the ECs.

$N = \{10, 20, 30, 40\}$ respectively. Correspondingly for the large batch, the percentages are larger for small values of N, i.e., $100\%, 93\%$ for $N = 10,\ N = 20$, respectively however, drop to the same rate of the medium batch, as the number of tasks grows, i.e., 78% and 65% for $N = 30,\ N = 40$. Specifically, having as a baseline the small batch, we remark that the average latency is for $N = 10$, 17%, and 36% less for the medium and large batch, respectively, and $N = 20$ is 5% 7.5% and 5% 3%. As the number of tasks increases, $N \geq 30$, the average latency is almost equal for all cases and close to the threshold of T_{max}. In parallel, the results for the average accuracy follow the same trend, as for $N = 10$ and $N = 20$, we observe an increase of $5\%, 8\%$ and $3\%, 7\%$ in accuracy for the medium and the large batch, respectively. However, again when $N \geq 30$, the differences are minimized as previously. As a result, we conclude that the average latency and average accuracy are strongly connected to the percentage of offloaded tasks and can significantly benefit from larger values of the maximum batch sizes until the ECs are stressed, where we observe a similar performance for all settings.

7.6.4 Impact of the Weights of the Objective Function

As the total cost is a weighted sum of the average latency and average accuracy as defined in (**??**)(7.7) in this experiment, we study the effects of λ_T and λ_A parameters. We select three different cases for the ratio, i.e., (a) latency-sensitive: $\frac{\lambda_A}{\lambda_T} = 0.5$, (b) balanced: $\frac{\lambda_A}{\lambda_T} = 0.5$ and

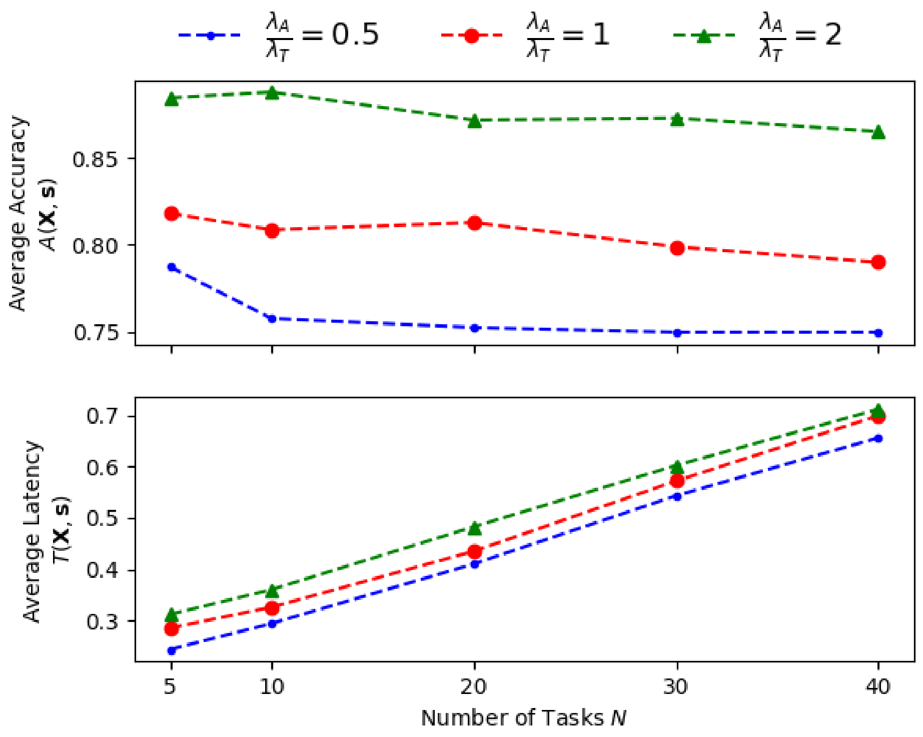


Figure 7.5: Average Latency and Average Accuracy over N for different Number of $\frac{\lambda_A}{\lambda_T}$ ratio.

(c) accuracy-sensitive: $\frac{\lambda_A}{\lambda_T} = 0.5$, that imply different application's requirements. In Fig. 7.5, the average accuracy and latency for $N = \{10, 20, 30, 40\}$ and $m = 3$ are presented. As expected, the average accuracy slightly drops and the average latency linearly increases to the number of tasks for all cases. Moreover, as one can notice, for the latency-sensitive application, the average latency is minimized compared to other cases, resulting though a declining average accuracy that equals the selected threshold $A_{min} = 0.75$ for $N > 5$. For the balanced case $A(\mathbf{X}, (x)) \approx 0.81$ keeping the latency at a slight increase of 107% on average than the latency-sensitive application. For the accuracy-sensitive application, we can see that $A(\mathbf{X}, (x)) \approx 0.87$ on average for the different cases of N which is 1.07% and 116% better than balanced and latency-sensitive respectively. However, the average latency increases on average 113%, compared to the latency-sensitive application. As a result, we can achieve different performance requirements by tweaking the values of λ_T and λ_A.

7.6.5 Effect of Data Rates

For the final experiment, we study the effect of different uniform distributions of the data rate R_{ij} on the total cost. We set $m = 3$, and all settings are in the best-effort mode, meaning that the thresholds T_{max}, A_{min} are not hard constraints in the problem. This is necessary to obtain solutions for all settings. In Fig. 7.6, one can notice that the system cost is minimized as the data rates increase, which is expected as the overhead of the transmission time is insignificant

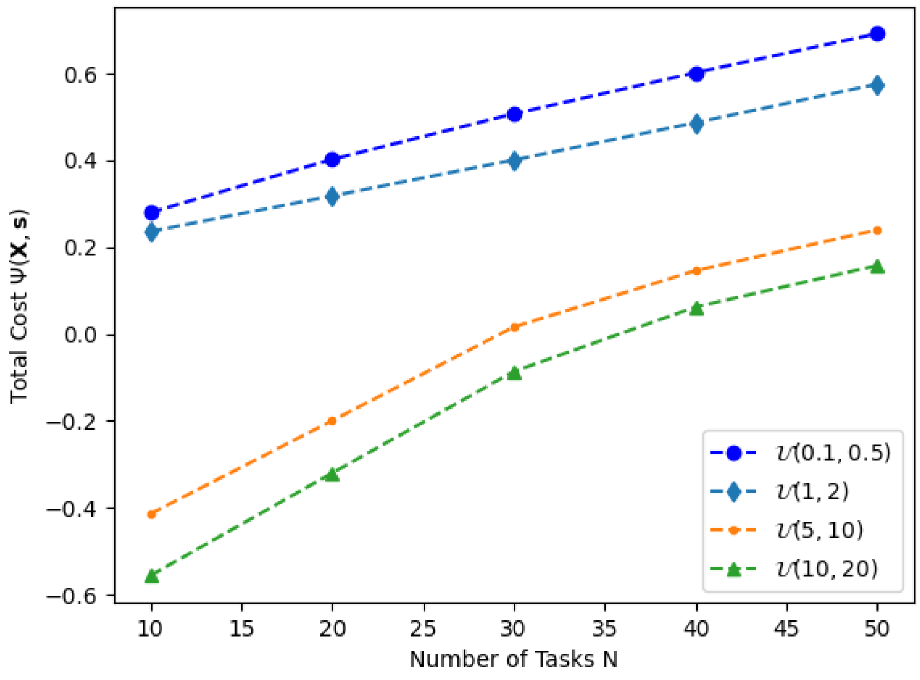


Figure 7.6: Total Cost for different distributions of data rate R_{ij}.

to the total latency. Specifically for data rates drawn by the following uniform distribution, i.e., $R = \mathcal{U}(0.1, 0.5)$, the total cost significantly increases, and the desired thresholds, i.e., $T_{max} = 0.8$ and $A_{min} = 0.75$, are constantly violated. Moreover, most of the tasks are executed locally, i.e., above 55%, as the transmission time heavily affects the total latency. We notice similar resutls for $R = \mathcal{U}(1, 2)$. The percentage of offloaded tasks for lower values of N is relatively higher, i.e., 73% for $N = 10$, however, drops to the same level as the previous case for $N > 30$. However, for this case, the thresholds for T_{max}, A_{min} are satisfied for $N < 30$. On the other hand, for faster data rates $R = \mathcal{U}(5, 10)$ and $R = \mathcal{U}(10, 20)$, the total cost is minimized, and the percentage of offloaded requests is above 90% for both cases for $N = [10, 20]$. However, as the number of tasks increases, we observe a decrease in the percentage of offloaded tasks, i.e., on average 70% when $N > 30$. Nonetheless, in both cases, the provided solution never exceeds the limits for T_{max}, A_{min}. As a result, for a large number of tasks, it is necessary to have faster data rates to keep the percentage of offloaded tasks high while keeping the overall performance of the system in a desired state.

7.7 Summary

In this chapter, we studied the problem of batch scheduling in GPU-enabled ECs with maximizing inference accuracy in a real-time Edge-assisted inference scenario. We modeled a joint optimization problem with a weighted sum function of the average latency and average accuracy of the task inference. We first optimized the inference accuracy given a scheduling decision and provided significant results for the transmission time of offloaded requests in the examined scenario. Based on the compression problem solution, we employed the SDR to obtain a solution in polynomial time. A randomization process mapped the solution to a near-optimal solution to the initial problem. Then, we solved both problems iteratively until they converged into a solution to the joint optimization problem. Results indicate that the proposed technique outperforms all other benchmark methods by having a close-to-optimal solution and achieving minimum total cost.

Chapter 8

Conclusions and Future Work

In this chapter, the main outcomes of this thesis are discussed in section 8.1. Next, our work-in-progress is summarized and interesting directions for further research in the future are identified in section 8.2.

8.1 Conclusions

The immense advancement of modern applications along with the stringent requirements for ever-expanding resources requires new sophisticated approaches in the journey of providing high QoS and QoE to users. Besides, many contradictive and diverse KPIs exist, and infrastructure providers and application developers must work together to achieve the desired overall optimal performance of applications and utilization of resources. Although many new architectural concepts have arisen in the past few years, there are still many challenges to be addressed to provide seamless operations and execution of the complex algorithms a CPS may have. The two major challenges are the task offloading strategies and the accompanied resource management mechanisms. Consequently, in this thesis, multiple aspects for solving these two problems have been studied, which will allow the network edge to become a reliable pool of resources for IoT devices in their quest for executing computing-intensive applications and transforming the world as we know it. Key problems have been identified, novel algorithms have been developed and their performance has been evaluated. Toward dealing with the challenges of the research topics introduced in the introductory chapter, we employed Control Theory approaches to achieve stability and convergence guarantees for the use cases addressed in the thesis. Moreover, modeling from Queuing Theory and several optimization problems are studied in the quest of finding a balance between the accuracy of applications and mission duration for CPS. On the other hand, to maximize the performance of the underlying edge infrastructures, holistic frameworks were designed to tackle the challenges that arise from real-world problems and to try to be synchronized with state-of-the-art standardized architectures. Thorough experimentation was conducted in real-world infrastructure and/or simulation to provide solid results regarding the performance benefits when adopting the proposed solutions. The conclusions to which we arrived throughout the realization of this thesis so far are the following:

- **Control-Theoretic Approaches can provide powerful tools to support resource management and offloading in CPS.** Due to the need for real-time decisions in task scheduling and optimal offloading problems control-theoretic approaches are appropriate. By providing a precise understanding of the behavior of the system, these approaches offer numerous benefits in terms of resource allocation and offloading. Through mathematical modeling and analysis of system dynamics, these approaches can identify areas of inefficiency in resource utilization and design control algorithms to regulate the behavior of the system. These algorithms can dynamically adjust resource allocation based on the system's current state and performance objectives, leading to improved efficiency, stability, and robustness. The control-theoretic approach also provides a framework for balancing the trade-off between resource utilization and system performance. Overall, in this thesis, we conclude that the utilization of control-theoretic methods in CPS can lead to significant advancements in resource allocation and offloading optimization.

- **Switching Offloading Strategies can be beneficial for both application performance and resource utilization.** The utilization of switching offloading strategies in cyber-physical systems (CPS) has the potential to greatly improve the performance and resource utilization of these systems. By dynamically distributing processing workloads between the edge and the cloud, the system can effectively leverage the strengths of both resources, leading to optimized resource utilization and application performance. For example, in certain scenarios, offloading processing tasks to the cloud can provide the necessary computational power and reduce the load on edge devices, leading to improved performance and resource utilization. Conversely, in other scenarios, keeping processing tasks at the edge can reduce network congestion and minimize latency, also leading to improved performance and resource utilization. All in all, switching offloading strategies offer flexible and adaptive approaches to resource utilization in CPS. By dynamically adjusting the distribution of processing workloads, these strategies can help ensure that the system is utilizing its resources in the most efficient and effective manner, leading to improved performance and resource utilization. Finally, Approximate Computing is also a powerful paradigm to embrace when dealing with task offloading in resource-constrained devices and a really interesting research topic as the problem of decreasing the quality to gain in time or utilization of resources, is fundamentally an extremely interesting trade-off.

- **Resource scheduling and dynamic resource allocation are key challenges that need to be jointly addressed.** Maximizing the performance of modern applications requires timely resource management of the virtualized resources. However, proactively deploying resources for meeting specific application requirements subject to a dynamic workload of incoming requests is extremely challenging. For example, if resource scheduling is not properly implemented, it can lead to bottlenecks in processing, causing delays and decreased system performance. Similarly, if dynamic resource allocation is not effectively managed, it can result in suboptimal resource utilization, leading to inefficiencies and reduced system performance. To address these challenges, a comprehensive and integrated approach is required, where resource scheduling and dynamic allocation are jointly considered and optimized. This approach can lead to improved resource utilization, reduced system latency, and increased system stability and robustness. To this end, in this thesis, the fundamental problems of task scheduling and resource autoscaling were jointly addressed and evaluated resulting in better performance when compared to other solutions

- **Application Profiling is another powerful tool to facilitate fine-grained resource management solutions but is a very complex task.** Appropriately mapping the resources to a dynamic workload and incorporating the dynamics and various performance criteria was a key challenge in this thesis. By leveraging the Application Profiling Mechanisms, the resource allocation and resource scheduling problems can benefit a lot by providing simpler, however, stable solutions. Hence, effective and efficient algorithms of low complexity are required, as the training phase of the application profiling may need much time or plenty of experiments. and while taking into account the constraints of the state and input variables. Exactly this is what we tried to do in all our works in this thesis.

- **The integration of holistic architectures with real-world frameworks rise new problems and challenges.** Finally, many of the research challenges encountered in this thesis were discovered when trying to integrate the proposed mechanisms with state-of-the-art frameworks such as Kubernetes and OpenStack. Trying to build holistic architectures compatible with these frameworks and aligned with the current standards and research trends of the Academia and the state of practice of the Industry was the driving force throughout this thesis.

8.2 Future Work

Concluding this thesis, this final section elucidates some of the possible future research directions that can be followed based on the outcomes of the presented work and the challenges faced during the research process. While the thesis treats resource allocation and computational offloading problems that concern some of the most dynamic procedures in the context of CPS, there still exists much room for further development and innovation.

One interesting direction is the examination of various KPIs from the infrastructure side for the problem of dynamic resource allocation. Hence, the optimization problem that decides the number of deployed resources introduced in Chapter 3, could be enhanced to concentrate on power optimization by minimizing the number of active servers. As mentioned previously, edge datacenters have a specific capacity of resources, making the problem of horizontal scaling of the cluster (i.e., adding a new server to the cluster) very exciting. For example, trying to offload as many tasks of a specific application as possible, while keeping the number of active servers low, results in maximum energy efficiency both for the devices and the edge servers. Moreover, optimizing energy cost in the device layer namely optimizing the energy consumption of devices either by offloading or executing tasks locally is also an interesting and challenging problem.

Another interesting work is to include non-deterministic/stochastic approaches that could be evaluated for estimation and control purposes and machine learning techniques could be integrated into the mobility prediction approaches to enable addressing errors in the predictions of dynamically estimated values of the position and number of the involved devices to tackle mobility aspects. It would be interesting to extend these novel methodologies for the position estimation, to further improve its accuracy, while including pattern recognition of users' mobility in the overall designed architecture. This can be achieved by exploiting Machine Learning techniques and combining them with more detailed modeling of the signal interference among the users, in order to ultimately offer a more effective and efficient offloading decision.

Regarding the resource allocation mechanism, future work could focus on further investigating improvements in the modeling and control of the application-specific VMs and leveraging different combinatorial optimization criteria to improve the resource management framework, introduced in Chapters 3 and 4. Furthermore, for time-sensitive applications, e.g., augmented or virtual reality ones, the resource profiles can be used to design feedback controllers that enable scale-up/down operations on the deployed resources and achieve fine-grained performance regulation.

A combination of the above is also a really challenging topic for future ideas and innovation. As a result, further investigating the Co-design aspect of resource allocation and offloading is crucial for optimizing overall performance. One possible research direction could focus on modifying and extending the capabilities of state-of-the-art resource orchestrator tools (e.g., Kubernetes) in order to develop fine-grained dynamic resource scaling mechanisms. Leveraging the work in Application Profiling to support the creation or modification of instances of containerized applications, and then we will develop customized resource-allocating mechanisms that support real-time resource scaling and container migration.

Another very interesting research direction is to focus on further examining offloading capabilities in the context of 3C, as studied in Chapter 6 especially the integration of planning algorithms and the adaptation of the offloading decisions based on safety guarantees for the robot's navigation. We also plan to investigate more sophisticated techniques to manipulate the unicycle dynamics and integrate them into the proposed set-based solution. Moreover, one possible research direction could cover the case of multiple robots moving in a common environment and using shared resources, thus requiring us to adapt our framework to interacting agents in both the resource utilization problem and the trajectory tracking and path planning problem. Finally, deep learning techniques will be investigated, for calculating and dynamically adapting the utility function coefficients based on different application criteria.

In another scenario, in the context of approximate computing for real-time data processing as studied in Chapter 7, we could adopt more sophisticated communication modeling (e.g., Non-Orthogonal Multiple Access), which could be approached as another subproblem or again employ SDR to obtain a solution. Also, to make a decentralized technique for real-time Edge-assisted inference, it would be very challenging to provide a model of a Reinforcement Learning agent for each ED, whose reward will depend on the total cost provided by our method.

Appendix A

Author's Publications

International Peer Reviewed Journals

- **Spatharakis, D.**, Dimolitsas, I., Dechouniotis, D., Papathanail, G., Fotoglou, I., Papadimitriou, P., & Papavassiliou, S. (2020). A scalable edge computing architecture enabling smart offloading for location based services. Pervasive and Mobile Computing, 67, 101217.
- Avgeris, M., **Spatharakis, D.**, Dechouniotis, D., Leivadeas, A., Karyotis, V., & Papavassiliou, S. (2022). ENERDGE: Distributed energy-aware resource allocation at the edge. Sensors, 22(2), 660.
- **Spatharakis D.**, M. Avgeris, N. Athanasopoulos, D. Dechouniotis and S. Papavassiliou, "Resource-Aware Estimation and Control for Edge Robotics: A Set-Based Approach," in IEEE Internet of Things Journal, vol. 10, no. 3, pp. 2003-2020, 1 Feb.1, 2023, doi: 10.1109/JIOT.2022.3141266.
- Saeik, F., Avgeris, M., **Spatharakis, D.**, Santi, N., Dechouniotis, D., Violos, J., Leivadeas, A., Athanasopoulos, N., Mitton, N. and Papavassiliou, S., 2021. Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions. Computer Networks, 195, p.108177.
- Papathanail, G., Pentelas, A., Fotoglou, I., Papadimitriou, P., Katsaros, K.V., Theodorou, V., Soursos, S., **Spatharakis, D.**, Dimolitsas, I., Avgeris, M. and Dechouniotis, D., 2020. MESON: Optimized Cross-Slice Communication for Edge Computing. IEEE Communications Magazine, 58(10), pp.23-28.
- Avgeris, M., **Spatharakis, D.**, Dechouniotis, D., Kalatzis, N., Roussaki, I. and Papavassiliou, S., 2019. Where there is fire there is smoke: a scalable edge computing framework for early fire detection. Sensors, 19(3), p.639.

International Conferences

- Dimolitsas, I., **Spatharakis, D.**, Dechouniotis, D., Zafeiropoulos, A., & Papavassiliou, S. (2023, June). Multi-Application Hierarchical Autoscaling for Kubernetes Edge Clusters. In 2023 IEEE International Conference on Smart Computing (SMARTCOMP) (pp. 291-296). IEEE.
- Saeik, F., Violos, J., Leivadeas, A., Avgeris, M., **Spatharakis, D.**, and Dechouniotis D., 2021, September. User association and behavioral characterization during task offloading at

the edge. In 2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom) (IEEE MeditCom2021). IEEE.

- **Spatharakis, D.**, Avgeris, M., Kakkavas G., Papadakis-Vlachopapadopoulos k., Dimolitsas I., Dechouniotis D., Karyotis V., and Papavassiliou S., 2021, July. Industrial robotics experimentation over federated next generation internet testbeds. In 2021 IEEE International Mediterranean Conference on communications and Networking (MeditCom): Demo Sessions (IEEE MeditCom 2021 - Demos). IEEE.
- **Spatharakis, D.**, Avgeris, M., Athanasopoulos, N., Dechouniotis, D. and Papavassiliou, S., 2020, November. A Switching Offloading Mechanism for Path Planning and Localization in Robotic Applications. In 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics) (pp. 77-84). IEEE.
- Dimolitsas, I., **Spatharakis, D.**, Dechouniotis, D., & Papavassiliou, S. (2022, December). AHP4HPA: An AHP-based Autoscaling Framework for Kubernetes Clusters at the Network Edge. In GLOBECOM 2022-2022 IEEE Global Communications Conference (pp. 2566-2571). IEEE. **(best paper award)**
- **Spatharakis, D.**, Dimolitsas, I., Vlahakis, E., Dechouniotis, D., Athanasopoulos, N., & Papavassiliou, S. (2022, October). Distributed Resource Autoscaling in Kubernetes Edge Clusters. In 2022 18th International Conference on Network and Service Management (CNSM) (pp. 163-169). IEEE.
- Dimolitsas I., Avgeris, M., **Spatharakis, D.**, Dechouniotis D., and Papavassiliou S., 2021, September. Enabling industrial network slicing orchestration: A collaborative edge robotics use case. In 2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom) (IEEE MeditCom2021). IEEE.
- Avgeris, M., **Spatharakis, D.**, Athanasopoulos, N., Dechouniotis, D. and Papavassiliou, S., 2019, August. Single Vision-Based Self-Localization for Autonomous Robotic Agents. In 2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW) (pp. 123-129). IEEE.
- Papathanail, G., Fotoglou, I., Demertzis, C., Pentelas, A., Sgouromitis, K., Papadimitriou, P., D. Spatharakis, I. Dimolitsas, D. Dechouniotis, and S. Papavassiliou, (2020, April). COSMOS: An orchestration framework for smart computation offloading in edge clouds. In NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium (pp. 1-6). IEEE.
- Dechouniotis, D., **Spatharakis, D.**, & Papavassiliou, S. (2022, April). Edge robotics experimentation over next generation IIoT testbeds. In NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium (pp. 1-3). IEEE.
- Dimolitsas, I., **Spatharakis, D.**, Dechouniotis, D., & Papavassiliou, S. (2022, September). A Delay-Aware Approach for Distributed Embedding Towards Cross-Slice Communication. In 2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom) (pp. 13-18). IEEE.
- Brochu, O., **Spatharakis, D.**, Dechouniotis, D., Leivadeas, A., & Papavassiliou, S. (2022, September). Benchmarking real-time image processing for offloading at the edge. In 2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom) (pp. 90-93). IEEE.

Bibliography

[1] Farzad Samie et al. "Computation offloading and resource allocation for low-power IoT edge devices." In: *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE. 2016, pp. 7–12.

[2] Mahadev Satyanarayanan et al. "The Role of Edge Offload for Hardware - Accelerated Mobile Devices." In: *GetMobile: Mobile Comp. and Comm.* 25.2 (Sept. 2021), pp. 5–13. ISSN: 2375-0529. DOI: 10.1145/3486880.3486882. URL: https://doi.org/10.1145/3486880.3486882.

[3] Firdose Saeik, Marios Avgeris, Dimitrios Spatharakis, et al. "Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions." In: *Computer Networks* (2021), p. 108177.

[4] U Cisco. "Cisco annual internet report (2018–2023) white paper." In: *Cisco: San Jose, CA, USA* (2020).

[5] Cisco. *Internet of things at a glance.* https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf. (accessed on 28 January 2020). 2017.

[6] Ericsson. *Ericsson mobility report - Q3 2019.* https://www.ericsson.com/4acd7e/assets/local/mobility-report/documents/2019/emr-november-2019.pdf. (accessed on 28 January 2020). 2019.

[7] Carlos Bernardos et al. *European Vision for the 6G Network Ecosystem.* June 2021. DOI: 10.13140/RG.2.2.19993.95849.

[8] Michael Armbrust et al. *Above the clouds: A berkeley view of cloud computing.* Tech. rep. Technical Report UCB/EECS-2009-28, EECS Department, University of California …, 2009.

[9] Peter Mell, Tim Grance, et al. "The NIST definition of cloud computing." In: (2011).

[10] Hoang T Dinh et al. "A survey of mobile cloud computing: architecture, applications, and approaches." In: *Wireless communications and mobile computing* 13.18 (2013), pp. 1587–1611.

[11] Abderrahime Filali et al. "Multi-access edge computing: A survey." In: *IEEE Access* 8 (2020), pp. 197017–197046.

[12] Ben Zhang et al. "The cloud is not enough: Saving iot from the cloud." In: *7th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 15).* 2015.

[13] Athar Ali Khan, Mubashir Husain Rehmani, and Abderrezak Rachedi. "When cognitive radio meets the Internet of Things?" In: *2016 international wireless communications and mobile computing conference (IWCMC).* IEEE. 2016, pp. 469–474.

[14] P Hédé et al. “Mobile-edge Computing Introductory Technical White Paper.” In: *Technical Report*. Mobile-edge Computing (MEC) industry initiative, 2014.

[15] Yun Chao Hu et al. “Mobile edge computing—A key technology towards 5G.” In: *ETSI white paper* 11.11 (2015), pp. 1–16.

[16] Ashkan Yousefpour et al. “All one needs to know about fog computing and related edge computing paradigms: A complete survey.” In: *Journal of Systems Architecture* 98 (2019), pp. 289–330.

[17] Minoru Uehara. “Mist computing: Linking cloudlet to fogs.” In: *International Conference on Computational Science/Intelligence & Applied Informatics*. Springer. 2017, pp. 201–213.

[18] Tarik Taleb et al. “On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration.” In: *IEEE Communications Surveys & Tutorials* 19.3 (2017), pp. 1657–1681.

[19] Yotam Harchol et al. “Making Edge-Computing Resilient.” MA thesis. EECS Department, University of California, Berkeley, Apr. 2019. URL: `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-9.html`.

[20] Alex Reznik et al. “Cloud RAN and MEC: A perfect pairing.” In: *ETSI White paper* 23 (2018), pp. 1–24.

[21] Joao F Santos et al. “NETWORK SOFTWARIZATION AND MANAGEMENT.” In: *IEEE Communications Magazine* (2020), p. 2.

[22] J.G. Herrera and J.F. Botero. “Resource allocation in NFV: A comprehensive survey.” In: *IEEE Transactions on Network and Service Management* 13.3 (2016), pp. 518–532. DOI: `10.1109/TNSM.2016.2598420`.

[23] N. Feamster, J. Rexford, and E. Zegura. “The road to SDN: an intellectual history of programmable networks.” In: *ACM SIGCOMM Computer Communication Review* 44.2 (2014), pp. 87–98. DOI: `10.1145/2602204.2602219`.

[24] Ioannis Dimolitsas et al. “Enabling industrial network slicing orchestration: A collaborative edge robotics use case.” In: *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE. 2021, pp. 215–220.

[25] Paul Pop et al. “Enabling fog computing for industrial automation through time-sensitive networking (TSN).” In: *IEEE Communications Standards Magazine* 2.2 (2018), pp. 55–61.

[26] Guanqing Liang et al. “Smart world: a better world.” In: *Science China Information Sciences* 59.4 (2016), pp. 1–5.

[27] Tarik Taleb, Ibrahim Afolabi, and Miloud Bagaa. “Orchestrating 5G network slices to support industrial internet and to shape next-generation smart factories.” In: *IEEE Network* 33.4 (2019), pp. 146–154.

[28] Guoqiang Hu, Wee Peng Tay, and Yonggang Wen. “Cloud robotics: architecture, challenges and applications.” In: *IEEE network* 26.3 (2012), pp. 21–28.

[29] Dezhen Song, Ajay Kumar Tanwani, and Ken Goldberg. “Networked-, Cloud- and Fog-Robotics.” In: 2019.

[30] Ioannis Dimolitsas et al. “A Delay-Aware Approach for Distributed Embedding Towards Cross-Slice Communication.” In: *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE. 2022, pp. 13–18.

[31] Andrea Zanella et al. “Internet of things for smart cities.” In: *IEEE Internet of Things journal* 1.1 (2014), pp. 22–32.

[32] Latif U. Khan et al. “Edge-Computing-Enabled Smart Cities: A Comprehensive Survey.” In: *IEEE Internet of Things Journal* 7.10 (2020), pp. 10200–10232. DOI: 10.1109/JIOT.2020.2987070.

[33] Marios Avgeris et al. “Where there is fire there is smoke: A scalable edge computing framework for early fire detection.” In: *Sensors* 19.3 (2019), p. 639.

[34] Zhaolong Ning et al. “Intelligent Edge Computing in Internet of Vehicles: A Joint Computation Offloading and Caching Solution.” In: *IEEE Transactions on Intelligent Transportation Systems* 22.4 (2021), pp. 2212–2225. DOI: 10.1109/TITS.2020.2997832.

[35] George Papathanail et al. “COSMOS: An orchestration framework for smart computation offloading in edge clouds.” In: *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE. 2020, pp. 1–6.

[36] Jie Bao et al. “Recommendations in location-based social networks: a survey.” In: *GeoInformatica* 19.3 (2015), pp. 525–565.

[37] Latif U Khan et al. “Edge-computing-enabled smart cities: A comprehensive survey.” In: *IEEE Internet of Things Journal* 7.10 (2020), pp. 10200–10232.

[38] Shuiguang Deng et al. “Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence.” In: *IEEE Internet of Things Journal* 7.8 (2020), pp. 7457–7469. DOI: 10.1109/JIOT.2020.2984887.

[39] Kun Cao et al. “A survey on edge and edge-cloud computing assisted cyber-physical systems.” In: *IEEE Transactions on Industrial Informatics* 17.11 (2021), pp. 7806–7819.

[40] Sanjit A Seshia et al. “Design automation of cyber-physical systems: Challenges, advances, and opportunities.” In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.9 (2016), pp. 1421–1434.

[41] Sparsh Mittal. “A survey of techniques for approximate computing.” In: *ACM Computing Surveys (CSUR)* 48.4 (2016), pp. 1–33.

[42] Elif Uysal, Onur Kaya, Anthony Ephremides, et al. “Semantic Communications in Networked Systems.” In: *arXiv preprint arXiv:2103.05391* (2021).

[43] Tarek Abdelzaher et al. “Five challenges in cloud-enabled intelligence and control.” In: *ACM Transactions on Internet Technology (TOIT)* 20.1 (2020), pp. 1–19.

[44] AR Arunarani, Dhanabalachandran Manjula, and Vijayan Sugumaran. “Task scheduling techniques in cloud computing: A literature survey.” In: *Future Generation Computer Systems* 91 (2019), pp. 407–415.

[45] Irfan Mohiuddin and Ahmad Almogren. “Workload aware VM consolidation method in edge/cloud computing for IoT applications.” In: *Journal of Parallel and Distributed Computing* 123 (2019), pp. 204–214.

[46] Dirk Merkel et al. “Docker: lightweight linux containers for consistent development and deployment.” In: *Linux j* 239.2 (2014), p. 2.

[47] Ioannis Dimolitsas et al. “Multi-Application Hierarchical Autoscaling for Kubernetes Edge Clusters.” In: *2023 IEEE International Conference on Smart Computing (SMARTCOMP)*. 2023, pp. 291–296. DOI: 10.1109/SMARTCOMP58114.2023.00074.

[48] Quyuan Luo et al. “Resource scheduling in edge computing: A survey.” In: *IEEE Communications Surveys & Tutorials* 23.4 (2021), pp. 2131–2165.

[49] Ejaz Ahmed et al. "Application optimization in mobile cloud computing: Motivation, taxonomies, and open challenges." In: *Journal of Network and Computer Applications* 52 (2015), pp. 52–68.

[50] Shichao Li et al. "Joint admission control and resource allocation in edge computing for internet of things." In: *IEEE Network* 32.1 (2018), pp. 72–79.

[51] Essam H Houssein et al. "Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends." In: *Swarm and Evolutionary Computation* 62 (2021), p. 100841.

[52] Dimitrios Dechouniotis et al. "Edge computing resource allocation for dynamic networks: The DRUID-NET vision and perspective." In: *Sensors* 20.8 (2020), p. 2191.

[53] Marios Avgeris et al. "ENERDGE: Distributed energy-aware resource allocation at the edge." In: *Sensors* 22.2 (2022), p. 660.

[54] Anupama Mampage, Shanika Karunasekera, and Rajkumar Buyya. "A holistic view on resource management in serverless computing environments: Taxonomy and future directions." In: *ACM Computing Surveys (CSUR)* 54.11s (2022), pp. 1–36.

[55] Dimitrios Spatharakis et al. "Industrial Robotics Experimentation over Federated Next Generation Internet Testbeds." In: *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE. 2021, pp. 1–2.

[56] Ajay Kumar Tanwani et al. "RILaaS: Robot inference and learning as a service." In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4423–4430.

[57] Mahbuba Afrin et al. "Resource allocation and service provisioning in multi-agent cloud robotics: A comprehensive survey." In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 842–870.

[58] Fabio Giust, Xavier Costa-Perez, and Alex Reznik. "Multi-access edge computing: An overview of ETSI MEC ISG." In: *IEEE 5G Tech Focus* 1.4 (2017), p. 4.

[59] Dimitrios Spatharakis et al. "A Lightweight Software Stack for IoT Interoperability within the Computing Continuum." In: *2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*. 2023, pp. 715–722. DOI: `10.1109/DCOSS-IoT58021.2023.00112`.

[60] Ioannis Dimolitsas et al. "AHP4HPA: An AHP-based Autoscaling Framework for Kubernetes Clusters at the Network Edge." In: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*. 2022, pp. 2566–2571. DOI: `10.1109/GLOBECOM48099.2022.10001214`.

[61] George Papathanail et al. "Meson: Optimized cross-slice communication for edge computing." In: *IEEE Communications Magazine* 58.10 (2020), pp. 23–28.

[62] OpenStack. https://www.openstack.org/. (Last Accessed on 07/01/2023).

[63] Kubernetes. https://kubernetes.io/. (Last Accessed on 07/01/2023).

[64] CT Chen. "Linear system theory and design: Oxford University Press." In: *New York, USA* (1999).

[65] N Athanasopoulos. "Stability analysis and control of linear and nonlinear constrained systems via Polyhedral Lyapunov functions." In: *Unpublished Ph. D. dissertation). University of Patras, Greece* (2010).

[66] Leonard Kleinrock. *Theory, volume 1, Queueing systems*. 1975.

[67] Christos G Cassandras and Stéphane Lafortune. *Introduction to discrete event systems*. Springer, 2008.

[68] C. Puliafito et al. “Fog computing for the internet of things: A Survey.” In: *ACM Transactions on Internet Technology (TOIT)* 19.2 (2019), pp. 1–41. DOI: 10.1145/3301443.

[69] I. Afolabi et al. “Dynamic resource provisioning of a scalable E2E network slicing orchestration system.” In: *IEEE Transactions on Mobile Computing* (2019). DOI: 10.1109/TMC.2019.2930059.

[70] ETSI. *Open Source NFV Management and Orchestration (MANO)*. https://osm.etsi.org/. 2020. (Last Accessed on 07/01/2023).

[71] K. Katsaros et al. “MESON: Facilitating cross-slice communications for enhanced service delivery at the edge.” In: *2019 European Conference on Networks and Communications (EuCNC) Poster Session*. IEEE. 2019.

[72] A. Leivadeas et al. “VNF placement optimization at the edge and cloud.” In: *Future Internet* 11.3 (2019), p. 69. DOI: 10.3390/fi11030069.

[73] C. Sonmez, A. Ozgovde, and C Ersoy. “Fuzzy workload orchestration for edge computing.” In: *IEEE Transactions on Network and Service Management* 16.2 (2019), pp. 769–782. DOI: 10.1109/TNSM.2019.2901346.

[74] B. Gao et al. “Winning at the starting line: Joint network selection and service placement for mobile edge computing.” In: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE. 2019, pp. 1459–1467. DOI: 10.1109/INFOCOM.2019.8737543.

[75] J. Xu et al. “Zenith: Utility-aware resource allocation for edge computing.” In: *2017 IEEE International Conference on Edge Computing (EDGE)*. IEEE. 2017, pp. 47–54.

[76] Wei Wang et al. “Virtual machine placement and workload assignment for mobile edge computing.” In: *2017 IEEE 6th International Conference on Cloud Networking (CloudNet)*. IEEE. 2017, pp. 1–6.

[77] Miloud Bagaa et al. “Coalitional game for the creation of efficient virtual core network slices in 5G mobile systems.” In: *IEEE Journal on Selected Areas in Communications* 36.3 (2018), pp. 469–484.

[78] K. Kumar et al. “A survey of computation offloading for mobile systems.” In: *Mobile Networks and Applications* 18.1 (2013), pp. 129–140. DOI: 10.1007/s11036-012-0368-0.

[79] Y. Shi, S. Chen, and X. Xu. “MAGA: A mobility-aware computation offloading decision for distributed mobile cloud computing.” In: *IEEE Internet of Things Journal* 5.1 (2017), pp. 164–174. DOI: 10.1109/JIOT.2017.2776252.

[80] L. Tang and S. He. “Multi-user computation offloading in mobile edge computing: A behavioral perspective.” In: *IEEE Network* 32.1 (2018), pp. 48–53. DOI: 10.1109/MNET.2018.1700119.

[81] P.A. Apostolopoulos, E.E. Tsiropoulou, and S. Papavassiliou. “Risk-aware Social Cloud Computing based on Serverless Computing Model.” In: *IEEE Global Communications Conference (GLOBECOM), 2019*. 2019.

[82] M. Chen and Y. Hao. “Task offloading for mobile edge computing in software defined ultra-dense network.” In: *IEEE Journal on Selected Areas in Communications* 36.3 (2018), pp. 587–597. DOI: 10.1109/JSAC.2018.2815360.

[83] X. Lyu et al. “Selective offloading in mobile edge computing for the green internet of things.” In: *IEEE Network* 32.1 (2018), pp. 54–60. DOI: 10.1109/MNET.2018.1700101.

[84] H. Yang et al. “WKNN indoor location algorithm based on zone partition by spatial features and restriction of former location.” In: *Pervasive and Mobile Computing* 60 (2019), p. 101085. DOI: 10.1016/j.pmcj.2019.101085.

[85] W. Kang and Y. Han. "SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization." In: *IEEE Sensors journal* 15.5 (2014), pp. 2906–2916. DOI: 10.1109/JSEN.2014.2382568.

[86] A. Correa et al. "Indoor pedestrian tracking by on-body multiple receivers." In: *IEEE Sensors Journal* 16.8 (2016), pp. 2545–2553. DOI: 10.1109/JSEN.2016.2518872.

[87] M. Avgeris et al. "Single Vision-Based Self-Localization for Autonomous Robotic Agents." In: *7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. 2019, pp. 123–129. DOI: 10.1109/FiCloudW.2019.00035.

[88] T. Zhou et al. "Unsupervised learning of depth and ego-motion from video." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1851–1858.

[89] R. Calheiros et al. "Workload prediction using ARIMA model and its impact on cloud applications' QoS." In: *IEEE Transactions on Cloud Computing* 3.4 (2014), pp. 449–458.

[90] M. Jia et al. "Cloudlet load balancing in wireless metropolitan area networks." In: *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE. 2016, pp. 1–9.

[91] N.H. Ho, P.H. Truong, and G.M. Jeong. "Step-detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone." In: *Sensors* 16.9 (2016), p. 1423. DOI: 10.3390/s16091423.

[92] L. Ljung. *System Identification: Theory for the User*. Prentice-hall, Inc., 1987. DOI: 10.1007/978-1-4612-1768-8_11.

[93] D. Ardagna et al. "Energy-aware autonomic resource allocation in multitier virtualized environments." In: *IEEE transactions on services computing* 5.1 (2010), pp. 2–19.

[94] R.E. Kalman. "A new approach to linear filtering and prediction problems." In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45. DOI: 10.1115/1.3662552.

[95] 5GinFIRE. *University of Bristol 5g Testbed*. https://5ginfire.eu/university-of-bristol-5g-testbed/. 2019. (Last Accessed on 07/01/2023).

[96] Google. *TensorFlow Platform*. https://www.tensorflow.org/. 2020. (Last Accessed on 07/01/2023).

[97] D. Dechouniotis et al. "A control-theoretic approach towards joint admission control and resource allocation of cloud computing services." In: *International Journal of Network Management* 25.3 (2015), pp. 159–180.

[98] Dimitrios Spatharakis et al. "A scalable edge computing architecture enabling smart offloading for location based services." In: *Pervasive and Mobile Computing* 67 (2020), p. 101217.

[99] Horizontal-Pod-Autoscaling. https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/. (Last Accessed on 07/01/2023).

[100] Cheol-Ho Hong and Blesson Varghese. "Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms." In: *ACM Computing Surveys (CSUR)* 52.5 (2019), pp. 1–37.

[101] Mostafa Ghobaei-Arani, Alireza Souri, and Ali A Rahmanian. "Resource management approaches in fog computing: a comprehensive review." In: *Journal of Grid Computing* 18.1 (2020), pp. 1–42.

[102] Shveta Verma and Anju Bala. "Auto-scaling techniques for IoT-based cloud applications: a review." In: *Cluster Computing* 24.3 (2021), pp. 2425–2459.

[103] Dah-Ming Chiu and Raj Jain. "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks." In: *Computer Networks and ISDN systems* 17.1 (1989), pp. 1–14.

[104] Eleftherios Vlahakis, Nikolaos Athanasopoulos, and Seán McLoone. "AIMD scheduling and resource allocation in distributed computing systems." In: *2021 60th IEEE Conference on Decision and Control (CDC).* 2021, pp. 4642–4647. DOI: `10.1109/CDC45484.2021.9683379`.

[105] Luciano Baresi et al. "A discrete-time feedback controller for containerized cloud applications." In: *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering.* 2016, pp. 217–228.

[106] Ali Shahidinejad, Mostafa Ghobaei-Arani, and Mohammad Masdari. "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach." In: *Cluster Computing* 24.1 (2021), pp. 319–342.

[107] Waheed Iqbal et al. "Predictive auto-scaling of multi-tier applications using performance varying cloud resources." In: *IEEE Transactions on Cloud Computing* (2019).

[108] László Toka et al. "Machine learning-based scaling management for kubernetes edge clusters." In: *IEEE Transactions on Network and Service Management* 18.1 (2021), pp. 958–972.

[109] Fabiana Rossi, Matteo Nardelli, and Valeria Cardellini. "Horizontal and vertical scaling of container-based applications using reinforcement learning." In: *2019 IEEE 12th International Conference on Cloud Computing (CLOUD).* IEEE. 2019, pp. 329–338.

[110] Prometheus. https://prometheus.io/. (Last Accessed on 07/01/2023).

[111] Custom-Pod-Autoscaler. https://custom-pod-autoscaler.readthedocs.io. (Last Accessed on 07/01/2023).

[112] Wei Ren et al. "Optimal Resource Scheduling and Allocation in Distributed Computing Systems." In: *arXiv preprint arXiv:2112.00708* (2021).

[113] Martin Corless et al. *AIMD dynamics and distributed resource allocation.* SIAM, 2016.

[114] R. Calheiros et al. "Workload prediction using ARIMA model and its impact on cloud applications' QoS." In: *IEEE Transactions on Cloud Computing* 3.4 (2014), pp. 449–458.

[115] Tin Kam Ho. "Random decision forests." In: *Proceedings of 3rd international conference on document analysis and recognition.* Vol. 1. IEEE. 1995, pp. 278–282.

[116] David R Cox. "The regression analysis of binary sequences." In: *Journal of the Royal Statistical Society: Series B (Methodological)* 20.2 (1958), pp. 215–232.

[117] Kashvi Taunk et al. "A Brief Review of Nearest Neighbor Algorithm for Learning and Classification." In: *2019 International Conference on Intelligent Computing and Control Systems (ICCS).* 2019, pp. 1255–1260. DOI: `10.1109/ICCS45141.2019.9065747`.

[118] Corinna Cortes and Vladimir Vapnik. "Support-vector networks." In: *Machine learning* 20.3 (1995), pp. 273–297.

[119] Payam Refaeilzadeh, Lei Tang, and Huan Liu. "Cross-validation." In: *Encyclopedia of database systems* 5 (2009), pp. 532–538.

[120] Dimitrios Spatharakis et al. "Distributed Resource Autoscaling in Kubernetes Edge Clusters." In: *2022 18th International Conference on Network and Service Management (CNSM).* IEEE. 2022, pp. 163–169.

[121] Dimitrios Dechouniotis, Dimitrios Spatharakis, and Symeon Papavassiliou. "Edge Robotics Experimentation over Next Generation IIoT Testbeds." In: *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium.* IEEE. 2022, pp. 1–3.

[122] P. Mach and Z. Becvar. "Mobile Edge Computing: A Survey on Architecture and Computation Offloading." In: *in IEEE Communications Surveys & Tutorials* 19.3 (2017), pp. 1628–1656.

[123] M. Avgeris et al. "Where there is fire there is SMOKE: a scalable edge computing framework for early fire detection." In: *Sensors* 19 (2019), p. 3.

[124] M. Aazam, S. Zeadally, and K. A. Harras. "Deploying Fog Computing in Industrial Internet of Things and Industry 4.0." In: *IEEE Trans. Ind. Inf* 14.10 (Oct. 2018), pp. 4674–4682.

[125] A. V. Dastjerdi et al. "Fog Computing: principles, architectures, and applications." In: *Internet of Things.* 2016, pp. 61–75.

[126] P. Pop et al. "Enabling Fog Computing for Industrial Automation Through Time-Sensitive Networking (TSN)." In: *IEEE Comm* 2.2 (June 2018), pp. 55–61.

[127] D. Song et al. *Networked-, cloud-and fog-robotics.* Springer, 2019.

[128] N. Tian al. "A Cloud-Based Robust Semaphore Mirroring System for Social Robots." In: *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE).* Munich, Aug. 2018, pp. 1351–1358.

[129] A. K. Tanwani et al. "A Fog Robotics Approach to Deep Robot Learning: Application to Object Recognition and Grasp Planning in Surface Decluttering." In: *2019 International Conference on Robotics and Automation (ICRA).* QC, Canada: Montreal, May 2019, pp. 4559–4566.

[130] A. Botta, L. Gallo, and G. Ventre. "Cloud, Fog, and Dew Robotics: Architectures for Next Generation Applications." In: *2019 7th IEEE International Conference on Mobile Cloud Computing.* and Engineering (MobileCloud), Newark, CA, USA: Services, Apr. 2019, pp. 16–23.

[131] L. A. Trinh et al. "Position rectification with depth camera to improve odometry-based localization." In: *2015 International Conference on Communications.* DaNang, Vietnam: Management and Telecommunications (ComManTel), Dec. 2015, pp. 147–152.

[132] S. M. LaValle and Planning Algorithms. "Cambridge: Cambridge University Press." In: 2006 ().

[133] C. G. Cassandras and S. Lafortune. *Introduction to discrete event systems, Second edition.* New York, NY: Springer, 2010.

[134] D. Pizarro et al. "Localization of mobile robots using odometry and an external vision sensor." In: *Sensors* 10.4 (2010), pp. 3655–3680.

[135] M. Avgeris et al. "Single Vision-Based Self-Localization for Autonomous Robotic Agents." In: *2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW).* Ed. by Turkey Istanbul. Aug. 2019, pp. 123–129.

[136] D. C. K. Yuen and B. A. MacDonald. "Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison." In: *IEEE Trans. Robot* 21.2 (Apr. 2005), pp. 217–226.

[137] A. Bais and R. Sablatnig. "Landmark Based Global Self-localization of Mobile Soccer Robots." In: *Computer Vision – ACCV 2006.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 842–851.

[138] K. Daniel et al. “Theta*: Any-Angle Path Planning on Grids.” In: *jair* 39 (Oct. 2010), pp. 533–579.

[139] C. Chen, M. Rickert, and A. Knoll. “Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering.” In: *in* 2015 (June 2015), pp. 1148–1153.

[140] P. E. Hart, N. Nilsson, and B. Raphael. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths.” In: *IEEE Trans. Syst. Sci* 4.2 (1968), pp. 100–107.

[141] J. Peng, Y. Huang, and G. Luo. “Robot Path Planning Based on Improved A* Algorithm.” In: *Cybernetics and Information Technologies* 15.2 (June 2015), pp. 171–180.

[142] F. Duchoň et al. “Path planning with modified a star algorithm for a mobile robot.” In: *Procedia Engineering* 96 (2014), pp. 59–69.

[143] D. D. Harabor and A. Grastien. “(2011.” In: *Online graph pruning for pathfinding on grid maps.* in Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August: August), 2011, pp. 7–11.

[144] R. Kalman. “A new approach to linear filtering and prediction problems.” In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45.

[145] T. Fraichard and A. Scheuer. “From Reeds and Shepp’s to Continuous-Curvature Paths.” In: *IEEE Trans. Robot* 20.6 (Dec. 2004), pp. 1025–1035.

[146] Dimitrios Spatharakis et al. “A Switching Offloading Mechanism for Path Planning and Localization in Robotic Applications.” In: *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics).* IEEE. 2020, pp. 77–84.

[147] Ben Kehoe et al. “A survey of research on cloud robotics and automation.” In: *IEEE Transactions on automation science and engineering* 12.2 (2015), pp. 398–409.

[148] Sylvia L. Herbert et al. “FaSTrack: A modular framework for fast and guaranteed safe motion planning.” In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC).* 2017, pp. 1517–1522. DOI: 10.1109/CDC.2017.8263867.

[149] Reza Olfati-Saber. “Near-identity diffeomorphisms and exponential/spl epsi/-tracking and/spl epsi/-stabilization of first-order nonholonomic se (2) vehicles.” In: *Proceedings of the 2002 american control conference (ieee cat. no. ch37301).* Vol. 6. IEEE. 2002, pp. 4690–4695.

[150] Ji-Chul Ryu and Sunil K Agrawal. “Differential flatness-based robust control of mobile robots in the presence of slip.” In: *The International Journal of Robotics Research* 30.4 (2011), pp. 463–475.

[151] Andrea Bajcsy et al. “An efficient reachability-based framework for provably safe autonomous navigation in unknown environments.” In: *2019 IEEE 58th Conference on Decision and Control (CDC).* IEEE. 2019, pp. 1758–1765.

[152] Chang Liu et al. “Path planning for autonomous vehicles using model predictive control.” In: *2017 IEEE Intelligent Vehicles Symposium (IV).* IEEE. 2017, pp. 174–179.

[153] Kristina Miller, Chuchu Fan, and Sayan Mitra. “Planning in dynamic and partially unknown environments.” In: *In Proceedings of 7th IFAC Conference on Analysis and Design of Hybrid Systems (ADHS’21).* 2021.

[154] Liang Zhang et al. “Optimized motion strategy for active target localization of mobile robots with time-varying connectivity.” In: *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS).* IEEE. 2019, pp. 185–187.

[155] Jingao Xu et al. “Edge assisted mobile semantic visual slam.” In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE. 2020, pp. 1828–1837.

[156] Sandeep Chinchali et al. “Network offloading policies for cloud robotics: a learning-based approach.” In: *Autonomous Robots* (2021), pp. 1–16.

[157] Manabu Nakanoya et al. “Task-relevant Representation Learning for Networked Robotic Perception.” In: *arXiv preprint arXiv:2011.03216* (2020).

[158] Asad Waqar Malik et al. “Symbiotic robotics network for efficient task offloading in smart industry.” In: *IEEE Transactions on Industrial Informatics* (2020).

[159] Marios Avgeris et al. “Single Vision-Based Self-Localization for Autonomous Robotic Agents.” In: *2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. 2019, pp. 123–129. DOI: `10.1109/FiCloudW.2019.00035`.

[160] Daniel Pizarro et al. “Localization of mobile robots using odometry and an external vision sensor.” In: *Sensors* 10.4 (2010), pp. 3655–3680.

[161] Michael Lemmon. “Event-triggered feedback in control, estimation, and optimization.” In: *Networked control systems* (2010), pp. 293–358.

[162] Shuai Liu, Daniel E Quevedo, and Lihua Xie. “Event-triggered distributed constrained consensus.” In: *International Journal of Robust and Nonlinear Control* 27.16 (2017), pp. 3043–3060.

[163] Clint Ferrin, Greg Droge, and Randall Christensen. “Zero-error tracking for autonomous vehicles through epsilon-trajectory generation.” In: *IEEE Control Systems Letters* 5.6 (2020), pp. 2084–2089.

[164] Franco Blanchini and Stefano Miani. *Set-Theoretic Methods in Control*. 1st. Birkhäuser Basel, 2007. ISBN: 0817632557.

[165] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[166] Mordechai Ben-Ari and Francesco Mondada. “Robotic Motion and Odometry.” In: *Elements of Robotics*. Cham: Springer International Publishing, 2018, pp. 63–93. ISBN: 978-3-319-62533-1. DOI: `10.1007/978-3-319-62533-1_5`. URL: `https://doi.org/10.1007/978-3-319-62533-1_5`.

[167] Xin Chen, Erika Abraham, and Sriram Sankaranarayanan. “Taylor model flowpipe construction for non-linear hybrid systems.” In: *2012 IEEE 33rd Real-Time Systems Symposium*. IEEE. 2012, pp. 183–192.

[168] Xin Chen. “Reachability analysis of non-linear hybrid systems using Taylor models.” PhD thesis. RWTH Aachen University, 2015.

[169] Thao Dang. “Approximate reachability computation for polynomial systems.” In: *International Workshop on Hybrid Systems: Computation and Control*. Springer. 2006, pp. 138–152.

[170] Tommaso Dreossi, Thao Dang, and Carla Piazza. “Parallelotope bundles for polynomial reachability.” In: *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*. 2016, pp. 297–306.

[171] Lester E Dubins. “On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents.” In: *American Journal of mathematics* 79.3 (1957), pp. 497–516.

[172] Lucas Liebenwein et al. "Sampling-Based Approximation Algorithms for Reachability Analysis with Provable Guarantees." In: *Proceedings of Robotics: Science and Systems*. Pittsburgh, Pennsylvania, June 2018. DOI: 10.15607/RSS.2018.XIV.014.

[173] Abdul Bais and Robert Sablatnig. "Landmark based global self-localization of mobile soccer robots." In: *Asian Conference on Computer Vision*. Springer. 2006, pp. 842–851.

[174] David CK Yuen and Bruce A MacDonald. "Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison." In: *IEEE Transactions on robotics* 21.2 (2005), pp. 217–226.

[175] Goran Vasiljević et al. "High-accuracy vehicle localization for autonomous warehousing." In: *Robotics and Computer-Integrated Manufacturing* 42 (2016), pp. 1–16.

[176] Weihua Chen and Tie Zhang. "An indoor mobile robot navigation technique using odometry and electronic compass." In: *International Journal of Advanced Robotic Systems* 14.3 (2017), p. 1729881417711643.

[177] Daniel B Faria et al. "Modeling signal attenuation in ieee 802.11 wireless lans-vol. 1." In: *Computer Science Department, Stanford University* 1 (2005).

[178] Rudolph Emil Kalman. "A new approach to linear filtering and prediction problems." In: (1960).

[179] K. Zhang et al. "Optimal delay constrained offloading for vehicular edge computing networks." In: *2017 IEEE International Conference on Communications (ICC)*. 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7997360.

[180] Xin Chen, Erika Ábrahám, and Sriram Sankaranarayanan. "Flow*: An Analyzer for Nonlinear Hybrid Systems." In: *Computer Aided Verification*. Ed. by Natasha Sharygina and Helmut Veith. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 258–263. ISBN: 978-3-642-39799-8.

[181] Xin Chen and Sriram Sankaranarayanan. "Decomposed reachability analysis for nonlinear systems." In: *2016 IEEE Real-Time Systems Symposium (RTSS)*. IEEE. 2016, pp. 13–24.

[182] Matthias Althoff, Dmitry Grebenyuk, and Niklas Kochdumper. "Implementation of Taylor models in CORA 2018." In: *Proc. of the 5th International Workshop on Applied Verification for Continuous and Hybrid Systems*. 2018.

[183] Dalton Cézane Gomes Valadares et al. "802.11 g signal strength evaluation in an industrial environment." In: *Internet of Things* 9 (2020), p. 100163.

[184] Dimitrios Spatharakis et al. "Resource-aware estimation and control for edge robotics: A set-based approach." In: *IEEE Internet of Things Journal* (2022).

[185] Weiqiang Liu, Fabrizio Lombardi, and Michael Shulte. "A retrospective and prospective view of approximate computing [point of view." In: *Proceedings of the IEEE* 108.3 (2020), pp. 394–399.

[186] A. Cristiano I. Malossi et al. "The transprecision computing paradigm: Concept, design, and applications." In: *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 2018, pp. 1105–1110. DOI: 10.23919/DATE.2018.8342176.

[187] Muhammad Asif Khan et al. "A survey on mobile edge computing for video streaming: Opportunities and challenges." In: *IEEE Access* (2022).

[188] Liang Dong et al. "WAVE: Edge-Device Cooperated Real-time Object Detection for Open-air Applications." In: *IEEE Transactions on Mobile Computing* (2022).

[189] Olivier Brochu et al. “Benchmarking real-time image processing for offloading at the edge.” In: *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE. 2022, pp. 90–93.

[190] Shaojun Zhang et al. “DyBatch: Efficient batching and fair scheduling for deep learning inference on time-sharing devices.” In: *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE. 2020, pp. 609–618.

[191] Thinh Quang Dinh et al. “Offloading in mobile edge computing: Task allocation and computational frequency scaling.” In: *IEEE Transactions on Communications* 65.8 (2017), pp. 3571–3584.

[192] Meng-Hsi Chen, Ben Liang, and Min Dong. “Multi-user multi-task offloading and resource allocation in mobile cloud systems.” In: *IEEE Transactions on Wireless Communications* 17.10 (2018), pp. 6790–6805.

[193] Mithun Mukherjee et al. “Distributed deep learning-based task offloading for UAV-enabled mobile edge computing.” In: *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE. 2020, pp. 1208–1212.

[194] Han Hu et al. “Mobility-aware offloading and resource allocation in a MEC-enabled IoT network with energy harvesting.” In: *IEEE Internet of Things Journal* 8.24 (2021), pp. 17541–17556.

[195] Seung-Wook Kim et al. “Edge-Network-Assisted Real-Time Object Detection Framework for Autonomous Driving.” In: *IEEE Network* 35.1 (2021), pp. 177–183. DOI: `10.1109/MNET.011.2000248`.

[196] Xukan Ran et al. “Deepdecision: A mobile deep learning framework for edge video analytics.” In: *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE. 2018, pp. 1421–1429.

[197] Luyang Liu, Hongyu Li, and Marco Gruteser. “Edge assisted real-time object detection for mobile augmented reality.” In: *The 25th annual international conference on mobile computing and networking*. 2019, pp. 1–16.

[198] Andrea Fresa and Jaya Prakash Varma Champati. “An Offloading Algorithm for Maximizing Inference Accuracy on Edge Device in an Edge Intelligence System.” In: *Proceedings of the 25th International ACM Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems*. 2022, pp. 15–23.

[199] Minoo Hosseinzadeh et al. “Joint compression and offloading decisions for deep learning services in 3-tier edge systems.” In: *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*. IEEE. 2021, pp. 254–261.

[200] Min Chen and Yixue Hao. “Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network.” In: *IEEE Journal on Selected Areas in Communications* 36.3 (2018), pp. 587–597. DOI: `10.1109/JSAC.2018.2815360`.

[201] Joseph Redmon et al. “You only look once: Unified, real-time object detection.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[202] Woosuk Kwon et al. “Nimble: Lightweight and parallel gpu task scheduling for deep learning.” In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 8343–8354.

[203] Jyrki Alakuijala et al. “JPEG XL next-generation image compression architecture and coding tools.” In: *Applications of Digital Image Processing XLII*. Vol. 11137. SPIE. 2019, pp. 112–124.

[204] Rosa Andrie Asmara et al. "Prediction of traffic density using yolo object detection and implemented in raspberry pi 3b+ and intel ncs 2." In: *2020 4th International Conference on Vocational Education and Training (ICOVET)*. IEEE. 2020, pp. 391–395.

[205] Xiaobo Zhao et al. "Improving the Accuracy-Latency Trade-off of Edge-Cloud Computation Offloading for Deep Learning Services." In: *2020 IEEE Globecom Workshops (GC Wkshps*. IEEE. 2020, pp. 1–6.

[206] Elie El Haber, Tri Minh Nguyen, and Chadi Assi. "Joint optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds." In: *IEEE Transactions on Communications* 67.5 (2019), pp. 3407–3421.

[207] Yves Pochet and Laurence A Wolsey. *Production planning by mixed integer programming*. Vol. 149. Springer, 2006.

[208] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[209] Fang Fang et al. "Optimal resource allocation for delay minimization in NOMA-MEC networks." In: *IEEE Transactions on Communications* 68.12 (2020), pp. 7867–7881.

[210] Zhi-Quan Luo et al. "Semidefinite relaxation of quadratic optimization problems." In: *IEEE Signal Processing Magazine* 27.3 (2010), pp. 20–34.

[211] Fuzhen Zhang. *Matrix theory: basic results and techniques*. Springer, 2011.

[212] James C Bezdek and Richard J Hathaway. "Some notes on alternating optimization." In: *Advances in Soft Computing—AFSS 2002: 2002 AFSS International Conference on Fuzzy Systems Calcutta, India, February 3–6, 2002 Proceedings*. Springer. 2002, pp. 288–300.

[213] Gui-Song Xia et al. "DOTA: A Large-Scale Dataset for Object Detection in Aerial Images." In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[214] Narendra Karmarkar. "A new polynomial-time algorithm for linear programming." In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. 1984, pp. 302–311.