ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ
ΠΛΗΡΟΦΟΡΙΚΗΣ

Τεχνολογίες και μηχανισμοί μοντελοποίησης και πρόβλεψης απόδοσης υπηρεσιοστρεφών εφαρμογών και υποδομών

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Γεώργιος Τ. Κουσιουρής

Επιβλέπων: Ομ. Καθηγητής Ε.Ν. Πρωτονοτάριος

Αθήνα, Φεβρουάριος 2012

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Τεχνολογίες και μηχανισμοί μοντελοποίησης και πρόβλεψης απόδοσης υπηρεσιοστρεφών εφαρμογών και υποδομών

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Γεώργιος Τ. Κουσιουρής

Συμβουλευτική Επιτροπή : Εμμανουήλ Ν. Πρωτονοτάριος
Θεοδώρα Α. Βαρβαρίγου
Γεώργιος Π. Παπαβασιλόπουλος

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την …….

.....	.....	.....
Ε.Ν. Πρωτονοτάριος Ομ. Καθηγητής Ε.Μ.Π.	Θ.Α. Βαρβαρίγου Καθηγήτρια Ε.Μ.Π.	Γ.Π. Παπαβασιλόπουλος Καθηγητής Ε.Μ.Π
.....	.....	.....
Σ. Παπαβασιλείου Αν. Καθηγητής Ε.Μ.Π	Β. Λούμος Καθηγητής Ε.Μ.Π	Α. Γ. Σταφυλοπάτης Καθηγητής Ε.Μ.Π

.....
Ε. Βαρβαρίγος
Καθηγητής Πανεπιστημίου Πατρών

Αθήνα, Φεβρουάριος 2012

.....

Γεώργιος Τ. Κουσιουρής

Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

ΠΡΟΛΟΓΟΣ

Η διδακτορική διατριβή που παρουσιάζεται στις επόμενες σελίδες εκπονήθηκε από το Δεκέμβριο του 2006 μέχρι τον Δεκέμβριο του 2011, στο εργαστήριο Τηλεπικοινωνιών του τομέα Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής, στη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου. Κατά την διάρκεια της εκπόνησης αυτής της διατριβής, είχα την ευκαιρία να ασχοληθώ με αρκετά ενδιαφέροντα επιστημονικά θέματα που αφορούν κυρίως στους τομείς της προδιαγραφής, του σχεδιασμού, της μοντελοποίησης και του ελέγχου υπηρεσιοστρεφών υποδομών και να αποκτήσω πολύτιμη εμπειρία και γνώσεις.

Θα ήθελα να ευχαριστήσω από τα βάθη της καρδιάς μου, τους καθηγητές μου κ.κ. Ε. Πρωτονοτάριο και κ. Θεοδώρα Βαρβαρίγου για το ενδιαφέρον που έδειξαν, για τις πολύτιμες συμβουλές τους και για την ιδιαίτερη στήριξη που μου παρείχαν κατά την διάρκεια αυτής της πορείας μου, καθώς επίσης τον καθηγητή κ. Γεώργιο Παπαβασιλόπουλο. Επίσης, θα ήθελα να ευχαριστήσω όλους τους συναδέλφους μου στην ερευνητική ομάδα με τους οποίους συνεργάστηκα άψογα και επιτυχώς όλα αυτά τα χρόνια.. Ιδιαίτερες ευχαριστίες ωστόσο θα ήθελα να απευθύνω στους στενούς μου συνεργάτες και κυρίως στους Ανδρέα Μενύχτα, Δημοσθένη Κυριαζή, Κλεοπάτρα Κωνσταντέλη, Γρηγόρη Κατσαρό και Σπυρίδωνα Γωγουβίτη με τους οποίους μοιραστήκαμε τις πάρα πολλές ώρες της ερευνητικής εργασίας .

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου για την πίστη της σε εμένα και τις επιλογές μου.

Γεώργιος Τ. Κουσιουρής

Ιανουάριος 2012

.

.

Πίνακας περιεχομένων

.

.

.

.

Ευρετήριο Σχημάτων

.

.

.

Ευρετήριο Πινάκων

.

.

Ευρετήριο Εξισώσεων

.

.

Περίληψη

Η ανάπτυξη υπηρεσιοστρεφών υποδομών παροχής λογισμικού και υλισμικού ως υπηρεσία καθιστά εφικτή την χρησιμοποίηση αυτών από εξωτερικούς χρήστες με τη μορφή πληρωμής με βάση τη χρήση. Για την εξασφάλιση της απαιτούμενης ποιότητας υπηρεσίας στην παροχή αυτών των υπηρεσιών, είναι απαραίτητος ένας διαστρωματικός μηχανισμός αντιστοίχισης και μοντελοποίησης των χαρακτηριστικών της εφαρμογής που προσφέρονται σαν όροι στα συμβόλαια επιπέδου υπηρεσιών σε χαρακτηριστικά φυσικών πόρων εκτέλεσης. Στην διαδικασία αυτή πρέπει να ξεπεραστούν αγκυλώσεις και περιορισμοί που προκύπτουν λόγω της ύπαρξης διαφορετικών οντοτήτων (Πάροχος Εφαρμογής, Πάροχος Πλατφόρμας και Πάροχος Υποδομών) κατά μήκος της αλυσίδας αξίας των υπηρεσιοστρεφών υποδομών/αρχιτεκτονικών.

Στην παρούσα διατριβή αναλύονται διαφορετικές υποψήφιες μεθοδολογίες δυναμικής δημιουργίας μοντέλων εφαρμογών σε υπηρεσιοστρεφείς υποδομές και επιλέγεται η καταλληλότερη σύμφωνα με τους προαναφερθέντες περιορισμούς. Επιπλέον αναλύονται οι αδυναμίες αυτής και εφαρμόζονται καινοτόμοι τρόποι αντιμετώπισής τους, ώστε να είναι εφικτή η εφαρμογή της στο ζητούμενο πλαίσιο. Η τελική μορφή αυτής (γενετικά βελτιστοποιημένα νευρωνικά δίκτυα) δοκιμάζεται σε διαφορετικού τύπου εφαρμογές με στόχο την πρόβλεψη της απόδοσής τους με βάση το χρησιμοποιούμενο υλικό και την διαμόρφωσή τους. Επιπλέον, αναπτύσεται ένας μηχανισμός 2 επιπέδων που επεκτείνει

.

την χρήση της και σε περιπτώσεις μη προκαθορισμένης και γνωστής ζήτησης της εφαρμογής. Μέσω της ανάλυσης των ιστορικών στοιχείων χρήσης αναγνωρίζονται μοτίβα συμπεριφοράς και δημιουργείται το κατάλληλο πλαίσιο συνδυασμένης πρόβλεψης.

Επιπλέον, σχεδιάζεται και υλοποιείται ένα αποσυζευγμένο υπηρεσιοστρεφές πλαίσιο για την ενθυλάκωση της επιλεγμένης μεθόδου σε περιβάλλοντα υπολογιστικών νεφών και το οποίο βασίζεται σε μαθηματικό λογισμικό ειδικού σκοπού. Με βάση το σχεδιασμό αυτό, είναι εφικτή η γρηγορότερη και ευκολότερη ανάπτυξη μεθόδων πρόβλεψης απόδοσης και η εναλλαγή τους. Επίσης λόγω της χαρακτηριστικής διαστρωμάτωσης, είναι εφικτή η εναλλαγή τεχνολογιών σε όλα τα ενδιάμεσα στρώματα, γεγονός που επιτρέπει την εκμετάλλευση καινοτόμων εξελίξεων στις αντίστοιχες υλοποιήσεις. Μέσω της ενδελεχούς ανάλυσης του πλαισίου αυτού, είναι εφικτή η ανίχνευση των κυριότερων σημείων καθυστέρησης και η δημιουργία εναλλακτικών μορφών υλοποίησης που μπορούν να εναλλάσονται σε πραγματικό χρόνο ώστε να ελαχιστοποιήσουν τις ανάγκες πόρων της υπηρεσίας πρόβλεψης.

Επιπρόσθετα, η διατριβή αυτή αναλύει την αλληλεπίδραση των εφαρμογών σε ένα κοινόχρηστο υπηρεσιοστρεφές πλαίσιο εκτέλεσης όπως τα Υπολογιστικά Νέφη. Επιλέγεται ένας περιορισμένος αριθμός τυπικών δοκιμίων και ανιχνεύεται η μείωση της απόδοσής τους λόγω του διαμοιρασμού των φυσικών πόρων. Παράμετροι που λαμβάνονται υπόψη κατά τη διαδικασία αυτή περιλαμβάνουν επίσης χαρακτηριστικά χρονοδρομολόγησης στις μονάδες επεξεργασίας καθώς και τρόπους ανάθεσης των εργασιών σε πολυεπεξεργαστικές αρχιτεκτονικές. Με βάση τα πειραματικά δεδομένα, μοντέλα πρόβλεψης της αλληλεπίδρασης αυτής αναπτύσονται, με βάση τα οποία ο

Πάροχος Υποδομών μπορεί να γνωρίζει εκ των προτέρων την επίδοση ενός συγκεκριμένου συνδυασμού εργασιών που ανατίθενται σε ένα φυσικό κόμβο. Με τον τρόπο αυτό μπορεί να επιλέξει τους βέλτιστους συνδυασμούς για την ομαλότερη λειτουργία της υποδομής και την εξασφάλιση της ποιότητας υπηρεσίας.

Λέξεις κλειδιά: Υπηρεσιοστρεφείς Υποδομές, Υπολογιστικό Νέφος, Τεχνητή Νοημοσύνη, Γενετικοί Αλγόριθμοι, Τεχνητά Νευρωνικά Δίκτυα, Πρόβλεψη απόδοσης, Εικονικοποίηση

.

.

Abstract

The advent of service oriented infrastructures providing software and hardware as a service has rendered feasible the utilization of the latter by external users in the form of pay-per-use resources. In order to ensure the demanded quality of service during the provisioning of these resources, a multilayer translation and modeling mechanism is necessary, in order to convert the application terms that are offered in traditional service level agreement contracts to resource level attributes. During this process, specific requirements must be met and limitations overcome, that originate mainly from the existence of different entities (Software-as-a-Service, Platform-as-a-Service and Infrastructure-as-a-Service providers) along the value chain of service oriented infrastructures and architectures.

In the current thesis different candidate methodologies are analyzed with regard to their ability to dynamically create application models in service oriented infrastructures and the fittest one is selected based on the aforementioned limitations and requirements. Its weaknesses are also investigated and innovative approaches are applied in order to render it applicable to the desired framework. The final form of this method (genetically optimized artificial neural networks) is validated in a variety of real world applications for predicting their performance based on the given hardware of execution, their high

level parameters and configuration. What is more, a two level mechanism (workload forecasting and translation) extends its use in cases where the application workload cannot be foreseen by the owner of the service. Through the proposed framework, historical data are analyzed and patterns of usage are discovered that aid in the full automation of the management framework.

Furthermore, a multi-layer, decoupled and service oriented framework, based on specialized numerical software, is designed and implemented for incorporating the selected method in cloud computing environments. Based on this scripting software, the development or evolution of the modeling methods is enhanced and aided. What is more, due to the decoupling of the layers, it is feasible to interchange technologies in all the involved layers without affecting the remaining framework, in order to exploit advances in the respective technological fields. Through the detailed analysis of the framework's behaviour the performance bottlenecks are discovered and alternative design methods are created in order to interchange between them based on the runtime usage of the service. This way the needed resources for the framework are minimized and its performance is optimized.

In addition, the thesis analyzes the performance interference between concurrently running virtual resources and applications in a multitenant service oriented execution framework like Clouds. A limited number of benchmarks is chosen that depicts characteristic usage of the hardware resources and the degradation of their performance is measured due to the coexistence in the same physical host. Parameters that are taken under consideration in this process include also real time scheduling configuration, necessary for guaranteeing a process's time on the CPU, along with assignment patterns

on multicore architectures. Based on these experimental data, suitable prediction models are created, based on which the IaaS provider may have a priori knowledge of the overhead inserted by the execution of a specific task combination on a physical host. Through this knowledge it may choose the optimal combinations for the smooth operation of the infrastructure and the guarantee of the Quality of Service offered by the resources.

Key words: Service Oriented Infrastructures, Cloud Computing, Artificial Intelligence, Genetic Algorithms, Artificial Neural Networks, Performance Prediction, Virtualization

.

.

1 *Εισαγωγή*

Στο παρόν κεφάλαιο παρατίθεται ο ορισμός του Υπολογιστικού Νέφους, του βασικού μοντέλου εφαρμοσμένων υπηρεσιοστρεφών υποδομών, καθώς και οι διαφορετικοί ρόλοι των οντοτήτων μέσα σε αυτό και των σχέσεων μεταξύ τους. Αναλύονται τα ανοικτά ερευνητικά θέματα και περιγράφονται οι κυριότερες προκλήσεις λόγω της δομής και της αρχιτεκτονικής της τεχνολογίας αυτής. Τέλος αναφέρεται η συνεισφορά της παρούσας διατριβής και η οργάνωση του κειμένου.

1.1 Ορισμός του Υπολογιστικού Νέφους

Σύμφωνα με τον οργανισμό NIST (National Institute of Standards and Technology) το Υπολογιστικό Νέφος (Cloud Computing) ορίζεται ως εξής [1]:

Το Υπολογιστικό Νέφος είναι ένα μοντέλο για τη πραγματοποίηση της συνεχούς, κατάλληλης, και κατά παραγγελία πρόσβασης μέσω δικτύου σε μια κοινή ομάδα διαμορφώσιμων πόρων υπολογισμού (π.χ., δίκτυα, κεντρικοί υπολογιστές, αποθηκευτικοί χώροι, εφαρμογές, και υπηρεσίες) που μπορούν να είναι διαθέσιμα γρήγορα και με την ελάχιστη διαχειριστική προσπάθεια ή αλληλεπίδραση φορέων παροχής υπηρεσιών.

Από την ίδια πηγή προκύπτουν και τα βασικά χαρακτηριστικά ενός Υπολογιστικού Νέφους:

- Αυτο-εξυπηρέτηση: ένας καταναλωτής μπορεί να αυξήσει τους υπολογιστικούς πόρους που χρησιμοποιεί χωρίς ανθρώπινη αλληλεπίδραση
- Πρόσβαση μέσω διαδικτύου: οι δυνατότητες αυτές παρέχονται μέσω δικτύου και μπορούν να προσπελαστούν μέσω εφαρμογών πελάτη
- Διαχείριση πόρων: οι πόροι του παρόχου εξυπηρετούν πολλαπλούς καταναλωτές με διαφορετικές φυσικές και εικονικές ανάγκες και ανατίθενται σε αυτούς δυναμικά. Οι καταναλωτές δεν γνωρίζουν την κατάσταση στο εσωτερικό της υποδομής και δεν μπορούν να ορίσουν τα φυσικά μηχανήματα που θα καταλάβουν οι πόροι τους (εκτός από την γενική γεωγραφική τοποθεσία)
- Ελαστικότητα πόρων: οι πόροι που χρησιμοποιεί ένας καταναλωτής μπορούν να μεταβάλλονται ελαστικά και γρήγορα, ώστε να ακολουθούν τη ζήτηση και τις ανάγκες του τελευταίου
- Καταγραφή χρησιμοποιούμενων πόρων: οι πάροχοι πρέπει να παρέχουν τρόπους καταγραφής των χρησιμοποιούμενων πόρων για λόγους χρέωσης και διαχείρισης.

Η επίτευξη των παραπάνω στόχων γίνεται μέσω των Υπηρεσιοστρεφών Υποδομών [2]. Σε αυτές, οι βασικές λειτουργίες παρέχονται με τη μορφή υπηρεσίας, δηλαδή με την έκθεση μιας κατάλληλης διεπαφής μέσω δικτύου, μέσω της οποίας ένας καταναλωτής ή ένα λογισμικό διαχείρισης μπορεί αυτόματα να καλέσει για να πραγματοποιηθεί μια πράξη (π.χ. εκκίνηση μιας εικονικής μηχανής, εκκίνηση μιας υπηρεσίας λογισμικού κλπ.)

1.1.1 Σχέση με Περιβάλλοντα Πλέγματος

Τα περιβάλλοντα Πλέγματος [3] ξεκίνησαν σαν μία πολλά υποσχόμενη τεχνολογία η οποία ενσωμάτωνε ετερογενείς υπολογιστικούς πόρους μεγάλης κλίμακας σε ένα κοινό σύστημα διαχείρισης. Οι πόροι αυτοί πιθανόν να άνηκαν στον ίδιο ή διαφορετικούς παρόχους (και αντίστοιχες γεωγραφικές τοποθεσίες) και η διαχείρισή τους επιτυγχανόταν μέσω των υπηρεσιοστρεφών αρχιτεκτονικών και υλοποιήσεων. Το υπηρεσιοστρεφές πλαίσιο φρόντιζε να παρέχει την απαραίτητη διασύνδεση και κοινή διαχείριση.

Παρόλο που η τεχνολογία αυτή ξεκίνησε πολύ δυναμικά, περιορίστηκε τελικά σε ακαδημαϊκούς και ερευνητικούς ρόλους ή σε πολύ συγκεκριμένες εφαρμογές που μπορούσαν να εκμεταλλευτούν τις δυνατότητές τους αλλά και να περιορίσουν τα μειονεκτήματά τους. Το βασικό μειονέκτημα ήταν η έλλειψη απομόνωσης μεταξύ των χρηστών, που δημιουργούσε προβλήματα σχετικά με την εγγύηση της παρεχόμενης υπηρεσίας, την ασφάλεια αλλά κυρίως με τις διαφορετικές απαιτήσεις περιβάλλοντος κάθε εφαρμογής. Έτσι, δύο εφαρμογές με διαφορετικές ανάγκες π.χ. σε λειτουργικό σύστημα, βιβλιοθήκες ή εκδόσεις λογισμικού δεν μπορούσαν να εκτελούνται στον ίδιο φυσικό πόρο.

Η βασική διαφορά των Υπολογιστικών Νεφών έγκειται στο γεγονός ότι χρησιμοποιούν την τεχνολογία της εικονικοποίησης (virtualization). Μέσω αυτής της τεχνικής, κάθε χρήστης είναι εντελώς απομονωμένος από τους υπόλοιπους και μπορεί να χειρίζεται κατά το δοκούν την διαμόρφωση του περιβάλλοντος εκτέλεσης των εφαρμογών του. Το γεγονός αυτό επέτρεψε στα Υπολογιστικά Νέφη να γίνουν σχεδόν αμέσως εμπορικά εκμεταλλεύσιμα και να μπορούν να χρησιμοποιηθούν για υπολογιστική ισχύ γενικού σκοπού. Μέσω αυτής της τεχνολογίας άτομα ή επιχειρήσεις

έχουν πρόσβαση σε ελαστικούς υπολογιστικούς πόρους ή λογισμικό με πληρωμή ανάλογα με τη χρήση, χωρίς την ανάγκη κεφαλαιακής επένδυσης εξαρχής ή μετέπειτα συντήρησης. Έτσι μπορούν να τα προσαρμόζουν ανάλογα με τις ανάγκες τους ή τη ζήτηση, μειώνοντας το κόστος.

1.2 Μοντέλα Επιχειρηματικότητας- Ρόλοι SPI

Το βασικό επιχειρηματικό μοντέλο που ακολουθείται στα Clouds είναι το λεγόμενο SPI (Software-Platform-Infrastructure [42]). Σε αυτήν την ενότητα θα αναλύσουμε τους διαφορετικούς ρόλους που προκύπτουν από αυτό, καθώς όπως θα δούμε στη συνέχεια διαδραματίζει σημαντικό ρόλο στις προκλήσεις που εμφανίζονται στο θέμα που μελετάται. Σύμφωνα λοιπόν με αυτό το διαχωρισμό, οι διακριτοί ρόλοι είναι οι ακόλουθοι:

- Υπεύθυνος για την ανάπτυξη/προμηθευτής εφαρμογής: μια οντότητα που μετασχηματίζει μια κανονική εφαρμογή σε μια αρθρωτή προσανατολισμένη προς τις υπηρεσίες έκδοση (λογισμικό ως υπηρεσία – Software as a Service-SaaS). Εκτός από την ανάπτυξη/προσαρμογή της εφαρμογής, αυτός ο ρόλος προσδιορίζει επίσης ποιοι είναι οι όροι που μπορούν να ρυθμιστούν από τους τελικούς καταναλωτές της εφαρμογής:
    1. Παράμετροι φόρτου εργασίας: διαμορφώσιμες παράμετροι με τις οποίες ένας πελάτης θέλει να εκτελέσει μια υπηρεσία (π.χ. αριθμός χρηστών, ανάλυση μιας εικόνας σε εφαρμογές πολυμέσων κ.λπ.)
    2. Ποιότητα των παραμέτρων υπηρεσίας (Quality of Service-QoS): η αναμενόμενη έξοδος της υπηρεσίας, αυτή που χρησιμοποιείται για να εκφράσει τα επίπεδα QoS (π.χ. χρόνος απόκρισης, πλαίσια ανά

δευτερόλεπτο μιας μετάδοσης πολυμέσων κ.λπ.). Αυτό είναι επίσης γνωστό ως Βασικοί Δείκτες Απόδοσης (Key Performance Indicators-KPIs)

- Προμηθευτής πλατφόρμας: μια οντότητα, Platform as a Service (PaaS), η οποία προσφέρει το πλαίσιο μέσω του οποίου ένα SaaS μπορεί να εκτελεσθεί σε μια υποδομή (Infrastructure as a Service-IaaS). Αυτός ο ρόλος ενσωματώνει ενέργειες όπως η μοντελοποίηση της εφαρμογής, η πρόβλεψη απόδοσής της, η παρακολούθηση, αξιολόγηση των γεγονότων και υλοποίηση διορθωτικών ενεργειών (π.χ. αύξηση των πόρων). Σε αυτήν την διαδικασία μπορεί να χρησιμοποιήσει τις πληροφορίες που παρέχονται από τον υπεύθυνο για την ανάπτυξη εφαρμογής μέσω μιας περιγραφής του τμήματος λογισμικού (π.χ. σε γλώσσα XML)
- Προμηθευτής υποδομής: μια οντότητα (π.χ. [4]) που προσφέρει τους πόρους υλικού (επεξεργαστές, αποθηκευτικοί χώροι, δίκτυο), κυρίως μέσω της εκτενούς χρήσης virtualization (Infrastructure as a Service-IaaS) και υπηρεσιοστρεφών διεπαφών.
- Καταναλωτής/πελάτης: μια οντότητα που χρησιμοποιεί μια εφαρμογή που προσφέρεται ως υπηρεσία. Ο καταναλωτής έρχεται σε επαφή με τον προμηθευτή PaaS που έχει καταστήσει αυτό το SaaS διαθέσιμο και ζητά αυτό το λογισμικό για μια συγκεκριμένη διαμόρφωση (παραμέτρους φόρτου εργασίας). Επιπλέον απαιτεί ορισμένα επίπεδα ποιότητας της υπηρεσίας (QoS), όπως αυτή υπολογίζεται από τους συγκεκριμένους βασικούς δείκτες απόδοσης (KPIs) της εφαρμογής. Ο προμηθευτής PaaS έρχεται έπειτα σε επαφή με έναν προμηθευτή

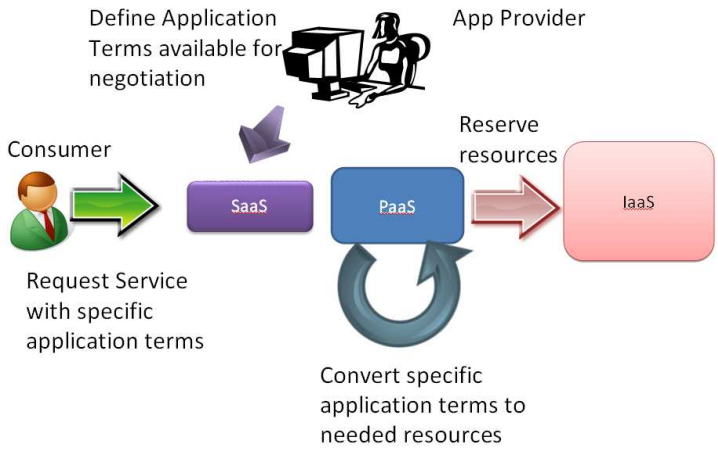
IaaS και ζητά τους πόρους υλικού που έχει προβλέψει ότι θα ικανοποιήσουν τις ανάγκες του καταναλωτή. Μετά από μια επιτυχή διαπραγμάτευση, η εφαρμογή (SaaS) εκκινείται (από το PaaS) στις υποδομές που παρέχονται από το IaaS.

Όλες οι προαναφερθείσες ενέργειες επικυρώνονται τυπικά μέσω των Συμφωνιών Επιπέδων Υπηρεσιών (Service Level Agreements-SLAs [6]) μεταξύ των συμβαλλόμενων μερών. Το SLA περιγράφει τη συμφωνία για αυτούς τους όρους και διατάξεις και δεσμεύει νομικά τους εμπλεκόμενους παρόχους. Χαρακτηριστικά, υπάρχουν δύο SLAs, ένα μεταξύ του καταναλωτή και του προμηθευτή PaaS (Application SLA, A-SLA), που αναφέρονται στους όρους εφαρμογής (παράμετροι φόρτου εργασίας και επίπεδα τιμών των παραμέτρων υπηρεσίας), και ένα μεταξύ των προμηθευτών PaaS και IaaS (Technical SLA, T-SLA), που εκφράζονται με όρους επιπέδων εικονικών υπολογιστικών πόρων.

1.3 Ανοικτά Ερευνητικά Θέματα

Σε αυτό το σύνθετο και πολυστρωματικό οικοσύστημα, ένα από τα κρισιμότερα βήματα είναι ο ρόλος του PaaS στη μετάφραση των καθορισμένων από τον καταναλωτή παραμέτρων εφαρμογής (φόρτος εργασίας και QoS, όπως αυτοί εκφράζονται στο A-SLA) στις ιδιότητες επιπέδων των πόρων (δεδομένου ότι αυτοί απαιτούνται για να εκφραστούν στο T-SLA). Εντούτοις, η ανταλλαγή των πληροφοριών μεταξύ των οντοτήτων που βρίσκονται στα διαφορετικά στρώματα είναι δύσκολη για τεχνικούς και επιχειρηματικούς λόγους. Από τη μία πλευρά, είναι εξαιρετικά απίθανο για αυτές τις οντότητες, ειδικά για τους προμηθευτές εφαρμογής και τους παρόχους υποδομής, να εκθέσουν/αποκαλύψουν τις εσωτερικές λειτουργικές παραμέτρους και τις διαδικασίες

των στρωμάτων τους σε άλλα στρώματα λόγω του κινδύνου πρόσβασης ανταγωνιστών σε αυτές τις πληροφορίες. Αφ' ετέρου, η ανταλλαγή τέτοιων πληροφοριών δεν είναι πάντα τεχνικά εφικτή. Κάθε οντότητα εστιάζει σε ένα ιδιαίτερο σύνολο παραμέτρων που μπορεί να ερμηνευθεί από αυτό το στρώμα και βασίζει σε αυτές τις παραμέτρους τους αντίστοιχους μηχανισμούς για την παροχή υπηρεσιών. Επιπλέον, η χρήση βιβλιοθηκών ή γενικά κώδικα τρίτων (third party software) για τα οποία δεν υπάρχει γνώση της εσωτερικής δομής δυσχεραίνει την ροή της πληροφορίας, ακόμα και αν αρθούν οι προαναφερθέντες περιορισμοί. Η έλλειψη προτύπων για τα Υπολογιστικά Νέφη είναι ένα άλλο μεγάλο ζήτημα που περιορίζει τον αποτελεσματικό διαμοιρασμό των πληροφοριών μεταξύ των διαφορετικών οντοτήτων. Μερικές ενδεχόμενες προσπάθειες προς αυτήν την κατεύθυνση περιλαμβάνουν τα OCCI[4] και OVF[5].



Σχήμα 1: Ρόλοι οντοτήτων και αλληλεπίδραση μεταξύ τους στα Υπολογιστικά Νέφη

Στο τωρινό οικοσύστημα Νεφών δεν υπάρχει τρόπος ή πρόβλεψη για την χαρτογράφηση αυτή από A-SLA σε T-SLA. Η συνηθέστερη μορφή είναι ένα στατικό SLA το οποίο δίνουν οι πάροχοι στους καταναλωτές και αναφέρεται σε παραμέτρους όπως η διαθεσιμότητα των πόρων. Ο καταναλωτής της υπηρεσίας πρέπει να διαλέξει από

Η σελίδα αυτή είναι σκόπιμα λευκή

μόνος του όμως πόσους πόρους χρειάζεται για μια συγκεκριμένη εφαρμογή με συγκεκριμένο φορτίο. Ο πάροχος του εξασφαλίζει μόνο ότι οι εικονικές του μηχανές θα είναι διαθέσιμες π.χ. το 99.9% του χρόνου.

Επιπλέον, ένα σημαντικό κενό στα υπάρχοντα Νέφη είναι η έλλειψη εγγυήσων απόδοσης πραγματικού χρόνου και ποιότητας της υπηρεσίας. Συγκεκριμένα, για τις εφαρμογές πραγματικού χρόνου (real-time), ο πάροχος πρέπει να λαμβάνει υπόψη του το βασικό χαρακτηριστικό αυτών, που είναι η πραγματοποίηση μιας πράξης από τη συγκεκριμένη εφαρμογή μέσα σε κάποιο αυστηρό χρονικό πλαίσιο.

Οι πάροχοι υπολογιστικών πόρων δίνουν πρόσβαση σε συγκεκριμένα κβαντισμένες εικονικές μηχανές (π.χ. μία εικονική μηχανή ανά πυρήνα) χωρίς να εγγυώνται όμως την πραγματική απόδοσή τους. Επιπλέον, για την υποστήριξη εφαρμογών πραγματικού χρόνου είναι απαραίτητη μια λεπτομερής ανάλυση κάθε εφαρμογής και αυστηρός έλεγχος των πόρων, ώστε να ικανοποιούνται οι χρονικοί περιορισμοί που εισέρχονται λόγω του πραγματικού χρόνου. Για να γίνει αυτό όμως, χρειάζεται μια λογική προ-σχεδιασμού της κατανομής των πόρων, που καθιστά απαραίτητο ένα μοντέλο in advance reservation. Διορθώσεις κατά τη διάρκεια της πραγματικής λειτουργίας μπορούν να γίνουν, αλλά πρέπει να είναι περιορισμένης κλίμακας ώστε να μην υπάρχουν παραβιάσεις στους χρονικούς περιορισμούς.

Επιπλέον, ένα πολύ σημαντικό θέμα που υπεισέρχεται σε διαμοιραζόμενες υποδομές είναι η αλληλεπίδραση μεταξύ των εικονικών πόρων που εκτελούνται σε ένα φυσικό πόρο και ανήκουν σε διαφορετικούς χρήστες. Αυτή η αλληλεπίδραση μπορεί να ποικίλει ανάλογα με τον τρόπο δρομολόγησης, την αρχιτεκτονική του φυσικού πόρου και το είδος των εφαρμογών που εκτελούνται μέσα στις εικονικές μηχανές. Άρα ένα πάροχος

Νέφους οφείλει να έχει τρόπους εύρεσης και ελαχιστοποίησης αυτής της αλληλεπίδρασης, ώστε να παρέχει την ποιότητα υπηρεσίας που υπόσχεται στους πελάτες του.

1.4 Προκλήσεις Πρόβλεψης Απόδοσης σε Υπολογιστικά Νέφη

Όπως είδαμε στη προηγούμενη παράγραφο, το στρώμα PaaS έχει την ευθύνη της μετατροπής/μετάφρασης των παραμέτρων που χρησιμοποιούνται στο επίπεδο εφαρμογής (SaaS) για να χαρακτηρίσουν το φόρτο εργασίας και τις παραμέτρους ποιότητας σε συμπαγείς παραμέτρους του στρώματος IaaS, δηλαδή υπολογιστικούς πόρους (κυρίως ταχύτητα CPU, πλήθος επεξεργαστών, μέγεθος μνήμης κλπ). Σε αυτή του την αποστολή έχει να αντιμετωπίσει σημαντικούς περιορισμούς που υπεισέρχονται λόγω της μορφής και της δομής των Υπολογιστικών Νεφών:

- Περιορισμός 1: εκτίμηση βασισμένη στη διαμόρφωση (παράμετροι φόρτου και KPIs επιπέδων εφαρμογής) κάθε υπηρεσίας, προκειμένου να επιτευχθεί η ακρίβεια στις προβλέψεις για κάθε ξεχωριστό instance της υπηρεσίας (δηλαδή για κάθε ξεχωριστό SLA)
- Περιορισμός 2: ο πηγαίος κώδικας του τμήματος λογισμικού δεν είναι διαθέσιμος στο PaaS, γεγονός που οφείλεται όπως είδαμε στις διαφορετικές οντότητες μεταξύ των στρωμάτων και της περιορισμένης ροής πληροφοριών
- Περιορισμός 3: δυνατότητα να ανιχνευθεί ο γραμμικός και μη γραμμικός συσχετισμός μεταξύ των εισόδων και των εξόδων του μοντέλου απόδοσης (απαραίτητου λόγω των προτάσεων λογισμικού όπως η if-then-else περίπτωση που εμφανίζεται στο Σχήμα 2)

• Περιορισμός 4: αυτοματοποιημένη και βελτιστοποιημένη παραγωγή των μοντέλων για κάθε κομμάτι λογισμικού

• Περιορισμός 5: υπηρεσιοστρεφές πλαίσιο για την υλοποίηση του αλγόριθμου πρόβλεψης και τη χρήση αυτού στον κύκλο εργασιών της πλατφόρμας

Επιπλέον, όσον αφορά την αλληλεπίδραση μεταξύ των ταυτόχρονα εκτελέσιμων εικονικών μηχανών στο στρώμα IaaS, ο πιο σημαντικός περιορισμός είναι η έλλειψη πληροφοριών από τον IaaS προς τον PaaS ως προς την τοπολογία των φυσικών πόρων και την έλλειψη πληροφοριών για τις εφαρμογές άλλων χρηστών που εκτελούνται στο ίδιο μηχάνημα..

```
1. If application-level-parameter=a
2.   do something that takes milliseconds
3. elsif application-level-parameter=b
4.   do something that takes hours
5. // or
6. For every pixel in frame
7.   process
```

Σχήμα 2: Ψευδοκώδικας ροής προγράμματος με μη γραμμικά χαρακτηριστικά

1.5 Συνεισφορά

Η βασική συνεισφορά της παρούσας διατριβής αναφέρεται σε όλα τα παραπάνω θέματα. Αναλύονται σχετικές εργασίες στον τομέα της πρόβλεψης απόδοσης προγραμμάτων λογισμικού και επιλέγεται η μεθοδολογία που ταιριάζει καλύτερα στο συγκεκριμένο πλαίσιο και τους περιορισμούς του. Για την μεθοδολογία αυτή, ακολουθούμε μια αυτοματοποιημένη διαδικασία βελτιστοποίησης, με σκοπό την παραγωγή

Η σελίδα αυτή είναι σκόπιμα λευκή

ικανοποιητικών μοντέλων απόδοσης χωρίς ανθρώπινη παρέμβαση. Η μεθοδολογία αυτή επεκτείνεται με έναν μηχανισμό 2 επιπέδων για την περαιτέρω αυτοματοποίηση της διαδικασίας. Μέσω του δεύτερου επιπέδου, επιτυγχάνεται και πρόβλεψη της μελλοντικής ζήτησης με βάση τα ιστορικά στοιχεία χρήσης της εφαρμογής.

Επιπλέον, υλοποιείται μια αποσυζευγμένη υπηρεσία πρόβλεψης απόδοσης με βάση εξειδικευμένα λογισμικά, στα οποία οποιαδήποτε μέθοδος μπορεί να εφαρμοστεί σαν plug-in, με απλή αντικατάσταση του αντίστοιχου αρχείου. Για την υπηρεσία αυτή αναλύσαμε την συμπεριφορά της και καταλήξαμε σε πολλαπλές μεθόδους λειτουργίας ώστε να βελτιστοποιηθεί η απόκρισή της σε πολλαπλά αιτήματα εξυπηρέτησης.

Επίσης, στα πλαίσια της διατριβής αναλύονται οι αλληλεπιδράσεις μεταξύ ταυτόχρονα εκτελούμενων εικονικών μηχανών στον ίδιο φυσικό πόρο και επιτυγχάνεται η μοντελοποίησή τους ώστε ο πάροχος IaaS να βελτιστοποιήσει την κατανομή τους στους διαθέσιμους πόρους και να ελαχιστοποιήσει το φαινόμενο αυτό.

Στα πλαίσια της εργασίας αυτής προέκυψαν οι παρακάτω δημοσιεύσεις

- Περιοδικά με κρίση
    1. George Kousiouris, Andreas Menychtas, Dimosthenis Kyriazis, Kleopatra Konstanteli Spyridon Gogouvitis, Gregory Katsaros and Theodora Varvarigou, “ Dynamic Design and Performance Analysis of a Multi-Layered, Decoupled Service-Oriented Framework for incorporating numerical software in SOI Performance Prediction ”, under revisions for IEEE Transactions on Services Computing
    2. George Kousiouris, Andreas Menychtas, Dimosthenis Kyriazis1, Spyridon Gogouvitis and Theodora Varvarigou, “ Dynamic, Behavioural-based

Estimation of Resource Provisioning based on High-level Application Terms in Cloud Platforms ”, submitted in Future Generation Computer Systems, Elsevier.

3. Tommaso Cucinotta, Fabio Checconi, George Kousiouris, Kleopatra Konstanteli, Spyridon Gogouvitis, Dimosthenis Kyriazis, Theodora Varvarigou, Alessandro Mazzetti, Zlatko Zlatev, Juri Papay, Michael Boniface, Sören Berger, Dominik Lamp, Thomas Voith and Manuel Stein, “ Virtualised e-Learning on the IRMOS real-time Cloud”, Springer Service Oriented Computing and Applications, 4(4), Oct 2010.
4. A.J. Ferrer, F. Hernandez, J. Tordsson, E. Elmroth, A. Ali-Eldin, C. Zsigri, R. Sirvent, J. Guitart, R.M. Badia, K. Djemame, W. Ziegler, T. Dimitrakos, S.K. Nair, G. Kousiouris, K. Konstanteli, T. Varvarigou, B. Hudzia, A. Kipp, S. Wesner, M. Corrales, N. Forgo, T. Sharif, and C. Sheridan "OPTIMIS: a Holistic Approach to Cloud Service Provisioning, , Future Generation Computer Systems, Elsevier, Vol. 28, No. 1, pp. 66-77, 2012. "
5. George Kousiouris, Tommaso Cucinotta, Theodora Varvarigou, "The Effects of Scheduling, Workload Type and Consolidation Scenarios on Virtual Machine Performance and their Prediction through Optimized Artificial Neural Networks , The Journal of Systems and Software (2011), Elsevier,Volume 84, Issue 8, August 2011, pp. 1270-1291, doi:10.1016/j.jss.2011.04.013."
6. Spyridon Gogouvitis, Kleopatra Konstanteli, Stefan Waldschmidt, George Kousiouris, Gregory Katsaros, Andreas Menychtas, Dimosthenis Kyriazis,

Theodora Varvarigou "Workflow management for soft real-time interactive applications in virtualized environments, , Future Generation Computer Systems, Elsevier, In Press, Corrected Proof, Available online 6 June 2011, ISSN 0167-739X, DOI: 10.1016/j.future.2011.05.017."

7. Dimosthenis Kyriazis, Andreas Menychtas, George Kousiouris, Karsten Oberle, Thomas Voith, Michael Boniface, Eduardo Oliveros, Tommaso Cucinotta, Soren Berger, "A Real-time Service Oriented Infrastructure ,", accepted on the GSTF International Journal on Computing, ISSN 2010-2283.

- Κεφάλαια σε βιβλία
  1. George Kousiouris, Dimosthenis Kyriazis, Theodora Varvarigou, Eduardo Oliveros, Patrick Mandic "Taxonomy and State of the Art of Service Discovery Mechanisms and their relation to the Cloud Computing Stack ,", in Book "Achieving Real-Time in Distributed Computing: From Grids to Clouds", in press.
  2. Matthew Addis, Michael Boniface, Juri Papay, Arturo Servin, Zlatko Zlatev, George Kousiouris "Modelling and Analysing QoS for Real-time Interactive Applications on the Cloud ,", in Book "Achieving Real-Time in Distributed Computing: From Grids to Clouds", in press.
- Διεθνή Συνέδρια
  1. George Kousiouris,George Vafiadis and Theodora Varvarigou, "A Front-end Hadoop based Data Management Service for Efficient Federated Clouds", to appear in 3rd IEEE International Conference on Cloud Computing

Technology and Science (IEEE CloudCom 2011), Athens, Greece, Nov 29-Dec 1, 2011

2. Benno Barnitzke, Wolfgang Ziegler, George Vafiadis, Srijith Nair, George Kousiouris, Marcelo Corrales, Oliver Waldrich, Nikolaus Forgo and Theodora Varvarigou, "Legal Restraints and Security Requirements on Personal Data and Their Technical Implementation in Clouds", in eChallenges 2011, 26-28 Oct. 2011, Florence, Italy
3. Dimosthenis Kyriazis,George Kousiouris, Andreas Menychtas, Anastasios Doulamis and Theodora A. Varvarigou, "Interactive Social TV on Service Oriented Environments: Challenges and Enablers ,", in 3rd International Conference in Games and Virtual Worlds for Serious Applications (VS GAMES 2011), 4-5 May 2011, Athens, Greece
4. George Kousiouris, Dimosthenis Kyriazis, Spyridon V. Gogouvitis, Andreas Menychtas, Kleopatra Konstanteli and Theodora A. Varvarigou, "Translation of Application-level Terms to Resource-level attributes across the Cloud Stack Layers ,", Computers and Communications (ISCC), 2011 IEEE Symposium on , vol., no., pp.153-160, June 28 2011-July 1 2011 doi: 10.1109/ISCC.2011.5984009
5. Tommaso Cucinotta, Fabio Checconi, George Kousiouris, Dimosthenis Kyriazis, Theodora Varvarigou, Alessandro Mazzetti, Zlatko Zlatev, Jury Papay, Michael Boniface, SA¶ren Berger, Dominik Lamp, Thomas Voith, Manuel Stein, "Virtualised e-Learning with Real-Time Guarantees on the IRMOS Platform ,", in Proceedings of the IEEE International Conference on

Service-Oriented Computing and Applications (SOCA 2010), Perth, Australia, December 2010. Best Paper Award!

6. Gregory Katsaros, George Kousiouris, Spyridon V. Gogouvitis, Dimosthenis Kyriazis, Theodora A. Varvarigou, "A service oriented monitoring framework for soft real-time applications ,", in Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2010), Perth, Australia, December 2010.
7. Spyridon V. Gogouvitis, Kleopatra Konstanteli, George Kousiouris, Gregory Katsaros, Dimosthenis Kyriazis, Theodora Varvarigou, "A Service Oriented Architecture for achieving QoS-aware Workflow Management in Virtualized Environments," 6th International Conference on Network and Service Management (CNSM 2010), Niagra Falls, Canada, October 2010. DOI
8. Andreas Menychtas,George Kousiouris,Dimosthenis Kyriazis and Theodora A. Varvarigou, "Minimizing Technical Complexities in Emerging Cloud Platforms ,", in Proceedings of Europar 2010 (CCPI workshop), Ischia, Italy, August 31-September 3, 2010, Lecture Notes in Computer Science, Vol. 6586
9. George Kousiouris, Dimosthenis Kyriazis, Kleopatra Konstanteli, Spyridon Gogouvitis, Gregory Katsaros and Theodora Varvarigou, "A Service-Oriented Framework for GNU Octave-Based Performance Prediction," Services Computing (SCC), 2010 IEEE International Conference on , vol., no., pp.114-121, 5-10 July 2010,doi: 10.1109/SCC.2010.37
10. George Kousiouris, Fabio Checconi, Alessandro Mazzetti, Zlatko Zlatev, Juri Papay, Thomas Voith, Dimosthenis Kyriazis, "Distributed Interactive Real-

time Multimedia Applications: A Sampling and Analysis Framework ,", in 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2010), July 2010, Brussels, Belgium.

11. Dimosthenis Kyriazis, Andreas Menychtas, George Kousiouris, Karsten Oberle, Thomas Voith, Michael Boniface, Eduardo Oliveros, Tommaso Cucinotta, Soren Berger "A Real-time Service Oriented Infrastructure ,", Annual International Conference on Real-time and Embedded Systems (RTES),Singapore, 2010.
12. Dimosthenis Kyriazis, Ralf Einhorn, Lars Furst, Michael Braitmaier, Dominik Lamp, Kleopatra Konstanteli, George Kousiouris, Andreas Menychtas, Eduardo Oliveros, Neil Loughran, Bassem Nasser, " A Methodology for engineering real-time interactive multimedia applications on service oriented infrastructures", in Proceedings of the IADIS International Conference Applied Computing, Timisoara, Romania 14-16 October 2010.
13. Michael Boniface, Bassem Nasser, Juri Papay, Stephen C. Phillips, Arturo Servin, Xiaoyu Yang, Zlatko Zlatev, Spyridon V. Gogouvitis, Gregory Katsaros, Kleopatra Konstanteli, George Kousiouris, Andreas Menychtas, Dimosthenis Kyriazis, "Platform-as-a-Service Architecture for Real-Time Quality of Service Management in Clouds," iciw, pp.155-160, 2010 Fifth International Conference on Internet and Web Applications and Services, 2010. DOI
14. Dimosthenis Kyriazis, Konstantinos Tserpes, George Kousiouris, Andreas Menychtas, Gregory Katsaros, Theodora Varvarigou,"Data Aggregation and

Analysis: A Grid-based approach for Medicine and Biology,", 4th Workshop on High Performance and Grid Computing in Medicine and Biology, The 2008 IEEE International Symposium on Parallel and Distributed Processing and Applications (ISPA), Sydney, Australia, 2008.

15. Spyridon V. Gogouvitis, George Kousiouris, Kleopatra Konstanteli, Theodoros Polychniatis, Andreas Menychtas, Dimosthenis Kyriazis, Theodora A. Varvarigou, "Realtime-enabled workflow management in service oriented infrastructures," Proceedings of the 1st ACM Workshop on Analysis and Retrieval of Events/Actions and Workflows in Video Streams, AREA 2008, Vancouver, British Columbia, Canada, October 31, 2008. DOI
16. Theodoros Polychniatis, Andreas Menychtas, Spyridon V. Gogouvitis, George Kousiouris, Kleopatra Konstanteli, Dimosthenis Kyriazis, Theodora Varvarigou, "Framework Services for Real-time SOIs,", 4th EGEE User Forum, Harbiye Askeri Museum, Istanbul - Turkey, 22-26 September, 2008.
17. Andreas Menychtas, Georgios Kousiouris, Dimosthenis Kyriazis, Theodoros Athanaileas, Dimitra Dionysiou, Georgios Stamatakos, Theodora Varvarigou, Dimitra Kaklamani and Nikolaos Uzunoglu, "Development and adaptation of a web enabled in silico oncology application in grid environment,", 3rd EGEE User Forum, Clermont, France, 2008.

1.6 Οργάνωση του εγγράφου

Το παρόν έγγραφο αποτελείται από έξι (6) κεφάλαια. Στις ενότητες των κεφαλαίων αυτών παρουσιάζεται ουσιαστικά και με αναλυτικό τρόπο το αντικείμενο της διδακτορικής διατριβής.

Στο Κεφάλαιο 2, παρουσιάζεται η βασική μεθοδολογία που ακολουθήθηκε για την επίλυση του προβλήματος που παρουσιάστηκε παραπάνω. Αναλύονται τα βασικά χαρακτηριστικά της, η αρχιτεκτονική της και η απόδοσή της σε 3 πειραματικές διατάξεις/εφαρμογές.

Στο Κεφάλαιο 3 παρουσιάζεται η υπηρεσιοστρεφής υλοποίηση που ενθυλακώνει τον αλγόριθμο του Κεφαλαίου 2 και αναλύεται η αρχιτεκτονική της, η ανάλυση της απόδοσής της και οι λεπτομέρειες υλοποίησης της.

Στο Κεφάλαιο 4 παρουσιάζεται η διασύνδεση του αλγορίθμου του Κεφαλαίου 2 με ένα αλγόριθμο πρόβλεψης χρονοσειράς για την συνδυασμένη χρήση τους και την περαιτέρω αυτοματοποίηση του πλαισίου.

Στο Κεφάλαιο 5 παρουσιάζεται η ανάλυση της αλληλεπίδρασης μεταξύ ταυτόχρονα εκτελούμενων εικονικών μηχανών στον ίδιο φυσικό πόρο και η μοντελοποίησή τους με βάση τις παραμέτρους μελέτης για την εκ των προτέρων πρόβλεψη αυτής της αλληλεπίδρασης.

Τέλος, στο Κεφάλαιο 6 περιλαμβάνεται η σύνοψη και τα συμπεράσματα που εξήχθησαν κατά την εκπόνηση της διδακτορικής διατριβής καθώς και η συνεισφορά και η καινοτομία που επιδεικνύει στον αντίστοιχο ερευνητικό χώρο. Επίσης συζητούνται θέματα μελλοντικής εργασίας και επέκτασης των ερευνητικών αποτελεσμάτων.

2 *Επιλογή Μεθοδολογίας Δημιουργίας Μοντέλων*

Στο κεφάλαιο αυτό περιγράφεται η βασική μεθοδολογία που επιλέχθηκε για τη δημιουργία των μοντέλων πρόβλεψης απόδοσης σε περιβάλλοντα Υπολογιστικών Νεφών και η οποία βασίζεται σε μηχανική μάθηση και συγκεκριμένα σε Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks-ANN). Η επιλογή αυτή βασίστηκε εν πολλοίς στους περιορισμούς που αναπτύχθηκαν στο προηγούμενο κεφάλαιο και με στόχο την ρεαλιστική χρησιμοποίηση της επιλεγμένης μεθόδου στα προαναφερθέντα περιβάλλοντα.

2.1 Εισαγωγή

Η προσέγγιση που επιλέχθηκε για την επίλυση αυτού του προβλήματος μεταφράσεων (των απαιτήσεων πελατών στις απαιτήσεις των πόρων) στο στρώμα PaaS παρουσιάζεται σε αυτό το κεφάλαιο η οποία συναντά τους προαναφερθέντες περιορισμούς. Η μέθοδός μας είναι βασισμένη στα τεχνητά νευρωνικά δίκτυα (ΤΝΔ) ή Artificial Neural Networks (ANNs). Η προτεινόμενη προσέγγιση

- λαμβάνει υπόψη ως predictors τις παραμέτρους φορτίου επιπέδου εφαρμογής μαζί με τις παραμέτρους χαρακτηρισμού της υπολογιστικής ικανότητας της πλατφόρμας εκτέλεσης, γεγονός που επιτρέπει την άμεση χαρτογράφηση της επιρροής τους στις εξόδους KPI που χρησιμοποιούνται για να εκφράσουν τα επίπεδα υπηρεσίας (QoS). Για κάθε τμήμα λογισμικού που προσφέρεται ως υπηρεσία, ένα ΤΝΔ δημιουργείται προκειμένου να απεικονιστεί αυτή η σχέση.
- αυτοματοποιεί και βελτιστοποιεί μια μέχρι τώρα χειρωνακτική και διαισθητική διαδικασία όπως τη φάση δημιουργίας και σχεδιασμού ΤΝΔ μέσω της χρήσης ενός γενετικού αλγορίθμου (ΓΑ-GA), σε ένα καινοτόμο υβριδικό συνδυασμό των δύο μεθοδολογιών. Πολυάριθμες παραλλαγές του προτεινόμενου αλγορίθμου δοκιμάζονται ανάλογα με την ύπαρξη ή όχι ενός ενδιάμεσου συνόλου επικύρωσης ή του τύπου επιστρεφόμενης μετρικής απόδοσης του ΓΑ και συγκρίνονται με μια τυχαία διαδικασία σχεδίου.
- ερευνά τα οφέλη του στατιστικού συνδυασμού περισσότερων του ενός μοντέλων ώστε να διαμορφώσει ένα πιό εύρωστο αποτέλεσμα με καλύτερες ικανότητες γενίκευσης
- επικυρώνει τη μέθοδο μέσω 3 πειραμάτων.

2.2 Σχετικές Εργασίες

Υπάρχουν διάφορες προσεγγίσεις για τους μηχανισμούς που στοχεύουν να υπολογίσουν τους πόρους που απαιτούνται για την εκτέλεση υπολογιστικών εφαρμογών. Αυτοί περιλαμβάνουν κυρίως τεχνικές όπως η αναλυτική μοντελοποίηση , η ανάλυση πηγαίου κώδικα και η πιθανοτική επεξεργασία των στατιστικών στοιχείων. Στις ακόλουθες παραγράφους παρουσιάζονται ενδεικτικές εργασίες από αυτούς τους τομείς και αναλύονται όσον αφορά τους περιορισμούς που αναφέρθηκαν στην εισαγωγή.

Η Network Weather Service (NWS) [22] μετρά τα φορτία πληροφοριών, επεξεργαστών και δικτύων των πόρων, και προβλέπει τους μελλοντικούς πόρους, ενώ άλλες εργασίες (π.χ. [23]) εστιάζουν στην ανάλυση του κώδικα πηγής μιας εφαρμογής προκειμένου να εκτελέσουν μια προσομοίωση και μια πρόβλεψη της απόδοσής του. Ένα πλαίσιο για QoS στις εφαρμογές πλέγματος συζητείται στο [9]. Σε αυτή την εργασία, χρησιμοποιούνται ένα μοντέλο υπολογισμού του χρόνου εκτέλεσης και ένα πρότυπο τιμολόγησης για τον καθορισμό της τιμής μιας εφαρμογής. Προκειμένου να καθοριστεί εάν οι περιορισμοί QoS του πελάτη μπορούν να εκπληρωθούν, για κάθε παράμετρο QoS ένα αντίστοιχο μοντέλο πρέπει να είναι σε ισχύ. Το Vienna Grid Environment (VGE) δεν ορίζει την πραγματική φύση των μοντέλων απόδοσης. Διευκρινίζει μια αφηρημένη διεπαφή για τα μοντέλα απόδοσης, παίρνοντάς τα ως έξοδο από την αναλυτική διαμόρφωση ή τα ιστορικά στοιχεία. Εντούτοις, η αναλυτική μοντελοποίηση στα πλαίσια των Υπολογιστικών Νεφών μπορεί να μην έχει διαθέσιμη λεπτομερή γνώση της εφαρμογής προκειμένου να προκύψουν οι εξισώσεις που απεικονίζουν την απόδοσή της (Περιορισμός 2).

Η βιβλιογραφία [10] προτείνει επίσης μια ανάλυση δύο σταδίων για τις multi-tier εφαρμογές προκειμένου να αποσυντεθεί το SLA στους στόχους χαμηλού επιπέδου. Σε πρώτη φάση, ένα σχέδιο εκτίμησης απαίτησης των πόρων εφαρμόζεται μέσω μιας μεθόδου σκιαγράφησης του προφίλ της εφαρμογής. Η σκιαγράφηση δημιουργεί τα λεπτομερή προφίλ κάθε συστατικού στην εφαρμογή. Ένα προφίλ συλλαμβάνει την απαίτηση κάθε τύπου συναλλαγής σε σχέση με τους φυσικούς πόρους, μέσω των ιστορικών στοιχείων ή της αξιολόγησης. Στο επόμενο βήμα, χρησιμοποιούνται αναλυτικά μοντέλα, με το προαναφερθέν μειονέκτημα.

Μια άλλη ενδιαφέρουσα εργασία παρουσιάζεται στο [11]. Η εφαρμογή, της οποίας οι ανάγκες πρέπει να μετρηθούν, εκτελείται κάτω από μια συγκεκριμένη δέσμευση φυσικών πόρων προκειμένου να καθοριστεί εάν η δεδομένη διαμόρφωση ικανοποιεί τους χρονικούς περιορισμούς. Εάν αυτό δεν ισχύει, αυτές οι παραμέτροι δέσμευσης αλλάζουν αναλόγως. Εάν υπάρχει ένα θετικό πλεόνασμα, οι πόροι μειώνονται και εάν είναι αρνητικό αυξάνονται έως ότου επιτυγχάνεται ένα ικανοποιητικό περιθώριο ασφάλειας. Αυτή η μέθοδος επιτυγχάνει υψηλά ποσοστά χρησιμοποίησης των πόρων, έναντι του κόστους δοκιμαστικών εκτελέσεων για κάθε εκτέλεση με διαφορετικές παραμέτρους επιπέδου εφαρμογής. Στην περίπτωση που εξετάζουμε, κάθε SLA έχει διαφορετικό σετ παραμέτρων. Άρα αυτή η διαδικασία πρέπει να επαναλαμβάνεται συνέχεια, για κάθε διαφορετικό instance της υπηρεσίας.

Μία ελπιδοφόρος μέθοδος που αναλύει τον πηγαίο κώδικα επιτρέποντας την προσομοίωση της απόδοσης συστημάτων περιγράφεται στο [8]. Η μέθοδος αυτή δημιουργεί ένα μοντέλο της εφαρμογής με έναν οδηγούμενο από παραμέτρους Conditional Data Flow Graph (CDFG) και την αρχιτεκτονική υλικού (HW) από έναν διαμορφώσιμο γράφο HW. Η καθυστέρηση εκτέλεσης για κάθε block στον γράφο εφαρμογής CDFG διαμορφώνεται από τις παραμέτρους εισόδου της εφαρμογής, γεγονός που επιτρέπει την ιδιαίτερα ευέλικτη εκτίμηση εκτέλεσης. Ο προσομοιωτής παίρνει τους γράφους CDFG και HW ως είσοδο και εκτελεί την προσομοίωση σε ένα χαμηλό επίπεδο για να προσδιορίσει τις λεπτομερείς δραστηριότητες HW. Το κύριο μειονέκτημα είναι ότι χρειάζεται τον πηγαίο κώδικα προκειμένου να παραχθεί το CDFG (παραβίαση του Περιορισμού 2), κατά συνέπεια δεν μπορεί να εφαρμοστεί στα υπηρεσιοστρεφή περιβάλλοντα με σαφείς διαχωρισμούς οντοτήτων όπως τα Νέφη.

Οι συγγραφείς του [12] προτείνουν ένα σχέδιο που χρησιμοποιεί πληροφορίες για προ-υπολογισμένες αλγοριθμικές πολυπλοκότητες ή προηγούμενες επιχειρησιακές συναλλαγές. Μια βάση δεδομένων γνώσης αποθηκεύει τις πληροφορίες για τις απαιτήσεις συστημάτων για τις διαφορετικές πολυπλοκότητες, ανάλογα με την εμπειρία που αποκτάται από το τρέξιμο παρόμοιων εργασιών στο παρελθόν. Η βάση δεδομένων πολυπλοκότητας αποθηκεύει τις προηγούμενες υπολογισμένες πολυπλοκότητες των εργασιών. Επιπλέον περιορίζει σημαντικά το ποσό δοκιμής για τις περιπτώσεις νέων εφαρμογών αλλά με γνωστή πολυπλοκότητα. Η βασισμένη στην πολυπλοκότητα εκτίμηση εκτέλεσης (δηλαδή σε μία μετρική όπως ο χρόνος εκτέλεσης χειρότερης περίπτωσης-WCET) είναι ιδιαίτερα κρίσιμη για τα συστήματα πραγματικού χρόνου, εντούτοις σε περισσότερο γενικά συστήματα μπορεί να οδηγήσει σε σημαντική υπερδέσμευση πόρων (Περιορισμός 1).

Η χρησιμοποίηση του πηγαίου κώδικα για την ανάλυση απόδοσης μιας εφαρμογής προτείνεται επίσης στο [13] στο οποίο παρουσιάζεται το εργαλείο PACE (Performance Analysis and Characterization Environment). Το σύστημα χαρακτηρίζει την εφαρμογή και το υλικό στο οποίο η εφαρμογή πρόκειται να εκτελεστεί, και συνδυάζει τα παραγόμενα μοντέλα για να παράγει τη πρόβλεψη εκτέλεσης. Η στατική ανάλυση πηγαίου κώδικα αποτελεί τη βάση αυτής της διαδικασίας, επικεντρώνοντας στη ροή ελέγχου της εφαρμογής, τη συχνότητα εκτέλεσης των διαφόρων διαδικασιών και τη δομή της επικοινωνίας. Από αυτά, προκύπτει ένα μοντέλο για την εφαρμογή. Μία λεπτομερής εργασία παρουσιάζεται στο [41], όπου εφαρμογές παράλληλης επεξεργασίας πρωτοκόλλου MPI (Message Passing Interface) ελέγχονται προκειμένου να προβλεφθεί αυτόματα πού θα εμφανιστεί bottleneck. Η ανάγκη για τον κώδικα πηγής εμποδίζει την

εφαρμογή της σε Clouds λόγω του Περιορισμού 1. Στο [7] μια προσέγγιση παρουσιάζεται για τη διαχείριση της ποιότητας υπηρεσίας QoS σε υπηρεσιοστρεφείς υποδομές που εστιάζει στην κατηγοριοποίηση των ποιοτικών χαρακτηριστικών και χρησιμοποιεί μια συγκεκριμένη XML-based γλώσσα για τις εφαρμογές και τους φορείς παροχής υπηρεσιών για να εκφράσει τις απαιτήσεις QoS.

Μια πιθανοτική προσέγγιση εμφανίζεται στο [14]. Σε αυτήν την περίπτωση οι συγγραφείς χρησιμοποιούν μια κατανομή πιθανότητας των προγενέστερων εκτελέσεων προκειμένου να καταλήξουν σε ένα ποσό πόρων που ικανοποιεί ένα ποσοστό των ιστορικών στοιχείων (π.χ. 95%). Αυτή η προσέγγιση πετυχαίνει στην επίτευξη ενός χαμηλού ποσοστού παραβιάσεων SLAs εντούτοις το ποσό υπερδέσμευσης πόρων εξαρτάται πάρα πολύ από τον τύπο της κατανομής και ειδικά από την απόκλιση των χρόνων εκτέλεσης. Όταν υπάρχει μια ανάγκη να καλυφθεί ένα συγκεκριμένο ποσοστό (π.χ. 95%) των προηγούμενων περιπτώσεων, οι πόροι που πρέπει να χρησιμοποιηθούν πρέπει να καλύπτουν τη χειρότερη περίπτωση μέσα σε αυτό το ποσοστό, ακόμη και για τις περιπτώσεις που έχουν σημαντικά λιγότερες απαιτήσεις (παραβίαση του Περιορισμού 1). Η εργασία μας χρησιμοποιεί επίσης τα ιστορικά στοιχεία, όμως η κύρια εστίασή της είναι να χρησιμοποιήσει αυτά τα στοιχεία προκειμένου να γίνουν κατανοητές οι σύνθετες σχέσεις που υπάρχουν μεταξύ των παραγόντων που επηρεάζουν τις ανάγκες των πόρων μιας εφαρμογής προκειμένου να επιτευχθεί ένα επιθυμητό επίπεδο QoS. Ο συσχετισμός αυτών των στοιχείων βοηθά στην παραγωγή μαθηματικών τύπων που μπορούν να χρησιμοποιηθούν άμεσα με τις αναγκαίες εισόδους, όπως είναι απαραίτητο από τον Περιορισμό 1.

Μια προσέγγιση μηχανικής μάθησης προκειμένου να προβλεφθεί ο χρόνος εκτέλεσης λογισμικού παρουσιάζεται στο [15]. Αυτή είναι η πλησιέστερη προς την προσέγγισή μας. Η κύρια διαφορά βρίσκεται στο γεγονός ότι στην παρούσα εργασία εφαρμόζεται μία μέθοδος αυτοματοποίησης και βελτιστοποίησης της δημιουργίας των μοντέλων μηχανικής μάθησης. Συνεπώς είναι και κατάλληλη για τα αυτοματοποιημένα πλαίσια υπηρεσιοστρεφούς λειτουργίας. Μια πολύ λεπτομερής ανάλυση των διάφορων τεχνικών απόδοσης ερευνάται στο [40]. Σύμφωνα με αυτήν, για έναν υπηρεσιοστρεφή τομέα, οι προσεγγίσεις βασισμένες σε ANNs (συγκεκριμένα νευρο-ασαφείς λύσεις) είναι μεταξύ των καλύτερων υποψηφίων για τέτοια περιβάλλοντα.

2.3 Σχέση των ΤΝΔ με τους Περιορισμούς

Όπως έγινε αντιληπτό στα προηγούμενα υποκεφάλαια, για ένα πλαίσιο πρόβλεψης απόδοσης στον τομέα των Υπολογιστικών Νεφών, υπάρχουν σημαντικοί περιορισμοί, οι οποίοι πρέπει να ληφθούν υπό εξέταση στην προτεινόμενη λύση. Σε αυτό το υποκεφάλαιο, το προτεινόμενο σχέδιο λύσης περιγράφεται έναντι κάθε ενός από τους περιορισμούς του Κεφαλαίου 1. Κατόπιν το γενικό πλαίσιο που παράγεται από αυτές τις έννοιες σχεδίου παρουσιάζεται, σε μια από κάτω προς τα επάνω προσέγγιση.

2.3.1 Καθορισμός των εισόδων και των εξόδων των μοντέλων

Όπως είδαμε από το Κεφάλαιο 1, τα μοντέλα πρόβλεψης απόδοσης πρέπει να είναι σε θέση να διασυνδέουν τις παραμέτρους του Συμβολαίου Επιπέδου Υπηρεσίας (SLA) με τους απαιτούμενους πόρους. Στο στρώμα PaaS, διάφορα τμήματα λογισμικού θα πρέπει να μοντελοποιηθούν. Για κάθε ένα από αυτά, ο αντίστοιχος υπεύθυνος για την ανάπτυξή του (ρόλος υπεύθυνων για την ανάπτυξη εφαρμογής στο στρώμα SaaS) δημιουργεί μια

περιγραφή XML του λογισμικού. Αυτή η διαδικασία περιγράφεται λεπτομερώς στο [30]. Η κύρια εκτίμηση σε αυτήν την φάση είναι να πληροφορηθεί το στρώμα PaaS ποιες είναι οι κρίσιμες παράμετροι που πρέπει να μοντελοποιηθούν σε ένα σχέδιο πρόβλεψης απόδοσης. Αυτές είναι οι παράμετροι επιπέδου εφαρμογής που μπορούν να διαμορφωθούν από τον πελάτη σε ένα SLA (όπως ο αριθμός χρηστών σε elearning εφαρμογή, ανάλυση ή βάθος χρώματος της εικόνας μιας πολυμεσικής εφαρμογής κ.λπ.). Αυτές οι διαμορφώσιμες παράμετροι αντιπροσωπεύουν επίσης έναν διαφορετικό φόρτο εργασίας που τίθεται από την εφαρμογή στους πόρους. Η υψηλότερη ανάλυση σημαίνει παραδείγματος χάριν ότι απαιτείται περισσότερη επεξεργασία από την CPU. Η επιλογή από τον πελάτη μιας από πολλές πιθανές παραμέτρους για κάθε συγκεκριμένη εκτέλεση μπορεί να οδηγήσει σε διαφορετικές ανάγκες για τους υπολογιστικούς πόρους.

Επιπλέον, ο υπεύθυνος για την ανάπτυξη της εφαρμογής πρέπει να διευκρινίσει ποιο είναι το KPI (η παράμετρος ποιότητας υπηρεσίας) του λογισμικού. Με βάση αυτήν την απόδοση, ο πελάτης θα υπαγορεύσει επίσης το επίπεδο επιθυμητού QoS και το στρώμα PaaS θα είναι σε θέση να ξέρει εάν αυτός ο στόχος εκπληρώνεται ή όχι.

Με τη χρησιμοποίηση αυτών των πληροφοριών που προέρχονται από το στρώμα SaaS, οι οποίες είναι τόσο γενικές που δεν επηρεάζουν τα επίπεδα εμπιστευτικότητας μεταξύ των στρωμάτων καθώς αυτή η πληροφορία υπάρχει ούτως η άλλως στο SLA, το πλαίσιο που περιγράφεται σε αυτό το κεφάλαιο θα πρέπει να είναι σε θέση να προσδιορίσει τις παραμέτρους του μοντέλου για κάθε εφαρμογή. Συνεπώς η επιλεγόμενη διαδικασία μοντελοποίησης θα πρέπει να είναι ευέλικτη ώστε να μπορεί να προσαρμόζει διαφορετικές εισόδους και εξόδους για κάθε εφαρμογή.

Τέλος, μια παράμετρος του μοντέλου πρέπει να είναι μια μετρική του υλικού, η οποία δείχνει την υπολογιστική δυνατότητα των πόρων. Αυτό γενικά υπαγορεύεται από τον προμηθευτή IaaS και συμφωνείται με το στρώμα PaaS εκ των προτέρων ως μετρική για τη διαπραγμάτευση. Μπορεί να είναι παραδείγματος χάριν η συχνότητα ενός επεξεργαστή, το μέγεθος RAM, κάποια συγκεκριμένη συγκριτική μέτρηση επιδόσεων (benchmark) κ.λπ. Οι μετρικές υλικού που χρησιμοποιούνται σε αυτήν την εργασία είναι γενικές. Αυτός ο τομέας είναι πολύ εκτεταμένος και ένας μεγάλος αριθμός μετρικών έχει προταθεί, όπως το MIPS, FLOPS κ.λπ., χωρίς να υπάρχει κατάληξη ως προς το καταλληλότερο. Λόγω της γενικής φύσης της προσέγγισης που ακολουθήθηκε σε αυτή την εργασία οι μετρικές μπορούν να αντικατασταθούν οποιαδήποτε στιγμή προκειμένου να ενσωματωθούν οι πρόοδοι στον αντίστοιχο τομέα.

Από τις ανωτέρω παραμέτρους που διευκρινίζονται, η εφαρμογή και οι παράμετροι επιπέδων υλικού θεωρούνται ως είσοδοι στο πρότυπο, ενώ το KPI θεωρείται ως έξοδος. Οι παράμετροι υλικού δεν περιλαμβάνονται κατευθείαν στην έξοδο για να είναι δυνατή η επικύρωση των μεθόδων αλλά και διότι μπορεί να υπάρχει κάποιο ανώτερο στρώμα βελτιστοποίησης το οποίο να δοκιμάζει διαφορετικούς συνδυασμούς υλικού ώστε να προσφέρει ελαφρώς χειρότερα επίπεδα ποιότητας υπηρεσίας για αρκετά χαμηλότερο κόστος.

2.3.2 Έλλειψη πηγαίου κώδικα και εκτεταμένων πληροφοριών για την εφαρμογή

Προκειμένου να μετριαστεί το πρόβλημα των ανεπαρκών πληροφοριών σχετικά με την εσωτερική δομή της εφαρμογής απαιτείται μια μέθοδος εκτίμησης μαύρου κουτιού, που θα είναι σε θέση να συλλάβει τις εξαρτήσεις των εξόδων του μοντέλου από τις εισόδους,

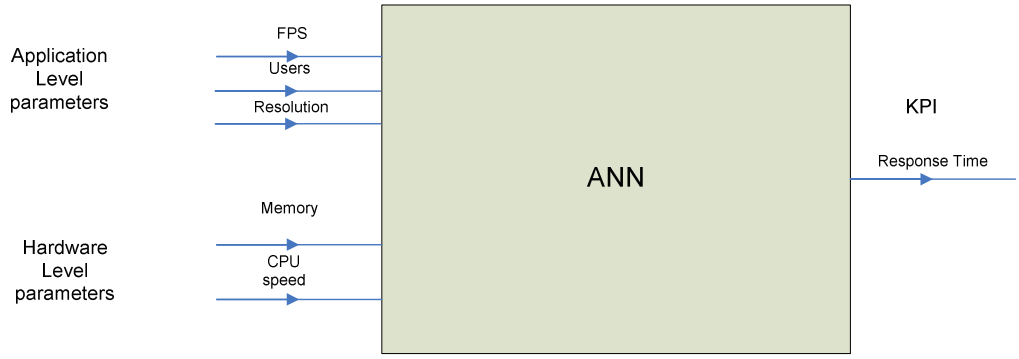
μέσω της χρησιμοποίησης μόνο των διαθέσιμων στοιχείων από προηγούμενες εκτελέσεις ή από μια περίοδο δοκιμών (benchmarking). Μία τέτοια μέθοδος είναι τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ) ή Artificial Neural Networks (ANNs). Γενικά, ένα ANN συσχετίζει την είσοδο με τα δεδομένα εξόδου προκειμένου να βρεθεί μια γενική προσέγγιση της λειτουργίας που περιγράφει την εξάρτηση της εξόδου από την είσοδο, αγνοώντας τοπικούς ή τμηματικούς παράγοντες αλλά μάλλον χρησιμοποιώντας μια ολιστική προσέγγιση. Έχοντας μετέπειτα την αλγεβρική εξίσωση που προκύπτει, είναι δυνατόν να εφαρμοστούν είσοδοι για τις οποίες δεν υπάρχουν διαθέσιμα στοιχεία και να προσδιοριστεί μία πρόβλεψη για την έξοδο. Έτσι το γενικό μοντέλο έχει τη μορφή που φαίνεται στο Σχήμα 3.

Μέσω της χρήσης τέτοιων αλγεβρικών κανόνων, το στρώμα PaaS είναι σε θέση να αναγνωρίσει την επίδραση μιας επιλογής για τους πόρους στο KPI ανά συγκεκριμένη εκτέλεση μιας εφαρμογής (δηλαδή ανά ξεχωριστό SLA), να ερευνήσει τις διαφορετικές διαμορφώσεις και να αποφασίσει για τη βέλτιστη ανταλλαγή μεταξύ των παρεχόμενων πόρων και του γενικού αποτελέσματος, βασισμένου στα επιχειρησιακά πρότυπα που ακολουθεί. Αυτό ικανοποιεί τους Περιορισμούς 1 και 2.

2.3.3 Γραμμικά και μη γραμμικά χαρακτηριστικά

Τα ANNs αποτελούνται από διάφορους νευρώνες που διασυνδέονται με τη χρήση βαρών σε μια δομή δικτύων. Αυτοί οι νευρώνες αντιπροσωπεύουν τις βασικές συναρτήσεις μεταφοράς που μπορούν να είναι γραμμικές ή μη γραμμικές, βηματικές κ.λπ. Μέσω ενός συνδυασμού τέτοιων βασικών συναρτήσεων μεταφοράς καθώς επίσης και των απαραίτητων βαρών που δίνονται στις συνδέσεις τους, μπορεί να δημιουργηθεί ένα μοντέλο δικτύου που απεικονίζει τις παραλλαγές στην απόδοση υπηρεσιών σε σχέση με

τους αποδιδόμενους πόρους και τις παραμέτρους εφαρμογής. Μια πιό λεπτομερής περιγραφή των συναρτήσεων μεταφοράς ANN είναι διαθέσιμη στο [44]. Μέσω αυτού του συνδυασμού, ο περιορισμός 3 ικανοποιείται.



Σχήμα 3: Είσοδοι και Έξοδοι για το μοντέλο ANN

2.3.4 Αυτοματοποιημένη και βελτιστοποιημένη διαδικασία δημιουργίας μοντέλων

Τα ANNs, παρά τα πλεονεκτήματά τους, έχουν επίσης και μειονεκτήματα.Τα βασικότερα από αυτά είναι τα ακόλουθα:

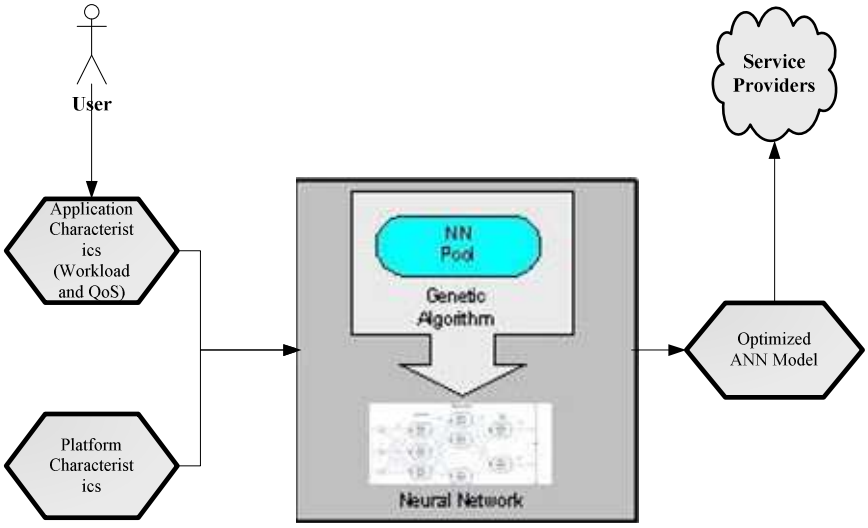
- Ανάγκη ενός καλού συνόλου δεδομένων εκπαίδευσης με χαμηλό θόρυβο
- Διαισθητικές επιλογές αρχιτεκτονικής
- Έλλειψη γενίκευσης

Προκειμένου να μετριαστεί το πρώτο μειονέκτημα, πραγματοποιείται μια ενδεικτική σειρά εκτελέσεων δοκιμής για τους διάφορους συνδυασμούς εισόδου και τα αποτελέσματα της εξόδου προωθούνται στο ANN, δημιουργώντας κατά συνέπεια το σύνολο εκπαίδευσης. Αυτή η διαδικασία περιγράφεται λεπτομερώς στο [36]. Η χρήση virtualization και χρονοδρομολογητή πραγματικού χρόνου βοηθά στη μείωση του θορύβου στα δείγματα. Η ανατροφοδότηση από αυτόν τον μηχανισμό σχετικά με τις

Η σελίδα αυτή είναι σκόπιμα λευκή

περιοχές με το μεγαλύτερο λάθος που χρειάζονται την περαιτέρω δειγματοληψία θα διερευνηθεί στο μέλλον. Εναλλακτικά, μπορούν να χρησιμοποιηθούν τα διαθέσιμα ιστορικά στοιχεία.

Ο περιορισμός των δύο άλλων μειονεκτημάτων περιγράφεται λεπτομερώς στην Παράγραφο 2.5. Εν περιλήψει, προκειμένου να αποφασιστούν οι κρίσιμες παράμετροι της αρχιτεκτονικής ενός ANN (όπως ο αριθμός στρωμάτων, συναρτήσεων μεταφοράς ανά στρώμα ή ο αριθμός νευρώνων ανά στρώμα), απαιτείται ένας εμπειρογνώμονας που θα διαθέσει τον απαραίτητο χρόνο να εξετάσει διαφορετικές διαμορφώσεις. Στην παρούσα προσέγγιση χρησιμοποιείται ένας γενετικός αλγόριθμος (GA [26], Σχήμα 4) που έχει σαν στόχο τη βελτιστοποίηση της δικτυακής αρχιτεκτονικής και της ικανότητας γενίκευσης.



Σχήμα 4: Βελτιστοποίηση ANN μοντέλου μέσω ΓΑ

Προκειμένου να μετριαστεί η έλλειψη γενίκευσης, ερευνάται ένα παραπάνω βήμα επεξεργασίας. Αυτό χρησιμοποιεί συνδυασμό διαφορετικών μοντέλων, έτσι ώστε η γενική απόδοσή τους να είναι πιό εύρωστη, και βασίζεται σε διαφορετικά στατιστικά

Η σελίδα αυτή είναι σκόπιμα λευκή

χαρακτηριστικά των μοντέλων. Λαμβάνονται υπόψη διάφορες προσεγγίσεις, όπως η λήψη της μέσης ή μεσαίας εκτίμησης από τα καλύτερα εκπαιδευμένα και γενικά δίκτυα, οι προσεγγίσεις σε ένα παράθυρο προκειμένου να καθοριστούν τα συμπληρωματικότερα δίκτυα, αφαίρεση ακραίων εκτιμήσεων (outlier removal) και εισαγωγή ενδιάμεσων βημάτων γενίκευσης ως μέτρηση της επιτυχίας του δικτύου. Μέσω αυτής της προσέγγισης ικανοποιείται ο περιορισμός 4.

2.3.5 Υπηρεσιοστρεφής υλοποίηση

Οποιαδήποτε προσέγγιση πρόβλεψης απόδοσης μέσα σε ένα υπηρεσιοστρεφές περιβάλλον πρέπει να προσφέρεται ως υπηρεσία μέσω μιας κατάλληλης υλοποίησης. Αυτό αντιμετωπίζεται σε αυτήν την εργασία μέσω της υλοποίησης [31] που περιγράφεται στο Κεφάλαιο 3 .

Αυτή η υπηρεσία έχει την ευθύνη να βρει από τις περιγραφές XML κάθε τμήματος λογισμικού τις απαραίτητες πληροφορίες όπως ο αριθμός εισόδων και εξόδων για το μοντέλο και το σύνολο δεδομένων από την βάση δεδομένων ιστορικών στοιχείων. Κατόπιν αναμεταδίδει αυτές τις πληροφορίες σε ένα στρώμα (βασισμένο σε ένα γενικό εργαλείο όπως το Octave ή το Matlab) που εφαρμόζει τον επιλεγμένο αλγόριθμο (ΓΑ-βελτιστοποιημένο ΤΝΔ) για την δημιουργία του μοντέλου ως επισυναπτόμενο εκτελέσιμο αρχείο (pluggable script). Τέλος, αυτό το μοντέλο αποθηκεύεται για τη μελλοντική χρήση στις προβλέψεις. Μέσω αυτής της υλοποίησης ικανοποιείται ο περιορισμός 5.

2.4 Μεθοδολογία Πλατφόρμας IRMOS

Για να αποκτηθούν οι απαραίτητες πληροφορίες που χρειάζονται και αναφέρθηκαν στα προηγούμενα κεφάλαια, η πλατφόρμα IRMOS χρησιμοποιήθηκε, στην διαδικασία σχεδίασης της οποίας συνέβαλε και η παρούσα διατριβή. Στις επόμενες παραγράφους αναλύονται περιληπτικά οι φάσεις αυτής της διαδικασίας (Σχήμα 5), που αναλύονται στη φάση περιγραφής της εφαρμογής από τον application developer μέσω των εργαλείων που παρέχονται από την πλατφόρμα και στην φάση δειγματοληπτικής εκτέλεσής της (benchmarking).

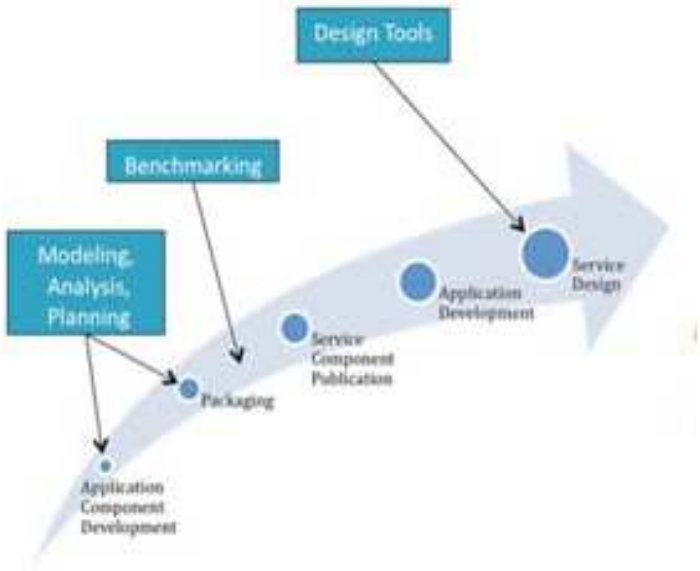
2.4.1 Φάση Περιγραφής-Δημιουργίας ASCD

Κατά τη διάρκεια αυτής της φάσης ο υπεύθυνος εφαρμογής (Application Developer) καλείται να δημιουργήσει την περιγραφή των επί μέρους συστατικών της (components) μέσω ενός εξειδικευμένου εργαλείου που παρέχεται από την πλατφόρμα ([117]). Το εργαλείο αυτό (Σχήμα 6) περιέχει το πρότυπο/υπόδειγμα (template) πληροφορίας που πρέπει να περιγραφεί και περιέχει όλους τους τύπους πληροφορίας (π.χ. παράμετροι ποιότητας υπηρεσίας, παράμετροι SLA) καθώς και τις δυνατές μορφές τους (ακέραιος αριθμός και το εύρος τιμών, διακριτό μέγεθος και οι δυνατές τιμές του κλπ.) . Έπειτα, το μοντέλο αυτό μετατρέπεται σε μορφή XML, και αποθηκεύεται στον PaaS πάροχο για χρήση από τις υπόλοιπες υπηρεσίες μοντελοποίησης, συμπεριλαμβανομένης και της υπηρεσίας πρόβλεψης απόδοσης που περιγράφεται στο Κεφάλαιο 3.

2.4.2 Φάση Benchmarking/ Application Sampling

Ο σκοπός αυτής της φάσης είναι η συντονισμένη εκτέλεση της εφαρμογής για διαφορετικούς συνδυασμούς παραμέτρων φόρτου εργασίας και υλισμικού. Κατά τη

διάρκεια αυτής, τα αποτελέσματα (επίπεδα παραμέτρων KPI) καταχωρούνται ώστε να χρησιμοποιηθεί το σύνολο δεδομένων για την εκπαίδευση των μοντέλων. Το βασικό σημείο προσοχής σε αυτή την περίπτωση είναι η επιλογή των συνδυασμών ώστε να αναπαριστούν διαφορετικές περιοχές του πεδίου έρευνας. Επιπλέον είναι απαραίτητο να γίνει δειγματοληψία για τα άκρα των μελετούμενων διαστημάτων, ώστε να εξασφαλιστεί ότι τα μοντέλα θα χρησιμοποιούνται για παρεμβολή (interpolation) και όχι για παρέκταση (extrapolation), χαρακτηριστικό που ενισχύει την ακρίβεια του συστήματος πρόβλεψης. .



Σχήμα 5: Φάσεις Προετοιμασίας Εφαρμογής Πλατφόρμας IRMOS

Επιπλέον, η ευρωστία ενός ANN μοντέλου εξαρτάται από τη σχέση μεταξύ του αριθμού διαθέσιμων σημείων για την εκπαίδευση και των παραμέτρων του δικτύου (βάρη και πολώσεις) που πρέπει να υπολογιστούν. Εάν το πλήθος του συνόλου εκπαίδευσης είναι υψηλότερο από τον αριθμό παραμέτρων του δικτύου που πρέπει να υπολογιστούν κατά τη διάρκεια της διαδικασίας, αυτό σημαίνει ότι κάθε μεταβλητή

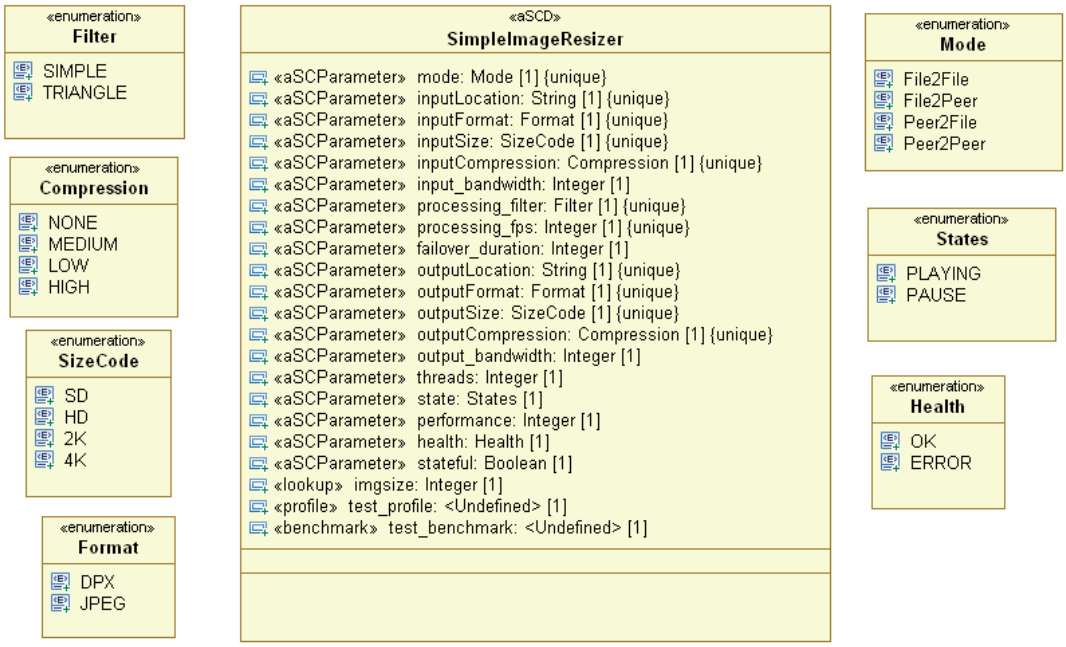
Η σελίδα αυτή είναι σκόπιμα λευκή

υπολογίζεται με τη λήψη του μέσου όρου από περισσότερα του ενός σημεία. Κατά συνέπεια, κανένα σημείο (που μπορεί να επηρεαστεί από τον τυχαίο θόρυβο κατά τη διάρκεια της συλλογής του συνόλου) δεν μπορεί να έχει σημαντικές επιπτώσεις στις προκύπτουσες τιμές. Λεπτομερώς, για ένα ANN με 3 στρώματα, με N εισόδους, K αριθμό νευρώνων στο κρυμμένο στρώμα και L εξόδους ο αριθμός άγνωστων μεταβλητών του δικτύου δίνεται από την Εξίσωση 1.

$$V = 2 * N + (K * N + K) + (L * K + L)$$

Εξίσωση 1: Πλήθος μεταβλητών ANN

Με βάση αυτή, μπορούμε να καθορίσουμε τον αριθμό των απαιτούμενων δειγμάτων για το μέγιστο μέγεθος δικτύου ή αντίστροφα, να καθορίσουμε το τελευταίο με βάση τα διαθέσιμα δείγματα, αν αυτά είναι δύσκολο ή χρονοβόρο να επεκταθούν.



Σχήμα 6: Παράδειγμα περιγραφής συστατικού εφαρμογής μέσω του πλαισίου IRMOS [116]

Η σελίδα αυτή είναι σκόπιμα λευκή

2.5 Εξελικτική Βελτιστοποίηση και Αυτοματοποίηση Διαδικασίας Παραγωγής Μοντέλων

Η αρχιτεκτονική και η δημιουργία ANN είναι μια διαδικασία που είναι βασισμένη στο ανθρώπινο ένστικτο και την εμπειρία, περιλαμβάνοντας εκτεταμένες δοκιμές. Σε ένα υπηρεσιοστρεφές πλαίσιο όμως η ανθρώπινη παρέμβαση πρέπει να διατηρείται στο ελάχιστο. Για να αφαιρεθεί ο ανθρώπινος παράγοντας από αυτήν την διαδικασία (και για λόγους αυτοματοποίησης και βελτιστοποίησης), ένας Γενετικός Αλγόριθμος (GA) χρησιμοποιείται προκειμένου να ληφθούν οι αποφάσεις σχετικά με τις παραμέτρους του ANN. Αυτό είναι μια καινοτόμος προσέγγιση καθώς μέχρι τώρα, οι ΓΑ έχουν χρησιμοποιηθεί για την βελτιστοποίηση ANN, αλλά όχι σε αυτή τη δυναμική κλίμακα και μορφή.

Η GA-βασισμένη βελτιστοποίηση ANNs έχει χρησιμοποιηθεί στο παρελθόν σε διάφορες σημαντικές εργασίες. Παραδείγματος χάριν στο [33], διάφορες παράμετροι του δικτύου (όπως τα βάρη σύνδεσης) επιλέγονται μέσω του ΓΑ. Προκειμένου να εφαρμοστεί ένας τέτοιος συνδυασμός απαιτείται μια στατική δομή για το δίκτυο. Σε άλλες περιπτώσεις [34] χρησιμοποιείται για την έμμεση βελτιστοποίηση, μέσω παραδείγματος χάριν της βελτιστοποίησης της διαδικασίας εκπαίδευσης. Στο [37] παρουσιάζεται μια άλλη προσέγγιση που συνδυάζει ΓΑ και ΤΝΔ, εντούτοις η δομή του δικτύου περιορίζεται σε δύο κρυμμένα στρώματα και στατικές συναρτήσεις μεταφοράς ενώ οι μεταβαλλόμενες παράμετροι περιορίζονται στον αριθμό κόμβων και τις συνδέσεις τους. Το ίδιο ισχύει για το [38]. Στην παρούσα εργασία, η δομή του ΤΝΔ μπορεί να είναι δυναμική. Ο αριθμός στρωμάτων και οι νευρώνες ανά στρώμα αποφασίζονται μέσω της διαδικασίας βελτιστοποίησης, χωρίς την ανάγκη να υπάρξει μια

προκαθορισμένη δομή. Επιπλέον κάθε στρώμα μπορεί να έχει διαφορετική συνάρτηση μεταφοράς, άλλη μία παράμετρος που εξετάζεται.

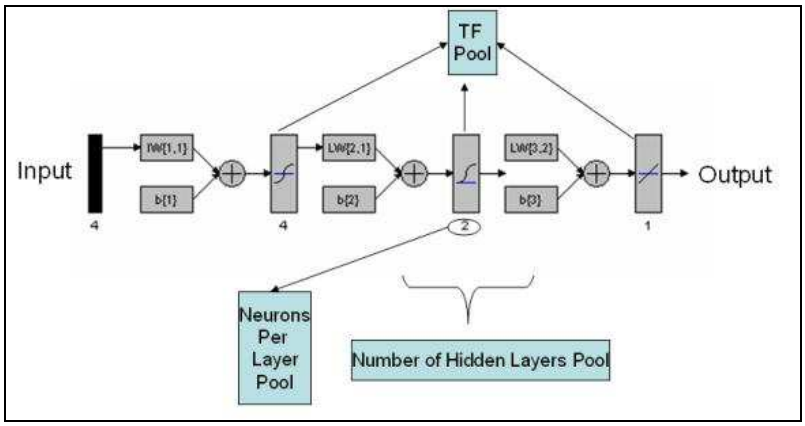
Όσον αφορά τη γενική βελτιστοποίηση ΤΝΔ, υπάρχουν σημαντικές εργασίες που στοχεύουν στον προσδιορισμό του βέλτιστου μεγέθους του δικτύου, από άποψη νευρώνων που χρησιμοποιούνται. Ο πιό διαδεδομένος είναι η Μπεϋζιανή συστηματοποίηση (Bayesian regularization)[16]. Σε αυτήν την μέθοδο, κατά τη διαδικασία εκπαίδευσης δεν λαμβάνεται υπόψη μόνο το λάθος αλλά και οι παράμετροι της συνάρτησης μεταφοράς. Αν και αυτή η μέθοδος φαίνεται να λειτουργεί καλά για διάφορες περιπτώσεις, έχουν υπάρξει υπόνοιες [17] ότι η συμμετοχή τους αμβλύνει τη συμπεριφορά του δικτύου. Ενώ αυτό είναι επιθυμητό σε διάφορες εφαρμογές, στην προσέγγιση λειτουργίας όπως αυτή η περίπτωση τα αποτελέσματα είναι αρνητικά ειδικά σε περιπτώσεις όπου ερευνώνται απότομες ανωμαλίες στη συμπεριφορά. Επιπλέον, όπως προτείνεται στο [18], οι αρχικές παράμετροι που δίνονται είναι περισσότερο διαισθητικής φύσης. Αντίθετα, στην παρούσα υλοποίηση λαμβάνονονται υπόψη διαφορετικές συναρτήσεις μεταφοράς όσον αφορά την αποδοτικότητα των συνδυασμών τους.

2.5.1 Κωδικοποίηση προβλήματος βελτιστοποίησης ΤΝΔ από ΓΑ

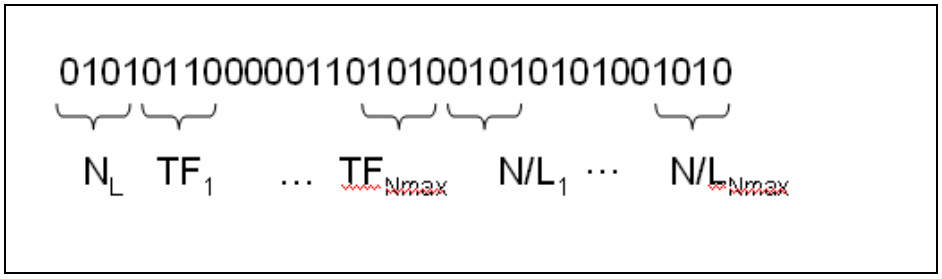
Το πρώτο βήμα προκειμένου να εφαρμοστεί ένας ΓΑ σε οποιοδήποτε πρόβλημα βελτιστοποίησης είναι να μετασχηματιστούν οι παράμετροι που ερευνώνται σε ένα σχήμα κατάλληλο για χειρισμό από τον ΓΑ, δηλαδή σε μια μορφή χρωμοσώματος. Κάθε λύση υποψηφίων κωδικοποιείται όπως ένα χρωμόσωμα με ένα bitstring που αντιπροσωπεύει τις πιθανές τιμές για τις παραμέτρους εισόδου της συνάρτησης που βελτιστοποιείται. Κάθε μέρος της σειράς αφιερώνεται σε μια παράμετρο. Κατόπιν οι

τελεστές του ΓΑ (μετάλλαξη, ελιτισμός, διασταύρωση) αξιολογούν την ικανότητα κάθε λύσης και αλλάζουν τα μέλη του πληθυσμού προκειμένου να βρεθεί ένας καλύτερος συνδυασμός.

Οι παράμετροι του δικτύου που κωδικοποιήθηκαν στο χρωμόσωμα εμφανίζονται στο Σχήμα 7. Αυτοί ήταν ο αριθμός κρυμμένων στρωμάτων, οι συναρτήσεις μεταφοράς για κάθε στρώμα και οι αριθμοί νευρώνων για κάθε στρώμα. Οι λειτουργίες μεταφοράς επιλέχτηκαν από μια ομάδα από γραμμικές και μη γραμμικές λειτουργίες που έχουν διαφορετικά χαρακτηριστικά ενώ τα βάρη σύνδεσης (connection weights) και οι πολώσεις (biases) του δικτύου καθορίστηκαν κατά τη διάρκεια της διαδικασίας εκπαίδευσης κάθε δικτύου.



Σχήμα 7: Παράμετροι του ΤΝΔ που βελτιστοποιούνται από το ΓΑ



Σχήμα 8: Κωδικοποίηση χρωμοσώματος για τις παραμέτρους ΤΝΔ

Η σελίδα αυτή είναι σκόπιμα λευκή

Η λογική χαρτογράφηση από το bitstring format στις παραμέτρους προβλήματος εμφανίζεται στο σχήμα 6, όπου NL είναι ο αριθμός στρωμάτων που χρησιμοποιούνται για κάθε λύση (που ενεργοποιείται), TF_i είναι η συνάρτηση μεταφοράς του i-οστού στρώματος και N/L_i οι νευρώνες του στρώματος i.

Εξαιτίας του γεγονότος ότι το μήκος χρωμοσώματος πρέπει να είναι στατικό και είναι άγνωστο εκ των προτέρων πόσα στρώματα θα δημιουργηθούν σε κάθε λύση υποψηφίων, TFs και N/Ls δημιουργούνται για το μέγιστο αριθμό στρωμάτων και έπειτα λαμβάνονται υπόψη τα μέρη του bitstring που χρησιμοποιούνται σύμφωνα με την πρώτη παράμετρο (αριθμός ενεργοποιημένων στρωμάτων). Αυτό δημιουργεί φυσικά ένα πρόσθετο φορτίο για το ΓΑ, καθώς ο τελευταίος έχει δημιουργήσει τις τιμές για ολόκληρο το bitstring και δεν γνωρίζει ότι ένα μέρος του δεν θα χρησιμοποιηθεί. Από τη γενική επιστρεφόμενη απόδοση των δικτύων θα προέλθουν συμπεράσματα για την ικανότητα των αχρησιμοποίητων περιπτώσεων που μπορεί να συγχύσουν τη διαδικασία σύγκλισης. Εντούτοις από τα πειράματα εξήχθη το συμπέρασμα ότι ακόμη και με αυτό το μειονέκτημα ο αλγόριθμος σύγκλινε μάλλον γρήγορα.

Μετά από τη δημιουργία του πληθυσμού, κάθε δίκτυο κατασκευάζεται δυναμικά λαμβάνοντας το συνδυασμό των παραμέτρων σύμφωνα με το χρωμόσωμα, εκπαιδεύεται και η δυνατότητά του να επιτύχει το στόχο επιστρέφεται στο ΓΑ, ως μέσο μέτρησης της απόδοσης.

Σε αυτό το σημείο πρέπει να αναφερθεί ότι αυτή η διαδικασία θα μπορούσε επίσης να εκτελεσθεί με έναν τυχαίο τρόπο, σε μια αναζήτηση ενός καλύτερου δικτύου. Εντούτοις έχουν υπάρξει έρευνες ([19][20]) που υποδεικνύουν ότι στη προσέγγιση ΓΑ το τυχαίο (που σε ένα ΓΑ απεικονίζεται στην επιλογή του ψηφίου της λύσης που θα

αλλοιώσει ο τελεστής mutation ή στα μέρη του χρωμοσώματος που λαμβάνουν μέρος στη διασταύρωση) συνδυάζεται με το φυσικό τρόπο της εξέλιξης προκειμένου να συγκλίνουν σε μια λύση γρηγορότερα. Στη παρούσα εργασία, μια σύγκριση γίνεται επίσης μεταξύ της τυχαίας και GA-βασισμένης προσέγγισης, η οποία επιβεβαιώνει την επιλογή των τελευταίων. Επιπλέον μπορεί εύκολα να επεκταθεί προκειμένου να περιληφθούν περισσότερα χαρακτηριστικά γνωρίσματα ενός ΤΝΔ, όπως οι συναρτήσεις εκπαίδευσης, οι μέθοδοι εκμάθησης ή οποιοδήποτε άλλο είδος παραμέτρων που μπορούν να έχουν επιπτώσεις στην απόδοση του δικτύου.

Ένα γνωστό πρόβλημα των ΤΝΔ είναι η απώλεια δυνατότητας γενίκευσης λόγω της υπερταύτισης με τα δεδομένα εκπαίδευσης. Αυτό σημαίνει ότι το δίκτυο είναι σε θέση να δώσει μια ακριβή απάντηση σε περιπτώσεις εισόδου που συμπεριλαμβάνονται στο σύνολο δεδομένων εκπαίδευσης αλλά είναι ανίκανο να παρέχει καλές εκτιμήσεις για τις περιπτώσεις που δεν έχουν συναντηθεί προηγουμένως. Προκειμένου να αντιμετωπιστεί αυτή η κατάσταση, τα μοντέλα από κάθε γενεά που έχουν την καλύτερη απόδοση όσον αφορά το κανονικό κριτήριο λάθους (λάθος εκπαίδευσης) σώζονται και μια πρόσθετη δοκιμή γενίκευσης εκτελείται σε ένα σύνολο στοιχείων που δεν έχει χρησιμοποιηθεί κατά τη διάρκεια της εκπαίδευσης. Κατόπιν το δίκτυο με το λιγότερο μέσο λάθος κατά τη δοκιμή γενίκευσης (όπως απεικονίζεται στην Εξίσωση 2) επιλέγεται ως τελικός υποψήφιος.

$$\underset{1,2...N}{Min} \frac{\sum_{i=1}^{K} [| f_j(x_i) - Y_i |]}{K}$$

Εξίσωση 2: Κριτήριο επιλογής καλύτερης λύσης

Όπου K: αριθμός καθορισμένων τιμών επικύρωσης

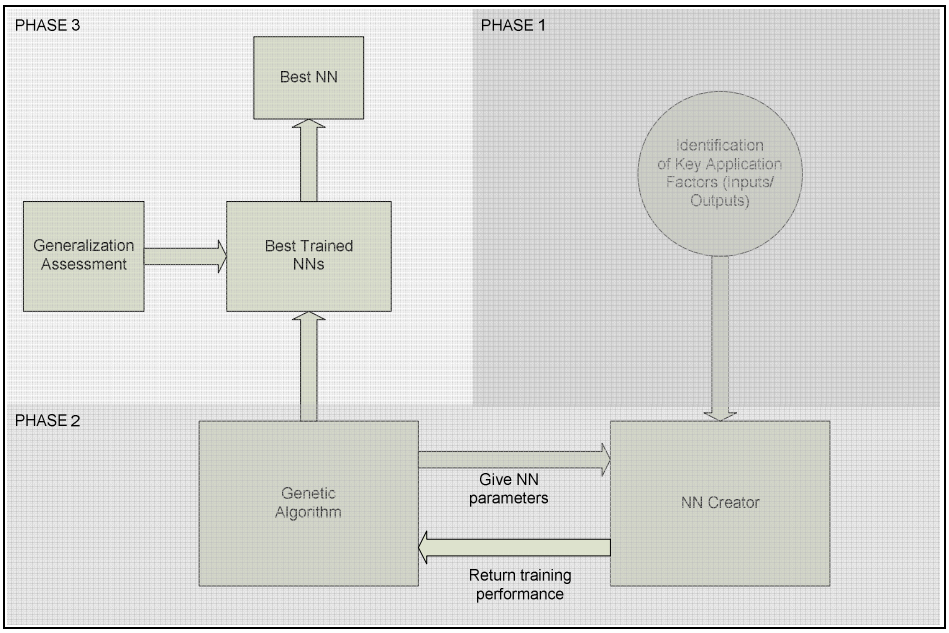
N: αριθμός μοντέλων ANN που εκπληρώνουν το στόχο εκπαίδευσης

f_j: ΤΝΔ μοντέλο υπολογισμού της κατ' εκτίμηση τιμής

x_i: διάνυσμα εισόδων για την i-οστή περίπτωση επικύρωσης της δοκιμής γενίκευσης

Y_i: διάνυσμα γνωστών αποτελεσμάτων για την i-οστή περίπτωση επικύρωσης της δοκιμής γενίκευσης

Το πλήρες εννοιολογικό διάγραμμα της προτεινόμενης προσέγγισης εμφανίζεται στο Σχήμα 9, όπου εμφανίζονται όλες οι φάσεις του γενικού αλγορίθμου πρόβλεψης (ο ψευδοκώδικας εμφανίζεται στο Σχήμα 10).



Σχήμα 9: Γενικό εννοιολογικό διάγραμμα

Εν περιλήψει οι φάσεις που έχουν προσδιοριστεί και έχουν περιγραφεί είναι οι ακόλουθες:

Η σελίδα αυτή είναι σκόπιμα λευκή

α. Φάση 1: Οι είσοδοι και οι έξοδοι της εφαρμογής εξάγονται από την περιγραφή XML (που παρέχεται από τον υπεύθυνο για την ανάπτυξη εφαρμογής), μαζί με το σύνολο εκπαίδευσης που αποτελείται από διάφορες εκτελέσεις δοκιμής ή ιστορικά στοιχεία.

β. Φάση 2: Το σύνολο δεδομένων περνούν στο ΤΝΔ, το οποίο λαμβάνει το υπόλοιπο των παραμέτρων (που αναφέρονται στο σχέδιο του δικτύου) από τον πληθυσμό ΓΑ και επιστρέφει το κριτήριο ικανότητας (λάθος μετά από τη διαδικασία εκπαίδευσης). Σε κάθε γενεά, όλα τα δίκτυα που έχουν καλύτερη απόδοση από κάποιο συγκεκριμένο όριο αποθηκεύονται σε μια δεξαμενή καλύτερων δικτύων.

γ. Φάση 3: Ο τελικός υποψήφιος (καλύτερο ΤΝΔ) επιλέγεται από την δεξαμενή καλύτερων δικτύων με τη χρήση της ενότητας αξιολόγησης της γενίκευσης (μικρότερο λάθος στο σύνολο επικύρωσης).

Μια άλλη παραλλαγή που ερευνήθηκε χρησιμοποιούσε ένα ενδιάμεσο βήμα μέτρησης γενίκευσης, το οποίο μπορεί να εξυπηρετήσει ως καλύτερη μετρική για το ΓΑ. Αυτή η μετρική μπορεί να χρησιμοποιηθεί προκειμένου να σώσει τα καλύτερα δίκτυα από κάθε γενεά που διαθέτουν αυτήν την ικανότητα γενίκευσης (γραμμή 10 στο Σχήμα 10) αλλά αυτό μπορεί επίσης να θεωρηθεί ως εξελικτική μετρική, δηλαδή η δυνατότητα ενός δικτύου για γενίκευση είναι ακριβώς το βασικό χαρακτηριστικό απόδοσης της λύσης, παρά η προσαρμογή στο σύνολο εκπαίδευσης (γραμμή 12 στο Σχήμα 10). Για αυτήν την προσέγγιση, το σύνολο δεδομένων πρέπει να διαιρεθεί σε 3 υποσύνολα, ένα για την εκπαίδευση, ένα για την ενδιάμεση μέτρηση γενίκευσης και ένα για την τελική μέτρηση γενίκευσης, η οποία είναι η τελική μετρική για την επιλογή ενός δικτύου. Αυτό σημαίνει ότι το σύνολο εκπαίδευσης πρέπει να μειωθεί σε σύγκριση με

την αρχική προσέγγιση. Η αντιστάθμιση (trade-off) μεταξύ αυτής της μείωσης και της επίδρασης της νέας μετρικής ερευνάται στην παράγραφο 2.7.3.

. Συνολικά, ερευνήθηκαν παραλλαγές βασισμένες στην εφαρμογή ή μη του ενδιάμεσου βήματος ελέγχου ποιότητας επικύρωσης, στο είδος του κριτηρίου που χρησιμοποιήθηκε προκειμένου να σωθούν τα καλύτερα δίκτυα (απόδοση εκπαίδευσης ή ενδιάμεση απόδοση επικύρωσης) και της ικανότητας που επιστρέφεται στο ΓΑ (απόδοση εκπαίδευσης ή ενδιάμεση απόδοση επικύρωσης) ως μέτρηση εξελικτικής απόδοσης. Τα αποτελέσματα αυτής της σύγκρισης παρουσιάζονται στην παράγραφο 2.7.3, μαζί με διαφορετικά αριθμητικά όρια για τις διάφορες παραλλαγές, αλλά και με μια προσέγγιση τυχαίας δημιουργίας ΤΝΔ.

2.6 Στατιστικός Συνδυασμός Μοντέλων

Όπως περιγράφηκε στα προηγούμενα τμήματα, ο αλγόριθμος αποθηκεύει σταδιακά διάφορα κατάλληλα δίκτυα σύμφωνα με το κριτήριο που επιλέγεται, δημιουργώντας κατά συνέπεια μια ομάδα των καλύτερων μοντέλων, πρίν επιλέξει ανάμεσα από αυτά τον τελικό υποψήφιο βασισμένο στην τελική δοκιμή γενίκευσης κατά τη διάρκεια της φάσης 3. Σε αυτό το σημείο ερευνήθηκε εάν θα ήταν καλύτερος ο συνδυασμός όλων αυτών καλύτερων μοντέλων για την παραγωγή μίας ενιαίας πρόβλεψης από την επιλογή ενός τελικού υποψηφίου.

Η γενική προσέγγιση πίσω από το συνδυασμό των μοντέλων για την εκτίμηση έγκειται στο γεγονός ότι σε πολλές περιπτώσεις των μεθόδων προσέγγισης λειτουργίας, το τελικό αποτέλεσμα μπορεί να προβλέπεται με ακρίβεια μόνο σε ένα περιορισμένο σύνολο του διαστήματος ενδιαφέροντος. Μια λύση είναι σε αυτήν την περίπτωση να δημιουργηθεί μια πιό εύρωστη μέθοδος πρόβλεψης, η οποία θα χρησιμοποιήσει ποικίλα

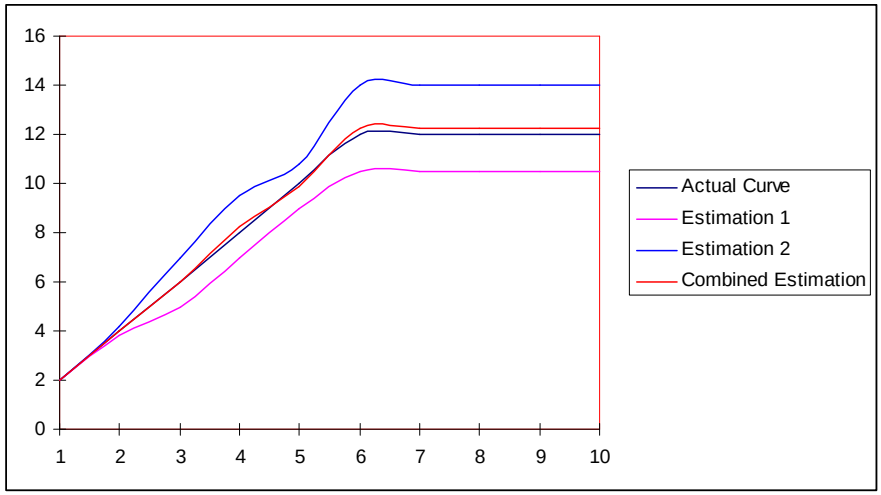
```
1. Create initial random population
2. For i=1 to Max generations
3.    For k=1 to Max population members
4.          Take chromosome and check the first parameter (number of layers activated)
5.          For j= 1 to Max number of activated layers
6.                Extract the parts of the bit-string that are activated
7.          End for
8.          Create ANN based on chromosome parameters
9.          Train network
10.         If Save_Criterion<global limit (arbitrary value defined at the beginning)
11.               Save net in global best (for generalization check)
12.         Save performance criterion (training or intermediate validation) for the GA
13.   End for
14.   Transform population with GA operators and create i+1 generation
15. End for
16. For every network inside global best
17.   Check error in independent final validation test
18.   If error< up to now best
19.         This network is the best
20. End for
```

Σχήμα 10: Ψευδοκώδικας για τη δημιουργία μοντέλου

μοντέλα σε μία προσπάθεια να ομαλοποιηθούν οποιαδήποτε λάθη στη διαδικασία εκτίμησης. Εάν δύο ή περισσότερα μοντέλα αποδεικνύονται συμπληρωματικά (όπως ενδεικτικά παρουσιάζεται στο Σχήμα 11), η συνδυασμένη εκτίμησή τους μπορεί να έχει

Η σελίδα αυτή είναι σκόπιμα λευκή

καλύτερα αποτελέσματα. Αυτός ο συνδυασμός εκτελέσθηκε μέσω των διαφορετικών στατιστικών συνδυασμών που προσδιορίζονται στις επόμενες παραγράφους.



Σχήμα 11: Παράδειγμα συμπληρωματικών μοντέλων πρόβλεψης

Μέση τιμή και διάμεσος

Αυτός ο συνδυασμός είναι πιθανό να παράγει πιο εύρωστα αποτελέσματα. Ο λόγος για αυτό εμφανίζεται στον ακόλουθο τύπο:

Ας υποθέσουμε ότι έχουμε N διαθέσιμα μοντέλα για την εκτίμηση. Η πραγματική τιμή της κατ' εκτίμηση εξόδου είναι T, και το λάθος του i_{th} μοντέλου για μια μεμονωμένη εκτίμηση είναι e_i. Όταν χρησιμοποιούνται διάφορα μοντέλα, για τα οποία δεν υπάρχει καμία απόλυτη εγγύηση ότι ένα αποδίδει καλύτερα από τα υπόλοιπα, με την επιλογή μόνο του ενός, στη χειρότερη περίπτωση μπορεί να συμβεί αυτό το μοντέλο να παράγει το μέγιστο λάθος του σε αυτήν την πρόβλεψη. Εάν χρησιμοποιείται η μέση τιμή της πρόβλεψης όλων των μοντέλων θα ισχύει:

$$\frac{1}{N}\sum_{i=1}^{N}(T+e_i)=\frac{1}{N}\sum_{i=1}^{N}T+\frac{1}{N}\sum_{i=1}^{N}e_i=T+\frac{1}{N}\sum_{i=1}^{N}e_i$$

Εξίσωση 3: Σφάλμα από χρησιμοποίηση μέσης τιμής πρόβλεψης από πληθώρα μοντέλων

Η σελίδα αυτή είναι σκόπιμα λευκή

το οποίο σημαίνει ότι το σφάλμα τώρα θα είναι το μέσο σφάλμα από όλα τα χρησιμοποιούμενα μοντέλα:

$$\left|\frac{1}{N}\sum_{i=1}^{N} e_i\right| < \max_i(|e_i|)$$

Εξίσωση 4: Σχέση μέσου σφάλματος με μέγιστο

Αυτή η μείωση θα είναι μεγαλύτερη στην περίπτωση όπου έχουμε διαφορετικά πρόσημα στο λάθος, αλλά ακόμη και στην περίπτωση που όλα τα μοντέλα παράγουν υπερεκτιμημένες τιμές παραδείγματος χάριν, πάλι το λάθος θα είναι λιγότερο από το μέγιστο. Αυτό είναι ιδιαίτερα χρήσιμο όταν δεν υπάρχει κανένας εγγυημένος τρόπος να είναι γνωστό ποιο δίκτυο παράγει την καλύτερη εκτίμηση. Σε πολλές περιπτώσεις ο καλύτερος υποψήφιος μπορεί να έχει ελάχιστη διαφορά στο συνολικό λάθος από το δεύτερο καλύτερο.

Αφαίρεση outliers

Γενικά ως ακραίες τιμές (outliers) θεωρούνται δεδομένα που είναι αριθμητικά απόμακρα από την κανονική κατανομή, συχνότερα αυτά που έχουν μια απόκλιση μεγαλύτερη από την τριπλάσια τυπική απόκλιση. Η αφαίρεση outliers, που προκαλείται κανονικά από τα λάθη στη συλλογή δεδομένων μιας πειραματικής διαδικασίας ή κάποιο άλλο παράγοντα που την επηρεάζει, είναι μια μέθοδος προκειμένου να βελτιωθεί το σύνολο δεδομένων με την εξάλειψη των μολυσμένων μετρήσεων. Στην περίπτωσή μας, θεωρήθηκε ότι αφού περισσότερες από μια εκτιμήσεις χρησιμοποιούνται, αυτή η μεθοδολογία μπορεί να εφαρμοστεί προκειμένου να αφαιρεθούν οι εκτιμήσεις που έχουν μια μεγάλη διαφορά από τις υπόλοιπες. Η αφαίρεση outliers εξόδου, δηλαδή των εκτιμήσεων των μοντέλων που απόκλιναν πάνω από 3 τυπικές αποκλίσεις από την μέση τιμή της εκτίμησης όλων

των μοντέλων, ερευνήθηκε προκειμένου να παρατηρηθεί η επίδραση στην τελική εκτίμηση.

Αναζήτηση του καταλληλότερου παραθύρου

Μια άλλη προσέγγιση που ερευνήθηκε χρησιμοποιούσε όχι την ολόκληρη σειρά των κατ' εκτίμηση αποτελεσμάτων από όλα τα μοντέλα αλλά ένα παράθυρο τους, που αρχίζει από τη τιμή στη μέση της ταξινομημένης σειράς όλων των αποτελεσμάτων και που ερευνά όλα τα παράθυρα των τιμών επάνω από και κάτω από αυτή τη τιμή. Ο ψευδοκώδικας για αυτό παρουσιάζεται στον ακόλουθο αριθμό (Σχήμα 12). Αυτό είναι μια εκτεταμένη έκδοση της δημοφιλούς μεθόδου αφαίρεσης των υψηλότερων και χαμηλότερων τιμών από μια ομάδα τιμών, πρίν εξαχθεί ο μέσος όρος. Στην παρούσα περίπτωση, αυτή η αφαίρεση επεκτάθηκε από 0 (εκφυλισμός σε μέθοδο μέσου όρου) έως k/2 (εκφυλισμός σε μέθοδο διαμέσου).

Εντούτοις, παρατηρήθηκε από την αξιολόγηση ότι ενώ τα αποτελέσματα από αυτούς τους συνδυασμούς ήταν αποδεκτά, η απόδοσή τους ήταν σημαντικά χειρότερη από το καλύτερο δίκτυο που παρήχθη από τις διάφορες παραλλαγές. Αυτό οφείλεται στο γεγονός ότι όλα τα πρότυπα ακολουθούν παρόμοιες αρχές σχεδίου, οπότε εμφανίζουν και σχετικά παρόμοια συμπεριφορά. Ο συνδυασμός διαφορετικών μεθόδων που παράγουν μοντέλα με αντίθετες πολώσεις θα μπορούσε να παρουσιάσει ενδιαφέρον για μελλοντική έρευνα.

2.7 Αξιολόγηση

Στις ανωτέρω παραγράφους, αναφέρθηκε μια λεπτομερής περιγραφή σχετικά με το πώς ΓΑ και ΤΝΔ μπορούν να συνδυαστούν σε έναν μηχανισμό που χαρτογραφεί την επίδραση των παραμέτρων φόρτου μιας εφαρμογής και του φυσικού πόρου εκτέλεσής

της στην προσφερόμενη ποιότητα υπηρεσίας. Ακολούθως, οι λεπτομέρειες της υλοποίησης παρουσιάζονται για αυτόν τον μηχανισμό μαζί με μια δοκιμή για τρία πραγματικά σενάρια εφαρμογής (κωδικοποίηση ακατέργαστου βίντεο σε MPEG4, μια εφαρμογή MPI/OpenMP και έναν εξυπηρετητή πραγματικού χρόνου που περιείχε μία εφαρμογή ηλεκτρονικής εκμάθησης με περιορισμούς στους χρόνους απάντησης). Επιπλέον, τρία (3) πειράματα πραγματοποιήθηκαν σε αυτήν την δοκιμή προκειμένου να αξιολογηθεί η λειτουργία της παρουσιαζόμενης προσέγγισης.

```
1. Take saved best networks for one variation(from GA)
2. Estimate outputs from all networks
3. Sort output estimations by numeric value
4. For k=1 to (Max number of best networks)/2
5.      For i=1 to Max validation case
6.              Take outputs that are k/2 above or below the middle value
7.              Calculate Mean of outputs in window k
8.      End for
9.      Count overall error of this window
10.     if this error< up_to_now_best
11.             save window k as the best
12. End for
```

Σχήμα 12: Ψευδοκώδικας παραθυρικής πρόβλεψης

2.7.1 Υλοποίηση Αλγορίθμου

Ένα αρχικό script για τον ΓΑ δημιουργείται προκειμένου να αντιμετωπιστεί ο πληθυσμός σε κάθε επανάληψη και για να εκκινήσει το script αρχικοποίησης και δημιουργίας του ΤΝΔ, δίνοντάς του τις τιμές που απεικονίζονται στα χρωμοσώματα. Το ANN script παίρνει τις μεταβλητές, αρχικοποιεί και εκπαιδεύει το ANN (feed-forward, back-propagation network, εκπαιδευμένο με το αλγόριθμο Levenberg-Marquardt[21]) και επιστρέφει το κριτήριο απόδοσης εκπαίδευσής του (MSE στην εκπαίδευση ή το μέσο απόλυτο ποσοστιαίο σφάλμα στο ενδιάμεσο σύνολο επικύρωσης, ανάλογα με την παραλλαγή αλγορίθμου που χρησιμοποιήθηκε) στο GA για την αξιολόγηση κάθε λύσης. Ολόκληρη η σειρά των διαθέσιμων συναρτήσεων μεταφοράς (στο περιβάλλον MATLAB) ελήφθη υπό εξέταση (tansig, logsig, purelin, hardlim, compet, hardlims,

Η σελίδα αυτή είναι σκόπιμα λευκή

netinv, poslin, radbas, satlin, satlins, softmax, tribas) και χρησιμοποιήθηκε ένα μέγιστο όριο δέκα (10) στρωμάτων και τριάντα (30) νευρώνων ανά στρώμα.

Οι βασικές ρυθμίσεις (Πίνακας 1) που χρησιμοποιήθηκαν για τις παραμέτρους του ΓΑ αφορούν κυρίως το χειρισμό πληθυσμών.

Πίνακας 1: Βασικές ρυθμίσεις ΓΑ

Μέγεθος πληθυσμού	**Μέλη Ελιτισμού**	**Διασταύρωση**	**Μετάλλαξη**	**Αριθμός Γενεών**
20	10%	80%	10%	30

Οι εκτελέσεις ΓΑ περιείχαν 30 γενεές, με συνέπεια 600 διαφορετικά νευρωνικά δίκτυα να εξετάζονται, 20 δίκτυα (μέγεθος πληθυσμών) σε κάθε μια από τις 30 γενεές. Ο αλγόριθμος τρέχει για περίπου 30 λεπτά έως 1 ώρα και ο αριθμός των δικτύων που σώζονται για τη δοκιμή γενίκευσης εξαρτάται από την παραλλαγή που χρησιμοποιείται (στις περιπτώσεις μας ήταν από 5 έως 192). Το χρονικό πλαίσιο της εκτέλεσης μπορεί να εμφανίζεται αυξημένο, αλλά αυτό εκτελείται μιά φορά για κάθε τμήμα εφαρμογής που το PaaS θέλει να μοντελοποιήσει.

Σε αυτό το σημείο πρέπει να τονιστεί ότι από το περιβάλλον MATLAB χρησιμοποιήθηκαν μόνο οι βασικές λειτουργίες που παρέχονται, παραδείγματος χάριν, η λειτουργία GA που είναι απαραίτητη για τη δημιουργία και το χειρισμό πληθυσμών και οι βασικές λειτουργίες που χρησιμοποιούνται για τη δημιουργία και την κατάρτιση ANN. Το δυσκολότερο μέρος ήταν ο καθορισμός του χρωμοσώματος και η δυναμική δημιουργία κάθε ANN βασισμένη στις τιμές του πρώτου, το οποίο ήταν μια πολύ σύνθετη διαδικασία.

2.7.2 Δοκιμαστικές Εφαρμογές

Για τους σκοπούς των πειραμάτων τρεις εφαρμογές ερευνήθηκαν προκειμένου να υπολογιστεί η απόδοσή τους βασισμένη σε αυτήν την προσέγγιση. Για όλες τις υπό έρευνα εφαρμογές καμία πληροφορία σχετικά με τον κώδικα πηγής δεν χρησιμοποιήθηκε (ικανοποίηση του περιορισμού 2), μόνο υψηλού επιπέδου πληροφορίες που προέρχονται από τον υπεύθυνο για την ανάπτυξη εφαρμογής.

Η πρώτη εφαρμογή ήταν μια κωδικοποίηση ακατέργαστου βίντεο στο φορμάτ MPEG4 με τον κωδικοποιητή FFMPEG [29]. Προκειμένου να ερευνηθεί η συμπεριφορά του κώδικα, χρησιμοποιήθηκαν τέσσερα χαρακτηριστικά/παράμετροι επιπέδου εφαρμογής που έχουν μια άμεση επίδραση στο φόρτο εργασίας που παράγεται από αυτήν. Αυτά ήταν η διάρκεια του βίντεο, τα πλαίσια ανά δευτερόλεπτο (FPS) που χρησιμοποιήθηκαν στη λήψη, η ανάλυση των εικόνων και ένας δείκτης της έντασης της κίνησης μέσα στο βίντεο (0 για ακίνητες εικόνες, 0.5 για μέτρια δραστηριότητα και 1 για έντονη αλλαγή πληροφορίας ανά πλαίσιο). Αυτός ο δείκτης είναι σημαντικός σε περιπτώσεις αλγορίθμων επεξεργασίας που συγκρίνουν τα επακόλουθα πλαίσια και εκτελούν διαδικασίες βασισμένες στις διαφορές μεταξύ τους. Μεγάλες διαφορές μεταξύ των πλαισίων, όπως σε μια ταινία δράσης παραδείγματος χάριν, έχουν σαν αποτέλεσμα ο χρόνος υπολογισμού να αυξηθεί. Ο γενικός χρόνος εκτέλεσης έως ότου τελειώσει η κωδικοποίηση επιλέχτηκε ως μετρική ποιότητας υπηρεσίας (KPI). Ανάλογα με το πόσο γρήγορα ο πελάτης θέλει να διεκπεραιώσει την αίτησή του, ο προμηθευτής PaaS θα επιλέξει έναν καλύτερο ή χειρότερο κόμβο υλικού και θα χρεώσει αναλόγως. Αυτές οι πληροφορίες παρέχονται από τον υπεύθυνο για την ανάπτυξη εφαρμογής μέσω μιας περιγραφής XML και χρησιμοποιούνται από το πλαίσιο.

Για τη δεύτερη εφαρμογή επιλέχτηκε.μια παράλληλη εφαρμογή solver γραμμικών εξισώσεων που χρησιμοποιεί τη Conjugate Gradient method [25] Αυτό ήταν υλοποιημένο και με τη τεχνολογία παράλληλης επεξεργασίας MPI [27] προκειμένου να είναι σε θέση να εκτελεστεί ταυτόχρονα σε πολλαπλούς κόμβους και OpenMP [28] προκειμένου να χρησιμοποιηθούν οι πολυεπεξεργαστικοί κόμβοι. Σαν παράμετρος επιπέδου εφαρμογής ορίστηκε το μέγεθος του συστήματος των εξισώσεων που χρησιμοποιήθηκε. Άλλο χαρακτηριστικό που μπορεί να επηρεάσει επίσης το χρόνο εκτέλεσης είναι ο condition number, ο οποίος έχει επιπτώσεις στο χρόνο σύγκλισης της λύσης του συστήματος εξισώσεων. Σε αυτήν την περίπτωση, δεδομένου ότι το σύστημα εξίσωσης ήταν δοκιμαστικό, ο condition number της μήτρας ήταν ίσος με το μέγεθος του συστήματος εξισώσεων.

Σαν είσοδοι προσδιορισμού των υλικών πόρων χρησιμοποιήθηκαν ο αριθμός κόμβων και ο αριθμός χρησιμοποιούμενων πυρήνων ανά κόμβο (οι διαφορετικές συχνότητες CPU δεν διαδραμάτισαν ρόλο δεδομένου ότι οι μετρήσεις έγιναν σε ομοιογενή cluster). Αυτή η εφαρμογή επιλέχτηκε δεδομένου ότι η συμπεριφορά του προγράμματος αλλάζει σοβαρά σε σχέση με τις επιλεγμένες παραμέτρους. Η αύξηση του αριθμού κόμβων δεν σημαίνει απαραιτήτως και βελτίωση του χρόνου εκτέλεσης, γεγονός που οφείλεται στις καθυστερήσεις επικοινωνίας που παρεμβάλλονται για το συγχρονισμό των νημάτων. Έτσι είναι μεγάλης σπουδαιότητας να ανιχνευθούν οι εξαρτήσεις σε αυτήν την περίπτωση, δεδομένου ότι με τη διάθεση περισσότερων από τους βέλτιστους κόμβους όχι μόνο θα οδηγηθούμε σε περισσότερους χρησιμοποιούμενους πόρους αλλά και στην υποβάθμιση της απόδοσης της εφαρμογής. Ο πελάτης σε αυτήν την περίπτωση θα θεωρήσει επίσης το χρόνο εκτέλεσης της εργασίας ως μέτρο ποιότητας (KPI). Αυτό

σημαίνει ότι το στρώμα PaaS πρέπει να έχει μια εκτίμηση του αριθμού των κόμβων ή των πυρήνων που πρέπει να διατεθούν για να επιτευχθεί ο περιορισμός KPI. Για τις εκτελέσεις, μια συστάδα 8 κόμβων χρησιμοποιήθηκε, με 2 διαθέσιμους πυρήνες/κόμβο.

Για την τρίτη εφαρμογή, επιλέχτηκε ένα από τα σενάρια του ερευνητικού προγράμματος IRMOS (διαδραστική εφαρμογή ηλεκτρονικής μάθησης πραγματικού χρόνου) [35]. Αυτή αποτελούνταν από έναν κεντρικό εξυπηρετητή που λαμβάνει τα αιτήματα με τοποθεσία από τους χρήστες και αποκρίνεται με τα κοντινά εκπαιδευτικά αντικείμενα γύρω από αυτούς. Το σημαντικό μέρος αυτής της εφαρμογής είναι ότι ο χρόνος που λαμβάνεται για την επεξεργασία αυτών των αιτημάτων μέσα στον κεντρικό εξυπηρετητή έχει χρονικούς περιορισμούς που πρέπει να ικανοποιηθούν. Αυτοί οι περιορισμοί προσφέρονται γενικά ως επίπεδο QoS της υπηρεσίας (δείκτης KPI). Από την άποψη της εφαρμογής, το σημαντικότερο χαρακτηριστικό που επηρεάζει τους πόρους που απαιτούνται προκειμένου να ικανοποιηθούν τα προαναφερθέντα επίπεδα QoS είναι ο αριθμός χρηστών που πρόκειται να έχουν πρόσβαση στην υπηρεσία. Μια άλλη παράμετρος σπουδαιότητας είναι το μέγεθος της βάσης δεδομένων, αλλά αυτό είναι στις περισσότερες περιπτώσεις στατικό και δεν αλλάζει συχνά.

Εξαιτίας του γεγονότος ότι αυτή η εφαρμογή έχει επίσης τους περιορισμούς πραγματικού χρόνου, σαν παράμετρος πόρων χρησιμοποιήθηκε η διαμόρφωση ενός χρονοπρογραμματιστή (real-time scheduler), εκτός από τις παραμέτρους υλικού όπως η ταχύτητα CPU. Στο πλαίσιο του IRMOS, χρησιμοποιείται ο χρονοπρογραμματιστής που περιγράφεται στο [32]. Για τον χρονοπρογραμματιστή αυτόν, οι σημαντικοί παράγοντες είναι το ποσοστό της CPU που ορίζεται να δοθεί στην εφαρμογή και η περίοδος στην οποία θα δίνεται αυτό το ποσοστό. Αυτό απεικονίζεται από τις παραμέτρους Q και P.

Q/P είναι το ποσοστό του μεριδίου CPU και το P είναι το χρονικό διάστημα στο οποίο αυτό το ποσοστό δίνεται. Η επίδραση της παραμέτρου P είναι σημαντική, σε περιπτώσεις διαδραστικών εφαρμογών παραδείγματος χάριν οπότε αυτό πρέπει να είναι σημαντικά χαμηλό προκειμένου να ικανοποιήσει τους περιορισμούς αλληλεπίδρασης. Για άλλες εφαρμογές όπως οι υπολογιστικά εντατικές επιστημονικές προσομοιώσεις, αυτό πρέπει να αυξηθεί προκειμένου να υπάρξει η καλύτερη χρησιμοποίηση μνήμης cache. Σε όλες τις περιπτώσεις, είναι ένας σημαντικός παράγοντας που πρέπει να ληφθεί υπόψη. Αυτή η εφαρμογή εκτελείται μέσα σε μια εικονική μηχανή που διαμορφώνεται για να χρησιμοποιήσει μόνο έναν πυρήνα προς το παρόν, οπότε ο αριθμός πυρήνων δεν ήταν μια παράμετρος για το υλικό. Η προβλεφθείσα έξοδος ήταν σε αυτήν την περίπτωση ο μέσος χρόνος απόκρισης των αιτημάτων που εξυπηρετήθηκαν και η τυπική απόκλιση αυτών των χρόνων ώστε να μπορεί να ανακατασκευαστεί η κατανομή τους. Αυτό είναι σημαντικό καθώς στις εφαρμογές πραγματικού χρόνου με χαλαρές απαιτήσεις το 95% των τιμών πρέπει να είναι κάτω από το όριο QoS που έχει οριστεί.

Στον ακόλουθο πίνακα (Πίνακας 2) απεικονίζεται η περίληψη των μοντέλων και των εισόδων και εξόδων τους. Γενικά, οι είσοδοι εφαρμογής προσδιορίζονται από τον υπεύθυνο για την ανάπτυξη εφαρμογής (και τον ιδιοκτήτη του λογισμικού) και οι είσοδοι προσδιορισμού πόρων προέρχονται από μια συμφωνία μεταξύ του στρώματος PaaS και IaaS. Σε αυτά τα πειράματα έχουν χρησιμοποιηθεί διαφορετικές παραμέτροι πόρων, προκειμένου να επικυρωθεί η γενική φύση της επιλογής μαύρων κουτιών. Επιπλέον, μέσω των χρησιμοποιημένων predictors (είσοδοι), μπορεί να αποκτηθεί η πρόβλεψη ανά συγκεκριμένη εκτέλεση SLA, ικανοποιώντας κατά συνέπεια τον περιορισμό 1.

Πίνακας 2: Καθορισμένες είσοδοι και έξοδοι για τα μοντέλα των πειραμάτων

Πείραμα	**Είσοδοι Μοντέλου**							**Εκτιμώμενες έξοδοι (KPIs)**	
	Επίπεδο εφαρμογής (φόρτος)				**Επίπεδο Υλικού**				
Κωδικοποίηση βίντεο	Διάρκεια	FPS	Ανάλυση	Δείκτης κίνησης	Ταχύτητα CPU		Αριθμός πυρήνων	Αποθηκευτικός χώρος	Χρόνος εκτέλεσης
MPI Επιλυτής Εξισώσεων	Μέγεθος συστήματος εξισώσεων				Αριθμός κόμβων	Πυρήνες/κόμβο		Χρόνος εκτέλεσης	
Διαδραστικός Εξυπηρετητλης πραγματικού χρόνου	Αριθμός χρηστών				P	%CPU (Q/P)	Ταχύτητα CPU	RAM	Μέσος χρόνος απόκρισης	Τυπική απόκλιση χρόνων απόκρισης

Είναι σημαντικό να παρατηρηθεί ότι ο κατάλογος των παραμέτρων που εξετάζονται εδώ μπορεί εύκολα να επεκταθεί, συμπεριλαμβανομένων άλλων που ο υπεύθυνος για την ανάπτυξη εφαρμογής μπορεί να θεωρήσει ως κρίσιμες. Αυτή η επέκταση έχει επιπτώσεις μόνο στον αριθμό εισόδων για το ANN και όχι στην υλοποίηση.

2.7.3 Πειράματα

Για όλα τα πειράματα, ελήφθη μια σειρά μετρήσεων, βασισμένη σε διαφορετικούς συνδυασμούς των εισόδων των μοντέλων. Γενικά, 70% του διαθέσιμου συνόλου δεδομένων που παρήχθη από τις εκτελέσεις δοκιμής χρησιμοποιήθηκε ως σύνολο εκπαίδευσης και 30% για την ανεξάρτητη επικύρωση που τέθηκε ως στόχος προκειμένου να ελεγχθούν η ικανότητα γενίκευσης και η ακρίβεια των παραχθέντων δικτύων. Για το ενδιάμεσο βήμα επικύρωσης χρησιμοποιήθηκε το 20% του αρχικού συνόλου δεδομένων, μειώνοντας κατά συνέπεια το σύνολο εκπαίδευσης στο 50%. Το τελικό σύνολο επικύρωσης δεν άλλαζε στο μέγεθος ή το περιεχόμενο.

Οι είσοδοι των μοντέλων που περιγράφονται στον παραπάνω πίνακα εφαρμόστηκαν στα παραχθέντα μοντέλα για τις περιπτώσεις επικύρωσης και το λάθος της πρόβλεψης εξόδου παρουσιάζεται στις ακόλουθες παραγράφους για τις διαφορετικές παραλλαγές που προσδιορίζονται. Για κάθε πείραμα, περιγράφεται η βελτιστοποιημένη δομή των προκυπτόντων δικτύων που παρήχθησαν.

Πείραμα 1

Για την κωδικοποίηση FFMPEG, η πρόβλεψη αφορούσε το χρόνο ολοκλήρωσης της μετατροπής από το ακατέργαστο βίντεο στο σχήμα MPEG4. Είναι σημαντικό να παρατηρηθεί ότι τα αποτελέσματα περιλαμβάνουν μόνο το σύνολο επικύρωσης, για λόγους αντικειμενικότητας. Η επιτυχία του δικτύου στο σύνολο εκπαίδευσης ήταν εξαιρετικά υψηλή (κριτήριο σφάλματος κοντά στο μηδέν) και δεν λήφθηκε υπόψη για τη μέτρηση της ακρίβειας πρόβλεψης. Επιπλέον, οι είσοδοι εφαρμογής χρησιμοποιήθηκαν προκειμένου να προβλεφθεί ο χώρος που θα απαιτούνταν για να αποθηκευτεί το αρχείο τοπικά. Χωριστά ANN δημιουργήθηκαν για την αποθήκευση και για τη χρονική εκτίμηση εκτέλεσης εξαιτίας του γεγονότος ότι οι είσοδοι ήταν διαφορετικές (το αρχείο μεγέθους αποθήκευσης εξαρτάται μόνο από τις παραμέτρους εφαρμογής και όχι από τις παραμέτρους πόρων). Ογδόντα (80) διαφορετικές διαμορφώσεις εκτελέσθηκαν, για μια ποικιλία των παραμέτρων εισόδου που έχουν περιγραφεί ανωτέρω. Για κάθε διαμόρφωση, 4 τρεξίματα εκτελέσθηκαν, από τα οποία εξήχθη η μέση τιμή του χρόνου εκτέλεσης. Αυτό που παρατηρήθηκε κατά τη συλλογή ήταν ο θόρυβος στο σύνολο δεδομένων, κυρίως εξαιτίας του γεγονότος ότι ενώ η εφαρμογή εκτελούνταν, άλλες διαδικασίες (ειδικά από το λειτουργικό σύστημα) εμφανίζονταν με τυχαίο τρόπο και

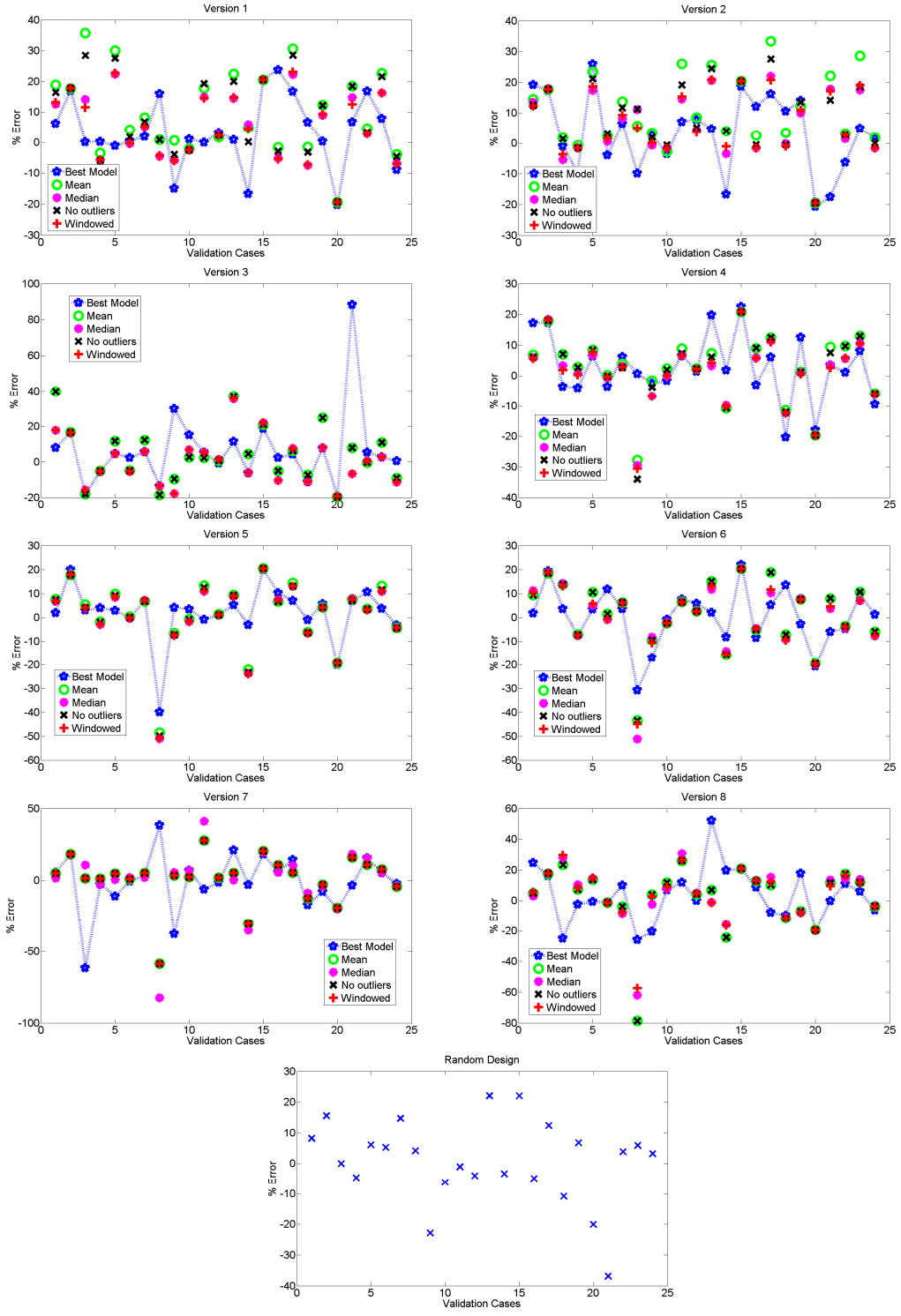
κατέλαμβαναν ποσά χρόνου στην CPU, επηρεάζοντας και τον χρόνο εκτέλεσης της εφαρμογής.

Συνολικά, οκτώ (8) διαφορετικές εκδόσεις της προτεινόμενης προσέγγισης εξετάστηκαν και συγκρίθηκαν με μια τυχαία διαδικασία σχεδιασμού δικτύων. Η τελευταία δημιούργησε 600 διαφορετικά δίκτυα με τυχαίο τρόπο (ίδιος αριθμός με το μέγιστο που παράγεται μέσω του GA), από την οποία επιλέχτηκε το καλύτερο. Κατόπιν, το σχέδιο που περιγράφηκε στο Κεφάλαιο 2.5 εφαρμόστηκε σε μια σειρά διαφορετικών διαμορφώσεων. Αυτές εξαρτήθηκαν από τη μετρική που επιστρεφόταν στον ΓΑ ως απόδοση για το δίκτυο, από τη χρησιμοποίηση ή όχι της ενδιάμεσης επικύρωσης και το κριτήριο που ίσχυε για την αποθήκευση των καλύτερων δικτύων από κάθε γενιά. Αυτές οι εκδόσεις εμφανίζονται λεπτομερώς στον ακόλουθο πίνακα (Πίνακας 3), όπου MSE είναι ο μέσος όρος των τετραγωνικών λαθών των δικτύων στο σύνολο δεδομένων εκπαίδευσης και το IVE είναι το λάθος (μέσο απόλυτο σφάλμα %) στο ενδιάμεσο σύνολο δεδομένων επικύρωσης.

Πίνακας 3: Σύνολο παραλλαγών παραμέτρων αλγόριθμου

Παραλλαγή	Ενδιάμεση επικύρωση	Επιστρεφόμενη μετρική απόδοσης ΓΑ	Κριτήριο αποθήκευσης
1	Οχι	MSE	MSE<10^{-27}
2	Οχι	MSE	MSE<10^{-28}
3	Οχι	MSE	MSE<10^{-29}
4	Ναι	MSE	IVE <15%
5	Ναι	IVE	IVE<15%
6	Ναι	IVE	IVE<8%
7	Ναι	IVE	IVE<15% and MSE <10^{-27}
8	Ναι	MSE	IVE<15% and MSE <10^{-27}

,



Σχήμα 13: Σύγκριση παραλλαγών αλγόριθμου, στατιστικού συνδυασμού και τυχαίας παραγωγής μοντέλων

Η σελίδα αυτή είναι σκόπιμα λευκή

Για όλες αυτές τις περιπτώσεις, η στατιστική ανάλυση, που παρουσιάστηκε στη παράγραφο 2.6, εξετάστηκε επίσης. Η χρήση του μέσου όρου, η διάμεσος, η αφαίρεση outliers και το καλύτερο παράθυρο συγκρίθηκαν με το γενικό καλύτερο μοντέλο που παρήχθη με κάθε έκδοση που εξετάστηκε. Στις ακόλουθες γραφικές παραστάσεις (Σχήμα 13), για κάθε έκδοση το λάθος της εκτίμησης του καλύτερου προκύπτοντος μοντέλου σχεδιάζεται για κάθε περίπτωση επικύρωσης. Σε μερικές γραφικές δεν υπήρχαν outliers οπότε η γραφική παράσταση είναι η ίδια με το μέσο όρο.

Ο ακόλουθος πίνακας (Πίνακας 4) περιέχει τη σύγκριση μεταξύ των κορυφαίων 5 διαφορετικών προσεγγίσεων και των παραλλαγών που ερευνώνται ανωτέρω, μαζί με την τυχαία επιλογή, για λόγους σύγκρισης. Επίσης εμφανίζονται οι λεπτομέρειες σχετικά με το σχέδιο των παραχθέντων δικτύων. Ενώ ο καλύτερος εκτιμητής (Παραλλαγή 4-καλύτερο μοντέλο) δεν είχε το καλύτερο μέσο απόλυτο σφάλμα, εντούτοις η ανώτερη απόδοσή του στο μέγιστο λάθος μαζί με τη μικρή διαφορά στο MAE το κάνει καταλληλότερο υποψήφιο για την τελική επιλογή. Από αυτόν τον πίνακα είναι εμφανές ότι οι προσεγγίσεις που παρουσιάζονται σε αυτό το κεφάλαιο ξεπερνούν την τυχαία μέθοδο σχεδίου. Επιπλέον, παρά το γεγονός ότι χρησιμοποιείται μία αυστηρή μετρική, όπως το μέσο απόλυτο σφάλμα και όχι μόνο το μέσο λάθος, τα αποτελέσματα είναι αξιόπιστα και είναι σε θέση να προβλέψουν με ακρίβεια την έξοδο, βασισμένη στους παράγοντες εισόδου. Ο συνδυασμός γραμμικών και μη γραμμικών χαρακτηριστικών γνωρισμάτων ήταν επίσης διαθέσιμος, ικανοποιώντας κατά συνέπεια τον περιορισμό 3.

Ένα πρόσθετο όφελος της παραλλαγής 4 είναι η μετρική για την αποθήκευση των δικτύων. Στις περιπτώσεις όπου χρησιμοποιείται το MSE, αυτό είναι μια αυθαίρετη αξία που μπορεί να αλλάζει από εφαρμογή σε εφαρμογή και από ένα σύνολο δεδομένων στο

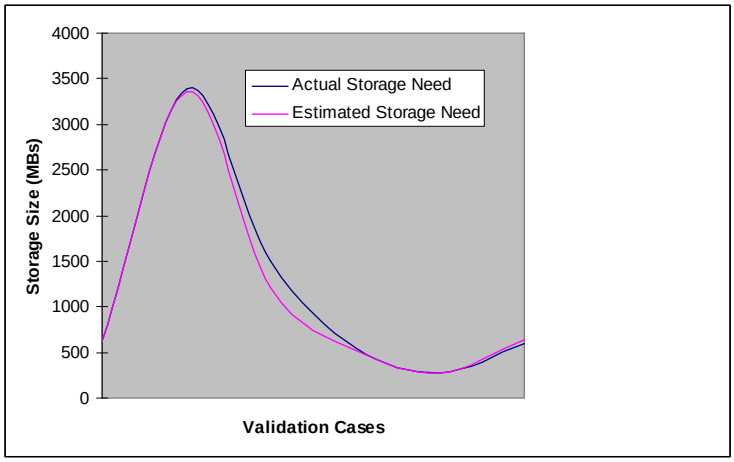
άλλο. Εντούτοις η ανάγκη να υπάρξει ένα απόλυτο σφάλμα επικύρωσης π.χ. μικρότερο από 15% είναι σταθερή. Από αυτήν την ανάλυση, μπορεί να εξαχθεί το συμπέρασμα ότι ο στατιστικός συνδυασμός των μοντέλων δεν βοήθησε σημαντικά στη βελτίωση της γενικής ποιότητας των εκτιμήσεων. Τρία από τα κορυφαία πέντε ήταν τα καλύτερα μοντέλα και οι μόνοι στατιστικοί συνδυασμοί που έφθασαν στα κορυφαία 5 θεώρησαν τη γενικά καλύτερη παραλλαγή (παραλλαγή 4). Αυτό είναι ενδεικτικό ότι όλα (ή τα περισσότερα από) τα σωζόμενα δίκτυα που παρήχθησαν από αυτήν την έκδοση ήταν τα ακριβέστερα και ότι αυτός ο παράγοντας βοήθησε τους στατιστικούς συνδυασμούς να φθάσουν σε ένα αυξανόμενο επίπεδο ακρίβειας. Συμπερασματικά, η παραλλαγή 4 είναι η προκύπτουσα καταλληλότερη προσέγγιση.

Πίνακας 4: Καλύτεροι υποψήφιοι από όλες τις παραλλαγές για τη πρόβλεψη χρόνου εκτέλεσης FFMPEG

Παραλλαγή	**Αριθμός Στρωμάτων**	**Νευρώνες/στρώμα**	**Συνάρτηση Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Μέγιστο Απόλυτο Σφάλμα (%)**
Παραλλαγή 4 (Καλύτερο μοντέλο)	3	6-13-1	Tansig-Softmax-Tansig	8.2145	22.4542
Παραλλαγή 5 (Καλύτερο μοντέλο)	4	6-10-13-1	Softmax-Satlins-Satlin-Satlins	7.6865	39.9613
Παραλλαγή 4 (Καλύτερο παράθυρο)	-	-	-	8.04	30.5848
Παραλλαγή 4 (Median)	-	-	-	8.0776	29.4879
Παραλλαγή 1 (Καλύτερο μοντέλο)	4	6-28-2-1	Tansig-Radbas-Tansig-Tansig	8.6853	23.7691
Καλύτερο τυχαίο	3	6-10-1	Satlins-Satlin-Purelin	10.2043	36.954

Μια σημαντική παρατήρηση είναι επίσης ότι παρά το γεγονός ότι στην παραλλαγή 4, το σύνολο εκπαίδευσης μειώθηκε από 70% σε 50% του αρχικού συνόλου δεδομένων προκειμένου να εκτελεσθεί η ενδιάμεση επικύρωση (σε σύγκριση με τις

παραλλαγές 1-3 που δεν χρησιμοποίησαν αυτό το βήμα), αυτή η μείωση ωφέλησε τελικά τη διαδικασία σχεδιασμού δικτύων, δεδομένου ότι την καθοδήγησε προς τα χαρακτηριστικά που εκφράζουν τις καλύτερες ικανότητες γενίκευσης. Επιπλέον, η ευρωστία της προσέγγισης εμφανίζεται στο γεγονός ότι ακόμη και ο στατιστικός συνδυασμός πολυάριθμων μοντέλων είναι χειρότερος από το μοναδικό καλύτερο υποψήφιο. Ο προαναφερθείς θόρυβος στο σύνολο δεδομένων δεν εμπόδισε την προσέγγιση από την παροχή ικανοποιητικών αποτελεσμάτων.



Σχήμα 14: Πραγματικές και εκτιμώμενες (από το καλύτερο ANN) αποθηκευτικές ανάγκες

Στη γενική διαδικασία δημιουργίας των μοντέλων, καμία ανθρώπινη επέμβαση δεν έγινε, με εξαίρεση τη δημιουργία περιγραφής XML και τη συλλογή του αναγκαίου πειραματικού συνόλου. Κατά συνέπεια ο περιορισμός 4 ικανοποιείται επίσης. Επιπλέον, η προσανατολισμένη στις υπηρεσίες εφαρμογή που παρουσιάστηκε εν περιλήψει στην 2.3.5 (και περιγράφεται λεπτομερώς στο Κεφάλαιο 3) επιτρέπει στην προσέγγιση να εφαρμοστεί σε ένα υπηρεσιοστρεφές σύστημα, ικανοποιώντας κατά συνέπεια τον περιορισμό 5. Η μόνη διαφορά είναι ότι πρέπει να χρησιμοποιηθεί το εργαλείο GNU Octave [46] αντί του Matlab, λόγω της ανικανότητας του τελευταίου να δημιουργήσει

Η σελίδα αυτή είναι σκόπιμα λευκή

ένα εκτελέσιμο πρόγραμμα με την εντολή newff (που απαιτείται για τη δημιουργία ANN).

Για την εκτίμηση του αναγκαίου αποθηκευτικού χώρου της εφαρμογής, εξαιτίας του γεγονότος ότι από την προηγούμενη ανάλυση ήταν εμφανές ότι η παραλλαγή 4 ήταν η καλύτερη προσέγγιση, το αντίστοιχο ANN δημιουργήθηκε άμεσα από αυτή. Τα κορυφαία 3 δίκτυα παρουσιάζονται στον παρακάτω πίνακα (Πίνακας 5). Από αυτά, το καλύτερο ANN για την εκτίμηση αποθήκευσης θεωρείται το πρώτο δεδομένου ότι έχει το λιγότερο μέσο λάθος εκτίμησης στην επικύρωση. Το μέγιστο λάθος είναι υψηλότερο από άλλα αλλά αυτό εμφανίστηκε μόνο για ένα δείγμα επικύρωσης, ενώ οι άλλες τιμές ήταν λιγότερο από 5%. Για αυτό το ANN, μια γραφική σύγκριση μεταξύ των εκτιμώμενων και των πραγματικών τιμών παρουσιάζεται στο Σχήμα 14.

Πίνακας 5: Καλύτερα δίκτυα για πρόβλψη αποθηκευτικού χώρου εφαρμογής FFMPEG

Αριθμός Στρωμάτων	**Νευρώνες/στρώμα**	**Συνάρτηση Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Μέγιστο Απόλυτο Σφάλμα (%)**
3	4-6-1	Tansig-Tansig-Purelin	5.55%	19.50%
3	4-5-1	Tansig-Purelin-Purelin	6.63%	16.69%
3	4-5-1	Tansig-Logsig-Purelin	7.10%	15.40%

Πείραμα 2

Για αυτό το πείραμα, η παραλλαγή 4 της προσέγγισης επιλέχτηκε άμεσα ως τελικός υποψήφιος λόγω της ανώτερης απόδοσής της στο πείραμα 1, το οποίο ήταν το πιό απαιτητικό σενάριο. Είκοσι (20) διαφορετικές εκτελέσεις πραγματοποιήθηκαν, κυρίως λόγω των περιορισμών πρόσβασης δοκιμών.

Πίνακας 6 περιέχει τα καλύτερα ANNs μετά από τον έλεγχο γενίκευσης μαζί με τη δυνατότητα κάθε δικτύου (μέσο και μέγιστο λάθος εκτίμησης) να προβλεφθεί ο χρόνος εκτέλεσης για κάθε παράλληλη εργασία στο σύνολο επικύρωσης.

Πίνακας 6: Καλύτερα ANNs για την πρόβλεψη χρόνου εκτέλεσης παράλληλων εφαρμογών MPI

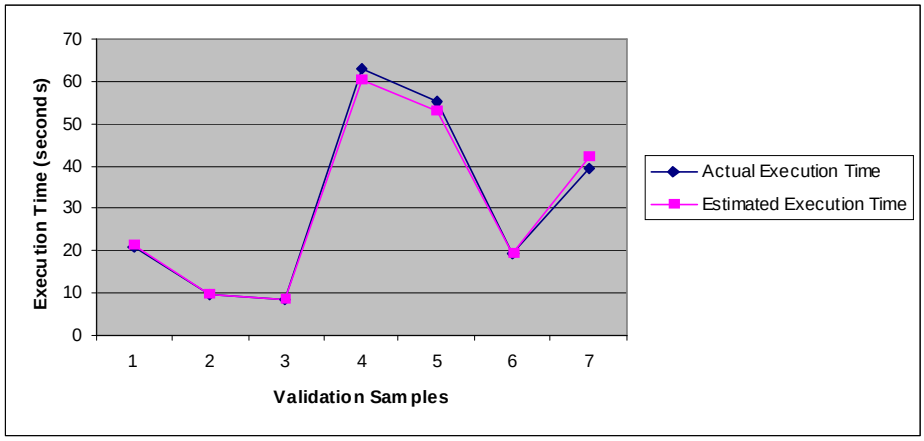
Αριθμός Στρωμάτων	**Νευρώνες/στρώμα**	**Συνάρτηση Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Μέγιστο Απόλυτο Σφάλμα (%)**
3	3-7-1	Tansig-Logsig-Purelin	2.84%	6.30%
3	3-7-1	Tansig-Radbas-Purelin	2.89%	8.40%

Από αυτόν τον πίνακα το καλύτερο ANN είναι το πρώτο, δεδομένου ότι έχει τις καλύτερες μέσες και μέγιστες τιμές λάθους σε σύγκριση με το δεύτερο. Για αυτό το ANN η λεπτομερής απόδοση εκτίμησης ανά δείγμα του συνόλου επικύρωσης εμφανίζεται στο Σχήμα 15. Από αυτές τις μετρήσεις η αποτελεσματικότητα της γενικής λύσης επιβεβαιώνεται, δεδομένου ότι η ακρίβεια της πρόβλεψης ανά εκτέλεση είναι υψηλή.

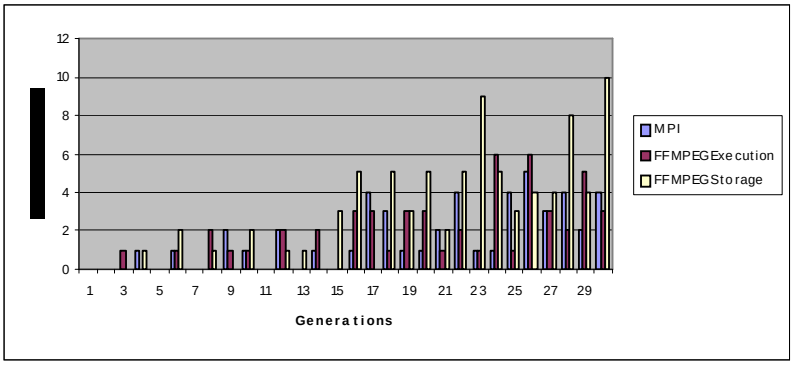
Στο Σχήμα 16 τα σωζόμενα δίκτυα ανά γενεά του ΓΑ εμφανίζονται για την παραλλαγή 4. Σε αυτήν την περίπτωση μπορεί να παρατηρηθεί η επίδραση της διαδικασίας σύγκλισης του ΓΑ. Ενώ στην αρχή (πρώτες 15 γενεές) οι αριθμοί υποψήφιων δικτύων που παράγονται και χαρακτηρίζονται από ένα ικανοποιητικό κριτήριο απόδοσης είναι σχετικά μικροί, αυτό αλλάζει καθώς οι γενεές αυξάνονται. Η έλλειψη σταθερότητας σε αυτήν την διαδικασία οφείλεται στο γεγονός ότι η τυχαιότητα είναι μέχρι ένα σημείο έμφυτη σε μια εξελικτική προσέγγιση λόγω της χρήσης τελεστών όπως η μετάλλαξη και η διασταύρωση (και οι οποίες καθορίζονται από κάποιες τυχαίες παραμέτρους). Αυτό είναι μια άλλη απόδειξη, εκτός από τη σύγκριση με την τυχαία προσέγγιση, ότι ο ΓΑ συγκλίνει καθώς οι γενεές περνούν.

Πείραμα 3

Για το τρίτο πείραμα, προκειμένου να συλλεχθούν οι μετρήσεις, ο αριθμός χρηστών άλλαζε από 1 έως 150, μαζί με την ταχύτητα της CPU, το μεγέθος RAM και τα Q,P (το Q είναι ο υπολογιστικός χρόνος που διατίθεται στην εφαρμογή σε κάθε χρονικό



Σχήμα 15: Πραγματικός και εκτιμώμενος χρόνος εκτέλεσης για την εφαρμογή MPI



Σχήμα 16: Σωζόμενα δίκτυα ανά γενιά ΓΑ

διάστημα P). Ειδικά για τις τιμές Q/P, τέσσερις διαφορετικές τιμές χρησιμοποιήθηκαν (20%, 40%, 60% και 80%) και για την κοκκοποίηση P ένα διάστημα από 10 έως 600 χιλιοστά του δευτερολέπτου. Συνολικά, 420 διαφορετικές εκτελέσεις πραγματοποιήθηκαν. Για κάθε εκτέλεση εφαρμόστηκε μια δειγματοληψία περίπου 800

Η σελίδα αυτή είναι σκόπιμα λευκή

τιμών για τους χρόνους απόκρισης των αιτημάτων που εξυπηρετήθηκαν. Από αυτά τα δείγματα ο μέσος χρόνος και η τυπική απόκλιση εξήχθησαν για κάθε εκτέλεση και ήταν οι τιμές στόχων για τα μοντέλα που εκπαιδεύονται. Περισσότερες λεπτομέρειες σχετικά με τη δοκιμή δειγματοληψίας μπορούν να βρεθούν στο [36]. Ακολουθούν ενδεικτικές γραφικές παραστάσεις (Σχήμα 17, Σχήμα 18,Σχήμα 19 και Σχήμα 20) που απεικονίζουν την εξάρτηση των παραμέτρων υπηρεσίας (χρόνος απόκρισης) του εξυπηρετητή από τις παραμέτρους φορτίου (χρήστες) και υλικού (ρύθμιση χρονοπρογραμματιστή).

Για την δημιουργία των μοντέλων ακολουθήθηκε η καλύτερη έκδοση που παρήχθη από το πείραμα 1. Τα καλύτερα δίκτυα που παρήχθησαν από αυτό το πείραμα εμφανίζονται στους ακόλουθους πίνακες (Πίνακας 7 και Πίνακας 8).

Πίνακας 7: Καλύτερα ANNs πρόβλεψης χρόνου απόκρισης εξυπηρετητή e-learning

Αριθμός Στρωμάτων	**Νευρώνες/στρώμα**	**Συνάρτηση Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Μέγιστο Απόλυτο Σφάλμα (%)**
5	5-3-3-2-1	Tansig-Purelin-Tansig-Tansig-Purelin	1.93%	17.89%
3	5-3-1	Tansig-Tansig-Tansig	3.71%	31.09%
4	5-2-2-1	Tansig-Tansig-Logsig-Tansig	3.91%	25.12%

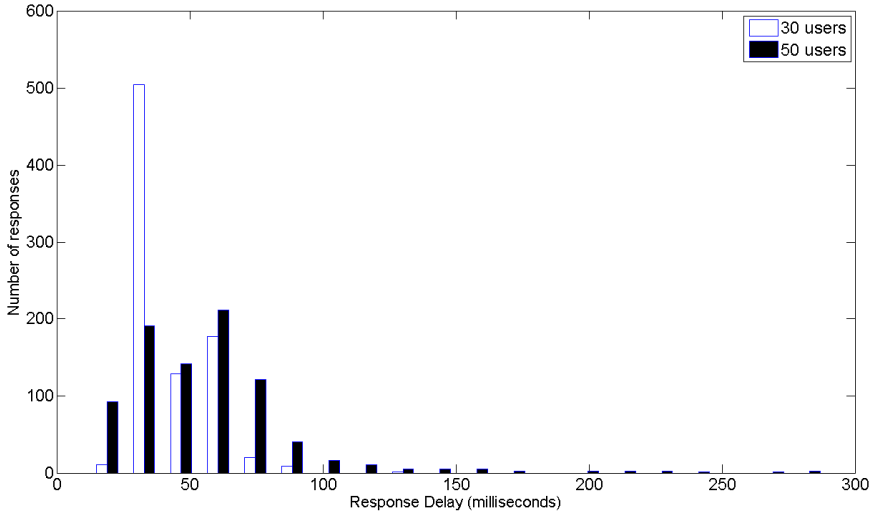
Πίνακας 8: Καλύτερα ANNs πρόβλεψης τυπικής απόκλισης εξυπηρετητή e-learning

Αριθμός Στρωμάτων	**Νευρώνες/στρώμα**	**Συνάρτηση Μεταφοράς ανά στρώμα**	**Μέσο Απόλυτο Σφάλμα (%)**	**Μέγιστο Απόλυτο Σφάλμα (%)**
3	5-15-1	Tansig-Tansig-Tansig	3.56%	22.27%
3	5-16-1	Tansig-Tansig-Purelin	3.80%	19.92%
3	5-2-1	Tansig-Tansig-Tansig	3.88%	23.19%

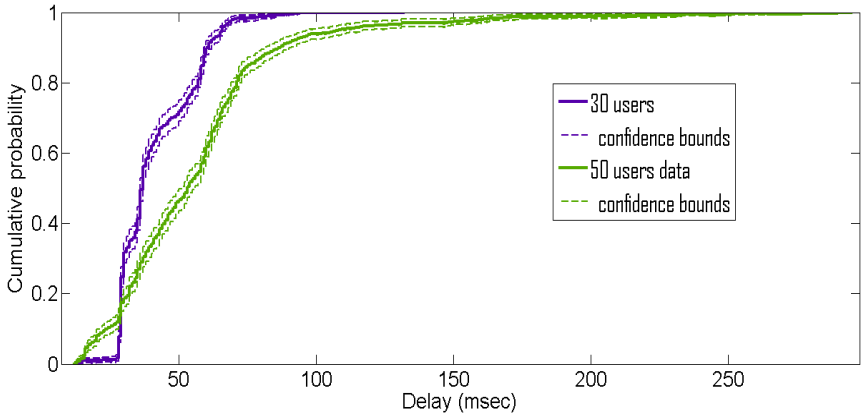
Από αυτά τα στοιχεία είναι εμφανές ότι πάλι έχουν παραχθεί αυτόματα πολύ ακριβή δίκτυα. Οι μέγιστες τιμές μπορούν να εμφανίζονται υψηλές, αλλά αυτές παρατηρούνται σε πολύ σπάνιες περιπτώσεις για το σύνολο δεδομένων επικύρωσης. Το % λάθος των προβλέψεων για κάθε μια από τις περιπτώσεις επικύρωσης για το καλύτερο δίκτυο παρουσιάζεται στο Σχήμα 21.

Ένα άλλο συμπέρασμα μπορεί να εξαχθεί όσον αφορά το όφελος χρήσης του χρονοπρογραμματιστή πραγματικού χρόνου. Εξαιτίας του γεγονότος ότι το ποσοστό της CPU που διατέθηκε στην εφαρμογή ήταν εγγυημένο, ο θόρυβος από την εκκίνηση άλλων διαδικασιών μειώθηκε σημαντικά, βοηθώντας το μηχανισμό στην παραγωγή ακριβέστερων αποτελεσμάτων.

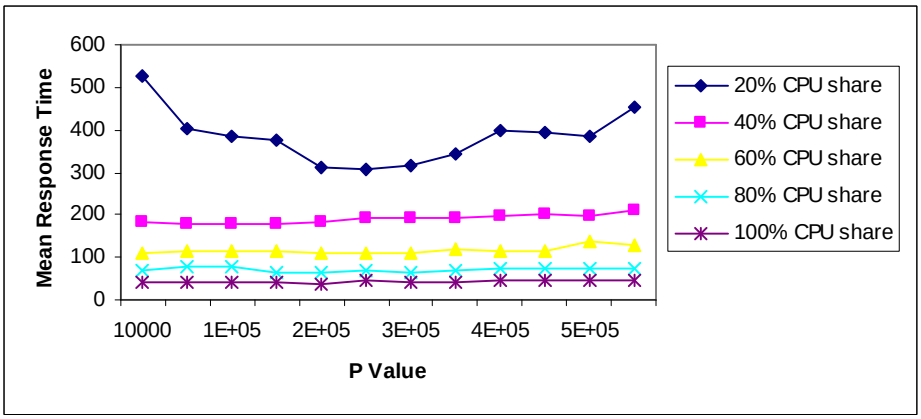
Σε αυτό το σημείο πρέπει να τονιστεί ότι η πρόβλεψη μέσου χρόνου απόκρισης μπορεί να αντικατασταθεί ή να συμπληρωθεί με οποιαδήποτε άλλη επιθυμητή μετρική, όπως όρια για το 95% των χρόνων απόκρισης του εξυπηρετητή. Το μόνο που αλλάζει είναι η προεπεξεργασία των πειραματικών στοιχείων για την παροχή των κατάλληλων δεδομένων. Από τα πειράματα φαίνεται ότι με αυτήν την προσέγγιση οι έξοδοι KPI μπορούν να προβλεφθούν με αυξημένη ακρίβεια για κάθε μεμονωμένη περίπτωση εκτέλεσης, λαμβάνοντας υπόψη την επιρροή των παραμέτρων φόρτου της εφαρμογής και των τιμών των φυσικών πόρων. Αυτή η πρόβλεψη μπορεί να ληφθεί, με έναν γενικό τρόπο, που οδηγεί στις βελτιωμένες προκρατήσεις των πόρων από το PaaS στο στρώμα IaaS. Το γεγονός ότι η επίδραση στα επίπεδα QoS μπορεί να προβλεφθεί με αυτή την ακρίβεια μπορεί να βοηθήσει τους προμηθευτές στην επιλογή των καταλληλότερων πόρων για κάθε περίπτωση εκτέλεσης της εφαρμογής.



Σχήμα 17: Κατανομή μετρήσεων στους χρόνους απόκρισης για 30 και 50 χρήστες

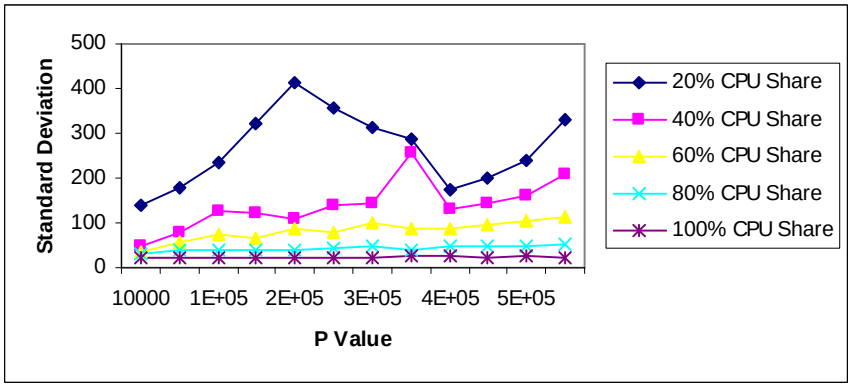


Σχήμα 18: Καμπύλη CDF για τους χρόνους απόκρισης 30 και 50 χρηστών

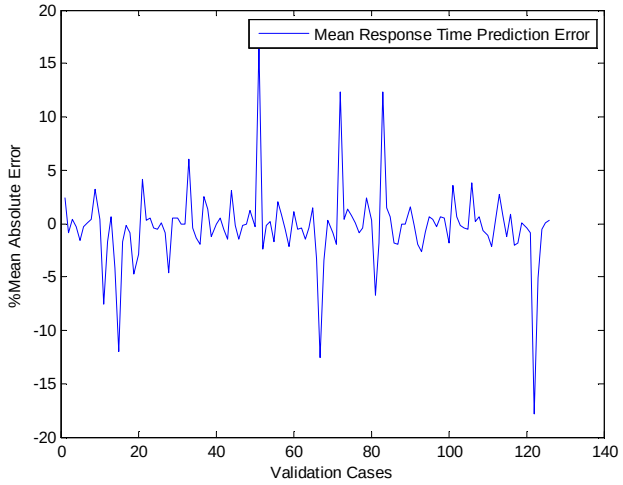


Σχήμα 19: Μέσος χρόνος απόκρισης για 110 χρήστες και μεταβαλλόμενα ποσοστά CPU και περιόδους χρονοπρογραμματισμού P

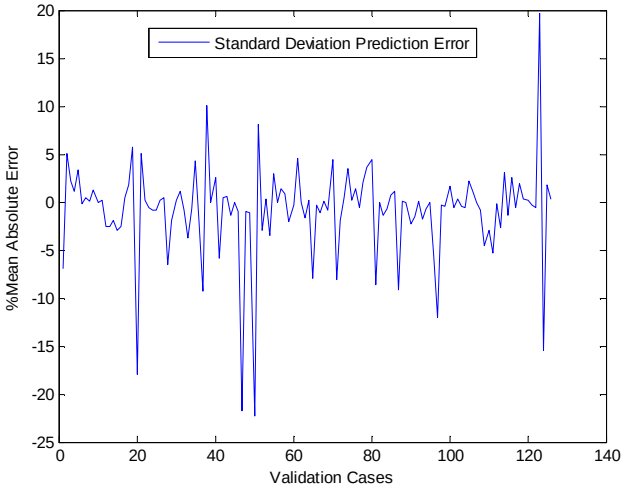
Η σελίδα αυτή είναι σκόπιμα λευκή



Σχήμα 20: Τυπική απόκλιση για 90 χρήστες και μεταβαλλόμενα ποσοστά CPU και περιόδους χρονοπρογραμματισμού P



(α)



(β)

Σχήμα 21: Μέσο απόλυτο σφάλμα πρόβλεψης (%) για το μέσο χρόνο απόκρισης (α) και τη τυπική απόκλιση (β) σε κάθε περίπτωση επικύρωσης

Η σελίδα αυτή είναι σκόπιμα λευκή

2.8 Συμπεράσματα

Σε αυτό το κεφάλαιο ερευνήθηκε μια αυτοματοποιημένη διαδικασία, μέσω της οποίας τα χαρακτηριστικά των φυσικών πόρων συσχετίζονται με παραμέτρους φόρτου εργασίας επιπέδου εφαρμογής προκειμένου να καθοριστούν τα γενικά παρεχόμενα επίπεδα KPI για μια εφαρμογή που εκτελείται σε ένα υπηρεσιοστρεφές πλαίσιο. Τα μοντέλα πρόβλεψης δημιουργήθηκαν μέσω μιας υβριδικής νευρο-γενετικής δομής και μετά από διερεύνηση διάφορων παραλλαγών αυτής. Με αυτήν την μεθοδολογία τα αποτελέσματα ήταν πολύ ενθαρρυντικά από άποψη ακρίβειας πρόβλεψης για κάθε μεμονωμένη περίπτωση εκτέλεσης, οδηγώντας κατά συνέπεια στη σημαντικά μικρότερη ανάγκη για δέσμευση πόρων. Στα πειράματα που πραγματοποιήθηκαν, οι παράμετροι ήταν αντιπροσωπευτικές για να απεικονίσουν ότι ένας ή περισσότεροι όροι/χαρακτηριστικά εφαρμογής μπορούν να συνδυαστούν με μια ή περισσότερες παραμέτρους υλικού προκειμένου να προβλεφθεί η επίδραση στα γενικά επίπεδα εφαρμογής KPI. Η μέθοδος ήταν επίσης γενική και ήταν σε θέση να χαρτογραφήσει οποιοδήποτε είδος αριθμητικών εισόδων σε οποιοδήποτε είδος αριθμητικών εξόδων χωρίς την ανάγκη για λεπτομερή γνώση του κώδικα πηγής.

Ένα άλλο βασικό σημείο είναι η αφαίρεση της ανάγκης για εμπειρογνώμονα τεχνητής νοημοσύνης που θα δημιουργήσει και θα καθορίσει με ακρίβεια το απαραίτητο ANN. Με την προτεινόμενη εφαρμογή οι παράμετροι σχεδίου δικτύων αποφασίζονται αυτόματα, χωρίς οποιαδήποτε ανάγκη για τη γνώση σχεδιασμού ANN, σε έναν σχετικά μικρό χρόνο και με ακριβή αποτελέσματα, καθιστώντας το κατά συνέπεια κατάλληλο για χρήση σε ένα προσανατολισμένο προς τις υπηρεσίες σύστημα. Κατά συνέπεια

ικανοποιούνται όλοι οι περιορισμοί που υπάρχουν στο τελευταίο και παρουσιάζονται στην εισαγωγή.

Για το μέλλον, ο στόχος είναι να βελτιωθεί η υλοποίηση προκειμένου να περιληφθούν όλες οι πιθανές παράμετροι μιας διαμόρφωσης ANN, όπως οι αλγόριθμοι εκπαίδευσης. Επίσης υπάρχει μια σειρά άλλων παραμέτρων που απαιτούν διερεύνηση και είναι δυνατόν να επηρεάζουν το χρόνο εκτέλεσης μιας εφαρμογής, όπως ο φόρτος εργασίας του κοινού κόμβου όπου αυτή θα εκτελεστεί, από την πλευρά του στρώματος IaaS.

3 *Υπηρεσιοστρεφές Πλαίσιο Λειτουργίας της Πρόβλεψης Απόδοσης*

Στο παρόν κεφάλαιο περιγράφεται το υπηρεσιοστρεφές πλαίσιο μέσω του οποίου τα μοντέλα πρόβλεψης απόδοσης μπορούν να χρησιμοποιηθούν σε περιβάλλοντα Υπολογιστικών Νεφών. Παρουσιάζεται η αρχιτεκτονική του συστήματος, οι λεπτομέρειες υλοποίησής του καθώς και μία μελέτη πάνω στην βελτιστοποίηση της απόδοσής του.

3.1 Ορισμός του Προβλήματος

Τα τελευταία χρόνια, οι σύγχρονες υποδομές ΤΠ (Τεχνολογία Πληροφορίας) έχουν μετατοπιστεί προς ένα υπηρεσιοστρεφές μοντέλο, που ωθείται από τεχνολογίες όπως οι υπηρεσιοστρεφείς αρχιτεκτονικές (SOAs) [2] και τα Υπολογιστικά Νέφη [47]. Σε αυτό το πλαίσιο, όλα τείνουν να προσφερθούν ως υπηρεσία, είτε είναι υποδομές, πλατφόρμες ή λογισμικό, όπως υπαγορεύει το μοντέλο SPI [48] (Software, Platform, Infrastructure).

Η λειτουργία αυτών των προσανατολισμένων στις υπηρεσίες υποδομών (Service Oriented Infrastructures-SOIs) ρυθμίζεται, όπως αναφέρθηκε στην εισαγωγή, μέσω των συμφωνιών επιπέδων υπηρεσιών (Service Level Agreements-SLAs [6]), που καθορίζουν το ρόλο των συμβαλλόμενων μερών στη συναλλαγή και τις λεπτομέρειες αυτής. Τέτοιες λεπτομέρειες μπορούν να είναι η υπηρεσία που θα προσφερθεί, η ποιότητα των παραμέτρων της (Quality of Service-QoS) καθώς επίσης και νομικά ζητήματα όπως η αποζημίωση σε περίπτωση που αυτά τα επίπεδα QoS δεν καλύπτονται από τον προμηθευτή. Η διαχείριση QoS είναι γενικά ένας πολύ σημαντικός και προκλητικός τομέας της έρευνας στα κατανεμημένα περιβάλλοντα.

Ένας κρίσιμος στόχος για τους παρόχους υπηρεσιών πριν από την υπογραφή του συγκεκριμένου SLA είναι η εκτίμηση των πόρων που απαιτούνται για να ικανοποιηθούν οι απαιτήσεις χρηστών για κάθε εφαρμογή που προσφέρεται ως υπηρεσία από την πλατφόρμα τους. Σε αυτήν την κατεύθυνση, η πρόβλεψη απόδοσης και η μοντελοποίηση είναι απαραίτητες για τον πάροχο προκειμένου να καθοριστούν οι απαραίτητοι πόροι για την ικανοποίηση των απαιτήσεων QoS της εφαρμογής και συγχρόνως να μεγιστοποιηθεί η χρησιμοποίηση των πόρων.

Μέχρι τώρα, εναλλακτικές λύσεις έχουν εφαρμοστεί προκειμένου να αντιμετωπίσουν αυτό το πρόβλημα, περιλαμβάνοντας μεθόδους γράφων ([8],[12]), μηχανικής μάθησης ([13][15]) και πιθανοτήτων [14]. Γενικά, ένα μοντέλο δημιουργείται προκειμένου να προβλεφθεί η έξοδος από την είσοδο, συνήθως με το να λάβει υπόψη τις κατάλληλες παραμέτρους ή με την ανάλυση των προηγούμενων ιστορικών στοιχείων. Η βάση των μοντέλων μπορεί να είναι γραμμικές συναρτήσεις, νευρωνικά δίκτυα, δέντρα ταξινόμησης και παλινδρόμησης ή στατιστικές ιδιότητες.

Αυτό που λείπει είναι ένα συγκεκριμένο πλαίσιο για τέτοιες μεθόδους, ειδικά σε ένα προσανατολισμένο προς τις υπηρεσίες περιβάλλον. Χρήσιμα και ευρέως διαδεδομένα εργαλεία, όπως το GNU Octave [46] καθιστούν την συγγραφή τέτοιων μεθόδων πολύ ευκολότερη σε σχέση με τις τυποποιημένες γλώσσες προγραμματισμού όπως η Java ή η C++. Το Octave είναι ένα λογισμικό αριθμητικών υπολογισμών ανοιχτού κώδικα, πολύ παρόμοιο με το Matlab. Μέσω αυτού του εργαλείου, οι προηγμένες μέθοδοι εκτίμησης μπορούν να γραφούν με τη μορφή script, κάνοντας την ανάπτυξη ή αντικατάσταση μεθόδων πολύ ευκολότερη και γρήγορη. Εντούτοις η αυτοματοποιημένη προσαρμογή λογισμικού όπως το Octave στα υπηρεσιοστρεφή περιβάλλοντα χρειάζεται ένα εξειδικευμένο πλαίσιο διαχείρισης. Μια πολύ λεπτομερής ανάλυση στα διάφορα σχέδια αυτού του είδους μπορεί να βρεθεί στο [59].

Ο στόχος αυτού του κεφαλαίου είναι να παρουσιαστεί ένα εύκαμπτο, γενικό και εύρωστο πλαίσιο υπηρεσιών για την εκτίμηση απόδοσης σε υπηρεσιοστρεφείς υποδομές (SOIs). Αυτό το πλαίσιο (IRMOS Mapping Service) είναι σε θέση να εφαρμόσει οποιαδήποτε μέθοδο εκτίμησης ως pluggable Octave script. Για αυτόν τον λόγο, όλες οι λειτουργίες που απαιτούνται από μια τέτοια υπηρεσία εξετάζονται, όπως η XML-βασισμένη εξαγωγή κρίσιμων πληροφοριών για μια εφαρμογή, η ενσωμάτωση του λογισμικού Octave για τη χρήση στον κύκλο της ζωής υπηρεσιών και η ανάκτηση των ιστορικών στοιχείων από τις βάσεις δεδομένων, ένα κύριο χαρακτηριστικό ενδιαφέροντος δεδομένου ότι όλες οι μέθοδοι χρησιμοποιούν τα τελευταία. Επιπλέον επιχειρείται μια λεπτομερής ανάλυση απόδοσης της συμπεριφοράς της υπηρεσίας, προκειμένου να προσδιοριστούν και να μετριαστούν οι αδυναμίες της. Για αυτόν τον λόγο, το βαρύ υπολογιστικό φορτίο της υπηρεσίας εξάγεται σε μια υποδομή νέφους,

προκειμένου να είναι εγγυημένες η εξελιξιμότητα, οι χρόνοι απόκρισης και η ομαλή συμπεριφορά. Διαφορετικοί τρόποι λειτουργίας διερευνώνται, βασισμένοι στην ανάλυση απόδοσης και μια μαθηματική ανάλυση σχετικά με τα όρια των ποσοστών άφιξης αιτημάτων.

Επιπλέον, χάρη στη διαστρωματωμένη και αποζευγμένη αρχιτεκτονική που έχει ακολουθηθεί για την εφαρμογή του μηχανισμού, παρουσιάζεται ο τρόπος με τον οποίο οι διαφορετικές τεχνολογίες μπορούν να εναλλαχθούν προκειμένου να ενσωματωθούν εξελίξεις, παραδείγματος χάριν στα πρότυπα και τα πρωτόκολλα υπηρεσιών Ιστού (Web Services-WS).

Το υπόλοιπο του κεφαλαίου είναι δομημένο ως εξής: η παράγραφος 3.2 παρουσιάζει τη σχετική εργασία στον τομέα της προσανατολισμένης στις υπηρεσίες εκτίμησης απόδοσης και της προσφοράς μαθηματικού λογισμικού μέσω των υπηρεσιών Ιστού/πλέγματος. Το επόμενο τμήμα (παράγραφος 3.3) περιγράφει λεπτομερώς τη λειτουργία της υπηρεσίας μαζί με τις συγκεκριμένες υποστηρικτικές ενέργειες ενώ η παράγραφος 3.4 εισάγει την προσέγγιση στρώματος υπηρεσιών και την υλοποίηση. Στην παράγραφο 3.5 αξιολογείται η λειτουργία της υπηρεσίας για τον εφαρμοσμένο μηχανισμο γενετικά βελτιστοποιημένων ΤΝΔ, όπως αυτός περιγράφηκε στα προηγούμενα κεφάλαια. Επιπλέον, παρουσιάζεται μια σειρά πειραμάτων (παράγραφος 3.6) προκειμένου να επικυρωθεί η απόδοση του πλαισίου υπηρεσιών ή να ρυθμιστεί εκ νέου η αρχιτεκτονική για λόγους βελτιστοποίησης (παράγραφος 3.7). Τέλος, η παράγραφος 3.8 ολοκληρώνει το κεφάλαιο με μια συζήτηση σχετικά με τη μελλοντική έρευνα και τις δυνατότητες για την τρέχουσα μελέτη.

3.2 Σχετικές Εργασίες

Διάφορες προσεγγίσεις μπορούν να συγκριθούν με την εργασία που περιγράφεται σε αυτό το κεφάλαιο. Αυτές αφορούν είτε τους προσανατολισμένους στις υπηρεσίες μηχανισμούς πρόβλεψης απόδοσης είτε τις προσφορές μαθηματικού λογισμικού στα κατανεμημένα περιβάλλοντα.

Το Network Weather Service [22], το οποίο αναφέρθηκε και στο προηγούμενο κεφάλαιο όσον αφορά τον πυρήνα της μεθόδου πρόβλεψης, περιλαμβάνει ένα σύνολο προγνωστών και αισθητήρων σε ένα κατανεμημένο περιβάλλον, προκειμένου να ανακτηθούν τα δεδομένα δυναμικά και να τροφοδοτηθούν στις ενότητες που εφαρμόζουν τις προηγμένες τεχνικές πρόβλεψης. Αν και αυτή η μέθοδος ήταν καινοτόμος, βασίστηκε στη γλώσσα C, καθιστώντας την εφαρμογή των χρησιμοποιούμενων τεχνικών δυσκολότερη και χρονοβόρα.

Το GridSolve [56] υπογραμμίζει την ευκολία για το χρήστη και περιλαμβάνει τον έλεγχο των πόρων, το χρονοπρογραμματισμό και την ανοχή σφαλμάτων σε επίπεδο υπηρεσίας. Εκτός από πελάτες FORTRAN και C, το GridSolve επιτρέπει στα επιστημονικά υπολογιστικά περιβάλλοντα (όπως το Matlab) να χρησιμοποιηθούν ως πελάτες, έτσι ώστε οι επιστήμονες να μπορούν να χρησιμοποιήσουν τους πόρους πλέγματος μέσα από τα προτιμώμενα περιβάλλοντά τους. Αυτή η προσπάθεια επιτυγχάνει οι εφαρμογές Matlab ή Octave να είναι σε θέση να εκμεταλλευτούν τους κατανεμημένους πόρους. Αυτό γίνεται επίσης για το Maple στο [57] και για ποικίλα πακέτα άλγεβρας υπολογιστών στο [60]. Ενώ αυτές οι εργασίες είναι σημαντικές, αμφισβητείται ο τρόπος χρησιμοποίησης των παραγόμενων υπηρεσιών στην διάρκεια του κύκλου ζωής ενός ενσωματωμένου, προσανατολισμένου στις υπηρεσίες

περιβάλλοντος. Είναι δυνατόν να θεωρηθούν κυρίως ως επιτυχής προσπάθεια να βελτιωθεί η απόδοση του λογισμικού μέσω των κατανεμημένων υποδομών. Άλλες εργασίες όπως η [51] έχουν σαν στόχο την προσφορά του εξουσιοδοτημένου λογισμικού μέσω των υποδομών Νέφους κυρίως για λόγους χορήγησης αδειών και σε pay-per-use βάση.

Στο [55] ακολουθείται μια αρχιτεκτονική πολύ παρόμοια με το σχέδιο που παρουσιάζεται εδώ, προκειμένου να παρασχεθούν μαθηματικές υπηρεσίες Ιστού βασισμένες στο λογισμικό Mapple. Αν και καινοτόμος, αυτή η προσέγγιση δεν ενσωματώνεται σε ένα πλέγμα/περιβάλλον Νέφους και εστιάζει κυρίως στην προσφορά των μαθηματικών υπηρεσιών. Στο πρόγραμμα GENSS [58], η κύρια εστίαση βρίσκεται στις τεχνικές αντιστοιχίας για την ανακάλυψη των μαθηματικών υπηρεσιών, από σημασιολογική άποψη. Οι συντάκτες του [62] καταδεικνύουν τη χρήση του Octave για τη δημιουργία των μοντέλων απόδοσης για τις διεπαφές δικτύων βασισμένες σε ουρές αναμονής, ενώ στο [63] ένα περιορισμένο μέρος των βιβλιοθηκών του Octave προσφέρεται μέσω των διεπαφών υπηρεσιών Ιστού για τη χρήση των λειτουργιών επεξεργασίας σήματος σε κινητά τηλέφωνα.

Μια πολύ ενδιαφέρουσα εργασία παρουσιάζεται στο [49]. Σε αυτήν την προσέγγιση, η κύρια προσοχή δίνεται στις διαφορές ανάμεσα στις SOA-based και τις κανονικές εφαρμογές και τις απαιτούμενες τροποποιήσεις προκειμένου οι τελευταίες να είναι αποτελεσματικότερες. Μια από αυτές είναι η ανάλυση απόδοσης που είναι απαραίτητη για την προσανατολισμένη στις υπηρεσίες εφαρμογή. Στο [74], ένα σύστημα βασισμένο σε κλειστό βρόχο ανατροφοδοτεί τα KPIs της εφαρμογής προκειμένου να οδηγήσει τη διαχείριση των πόρων. Αυτό είναι μια ιδανική περίπτωση όταν δεν

απαιτείται SLA για να δεσμεύσει a priori τους πόρους, αλλά η εφαρμογή μπορεί να ρυθμίσει από μόνη της τους πόρους της, χωρίς αυστηρούς χρονικούς περιορισμούς. Στο [75] χρησιμοποιείται ένα πλαίσιο απόδοσης βασισμένο σε benchmarking και στατιστικά συμπεράσματα για τις υπηρεσίες που είναι προσανατολισμένες σε μηνύματα. Μια λεπτομερής έρευνα που αφορά συστήματα απόδοσης για component-based λογισμικά μπορεί να βρεθεί στο [50].

Η αρχιτεκτονική σχεδίαση που παρουσιάζεται σε αυτό το κεφάλαιο στοχεύει στην παροχή του προσανατολισμένου στις υπηρεσίες πλαισίου για τις μεθόδους πρόβλεψης απόδοσης με έναν γενικό και απλοποιημένο τρόπο. Το τελευταίο είναι ιδιαίτερα σημαντικό προς χρήση από τους ανθρώπους που είναι δυνατόν να είναι εμπειρογνώμονες στο πεδίο εκτίμησης απόδοσης αλλά δεν εμπλέκονται άμεσα με τα ζητήματα και την ανάπτυξη SOIs. Επιπλέον, η σχεδιασμένη υπηρεσία μπορεί να χρησιμοποιηθεί σε αυτοματοποιημένα περιβάλλοντα, π.χ. κατά τη διάρκεια της διαδικασίας διαπραγμάτευσης SLA μεταξύ ενός φορέα παροχής υπηρεσιών και του πελάτη. Τα αποτελέσματα ανάλυσης απόδοσης μπορούν να χρησιμοποιηθούν επίσης σε οποιαδήποτε παρόμοια προσέγγιση, που ενσωματώνει το εξειδικευμένο λογισμικό στο κατώτατο σημείο μιας προσανατολισμένης προς τις υπηρεσίες προσέγγισης.

3.3 Υπηρεσιοστρεφές Πλαίσιο Εκτίμησης Απόδοσης IRMOS Mapping service

Το υπηρεσιοστρεφές πλαίσιο εκτίμησης εκτέλεσης που παρουσιάζεται σε αυτό το κεφάλαιο (Υπηρεσία Χαρτογράφησης- Mapping Service) χρησιμοποιείται στην πλατφόρμα IRMOS [39], η οποία προσφέρει υπηρεσιοστρεφείς υποδομές για εφαρμογές

πραγματικού χρόνου. Για να συμπεριληφθεί μια εφαρμογή στον κατάλογο του φορέα παροχής υπηρεσιών IRMOS, πρέπει να ακολουθηθούν διάφορα βήματα προκειμένου να την προσαρμόσουν στο πλαίσιο. Εν συντομία, ο υπεύθυνος για την ανάπτυξη εφαρμογής πρέπει να δημιουργήσει μια περιγραφή XML της, συμπεριλαμβανομένου ενός συνόλου χαρακτηριστικών για τις λειτουργικές και μη λειτουργικές απαιτήσεις της, όπως αναφέρθηκε στο Κεφάλαιο 2.4.1. Αυτή η περιγραφή καλείται «περιγραφή τμημάτων υπηρεσιών εφαρμογής» (Application Service Component Description-ASCD). Κατόπιν, ο φορέας παροχής υπηρεσιών (PaaS) πρέπει να περάσει από τη διαδικασία δημιουργίας κανόνων αντιστοίχισης, που χρησιμοποιούνται για να συνδέουν τις απαιτήσεις χρηστών, τα χαρακτηριστικά της εφαρμογής και τους πόρους που απαιτούνται προκειμένου να ικανοποιηθούν τα επίπεδα QoS που θα συμφωνηθούν σε ένα SLA. Το σχέδιο και η εφαρμογή που ακολουθήθηκαν είναι γενικά, επιτρέποντας κατά συνέπεια στην υπηρεσία αντιστοίχισης να χρησιμοποιηθεί ακόμη και εκτός της προαναφερθείσας πλατφόρμας. Η γενική αρχιτεκτονική της πλατφόρμας IRMOS περιγράφεται στο [64]. Σε αυτό το σημείο πρέπει να τονιστεί ότι ενώ τα συμφέροντα του υπεύθυνου για την ανάπτυξη εφαρμογής και του φορέα παροχής υπηρεσιών φαίνονται αντιφατικά (ο πάροχος της πλατφόρμας μπορεί π.χ. να εμφανίζει την εφαρμογή ως πιο απαιτητική ώστε να πολλαπλασιάσει τους χρησιμοποιούμενους πόρους και άρα την τιμή), αυτό δεν ισχύει λόγω της αλυσίδας αξίας που ακολουθείται. Ο υπεύθυνος για την ανάπτυξη της εφαρμογής προσαρμόζει την τελευταία στην πλατφόρμα IRMOS, προκειμένου να χρησιμοποιηθεί από άλλα συμβαλλόμενα μέρη (πελάτες), μέσω του μοντέλου SaaS. Αντίστοιχα όμως και άλλες παρόμοιες εφαρμογές μπορούν να προσφέρονται μέσω του ίδιου παρόχου. Έτσι είναι προς το συμφέρον του να βοηθήσει την πλατφόρμα για να προβλέψει σωστά τους

ελάχιστους απαραίτητους πόρους, έτσι ώστε η εφαρμογή να τρέχει ομαλά, προκειμένου να είναι ανταγωνιστική σε σύγκριση με τις παρόμοιες εφαρμογές. Επιπλέον, μπορούν να υπάρξουν προμηθευτές περισσότερων από μιας πλατφορμών IRMOS, οπότε για τον ίδιο λόγο (ανταγωνιστικότητα) ο πάροχος θέλει να επιτύχει ακριβείς προβλέψεις ώστε να ελαχιστοποιήσει το κόστος της προσφοράς του προς τον πελάτη.

Προκειμένου να σχεδιαστεί μια τέτοια υπηρεσία, πρέπει να ληφθούν υπόψη όλες οι διαφορετικές λειτουργίες που είναι απαραίτητες για να υπάρξει ένα πλήρως λειτουργικό πλαίσιο. Οι υψηλού επιπέδου λειτουργίες που πρέπει να παρασχεθούν είναι

α) δημιουργία μοντέλου πρόβλεψης ανά ASCD βασισμένη στην χρησιμοποιούμενη μέθοδο εκτίμησης

β) διάθεση μοντέλου προς την πλατφόρμα για τη διαδικασία εκτίμησης κατά την online λειτουργία.

Επιπλέον, αυτές οι διαδικασίες πρέπει να είναι πλήρως αυτόματες προκειμένου να ενταχθούν στην υπηρεσιοστρεφή υποδομή.

3.3.1 Φάση Δημιουργίας Μοντέλου (Create Model)

Για τη δημιουργία ενός μοντέλου είναι απαραίτητα τα παρακάτω βήματα:

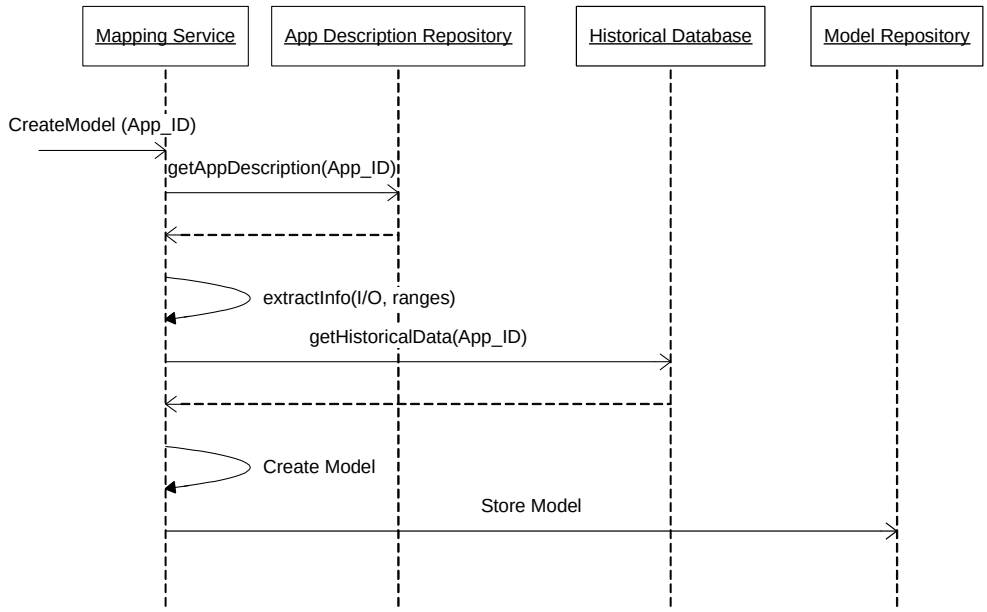
- Αίτηση προς την υπηρεσία, συμπεριλαμβανομένου ενός συγκεκριμένου προσδιοριστικού για την εφαρμογή της οποίας το μοντέλο πρόκειται να δημιουργηθεί
- Απόκτηση των πληροφοριών που παρείχε ο υπεύθυνος ανάπτυξης σχετικά με την εφαρμογή (περιλαμβάνει τις εισόδους και εξόδους του μοντέλου, το εύρος τιμών τους κ.λπ.)
- Απόκτηση των ιστορικών στοιχείων από μια βάση δεδομένων που πρόκειται να χρησιμοποιηθούν ως dataset για τη μέθοδο πρόβλεψης

- Διαβίβαση των παραμέτρων στο Octave
- Octave script που υλοποιεί την συγκεκριμένη μέθοδο και η εκτέλεσή του στον κύκλο ζωής της υπηρεσίας
- Αποθήκευση των μοντέλων για μελλοντική χρήση

Ένα βασικό σημείο εδώ είναι η εξαγωγή των πληροφοριών σχετικά με την εφαρμογή. Αυτό μπορεί να γίνει μέσω ενός σχήματος XML, το οποίο γίνεται concrete από τον υπεύθυνο εφαρμογής και περιέχει τις κρίσιμες πληροφορίες σχετικά με τις εισόδους και τις εξόδους του λογισμικού. Αυτό που θεωρείται ως κρίσιμη πληροφορία είναι οι είσοδοι ή τα χαρακτηριστικά που επηρεάζουν την απόδοση μιας εφαρμογής και θεωρούνται ως προβλέπτες και οι έξοδοι που χαρακτηρίζουν την ποιότητα της προσφερόμενης υπηρεσίας (QoS levels). Επιπλέον, για αυτά τα στοιχεία, το εύρος των πιθανών τιμών πρέπει να προσδιοριστεί ή οι διακριτές τιμές τους σε περίπτωση που είναι διακριτά μεγέθη. Αυτό γίνεται επειδή σε ποικίλες μεθόδους απαιτείται η κανονικοποίηση των δεδομένων σε κοινό διάστημα (συνήθως το [0,1] ή το [-1,1]). Για τη μετατροπή αυτή οι τιμές πρέπει να περιγραφούν κατά σειρά σπουδαιότητας (ascending or descending order). Το χρησιμοποιούμενο σχήμα περιγράφεται στο [30]. Για αυτόν τον λόγο, η υπηρεσία πρέπει να είναι σε θέση να επεξεργαστεί την περιγραφή XML και να εξαγάγει όλες τις απαραίτητες πληροφορίες για την εφαρμογή.

Από τα προηγούμενα βήματα, τα απαραίτητα συστατικά της υπηρεσίας μπορούν να προσδιοριστούν. Αυτά περιλαμβάνουν το κεντρικό μέρος (Mapping Service) που είναι αρμόδιο για το συντονισμό της ολόκληρης διαδικασίας και των κύριων στόχων επεξεργασίας. Η αποθήκη περιγραφής (App. Description Repository) περιέχει τις περιγραφές XML των εφαρμογών, ενώ η βάση ιστορικών δεδομένων (Historical

database) χρησιμοποιείται για να αποθηκεύσει τα στοιχεία από τις προηγούμενες εκτελέσεις. Αυτά μπορούν να προέλθουν από τα ήδη υπάρχοντα ιστορικά στοιχεία ή από ειδική διαδικασία δειγματοληψίας. Τέλος, η αποθήκη κανόνων/μοντέλων (Model Repository) περιέχει τους αποθηκευμένους κανόνες μετά από την περάτωση αυτής της λειτουργίας (model creation). Το διάγραμμα ακολουθίας για τη λειτουργία CreateModel της υπηρεσίας εμφανίζεται στο Σχήμα 22.



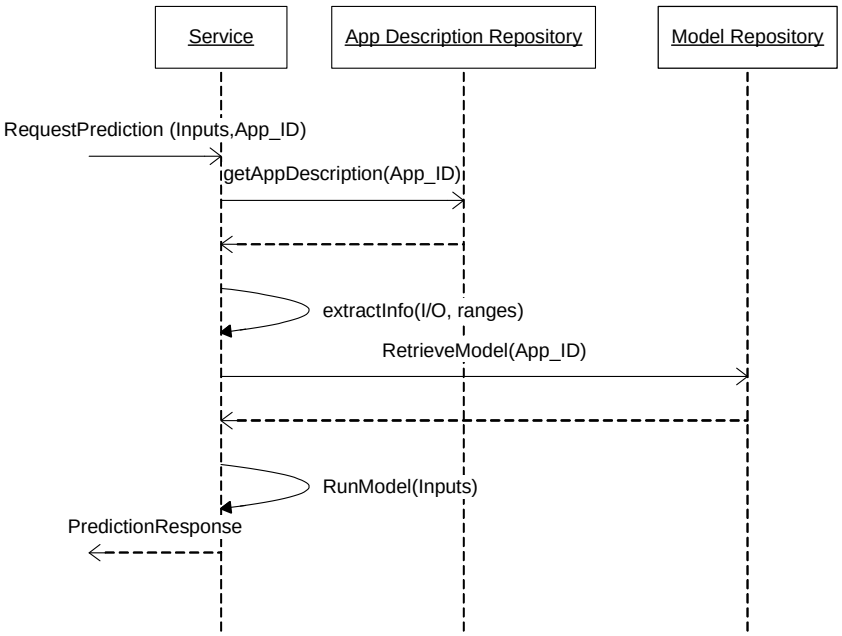
Σχήμα 22: Ακολουθιακό Διάγραμμα για τη λειτουργία CreateModel

Αυτή η φάση προβλέπεται να ακολουθηθεί κατά τη διάρκεια της προσαρμογής της εφαρμογής στην πλατφόρμα. Εάν αυτό μπορεί να περιληφθεί άμεσα στη διαπραγμάτευση χρόνου εκτέλεσης εξαρτάται κυρίως από την εφαρμοζόμενη μέθοδο και τη δυνατότητα δημιουργίας μοντέλων σε ένα λογικό χρονικό διάστημα. Σε πολλές περιπτώσεις αυτές οι μέθοδοι είναι χρονοβόρες, περιλαμβάνοντας δαπανηρούς αριθμητικούς υπολογισμούς. Αυτή η καθυστέρηση επιδεινώνεται από το γεγονός ότι το Octave είναι πιό αργό από τις παραδοσιακές γλώσσες προγραμματισμού, λόγω του ενδιάμεσου scripting layer. Εντούτοις δεδομένης της ποικιλομορφίας και της ευκολίας

Η σελίδα αυτή είναι σκόπιμα λευκή

υλοποίησης και του γεγονότος ότι τα μοντέλα δεν είναι απαραίτητο να δημιουργηθούν στην online διαπραγμάτευση αλλά σε μια προηγούμενη (σε κάθε περίπτωση αυτοματοποιημένη) φάση, αυτή η καθυστέρηση δεν είναι ένας σοβαρός παράγοντας.

Επιπλέον, ενώ θεωρείται γενικά ως διαδραστικό περιβάλλον, το Octave μπορεί επίσης να χρησιμοποιηθεί μέσω της γραμμής εντολών (command line interface-cli), με έναν αυτοματοποιημένο τρόπο, επιτρέποντας κατά συνέπεια να χρησιμοποιηθεί σε μια προσέγγιση SOA.



Σχήμα 23: Ακολουθιακό διάγραμμα για την λειτουργία RequestPrediction

3.3.2 Φάση Χρήσης Μοντέλου (Use Model-Request Prediction)

Για να χρησιμοποιηθεί ένα μοντέλο στη διάρκεια ζωής των υπηρεσιών απαιτούνται τα ακόλουθα βήματα:

- Αίτημα για μια εκτίμηση βασισμένη στους predictors του μοντέλου (όπως οι εισαγωγές/τα χαρακτηριστικά εφαρμογής)

Η σελίδα αυτή είναι σκόπιμα λευκή

- Απόκτηση των πληροφοριών σχετικά με την εφαρμογή (αυτό περιλαμβάνει τις εισόδους, τα αποτελέσματα, τις πιθανές τιμές των παραμέτρων κ.λπ., γενικά το ASCD)
- Το μοντέλο ανακτάται από το αρχείο και εκτελείται με το τρέχον σύνολο εισόδων-predictors
- Επιστροφή της απάντησης στο φορέα παροχής υπηρεσιών

Το ακολουθιακό διάγραμμα αυτής της φάσης εμφανίζεται στο Σχήμα 23. Οι διεπαφές πρέπει να εφαρμοστούν προς το εξωτερικό πλαίσιο (σε αυτήν την περίπτωση το πλαίσιο IRMOS που περιγράφεται στο [64]) για την αποδοχή των αιτημάτων εκτίμησης και προς την αποθήκη μοντέλων για την ανάκτηση κατά τη διάρκεια του χρόνου εκτέλεσης (στις διαπραγματεύσεις SLA παραδείγματος χάριν) των μοντέλων. Επίσης, μία διεπαφή προς την αποθήκη περιγραφής εφαρμογής (ASCD Repository) είναι απαραίτητη προκειμένου να ληφθούν οι πληροφορίες για τις εισόδους για λόγους κανονικοποίησης. Η κύρια εσωτερική λειτουργία, εκτός από την εξαγωγή XML που περιγράφεται προηγουμένως, είναι να εκτελεσθούν αυτά τα μοντέλα μέσω των κατάλληλων Octave scripts προκειμένου να αποκτηθεί η κατ' εκτίμηση απάντηση που επιστρέφεται στον πελάτη.

Αυτή η φάση είναι μη απαιτητική υπολογιστικά και χρησιμοποιείται κατά τη διάρκεια των διαπραγματεύσεων SLA εξαιτίας του γεγονότος ότι το μοντέλο δημιουργείται ήδη κατά τη διάρκεια της φάσης 1 και ο χρόνος να εκτελεστεί αυτό είναι συνήθως σύντομος.

3.4 Υλοποίηση Υπηρεσίας

Τα διαφορετικά στρώματα στην υπηρεσιοστρεφή αρχιτεκτονική του μηχανισμού εκτίμησης απεικονίζονται στο Σχήμα 24. Ο διαχωρισμός σε στρώματα γίνεται ώστε να

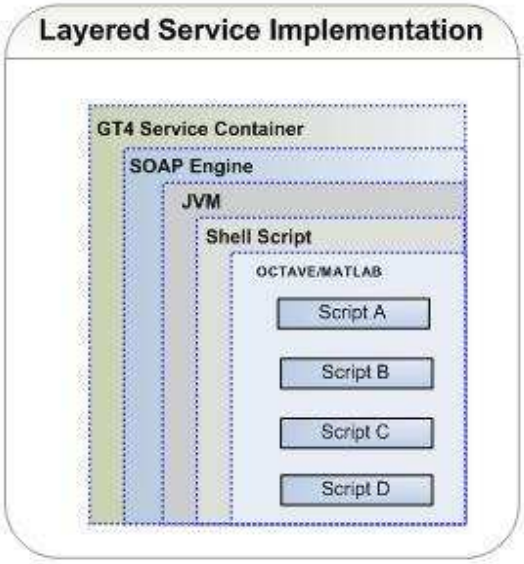
υπάρχει εκμετάλλευση έτοιμων λύσεων λογισμικού (όπως τα Octave και Matlab), οι οποίες μπορούν εύκολα να εφαρμόσουν τις διάφορες μεθόδους εκτίμησης. Προκειμένου όμως να γνωστοποιηθούν οι πληροφορίες και τα ορίσματα από τον πελάτη στο κατώτατο στρώμα θεωρήθηκε ασφαλέστερο να αποσυζευκτούν τα στρώματα ώστε οι αλληλεξαρτήσεις να περιοριστούν μόνο στις διεπαφές. Αυτό αυξάνει επίσης την ευελιξία της αρχιτεκτονικής καθώς καθιστά εύκολη την αλλαγή σε οποιοδήποτε στρώμα με μηδενικές επεμβάσεις στα υπόλοιπα. Παραδείγματος χάριν, για το λειτουργικό σύστημα, το κύριο συστατικό είναι το shell script. Για ολόκληρο το Octave layer, η βασική διεπαφή είναι μια γραμμή κώδικα στο shell script που ορίζει ποιο είναι το εκτελέσιμο αρχείο.

Η προκύπτουσα εφαρμογή είναι επίσης γενική και πλήρως αποσυνδεμένη από τον αλγόριθμο του μοντέλου, δεδομένου ότι οποιαδήποτε στιγμή ο υπεύθυνος για την ανάπτυξη μπορεί να μετατρέψει τις μεθόδους εκτίμησης με απλή αντικατάσταση των Octave scripts (Α, Β, Γ ή Δ). Στην παρούσα περίπτωση, με τη χρήση μιας τέτοιας αποσυνδεδεμένης λύσης, ήταν εφικτό να εφαρμοστεί εύκολα ο μηχανισμός με περισσότερες από μια μεθόδους όπως αναφέρεται στο [31]. Αυτές οι μέθοδοι μπορούν, οποιαδήποτε στιγμή, να αυξηθούν ή να συνδυαστούν προκειμένου να υπάρξει μια πιό εύρωστη προσέγγιση εκτίμησης. Οι λεπτομέρειες υλοποίησης για το κάθε στρώμα παρουσιάζονται στις επόμενες παραγράφους.

3.4.1 Στρώμα Μεσολογισμικού (Globus Toolkit 4)

Η υπηρεσιοστρεφής έκδοση του μηχανισμού που παρουσιάστηκε σε αυτό το κεφάλαιο υλοποιήθηκε αρχικά χρησιμοποιώντας την έκδοση Globus Toolkit 4 (GT4) και εκτελείται κάτω από το αυτόνομο container που το GT4 προσφέρει. Το GT4 [45] είναι

ένα ανοικτού κώδικα υπηρεσιοστρεφές μεσολογισμικό. Εφαρμόζει την WS-Security και άλλες προδιαγραφές σχετικά με την ασφάλεια, καθώς επίσης και το Web Services Resource Framework (WSRF), WS-Addressing και WS-BaseNotification.



Σχήμα 24: Διακριτά στρώματα υπηρεσιοστρεφούς αρχιτεκτονικής

Η διεπαφή WS (Web service) μπροστά από το μηχανισμό ενεργεί ως πύλη που προσφέρει single-point πρόσβαση στη λειτουργία της υπηρεσίας αποκρύπτοντας τις λεπτομέρειες και την πολυπλοκότητά της από τους πελάτες. Λεπτομερέστερα, ο πελάτης της υπηρεσίας στέλνει ένα αίτημα πρωτοκόλλου SOAP στη λειτουργία που εκτίθεται από τη διεπαφή WS. Το μήνυμα SOAP περιλαμβάνει τις εισόδους που απαιτούνται, δηλ. το ID του software component για το οποίο πρέπει να δημιουργηθεί ένα μοντέλο. Αυτό το μήνυμα συλλαμβάνεται από τη μηχανή SOAP που εκτελείται μέσα στο GT4 και μετατρέπει τις συμπεριλαμβανόμενες πληροφορίες σε αντικείμενα της Java πρίν περάσει στον πυρήνα της υπηρεσίας για την περαιτέρω επεξεργασία. Οι διαδικασίες που περιγράφονται στην Παράγραφο 3.3 εκτίθενται στον εξωτερικό κόσμο μέσω των τυποποιημένων περιγραφών WSDL.

Η σελίδα αυτή είναι σκόπιμα λευκή

Πρόσφατα, η τεχνολογία έχει μετατοπιστεί προς τα νέα πρωτόκολλα υπηρεσιών Ιστού (WS protocols) όπως το REST [52]. Το σημαντικότερο πλεονέκτημα του REST σε σχέση με το SOAP είναι η ευκολότερη εφαρμογή και οι υψηλότερες ταχύτητες. Προκειμένου να επικυρωθεί η στρωματοποιημένη προσέγγιση, αντικαταστάθηκε το GT4 στρώμα υπηρεσιών με μια RESTful έκδοση βασισμένη στο Jersey [53], μια υλοποίηση του REST. Η εφαρμογή Ιστού περιλήφθηκε σε έναν κεντρικό εξυπηρετητή Glassfish [54] και εγκαταστάθηκε στον ίδιο υπολογιστή με τη GT4 έκδοση. Η ίδια σειρά πειραμάτων πραγματοποιήθηκε, για τη σύγκριση των ταχυτήτων στις απαντήσεις των υπηρεσιών. Στο νέο στρώμα της WS, χρησιμοποιήθηκε η ίδια εφαρμογή πυρήνα της Java χωρίς οποιεσδήποτε τροποποιήσεις. Η μοναδική προσθήκη ήταν μία Java κλάση 60 γραμμών που περιείχε τους RESTful ορισμούς για τον προσδιορισμό των μεθόδων πρόσβασης στην υπηρεσία (POST μέθοδος για το RequestPrediction). Επιπλέον, ο πελάτης τροποποιήθηκε προκειμένου να κληθεί η REST διεπαφή.

Τα αποτελέσματα από τη σύγκριση των δύο εφαρμογών περιγράφονται λεπτομερώς στην Παράγραφο 3.6.

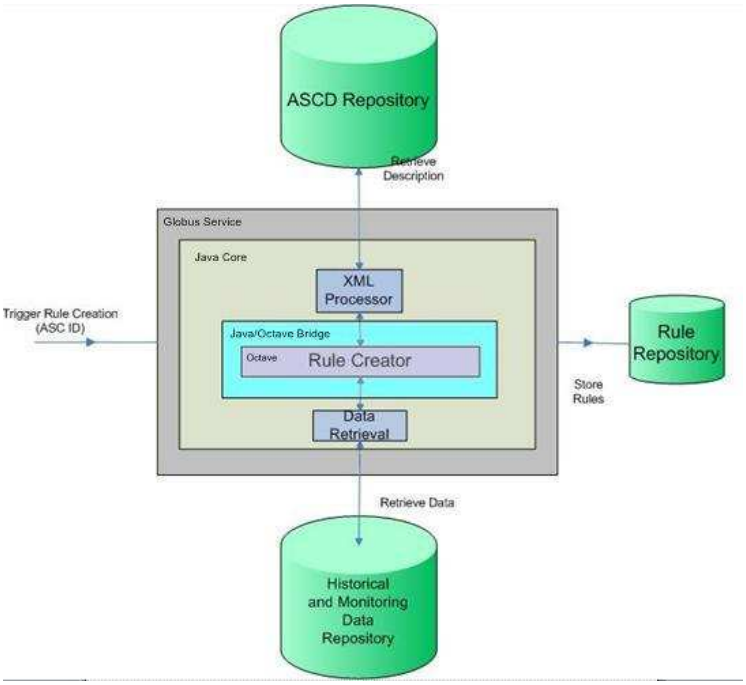
3.4.2 Στρώμα διασύνδεσης (Πυρήνας Java)

Ο πυρήνας Java που είναι στο κέντρο του πλαισίου είναι το κρισιμότερο μέρος. Είναι αρμόδιος για τη λήψη της ταυτότητας του τμήματος λογισμικού για το οποίο πρέπει να δημιουργηθεί το μοντέλο, την ανάκτηση της περιγραφής XML (ASCD) του και την επεξεργασία του. Αυτή η επεξεργασία εκτελείται βασιζόμενη στα ονόματα ετικετών μέσα στο ASCD, ανεξάρτητα από το επίπεδο βάθους όπου οι πληροφορίες αποθηκεύονται. Αυτό βοηθά τον υπεύθυνο για την ανάπτυξη στην αποσύζευξη της δομής του ASCD από το υπόλοιπο του πλαισίου, με μόνη εξάρτηση τα ονόματα ετικετών

(XML labels). Με βάση τις πληροφορίες που εξάγονται από το ASCD, ενεργοποιείται η διεπαφή SQL και ανακτώνται και κανονικοποιούνται τα δεδομένα από την βάση ιστορικών στοιχείων (DB MySQL). Τα δεδομένα αυτά αφορούν προηγούμενες εκτελέσεις και περιέχουν τιμές εισόδων της εφαρμογής (π.χ. αριθμό χρηστών), το υλικό στο οποίο εκτελέστηκε η εφαρμογή και τα αποτελέσματα QoS (π.χ. απόκριση εξυπηρετητή στις αιτήσεις χρηστών). Η κανονικοποίηση λαμβάνει υπόψη τις σειρές ή τις απαριθμήσεις των εισόδων και των παραμέτρων τους που εξήχθησαν προηγουμένως από την περιγραφή XML. Ο μετασχηματισμός του συνόλου δεδομένων σε ένα κοινό προκαθορισμένο διάστημα είναι απαραίτητος για εσωτερικούς λόγους επεξεργασίας των διάφορων μεθόδων εκτίμησης. Για τα συνεχή μεγέθη εισόδων, αυτό είναι αρκετά εύκολο, λαμβάνοντας μόνο την ελάχιστη και μέγιστη τιμή που δηλώνεται στο ASCD. Για τα διακριτά μεγέθη ήταν αρκετά δυσκολότερο, δεδομένου ότι ο κατάλογος πιθανών τιμών είναι δυναμικός για κάθε περίπτωση των εισαγωγών, οπότε οι κατάλληλοι μηχανισμοί ανάθεσης έπρεπε να εφαρμοστούν. Αυτοί οι μηχανισμοί επεκτείνονται και στις δύο φάσεις της υπηρεσίας (CreateModel και RequestPrediction) εξαιτίας του γεγονότος ότι οι είσοδοι που παραλαμβάνονται κατά τη διάρκεια του δεύτερου πρέπει πάλι να κανονικοποιηθούν προκειμένου να χρησιμοποιηθούν στα μοντέλα.

Στη συνέχεια η διεπαφή Java2Octave εκμεταλλεύεται μια κατάλληλη βιβλιοθήκη [61] για την αποθήκευση αυτών των στοιχείων ως αναγνώσιμα αρχεία δεδομένων Octave. Έπειτα η διεπαφή Java2System εκτελεί το shell script, που φορτώνει το περιβάλλον Octave το οποίο είναι αρμόδιο για να δημιουργήσει τα μοντέλα, χρησιμοποιώντας τα ιστορικά δεδομένα. Η χρήση μιας γενικής μεθόδου αποθήκευσης

των δεδομένων , π.χ. CSV (comma separated values), βοηθά στην περαιτέρω αποσύζευξη των στρωμάτων.



Σχήμα 25: Εσωτερική Δομή Υπηρεσίας Εκτίμησης

Μηχανισμοί κλειδώματος νημάτων έχουν εφαρμοστεί κατά τη διάρκεια αυτού του σταδίου έτσι ώστε τα επικαλύπτοντα αιτήματα που δημιουργούν τα πρότυπα να μην παρεμποδίζουν το ένα το άλλο. Η απόφυγή της διεπαφής Java2Octave για την εκτέλεση των Octave scripts έγινε διότι η μηχανή Octave που παρέχεται στη βιβλιοθήκη δεν υποστηρίζει τη χρήση πρόσθετων πακέτων που είναι πολύ χρήσιμα. Κατόπιν, μια διεπαφή Java2System εκτελείται προκειμένου να ξεκινήσει η εκτέλεση των Octave scripts ως εντολή συστήματος (shell script). Το κρισιμότερο μέρος σε αυτήν την διαδικασία είναι να συλληφθεί αποτελεσματικά το I/O που παράγεται από την εντολή συστήματος έτσι ώστε το σύστημα να αποκρίνεται κανονικά. Αυτός ο χειρισμός επιτυγχάνεται επίσης μέσω της κατάλληλης χρήσης των νημάτων για τη σύλληψη των output και error streams [70]. Ο χρόνος που το Octave μπορεί να εκτελεστεί προκειμένου

Η σελίδα αυτή είναι σκόπιμα λευκή

να παραχθεί το μοντέλο εξαρτάται από το timeout της WS κλήσης και είναι διαμορφώσιμος. Με την τρέχουσα υλοποίηση αυτό μπορεί να τεθεί σε οποιοσδήποτε επιθυμητό διάστημα.

Όταν η εκτέλεση τελειώνει, το παραχθέν μοντέλο μεταφέρεται στην αποθήκη κανόνων, έτσι ώστε μπορεί να ανακτηθεί στο μέλλον για την φάση χρήσης. Η εσωτερική δομή της υπηρεσίας εμφανίζεται στο Σχήμα 25.

3.4.3 Στρώμα Λειτουργικού Συστήματος

Για το στρώμα επιπέδου λειτουργικού συστήματος (OS), χρησιμοποιήθηκε το Fedora Core 9. Η κύρια λειτουργία αυτού του στρώματος, όπως αναφέρθηκε στην εισαγωγή αυτού του τμήματος, είναι να επιτραπεί η αποσύζευξη κάθε στρώματος από το προηγούμενο. Από άποψη λειτουργίας, το shell script χρησιμοποιείται κυρίως για την εκτέλεση του περιβάλλοντος Octave και για διάφορες βοηθητικές ενέργειες όπως ο χειρισμός καταλόγων και αρχείων.

3.4.4 Στρώμα Μαθηματικού Λογισμικού (GNU Octave)

Για την υλοποίηση του πυρήνα των μεθόδων εκτίμησης επιλέχτηκε το λογισμικό GNU Octave [46]. Το Octave είναι μια scripting γλώσσα υψηλού επιπέδου. Περιλαμβάνει διάφορα πρόσθετα πακέτα λειτουργιών εκτός από την κύρια έκδοση, οι οποίες βοηθούν σημαντικά στην εφαρμογή των προηγμένων αλγορίθμων αποτελεσματικότερα και με έναν πιό αφηρημένο τρόπο από τις τυποποιημένες γλώσσες προγραμματισμού όπως η C και η Java. Αυτό το γεγονός εξασφαλίζει την πολύ πιο γρήγορη υλοποίηση μεθόδων. Άλλες λύσεις όπως το Matlab ερευνήθηκαν επίσης, αλλά εγκαταλείφθηκαν εξαιτίας του γεγονότος ότι ενώ το τελευταίο υποστηρίζει σε πολλές περιπτώσεις τη μετατροπή των scripts σε executables, ένα βήμα που είναι απαραίτητο για την αυτοματοποιημένη

εκτέλεση των μεθόδων στο παρουσιασμένο προσανατολισμένο στις υπηρεσίες πλαίσιο, αυτό δεν συμβαίνει για όλες τις διαθέσιμες λειτουργίες, περιορίζοντας κατά συνέπεια τους αλγορίθμους υλοποίησης.

Χρησιμοποιήθηκαν δύο scripts, ένα για τη δημιουργία των μοντέλων (υπό μορφή μαθηματικών κανόνων ή αλγεβρικών λειτουργιών που συνδέουν τις εισόδους με την κατ' εκτίμηση έξοδο) και ένα για την εκτέλεση τους με τις εφαρμοζόμενες εισόδους κατά τη διάρκεια της φάσης εκτίμησης χρόνου εκτέλεσης.

Στην αρχική εφαρμογή αυτής της προσέγγισης [31], χρησιμοποιήθηκε μια απλή μέθοδος γραμμικής παλινδρόμησης πολλών μεταβλητών προκειμένου να επικυρωθούν τα πειράματα. Σε αυτήν την εργασία, αυτή η μέθοδος έχει αντικατασταθεί με έναν περιπλοκότερο αλγόριθμο. Αυτή η αντικατάσταση έχει υλοποιηθεί τροποποιώντας μόνο το Octave script στο κατώτατο σημείο των στρωμάτων εφαρμογής, επικυρώνοντας κατά συνέπεια την ευελιξία των προτεινόμενων μεθόδων εκτίμησης και του πλαισίου εφαρμογής τους.

Η ενισχυμένη μέθοδος εκτίμησης είναι βασισμένη στα τεχνητά νευρωνικά δίκτυα (ANNs), τα οποία είναι η βάση του μοντέλου για κάθε ASC. Ο λόγος για τη χρησιμοποίηση των ANNs είναι ότι είναι σε θέση να συλλάβουν και τις γραμμικές και μη γραμμικές εξαρτήσεις (όπως επίσης προσδιορίζεται στο [66]) μεταξύ της εισόδου και της εξόδου που επηρεάζουν την απόδοση του τμήματος λογισμικού. Περισσότερες λεπτομέρειες στη χρήση των ANNs ως μοντέλων πρόβλεψης απόδοσης μπορούν να βρεθούν στα [68][69]. Προκειμένου να βελτιστοποιηθεί η δημιουργία μοντέλων και να δημιουργηθεί ένα βελτιστοποιημένο ANN για κάθε ASC χρησιμοποιήθηκε ο αλγόριθμος ([65]) που έχει ήδη αναλυθεί στο Κεφάλαιο 2.

Γενικά, αυτή η διαδικασία στο παρελθόν βασίστηκε στην ανθρώπινη εμπειρία και τη λεπτομερή εξέταση. Στην παρούσα προσέγγιση, αυτό έχει αυτοματοποιηθεί μέσω της χρήσης του ΓΑ, έτσι ώστε μπορεί να χρησιμοποιηθεί σε ένα προσανατολισμένο προς τις υπηρεσίες περιβάλλον.

Η επίδραση της νέας μεθόδου μπορεί να συνοψιστεί στα ακόλουθα σημεία:

- Βελτιωμένη ακρίβεια των μοντέλων που δημιουργούνται
- Μεγαλύτερο υπολογιστικό φορτίο, ειδικά για τη λειτουργία createModel, δεδομένου ότι η νέα μέθοδος έχει πολύ περισσότερες υπολογιστικές απαιτήσεις από την γραμμική παλινδρόμηση πολλών μεταβλητών .

3.5 Επικύρωση υλοποίησης

Προκειμένου να επικυρωθεί η λειτουργία της προτεινόμενης υλοποίησης όσον αφορά την αποδοτικότητα, την απόδοση και την ευκολία χρήσης, χρησιμοποιήθηκε μια εφαρμογή για την πρόβλεψη του επιπέδου QoS της. Σαν predictors επιλέχτηκαν συγκεκριμένα χαρακτηριστικά εισόδου, που στο πλαίσιο του IRMOS περιλαμβάνονται στο ASCD, όπως περιγράφεται στη παράγραφο 2.7.2. Ο προσδιορισμός των εισόδων του μοντέλου βρίσκεται σε αυτή την περιγραφή XML, αλλά δεν απαιτεί την εκτενή γνώση του εσωτερικού κώδικα παρά μόνο εμπειρία που αποκτάται από τη χρησιμοποίηση της εφαρμογής. Ο υπεύθυνος για την ανάπτυξη δεν χρειάζεται να είναι ειδήμονας στην εφαρμογή ή στην πρόβλεψη απόδοσης για να τους προσδιορίσει.

3.5.1 Μελέτη εργασίας

Ως εφαρμογή χρησιμοποιήθηκε η μετατροπή ακατέργαστου βίντεο σε κωδικοποίηση MPEG4 με τον κωδικοποιητή FFMPEG [29]. Προκειμένου να προβλεφθεί ο χρόνος

εκτέλεσης του κωδικοποιητή (που θεωρείται ως επίπεδο QoS που προσφέρεται στο SLA), τέσσερις παράμετροι του βίντεο εισαγωγής θεωρήθηκαν ως predictors, που έχουν μια άμεση επίδραση στο χρόνο εκτέλεσης (Πείραμα 1 Κεφαλαίου 2.7.3).

Η επιλεγμένη μέθοδος πρόβλεψης περιγράφεται επίσης στο Κεφάλαιο 2. Στο Σχήμα 26 παρουσιάζεται η περιγραφή XML (ASCD) του τμήματος λογισμικού, η οποία προκύπτει από το UML διάγραμμα που δημιουργεί ο υπεύθυνος εφαρμογής και αναλύθηκε στο Κεφάλαιο 2.4.1. Από αυτήν την περιγραφή, η υπηρεσία χαρτογράφησης είναι σε θέση να επεξεργαστεί και να αντιληφθεί ότι το μοντέλο πρέπει να έχει τέσσερις εισόδους και μια έξοδο, μαζί με τα απαραίτητα διαστήματα για την κανονικοποίηση κάθε μίας.

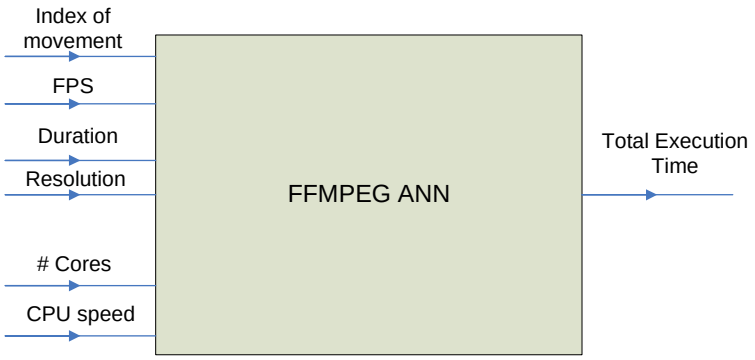
Επίσης χρησιμοποιούνται και 2 είσοδοι στο μοντέλο για την περιγραφή των χαρακτηριστικών του φυσικού κόμβου εκτέλεσης (hardware) και αποτελούνται από τον αριθμό των πυρήνων του επεξεργαστή και την ταχύτητά τους. Αυτές οι είσοδοι είναι ίδιες για όλα τα components. Το συνολικό μοντέλο απεικονίζεται στο Σχήμα 27.

3.5.2 Τεστ συστήματος

Προκειμένου να μετρηθεί η απόδοση του συστήματος δοκιμάστηκαν διάφορα σενάρια. Συνολικά, οι μετρήσεις έχουν εκτελεσθεί, και για τις δύο φάσεις που περιγράφονται στην Παράγραφο 3.3. Περισσότερη έμφαση έχει δοθεί στη μέθοδο requestPrediction, η οποία είναι κρισιμότερη από άποψη χρονικής απόδοσης. Αυτό οφείλεται στο γεγονός ότι χρησιμοποιείται κατά τη διάρκεια του σταδίου διαπραγμάτευσης των SLA, όπου λαμβάνουν μέρος και οι τελικοί χρήστες.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Model>
  - <id name="ComponentID" type="string">
      <valuesAllowed value="FFmpegEncoder" type="string" />
    </id>
  - <predictors name="VideoDuration" type="int">
      <MinValue value="0" type="int" />
      <MaxValue value="1000" type="int" />
    </predictors>
  - <predictors name="FramesPerSecond" type="int">
      <MinValue value="0" type="int" />
      <MaxValue value="40" type="int" />
    </predictors>
  - <predictors name="VideoResolution" type="enum">
      <valuesAllowed type="string" value="XGA" />
      <valuesAllowed type="string" value="EGA" />
      <valuesAllowed type="string" value="VGA" />
    </predictors>
  - <predictors name="IndexOfMovement" type="float">
      <MinValue value="0" type="float" />
      <MaxValue value="1" type="float" />
    </predictors>
  - <outputs name="ExecutionTime" type="float">
      <MinValue value="0" type="float" />
      <MaxValue value="1000000" type="float" />
    </outputs>
</Model>
```

Σχήμα 26: Περιγραφή XML (ASCD) του component της υπηρεσίας κωδικοποίησης



Σχήμα 27: ANN μοντέλο για το component κωδικοποίησης

Λεπτομερώς, εξετάστηκαν οι ακόλουθες περιπτώσεις:

- Αυτόνομη εκτέλεση του μοντέλου Octave, προκειμένου να ερευνηθεί ποιος είναι ο baseline χρόνος που απαιτείται για να υπολογίσει την εκτίμηση του μοντέλου.

Η σελίδα αυτή είναι σκόπιμα λευκή

Διαφορετικοί αριθμοί ταυτόχρονων εκτελέσεων έχουν ερευνηθεί ώστε να μελετηθεί η επίδραση πολλαπλών χρηστών στους χρόνους ολοκλήρωσης.

- Εκτέλεση μέσω υπηρεσιών του Octave model (requestPrediction method), προκειμένου να μετρηθεί η επιβάρυνση από το υπηρεσιοστρεφές πλαίσιο, για διαφορετικούς αριθμούς ταυτόχρονων κλήσεων (πελάτες).
- Εκτέλεση batch για πολλαπλές προβλέψεις. Αυτό βασίστηκε στο ότι ο μεγαλύτερος χρόνος για τη λειτουργία απάντησης αφιερώνεται για την αρχικοποίηση του περιβάλλοντος του Octave. Κατά συνέπεια, με αυτή τη μέτρηση μπορεί να παρατηρηθεί ο καθαρός χρόνος επεξεργασίας για μια πρόβλεψη μέσα στο περιβάλλον Octave. Αυτή η μέτρηση πραγματοποιήθηκε και για την υπηρεσία και για την αυτόνομη εκτέλεση.
- Χρήση πελατών δημιουργίας αιτήσεων από τον ίδιο ή διαφορετικούς φυσικούς κόμβους, για να παρατηρηθεί η επιβάρυνση των πελατών στην απόδοση υπηρεσιών. Αυτό είναι κρίσιμο για τις περιπτώσεις όπου οι υπηρεσίες συγχωνεύονται στον ίδιο φυσικό κόμβο. Η καλούσα υπηρεσία μπορεί να εδρεύει στο ίδιο υλικό.
- Αλλαγή του στρώματος της WS, για αντικατάσταση της SOAP τεχνολογίας με το πρωτόκολλο REST και παρατήρηση της διαφοράς στην απόδοση.
- Μετρήσεις των εσωτερικών χρόνων απόκρισης για τις βασισμένες στην υπηρεσία περιπτώσεις. Αυτό ήταν το χρονικό διάστημα μετά τη λήψη της κλήσης και μέχρι την ολοκλήρωση της επεξεργασίας. Με την αφαίρεση των εσωτερικών χρόνων από τους γενικούς χρόνους απόκρισης είναι δυνατό να παρατηρηθεί η καθυστέρηση που παρεμβάλλεται από τα διαφορετικά πρωτόκολλα υπηρεσιών (REST και SOAP). Η σύγκριση των αυτόνομων εκτελέσεων Octave (καθαρός χρόνος επεξεργασίας της

Octave) με τους εσωτερικούς χρόνους απόκρισης των εκδόσεων υπηρεσιών καθιστά δυνατή την παρατήρηση της καθυστέρησης που παρεμβάλλεται από την επεξεργασία που απαιτείται μέσα στην υπηρεσία και αφορά την επεξεργασία XML, την πρόσβαση στις αποθήκες ASCD κ.λπ., όπως αυτό απεικονίζεται στο Σχήμα 25.

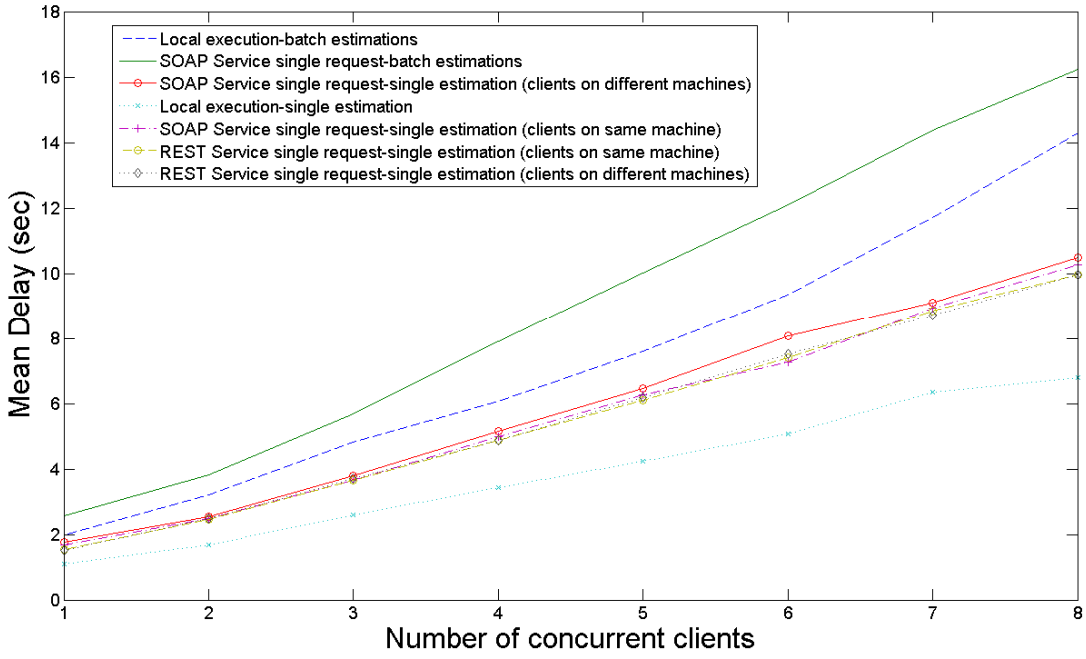
Για κάθε περίπτωση πραγματοποιήθηκαν περίπου 500 εκτελέσεις για κάθε πελάτη, έτσι ώστε να αποκτηθεί ένα ικανοποιητικό στατιστικό δείγμα. Επιπλέον, για την αυτόνομη εκτέλεση, προκειμένου να αποφευχθούν φαινόμενα caching που θα επηρέαζαν τις μετρήσεις, το ID των μοντέλων επιλέχτηκε τυχαία στις διαδοχικές εκτελέσεις.

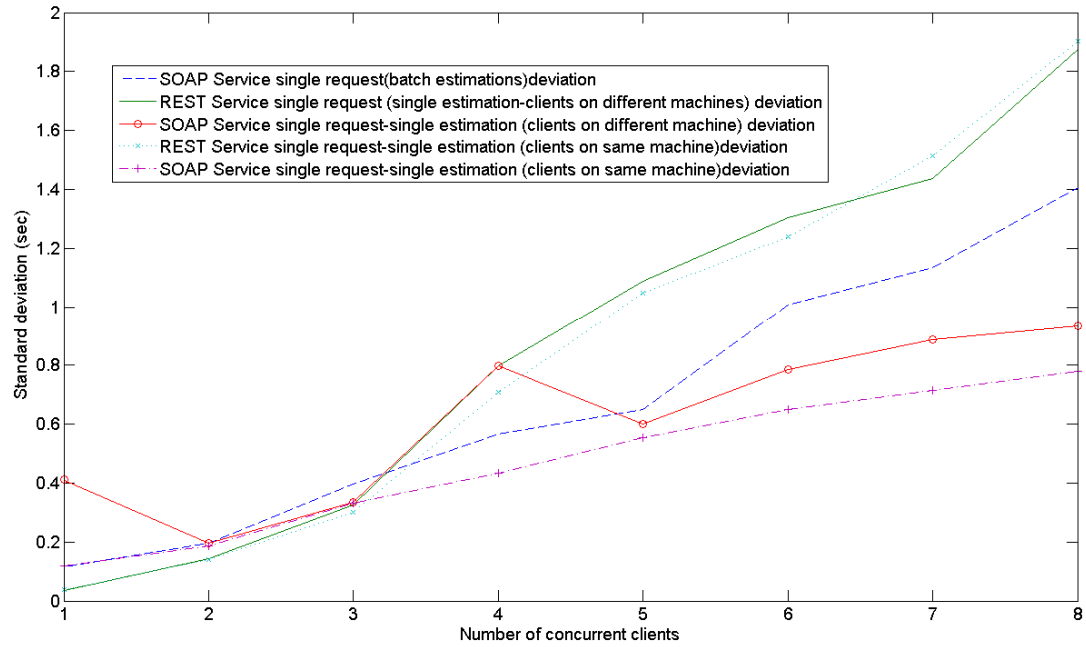
Προκειμένου να είναι βέβαιο ότι ο επιθυμητός αριθμός ταυτόχρονων κλήσεων θα εκτελούνταν, δημιουργήθηκαν αντίστοιχοι πελάτες. Κάθε πελάτης εκτελούσε μια κλήση requestPrediction για 500 φορές με έναν διαδοχικό τρόπο. Κατά συνέπεια εξασφαλίστηκε ότι για τα επιθυμητά τρεξίματα ο ίδιος αριθμός κλήσεων θα ήταν εκκρεμής. Κάθε πελάτης τεκμηρίωνε επίσης τους χρόνους απόκρισης των αιτημάτων. Τα αποτελέσματα συγχωνεύθηκαν και εξήχθησαν οι μέσοι χρόνοι.

3.6 Μετρήσεις

3.6.1 Μέθοδος RequestPrediction

Στο Σχήμα 28 απεικονίζονται τα αποτελέσματα από τα πειράματα που εκτίθενται λεπτομερώς στην προηγούμενη παράγραφο. Οι επιμέρους καθυστερήσεις συμπεριλαμβάνονται για όλα τα σενάρια που αναφέρονται στο προηγούμενο τμήμα, προκειμένου να παρατηρηθεί η επίδραση των διάφορων παραμέτρων στις χρονικές αποκρίσεις.





Σχήμα 28: Ολικοί χρόνοι απόκρισης και τυπική απόκλιση για διαφορετικές διαμορφώσεις και αριθμούς ταυτόχρονων κλήσεων

Η σελίδα αυτή είναι σκόπιμα λευκή

Από τη σύγκριση μεταξύ της εκτέλεσης υπηρεσιών και της τοπικής εκτέλεσης μπορεί να εξαχθεί το συμπέρασμα ότι η καθυστέρηση υπηρεσιών είναι της τάξης των 0.5 δευτερολέπτων (για 1 πελάτη). Αυτό περιλαμβάνει το χρόνο κλήσης υπηρεσιών και τους εσωτερικούς χρόνους επεξεργασίας για την υπηρεσία, αμέσως πριν από το ξεκίνημα του Octave. Αυτή η διαφορά αυξάνεται κατά τρόπο σχεδόν γραμμικό καθώς ο αριθμός ταυτόχρονων πελατών αυξάνεται.

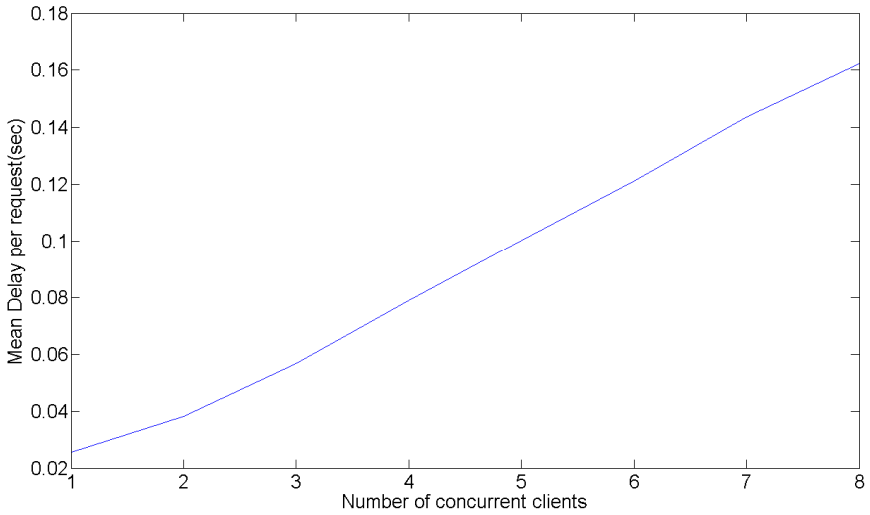
Από τη σύγκριση μεταξύ της κλήσης υπηρεσιών από τις ίδιες και διαφορετικές μηχανές μπορεί να εξαχθεί το συμπέρασμα ότι η επιβάρυνση εξαιτίας των πελατών που χρησιμοποιούν την ίδια μηχανή δεν είναι σημαντική. Αυτό μπορεί να είναι χρήσιμο σε περιπτώσεις όπου απαιτείται να συγκεντρωθούν όλες οι υπηρεσίες σε έναν φυσικό κόμβο για λόγους σύμπτυξης. Αυτή η σύγκριση προσπαθεί να μετρήσει δύο παράγοντες που μπορούν να επηρεάσουν την απόδοση. Υπέρ των πελατών που χρησιμοποιούν διαφορετικές μηχανές είναι το γεγονός ότι κανένας πόρος δεν δεσμεύεται για να εκτελεστούν οι πελάτες, κατά συνέπεια περισσότεροι πόροι αφήνονται για τον κεντρικό εξυπηρετητή. Υπέρ της ίδιας μηχανής είναι το γεγονός ότι δεν παρεμβάλλεται καθυστέρηση δικτύου λόγω της τοποθεσίας των αιτημάτων και των απαντήσεων. Είναι εμφανές ότι οι δύο παράγοντες είναι σχεδόν ισοδύναμοι, με μια εξαίρεση της SOAP-based υπηρεσίας, η οποία παρουσιάζει ελαφρώς αυξανόμενη καθυστέρηση. Εντούτοις αυτό το συμπέρασμα μπορεί να ισχύει μόνο για τη συγκεκριμένη υπηρεσία. Σε αυτήν την περίπτωση, οι εσωτερικοί χρόνοι επεξεργασίας για το Octave είναι σημαντικά υψηλότεροι από τις καθυστερήσεις υπηρεσιών, γεγονός που βοηθάει στη μείωση της επίδρασης των συγκεκριμένων παραμέτρων.

Από τη σύγκριση μεταξύ του τοπικού μεμονωμένου αιτήματος και του αιτήματος batch μπορεί να εξαχθεί το συμπέρασμα ότι η απόδοση επιδεινώνεται όταν εσωκλείει πολλαπλές εκτιμήσεις batch (100 σε αυτήν την περίπτωση). Αυτή η απόδοση επιδεινώνεται γρηγορότερα από την απόδοση του ενιαίου αιτήματος, όπως αναμένεται. Εντούτοις, αν σχεδιαστούν οι καθυστερήσεις ανά αίτημα (όπως γίνεται στο Σχήμα 29), μπορεί να παρατηρηθεί ότι ενώ η γενική εκτέλεση επιδεινώνεται, οι χρόνοι ανά εκτίμηση είναι της τάξης των χιλιοστών του δευτερολέπτου, η οποία είναι μια σημαντική βελτίωση σε σχέση με τη τάξη των δευτερολέπτων που παρατηρείται για τις μεμονωμένες αιτήσεις εκτίμησης. Τα ίδια συμπεράσματα μπορούν να εξαχθούν επίσης από τη σύγκριση μεταξύ των αντίστοιχων υπηρεσιοστραφών υλοποιήσεων. Η ερμηνεία για αυτό είναι ότι ο χρόνος εκκίνησης του Octave είναι ο σημαντικότερος λόγος για την εικονιζόμενη καθυστέρηση. Όταν έχουμε batch εκτιμήσεις, αυτός ο χρόνος εκκίνησης εμφανίζεται μόνο μιά φορά για όλες τις εκτιμήσεις.

Ο χρόνος εκκίνησης είναι της τάξης του 1 δεύτερου (για 1 πελάτη), όπως μπορεί να φανεί κατά προσέγγιση από τις διαφορές μεταξύ της μεμονωμένης και batch εκτέλεσης στο Σχήμα 28. Μετά από αυτό το διάστημα, για κάθε εκτίμηση ο χρόνος που απαιτείται από το Octave προκειμένου να εφαρμοστούν οι είσοδοι στο μοντέλο και να παραχθεί η έξοδος είναι της τάξης των 20 χιλιοστών του δευτερολέπτου για 1 πελάτη. Αυτή η περίοδος αυξάνεται γραμμικά σε σχέση με τον αριθμό ταυτόχρονων κλήσεων στον κεντρικό εξυπηρετητή.

Άλλα συμπεράσματα που μπορούν να εξαχθούν είναι η σαφής γραμμική σχέση μεταξύ των παρατηρηθεισών καθυστερήσεων και του αριθμού ταυτόχρονων πελατών που αποστέλουν τα αιτήματα. Επιπλέον, η εκτέλεση των πελατών σε διαφορετικές

μηχανές αυξάνει τη τυπική απόκλιση των καθυστερήσεων. Αυτή η αύξηση είναι λογική καθώς οι καθυστερήσεις επηρεάζονται επίσης από άλλες παραμέτρους, όπως οι παραλλαγές της πυκνότητας της κυκλοφορίας και των συνδέσεων δικτύου μεταξύ των πελατών και του κεντρικού εξυπηρετητή.

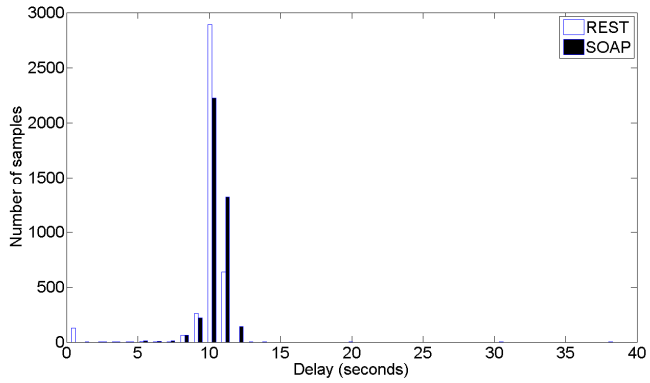


Σχήμα 29: Χρόνοι απόκρισης ανά αίτημα στην συγκεντρωτική αίτηση (batch)

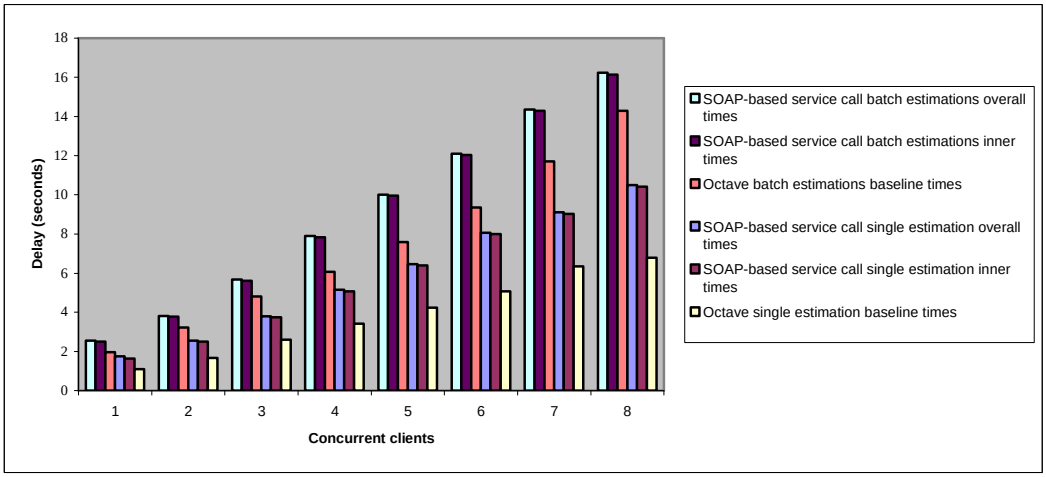
3.6.1.1 Σύγκριση μεταξύ REST και SOAP υλοποιήσεων

Ένα ενδιαφέρον συμπέρασμα είναι ότι συνολικά η REST εφαρμογή είναι ελαφρώς ταχύτερη από το SOAP, όπως απεικονίζεται στο Σχήμα 28. Οι κατανομές των μεμονωμένων χρόνων απόκρισης απεικονίζονται στο Σχήμα 30. Η REST εφαρμογή συγκεντρώνει σχεδόν όλες τις τιμές γύρω από τη τιμή των 10 δευτερολέπτων σε σύγκριση με την περίπτωση SOAP όπου αυτή η συγκέντρωση κατανέμεται στο εύρος 10-12 δευτερολέπτων. Πρέπει να τονιστεί ότι αυτοί είναι οι γενικοί χρόνοι, εντούτοις η εφαρμογή κάτω από το πρωτόκολλο επικοινωνίας είναι ίδια. Κατά συνέπεια η διαφορά μπορεί να οφείλεται μόνο στη διαφορά στα πρωτόκολλα υπηρεσιών ή κάποιο τυχαίο

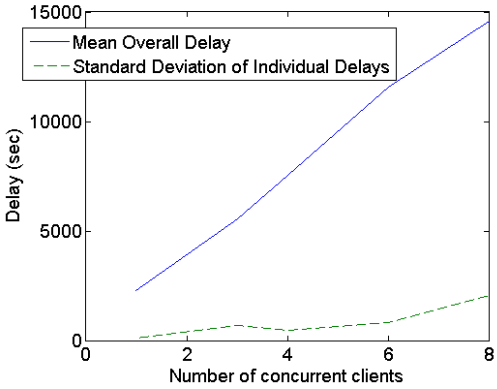
Η σελίδα αυτή είναι σκόπιμα λευκή



Σχήμα 30: Κατανομή χρόνων απόκρισης για τα 2 πρωτόκολλα επικοινωνίας (REST και SOAP) για 8 ταυτόχρονους πελάτες



Σχήμα 31: Σύγκριση των καθυστερήσεων στα 3 διαφορετικά βήματα της υπηρεσίας (Κλήση, Εσωτερική επεξεργασία, Καθαρή επεξεργασία Octave) για μεμονωμένες και batch κλήσεις.



Σχήμα 32: Καθυστερήσεις για την λειτουργία CreateModel

Η σελίδα αυτή είναι σκόπιμα λευκή

λόγο. Αλλά με 4000 δείγματα που συγκεντρώνονται για κάθε περίπτωση (500 κλήσεις ανά πελάτη), οι τυχαίες επιδράσεις ελαχιστοποιούνται. Όπως φαίνεται στο Σχήμα 28, οι τυπικές αποκλίσεις αυξάνονται σημαντικά για τις περιπτώσεις REST. Οπότε συνολικά, οι μέσοι χρόνοι είναι παρόμοιοι, το REST είναι ταχύτερο, εντούτοις η κατανομή των τιμών δείχνει ότι το πρωτόκολλο SOAP είναι ομαλότερο. Η ποικιλομορφία των χρόνων απόκρισης για τις τιμές που είναι κοντά στο μηδέν (~0.2 δευτερόλεπτα) για την εφαρμογή REST είναι ενδεικτική ότι η κλήση απέτυχε. Αυτό μπορεί να υποτεθεί με ασφάλεια δεδομένου ότι το περιβάλλον Octave χρειάζεται περίπου 1 δεύτερο για να αρχικοποιηθεί. Αυτή η αποτυχία εμφανίζεται για 128 περιπτώσεις των 4008 δειγμάτων που συλλέγονται, κατά συνέπεια είναι περίπου 3.2%. Αυτό δεν συμβαίνει για την εφαρμογή SOAP. Αυτό μπορεί επίσης να εξηγήσει τις αυξανόμενες τιμές τυπικής απόκλισης που εμφανίζονται στο REST.

Όπως έγινε αντιληπτό στις προηγούμενες παραγράφους, η διαφορά στην καθυστέρηση μεταξύ των REST και SOAP πρωτοκόλλων είναι οριακή. Εντούτοις, αυτή η οριακή επικράτηση του REST είναι στους γενικούς χρόνους απόκρισης. Στην υπηρεσία που εξετάστηκε, η σημαντικότερη καθυστέρηση είναι λόγω της Octave και της εσωτερικής επεξεργασίας της υπηρεσίας. Προκειμένου να απεικονιστεί σαφώς ποια είναι η διαφορά μεταξύ των καθαρών χρόνων κλήσης υπηρεσιών (και επομένως να συγκριθούν απευθείας τα δύο πρωτόκολλα), μια πιό λεπτομερής σύγκριση των χρόνων μεταξύ όλων των σταδίων της υπηρεσίας (γενική κλήση υπηρεσιών, χρόνος Octave και γενικός εσωτερικός χρόνος που περιλαμβάνει την Octave και την επεξεργασία xml, πρόσβαση FTP κ.λπ.) εμφανίζεται στο Σχήμα 31. Μετά από την αφαίρεση των εσωτερικών χρόνων από τους ολικούς χρόνους υπηρεσιών, χωρίς να λαμβάνονται υπόψη

οι περιπτώσεις αποτυχίας του REST, και συγκρίνοντας τις μέσες τιμές για 8 ταυτόχρονους πελάτες, ο μέσος χρόνος κλήσης REST προσδιορίστηκε σε 6.68 χιλιοστά του δευτερολέπτου ενώ ο χρόνος κλήσης SOAP σε 79.81 χιλιοστά του δευτερολέπτου. Κατά συνέπεια υπάρχει διαφορά μιας τάξης μεγέθους μεταξύ των δύο πρωτοκόλλων. Στην παρούσα περίπτωση, επειδή ο εσωτερικός χρόνος επεξεργασίας της υπηρεσίας είναι αρκετά μεγάλος, η διαφορά αυτή δεν έχει σημαντικές επιπτώσεις στο τελικό αποτέλεσμα. Σε περιπτώσεις όμως που ο χρόνος αυτός είναι αρκετά μικρότερος, ο χρόνος κλήσης του πρωτοκόλλου αναμένεται να διαδραματίζει πιο σημαντικό ρόλο.

Χρήσιμα ποσοτικά συμπεράσματα είναι δυνατόν να εξαχθούν από το Σχήμα 31. Περίπου 66% της γενικής καθυστέρησης οφείλεται στο χρόνο που απαιτείται από την Octave. Εντούτοις, μελετώντας το Σχήμα 29, μπορεί να παρατηρηθεί ότι δεν είναι ο χρόνος επεξεργασίας μέσα στο Octave που προκαλεί αυτήν την σημαντική καθυστέρηση, αλλά ο χρόνος εκκίνησης για το περιβάλλον. Μόλις αρχίσει το Octave, η επεξεργασία έχει διάρκεια της τάξης των χιλιοστών του δευτερολέπτου για κάθε υπολογισμό του μοντέλου. Σχεδόν 30% της καθυστέρησης οφείλεται στην εσωτερική επεξεργασία που απαιτείται για την απόκτηση του ASCD από την βάση και την εξαγωγή των αναγκαίων πληροφοριών.

3.6.2 Λειτουργία CreateModel

Για τη λειτουργία CreateModel, τα αποτελέσματα εμφανίζονται στο Σχήμα 32. Πάλι οι καθυστερήσεις φαίνονται να είναι γραμμικές έναντι του αριθμού ταυτόχρονων αιτημάτων που υποβάλλονται στον κεντρικό εξυπηρετητή. Μια προσδοκώμενη έκβαση είναι ότι αυτή η λειτουργία διαρκεί πολύ περισσότερο χρόνο σε σχέση με το requestPrediction, λόγω της φύσης του αλγορίθμου που περιγράφεται στο Κεφάλαιο 2.

Εντούτοις, δεδομένου ότι αυτός ο χρόνος είναι διαφορετικός για κάθε εκτέλεση, λόγω της τυχαιότητας των παραμέτρων που επιλέγονται μέσω του GA, έχει σχεδιαστεί επίσης η τυπική απόκλιση των εκτελέσεων. Για κάθε σημείο στη γραφική, 30 διαφορετικές εκτελέσεις πραγματοποιήθηκαν και η μέση τιμή από αυτές θεωρήθηκε ως καθυστέρηση. Η τυπική απόκλιση των μετρήσεων είναι μέσα στις αναμενόμενες τιμές.

3.7 Βελτίωση Σχεδίασης Υπηρεσίας Μέσω της Ανάλυσης Απόδοσης

Η ανάλυση απόδοσης της συμπεριφοράς της υπηρεσίας που παρουσιάστηκε στο προηγούμενο τμήμα είναι πολύ ευεργετική εάν χρησιμοποιηθεί ως ανατροφοδότηση στο σχεδιασμό της. Μέσω αυτής της ανάλυσης, τα αδύνατα και ισχυρά σημεία της εφαρμογής έχουν προσδιοριστεί και είναι δυνατόν να συνοψιστούν ως εξής:

- Η μεγάλη πλειοψηφία της καθυστέρησης κατά τη διάρκεια της λειτουργίας requestPrediction οφείλεται στην έναρξη του Octave. Ο καθαρός χρόνος επεξεργασίας για την εκτέλεση των μοντέλων είναι πολύ χαμηλός. Η συγκέντρωση κατά συνέπεια των αιτημάτων σε μια ενιαία κλήση προς το Octave και η εκτέλεση των εκτιμήσεων σε μορφή batch είναι πολύ ευεργετική (δεδομένου ότι η καθυστέρηση που οφείλεται στην εκκίνηση του Octave συμβαίνει μόνο μια φορά).
- Η λειτουργία createModel παρουσιάζει γραμμική αύξηση στις καθυστερήσεις που οδηγεί σε απαράδεκτους χρόνους. Κατά συνέπεια απαιτείται νε ευρεθεί μια λύση προκειμένου να μετριαστεί αυτή η επίδραση.

Στις ακόλουθες παραγράφους, περιγράφονται οι αλλαγές στο σχεδιασμό που υπαγορεύονται από τα προαναφερθέντα συμπεράσματα.

3.7.1 Συγκέντρωση φορτίου (Load Aggregation) σε στάδιο προεπεξεργασίας

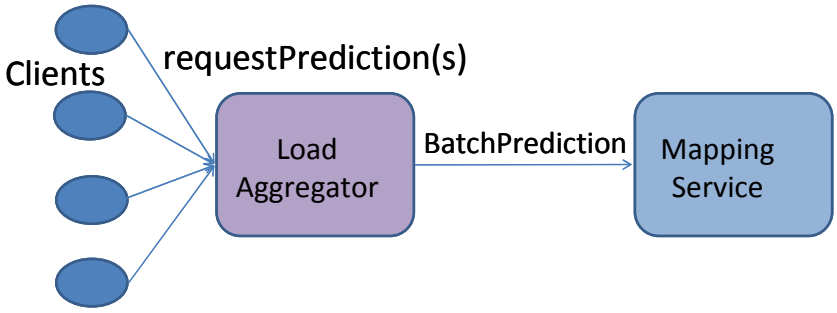
Η προσθήκη ενός τμήματος Load Aggregation (Σχήμα 33) ως front-end για την υπηρεσία μπορεί να εξυπηρετήσει το σκοπό της συγκέντρωσης των εισερχόμενων αιτημάτων από τους μεμονωμένους πελάτες και να μειώσει τον αριθμό εκκινήσεων του Octave.

Αυτά τα αιτήματα μπορούν να συγχωνευθούν σε μια ουρά αναμονής έως ότου ικανοποιείται ένα κριτήριο (μέγιστος αριθμός αιτημάτων ή χρονικού διαστήματος συγκέντρωσης) και έπειτα η διαδικασία μπορεί να συνεχιστεί (εκκίνηση Octave και εσωτερική επεξεργασία) ως μαζική επεξεργασία (batch processing). Με αυτόν τον τρόπο είναι δυνατόν να περιοριστούν οι καθυστερήσεις που οφείλονται στον χρόνο εκκίνησης, ο οποίος όπως διαπιστώθηκε από τις μετρήσεις αντιστοιχεί σε σχεδόν 66% της καθυστέρησης. Εντούτοις μια ανάλυση πρέπει να υλοποιηθεί προκειμένου να ερευνηθεί εάν αυτή η εφαρμογή ωφελεί πραγματικά τους χρόνους απόκρισης και ποιες είναι οι καταλληλότερες παράμετροι που πρέπει να επιλεχτούν (π.χ. χρόνος αναμονής πρίν προωθηθούν τα εισερχόμενα αιτήματα στο Octave).

Θεωρούμε ως *R(t)* τον ρυθμό άφιξης αιτημάτων, T_{PRE} την καθυστέρηση προεπεξεργασίας (συμπεριλαμβανομένων των χρόνων επεξεργασίας XML και προσκόμισης των ASCD), T_{OS} τον χρόνο εκκίνησης του Octave και T_{OP} το χρόνο επεξεργασίας μέσα στο περιβάλλον Octave. Όπως διαπιστώθηκε στις μετρήσεις, αυτοί οι χρόνοι συσχετίζονται άμεσα με τον αριθμό ***n*** των ταυτόχρονων αιτημάτων που υποβάλλονται προς τον κεντρικό εξυπηρετητή και μπορούν να υπολογιστούν κατά τη διάρκεια του χρόνου εκτέλεσης της υπηρεσίας. Η μέση τιμή της συνολικής καθυστέρησης T_{NL} για ένα αίτημα (χωρίς συγκέντρωση φορτίων) δίνεται από την Εξίσωση 5, εάν χρησιμοποιηθούν οι μέσες τιμές που εξάγονται από τις μετρήσεις.

$$T_{NL} = T_{PRE}(n) + T_{OS}(n) + T_{OP}(n)$$

Εξίσωση 5: Καθυστέρηση εξυπηρέτησης n ταυτόχρονων αιτημάτων χωρίς συγκέντρωση φορτίου



Σχήμα 33: Προσθήκη σταδίου προεπεξεργασίας κλήσεων για συγκέντρωση αιτημάτων

Μερικές ενδεικτικές τιμές για τις παραμέτρους της προηγούμενης σχέσης έχουν εξαχθεί από τις μετρήσεις και εμφανίζονται στην Εξίσωση 6.

$$T_{PRE} = 0.394 * n + 0.0086$$

$$T_{OCTAVE}(n) = T_{OS}(n) + T_{OP}(n) = 0.85 * n + 0.081$$

Εξίσωση 6: Εκτιμώμενες παράμετροι για μεμονωμένες κλήσεις προς την υπηρεσία χωρίς συγκέντρωση φορτίου

Για μια δεδομένη χρονική περίοδο, ο αριθμός *n* ταυτόχρονων κλήσεων μπορεί να βρεθεί εάν μετρηθεί ο ολικός χρόνος (από το μέσο όρο των μεμονωμένων χρόνων απόκρισης) και λύσουμε την Εξίσωση 6.

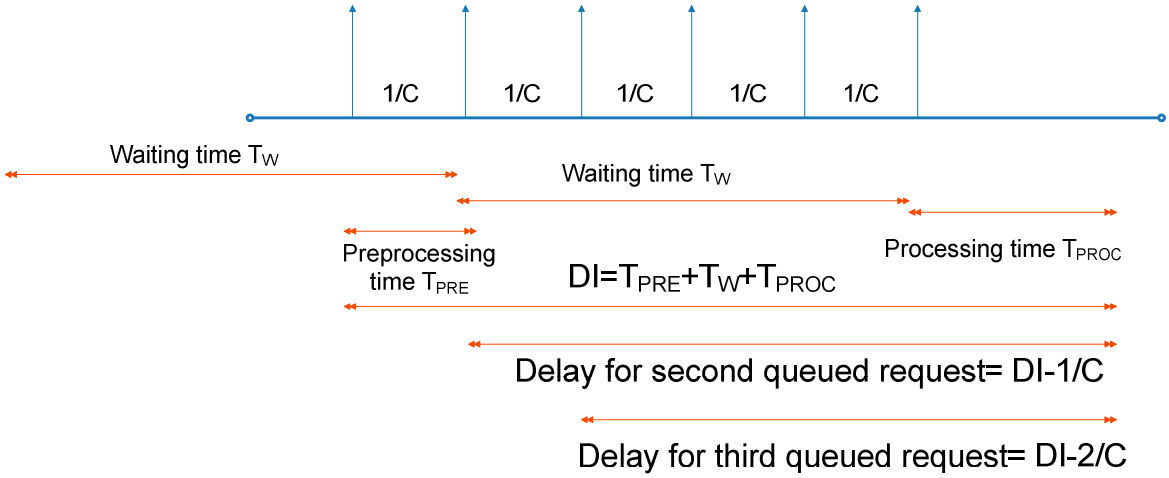
Τώρα ας θεωρηθεί μια αναμονή για χρονικό διάστημα T_W πρίν προωθηθούν τα αιτήματα προς το περιβάλλον Octave. Κατά τη διάρκεια αυτής της περιόδου, όλα τα αιτήματα που καταφθάνουν και περνάνε το στρώμα προεπεξεργασίας (κλήση υπηρεσιών και xml επεξεργασία) εισάγονται σε μια ουρά αναμονής. Στο τέλος του T_W το Octave εκκινεί και όλα τα αιτήματα στην ουρά αναμονής υποβάλλονται σε επεξεργασία.

Η σελίδα αυτή είναι σκόπιμα λευκή

Τα γενικά αιτήματα N_R που έχουν φθάσει κατά τη διάρκεια αυτού του διαστήματος (0 έως T_W) δίνονται από την Εξίσωση 7, εάν εξετάζουμε ένα σταθερό ρυθμό αφίξεων C:

$$N_R = \int_0^{T_W} R(t)dt = C * T_W$$

Εξίσωση 7: Αριθμός συγκεντρωμένων αιτημάτων στο διάστημα T_W



Σχήμα 34: Σχηματική παρουσίαση για την Εξίσωση 8 και την Εξίσωση 9

Αυτά δεν είναι ακριβώς τα αιτήματα που πρέπει να εξυπηρετηθούν επειδή δεν έχουν τελειώσει την προεπεξεργασία (στο μέρος επεξεργασίας XML) εγκαίρως προκειμένου να εισαχθούν στην ουρά αναμονής. Εντούτοις, δεδομένου ότι το ίδιο συμβαίνει για το προηγούμενο χρονικό διάστημα, τα υπόλοιπα αιτήματα από εκείνη την περίοδο εξυπηρετούνται κατά τη διάρκεια της τρέχουσας. Κατά συνέπεια τα γενικά αιτήματα δίνονται πράγματι από την Εξίσωση 7, εάν εξετάζουμε ένα σταθερό ρυθμό για το $R(t)=C$. Το διάστημα καθυστέρησης DI ώστε να εξυπηρετηθούν όλα τα αιτήματα δίνεται από την Εξίσωση 8, για τη χειρότερη περίπτωση που απεικονίζεται στο Σχήμα 34. Σε αυτήν την περίπτωση, η προεπεξεργασία XML ολοκληρώνεται αμέσως μετά από

Η σελίδα αυτή είναι σκόπιμα λευκή

το T_W του προηγούμενου διαστήματος. Κατά συνέπεια δεν υπάρχει καμία επικάλυψη μεταξύ των T_W και T_{PRE} και προστίθενται και τα δύο πλήρως στο *DI*, μαζί με το χρόνο επεξεργασίας που απαιτείται από το Octave. T_{OCTAVE} είναι ο χρόνος που απαιτείται από το Octave για να υπολογίσει την απάντηση, ο οποίος αναλύεται στο χρόνο που απαιτείται για να εκκινήσει το περιβάλλον (για 1 χρήστη) και το χρόνο που απαιτείται για μια εκτίμηση. Ο τελευταίος πολλαπλασιάζεται με τον αριθμό εκκρεμών αιτημάτων που συγκεντρώνονται κατά τη διάρκεια του T_W. T_{PRE} είναι ο χρόνος προεπεξεργασίας για κάθε αίτημα και δεν επηρεάζεται από τον αριθμό ταυτόχρονων κλήσεων εάν $T_{PRE} < 1/C$.

Σε αυτό το σημείο πρέπει να τονιστεί ότι χρησιμοποιήθηκε η υπόθεση του σταθερού ρυθμού αιτημάτων προκειμένου να απλοποιηθεί η ανάλυση. Στις πραγματικές καταστάσεις αναμένεται μια πιο πιθανοτική μορφή αφίξεων αιτημάτων.

$$DI = T_{PRE} + T_W + T_{OCTAVE} =$$
$$= T_{PRE}(1) + T_W + T_{OS}(1) + N_R * T_{OP}(1)$$

Εξίσωση 8: Συνολική καθυστέρηση εξυπηρέτησης αιτημάτων με συγκέντρωση φορτίου. Λόγω της συγκέντρωσης n=1

Για να προσδιοριστεί η ολική μέση καθυστέρηση ανά αίτημα T_L αθροίζονται οι επιμέρους καθυστερήσεις για κάθε αίτημα (Εξίσωση 9)

$$T_L = \frac{\sum_{k=0}^{N_R - 1} (DI - k * \frac{1}{C})}{N_R}$$

Εξίσωση 9: Μέση καθυστέρηση ανά αίτημα για την συγκέντρωση φορτίου

Η συνθήκη για ενεργοποίηση του Load Aggregation είναι όταν ο χρόνος χωρίς συγκέντρωση T_{NL} είναι μεγαλύτερος από τον χρόνο με συγκέντρωση T_L. Χρησιμοποιώντας τους τύπους που προέκυψαν από τις προηγούμενες παραγράφους και λύνοντας την ανισότητα $T_L < T_{NL}$ είναι δυνατόν να ευρεθούν τα όρια για τον χρόνο αναμονής T_W (Εξίσωση 10).

$$\frac{\sum_{k=0}^{N_R-1}(DI-\frac{k}{C})}{N_R} < T_{NL} \Rightarrow N_R * DI - \frac{N_R(N_R-1)}{2C} < N_R * T_{NL} \Rightarrow$$

$$T_{PRE} + T_W + T_{OS}(1) + N_R * T_{OP}(1) < T_{NL} + \frac{N_R - 1}{2C} \Rightarrow$$

$$T_W + C * T_{OP}(1) * T_W - \frac{C * T_W - 1}{2C} < T_{NL} - T_{PRE} - T_{OS}(1) \Rightarrow$$

$$T_W < \frac{2 * C * (T_{NL} - T_{PRE} - T_{OS}(1)) - 1}{C + 2 * C^2 * T_{OP}(1)}$$

Εξίσωση 10: Συνθήκη υπολογισμού του T_W βασει του ρυθμού αφίξεων C

Ένας απλός έλεγχος λογικής δείχνει ότι εάν το 1/C είναι πολύ υψηλό (δείχνοντας αραιά αιτήματα) τότε το C είναι πολύ χαμηλό, οπότε σ'αυτή την περίπτωση το T_W θα έχει μια αρνητική αξία, δείχνοντας ότι η συνάθροιση φορτίων δεν πρέπει να επιτραπεί. Αυτό αναμένεται καθώς όταν τα αιτήματα είναι αραιά, το n στην Εξίσωση 5 θα είναι πάντα 1.

Επιπλέον, είναι δυνατόν να δεσμευτεί το T_W να είναι μεγαλύτερο από 1/C (inter-arrival χρόνος αιτημάτων), επειδή για T_W<1/C η φάση συνάθροισης φορτίων εκφυλίζεται στη φάση χωρίς συγκέντρωση. Κάθε αίτημα εξυπηρετείται αυτόνομα και όχι μέσω της εκτίμησης batch. Έτσι χρειάζεται να ισχύει:

$$\frac{1}{C} < \frac{2 * C * (T_{NL} - T_{PRE} - T_{OS}(1)) - 1}{C + 2 * C^2 * T_{OP}(1)}$$

το οποίο μετά από επίλυση καταλήγει στην Εξίσωση 11.

$$\frac{1}{C} < T_{NL} - T_{PRE} - T_{OS}(1) - T_{OP}(1)$$

Εξίσωση 11: Όριο του C για την εφαρμογή της συγκέντρωσης φορτίου

Συμπερασματικά, εάν η Εξίσωση 11 ικανοποιείται κατά τη διάρκεια του χρόνου εκτέλεσης, είναι δυνατόν να επιτραπεί η συνάθροιση φορτίων, με έναν χρόνο αναμονής που προκύπτει από την Εξίσωση 10. Διαφορετικά η συνάθροιση φορτίων δεν είναι ωφέλιμη.

Σε αυτό το σημείο πρέπει να τονιστεί ότι η κύρια πρόθεση ήταν να χρησιμοποιηθεί ένας ικανοποιητικός αριθμός πελατών προκειμένου να προσδιοριστούν οι σχέσεις μεταξύ αυτής της παραμέτρου και της συμπεριφοράς της υπηρεσίας, προκειμένου να χρησιμοποιηθεί στην ανάλυση. Οι υψηλότεροι αριθμοί πελατών μπορούν να βελτιώσουν την ακρίβεια της κατ' εκτίμηση εξάρτησης, η οποία μπορεί να ενσωματωθεί άμεσα στην προαναφερθείσα ανάλυση (π.χ. εκθετική παρά γραμμική εξάρτηση μεταξύ των πελατών και των χρόνων απόκρισης). Αυτό προορίζεται να μελετηθεί στο κοντινό μέλλον.

3.7.2 Προώθηση της λειτουργίας createModel στο Υπολογιστικό Νέφος

Όπως μπορεί να παρατηρηθεί από τα πειράματα (Σχήμα 32), η αύξηση στους χρόνους απόκρισης createModel είναι σχεδόν γραμμική. Επειδή ο χρόνος επεξεργασίας για αυτήν την μέθοδο δημιουργίας μοντέλων είναι υψηλός (της τάξης των χιλιάδων δευτερολέπτων) και ότι πρέπει να τηρούνται κάποιοι ανεκτοί χρόνοι έναντι του αριθμού αιτημάτων, η προαναφερθείσα εφαρμογή δεν καλύπτει αυτήν την τελευταία απαίτηση.

Ιδανικά, προκειμένου να είναι σε θέση να ικανοποιήσει την ελαστική ζήτηση, η υπηρεσία πρέπει να μεταναστεύσει σε έναν πάροχο Υπολογιστικού Νέφους. Αυτό σημαίνει ότι μια νέα εικόνα της υπηρεσίας πρέπει να δημιουργηθεί στο Νέφος για κάθε νέο αίτημα για την δημιουργία μοντέλων.

Με αυτόν τον τρόπο, η απόδοση της λειτουργίας createModel θα είναι σταθερή και παρόμοια με την καθυστέρηση που απεικονίζεται για μια κλήση. Η διαφορά θα βρισκόταν στην πρόσθετη καθυστέρηση που παρεμβάλλεται προκειμένου να ενεργοποιηθεί η κλήση προς τον προμηθευτή Νέφους , το χρόνο που απαιτείται για να ενεργοποιηθεί η εικονική μηχανή που περιέχει την υπηρεσία και στο συμπληρωματικό κόστος. Αυτή η καθυστέρηση ήταν της τάξης των 40-60 δευτερολέπτων αλλά είναι ικανοποιητική αναμονή, αν θεωρηθεί ότι είναι εγγυημένοι οι χρόνοι για τη δημιουργία ενός μοντέλου (και ίσοι με περίπου 2500 δευτερόλεπτα χωρίς γραμμική αύξηση σε σχέση με τις ταυτόχρονες κλήσεις). Ο ψευδοκώδικας για αυτές τις ενέργειες εμφανίζεται στο Σχήμα 35.

Η έκδοση του εξυπηρετητή της υπηρεσίας που περιλαμβάνεται σε κάθε VM είναι η ίδια. Η VM μπορεί να φορτωθεί στον πάροχο και να δημιουργούνται όσα στιγμιότυπα χρειάζονται. Τα μοντέλα αποθηκεύονται άμεσα στην αποθήκη ASCD που υπάρχει εκτός του Νέφους. Ο λόγος για αυτό είναι ότι τα VM στο Νέφος θα εξαφανιστούν μετά από κάθε λειτουργία createModel που εξυπηρετούν. Ένα όριο βασισμένο στην αξία της έναρξης και της χρησιμοποίησης των νέων VM στο Νέφος μπορεί επίσης να υπάρξει, προκειμένου να επιτευχθεί μια ζυγισμένη συμπεριφορά μεταξύ της απόδοσης και του κόστους.

```
1. for each incoming createModel request
2.      call Cloud API (addVM)
3.      wait until VM is booted
4.      forward request to VM server
5.      get response (model stored)
6.      forward response to original caller (Boolean
   indication for model creation)
7.      terminate VM
8. end
```

Σχήμα 35: Ψευδοκώδικας για προώθηση αιτήσεων σε περιβάλλον Υπολογιστικού Νέφους

3.8 Συμπεράσματα

Σε αυτό το κεφάλαιο παρουσιάστηκε ένα προσανατολισμένο στις υπηρεσίες πλαίσιο για μεθόδους πρόβλεψης απόδοσης βασισμένες στο Octave. Αυτό το πλαίσιο μπορεί να εφαρμοστεί σε ένα πραγματικό περιβάλλον παραγωγής, προκειμένου να βοηθήσει στη διαδικασία διαπραγμάτευσης SLA μεταξύ των καταναλωτών και των φορέων παροχής υπηρεσιών σε SOIs. Σε αυτό το πλαίσιο η προτεινόμενη αρχιτεκτονική είναι γενική και ελαφρώς διασυνδεδεμένη, έτσι ώστε οποιαδήποτε στιγμή τα συγκεκριμένα στρώματα να μπορούν να αφαιρεθούν και να αντικατασταθούν από πιό προηγμένες τεχνολογίες. Οι μέθοδοι εκτίμησης απόδοσης έχουν μειωθεί σε scripts επιπέδου Octave, ένα γεγονός που ενισχύει την αποσύζευξη του υπηρεσιοστρεφούς πλαισίου από τις πραγματικές μεθόδους

Η σελίδα αυτή είναι σκόπιμα λευκή

εκτίμησης. Η ευελιξία της προσέγγισης έχει αποδειχθεί μέσω της χρήσης των διαφορετικών πρωτοκόλλων της WS και της ευκολίας αλλαγής μεθόδου εκτίμησης.

Από τις μετρήσεις προέκυψε ότι ο χρόνος εκκίνησης Octave είναι η σημαντικότερη καθυστέρηση της υπηρεσίας (~1 δεύτερο για 1 πελάτη ή 66% της ολικής καθυστέρησης). Οι διαφορές στα πρωτόκολλα WS μπορούν να βελτιώσουν τους καθαρούς χρόνους κλήσης υπηρεσιών κατά ένα μέγεθος, αλλά σε αυτή τη συγκεκριμένη εφαρμογή αυτό δεν μεταφράζεται σε ένα σημαντικό γενικό κέρδος απόδοσης (~4%). Αυτό οφείλεται στους αυξημένους εσωτερικούς χρόνους επεξεργασίας. Η REST υλοποίηση πάσχει επίσης από μια αποτυχία 1.5-3% των κλήσεων ανάλογα με τον αριθμό των ταυτόχρονων κλήσεων. Ο καθαρός χρόνος εκτίμησης μέσα στο Octave είναι περίπου 20 χιλιοστά του δευτερολέπτου (για 1 πελάτη). Η απαραίτητη XML επεξεργασία είναι περίπου 0.4 δευτερόλεπτα ή 30% της καθυστέρησης (για 1 πελάτη). Όλες οι καθυστερήσεις παρουσιάζουν γραμμική εξάρτηση από τον αριθμό των ταυτόχρονων αιτημάτων επεξεργασίας προς τον εξυπηρετητή.

Επιπλέον, τα όρια της συνάθροισης φόρτου εργασίας έχουν ερευνηθεί έτσι ώστε η χρονική καθυστέρηση εκκίνησης Octave να ελαχιστοποιηθεί, σε ένα δυναμικό σχέδιο λειτουργίας, που προορίζεται για ρύθμιση κατά τη διάρκεια του χρόνου εκτέλεσης. Για τις πιό χρονοβόρες διαδικασίες, όπως την δημιουργία μοντέλων, μια μετάθεση φορτίου προς ένα πάροχο Νέφους μπορεί να σταθεροποιήσει τους χρόνους απόκρισης.

Για το μέλλον, ένας στόχος είναι να επεκταθεί η χρήση αυτού του πλαισίου πρόβλεψης για προβλήματα βελτιστοποίησης. Ο κύριος στόχος είναι να αντικατασταθεί το πλαίσιο εκτίμησης βασισμένο στο Octave με ένα πλαίσιο βελτιστοποίησης βασισμένο στο εργαλείο GAMS [67], προκειμένου να χρησιμοποιηθεί κατά τη διάρκεια του ελέγχου

αποδοχής των υπηρεσιών στις υποδομές Νέφους. Τα προβλήματα βελτιστοποίησης παρουσιάζουν παρόμοια χαρακτηριστικά με τις περιπτώσεις εκτίμησης απόδοσης, ειδικά όσον αφορά την ενσωμάτωση του εξειδικευμένου λογισμικού και την μετάθεση σε μια υποδομή νέφους για την ικανοποίηση χρονικών περιορισμών. Τα συμπεράσματα και οι σχεδιαστικές αρχές που αναπτύχθηκαν σε αυτό το κεφάλαιο μπορούν να επεκταθούν σε οποιαδήποτε παρόμοια υπηρεσία που χρησιμοποιεί εξειδικευμένο ενσωματωμένο λογισμικό στο κατώτατο στρώμα.

4 *Αυτοδιαχειριζόμενοι Μηχανισμοί Πρόβλεψης 2 Επιπέδων*

Στο παρόν κεφάλαιο περιγράφεται το πρόβλημα της αυτοματοποίησης της διαχείρισης των πόρων μέσα από ένα πλαίσιο που δύναται να παρακολουθεί την κυμαινόμενη ζήτηση και να προσαρμόζει ανάλογα τις διαθέσιμες μηχανές. Σε αντίθεση με τα προηγούμενα κεφάλαια, όπου ο πελάτης εισάγει κατά τη δημιουργία του SLA όλους τους συγκεκριμένους όρους υψηλού επιπέδου (παραμέτρους φόρτου και παραμέτρους ποιότητας υπηρεσίας), στο παρόν κεφάλαιο η υποδομή είναι υπεύθυνη για την ανίχνευση των παραμέτρων φόρτου και ο ορισμός στο SLA αφορά μόνο τα επίπεδα υπηρεσίας

4.1 Ορισμός του Προβλήματος

Όπως είδαμε στην εισαγωγή, κατά τη δημιουργία ενός SLA ο πελάτης επιζητά την δημιουργία μίας υπηρεσίας με συγκεκριμένους όρους υψηλού επιπέδου (παραμέτρους φόρτου και παραμέτρους ποιότητας υπηρεσίας). Αυτό το μοντέλο

λειτουργεί στις περιπτώσεις που ο πελάτης έχει επίγνωση του φόρτου. Για παράδειγμα ο διευθυντής ενός σχολείου που επισκέπτεται ένα μουσείο, στο οποίο η εφαρμογή e-learning που παρουσιάστηκε στα προηγούμενα κεφάλαια είναι διαθέσιμη, γνωρίζει το πλήθος των μαθητών που θα παρακολουθήσουν την εφαρμογή. Τί γίνεται όμως στις περιπτώσεις που ο πελάτης δεν έχει επίγνωση της αναμενόμενης κίνησης; Για παράδειγμα, ένας κάτοχος ενός ιστοτόπου που απευθύνεται στο γενικό κοινό δεν μπορεί να γνωρίζει εκ των προτέρων πόση απήχηση θα έχει και το μέγεθος της ζήτησης που θα χρειαστεί να εξυπηρετήσει.

Άρα, μια σημαντική πρόκληση για την εκμετάλλευση του Υπολογιστικού Νέφους είναι η αυτόνομη διαχείριση των δεσμεύσεων ποιότητας υπηρεσίας για τους πελάτες καθ' όλη τη διάρκεια του κύκλου της ζωής μιας υπηρεσίας. Η πολυπλοκότητα αυτού του προβλήματος αυξάνεται εξαιτίας της κυμαινόμενης συμπεριφοράς των χρηστών κατά τη διάρκεια του κύκλου ζωής. Τα τωρινά εργαλεία και οι υπηρεσίες που παρέχονται στο στρώμα PaaS λύνουν αρκετά θέματα (π.χ. τη διαμόρφωση εφαρμογής και εκτέλεσης, τη θέσπιση και παρακολούθηση της εφαρμογής, την αξιολόγηση των γεγονότων και των διορθωτικών ενεργειών, κ.λπ.). Παρόλαυτα, η διαχείριση του κύκλου της ζωής υπηρεσιών που βασίζεται στην ανάλυση της συμπεριφοράς των χρηστών απαιτεί μηχανισμούς που αφαιρούν την προαναφερθείσα πολυπλοκότητα και παρέχουν κατά παραγγελία τα μοτίβα χρήσης για τις υψηλού επιπέδου παραμέτρους φόρτου εργασίας εφαρμογής. Έτσι μπορεί να επιτευχθεί η βέλτιστη και αυτοματοποιημένη διαχείριση των επιπέδων QoS μέσω της πρόβλεψης πιθανών παραβιάσεων SLA και της εφαρμογής κατάλληλων ενεργειών (π.χ.. παροχή πόρων για εξυπηρέτηση των κατ' εκτίμηση απαιτήσεων).

Σε αυτό το κεφάλαιο η προσέγγιση περιλαμβάνει έναν μηχανισμό ανάλυσης χρονοσειράς για τις υψηλού επιπέδου παραμέτρους όσον αφορά το φόρτο εργασίας της εφαρμογής (π.χ. αριθμός χρηστών, συχνότητα των αιτημάτων, γεωγραφική κατανομή, κ.λπ.), και ένα μηχανισμό αντιστοίχισης (παρόμοιο με αυτούς που παρουσιάστηκαν για τις προηγούμενες εφαρμογές) για να μετατρέψει αυτές τις υψηλού επιπέδου παραμέτρους στις ιδιότητες των πόρων, οι οποίες ζητούνται από τους προμηθευτές IaaS. Και οι δύο μηχανισμοί είναι βασισμένοι στα τεχνητά νευρωνικά δίκτυα (ANNs), τα οποία προσαρμόζονται σε κάθε κομμάτι λογισμικού και με τους οριζόμενους από την εφαρμογή όρους.

Τα σημαντικότερα πλεονεκτήματα της ενισχυμένης προσέγγισης είναι τα ακόλουθα: (ι) επιτρέπει στα περιβάλλοντα να αντιμετωπίσουν και εφαρμογές πραγματικού χρόνου μέσω της παροχής ακριβών εκτιμήσεων κατά τη διάρκεια του χρόνου εκτέλεσης όσον αφορά τον προσδοκώμενο φόρτο εργασίας - που οδηγεί στην απαραίτητη παροχή των πόρων ελάχιστα πριν τη ζήτηση, (ii) την ελαχιστοποίηση δαπανών και την παροχή εγγυήσεων QoS μέσω της δυναμικής ανίχνευσης παραβίασης SLA, (iii) αυξανόμενη επιχειρησιακή ευκινησία μέχρι το μειωμένο χρόνο στον οποίο η πρόσθετη ικανότητα μπορεί να προσφερθεί, (iv) βελτιστοποιημένη χρησιμοποίηση υποδομής μέσω της εκμετάλλευσης των προσδοκώμενων πληροφοριών φόρτου εργασίας, (v) δυνατότητα εφαρμογής στα διαφορετικά τμήματα λογισμικού λόγω της γενικής και εύκαμπτης φύσης των ANNs.

4.2 Σχετικές Εργασίες

Μία μεγάλη ποικιλία εργασιών έχουν δημοσιευτεί σε μια προσπάθεια να αντιμετωπιστούν τα προαναφερθέντα θέματα. Μερικές από τις πλέον χαρακτηριστικές αναφέρονται σε αυτό το κεφάλαιο. Στο [98], το πλαίσιο LoM2HiS παρουσιάζεται. Σε αυτήν την περίπτωση οι συγγραφείς παρέχουν ένα πλαίσιο για την αντίστροφη διαδικασία από αυτή που προτείνουμε, δηλαδή τη μετάφραση όρων χαμηλού επιπέδου στους όρους εφαρμογής υψηλού επιπέδου που χρησιμοποιούνται σε συμφωνίες επιπέδων υπηρεσιών υπολογιστικών νεφών. Επιπλέον, εστιάζουν σε ποιο γενικά χαρακτηριστικά και όρους όπως η διαθεσιμότητα.

Μια ενδιαφέρουσα εργασία, που συγκρίνει τα Μπεϋζιανά δίκτυα (BNs) με ANNs παρουσιάζεται στο [99]. Σύμφωνα με τα συμπεράσματα, τα ANNs έχουν καλύτερους χρόνους απόκρισης μοντέλων αλλά χειρότερους χρόνους δημιουργίας τους. Στην εργασία μας θεωρείται ότι οι χρόνοι απόκρισης είναι σημαντικότεροι από τους χρόνους δημιουργίας. Το μοντέλο κατασκευάζεται μιά φορά και μπορεί να αναζωογονηθεί σε τακτά χρονικά διαστήματα αλλά χωρίς στενούς χρονικούς περιορισμούς. Επιπλέον, η προσέγγισή μας στοχεύει στον προσδιορισμό της εξάρτησης της απόδοσης κάθε υπηρεσίας από τις εισόδους της, οδηγώντας κατά συνέπεια σε συγκεκριμένη εκτίμηση ανά SLA.

Μια προσέγγιση διπλού SLA μεταξύ των διαφορετικών οντοτήτων σε ένα περιβάλλον υπολογιστικών νεφών αναφέρεται στο [100]. Το SLA διαιρείται σε SLA εφαρμογής (μεταξύ του καταναλωτή και του PaaS) και τεχνικό SLAs (μεταξύ του PaaS και του IaaS). Ενδιαφέροντα πρότυπα για την επίτευξη της διαλειτουργικότητας μεταξύ των διαφορετικών προσεγγίσεων σχετικά με την χρήση μοντέλων αναφέρονται στο

[101]. Επιπλέον, αυτό το πλαίσιο μπορεί να χρησιμοποιηθεί ως σχήμα για τον καθορισμό των εισόδων και των αποτελεσμάτων των μοντέλων.

Μια ενδιαφέρουσα προσέγγιση παρουσιάζεται επίσης στο [102]. Οι συγγραφείς εισάγουν έναν μηχανισμό για το δυναμικό επανασχηματισμό και τη διάθεση των πόρων στις εικονικές μηχανές σύμφωνα με τις απαιτήσεις εφαρμογής. Η προσέγγιση είναι βασισμένη σε έναν μηχανισμό ενισχυμένης μάθησης (συγκεκριμένα VCONF) για την εξελιξιμότητα και την προσαρμοστικότητα, οι οποίες επιτρέπουν την αυτόματη διαμόρφωση των VM στο στρώμα IaaS. Στο ίδιο θέμα, οι συγγραφείς στο [103] συνδυάζουν την ενισχυμένη μάθηση (για την offline κατάρτιση) με μοντέλα αναμονής ουράς (για τη σε απευθείας σύνδεση και διαχείριση του περιβάλλοντος). Εκτός αυτού, η παρουσιαζόμενη προσέγγιση αναφέρει ότι η ενισχυμένη εκμάθηση είναι βέλτιστη για την υψηλή απόδοση στην εκπαίδευση.

Μια άλλη ενδιαφέρουσα προσέγγιση συζητείται στο [104] για την πρόβλεψη φόρτου εργασίας και βασίζεται σε Kriging surrogate models για την παροχή ενός αυτόνομου ελεγκτή. Τα πολυδιάστατα αναπληρωματικά μοντέλα χρησιμοποιούνται για να προσεγγίσουν το σχεδιάγραμμα απόδοσης των εφαρμογών μέσω μιας λειτουργίας χρησιμότητας, που χρησιμοποιείται κατόπιν για να υποστηρίξει την απόφαση των ελεγκτών. Ενώ είναι λεπτομερής στο μέρος της μετάφρασης/εκτίμησης, αυτή η προσέγγιση αναφέρει την ιδέα της ενσωματωμένης μελλοντικής πρόβλεψης φόρτου εργασίας αλλά δεν την υλοποιεί συνδυασμένα σε επίπεδο μηχανισμού.

Στο τομέα της αποδόμησης εργασιών και δρομολόγησης, το [105] χρησιμοποιεί τεχνικές εκμάθησης μηχανών για να παράγει τα μοντέλα ΤΝΔ πρόβλεψης απόδοσης που βασίζονται σε ιστορικά στοιχεία. Μια μηχανή που μαθαίνει μέσω των ταξινομητών

παλινδρόμησης περιγράφεται στο [106], σε μια προσέγγιση που επιτρέπει την πρόβλεψη των παραβιάσεων SLA κατά το χρόνο εκτέλεσης και βασίζεται στις μετρούμενες πληροφορίες και τα γεγονότα που προέρχονται από τα στοιχεία QoS της διαδικασίας σύνθετων υπηρεσιών. Αυτή η προσέγγιση χρησιμοποιεί επίσης τις γλώσσες ροής εργασιών (BPEL) και σημεία ελέγχου μαζί με μοντέλα παλινδρόμησης προκειμένου να προσδιοριστεί σε κανονική λειτουργία η ολοκλήρωση μιας συγκεκριμένης υπηρεσίας.

Μια υποδομή που αυξομειώνεται σύμφωνα με το φόρτο εργασίας παρουσιάζεται στο [107]. Η πρόβλεψη φόρτου εργασίας εκτελείται μέσω ενός αυτοανάδρομου μοντέλου, ενώ επιτυγχάνονται βελτιώσεις μέσω ομαδοποίησης των ιστορικών πληροφοριών. Οι συγγραφείς στο [108] παρουσιάζουν ένα μοντέλο πρόβλεψης (συγκεκριμένα μοντέλο παλινδρόμησης ανίχνευσης μοτίβων) της χρονικής σειράς που μπορεί να χρησιμοποιηθεί κατά τη διάρκεια του χρόνου εκτέλεσης χρησιμοποιώντας φιλτραρισμένα δεδομένα των πόρων.

Αυτό που προκύπτει από την ανάλυση σχετικών εργασιών είναι ότι ενώ υπάρχουν πολυάριθμες εξαιρετικά σημαντικές προσεγγίσεις, είναι συνήθως περιορισμένες σε ένα από τα δύο στρώματα (μετάφραση και πρόβλεψη συμπεριφοράς) που εξετάζονται σε αυτό το κεφάλαιο. Ένα άλλο ενδιαφέρον συμπέρασμα είναι ότι τα ANNs θεωρούνται από όλο και περισσότερους ερευνητές ως ιδανικά για αυτούς τους τύπους προβλημάτων.

4.3 Δομή Μηχανισμού Πρόβλεψης Δύο (2) Επιπέδων

Ο σημαντικότερος στόχος των Υπολογιστικών Νεφών είναι να προσφερθούν οι υποδομές που είναι εύκολα ή αυτόματα διαχεριζόμενες. Προκειμένου να επιτευχθεί αυτό, διάφορες προσεγγίσεις έχουν ακολουθηθεί, που αναλύθηκαν στο προηγούμενο υποκεφάλαιο. Σε αυτό το έγγραφο ακολουθείται μια προσέγγιση δύο επιπέδων:

• Στρώμα Αντιστοίχισης-Μετάφρασης (Translation Layer): ένα μοντέλο δημιουργείται για την εφαρμογή που έχει το στόχο να μετατρέψει τους όρους A-SLA (χαρακτηριστικά φόρτου εργασίας και KPIs της εφαρμογής) σε όρους T-SLA (διαμόρφωση υλικού στο IaaS). Η λογική αυτή αναφέρθηκε με λεπτομέρεια στο Κεφάλαιο 2

• Στρώμα Αναγνώρισης Συμπεριφοράς (Behavioural Layer): ένα μοντέλο δημιουργείται προκειμένου να απεικονιστούν οι τρόποι μεταβολής της χρήσης της υπηρεσίας και να είναι σε θέση να προβλέψει τις μελλοντικές παραμέτρους φόρτου εργασίας της τελευταίας. Με την προσθήκη αυτού του στρώματος, ο πελάτης πρέπει τώρα να διευκρινίσει μόνο το αναγκαίο KPI και όχι τις παραμέτρους φόρτου εργασίας.

Και τα δύο στρώματα είναι βασισμένα στα τεχνητά νευρωνικά δίκτυα (ANNs), αλλά διαφορετικού τύπου και αρχιτεκτονικής. Ακολουθεί η ανάλυση του Στρώματος Αναγνώρισης Συμπεριφοράς και ο συνδυασμός του με το Στρώμα Μετάφρασης.

4.3.1 Στρώμα Αναγνώρισης Συμπεριφοράς (Behavioural Layer)

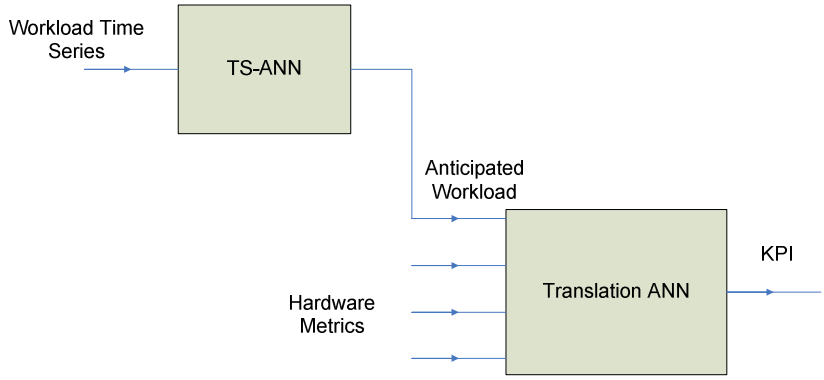
Οποιαδήποτε δυναμική προσέγγιση πρέπει να είναι σε θέση να ενεργήσει προτού εμφανιστεί μια κρίσιμη αλλαγή και επηρεάσει τους χρόνους απόκρισης του συστήματος προκειμένου να επιτευχθεί η βελτιστοποιημένη διαχείριση,. Αυτή η αλλαγή μπορεί να είναι μια αιχμή στο φόρτο εργασίας. Όπως φάνηκε στις σχετικές εργασίες, οι προσεγγίσεις που βασίζονται στον έλεγχο πρέπει πρώτα να ανιχνεύσουν την επιδείνωση στην απόδοση συστημάτων και έπειτα να ενεργήσουν για να βελτιώσουν τους πόρους και έτσι το χρόνο απόκρισης της υπηρεσίας. Εντούτοις μια τέτοια δράση μπορεί να οδηγήσει σε ταλαντευόμενη συμπεριφορά, ειδικά σε περιπτώσεις γρήγορης διακύμανσης του φόρτου μιας υπηρεσίας. Εάν παραδείγματος χάριν η συχνότητα των αλλαγών είναι

υψηλότερη από την ανταπόκριση του συστήματος (που μεταφράζεται κυρίως στο χρόνο για την αύξηση των πόρων), αυτό θα οδηγήσει στις περιπτώσεις όπου το τελευταίο είναι πάντα ένα βήμα πίσω.

Για να αντιμετωπιστεί αυτό το πρόβλημα, προστέθηκε το Στρώμα Αναγνώρισης Συμπεριφοράς, το οποίο αποτελείται από έναν μηχανισμό πρόβλεψης συμπεριφοράς/φόρτου εργασίας (ANN αναγνώρισης χρονοσειράς). Αυτός παίρνει ως είσοδο τη χρονοσειρά των ποικίλων χαρακτηριστικών γνωρισμάτων ή των εισόδων φόρτου εργασίας, όπως ο αριθμός χρηστών. Μέσω της ακριβούς πρόβλεψης αυτών των χαρακτηριστικών πολλά χρονικά βήματα πριν εμφανιστούν, το Στρώμα Μετάφρασης μπορεί να προβλέψει τη ποιότητα της υπηρεσίας με τους δεδομένους ή διαφορετικούς πόρους. Ρυθμίζοντας τους τελευταίους εκ των προτέρων, μπορεί να επιτευχθεί προσαρμογή στους ποικίλους φόρτους εργασίας οριακά προτού εμφανιστούν, βελτιστοποιώντας κατά συνέπεια τη διαχείριση και το κέρδος των πόρων. Ο συνδυασμένος μηχανισμός απεικονίζεται στο Σχήμα 36.

Η ενότητα TS-ANN είναι βασισμένη σε ένα μη γραμμικό αυτοανάδρομο δίκτυο με εξωγενείς εισόδους (Non-Linear Autoregressive Network with Exogenous Inputs-NARX) ([109]). Αυτό χρησιμοποιεί τον τύπο που απεικονίζεται στην Εξίσωση 12 προκειμένου να προβλεφθούν οι μελλοντικές τιμές και έχει χρησιμοποιηθεί επιτυχώς στον τομέα πρόβλεψης χρονοσειράς ([110]). Χρησιμοποιεί κυρίως τις προηγούμενες τιμές του αρχικού σήματος y(t) μαζί με μια εξωγενή είσοδο u(t). Στην περίπτωση που εξετάζεται, θεωρείται ότι y=u, έτσι ώστε να προβλέπονται οι μελλοντικές τιμές του φόρτου με πληροφορίες μόνο σχετικά με τις προηγούμενες τιμές της ίδιας παραμέτρου. Επιπλέον, οι μη γραμμικές λειτουργίες των νευρώνων του ANN βοηθούν να

προσαρμοστούν στις ξαφνικές αλλαγές συμπεριφοράς, όπως τις αιχμές στη δραστηριότητα.



Σχήμα 36: Συνδυασμός χρονοσειράς πρόβλεψης (TS-ANN) και μηχανισμού μεταφράσεων (μετάφραση ANN) για την εκ των προτέρων ρύθμιση των απαραίτητων πόρων

$$y(t) = f(y(t-1), y(t-2,..., y(t-n_y),$$
$$u(t-1), u(t-2), ...u(t-n_u))$$

Εξίσωση 12: Συνάρτηση του NARX ANN

Μέσω της διαδικασίας εκπαίδευσης, το δίκτυο μαθαίνει πώς οι προηγούμενες τιμές έχουν επιπτώσεις στις ακόλουθες με την αναγνώριση ακολουθιών και διαμορφώνει τους συντελεστές κάθε παράγοντα (βάρη και πολώσεις). Κατόπιν, το σύνολο επικύρωσης χρησιμοποιείται προκειμένου να εξαχθεί το λάθος της προσέγγισης. Στο δίκτυο δίνεται το πρόσφατο παρελθόν της χρονοσειράς και αρχίζει να προβλέπει τις τιμές του συνόλου επικύρωσης. Αυτές συγκρίνονται έναντι των πραγματικών στο τμήμα αξιολόγησης.

Προκειμένου να βελτιστοποιηθεί η δομή του NARX ANN και να βελτιωθεί η απόδοσή της, χρησιμοποιήθηκε πάλι ένας γενετικός αλγόριθμος (GA) που καθόρισε τις κρισιμότερες παραμέτρους του δικτύου μέσω μιας επαναληπτικής εξελικτικής διαδικασίας. Σχετικά με το προηγούμενο στρώμα και την προσέγγιση ANN μετάφρασης,

Η σελίδα αυτή είναι σκόπιμα λευκή

σε αυτήν την περίπτωση θεωρήθηκαν και πρόσθετα χαρακτηριστικά γνωρίσματα ως παραμέτροι για το GA όπως τα εξής:

- Τύπος συνάρτησης εκπαίδευσης που χρησιμοποιείται
- Μέγεθος των προηγούμενων τιμών που απαιτούνται για μια πρόβλεψη
- Αριθμός εποχών που χρησιμοποιούνται για την εκπαίδευση

4.4 Υπόθεση Εργασίας και Επικύρωση Ακρίβειας Μηχανισμού

Για την επικύρωση του μηχανισμού χρησιμοποιήθηκε ένας ιστότοπος μορφής Wikipedia (βασισμένος στο [111]). Για αυτήν την εφαρμογή θεωρήθηκαν 3 είσοδοι, ο αριθμός χρηστών που έχουν πρόσβαση στον κεντρικό υπολογιστή, οι πυρήνες και η μνήμη RAM του VM. Για το VM χρησιμοποιήθηκε το Υπολογιστικό Νέφος Flexiscale ([112]). Ο μέσος χρόνος απόκρισης του εξυπηρετητή ορίστηκε σαν έξοδος KPI.

Οι ιδιοκτήτες του κεντρικού εξυπηρετητή το προσφέρουν στο ευρύ κοινό και χρειάζονται το KPI του να είναι κάτω από ένα συγκεκριμένο όριο. Κατά συνέπεια είναι ευθύνη του στρώματος PaaS να διατηρηθεί αυτό το όριο. Για το λόγο αυτό, χρησιμοποιεί τον μηχανισμό πρόβλεψης χρηστών για τις επόμενες χρονικές περιόδους, σε συνδυασμό με ένα ANN μετάφρασης με βάση τις παραπάνω εισόδους και εξόδους.

4.4.1 Δημιουργία ΤΝΔ Μετάφρασης

Για την δημιουργία του ΤΝΔ Μετάφρασης χρειάζεται ένα σύνολο δεδομένων για την εκπαίδευση και επικύρωση του δικτύου. Αυτό δημιουργήθηκε μέσω δειγματοληψίας. Αυτή η διαδικασία περιελάμβανε την λειτουργία του εξυπηρετητή σε διαφορετικές εικονικές μηχανές και την δημιουργία αιτημάτων μέσα από μία πειραματική διάταξη που προσομοίωνε διαφορετικούς αριθμούς χρηστών. Το αρχικό σύνολο δειγματοληψίας ήταν

στο φάσμα 10-1150 χρηστών, με βήμα 10. Τρεις εικονικοί πόροι στην πλατφόρμα Νέφους Flexiscale χρησιμοποιήθηκαν, ένας με 2 πυρήνες και 2 GB RAM, ένας με 4 πυρήνες και 4 GB RAM και ένας με 8 πυρήνες και 8 GB RAM. Αυτός ο τρόπος επιλέχτηκε επειδή τα ιστορικά στοιχεία τείνουν να κατανέμονται γύρω από ένα συγκεκριμένο διάστημα χρήσης, οπότε είναι γενικά ευκολότερο να προβλεφθεί η συμπεριφορά στην ιδιαίτερη περιοχή, λόγω υπερπροσφοράς δειγμάτων. Μέσω μιας γενικότερης διαδικασίας δειγματοληψίας η προτεινόμενη μέθοδος μπορεί να επικυρωθεί καλύτερα. Επιπλέον, σε πολλές περιπτώσεις, όπως την έναρξη λειτουργίας της υπηρεσίας, είναι πιθανόν να μην υπάρχουν διαθέσιμα ιστορικά στοιχεία. Ο ψευδοκώδικας για τον δειγματολήπτη εμφανίζεται στο Σχήμα 37.

Τα αποτελέσματα (ποσοστό των time outs, της μέσης καθυστέρησης απάντησης και της τυπικής απόκλισης των καθυστερήσεων) εμφανίζονται στο Σχήμα 38 για όλες τις διαμορφώσεις υλικού. Μετά από τη δειγματοληψία, 70% του συνόλου δεδομένων που αποκτήθηκε μέσω αυτής της διαδικασίας χρησιμοποιήθηκε προκειμένου να εκπαιδευθούν τα μοντέλα δικτύων, με τον ίδιο αλγόριθμο που αναπτύχθηκε στο Κεφάλαιο 2.5. Κατόπιν, το υπόλοιπο 30% χρησιμοποιήθηκε ως επικύρωση προκειμένου να ελεγχθεί η ακρίβεια των δικτύων στην πρόβλεψη των επιπέδων QoS (KPI) της εφαρμογής. Ο Πίνακας 9 περιέχει το μέσο απόλυτο σφάλμα (MAE) στο σύνολο επικύρωσης , μαζί με τα δομικά χαρακτηριστικά του δικτύου. Το παραχθέν ΤΝΔ είναι feed-forward back-propagation δίκτυο, που εκπαιδεύτηκε με τον αλγόριθμο levenberg-Marquardt για 150 εποχές.

Προκειμένου να συγκριθεί η προσέγγιση, χρησιμοποιήθηκε η μέθοδος γραμμικής παλινδρόμησης πολλών μεταβητών, με τις ίδιες εισόδους και στο ίδιο σύνολο

δεδομένων. Τα αποτελέσματα αυτής της μεθόδου, που ήταν πολύ αποθαρρυντικά, εμφανίζονται επίσης στον παρακάτω πίνακα (Πίνακας 9). Τα μεμονωμένα λάθη στην επικύρωση για την ίδια μέθοδο (mvregress) εμφανίζονται στο Σχήμα 39. Από αυτά είναι εμφανές ότι η συγκρινόμενη μέθοδος εμφανίζει πολύ σημαντικά σφάλματα, ακόμη και στην επικύρωση, και δεν μπορεί να χρησιμοποιηθεί με ασφάλεια κατά τη διάρκεια οποιασδήποτε κανονικής λειτουργίας.

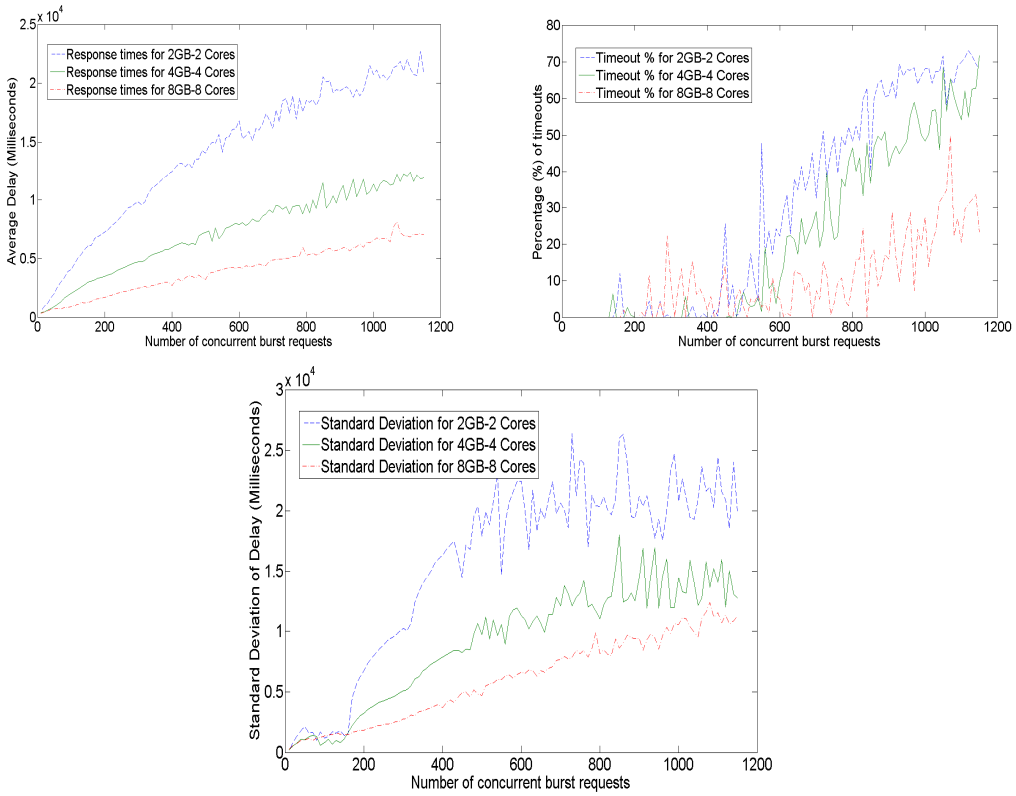
```
1.while traceline!= null
2.      hitSize=str2int(traceline)
3.      for i=1:hitSize
4.              raise separate Client threads
5.              for each thread //happens in parallel
6.                      for samples=1:max_samples
7.                              Get page
8.                              Log delays
9.                      end
10.             end
11.     end
12.     wait for all threads to finish
13.end while
```

Σχήμα 37: Ψευδοκώδικας δειγματολήπτη για τη προσομοίωση κίνησης

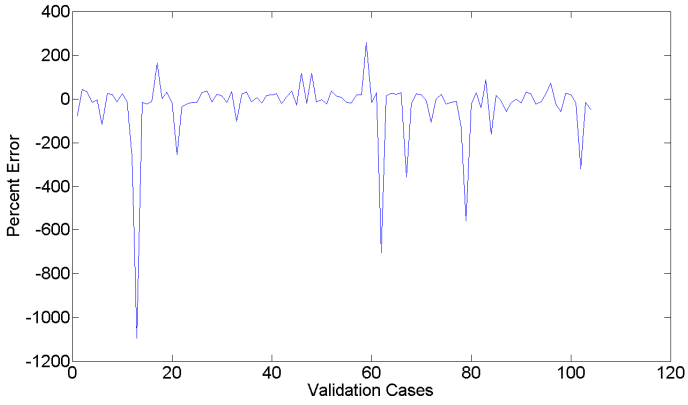
Πίνακας 9: Ακρίβεια Μεθόδων Στο Σύνολο Επικύρωσης και Δομικά Χαρακτηριστικά

Μέθοδος Πρόβλεψης	Νευρώνες ανά στρώμα	Συναρτήσεις Μεταφοράς ανά στρώμα	Μέσο Απόλυτο Σφάλμα Επικύρωσης
Wikipedia Server	3-3-2-1	Tansig-Tansig-Tansig-Purelin	3.46%
Multivariate Regression for Wikipedia Server	-	-	66.06%

Η σελίδα αυτή είναι σκόπιμα λευκή



Σχήμα 38: Συγκριτικά αποτελέσματα για τους χρόνους απόκρισης πειράματος δειγματοληψίας εξυπηρετητή Wikipedia, το ποσοστό των εκπρόθεσμων αιτήσεων και τη τυπική απόκλιση



Σχήμα 39: Σφάλματα επικύρωσης % για την μέθοδο γραμμικής παλινδρόμησης

Η σελίδα αυτή είναι σκόπιμα λευκή

4.5 Δειγματοστρεφής Προσομοίωση Κανονικής Λειτουργίας

Προκειμένου να αξιολογηθεί η προτεινόμενη συνδυασμένη προσέγγιση, ένα πραγματικό σύνολο δεδομένων χρειαζόταν, ώστε να πραγματοποιηθεί μια δειγματοστρεφής προσομοίωση κανονικής λειτουργίας (trace-driven simulation) . Αυτό θα έπρεπε να βασίζεται σε ένα ρεαλιστικό μοτίβο κίνησης που θα χρησιμοποιούνταν σαν είσοδος στο TS ANN για την πρόβλεψη της μελλοντικής δραστηριότητας του κεντρικού εξυπηρετητή. Κατόπιν η μετάφραση μέσω του Translation ANN θα εκτελούσε μια δεύτερη πρόβλεψη επιπέδων QoS για τους χρησιμοποιημένους πόρους βασισμένη σε αυτήν την πρόβλεψη της προσδοκώμενης χρήσης,. Αυτή η γενική πρόβλεψη τελικά συγκρίνεται με τις πραγματικές τιμές των χρόνων απόκρισης για τον πραγματικό αριθμό του φόρτου εργασίας.

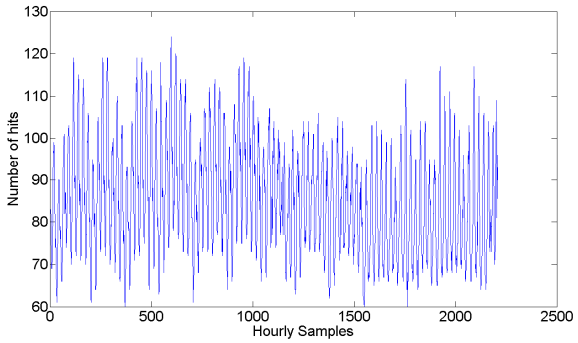
Έτσι κατά τη διάρκεια αυτής της κανονικής λειτουργίας, η αξιολόγηση είναι βασισμένη αρχικά στην πρόβλεψη των επόμενων τιμών των προσβάσεων για τον κεντρικό εξυπηρετητή τύπου Wikipedia. Με βάση αυτές τις αναμενόμενες τιμές, προβλέπουμε μέσω της μετάφρασης ANN το χρόνο απόκρισης για τη δεδομένη υποδομή. Τελικά αξιολογούμε το γενικό πλαίσιο πρόβλεψής μας μέσω της σύγκρισης του προβλεφθέντος χρόνου απόκρισης για την προβλεφθείσα κυκλοφορία με τον πραγματικό χρόνο απόκρισης της πραγματικής κυκλοφορίας.

Αυτό το συνδυασμένο σύνολο δεδομένων, σύμφωνα με την πραγματοποιηθείσα έρευνα στο διαδίκτυο, δεν ήταν διαθέσιμο. Προκειμένου να δημιουργηθεί, τα στοιχεία κυκλοφορίας από τις αιτήσεις ανά ώρα προς την αγγλική έκδοση της Wikipedia συλλέχθηκαν. Τα δείγματα περιελάμβαναν μια περίοδο τριών μηνών, από τον Ιούλιο μέχρι το Σεπτέμβριο 2011 και διαιρέθηκαν με 1000000 προκειμένου να μειωθεί ο χρόνος

που απαιτήθηκε για τη διαδικασία δειγματοληψίας και την ανάγκη για τους πόρους των πειραμάτων. Αυτό δεν αλλάζει τη μορφή του μοτίβου κυκλοφορίας επομένως και τη δυνατότητα του TS ANN να την προσδιορίσει. Η μορφή του γενικού δείγματος εμφανίζεται στο Σχήμα 40.

Προκειμένου να ληφθούν οι χρόνοι απόκρισης για την αξιολόγηση κανονικής λειτουργίας, πραγματοποιήθηκε μια δειγματοστρεφής προσομοίωση. Σε αυτή, το δείγμα αιτήσεων της Wikipedia χρησίμευσε ως είσοδος σε έναν πολυνηματικό πελάτη (multithreaded client) που είχε στόχο να προσομοιώσει τον αριθμό χρηστών που ζητούν τις σελίδες από τον εξυπηρετητή. Ο πελάτης υλοποιήθηκε με τη μορφή burst mode, που σημαίνει ότι όλα τα νήματα δημιουργούνταν την ίδια χρονική στιγμή, για την αναπαράσταση πιο έντονου φορτίου. Κάθε νήμα συνέλεξε 5 δείγματα των κλήσεων προς τον κεντρικό υπολογιστή, προκειμένου να συγκεντρωθεί ένα ικανοποιητικό σύνολο δεδομένων. Το αρχικό και τελικό 10%.από τα δείγματα απορρίφθηκε προκειμένου να αποβληθούν οποιαδήποτε μεταβατικά φαινόμενα, όπως προτείνεται στο [104]. Αυτό είναι απαραίτητο προκειμένου να αφαιρεθούν οι τιμές των χρόνων απόκρισης που δεν αντιπροσωπεύουν το σωστό αριθμό χρηστών (επειδή παραδείγματος χάριν δεν έχουν εκκινήσει όλα τα νήματα στην αρχή ή μερικά νήματα έχουν ολοκληρωθεί στο τέλος του διαστήματος μέτρησης). Κατά τη διάρκεια του χρόνου δειγματοληψίας, η γενική καθυστέρηση για κάθε κλήση μετρήθηκε στην πλευρά πελάτη. Ο ψευδοκώδικας για την πλευρά πελάτη εμφανίζεται στο Σχήμα 41.

Επιπλέον, διαφορετικά μεγέθη εικονικών μηχανών χρησιμοποιήθηκαν για τον κεντρικό εξυπηρετητή στο Υπολογιστικό Νέφος Flexiscale, προκειμένου να αποκτηθούν οι χρόνοι απόκρισης για τους διαφορετικούς πόρους.



Σχήμα 40: Μορφή των αιτήσεων ανά ώρα για την αγγλική Wikipedia (διαιρεμένες με 1000000) για την παρατηρηθείσα περίοδο (Ιουλίου-Σεπτεμβρίου 2011)

```
1.while traceline!= null
2.      hitSize=str2int(traceline)
3.      for i=1:hitSize
4.              raise separate Client threads
5.              for each thread //happens in parallel
6.                      for samples=1:max_samples
7.                              Get page
8.                              Log delays
9.                      end
10.             end
11.     end
12.     wait for all threads to finish
13.end while
```

Σχήμα 41: Ψευδοκώδικας για τον πολυνηματικό προσομοιωτή κλήσεων

Όπως αναφέρθηκε νωρίτερα, το μοτίβο κυκλοφορίας που συλλέχθηκε από την αγγλική Wikipedia χρησίμευσε ως είσοδος για το TS-ANN. Οι διαθέσιμες τιμές ήταν 2208 και χωρίστηκαν σε 2 μέρη. Οι πρώτες 1500 τιμές αποτέλεσαν το σύνολο εκπαίδευσης και επικύρωσης. . Οι τελικές 708 τιμές χρησιμοποιήθηκαν για τη δοκιμή κανονικής λειτουργίας. Αυτό εξυπηρέτησε δύο σκοπούς:

- Αξιολόγηση του τελικώς επιλεγμένου TS-ANN σε ένα χρονικό διάστημα που δεν χρησιμοποιήθηκε με κανένα τρόπο κατά τη διάρκεια της εκπαίδευσης, αντιπροσωπεύοντας μια περίοδο κανονικής λειτουργίας σε μια πραγμματική υποδομή

Η σελίδα αυτή είναι σκόπιμα λευκή

- Παροχή των προβλέψεων για αυτήν την περίοδο στη μετάφραση ANN για την απόκτηση των προβλεφθέντων αποτελεσμάτων χρόνου απόκρισης βασισμένων στην προσδοκώμενη χρήση.

Ο Πίνακας 10 περιέχει το δίκτυο, μαζί με την ακρίβειά του στη δοκιμή κανονικής λειτουργίας. Προκειμένου να συγκριθεί η προσέγγιση TS-ANN, χρησιμοποιήθηκε η ανοικτή βιβλιοθήκη OpenForecast της Java [113]. Το OpenForecast παρέχει ποικίλες μεθόδους και μια γενική μέθοδο για να λάβει το καλύτερο μοντέλο από όλες. Αυτή η λειτουργία χρησιμοποιήθηκε. Ο Πίνακας 10 περιέχει επίσης τη σύγκριση μεταξύ των δύο προσεγγίσεων.

Πίνακας 10: Χαρακτηριστικά και πρόβλεψη του TS ANN στην κανονική λειτουργία και σύγκριση με την βιβλιοθήκη OpenForecast

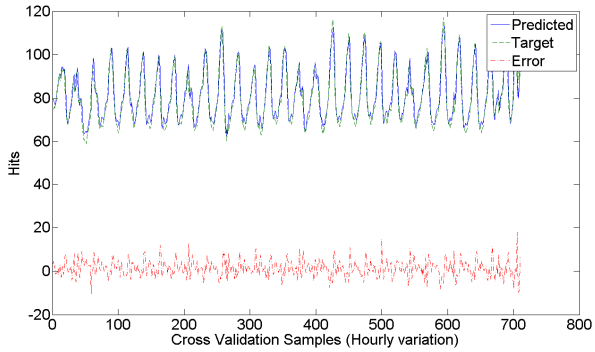
Μέθοδος	Προηγούμενες Τιμές	Μέσο Απόλυτο Σφάλμα(%)	Μέγιστο Απόλυτο Σφάλμα (%)
TS-ANN (Logsig-Purelin)	4	3.70	22.01
OpenForecast Best Method	10	5.32%	36.77%

Η ακρίβεια του TS-ANN για τη κανονική λειτουργία εμφανίζεται με περισσότερες λεπτομέρειες στο Σχήμα 42. Από αυτό είναι εμφανές ότι είναι σε θέση να προβλέψει τις πραγματικές τιμές με αυξανόμενη ακρίβεια για ένα μακροχρόνιο διάστημα πρόβλεψης. Η μέθοδος OpenForecast δεν απεικονίζεται στο Σχήμα 42α) για την καλύτερη ευκρίνεια των γραφικών παραστάσεων. Τα μεμονωμένα λάθη τοις εκατό και για τις δύο προσεγγίσεις συνδυάζονται στο Σχήμα 42β). Από αυτό η βελτιωμένη κατανομή των τιμών για το TS-ANN είναι εμφανής.

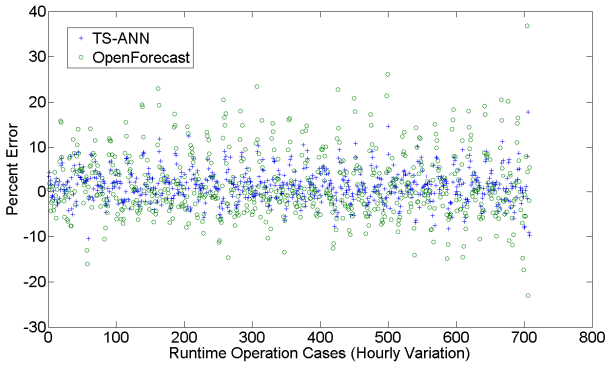
Οι προβλεφθείσες τιμές του TS-ANN που ελήφθησαν από αυτό το βήμα προωθήθηκαν στη μετάφραση ANN προκειμένου να εξαχθεί ο προσδοκώμενος χρόνος απόκρισης του κεντρικού εξυπηρετητή. Κατόπιν αυτές οι τιμές συγκρίθηκαν με τους πραγματικούς χρόνους που λήφθηκαν κατά τη διάρκεια της δειγματοστρεφούς προσομοίωσης για τις τελευταίες 708 τιμές. Αυτές οι τιμές δεν χρησιμοποιήθηκαν καθόλου κατά τη διάρκεια της εκπαίδευσης και της επιλογής και των δύο τύπων ANNs. Κατά συνέπεια μετρήθηκε το συνολικό σφάλμα της προσέγγισης μέσω του τελικού σφάλματος της μετάφρασης ANN καθώς περιελάμβανε και το σφάλμα πρόβλεψης του TS-ANN. Τα αποτελέσματα και για τις τρεις διαμορφώσεις υλικού εμφανίζονται στο Σχήμα 43. Ο Πίνακας 11 περιέχει την ακρίβεια της μετάφρασης ANN που μετρήθηκε κατά τη διάρκεια της κανονικής λειτουργίας έναντι της ακρίβειας επικύρωσης που παρουσιάστηκε στα προηγούμενα τμήματα. Είναι εμφανές ότι ενώ αυτό το γενικό λάθος (συνδυασμένες TS-ANN και μετάφραση ANN) είναι υψηλότερο από την περίπτωση επικύρωσης, είναι ακόμα πολύ ακριβές με μια μέση απόλυτη τιμή μικρότερη από 10%. Στο Σχήμα 44 τα % σφάλματα της αξιολόγησης κανονικής λειτουργίας απεικονίζονται, ανεξάρτητα από τον χρησιμοποιούμενο τύπο μηχανής. Ένα ενδιαφέρον συμπέρασμα από αυτή τη γραφική είναι ότι η προτεινόμενη μέθοδος υπερεκτιμά κυρίως το ποσό πόρων που απαιτούνται, καθώς η πλειοψηφία των σφαλμάτων είναι θετική και οι θετικές μέγιστες τιμές είναι υψηλότερες από τις αντίστοιχες αρνητικές.

Πίνακας 11: Ακρίβεια μετάφρασης ANN χρόνου απόκρισης σε κανονική λειτουργία σε σύγκριση με τη φάση επικύρωσης

Μέσο Απόλυτο Σφάλμα Επικύρωσης	Μέγιστο Απόλυτο Σφάλμα Επικύρωσης	Μέσο Απόλυτο Σφάλμα Κανονικής Λειτουργίας	Μέγιστο Απόλυτο Σφάλμα Κανονικής Λειτουργίας
3.46%	14.7%	9.61%	45.03%

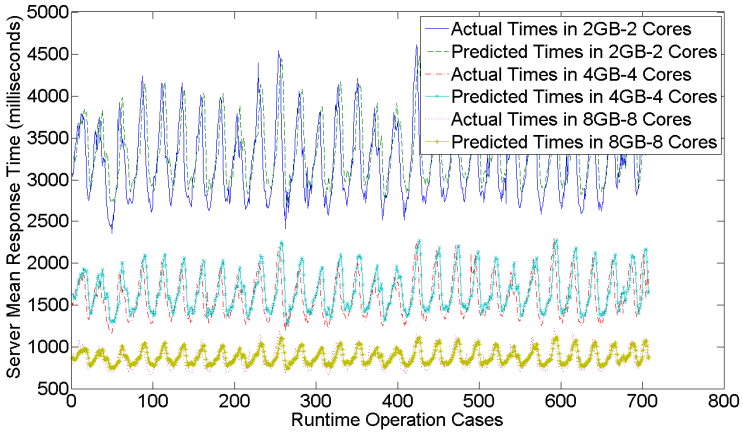


(α)



(β)

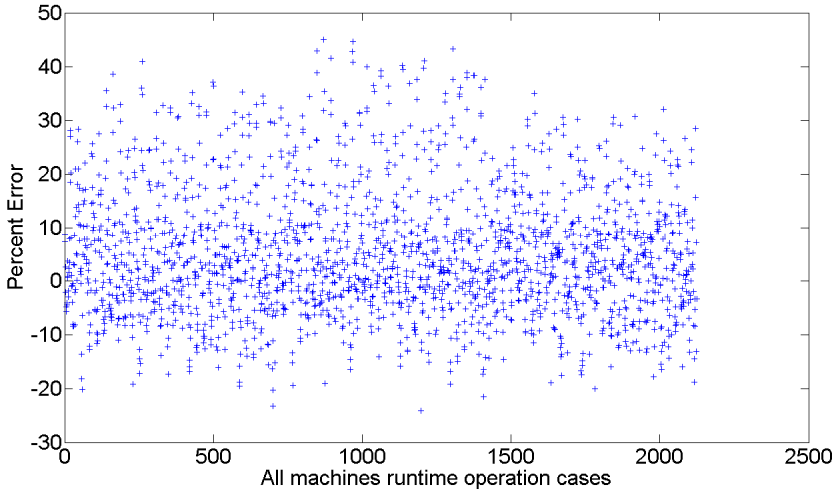
Σχήμα 42: Πρόβλεψη της προσδοκώμενης κυκλοφορίας από το TS-ANN για τη κανονική λειτουργία: α) στόχος και προβλεφθέν λάθος τοις εκατό β) σε κάθε περίπτωση επικύρωσης για το TS-ANN και το μοντέλο OpenForecast



Σχήμα 43: Συνολικό σφάλμα προσέγγισης για τις περιπτώσεις προσομοίωσης κανονικής λειτουργίας χρόνου εκτέλεσης (συμπεριλαμβανομένου του λάθους αναμονής και του λάθους μεταφράσεων)

Η σελίδα αυτή είναι σκόπιμα λευκή

Αυτό οδηγεί σε ασφαλέστερη λειτουργία, ειδικά σε περιπτώσεις εφαρμογών πραγματικού χρόνου, σε σύγκριση με μια προσέγγιση που υποεκτιμά τους απαιτούμενους πόρους. Επιπλέον, μειώνει το μέγεθος ενός συντελεστή ασφάλειας βασισμένου σε μια πιθανοτική προσέγγιση για την αποφυγή των παραβιάσεων SLA σε ένα δεδομένο ποσοστό (π.χ. 95%).



Σχήμα 44: Συνολικό % σφάλμα για όλες τις εικονικές μηχανές για τις περιπτώσεις προσομοίωσης κανονικής λειτουργίας

4.6 Συμπεράσματα

Το πρόβλημα μεταφράσεων από τους όρους επιπέδου εργασίας (φόρτος εργασίας και QoS) στις ιδιότητες επιπέδου πόρων είναι ένα σημαντικό πρόβλημα στη σύγχρονη διαστρωμάτωση των υπηρεσιοστρεφών υποδομών Υπολογιστικού Νέφους, γεγονός που οφείλεται στους διακριτούς ρόλους μεταξύ των στρωμάτων (SaaS, PaaS, IaaS) και του περιορισμένου ποσού πληροφοριών που είναι διαθέσιμο από ένα στρώμα σε άλλο. Επιπλέον, η αυτοδιαχείριση μιας υπηρεσίας που εκτελείται σε μια εικονικοποιημένη υποδομή απαιτεί την a priori γνώση της προσδοκώμενης μελλοντικής κυκλοφορίας.

Η σελίδα αυτή είναι σκόπιμα λευκή

Σε αυτό το κεφάλαιο, μια γενική προσέγγιση που είναι βασισμένη σε μια δύο επιπέδων πρόβλεψη (μελλοντική συμπεριφορά και μετάφραση φόρτου εργασίας σε KPIs και τους αναγκαίους πόρους) χρησιμοποιήθηκε ως διαμεσολαβητής για το στρώμα PaaS, στην προσπάθεια του τελευταίου να υπολογιστεί πόσοι πόροι απαιτούνται για μια περίπτωση εφαρμογής με μεταβαλλόμενες απαιτήσεις. Εξαιτίας της αυτοματοποίησης αυτής, ο ιδιοκτήτης της εφαρμογής δεν χρειάζεται να γνωρίζει τις μελλοντικές τιμές των όρων υψηλού επιπέδου (π.χ. αναμενόμενη κίνηση), παρά μόνο την επιθυμητή ποιότητα υπηρεσίας.

Η συνδυασμένη προσέγγιση αξιολογήθηκε μέσω μιας λεπτομερούς δειγματοστρεφούς προσομοίωσης πραγματικού τρόπου λειτουργίας από τον πραγματικό κόσμο, που δίνει ελπιδοφόρα αποτελέσματα για τη γενική δυνατότητα να προβλεφθεί και να προ-διαμορφωθεί η συμπεριφορά του συστήματος. Η ακρίβεια του συστήματος στη συνδυασμένη πρόβλεψη παρουσίασε μεν μείωση σε σχέση με το απλό επίπεδο μετάφρασης αλλά παρέμεινε σε πολύ ικανοποιητικά επίπεδα, μικρότερα του 10%.

Για το μέλλον, μια πτυχή που αξίζει να διερευνηθεί είναι η ενσωμάτωση περισσότερων παραμέτρων διαμόρφωσης όπως η ενεργειακή αποδοτικότητα, είτε άμεσα στη φάση μετάφρασης, για τη βελτιστοποίηση της συμπεριφοράς εφαρμογής, είτε έμμεσα μέσω της πρόβλεψης (με την ενότητα χρονικής σειράς) της παραγωγής της ενέργειας από τις ανανεώσιμες πηγές στο στρώμα υποδομής. Ειδικά η τελευταία μπορεί να υπόκειται σε μεγάλες παραλλαγές (λόγω της αντίστοιχης μεταβολής των καιρικών φαινομένων) και μπορεί να οδηγήσει έναν προμηθευτή IaaS στη βέλτιστη διαχείριση των πόρων όσον αφορά την κατανάλωση ενέργειας. Το γεγονός αυτό θα βοηθήσει και στην

συμμόρφωση με τα ολοένα και αυστηρότερα νομικά πλαίσια εκπομπής ρύπων, ειδικά εντός της επικράτειας της Ευρωπαϊκής Ένωσης.

5 *Επίδραση Παραγόντων Υποδομής Στην Απόδοση Εφαρμογών*

Στο παρόν κεφάλαιο περιγράφεται το πρόβλημα της διαφοροποίησης της απόδοσης των πόρων που διαθέτει ένας πάροχος Υπολογιστικού Νέφους στους εξωτερικούς πελάτες της υποδομής. Οι πόροι αυτοί έχουν την μορφή Εικονικών Μηχανών, οι οποίες καταλαμβάνουν συγκεκριμένο ποσοστό ενός επεξεργαστή.

5.1 Ορισμός του Προβλήματος

Κατά τη διάρκεια των τελευταίων ετών, συγκέντρωση διαφορετικών διακομιστών σε ένα φυσικό υπολογιστικό πόρο μέσω της χρήσης τεχνικών virtualization και εργαλεία όπως το VMware [76], XEN [77], KVM [78] χρησιμοποιείται ευρέως σε κέντρα δεδομένων ή σε νέα πρότυπα υπολογιστών όπως το Cloud Computing [1]. Μέσω αυτής της τεχνικής, εφαρμογές με διαφορετικά χαρακτηριστικά και απαιτήσεις μπορούν να εκτελούνται μέσα σε Εικονικές Μηχανές (Virtual Machines-VMs) στον ίδιο φυσικό

πολυεπεξεργαστικό πόρο. Με αυτό τον τρόπο, επιτυγχάνονται αυξημένα επίπεδα ασφάλειας, ανοχή σφαλμάτων, απομόνωση, δυναμικότητα στην κατανομή των πόρων και ευκολία διαχείρισης με μικρότερο αριθμό μηχανημάτων.

Ωστόσο, ορισμένα ζητήματα προκύπτουν από αυτή την εξέλιξη στην πληροφορική, όπως η υποβάθμιση της απόδοσης των εφαρμογών, λόγω της χρήσης του στρώματος του virtualization [89] . Εκτός από το επιπλέον επίπεδο, μια σειρά άλλων παραμέτρων μπορεί επίσης να επηρεάσει την απόδοση σε κατανεμημένα και εικονικά περιβάλλοντα λόγω της συνδρομολόγησης περισσότερων της μίας VM ανά πόρο. Αυτές οι παράμετροι μπορεί να είναι το ποσοστό της CPU που διατίθεται για τη VM και ο τρόπος (περίοδος) που το ποσοστό αυτό εκχωρείται και μπορεί να επηρεάσει την απόδοση. Επιπλέον, οι αποφάσεις ενοποίησης, ανάλογα με την πολυπύρηνη αρχιτεκτονική και το επίπεδο διαμοιρασμού εσωτερικών κυκλωμάτων μεταξύ των πυρήνων, μπορεί να έχουν επιπτώσεις στην απόδοση των εφαρμογών. Επίσης, διαφορετικοί τύποι εφαρμογών έχουν διαφορετικές επιπτώσεις ο ένας στον άλλο στην περίπτωση ταυτόχρονης εκτέλεσής τους στον ίδιο φυσικό πόρο. Αυτό συμβαίνει λόγω των εσωτερικών χαρακτηριστικών κάθε εφαρμογής, όπως το είδος των υπολογισμών, η χρήση των ειδικών υπο-κυκλωμάτων της CPU, τον τρόπο πρόσβασης στη μνήμη κλπ.

Η υποβάθμιση της απόδοσης σε αυτήν την περίπτωση μπορεί να οδηγήσει στην ανάγκη ανάθεσης περισσότερης επεξεργαστικής ισχύος σε κάθε εφαρμογή ώστε να ισοσκελιστεί η αλληλεπίδραση με τις ταυτόχρονα εκτελούμενες εφαρμογές στις γειτονικές VM. Ωστόσο, η τρέχουσα επιχειρηματική λογική SPI των υποδομών Υπολογιστικού Νέφους [1] προσδιορίζει διακριτούς ρόλους μεταξύ της οντότητας που προσφέρει μια πλατφόρμα (PaaS παρόχου) και του φορέα που προσφέρει τους πόρους

που η πλατφόρμα αυτή μπορεί να χρησιμοποιεί (IaaS ή Cloud πάροχο). Το μοντέλο αυτό, σε συνδυασμό με τα προαναφερθέντα θέματα, δημιουργεί το εξής πρόβλημα:

• Ο φορέας της πλατφόρμας PaaS έχει υπολογίσει ότι αρκούν X πόροι για την επίτευξη των χαρακτηριστικών ποιότητας υπηρεσίας (QoS) της εφαρμογής που εκτελείται στο εσωτερικό μιας VM. Η εκτίμηση αυτή πιθανότατα έχει γίνει μετά την παρατήρηση της συμπεριφοράς της εφαρμογής κατά την εκτέλεσή της χωρίς ταυτόχρονες εικονικές μηχανές στον ίδιο πόρο.

• Ο πάροχος IaaS στον οποίο υποβάλλεται η αίτηση, σε μια προσπάθεια να μεγιστοποιήσει το κέρδος, θα δρομολογήσει αυτή τη VM για εκτέλεση μαζί με άλλες VMs που είναι εισηγμένες στην υπολογιστική υποδομή. Ο τρόπος αυτής της ανάθεσης είναι άγνωστος για το στρώμα PaaS. Αυτό δεν σημαίνει ότι ο πάροχος IaaS θα δώσει λιγότερους πόρους. Το ίδιο ποσό που ζητείται θα διατεθεί.

• Η μείωση της απόδοσης λόγω της προαναφερθείσας αλληλεπίδρασης μεταξύ των εικονικών μηχανών θα μεταφραστεί σε μείωση των επιπέδων ποιότητας υπηρεσίας για την εφαρμογή.

• Η παραβίαση της εμπιστοσύνης μεταξύ των PaaS και IaaS παρόχων είναι αναπόφευκτη. Ο πρώτος θα θεωρήσει ότι ο τελευταίος τον έχει εξαπατήσει και ότι η ανάθεση των πόρων στην εικονική του μηχανή ήταν μικρότερη από ό, τι συμφωνηθεί. Ο τελευταίος θα θεωρήσει ότι ο πρώτος έκανε μια κακή εκτίμηση και προσπαθεί να μεταφέρει την ευθύνη στην υπολογιστική υποδομή.

Ως εκ τούτου, είναι επιτακτική ανάγκη για έναν πάροχο IaaS να είναι σε θέση να προβλέψει με ακρίβεια την αλληλεπίδραση που παράγεται από τη στρατηγική συγχώνευσης και ως εκ τούτου να αυξήσει εκ των προτέρων τον αριθμό των πόρων που

διατίθενται σε μια εικονική μηχανή αναλόγως. Αυτό όχι μόνο θα οδηγήσει σε βελτίωση της φήμης, αλλά θα δώσει τη δυνατότητα στον IaaS πάροχο να βελτιστοποιήσει την κατανομή των VMs ανά πόρο ώστε να ελαχιστοποιήσει την αλληλεπίδραση μεταξύ τους, ανάλογα με τον τύπο του φόρτου εργασίας αυτών των VMs και την εκτιμώμενη παρεμβολή ενός τύπου φόρτου εργασίας πάνω στους άλλους .

Ο σκοπός του παρόντος κεφαλαίου είναι να ληφθεί υπόψη ένα ευρύ φάσμα των παραμέτρων που επηρεάζουν την απόδοση των εφαρμογών όταν εκτελούνται σε συγχωνευμένες και εικονικές υποδομές και την ποσοτικοποίηση αυτής της επιβάρυνσης. Ως εκ τούτου, οι IaaS πάροχοι μπορούν να εκτελούν ευφυείς αποφάσεις σχετικά με την τοποθέτησή της VM στους φυσικούς κόμβους. Αυτό θα μετριάσει την παραπάνω πρόβλημα και θα βοηθήσει το στρώμα IaaS να ελαχιστοποιήσει την ανάγκη για ανάθεση αυξημένης ισχύος. Ως ενδεικτικές εφαρμογές θεωρούμε τα 6 Matlab benchmarks, εξαιτίας του γεγονότος ότι αντιπροσωπεύουν μια ευρεία περιοχή υπολογιστικού φόρτου εργασίας, από μαθηματικούς υπολογισμούς μέχρι επεξεργασία γραφικών, και είναι εύκολο να επαναχρησιμοποιηθούν από άλλους ερευνητές. Διερευνώνται οι ακόλουθες πτυχές:

• Αποφάσεις χρονοδρομολόγησης εργασιών πάνω στην CPU με βάση τον EDF-based χρονοπρογραμματισμό (μια εργασία έχει μερίδιο Q σε κάθε περίοδο P), τόσο σε όρους ποσοστού της CPU που έχει ανατεθεί (Q / P) όσο και στο μέγεθος της περιόδου για κάθε ανάθεση (P). Αυτός ο τύπος προγραμματισμού [94], ο οποίος είναι πολύ δημοφιλής στον τομέα των συστημάτων πραγματικού χρόνου, χρησιμοποιείται για να εξασφαλιστεί ότι οι αναθέσεις των ποσοστών CPU στις VMs θα ακολουθηθούν και για να εξασφαλίσει χρονική απομόνωση μεταξύ της εκτέλεσης διαφορετικών εργασιών.

• Διαφορετικοί συνδυασμοί benchmark tests, ώστε να ανακαλύψουμε εκείνα που προκαλούν λιγότερες παρεμβολές (και ως εκ τούτου επιβάρυνση), όταν εκτελούνται ταυτόχρονα στον ίδιο κόμβο

• Διαφορετικές αποφάσεις τοποθέτησης και πώς αυτές επηρεάζουν την επιβάρυνση για τον ίδιο συνδυασμό benchmarks (τοποθέτηση στον ίδιο πυρήνα, σε γειτονικούς ή μη γειτονικούς πυρήνες σε ένα φυσικό κόμβο, σε σύγκριση με την αυτόνομη εκτέλεση). Ως γειτονικούς πυρήνες θεωρούμε αυτούς που μοιράζονται μνήμη L2 cache.

• Ικανότητα πρόβλεψης εκ των προτέρων αυτής της επιβάρυνσης, προκειμένου να επιλεγεί ο πλέον κατάλληλος τρόπος ανάθεσης VMs ανά πυρήνα που θα προκαλεί την μικρότερη επιβάρυνση. Για την επίτευξη αυτού του στόχου, τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ) χρησιμοποιούνται, τα οποία δημιουργούνται μέσω ενός εξελικτικού (GA-based) αλγόριμου. Ο στόχος είναι ο αυτόματος σχεδιασμός και η βελτιστοποίηση των παραμέτρων των δικτύων. Η τοπολογία των δικτύων είναι δυναμική όσον αφορά τον αριθμό των κρυμμένων στρωμάτων, τους νευρώνες ανά στρώμα και τις συναρτήσεις μεταφοράς και αποφασίζεται μέσα από τη διαδικασία βελτιστοποίησης του Γενετικού Αλγόριθμου (ΓΑ).

Το υπόλοιπο του κεφαλαίου είναι δομημένο ως εξής. Στη Παράγραφο 5.2, αναφέρονται παρόμοιες προσεγγίσεις στον σχετικό τομέα, ενώ στη Παράγραφο 5.3 γίνεται εκτεταμένη ανάλυση των παραμέτρων έρευνας. Η Παράγραφος 5.4 περιέχει την περιγραφή των δοκιμών και της διαδικασίας μέτρησης, ενώ η Παράγραφος 5.5 παρουσιάζει τα αναλυτικά αποτελέσματα από τα πειράματα. Η Παράγραφος 5.6 περιγράφει την Νευρογενετική προσέγγιση προκειμένου να προβλεφθεί εκ των προτέρων το αποτέλεσμα και τη συγκρίνει με την πολυ-μεταβλητή μέθοδο γραμμικής

παλινδρόμησης, ενώ η Παράγραφος 5.7 περιγράφει τα γενικά συμπεράσματα από τη μελέτη αυτή και τις προθέσεις για το μέλλον.

5.2 Σχετικές Εργασίες

Στο παρελθόν, έχουμε ερευνήσει εν μέρει ορισμένες από τις παραμέτρους που εξετάζονται εδώ. Σε εκείνη την περίπτωση ([36][68]), το κύριο βάρος δόθηκε στην επίδραση των παραμέτρων χρονοπρογραμματισμού (% των μεριδίων CPU και της περιόδου) στην ποιότητα υπηρεσίας που προσφέρεται από συγκεκριμένες εφαρμογές (κυρίως διαδραστικές), όταν αυτές εκτελούνται κατ'αποκλειστικότητα σε έναν υπολογιστικό κόμβο.

Δεδομένου ότι οι τεχνολογίες virtualization έχουν γίνει πολύ δημοφιλείς κατά τη διάρκεια των τελευταίων ετών, ένας μεγάλος αριθμός από ενδιαφέρουσες ερευνητικές προσπάθειες υπάρχουν. Στο [79], οι συγγραφείς ερευνούν την επιβάρυνση που εισάγεται μέσω της χρήσης των διαφόρων εργαλείων virtualization, με σκοπό να ανακαλύψουν τον πιο αποτελεσματικό. Στο [80], ο βασικός στόχος είναι η κλιμάκωση του αριθμού εικονικών μηχανών με ανάμικτο φόρτο εργασίας και την επίδραση στην απόδοσή τους όταν ενσωματώνονται στον ίδιο φυσικό πόρο. Τα κριτήρια αξιολόγησης βασίζονται σε εφαρμογές διακομιστή. Με την εκτεταμένη χρήση των τεχνολογιών σύννεφου, εφαρμογές που οραματίστηκε να είναι μέρος του φόρτου εργασίας τους μπορεί να έχουν πιο πολύπλοκο φόρτο εργασίας εκτός από τα παραδοσιακά client server applications. Τέτοια παραδείγματα μπορεί να είναι η περίπτωση του Digital Film Postproduction [93] του έργου IRMOS ή η διαθεσιμότητα λογισμικού μαθηματικού προγραμματισμού και των εργαλείων βελτιστοποίησης όπως το GAMS σε υποδομές cloud [51]. Επιπλέον, καμία χρονική απομόνωση μεταξύ των εικονικών μηχανών δεν χρησιμοποιείται.

Μια πολύ ενδιαφέρουσα παρόμοια προσέγγιση εμφανίζεται στο [81]. Στην περίπτωση αυτή, οι συγγραφείς μελετούν την επίδραση στις επιδόσεις συνδυασμών στοιχειωδών εφαρμογών όταν τρέχουν μέσα σε συνδρομολογημένες VMs. Μία ενδιαφέρουσα μεθοδολογία βαθμολόγησης παρουσιάζεται που μπορεί επίσης να χρησιμοποιηθεί για την ταξινόμηση των εφαρμογών. Τα χαρακτηριστικά του φόρτου εργασίας για κάθε εφαρμογή συγκεντρώνονται προκειμένου να χρησιμοποιηθούν ως μια σύγκριση στο μέλλον, για τον εντοπισμό και την ταξινόμηση άγνωστων εφαρμογών που πρόκειται να εισαχθούν στην υποδομή και ως προβλέπτες σε ένα μοντέλο γραμμικής παλινδρόμησης. Η κύρια διαφορά της προσέγγισής μας έγκειται στο γεγονός ότι χρησιμοποιούμε έναν περιορισμένο αριθμό ειδικών benchmark tests, με τη χρήση του προγραμματισμού πραγματικού χρόνου και μια διαφορετική μέθοδο πρόβλεψης που βασίζεται σε ΤΝΔ. Ειδικά η χρήση του real time χρονοπρογραμματισμού εξασφαλίζει τη χρονική απομόνωση των VMs όταν εκτελούνται ταυτόχρονα στον ίδιο επεξεργαστή και τη δίκαιη κατανομή των ποσοστών των πόρων που χρησιμοποιεί κάθε VM. Επιπλέον, ερευνούμε την επίδραση διαφορετικών τρόπων ανάθεσης των VMs στους πυρήνες ενός πολυ-επεξεργαστικού φυσικού πόρου με διαφορετικά επίπεδα διαμοιρασμού φυσικών κυκλωμάτων όπως μνήμες, δίαυλοι κλπ. και διαφορετικές ρυθμίσεις όσον αφορά το ποσοστό CPU και την περίοδο ανάθεσης για κάθε στοιχειώδη εφαρμογή.

Στο [82], μια πολλά υποσχόμενη προσέγγιση για το δυναμικό χειρισμό των πόρων στους οποίους εκτελούνται ταυτόχρονα VMs απεικονίζεται, με σκοπό τη βελτιστοποίηση της χρήσης του συστήματος και των χρόνων απόκρισης της εφαρμογής. Βασίζεται σε συστάσεις offline και online διορθώσεις. Ωστόσο αυτό το έργο επικεντρώνεται γύρω από

εφαρμογές βάσεων δεδομένων. Στο [83], ο σχεδιασμός και μία μεθοδολογία επικύρωσης ειδικών benchmarks για εικονικές μηχανές παρουσιάζεται, όπου και διερευνώνται οι επιπτώσεις ταυτόχρονης εκτέλεσης. Στο [84], ένας αριθμός διαφορετικών αναθέσεων εργασιών σε πυρήνες μελετάται στο πλαίσιο της έρευνας, όπως η εκτέλεση στον ίδιο ή σε γειτονικούς πυρήνες, μέσω Hyper-Threading ή όχι. Τα πειράματα έχουν επικεντρωθεί σε 3 benchmarks και χωρίς να λαμβάνονται υπό εξέταση παραμέτροι χρονοπρογραμματισμού της CPU. Οι μέθοδοι πρόβλεψης βασίζονται σε αναλυτική μοντελοποίηση.

Στο [85], το VSCBenchmark περιγράφεται, μια ανάλυση των δυναμικών επιδόσεων των VMs όταν ξεκινούν νέες εικονικές μηχανές. Οι συγγραφείς διερευνούν τη συμπεριφορά των συν-εκτελούμενων VMs τόσο για την έναρξη όσο και για τη διατήρηση της εκτέλεσής τους στον ίδιο φυσικό κόμβο. Η διαδικασία εκκίνησης για μία VM καταναλώνει μεγάλη υπολογιστική ισχύ, γεγονός που μπορεί να οδηγήσει άλλες συν-εκτελούμενες VMs σε έλλειψη πόρων. Στην περίπτωση που μελετάται εδώ, αυτό αποφεύγεται με τη χρήση του χρονοπρογραμματισμού πραγματικού χρόνου που περιορίζει τους πόρους που έχουν ανατεθεί σε μια εργασία (όπως ένα VM) για ένα προκαθορισμένο ποσοστό. Έτσι επιτυγχάνεται χρονική απομόνωση μεταξύ των εργασιών που εκτελούνται ταυτόχρονα.

Στο [86], μια ενδιαφέρουσα προσέγγιση για την δρομολόγηση VMs παρουσιάζεται, με δύο τρόπους λειτουργίας (υψηλής απόδοσης και ταυτόχρονη), η οποία στοχεύει στην αποτελεσματική διαχείριση του χρόνου της CPU για τα στάδια όπου οι VMs έχουν πολλαπλά threads που τρέχουν ταυτόχρονα. Αυτή η προσέγγιση επικεντρώνεται περισσότερο στις πτυχές χρονοπρογραμματισμού και δεν λαμβάνει

υπόψη την επίδραση από την ταυτόχρονη εκτέλεσηVMs στον ίδιο υπολογιστικό κόμβο. Ο προσομοιωτής VirtualGEMS για τον έλεγχο διαφορετικών configurations της L2 μνήμης cache μεταξύ διαφορετικών ταυτόχρονα εκτελούμενων VMs παρουσιάζεται στο [87].

Ένα εργαλείο συγκριτικής αξιολόγησης ειδικά για την απόδοση VM απεικονίζεται στο [88]. Ενώ επισημαίνει κάποια σημαντικά χαρακτηριστικά για τη συγκριτική αξιολόγηση των virtualized εφαρμογών, επικεντρώνεται κυρίως σε έναν τύπο του φόρτου εργασίας (linux kernel compilation). Μια έρευνα σε τεχνολογίες virtualization παράλληλα με τα αποτελέσματα των επιδόσεων για διαφορετικές συνθέσεις περιγράφεται στο [90]. Στα [95] και [96], μια έρευνα πραγματοποιείται σχετικά με τη δυνατότητα να βελτιωθεί η χρονική απομόνωση ανάμεσα σε Virtual Machines που εκτελούνται ταυτόχρονα στον ίδιο πυρήνα, χρησιμοποιώντας τον χρονοπρογραμματιστή IRMOS real-time scheduler, με έμφαση σε υψηλής έντασης υπολογιστικά και δικτυακά φορτία.

Στο [97] παρουσιάζεται μια πολύ ενδιαφέρουσα προσέγγιση, για μια προσαρμοστική σε πραγματικό χρόνο διαχείριση των VMs σε φορτία εργασίας High Performance Computing (HPC). Αυτός ο μηχανισμός χρησιμοποιείται κατά τη διάρκεια της λειτουργίας της υποδομής HPC, στην οποία τα VMs έχουν μια προκαθορισμένη περίοδο εκτέλεσης που αποφασίζεται από την πολιτική κοινής χρήσης της υποδομής. Ο κύριος στόχος είναι να υπολογίζει σε πραγματικό χρόνο, αν η εργασία που περιλαμβάνεται στο VM πρόκειται να ολοκληρωθεί εγκαίρως ή όχι. Εκείνες που έτσι κι αλλιώς δεν θα τελειώσουν εντός της προθεσμίας τερματίζονται πρόωρα, προκειμένου να βελτιωθεί το ποσοστό επιτυχίας της συνολικής υποδομής. Κατά την άποψή μας, το έργο

αυτό μπορεί να χρησιμοποιηθεί σε συνδυασμό με την προσέγγιση που παρουσιάζεται στο παρόν έγγραφο. Τα μοντέλα που περιγράφονται εδώ μπορούν να χρησιμοποιηθούν για την παρατήρηση και την ελαχιστοποίηση του overhead της ειδικής χορήγησης μεριδίου για τους διάφορους κόμβους, ενώ το έργο που παρουσιάζεται στο [97] μπορεί να χρησιμοποιηθεί για την βελτιστοποίηση της virtualized υποδομής κατά τη διάρκεια της λειτουργίας της πλατφόρμας.

Σχετικές εργασίες για βελτιστοποίηση ΤΝΔ μέσω ΓΑ παρουσιάστηκαν στο Κεφάλαιο 2.2.

5.3 Παράμετροι έρευνας

Σε ένα λειτουργικό σύστημα γενικού σκοπού, όταν δύο εργασίες αρχίσουν να εκτελούνται, ανταγωνίζονται μεταξύ τους για τον υποκείμενο υπολογιστικό πόρο. Η ισχύς του τελευταίου που εκχωρείται σε κάθε εργασία χωρίζεται ανάλογα με τις χρονικές ανάγκες και τις απαιτήσεις τους. Ωστόσο, αυτό έχει ως αποτέλεσμα την αδυναμία πρόβλεψης της ποιότητας υπηρεσίας που προσφέρεται από μια εφαρμογή που τρέχει μέσα σε ένα VM, σε κοινά χρησιμοποιούμενες υποδομές. Το ποσοστό της CPU που διατίθενται για αυτό μπορεί να διαφέρει σημαντικά ανάλογα με τις συστεγαζόμενες εργασίες-εικονικές μηχανές (όπως γίνεται στην εκκίνηση ενός άλλου VM στο [84]). Μια λύση για τον περιορισμό αυτό προέρχεται από τα συστήματα πραγματικού χρόνου. Μέσω της χρήσης χρονοδρομολογητών εργασιών βασισμένων στον Earliest Deadline First αλγόριθμο, το ποσοστό CPU που αποδίδεται σε μια εργασία μπορεί να είναι εγγυημένο. Σε γενικές γραμμές, για αυτόν τον τύπο χρονοπρογραμματισμού, υπάρχουν δύο σημαντικές παράμετροι, ο προϋπολογισμός Q και η περίοδος P. Q είναι ο καθαρός υπολογιστικός χρόνος CPU στο φυσικό κόμβο που έχει εκχωρηθεί σε μια εργασία κατά

τη διάρκεια μιας περιόδου P. Έτσι, ο λόγος Q / P είναι το ποσοστό εκχώρησης που προκύπτει στο μηχάνημα. Ωστόσο, για ένα συγκεκριμένο ποσοστό, διαφορετικά granularities μπορεί να υπάρχουν, ανάλογα με το P (για παράδειγμα, το 50% μπορεί να σημαίνει 50 χιλιοστά του δευτερολέπτου κάθε 100 χιλιοστά του δευτερολέπτου ή μπορεί να σημαίνει 250 κάθε 500 χιλιοστά του δευτερολέπτου). Ένα ενδιαφέρον ερώτημα είναι πώς αυτές οι παράμετροι επηρεάζουν την απόδοση των εφαρμογών.

Όπως διερευνήθηκε στα [36] και [68], η επίδραση των παραμέτρων χρονοπρογραμματισμού Q και P μπορεί να επηρεάσει την απόδοση των εφαρμογών με διάφορους τρόπους. Για παράδειγμα, στις παραπάνω εργασίες μας που διερευνούν διαδραστικές εφαρμογές, διαφορετικές κοκκοποιήσεις του ίδιου ποσοστού ανάθεσης CPU μπορούν να οδηγήσουν σε αύξηση της τυπικής απόκλισης των χρόνων απόκρισης ενός διακομιστή. Το συμπέρασμα από την ανάλυση αυτή ήταν ότι για διαδραστικές εφαρμογές, η χρήση μικρής περιόδου χρονοπρογραμματισμού P οδηγεί σε πιο σταθερή απόδοση για το ίδιο ποσοστό της CPU. Στο κεφάλαιο αυτό εστιάζουμε σε 6 εφαρμογές υπολογιστικής έντασης. Για τις εφαρμογές αυτές παρατηρείται η επίδραση των αποφάσεων χρονοπρογραμματισμού (τόσο για το ποσοστό της χορηγηθείσας CPU όσο και για το granularity αυτής) στις μετρικές απόδοσής τους. Επιπλέον, παρατηρείται η επίδραση της συν-εκτέλεσης στον ίδιο κόμβο VMs με διαφορετικό τύπο φορτίου και για το ίδιο ποσοστό CPU.

Όπως αναφέρεται στο [91], η μετρική, η οποία χρησιμοποιείται σήμερα από τους παρόχους υπολογιστικής υποδομής, προκειμένου να περιγράψει τις δυνατότητες του υλικού είναι κυρίως η συχνότητα της CPU. Ωστόσο αυτή τη μέτρηση δεν είναι καθόλου επαρκής, δεδομένου ότι δεν αντικατοπτρίζει την ικανότητα του φυσικού κόμβου για την

επίλυση προβλημάτων (δεν λαμβάνουμε σοβαρά υπόψη και άλλους παράγοντες όπως η ταχύτητα διαύλου, η ταχύτητα της μνήμης κλπ.). Άλλες προσεγγίσεις, όπως οι Berkeley dwarfs [73] ή τα MATLAB benchmarks είναι περισσότερο σε θέση να συλλάβουν την υπολογιστική δυνατότητα ενός κόμβου. Στην εργασία αυτή η δεύτερη προσέγγιση επιλέχθηκε. Τα MATLAB benchmarks [71] αποτελούνται από 6 υπολογιστικά τεστ, που μετρούν την ικανότητα του φυσικού κόμβου (και του Matlab) να επιλύσει προβλήματα με διαφορετικές υπολογιστικές απαιτήσεις. Συνεπώς, αυτά τα τεστ (Πίνακας 12) χρησιμοποιούνται τόσο για τον καθορισμό της υπολογιστικής ικανότητας υλικού (βαθμολογία τεστ) όσο και το χαρακτηρισμό του είδους του φόρτου εργασίας (είδος τεστ).

Πίνακας 12: Λεπτομέρειες σχετικά με τα MATLAB benchmarks και τα ειδικά μοντέλα υπολογιστικών εργασιών που εκπροσωπούν [71]

Τεστ 1	Πράξεις κινητής υποδιαστολής με τακτικές προσβάσεις μνήμης
Τεστ 2	Πράξεις κινητής υποδιαστολής με ακανόνιστο τρόπο πρόσβασης μνήμης
Τεστ 3	Δομές δεδομένων
Τεστ 4	Εργασίες κινητής υποδιαστολής και μικτών ακεραίων
Τεστ 5	2-Δ γραφικά
Τεστ 6	3-Δ γραφικά

Επιπλέον, αντιπροσωπεύουν μια ευρεία περιοχή των τύπων του φόρτου εργασίας με περιορισμένο αριθμό τεστ και αυτός είναι ο λόγος που η συγκεκριμένη προσέγγιση επιλέχθηκε. Γι 'αυτά τα τεστ, θα διερευνηθεί η επίδραση της κατανομής CPU και της συνδυασμένης τους εκτέλεσης σε ταυτόχρονα εκτελούμενα VMs στον ίδιο φυσικό πόρο. Αναμένεται ότι οι διαφορετικοί συνδυασμοί θα έχουν διαφορετική επίδραση, λόγω της φύσης των υπολογισμών. Η υποβάθμιση της απόδοσης αντικατοπτρίζεται στη

βαθμολογία του κάθε τεστ. Χαμηλότερες βαθμολογίες του τεστ δηλώνουν καλύτερες επιδόσεις. Σε γενικές γραμμές, το σκορ του τεστ είναι ο χρόνος που απαιτείται για να ολοκληρώσει τους απαραίτητους υπολογισμούς.

Επιπλέον, διερευνώνται διαφορετικά σενάρια κατανομής VMs σε πυρήνες για αρχιτεκτονικές πολλαπλών πυρήνων. Σε γενικές γραμμές, ανάλογα με την αρχιτεκτονική της κάθε CPU, οι πυρήνες μπορεί να έχουν διαφορετικά επίπεδα διαμοιρασμού κοινών πόρων όπως οι μνήμες cache, οι δίαυλοι κλπ. Οι VMs ανατίθενται με ποικίλους τρόπους στους πυρήνες του ίδιου φυσικού κόμβου με σκοπό να εξεταστεί η επίδραση παρεμβολών ανά σενάριο για τις επιδόσεις τους. Αυτή αναμένεται να μειωθεί λόγω της χρονικής απομόνωσης όταν μοιράζονται τον ίδιο πυρήνα (με εξαίρεση την επιρροή στην cache μνήμη). Ωστόσο, είναι ενδιαφέρον να δούμε τι θα συμβεί όταν τα VMs που περιέχουν τα μεμονωμένα τεστ εκτελούνται σε διαφορετικούς πυρήνες στο ίδιο φυσικό κόμβο, με αποτέλεσμα να εκτελούνται σε πραγματικό παραλληλισμό και να ανταγωνίζονται για τα ίδια υπο-κυκλώματα.

Έτσι, οι παράμετροι που διερευνώνται είναι:

• διαφορετικές αναθέσεις % πυρήνα για μία μόνο εικονική μηχανή που εκτελεί ένα στοιχειώδες τεστ

• διαφορετικές granularities για το ίδιο ποσοστό χορήγησης CPU για μία μόνο VM που εκτελεί ένα στοιχειώδες τεστ

• αμοιβαία παρεμβολή για κάθε συνδυασμό τεστ κατά την παράλληλη εκτέλεση VMs στον ίδιο πυρήνα (shared L1 και L2 cache μνήμη) και για μια ποικιλία περιόδων χρονοπρογραμματισμού P

• αμοιβαία παρεμβολή για κάθε συνδυασμό τεστ κατά την παράλληλη εκτέλεση VMs σε παρακείμενους πυρήνες (shared L2 cache μνήμη) του ίδιου φυσικού πόρου και για μια ποικιλία περιόδων χρονοπρογραμματισμού P

• αμοιβαία παρεμβολή για κάθε συνδυασμό τεστ κατά την παράλληλη εκτέλεση VMs σε μη παρακείμενους πυρήνες (μη κοινή L1/L2 cache μνήμη) του ίδιου φυσικού πόρου και για μια ποικιλία περιόδων χρονοπρογραμματισμού P

• χρόνους ping για διαφορετικές περιόδους, αναθέσεις πυρήνα% και περιόδους χρονοπρογραμματισμού P των VMs, προκειμένου να διερευνήσουμε την επίδραση στην ανταπόκριση του δικτύου.

5.4 Δοκιμές και διαδικασία μετρήσεων

Για τον χρονοπρογραμματιστή πραγματικού χρόνου, η λύση που παρουσιάζεται στο [32] επιλέχθηκε. Ο κεντρικός υπολογιστής των πειραμάτων ήταν μια τετραπλού πυρήνα (2,4 Ghz) CPU, με 8 GB μνήμης RAM και 8MB της L2 cache (χωρίζεται σε δύο μέρη, 4MB ανά ζευγάρι πυρήνα). Η διαχείριση ενέργειας είχε απενεργοποιηθεί για να μην αλλάξει δυναμικά την ταχύτητα του επεξεργαστή για εξοικονόμηση ενέργειας. Ο φυσικός κόμβος ήταν εξοπλισμένος με λειτουργικό σύστημα Ubuntu Linux 10.10, το λειτουργικό σύστημα των VM είναι τα Windows 7 (με Cygwin και OpenSSH) και το hypervisor της εικονικοποίησης είναι το KVM. Το εκτελέσιμο Matlab που περιέχει τα τεστ λαμβάνει ως ορίσματα κατά τη διάρκεια του πειράματος τον αριθμό του τεστ προς εκτέλεση, ένα αναγνωριστικό εκτέλεσης και τις παραμέτρους προγραμματισμού (Q, P) για σκοπούς τεκμηρίωσης. Η διαδικασία αυτή εκτελέστηκε σε Matlab R2007b. Προκειμένου να είναι σε θέση να τρέξει μεμονωμένα τεστ και όχι το σύνολο της σουίτας, η παραλλαγή που παρέχεται στο [92] χρησιμοποιήθηκε. Αυτό συνδυάστηκε με κατάλληλη λογική ώστε να

εκτελεί τα τεστ για ένα συγκεκριμένο χρονικό διάστημα και όχι για ένα συγκεκριμένο αριθμό εκτελέσεων. Ο λόγος είναι ότι κάθε ένα από τα 6 στοιχειώδη τεστ έχει διαφορετικό χρόνο εκτέλεσης, η οποία επηρεάζεται επίσης από τους πόρους που έχουν διατεθεί. Έτσι, ο συγχρονισμός τους δεν μπορεί να βασίζεται στον αριθμό των εκτελέσεων. Επιπλέον, για να εκτελεστεί κάθε τεστ αρκετές φορές για την συγκέντρωση αξιόπιστου στατιστικού δείγματος, αυτή η διάρκεια ήταν σημαντικά μεγαλύτερη, έτσι ώστε να συγκεντρωθούν δεδομένα για εκατοντάδες εκτελέσεις ενός τεστ ανά configuration. Από αυτό το δείγμα, η μέση τιμή των χρόνων εκτέλεσης εξήχθη για κάθε περίπτωση.

Τα πειράματα συντονίζονται από ένα module Java (Coordinator), που καθηλώθηκε σε ένα πυρήνα (CPU 0). Ο κώδικας αυτός υλοποιεί το σύστημα διασύνδεσης προς το χρονοπρογραμματιστή πραγματικού χρόνου, προκειμένου να αλλάξουν οι παράμετροι Q, P που καθορίζουν το ποσοστό και την περίοδο ανάθεσης υπολογιστικής ισχύος στις VMs. Τα VMs ήταν καθηλωμένα είτε στον έναν πυρήνα (CPU 2) και τα 2 ή κάθε ένα σε διαφορετικούς πυρήνες (CPU 1,2 ή 3), ανάλογα με το πείραμα. Μετά το configuration του scheduler, ο Coordinator συνδέεται σε κάθε εικονική μηχανή μέσω SSH και πραγματοποιεί την έναρξη των διαφορετικών συνδυασμών για τα εκτελέσιμα Matlab για ένα συγκεκριμένο χρονικό διάστημα. Ο χρόνος μετρήθηκε από το εσωτερικό του Matlab, και κάθε εκτέλεση με διαφορετική σύνθεση έγινε μέχρι να περάσει ένα ορισμένο χρονικό διάστημα (500 δευτερόλεπτα). Αυτό οδηγεί σε μια μικρή παραβίαση της χρονικής περιόδου λόγω του ότι ο χρόνος ελέγχεται μετά από εκτέλεση κάθε τεστ, αλλά ήταν πολύ μικρότερη από τη συνολική διάρκεια. Είναι κρίσιμο να αναφερθεί ότι οι λειτουργίες που χρησιμοποιούνται για τη μέτρηση του χρόνου με βάση την λειτουργία

ρολογιού του Matlab είναι κάθε άλλο παρά ακριβείς. Παρατηρήθηκε ότι με την υψηλή χρήση μέσα στο VM, οι μηχανισμοί αυτοί χάνουν εντελώς την αίσθηση του χρόνου λόγω προφανώς έλλειψης πόρων ώστε να κρατάνε τον ρυθμό μέτρησης (ενδεικτικά, ενώ η προσομοίωση έτρεχε για 2 ώρες, μέσα στο VM μόνο 5 λεπτά είχαν μετρηθεί). Γι 'αυτό η tic-toc λειτουργία μέτρησης του Matlab χρησιμοποιήθηκε, η οποία χρησιμοποιεί κατευθείαν το ρολοι του φυσικού πόρου.

Ο ψευδο-κώδικας για τον Coordinator φαίνεται στο Σχήμα 45 και για το εκτελέσιμο μέσα στο VM στο Σχήμα 46. Ο συνολικός σχεδιασμός της μετρητικής διάταξης και οι αλληλεπιδράσεις μεταξύ των αναγκαίων συστατικών στοιχείων λογισμικού εμφανίζονται στο Σχήμα 47. Αναλυτικά, τα διάφορα βήματα που χρειάζονται είναι:

• Έναρξη του Java Coordinator. Αυτό το component έχει στην ευθύνη του την παρακολούθηση των βρόχων εκτέλεσης για τις διάφορες παραμέτρους των πειραμάτων, όπως περιόδους χρονοπρογραμματισμού, τα ποσοστά CPU που έχουν ανατεθεί και τους συνδυασμούς των τεστ. Επιπλέον, εγείρει τα νήματα (threads) που επικοινωνούν με τον χρονοπρογραμματιστή πραγματικού χρόνου (μέσω ενός Python script) για τη δημιουργία της κατανομής της ισχύος της CPU σε κάθε VM.

• Το επόμενο βήμα είναι να ενεργοποιήσει διαφορετικά νήματα, ώστε να συνδεθεί με τα VMs και να ξεκινήσει τα εκτελέσιμα Matlab με τα συγκεκριμένα ορίσματα (συνδυασμούς τεστ). Αυτά τα νήματα συγχρονίζονται στο τέλος της εκτέλεσής τους, προκειμένου να εξασφαλιστεί ο συγχρονισμός και η ταυτόχρονη εκτέλεση του επόμενου συνδυασμού. Είναι κρίσιμο να αναφερθεί ότι προκειμένου να εξασφαλιστεί μια σταθερή εκτέλεση των νημάτων για απεριόριστο χρονικό διάστημα, η υλοποίηση που προτείνεται

στο [70] ακολουθήθηκε. Αυτή εξασφαλίζει ότι οι ροές εκροής και λάθους (output and error streams) συλλαμβάνονται αποτελεσματικά. Όταν ολοκληρωθούν οι δύο εκτελέσεις στο εσωτερικό των VMs, ο Coordinator προχωρά με την επόμενη διαμόρφωση που υπαγορεύεται από το βρόχο.

Τα αποτελέσματα κάθε VM κρατήθηκαν σε ένα αρχείο καταγραφής τοπικά στο VM. Στο τέλος συγχωνεύτηκαν για την απόκτηση της συνολικής πληροφορίας. Ο Πίνακας 13 περιέχει λεπτομέρειες σχετικά με την πλατφόρμα δοκιμών.

Πίνακας 13: Λεπτομέρειες μετρητικής διάταξης

Host OS	Linux Ubuntu 10.10
Guest OS	Windows 7
Hypervisor	KVM
Benchmarks	Matlab R2007b executable

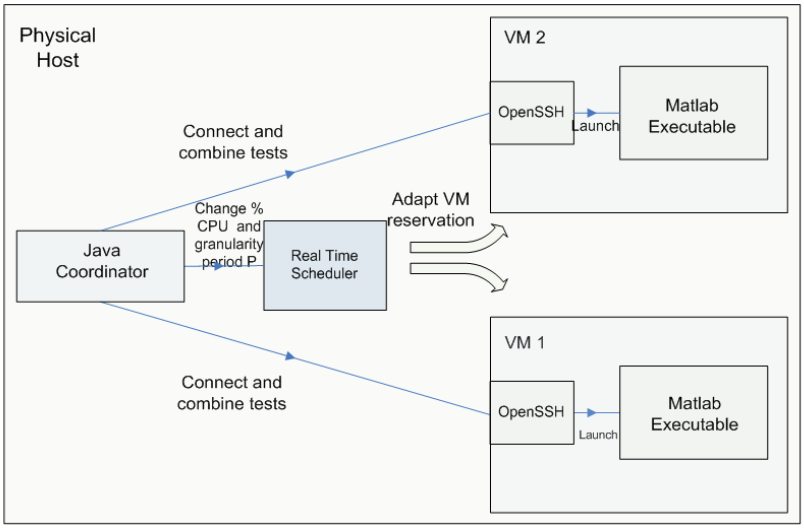
```
1. for different combinations of P
2.              for different combinations of Q/P
3.                              set scheduling params for each VM
4.                              for all unique combinations of tests
5.                                      raise threads to connect and execute
6.                                      tests inside the VMs
7.                                      synchronize threads
8.                              end
9.              end
10.end
11.gather results from VMs and merge based on execution ID
```

Σχήμα 45: Ψευδο-κώδικας για ροή εργασιών Java Coordinator

Η σελίδα αυτή είναι σκόπιμα λευκή

```
1. Take input arguments (runDuration, test number X, execution ID)
2. Measure time (tic)
3. while (duration< testDuration)
                        Run Test X (1 time)
                        Add result to array
                        Measure time (toc)
        End
4. Extract statistical data (mean, std dev) from the individual values
5. Enter log data
```

Σχήμα 46: Ψευδο-κώδικας για τη δομή του εκτελέσιμου Matlab μέσα στις εικονικές μηχανές



Σχήμα 47: Δομή λογιμσικού και διασυνδέσεις για τη διαδικασία μετρήσεων

Η διάρκεια κάθε περιόδου (για έναν συγκεκριμένο συνδυασμό τεστ με μια συγκεκριμένη διαμόρφωση υλικού) ορίστηκε σε 500 δευτερόλεπτα. Αυτός ήταν αρκετός χρόνος για να εξασφαλιστεί ότι τα νήματα θα εκτελούνται ταυτόχρονα για την πλειοψηφία του χρόνου, χωρίς οι τυχαίες καθυστερήσεις σύνδεσης ssh να επηρεάζουν αυτή τη διαδικασία. Οι καθυστερήσεις σύνδεσης ssh κυμαίνονται μεταξύ 2,8 και 3,8 δευτερολέπτων. Για το λόγο αυτό ο συγχρονισμός νημάτων έγινε στο τέλος του 500 δευτερολέπτων. Το νήμα που τερμάτιζε πρώτο περίμενε το δεύτερο για να ολοκληρωθεί.

Η σελίδα αυτή είναι σκόπιμα λευκή

Ήταν, επίσης, αρκετός χρόνος ώστε κάθε τεστ μέσα στις VMs να εκτελεστεί πολλές φορές (σε τάξη των εκατοντάδων), οδηγώντας έτσι σε επαρκή συλλογή δείγματος.

Επιπλέον, ένα αναγνωριστικό εκτέλεσης (execution ID) δινόταν ως όρισμα στο εσωτερικό των VMs, για να συμπεριληφθεί στα αποτελέσματα. Αυτό ήταν ένας μοναδικός αριθμός για κάθε συνδυασμό . Αυτό είναι απαραίτητο, λόγω του γεγονότος ότι παρατηρήθηκε ότι για μια σειρά τυχαίων λόγων (όπως broken TCP pipes κλπ) ένας περιορισμένος αριθμός των συνδυασμών δεν κατάφερε να αρχίσει ή να τελειώσει. Έτσι η μία εκτέλεση που πετύχαινε να ξεκινήσει στην ουσία εκτελούνταν χωρίς φορτίο παρεμβολής από την άλλη VM. Για τις περιπτώσεις αυτές το ID δεν καταγραφόταν στα αποτελέσματα και των 2 VM. Μέσω της σύγκρισης μεταξύ των 2 αρχείων φιλτράρονταν, οι προαναφερθείσες περιπτώσεις και επαναλαμβανόταν το πείραμα για αυτές.

5.5 Αναλυτικά αποτελέσματα

Στις επόμενες παραγράφους τα διαφορετικά σενάρια ανάθεσης VMs σε πυρήνες περιγράφονται, μαζί με το αποτέλεσμα των μετρήσεων. Για όλες τις περιπτώσεις, η περίοδος P χρονοπρογραμματιστή ερευνήθηκε από 200 έως 800 χιλιοστά του δευτερολέπτου. Μια περίοδος προθέρμανσης για το σύστημα χρησιμοποιήθηκε, όπως προτείνεται από τη βιβλιογραφία, προκειμένου τα VMs να συμπεριφέρονται με ένα πιο σταθερό τρόπο. Αυτός είναι ο λόγος για τον οποίο τα πειράματα ξεκίνησαν από 150 χιλιοστά του δευτερολέπτου, αλλά τα αποτελέσματα δεν χρησιμοποιήθηκαν για τη συγκεκριμένη τιμή. Κάθε σημείο δεδομένων στα γραφήματα που εμφανίζονται στις επόμενες ενότητες είναι η μέση τιμή των επιμέρους βαθμολογιών των τεστ για όλες τις εκτελέσεις μέσα στο διάστημα των 500 δευτερολέπτων. Σε γενικές γραμμές σε κάθε

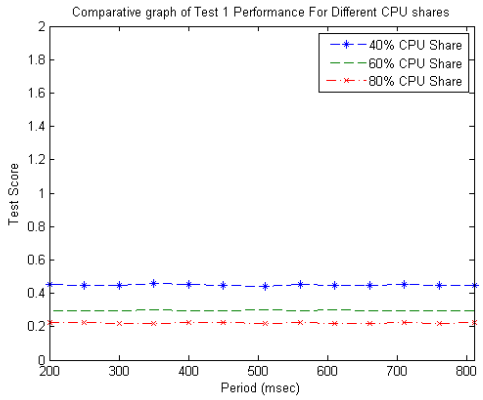
τέτοιο διάστημα, κάθε τεστ εκτελούνταν εκατοντάδες φορές μέχρι να παρέλθουν τα 500 δευτερόλεπτα.

5.5.1 Μεμονωμένη VM σε πυρήνα

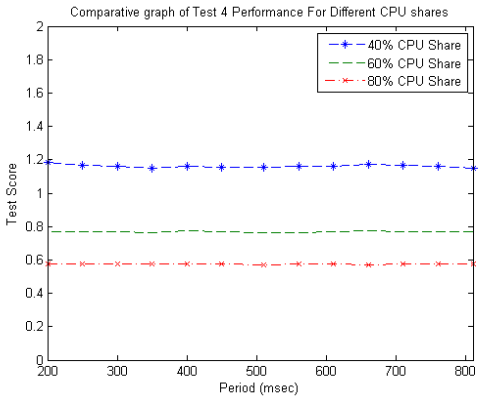
Σε αυτό το σενάριο το μέγιστο ποσοστό CPU που αποδίδεται στην μία VM που εκτελείται είναι 80% ενός πυρήνα. Αυτό οφείλεται στο γεγονός ότι το υπόλοιπο 20% παραχωρείται για τις εργασίες του κεντρικού λειτουργικού συστήματος . Σε αυτή τη διαμόρφωση, κάθε VM εκτελείται ως αυτόνομη σε ένα πυρήνα. Καμία άλλη VM δεν είναι σε λειτουργία. Αυτό έγινε για να διαπιστώσουμε την επίδραση της διασποράς των παραμέτρων χρονοπρογραμματισμού και την ποσοστιαία κατανομή CPU, καθώς και για την απόκτηση του βασικού σκορ εκτέλεσης (standalone baseline test score) για σύγκριση με τα σενάρια συνεκτέλεσης των VMs στον ίδιο φυσικό πόρο. Οι παράμετροι χρονοπρογραμματισμού είναι το X μερίδιο CPU που παραχωρείται στην VM σε μια περίοδο P. Σε κάθε περίοδο, στη VM δίνονται X * P χιλιοστά του δευτερολέπτου του χρόνου της CPU. Για παράδειγμα, αν έχουμε μια περίοδο προγραμματισμού των 500 χιλιοστών του δευτερολέπτου και το μερίδιο της CPU είναι 40%, αυτό σημαίνει ότι στη VM θα δοθούν 200 χιλιοστά του δευτερολέπτου του χρόνου της CPU κάθε 500 χιλιοστά του δευτερολέπτου.

Στις παρακάτω γραφικές παραστάσεις (Σχήμα 48) εμφανίζεται η βαθμολογία για κάθε τεστ, για διαφορετικά μερίδια της CPU και περιόδους ανάθεσης. Χαμηλότερες βαθμολογίες του τεστ δηλώνουν καλύτερες επιδόσεις. Στην ουσία, η βαθμολογία του τεστ είναι ο χρόνος που απαιτείται για την ολοκλήρωση μιας εκτέλεσής του.

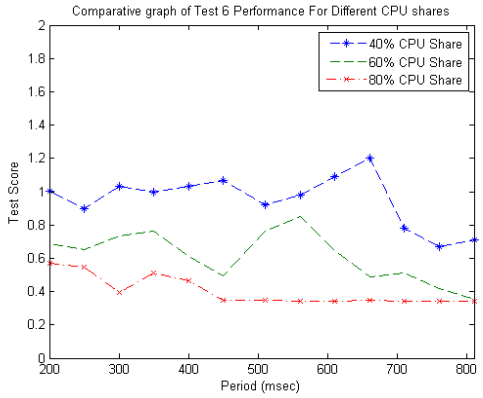
Για τις δοκιμές 1-4, το συμπέρασμα είναι ότι η συμπεριφορά είναι ανεξάρτητη από τη διακριτότητα της περιόδου του χρονοπρογραμματισμού. Αυτό είναι αναμενόμενο



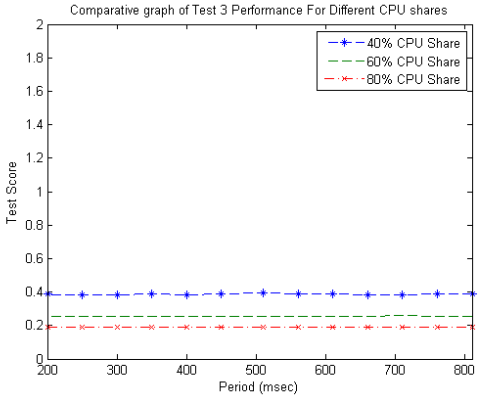
α) Αυτόνομη Εκτέλεση Τεστ 1



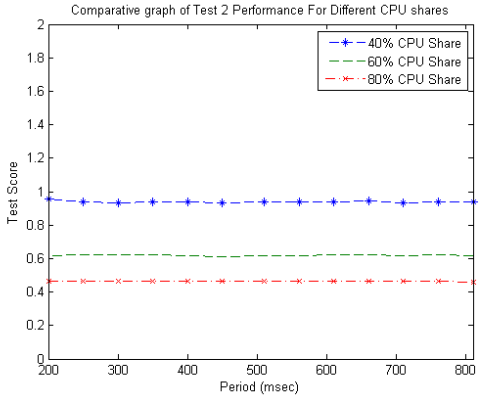
β) Αυτόνομη Εκτέλεση Τεστ 4



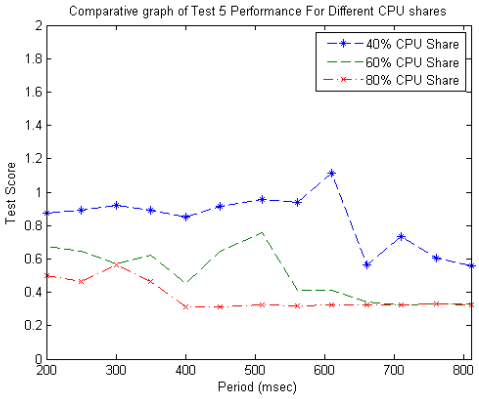
γ) Αυτόνομη Εκτέλεση Τεστ 6



δ)Αυτόνομη Εκτέλεση Τεστ 3



ε) Αυτόνομη Εκτέλεση Τεστ 2



στ) Αυτόνομη Εκτέλεση Τεστ 5

Σχήμα 48: Βαθμολογίες τεστ για διαφορετικά ποσοστά CPU και διακριτότητα (περίοδος προγραμματισμού) για εκτελέσεις χωρίς παρεμβολή (συν-προγραμματισμένες VMs).

Η σελίδα αυτή είναι σκόπιμα λευκή

αφού στην μεμονωμένη εκτέλεση, η επίδραση των παρεμβολών στη μνήμη cache από την αλλαγή εργασιών είναι ανύπαρκτη. Όσον αφορά τη βελτίωση της απόδοσης σε σχέση με τα μερίδα CPU, αυτή είναι σχεδόν γραμμική, όπως φαίνεται από την αντιστρόφως ανάλογη βαθμολογία των τεστ.

Για τις δοκιμές 5-6 (γραφικά τεστ), η συμπεριφορά είναι πιο ταλαντευόμενη, το οποίο μπορεί να αποδοθεί στη δυσκολία του στρώματος του virtualization για την πρόσβαση ή την εξομοίωση της κάρτας γραφικών.

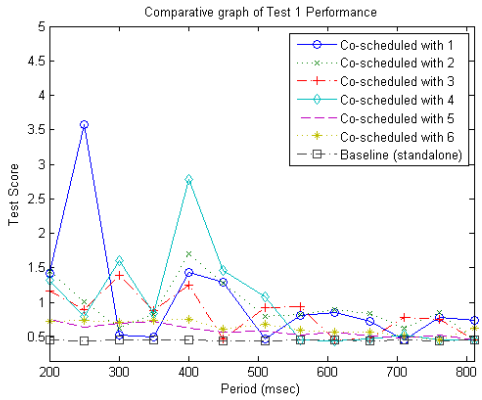
5.5.2 Συνεκτελούμενες VMs στον ίδιο πυρήνα

Το δεύτερο πείραμα στοχεύει στην εξέταση της επίδρασης της παρεμβολής μεταξύ 2 VMs που εκτελούνται ταυτόχρονα στον ίδιο πυρήνα λόγω της χρήση κοινών L1 και L2 μνημών cache. Για το λόγο αυτό, σε κάθε VM κατανέμεται το 40% της ικανότητας του πυρήνα. Οι βαθμολογίες των τεστ συγκρίνονται με τις βαθμολογίες των τεστ για το 40% CPU του πειράματος Α. Τα αποτελέσματα φαίνονται στο Σχήμα 49. Αυτά δείχνουν τα αποτελέσματα του κάθε τεστ, όταν εκτελείται στον ίδιο πυρήνα με κάθε συνδυασμό από τα άλλα τεστς. Γι 'αυτό το (i, j) τεστ δεν συμπίπτει με το (j, i). Χαμηλότερα αποτελέσματα σημαίνουν καλύτερη απόδοση.

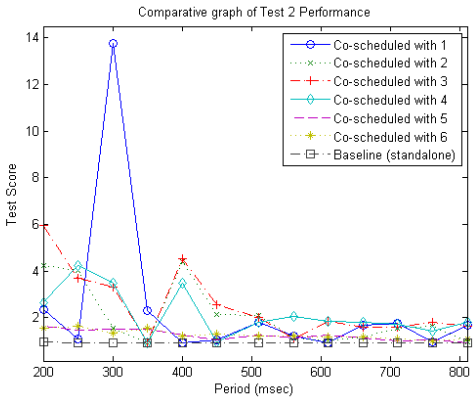
Είναι προφανές από τα διαγράμματα ότι, όσο η περίοδος χρονοπρογραμματισμού P αυξάνεται, οι γενικές επιδόσεις των συνεκτελούμενων τεστ βελτιώνονται. Αυτό οφείλεται κυρίως στο γεγονός ότι με μεγαλύτερες περιόδους (και το ίδιο% της CPU) οι εφαρμογές (VMs) έχουν περισσότερο χρόνο για να εκμεταλλευτούν στοιχεία όπως η μνήμη cache. Μία χαμηλότερη τιμή περιόδου σημαίνει πιο συχνή εναλλαγή εργασιών στη CPU και άρα χαμηλότερη χρησιμοποίηση της cache λόγω της μόλυνσης μεταξύ των εναλλασόμενων εργασιών. Επιπλέον, η επιβάρυνση από την εναλλαγή της κατάστασης

των εργασιών (επιβάρυνση scheduler) είναι αυξημένη. Ένα άλλο συμπέρασμα ήταν ότι σε κάποια σημεία τα γραφήματα φάνηκε να απεικονίζουν μερικές ανωμαλίες, όπως το Test 3 γράφημα (Σχήμα 49γ). Ωστόσο, μετά από προσεκτική παρατήρηση των αποτελεσμάτων, φάνηκε ότι όταν οι υψηλές τιμές ενός τεστ εμφανίζονταν, υπήρχαν χαμηλές τιμές για τα άλλα τεστ. Αυτό μας οδήγησε στο να σχεδιάσουμε το συνδυασμό των επιδόσεων (προστιθέμενα σκορ) και των δύο συν-εκτελούμενων VMs, ο οποίος έδειξε ένα πιο λογικό γράφημα σε επίπεδο συστήματος. Φαίνεται ότι αυτό που έχει σημασία στην περίπτωση αυτή είναι ποια εργασία ξεκινάει πρώτη. Στην περιγραφή της δοκιμαστικής διάταξης αναφέρεται ότι τα νήματα συγχρονίζονται στο τέλος της περιόδου δοκιμής, επειδή αυτό δεν μπορεί να συμβεί στην αρχή. Τα νήματα ξεκινούν ταυτόχρονα, αλλά εξαιτίας μιας σειράς τυχαίων λόγων (όπως η καθυστέρηση ssh για παράδειγμα), η εκτέλεση του κάθε τεστ στις αντίστοιχες VMs δεν ξεκινά την ίδια ακριβώς στιγμή. Η καθυστέρηση αυτή ήταν της τάξεως των 3 δευτερολέπτων, όμως ήταν πολύ χαμηλότερη από την επιλεγμένη διάρκεια εκτέλεσης (500 δευτερόλεπτα) του κάθε συνδυασμού. Από τα γραφήματα συστήματος μπορούν να εξαχθούν πιο ακριβή συμπεράσματα. Αυτά συνδυάζονται με τα άλλα σενάρια και απεικονίζονται στο Σχήμα 50.

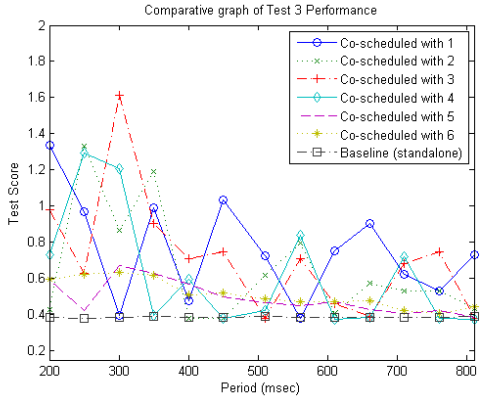
Επιπλέον, φαίνεται να υπάρχουν ιδιαίτερα υψηλές τιμές για P=250msec. Αυτό συμβαίνει για σχεδόν όλα τα γραφήματα και είναι αντίθετο προς τη γενική τάση των γραφημάτων. Λαμβάνοντας υπ 'όψιν τον τρόπο που έχουν κατασκευαστεί οι βρόχοι στο Σχήμα 45, μπορούμε να συμπεράνουμε ότι αυτό συμβαίνει λόγω κάποιας εσωτερικής παρέμβασης στο φιλοξενούμενο λειτουργικό σύστημα (π.χ. μια εργασία του λειτουργικού συστήματος μέσα στο VM που ξεκινά να εκτελείται) και επηρεάζει τις μετρήσεις που λαμβάνονται σε αυτό το χρονικό διάστημα .



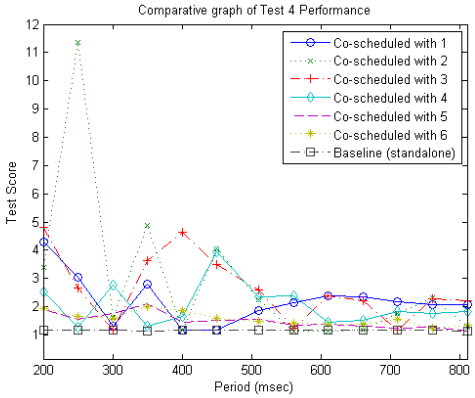
α) Απόδοση Τεστ 1



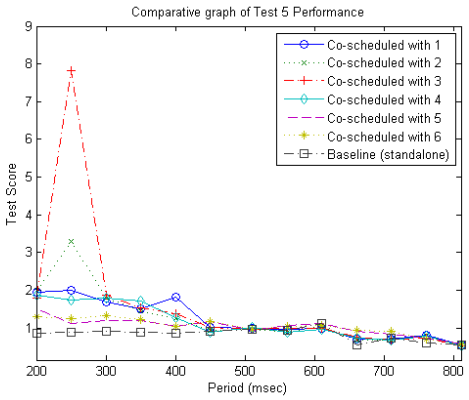
β) Απόδοση Τεστ 2



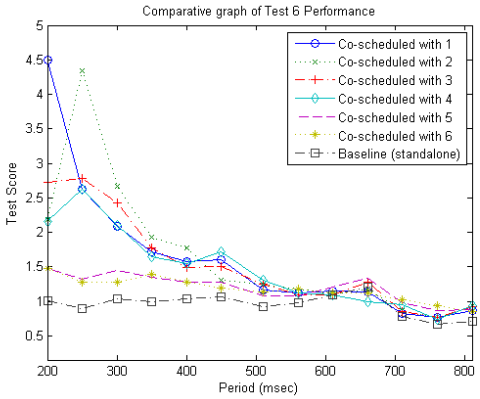
γ) Απόδοση Τεστ 3



δ) Απόδοση Τεστ 4



ε) Απόδοση Τεστ 5



στ) Απόδοση Τεστ 6

Σχήμα 49: Επίδραση στα ατομικά αποτελέσματα τεστ για διαφορετικούς συνδυασμούς, όταν κάθε ένα από τα άλλα τεστ τρέχει στην δεύτερη VM, σε σύγκριση με την απόδοσή του όταν εκτελείται ως standalone

Η σελίδα αυτή είναι σκόπιμα λευκή

5.5.3 Συνεκτελούμενες VMs σε πυρήνες με άμεση γειτνίαση

Το τρίτο πείραμα έχει ως στόχο τη διερεύνηση της επίδρασης στην απόδοση των VMs όταν εκτελούνται σε διαφορετικούς αλλά σε άμεση γειτνίαση πυρήνες στην ίδια CPU. Αυτό γίνεται για να προσδιορίσουμε το αποτέλεσμα της παρεμβολής μνήμης L1 cache, σε σύγκριση με τα αποτελέσματα του προηγούμενου πειράματος. Με αυτή τη ρύθμιση, οι πυρήνες έχουν διαφορετικές μνήμες cache L1, αλλά κοινές L2. Το επιλεγμένο ποσοστό χρησιμοποίησης για αυτό το πείραμα ήταν 40% ανά πυρήνα. Ο λόγος για την επιλογή του ποσοστού αυτού είναι για να έχει συγκρίσιμα αποτελέσματα με το προηγούμενο πείραμα, για το οποίο το ανώτατο όριο ανάθεσης για τις δύο εργασίες (VMs) σε έναν πυρήνα είναι 40% ανά εργασία (μετά την αφαίρεση περίπου του 20% που είναι πάντα διαθέσιμη για τα καθήκοντα του συστήματος). Σύγκριση με τα άλλα σενάρια περιλαμβάνεται στο Σχήμα 50.

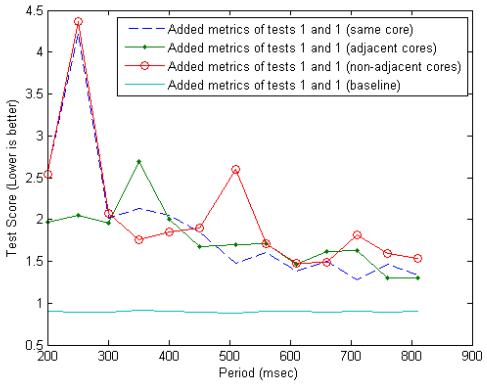
5.5.4 Συνεκτελούμενες VMs σε πυρήνες με έμμεση γειτνίαση

Η τέταρτη ρύθμιση παραμέτρων αποφασίστηκε να εναποθέτει τις VMs σε μη παρακείμενους πυρήνες. Ο λόγος για αυτή τη ρύθμιση είναι ότι ο φυσικός κόμβος των πειραμάτων έχει 4 πυρήνες, οι οποίοι μοιράζονται μια μνήμη L2 cache ανά ζεύγη. Έτσι, η ανάθεση VMs σε μη παρακείμενους πυρήνες οδηγεί σε εκτελέσεις οι οποίες δεν έχουν κανένα κοινό επίπεδο cache, σε αντίθεση με τα προηγούμενα πειράματα. Για όλα τα σενάρια, οι επιδόσεις των VMs ανά συνδυασμό τεστ απεικονίζεται στο Σχήμα 50. Στο σχήμα αυτό, έχουμε προσθέσει τις βαθμολογίες των δύο VMs και απεικονίζονται όλα τα σενάρια ανάθεσης για σκοπούς σύγκρισης.

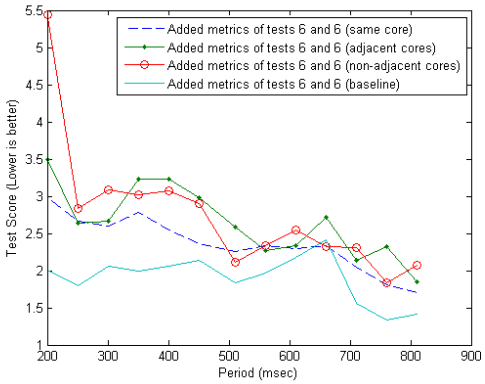
Η πρόβλεψη κατά την έναρξη αυτών των πειραμάτων ήταν ότι η παρεμβολή μεταξύ των ταυτόχρονα εκτελούμενων VMs θα επηρέαζε σημαντικά τις επιδόσεις τους.

Αυτό επικυρώθηκε από όλες τις ελεγχόμενες περιπτώσεις. Οι βαθμολογίες αναφοράς (Σχήμα 48) είναι πολύ καλύτερες από οποιαδήποτε άλλη ρύθμιση, παρόλο που το ποσοστό του πυρήνα που είχε ανατεθεί στα VMs ήταν το ίδιο. Επιπλέον, διαφορετικά τεστ δημιουργήσουν διαφορετικούς βαθμούς παρεμβολών, με αποτέλεσμα την επικύρωση της αρχικής παραδοχής ότι χρειάζεται προσεκτική ανάλυση για τη βελτιστοποίηση της ομαδοποίησης VMs στους διαθέσιμους κόμβους. Υψηλότερες περιόδοι για τον real-time scheduler φαίνεται επίσης ότι ωφελούν την απόδοση, λόγω της καλύτερης αξιοποίησης της cache εκ μέρους των εργασιών.

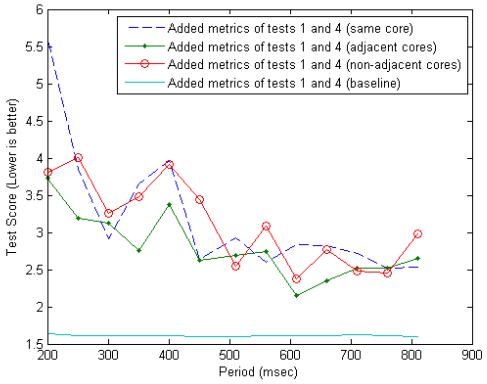
Ωστόσο, ήταν επίσης αναμενόμενο ότι η μετάβαση από στενά συνδεδεμένες ρυθμίσεις (ίδιο πυρήνα) σε πιο χαλαρά συνδεδεμένες αυτές (παρακείμενοι και μη παρακείμενοι πυρήνες) θα ωφελούσε την απόδοση των VMs. Μετά από παρατήρηση των γραφημάτων, μπορούμε να συμπεράνουμε ότι αυτό δεν συμβαίνει. Δεδομένου ότι για τις χαλαρά συνδεδεμένες περιπτώσεις έχουμε ανεξαρτησία στο επίπεδο της L1 cache (για τους παρακείμενους πυρήνες) και L1/L2 ανεξαρτησία cache (για τους μη παρακείμενους πυρήνες) μπορεί να υποτεθεί ότι η συμφόρηση στην οποία οφείλεται η έλλειψη βελτίωσης της απόδοσης έχει να κάνει με τη πρόσβαση στη μνήμη RAM. Ο συντελεστής της επιρροής σε αυτή την περίπτωση μπορεί να είναι ο δίαυλος μνήμης (FSB) που είναι απαραίτητος για την πρόσβαση στην RAM. Στο φυσικό κόμβο που χρησιμοποιήθηκε, αυτό ήταν ένα 1066MHz bus. Ωστόσο, αυτό το εύρος ζώνης κατανέμεται μεταξύ των διαφορετικών πυρήνων που προσπαθούν να αποκτήσουν ταυτόχρονα πρόσβαση στη κύρια μνήμη. Εάν περισσότεροι από ένας πυρήνες προσπαθούν να αποκτήσουν πρόσβαση την ίδια στιγμή, το διαθέσιμο εύρος ζώνης θα μοιραστεί μεταξύ τους.



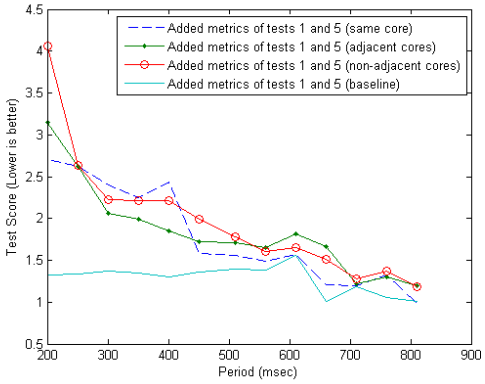
α) Συνδυασμένη Απόδοση Τεστ 1 και 1



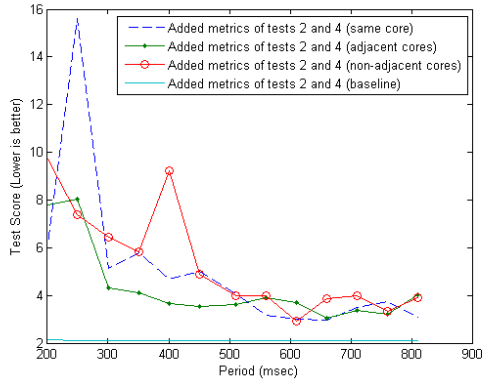
β) Συνδυασμένη Απόδοση Τεστ 6 και 6



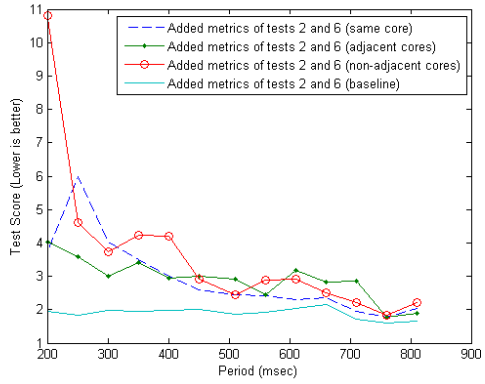
γ) Συνδυασμένη Απόδοση Τεστ 1 και 4



δ) Συνδυασμένη Απόδοση Τεστ 1 και 5

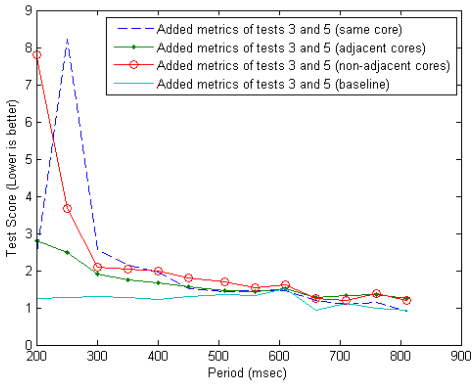


ε) Συνδυασμένη Απόδοση Τεστ 2 και 4

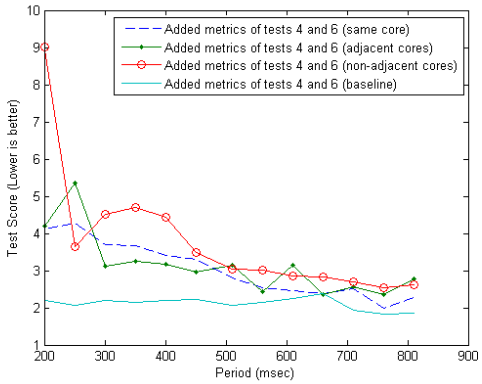


στ) Συνδυασμένη Απόδοση Τεστ 2 και 6

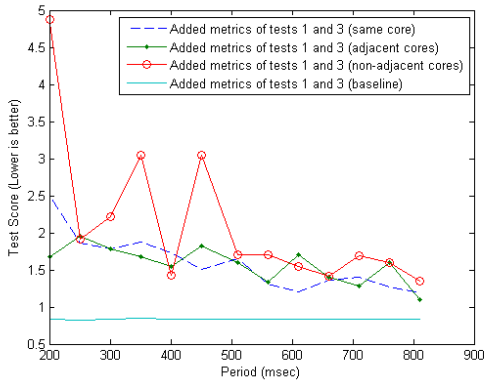
Η σελίδα αυτή είναι σκόπιμα λευκή



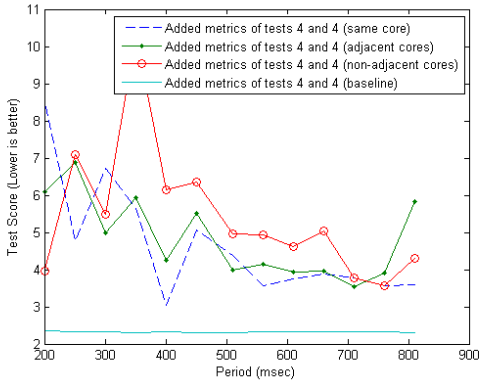
ζ) Συνδυασμένη Απόδοση Τεστ 3 και 5



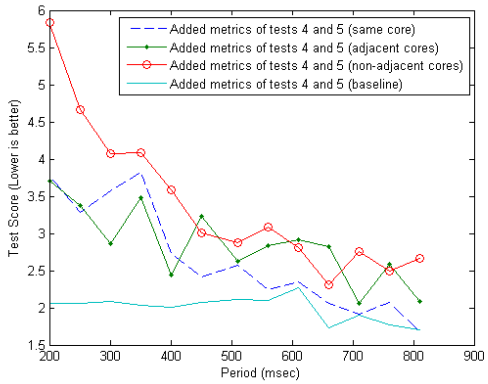
η) Συνδυασμένη Απόδοση Τεστ 4 και 6



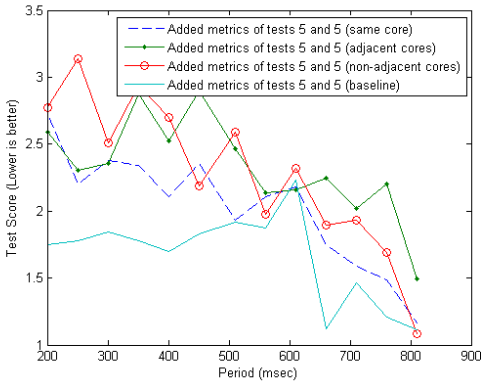
θ) Συνδυασμένη Απόδοση Τεστ 1 και 3



ι) Συνδυασμένη Απόδοση Τεστ 4 και 4

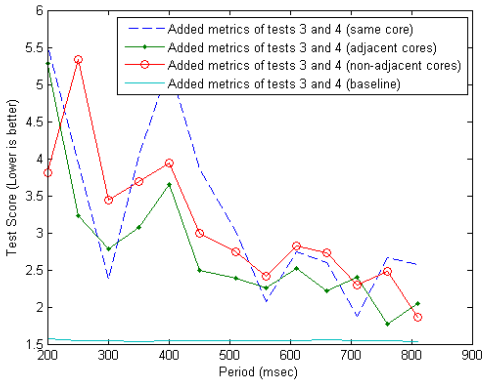


κ) Συνδυασμένη Απόδοση Τεστ 4 και 5

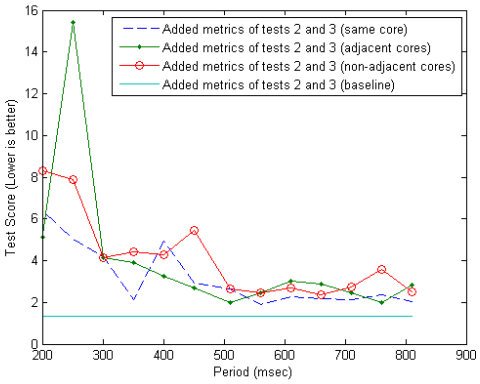


κα) Συνδυασμένη Απόδοση Τεστ 5 και 5

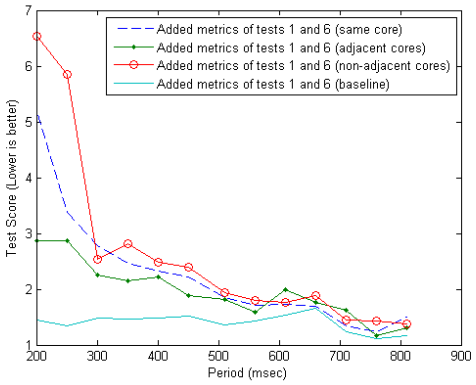
Η σελίδα αυτή είναι σκόπιμα λευκή



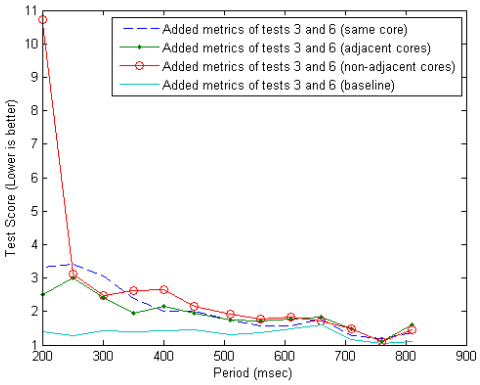
κβ) Συνδυασμένη Απόδοση Τεστ 3 και 4



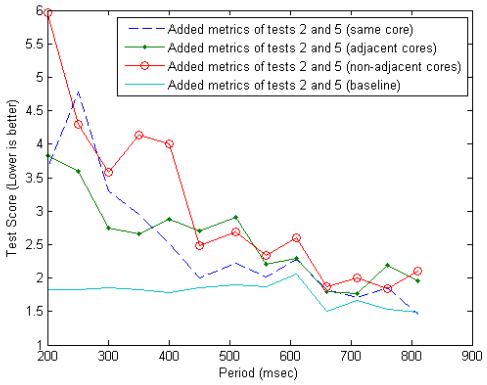
κγ) Συνδυασμένη Απόδοση Τεστ 2 και 3



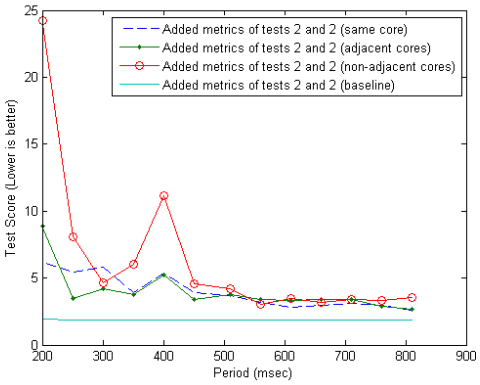
κδ) Συνδυασμένη Απόδοση Τεστ 1 και 6



κε) Συνδυασμένη Απόδοση Τεστ 3 και 6

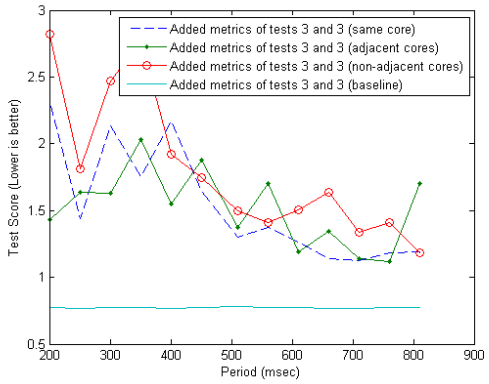


κστ) Συνδυασμένη Απόδοση Τεστ 2 και 5

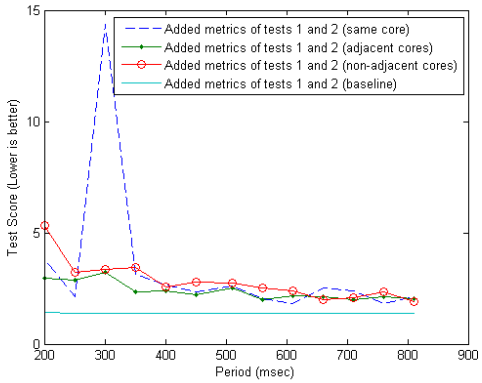


κζ) Συνδυασμένη Απόδοση Τεστ 2 και 2

Η σελίδα αυτή είναι σκόπιμα λευκή



κη) Συνδυασμένη Απόδοση Τεστ 3 και 3



κθ) Συνδυασμένη Απόδοση Τεστ 1 και 2

Σχήμα 50: Σύγκριση των βαθμολογιών επιπέδου συστήματος (άθροισμα των σκορ και από τις 2 VMs) για όλα τα σενάρια ανάθεσης και για το 40% της ισχύος πυρήνα ανά VM.

Έτσι, κατά την κατανομή των VMs σε διαφορετικούς πυρήνες (με το ίδιο μερίδιο 40%), το διαιρεμένο εύρος ζώνης είναι ένας περιοριστικός παράγοντας σε σύγκριση με το σενάριο εκτέλεσης στον ίδιο πυρήνα. Για τον ίδιο πυρήνα, οι VMs εκτελούνται η μία μετά την άλλη, σε μια διαδοχική, ψευδο-παράλληλη μορφή. Κατά τη διάρκεια αυτής της περιόδου, κάθε VM έχει όλο το διαθέσιμο εύρος ζώνης του FSB. Όταν εκτελούνται σε διαφορετικούς πυρήνες, για το χρονικό διάστημα κατά το οποίο επικαλύπτονται οι περιόδοι ενεργοποίησης των VMs, το εύρος ζώνης τους σχετικά με το FSB θα διαιρεθεί. Δεδομένου ότι ο αριθμός και ο χρόνος των προσβάσεων RAM μπορεί να είναι κυμαινόμενοι, αυτό μπορεί να εξηγήσει γιατί σε ορισμένες περιπτώσεις μία μορφή ανάθεσης είναι καλύτερη από την άλλη, ανάλογα με το φαινόμενο που επικρατεί (οι παρεμβολές cache ή το εύρος ζώνης του κοινού FSB).

Για την καλύτερη απεικόνιση της υποβάθμισης της απόδοσης ανάλογα με τους συνδυασμούς των τεστ που συνεκτελούνται, το % ποσοστό υποβάθμισης ανά συνδυασμό τεστ απεικονίζεται στο Σχήμα 51, για P = 800 msec. Επιλέξαμε τη συγκεκριμένη περίοδο διότι αποδείχθηκε από τις προηγούμενες παραγράφους ότι για αυτήν την τιμή της P το

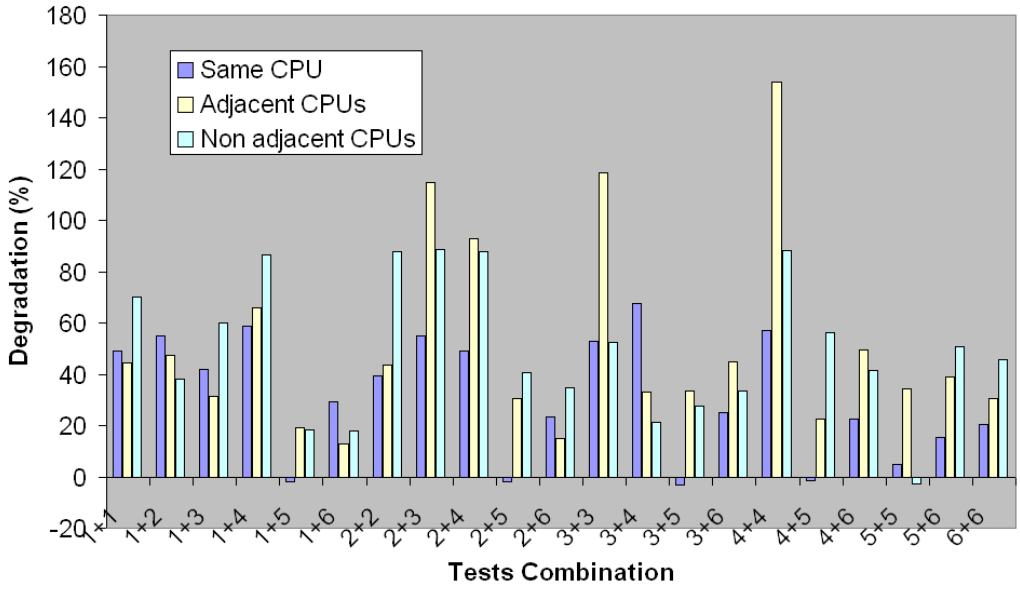
Η σελίδα αυτή είναι σκόπιμα λευκή

σύστημα είχε τη μικρότερη επιβάρυνση. Η υποβάθμιση της απόδοσης είναι η σχέση μεταξύ των προστιθέμενων αποτελεσμάτων των τεστ στις ταυτόχρονες εκτελέσεις (για τα διάφορα σενάρια ανάθεσης)και των προστιθέμενων βαθμολογιών από τις αυτόνομες (standalone) εκτελέσεις και προέρχεται από την Εξίσωση 13.

$$\%Degradation = 100 * \frac{(T_{A\_I} + T_{B\_I}) - (T_{A\_BASE} + T_{B\_BASE})}{T_{A\_BASE} + T_{B\_BASE}}$$

Εξίσωση 13: Ορισμός ποσοστιαίας υποβάθμισης απόδοσης

Στον τύπο αυτό, A και B είναι οι αριθμοί τεστ ανά συνδυασμό, I είναι το σενάριο ανάπτυξης (ίδιο πυρήνα, παρακείμενοι πυρήνες, μη παρακείμενοι πυρήνες) και BASE είναι το σκορ όταν τα τεστ εκτελούνται ως standalone, χωρίς οποιοδήποτε άλλο τεστ να εκτελείται στο φυσικό κόμβο.



Σχήμα 51: Ποσοστό της επιβάρυνσης της βαθμολογίας για όλους τους συνδυασμούς τεστ και τα σενάρια ανάθεσης και P = 800msec, προκειμένου να καταδειχθεί το εύρος της υποβάθμισης και ο προσδιορισμός των βέλτιστων συνδυασμών.

Ο στόχος αυτού του σχήματος είναι να δείξει το συγκεκριμένο ποσοστό επιβάρυνσης της απόδοσης των τεστ για διαφορετικά σενάρια και για διαφορετικούς

Η σελίδα αυτή είναι σκόπιμα λευκή

συνδυασμούς τεστ και να εντοπίσει τυχόν μοτίβα που μπορεί να είναι επωφελή για την εξαγωγή γενικών συμπερασμάτων. Από αυτό φαίνεται ότι η υπολογιστική επιβάρυνση μιας VM μπορεί να κυμαίνεται από σχεδόν 0 και μέχρι 160%, ανάλογα με τα συνεκτελούμενα τεστ και το σενάριο ανάπτυξης.

Από αυτό το γράφημα είναι επίσης προφανές ότι ο καλύτερος συνδυασμός είναι για όλα τα τεστ με το τεστ 5 και το δεύτερο καταλληλότερο είναι το τεστ 6. Ο λόγος για τον οποίο αυτές οι εφαρμογές γραφικών είναι καταλληλότερες μπορεί να αποδοθεί στο γεγονός ότι χρησιμοποιούν μια μικρή ποσότητα δεδομένων, για την οποία πραγματοποιούν μεγάλο αριθμό υπολογισμών. Χειρότερος υποψήφιος για την από κοινού εκτέλεση είναι το τεστ 4, ενώ η τοποθέτηση σε παρακείμενους πυρήνες φαίνεται να παράγει τη μέγιστη επιβάρυνση (συνδυασμός της επίδρασης στην L2 cache και στον FSB δίαυλο). Αυτό είναι απόδειξη της σημασίας της έρευνας των παρεμβολών που προκαλούνται μεταξύ συν-προγραμματισμένων VMs. Η ταυτόχρονη εκτέλεσή τους στον ίδιο φυσικό πόρο μπορεί να οδηγήσει σε σημαντική υποβάθμιση των χαρακτηριστικών QoS των εφαρμογών που λειτουργούν στις virtualized υποδομές. Επιπλέον, στις περισσότερες περιπτώσεις η χρήση παρακείμενων πυρήνων έχει τη χειρότερη επίδοση, λόγω παρεμβολών μνήμης L2 cache και τη διαίρεση του εύρους ζώνης διαύλου μνήμης RAM. Η παρέμβαση L1 cache φαίνεται να επηρεάζει μόνο έναν περιορισμένο αριθμό των συνδυασμών (όπως το 1 και 2).

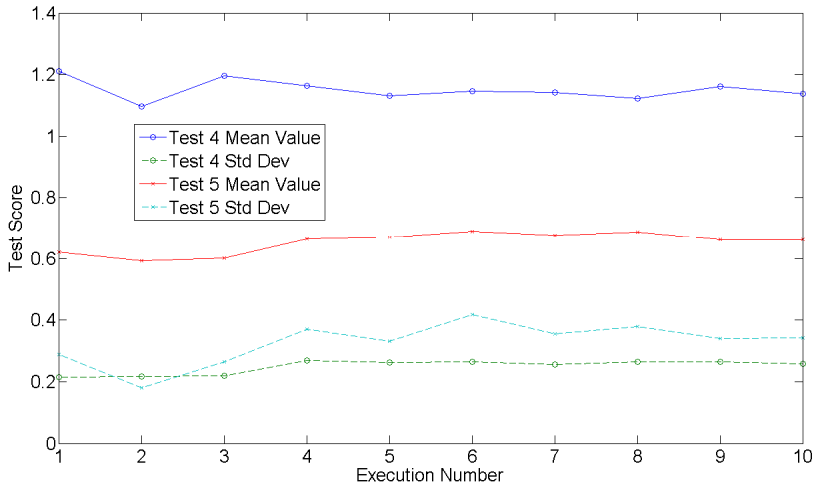
5.5.5 Στατιστική Ανάλυση των μετρήσεων

Προκειμένου να συγκεντρωθεί το σύνολο δεδομένων που παρουσιάστηκε στα προηγούμενα διαγράμματα, κάθε διαμόρφωση (συγκεκριμένη περίοδος χρονοπρογραμματισμού P, συγκεκριμένος συνδυασμός τεστ και σενάριο εγκατάστασης)

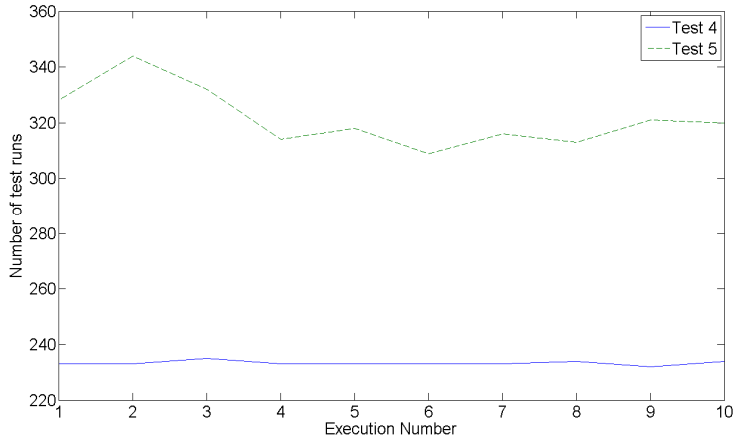
αφέθηκε να εκτελεστεί για 500 δευτερόλεπτα. Αυτό σημαίνει ότι κάθε σημείο στα γραφήματα (σύμφωνα με το setup) παρήχθη από τη μέση τιμή της βαθμολογίας του τεστ σε όλες τις εκτελέσεις του μέσα στο διάστημα των 500 δευτερολέπτων. Ο αριθμός των εκτελέσεων αυτών κυμαινόταν, λόγω του γεγονότος ότι κάθε δοκιμή είχε διαφορετικές υπολογιστικές ανάγκες, καθώς και επιβάρυνση από την ταυτόχρονη λειτουργία της άλλης VM. Ωστόσο, ήταν της τάξης των εκατοντάδων τιμών για κάθε σημείο δεδομένων Αυτό είναι ενδεικτικό και των αποτελεσμάτων των τεστ, που δείχνουν το χρόνο που χρειάζεται για να εκτελεστεί μία φορά.

Προκειμένου να επικυρωθούν περαιτέρω τα δεδομένα που παρήχθησαν, επαναλάβαμε ένα μέρος των πειραμάτων για 10 φορές. Αυτό σημαίνει ότι οι εκτελέσεις με μια συγκεκριμένη διαμόρφωση (συνδυασμοί δοκιμής, Q/P αναλογία, περίοδος χρονοπρογραμματισμού P) εκτελέσθηκαν 10 φορές προκειμένου να ερευνηθεί η διαφορά μεταξύ των τιμών των μη-διαδοχικών εκτελέσεων με τις ίδιες συνθήκες. Κατά τη διάρκεια αυτού του πειράματος, κάθε διαμόρφωση (διάστημα 500 δευτερολέπτων) εκτελέστηκε για 10 φορές κατά τρόπο μη-διαδοχικό. Τα αποτελέσματα (μέσες τιμές και τυπικές αποκλίσεις των βαθμολογιών των τεστ σε κάθε ένα από τα δέκα διαστήματα των 500 δευτερολέπτων) απεικονίζονται στο Σχήμα 52. Από αυτό μπορεί να παρατηρηθεί ότι η διαφορά μεταξύ των διαφορετικών εκτελέσεων είναι πολύ μικρή.

Επιπλέον, αποθηκεύτηκε ο αριθμός εκτελέσεων για κάθε τεστ σε κάθε μια από τις 10 επαναλήψεις και παρουσιάζεται στο Σχήμα 53. Από αυτό μπορεί να παρατηρηθεί ότι για κάθε επανάληψη του πειράματος (από την οποία η μέση τιμή και η τυπική απόκλιση που απεικονίζονται στο Σχήμα 52 εξήχθησαν) περίπου 320 εκτελέσεις πραγματοποιήθηκαν για το τεστ 4 και περίπου 230 για το τεστ 5. Η διαφορά στον αριθμό



Σχήμα 52: Διαφοροποίηση των μέσων χρόνων και των τυπικών αποκλίσεων των βαθμολογιών των τεστ για 10 διαφορετικές μη-διαδοχικές εκτελέσεις της ίδιας διαμόρφωσης (συνδυασμός τεστ 4 και 5, period P=700 msec, εκτέλεση στον ίδιο πυρήνα).

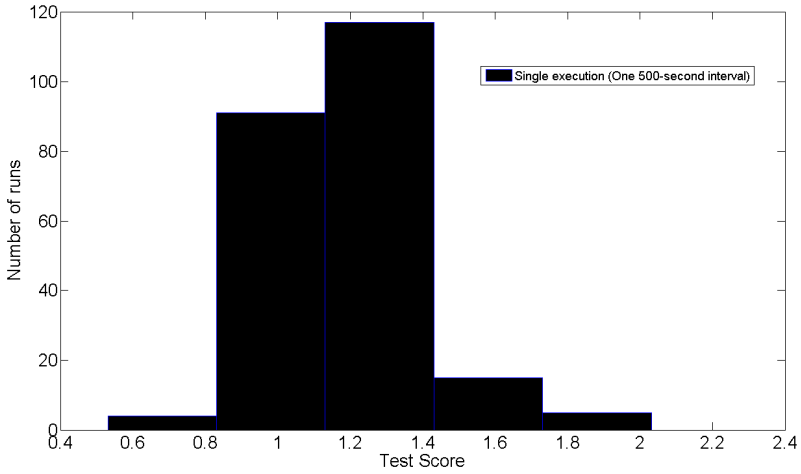


Σχήμα 53: Αριθμός εκτελέσεων κάθε τεστ σε κάθε μία από τις επαναλήψεις των 500 δευτερολέπτων και για την ίδια διαμόρφωση (συνδυασμός τεστ 4 και 5, period P=700 msec, εκτέλεση στον ίδιο πυρήνα)

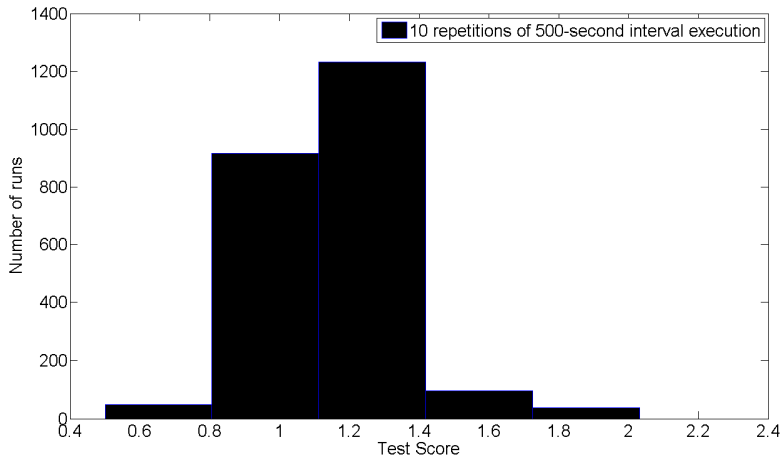
εκτελέσεων μεταξύ των δύο τεστ οφείλεται στο γεγονός ότι κάθε τεστ έχει διαφορετικούς τύπους υπολογισμών, κατά συνέπεια και διαφορετικό χρόνο εκτέλεσης. Επιλέχτηκε να συγχρονιστούν οι εκτελέσεις με βάση ένα δεδομένο χρονικό διάστημα (και όχι αριθμό εκτελέσεων) προκειμένου να εξασφαλιστεί η ταυτόχρονη εκτέλεση των υπολογισμών. Αν βασιζόμασταν στον ίδιο αριθμό εκτελέσεων θα είχαμε το πρόβλημα ότι το ένα τεστ

Η σελίδα αυτή είναι σκόπιμα λευκή

θα τελείωνε γρηγορότερα από το άλλο. Κατά συνέπεια οι υπόλοιπες εκτελέσεις του πιο αργού τεστ θα εκτελούνταν όπως στην αυτόνομη (standalone) φάση. Από τις ανωτέρω γραφικές παραστάσεις μπορούμε να καταλήξουμε στο συμπέρασμα ότι εξαιτίας του γεγονότος ότι οι εκτελέσεις κάθε τεστ σε κάθε επανάληψη είναι αρκετά υψηλές, η επανάληψη των πειραμάτων θα παράγει παρόμοιες τιμές.



(a)



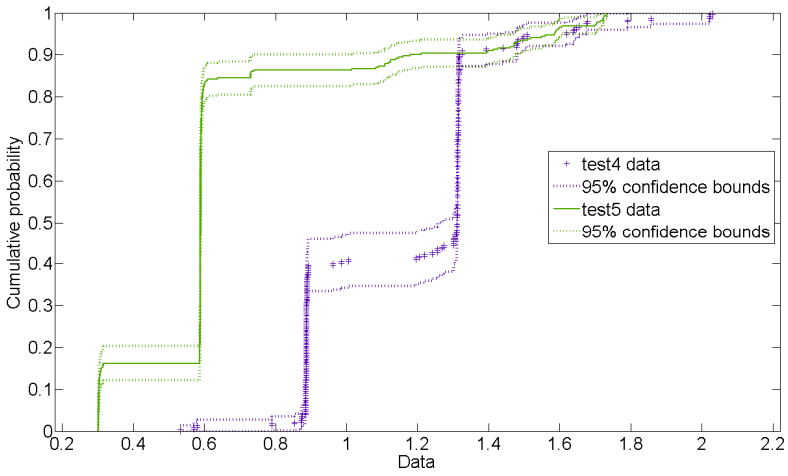
(b)

Σχήμα 54: Ιστόγραμμα της κατανομής των μεμονωμένων χρόνων κάθε εκτέλεσης τεστ για το α) 1 διάστημα 500 δευτερολέπτων β) 10 επαναλήψεις του διαστήματος 500 δευτερολέπτων για την ίδια διαμόρφωση (συνδυασμός τεστ 4 και 5, period P=700 msec, εκτέλεση στον ίδιο πυρήνα)

Η σελίδα αυτή είναι σκόπιμα λευκή

Η κατανομή των μεμονωμένων τιμών των αποτελεσμάτων (βαθμολογιών τεστ) για το τεστ 4 εμφανίζεται στο Σχήμα 54, για μια από τις 10 επαναλήψεις (Σχήμα 54a) και για τις 10 επαναλήψεις (Σχήμα 54b). Από αυτήν την σύγκριση μπορεί να εξαχθεί το συμπέρασμα ότι οι δύο κατανομές είναι σχεδόν όμοιες, παρουσιάζοντας τις αιχμές στα ίδια διαστήματα τιμών (0.8-1.2 και 1.2-1.4).

Όσον αφορά τη στατιστική ανάλυση των δειγμάτων που συλλέγονται μέσα σε μια εκτέλεση, τα διαστήματα εμπιστοσύνης απεικονίζονται στο Σχήμα 55 για τα δύο τεστ (4 και 5) του επιλεγμένου τρεξίματος. Για αυτή την περίπτωση (διάστημα 500 δευτερολέπτων), το τεστ 4 εκτελέσθηκε 232 φορές ενώ το τεστ 5 321 φορές. Προκειμένου να υποβληθούν σε επεξεργασία τα αποτελέσματα, το dfittool του Matlab χρησιμοποιήθηκε.



Σχήμα 55: Συνάρτηση συσσωρευτικής κατανομής και διαστήματα εμπιστοσύνης για τα δείγματα βαθμολογίας των τεστ 4 και 5 για scheduling period=700 msec, εκτέλεση στον ίδιο πυρήνα.

Από αυτό μπορούμε να παρατηρήσουμε ότι, ειδικά για το τεστ 5 υπάρχουν δύο σημαντικοί τομείς της συγκέντρωσης των βαθμολογιών του τεστ, γύρω από τις 0.9 και 1.3 τιμές. Αυτό μπορεί να αποδοθεί στις ανενεργές περιόδους που παρεμβάλλονται από

Η σελίδα αυτή είναι σκόπιμα λευκή

το χρονοπρογραμματιστή ανάμεσα στις εργασίες που εκτελούνται. Αυτή η διαμόρφωση είχε ένα μερίδιο 40% για μία περίοδο 700 msec. Αυτό σημαίνει ότι για 60% του χρόνου (420 msec) η υπολογιστική εργασία απενεργοποιείται. Κατά συνέπεια εάν το τεστ κατορθώνει να τελειώσει πριν από την ανενεργό περίοδο δεν θα έχει την πρόσθετη καθυστέρηση των 420 msec . Εάν όχι, θα τελειώσει μετά από το διάστημα απενεργοποίησης. Αυτό συμφωνεί επίσης με τη διαφορά μεταξύ των δύο σημαντικών τιμών της κατανομής.

5.5.6 Απόκριση δικτύου

Κατά τη διάρκεια των αρχικών δοκιμών για τα πειράματα, παρατηρήθηκε ότι για τις χαμηλές τιμές της περιόδου P (κάτω από 100msec) η απόκριση των VM ήταν εξαιρετικά αργή, ακόμη και για μια απλή σύνδεση ssh ή ένα ping. Για αυτόν τον λόγο αποφασίστηκε να πραγματοποιηθεί ένα ακόμη πείραμα προκειμένου να διερευνηθεί η επιδείνωση των χρόνων απόκρισης δικτύου των VM.

Αυτή η έρευνα περιέλαβε την αποστολή δοκιμαστικών πακέτων προς το VM (pinging) για μια ορισμένη χρονική περίοδο (300 δευτερόλεπτα) για διαφορετικές διαμορφώσεις. Επίσης χρησιμοποιήθηκαν διαφορετικές περίοδοι pinging. Ο αριθμός δειγμάτων μπορεί να προέλθει από τη διαίρεση της περιόδου των 300 δευτερολέπτων με τη περίοδο pinging. Διαφορετικά ποσοστά αναθέσεων CPU λήφθησαν επίσης υπό εξέταση και διαφορετικές περίοδοι P για τον scheduler. Ο Πίνακας 14 περιέχει τις λεπτομερείς τιμές.

Προκειμένου να οδηγηθεί η VM στο πλήρες φορτίο, δηλαδή το utilization της να είναι στο μέγιστο που της ανατέθηκε, δύο προσεγγίσεις υλοποιήθηκαν. Η πρώτη ήταν μέσω των συνεχών εκτελέσεων των MATLAB τεστ. Προκειμένου να αποφευχθεί η

επίδραση κάποιας συγκεκριμένης παρέμβασης υπολογισμού που θα μπορούσε να προκληθεί από αυτά τα τεστ (όπως το αυξανόμενο I/O που θα μπορούσε να αυξήσει τις εξυπηρετούμενες διακοπές υλικού και να έχει επιπτώσεις στις απαντήσεις στα δοκιμαστικά πακέτα), ένα δεύτερο εκτελέσιμο πρόγραμμα δημιουργήθηκε. Αυτό αποτελούνταν από έναν άπειρο βρόχο που υπολόγιζε την τετραγωνική αξία του δείκτη. Τα αποτελέσματα ήταν σχεδόν ίδια. Στις ακόλουθες γραφικές παραστάσεις (Σχήμα 56), η μέση τιμή και η τυπική απόκλιση των χρόνων απόκρισης απεικονίζονται, για το πιό επίπονο πείραμα (pinging δευτερόλεπτα period=0.5).

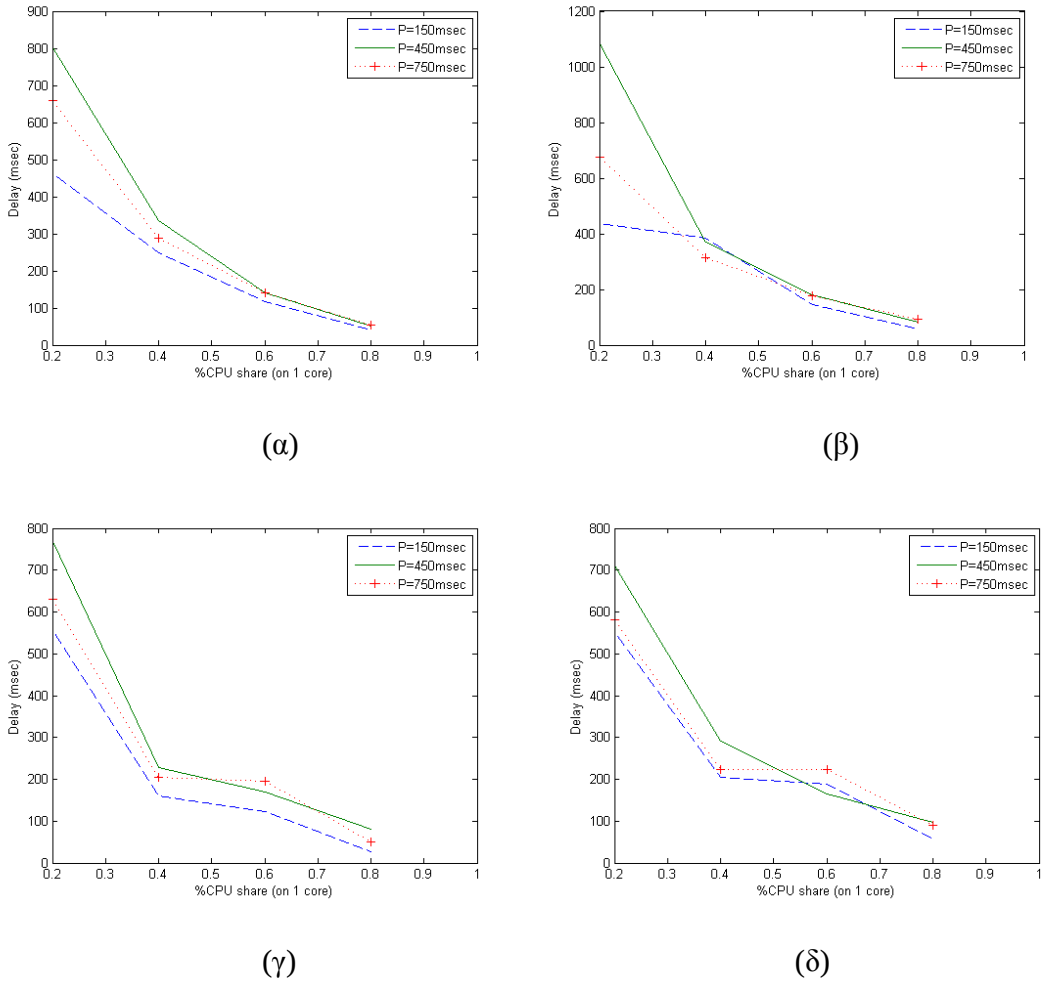
Πίνακας 14: Παράμετροι διαμόρφωσης για το πείραμα δικτυακής απόκρισης

Περίοδος ping	0.5,1, 1.5 και 2 δευτερόλεπτα
Περίοδος χρονοπρογραμματισμού P	150-800 msec (βήμα 50 msec)
Κατάσταση VM	Πλήρες φορτίο
Q/P (ποσοστό ανάθεσης CPU στη VM)	20, 40, 60 and 80%

Είναι επιτακτικό να τονιστεί ότι η δοκιμή άρχισε από P=50msec της περιόδου χρονοπρογραμματισμού, εντούτοις τα VM παρουσίασαν ελάχιστη ανταπόκριση, με πολύ μεγάλο ποσοστό πακέτων να εμφανίζουν time out. Η πλήρης απόκριση παρουσιάστηκε από τις τιμές των 150msec. Το συμπέρασμα που προκύπτει από αυτές τις μετρήσεις είναι ότι οι χαμηλές τιμές του P είναι ευεργετικότερες για τις μετρικές δικτύων (και για τη μέση τιμή και την τυπική απόκλιση).

Επιπλέον, μια θεωρητική προσδοκία ήταν ότι οι χρόνοι απόκρισης θα επιδεινώνονταν καθώς οι τιμές P αυξάνονται, εξαιτίας του γεγονότος ότι για τα χαμηλά ποσοστά των αναθέσεων CPU, οι μεγάλες ανενεργές περίοδοι θα είχαν επιπτώσεις ειδικά στους χρόνους απόκλισης. Εντούτοις, όπως βλέπουμε από τις γραφικές παραστάσεις, οι υψηλότερες τιμές και για τις δύο μετρικές είναι για τις μέσες περιπτώσεις του P (για την

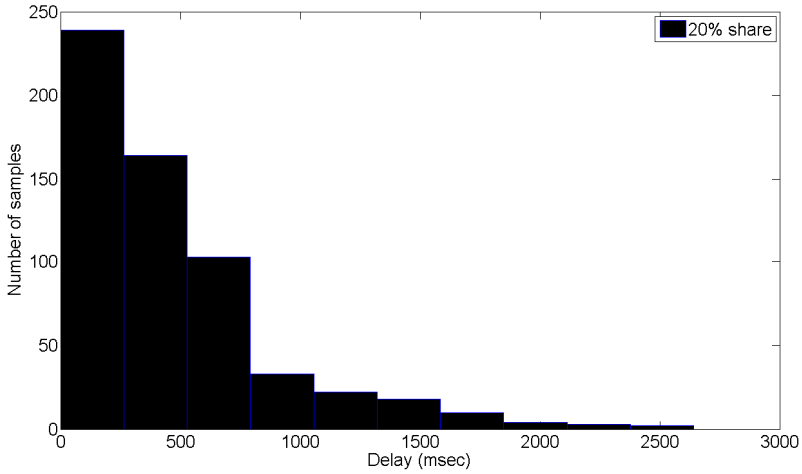
περίπτωση 450msec). Μετά από αυτό το μέγιστο, οι μετρικές δικτύων φαίνονται να βελτιώνονται πάλι, εντούτοις όχι στα επίπεδα των χαμηλών τιμών του P. Παρόμοια συμπεριφορά παρατηρήθηκε για τις παρακείμενες τιμές του P (200mec, 400msec και msec 800). Αυτές παραλήφθησαν για την καλύτερη αναγνωσιμότητα των γραφικών παραστάσεων. Ένα άλλο ενδιαφέρον συμπέρασμα είναι ότι η pinging περίοδος δεν φαίνεται να έχει επιπτώσεις σε αυτήν την συμπεριφορά.



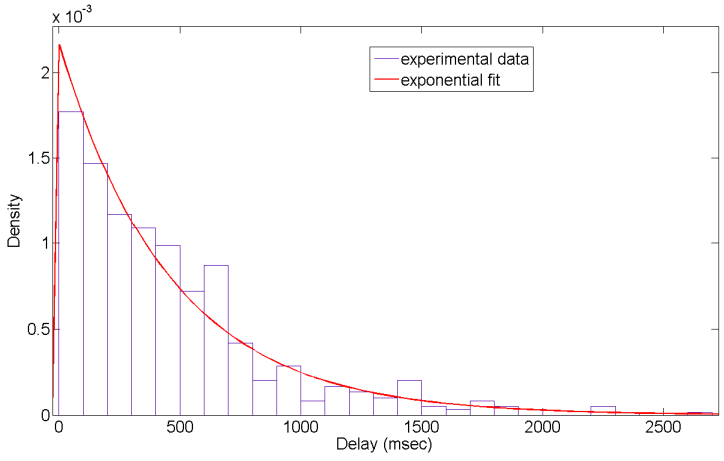
Σχήμα 56: α) Μέση τιμή των χρόνων απόκρισης για πλήρες φορτίο και διάφορες διαμορφώσεις της περιόδου χρονοπρογραμματισμού P και της pinging περιόδου 0.5 δευτερολέπτων β) τυπική απόκλιση των χρόνων απόκρισης για το πλήρες φορτίο και διάφορες διαμορφώσεις της περιόδου χρονοπρογραμματισμού P και της pinging περιόδου 0.5 δευτερολέπτων γ) μέση τιμή απόκρισης για περίοδο pinging 2 seconds δ) τυπική απόκλιση απόκρισης για περίοδο pinging 2 seconds.

Η σελίδα αυτή είναι σκόπιμα λευκή

Η ομοιότητα των μετρικών για τις υψηλές τιμές της κατανομής Q/P (%CPU μερίδιο) οφείλεται στο γεγονός ότι για τα ποσοστά κοντά σε 100% (όπως την περίπτωση 80%), η CPU αφιερώνεται σχεδόν εξολοκλήρου στον στόχο (το task της VM), έτσι η περίοδος αυτής της κατανομής δεν επηρεάζει τη συμπεριφορά της VM.



α) Ιστόγραμμα της κατανομής των μεμονωμένων χρόνων απόκρισης

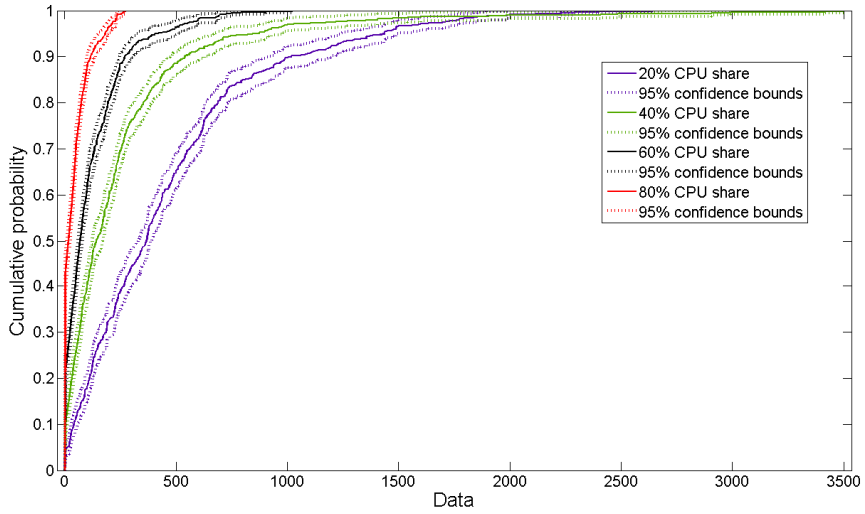


β) Αντιστοίχιση εκθετικής κατανομής στα πειραματικά δεδομένα

Σχήμα 57: α) Κατανομή των μεμονωμένων καθυστερήσεων β) Αντιστοίχιση εκθετικής κατανομής στα πειραματικά δεδομένα περιόδου P=150 χιλιοστών του δευτερολέπτου, περίοδο pinging 0.5 δευτερολέπτων και μερίδιο CPU 60%

Η σελίδα αυτή είναι σκόπιμα λευκή

Για τη στατιστική ανάλυση των μετρήσεων χρησιμοποιήσαμε τη διαμόρφωση για P=150 msec, περίοδο pinging 0.5 δευτερολέπτων και μερίδιο CPU 60%. Η κατανομή των 600 μεμονωμένων τιμών που συλλέχθηκαν κατά τη διάρκεια του πειράματος απεικονίζεται στο Σχήμα 57α. Στο Σχήμα 57β η ταύτιση μιας εκθετικής κατανομής σε αυτές τις τιμές παρουσιάζεται, με μέσο όρο value= 461.544, variance= 213023, std error= 18.8739 και covariance= 356.226. Οι εκθετικές κατανομές χρησιμοποιούνται συνήθως για τη μοντελοποίηση χρόνων απόκρισης δικτύων, όπως επίσης παρουσιάζεται στο [68]. Η καμπύλη CDF και τα διαστήματα εμπιστοσύνης για όλα τα μερίδια CPU αυτής της διαμόρφωσης παρουσιάζονται στο Σχήμα 58, από το οποίο το όφελος από την ανάθεση μεγαλύτερου μεριδίου απεικονίζεται με έναν πιο ποσοτικό τρόπο.



Σχήμα 58: Καμπύλη CDF και διαστημάτα εμπιστοσύνης για όλα τα μερίδια CPU, για περιόδο χρονοπρογραμματισμού 150 χιλιοστών του δευτερολέπτου, περίοδο pinging 0.5 δευτερολέπτων.

Η σελίδα αυτή είναι σκόπιμα λευκή

5.6 Πρόβλεψη επίδρασης μέσω ΝευροΓενετικών μοντέλων

Τα πειράματα που παρουσιάστηκαν ανωτέρω είναι χρήσιμα για έναν προμηθευτή υποδομής Υπολογιστικού Νέφους προκειμένου να έχει μια γενικευμένη εικόνα για το πώς η απόδοση μιας εφαρμογής επιδεινώνεται από τις αποφάσεις συντοποθέτησης άλλων εικονικών μηχανών στον ίδιο κόμβο. Τα συμπεράσματα αυτά θα μπορούσαν να εξαχθούν ως κανόνες fuzzy logic που θα συμβούλευαν τον προμηθευτή κατά τη διάρκεια της πολιτικής κατανομής των VM. Εντούτοις, για να αξιοποιήσουμε τις πλήρεις δυνατότητες αυτής τη προσέγγισης, ένα κατάλληλο γενικό μοντέλο πρέπει να σχεδιαστεί και να υλοποιηθεί. Αυτό το μοντέλο πρέπει να είναι σε θέση να εξάγει τις ακριβείς ποσοτικές προβλέψεις σχετικά με την προσδοκώμενη απόδοση των εφαρμογών ανάλογα με τους τρόπους δρομολόγησης. Πρέπει να είναι σε θέση να πάρει ως είσοδο τους όρους της εφαρμογής και να παράγει το προσδοκώμενο επίπεδο εκτέλεσης (αποτέλεσμα τεστ), κατά συνέπεια ποσοτικοποιώντας ακριβώς την αναμενόμενη επιβάρυνση.

Προκειμένου να υλοποιηθεί ένα τέτοιο σύστημα, η γενικότερη μέθοδος μαύρου κουτιού επιλέχτηκε, ένα τεχνητό νευρωνικό δίκτυο (ΤΝΔ). Τα ΤΝΔ είναι μια μεθοδολογία που στοχεύει να μιμηθεί τη συμπεριφορά του ανθρώπινου εγκεφάλου και έχει χρησιμοποιηθεί μέχρι τώρα για διάφορες εφαρμογές (π.χ. αναγνώριση μοτίβων, προσέγγιση λειτουργίας, ταξινόμηση ή πρόβλεψη χρονοσειράς). Η κύρια δυνατότητά τους έγκειται στην ανίχνευση της συμπεριφοράς ενός συστήματος βασιζόμενα μόνο σε ένα σύνολο εκπαιδευτικών δεδομένων που περιλαμβάνει διάφορες εισόδους και τις αντίστοιχες εξόδους του συστήματος. Γενικά, το ΤΝΔ συσχετίζει τα δεδομένα εισόδου με τα δεδομένα εξόδου προκειμένου να βρεθεί μια γενική προσέγγιση της λειτουργίας του συστήματος που περιγράφει την εξάρτηση της εξόδου από την είσοδο. Επιλέχτηκε

καθώς όπως καταδεικνύεται στην προηγούμενη ενότητα, υπάρχουν πολυάριθμοι, ακόμη και κρυμμένοι, παράγοντες που μπορούν να έχουν επιπτώσεις στην απόδοση μιας εφαρμογής όπως οι λεπτομέρειες σχεδίου υλικού. Η εφαρμογή μιας αναλυτικής προσέγγισης (analytical modeling) είναι σχεδόν αδύνατη, λαμβάνοντας υπόψη την ποικιλία αυτών των παραγόντων. Αλλά ακόμα κι αν κάποιος αφιερώσει προσπάθεια για να δημιουργήσει ένα λεπτομερές αναλυτικό μοντέλο, αυτό θα κριθεί άχρηστο με την επόμενη γενεά επεξεργαστών ή ένα διαφορετικό αρχιτεκτονικό σχέδιο. Το ΤΝΔ προσφέρει μια γενικότερη και εύκαμπτη προσέγγιση, και λαμβάνοντας υπόψη μόνο ένα κατάλληλο εκπαιδευτικό σύνολο δεδομένων, μπορεί να επιτύχει ένα γρήγορο και ικανοποιητικό αποτέλεσμα.

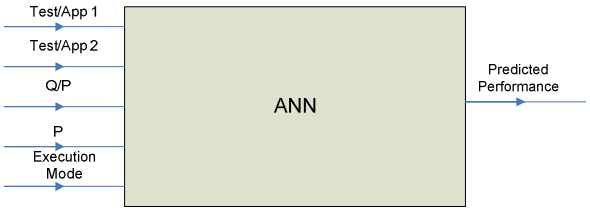
Οι καθορισμένες είσοδοι και έξοδοι του μοντέλου ΤΝΔ απεικονίζονται στο Σχήμα 59 και περιλαμβάνουν τις παραμέτρους που έχουν εξεταστεί στα προηγούμενα τμήματα. Ο Πίνακας 15 περιέχει την κλίμακα των εξεταζόμενων τιμών.

Πίνακας 15: Δυνατές τιμές Εισόδων Μοντέλου Υποδομής ΤΝΔ

Αριθμός Τεστ 1	Από 1 έως 6 (Αριθμός Matlab Benchmark)
Αριθμός Τεστ 2 (συνεκτελούμενη εργασία)	Από 1 έως 6 (Αριθμός Matlab Benchmark)
Q/P (Ποσοστό % ανάθεσης πυρήνα)	0 έως 80 %
Περίοδος χρονοπρογραμματιστή P	200 έως 800 msec
Είδος εκτέλεσης (Execution Mode)	Αυτόνομη, Ίδιος πυρήνας, Παρακείμενοι Πυρήνες, Μη παρακείμενοι πυρήνες

Όλες οι είσοδοι και οι έξοδοι κανονικοποιήθηκαν στο (-1,1) αριθμητικό διάστημα. Για τις μη αριθμητικές εισόδους, ορίστηκαν διαφορετικές τιμές για κάθε κατάσταση. Η

κανονικοποίηση είναι απαραίτητη για την εσωτερική αντιπροσώπευση στο ΤΝΔ και την καλύτερη απεικόνιση του συστήματος. Τα πειραματικά δεδομένα που παρουσιάστηκαν στις προηγούμενες παραγράφους (1100 διαφορετικές εκτελέσεις, κάθε μια για 500 δευτερόλεπτα) χρησιμοποιήθηκαν για την εκπαίδευση (50%), την ενδιάμεση επικύρωση κατά τη διάρκεια της εκπαίδευσης (20%) και την ανεξάρτητη τελική επικύρωση (30%) του ΤΝΔ.



Σχήμα 59: Μοντέλο ΤΝΔ (Artificial Neural Network-ANN) για πρόβλεψη αλληλεπίδρασης εικονικών μηχανών

5.6.1 Βελτιστοποίηση της δομής του ΤΝΔ

Για την βελτιστοποίηση της δομής του ΤΝΔ χρησιμοποιήθηκε ο αλγόριθμος που παρουσιάστηκε στο Κεφάλαιο 2. 13 διαφορετικές συναρτήσεις μεταφοράς εξετάστηκαν (όπως περιγράφονται στο [44]) με μέγιστο όριο δέκα (10) στρωμάτων και τριάντα (30) νευρώνων ανά στρώμα. Τέσσερεις ανεξάρτητες εκτελέσεις χρησιμοποιήθηκαν (30, 50, 100 και 150 γενεές). Κάθε γενεά ερεύνουσε 20 πιθανές λύσεις (μέγεθος πληθυσμού). Ο

Πίνακας 16 περιέχει την δομή ANN και τη γενική ακρίβεια πρόβλεψης για αυτές τις εκτελέσεις. Σε αυτόν τον πίνακα, μόνο το καλύτερο μοντέλο από κάθε εκτέλεση απεικονίζεται. Το μέσο απόλυτο σφάλμα και για τις 328 ανεξάρτητες περιπτώσεις επικύρωσης είναι μεταξύ 4.5 και 5.94, ανάλογα με τον αριθμό γενεών που χρησιμοποιήθηκαν, αποτέλεσμα που θεωρείται πολύ ικανοποιητικό. Μόνο οι τιμές στο

Η σελίδα αυτή είναι σκόπιμα λευκή

Πίνακας 16: Βέλτιστα ΤΝΔ ανά γενεές ΓΑ

Αριθμός Στρωμάτων/ Γενεές ΓΑ	Νευρώνες ανά στρώμα	Συναρτήσεις Μεταφοράς ανά στρώμα	Μέσο Απόλυτο Σφάλμα (%)	Τυπική Απόκλιση	Διάρκεια
4/30	5-7-19-1	Satlin-Radbas-Radbas-Satlins	5.94	16.47	40 λεπτά
3/50	5-7-1	Tansig-Tansig-Purelin	5.06	13.85	1.1 ώρες
4/100	5-10-18-1	Radbas- Softmax-Radbas- Netinv	4.59	14.80	3 ώρες
4/150	5-24-8-1	Purelin-Logsig-Logsig-Satlins	5.16	14.61	5 ώρες

ανεξάρτητο σύνολο επικύρωσης χρησιμοποιήθηκαν για τον υπολογισμό αυτού του λάθους. Αυτά τα στοιχεία δεν έχουν χρησιμοποιηθεί κατά τη διάρκεια της εκπαίδευσης.

Η επιδείνωση στην απόδοση για 150 γενεές δεν αναμενόταν αλλά μπορεί να αποδοθεί στο γεγονός ότι κάθε τρέξιμο GA εξαρτάται από μερικές τυχαίες παραμέτρους, όπως τις αρχικές τιμές του χρωμοσώματος ή την τυχαιότητα στην επιλογή των χαρακτηριστικών των τελεστών mutation και crossover. Σε μια πραγματική κατάσταση, αυτό μπορεί να αντιμετωπιστεί εύκολα με την προσθήκη γενεών έρευνας έως ότου ικανοποιείται ένα ειδικό κριτήριο απόδοσης. Οι καταλληλότερες λύσεις βρίσκονται με τις 50 και 100 γενεές. Η διαφορά βρίσκεται στο γεγονός ότι για 100 γενεές υπάρχει καλύτερο μέσο λάθος αλλά χειρότερη τυπική απόκλιση. Η τελική επιλογή εξαρτάται από το trade-off μεταξύ αυτών των χαρακτηριστικών.

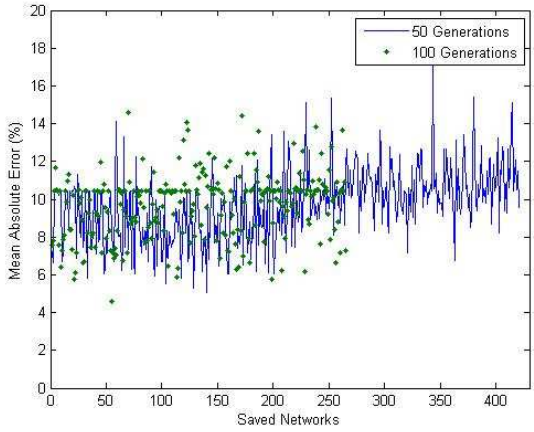
Ένα άλλο ενδιαφέρον συμπέρασμα είναι ότι τα μικρότερα δίκτυα είναι ικανότερα να συλλάβουν τις εξαρτήσεις μεταξύ των εισόδων και των εξόδων. Ακόμα κι αν ο αριθμός στρωμάτων είχε μέγιστη τιμή 10, μόνο τα δίκτυα 3 ή 4 στρωμάτων παρουσίασαν καλή εκτέλεση (3 είναι το ελάχιστο όριο για ΤΝΔ). Αυτό μπορεί να αποδοθεί στην πιό εύρωστη συμπεριφορά των μικρών δικτύων λόγω της ακόλουθης πτυχής. Η επιρροή της εισόδου j στην έξοδο i εξαρτάται από όλες τις εναλλακτικές πορείες που τις συνδέουν και η αύξηση αυτού του αριθμού πορειών (λόγω του υψηλότερου αριθμού στρωμάτων) φαίνεται να αυξάνει τη τυπική απόκλιση της πρόβλεψης.

Στο Σχήμα 60 η εξέλιξη του μέσου απόλυτου σφάλματος των σωζόμενων δικτύων απεικονίζεται για τις δύο καλύτερες εκτελέσεις (50 και 100 γενεών). Αυτά τα δίκτυα σώζονται καθώς το GA προχωρεί και εάν το ενδιάμεσο κριτήριο επικύρωσης ικανοποιείται (λάθος στην ενδιάμεση επικύρωση <10%). Η έλλειψη σταθερότητας (συνεχώς μειωμένο μέσο λάθος) σε αυτήν την διαδικασία είναι εξαιτίας του γεγονότος ότι η τυχαίοτητα είναι μέχρι ένα σημείο έμφυτη σε μια εξελικτική προσέγγιση λόγω της χρήσης των τελεστών όπως η μετάλλαξη και η διασταύρωση.

Επιπλέον, από αυτήν την γραφική παράσταση είναι εμφανές ότι ένας μεγάλος αριθμός καλής ποιότητας δικτύων παράγεται μέσω της προτεινόμενης διαδικασίας βελτιστοποίησης. Φαίνεται επίσης ότι τα δίκτυα που σώζονται στις τελευταίες γενεές του ΓΑ χάνουν σε ποιότητα. Αυτό μπορεί να αποδοθεί στο γεγονός ότι όταν βρίσκει ο ΓΑ μια καλή λύση, εστιάζει επίσης στις κοντινές παρόμοιες διαμορφώσεις. Εάν η καλή λύση είναι το τοπικό ή ολικό βέλτιστο, κατόπιν οι κοντινές λύσεις θα εξεταστούν και ένας μεγάλος αριθμός των υποψηφίων θα αφιερωθεί σε αυτήν την περιοχή μάταια (άλλες λύσεις βασισμένες στη διασταύρωση θα ψάξουν το υπόλοιπο διάστημα δειγμάτων).

Εντούτοις αυτό είναι επίσης μια από τις δυνάμεις των ΓΑ, δεδομένου ότι μέσω αυτής της διαδικασίας στα αρχικά στάδια μεγάλοι τομείς του διαστήματος λύσεων δοκιμάζονται και κατόπιν οι καλύτεροι εξετάζονται λεπτομερώς προκειμένου να βρεθεί η βέλτιστη λύση σε εκείνο το συγκεκριμένο διάστημα.

Η κατανομή των λαθών του δικτύου στην πρόβλεψη της απόδοσης (βαθμολογίες τεστ) συγκρίνεται άμεσα με την μέθοδο της γραμμικής παλινδρόμησης πολλαπλών μεταβλητών στο Σχήμα 61. Αυτή η ακρίβεια είναι βασισμένη μόνο στην ανεξάρτητη επικύρωση, για λόγους αντικειμενικότητας.



Σχήμα 60: Χρονική εξέλιξη της απόδοσης (μέσο απόλυτο σφάλμα στην ανεξάρτητη επικύρωση) των αποθηκευόμενων δικτύων (MAE<10% στην ενδιάμεση επικύρωση)

5.6.2 Σύγκριση με γραμμική παλινδρόμηση πολλαπλών μεταβλητών

Προκειμένου να συγκριθεί η προσέγγισή μας, η μέθοδος γραμμικής παλινδρόμησης πολλών μεταβλητών επιλέχτηκε, η οποία χρησιμοποιείται επίσης στο [81]. Η λειτουργία MATLAB mvregress χρησιμοποιήθηκε, στο ίδιο κανονικοποιημένο σύνολο δεδομένων. Η μόνη διαφορά ήταν ότι για αυτή τη μέθοδο τα δεδομένα εκπαίδευσης και ενδιάμεσης επικύρωσης συγχωνεύθηκαν. Σε αυτήν την μέθοδο, δεν υπάρχει ανάγκη για ενδιάμεσο βήμα επικύρωσης.

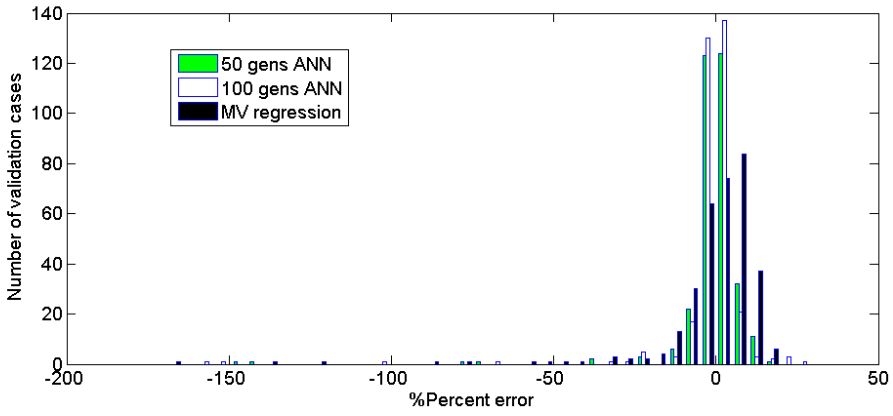
Η σελίδα αυτή είναι σκόπιμα λευκή

Για την επικύρωση των αποτελεσμάτων, το ίδιο 30% ανεξάρτητο σύνολο δεδομένων επικύρωσης χρησιμοποιήθηκε, για λόγους αντικειμενικότητας. Ο Πίνακας 17 περιέχει τις λεπτομέρειες της συνάρτησης προσέγγισης. Το κελί συντελεστών περιέχει στην αντίστοιχη σειρά τους πολλαπλασιαστές για καθεμία από τις 5 εισόδους που απεικονίζονται στο Σχήμα 59, συν το σταθερό παράγοντα. Οι τιμές τους είναι χαμηλές επειδή έχουν παραχθεί για τις κανονικοποιημένες τιμές του συνόλου δεδομένων στο [-1.1] διάστημα.

Πίνακας 17: Συντελεστές και απόδοση μεθόδου γραμμικής παλινδρόμησης

Συντελεστές	Μέσο Απόλυτο Σφάλμα	Τυπική Απόκλιση	Διάρκεια
0.0218, -0.0171, 0.8326, -0.0801, 0.0631, 0	8.78%	18.15	~1 δευτερόλεπτο

Τα αποτελέσματα από αυτήν την μέθοδο έχουν ενσωματωθεί στο Σχήμα 61 για την άμεση σύγκριση με τα καλύτερα μοντέλα ANN από τη μέθοδο βελτιστοποίησης.



Σχήμα 61: Ιστόγραμμα σύγκρισης λαθών του GA-ANN μοντέλου έναντι της γραμμικής παλινδρόμησης πολλαπλών μεταβλητών

Από τη σύγκριση των δύο προσεγγίσεων μπορεί να εξαχθεί το συμπέρασμα ότι τα βελτιστοποιημένα ΤΝΔ έχουν καλύτερο μέσο απόλυτο σφάλμα απόδοσης και μικρότερη τυπική απόκλιση. Αυτό είναι επίσης εμφανές από τη κατανομή των λαθών όπως

Η σελίδα αυτή είναι σκόπιμα λευκή

απεικονίζεται στο ιστόγραμμα. Ενώ τα ANN μοντέλα εμφανίζουν την υψηλή συγκέντρωση των λαθών γύρω από το 0, η γραμμική παλινδρόμιση το εμφανίζει γύρω από το 8-10%.

Οι τιμές τυπικής απόκλισης είναι υψηλές και για τις δύο μεθόδους, αλλά αυτό οφείλεται σε συγκεκριμένες τιμές του συνόλου επικύρωσης για τις οποίες οι μετρήσεις εμφανίζουν μεγάλες διαφορές. Αυτό συμβαίνει για τις αιχμές των γραφικών παραστάσεων που παρουσιάζονται στην παράγραφο 5.5 και μπορεί να αποδοθεί σε τυχαίους παράγοντες που έχουν επιπτώσεις στην απόδοση συστημάτων (π.χ. εργασίες του λειτουργικού συστήματος στο εσωτερικό των VM κ.λπ.), όπως στην περίπτωση P=250 msec που αναφέρθηκε στα προηγούμενα τμήματα.

Εντούτοις, επειδή η προτεινόμενη μέθοδος έχει πολύ μεγαλύτερες υπολογιστικές ανάγκες (όπως φαίνεται από τον χρόνο εκτέλεσης που απαιτείται), για τα περιβάλλοντα όπου τα νέα μοντέλα πρέπει να δημιουργούνται γρήγορα και δυναμικά η πολλών μεταβλητών μέθοδος παλινδρόμησης προσφέρει μια ικανοποιητική απόδοση με έναν πολύ γρήγορο τρόπο. Στα πλαίσια της έρευνας σε αυτή τη διατριβή (υποδομή/προμηθευτής Νέφους που θέλει να έχει ένα μοντέλο για την αποδοτικότερη κατανομή των VM στους κόμβους), οι υπολογιστικές ανάγκες της βελτιστοποιημένης μεθόδου ANN δεν θεωρούνται περιοριστικές. Τα μοντέλα θα δημιουργηθούν μιά φορά, και μπορούν να αναπαραχθούν στο μέλλον σε περίπτωση συγκέντρωσης νέων δεδομένων. Εντούτοις, στενοί χρονικοί περιορισμοί συγχρονισμού για την παραγωγή των μοντέλων δεν υπάρχουν.

Ο Πίνακας 18 απεικονίζει τις λεπτομερείς βελτιώσεις % που επιτυγχάνονται με τη συγκρινόμενη μέθοδο. Για το χρόνο απόκρισης των μοντέλων, δηλαδή το χρόνο που

απαιτείται για κάθε μοντέλο για να παρέχει την πρόβλεψη μετά από την εφαρμογή των εισόδων, αυτό ήταν καλύτερο για την πολλών μεταβλητών παλινδρόμηση. Εντούτοις, και για το ΤΝΔ ήταν πολύ γρήγορο (~10 χιλιοστά του δευτερολέπτου), τόσο ώστε να μη δημιουργεί προβληματισμούς.

Πίνακας 18: % Βελτίωση των μετρικών σφάλματος μέσω των βελτιστοποιημένων ΤΝΔ

Μέθοδος	%Βελτίωση (ANN Μέσο Απόλυτο Σφάλμα σε σχέση με MVregress)	%Βελτίωση (ANN Τυπική Απόκλιση σε σχέση με MVregress)
GA-ANN 50 Γενεών	42.36%	23.64%
GA-ANN 100 Γενεών	47.61%	18.4%

5.7 Συμπεράσματα

Σαν συμπέρασμα, η εμφάνιση των virtualized και συνχρησιμοποιούμενων υποδομών όπως τα Υπολογιστικά Νέφη συνοδεύεται από μια σημαντική πρόκληση για τους παρόχους και τις εφαρμογές. Οι εφαρμογές που εκτελούνται στο εσωτερικό των εικονικών μηχανών επηρεάζονται από πολλούς παράγοντες όπως η κατανομή άλλων VM στον ίδιο φυσικό υπολογιστικό κόμβο. Σε αυτό το κεφάλαιο, οι επιπτώσεις αυτών των αποφάσεων κατανομής ερευνήθηκαν, με την εξέταση ένος σημαντικού αριθμού παραμέτρων όπως οι αποφάσεις χρονοδρομολόγησης πραγματικού χρόνου, οι τύποι φόρτων εργασίας και τα διαφορετικά σενάρια κατανομής. Η επιβάρυνση που δημιουργείται από αυτές τις παραμέτρους μπορεί να φθάσει μέχρι και στο 150%, ενώ η προσεκτική επιλογή της κατανομής των VM ανα κόμβο και του σεναρίου της

τοποθέτησης μπορεί να ελαχιστοποιήσει ή ακόμα και να ακυρώσει αυτήν την επίδραση. Επίσης, υψηλότερες περίοδοι χρονοδρομολόγησης οδηγούν σε μια σημαντική μείωση της επιβάρυνσης. Επιπλέον, η εφαρμογή μιας τέτοιας μεθόδου ανάλυσης απόδοσης στο σχεδιασμό νέου υλικού μπορεί να οδηγήσει στε βελτιστοποιημένες multi-core αρχιτεκτονικές για τα περιβάλλοντα Υπολογιστικού Νέφους, μέσω του προσδιορισμού των δυσχερειών όπως ο κοινός δίαυλος μνήμης.

Προκειμένου να παρασχεθεί ένα πλαίσιο αυτόματης απόφασης που θα καθοδηγήσει τις υποδομές στις κατάλληλες διαμορφώσεις και θα αφαιρέσει την ανάγκη της ανθρώπινης παρέμβασης, ένα βελτιστοποιημένο μοντέλο ΤΝΔ μπορεί να χρησιμοποιηθεί για να ποσοτικοποιήσει και να προβλέψει με ακρίβεια την απόδοση των εφαρμογών για μια δεδομένη διαμόρφωση. Το μοντέλο που δημιουργείται από αυτήν την διαδικασία είναι αποτελεσματικό (σφάλμα<5%), γενικό και μπορεί να επεκταθεί οποιαδήποτε στιγμή προκειμένου να περιληφθούν νέα σημαντικά σενάρια ή παράγοντες. Μέσω της χρήσης αυτού του μηχανισμού, μια υποδομή Υπολογιστικού Νέφους μπορεί να έχει την a priori γνώση της απόδοσης μιας συγκεκριμένης διαμόρφωσης. Κατά συνέπεια είναι σε θέση να βελτιστοποιήσει τη διαχείριση των φυσικών υπολογιστικών πόρων.

Για το μέλλον, μια ενδιαφέρουσα πτυχή που θα μπορούσε να διερευνηθεί είναι η αυτόματη ανίχνευση του τύπου φόρτου εργασίας που αντιστοιχεί σε κάθε εφαρμογή. Σε αυτήν την μελέτη, η ανάλυση βασίστηκε στις 6 δοκιμές επίδοσης Matlab, οι οποίες απεικονίζουν τους διαφορετικούς τύπους υπολογισμών. Για τις πραγματικές εφαρμογές αυτό σημαίνει ότι πρέπει να αντιστοιχηθούν σε μια ή περισσότερες από αυτά τα στοιχειώδη τεστ, με βάση παραδείγματος χάριν τη σύγκριση των ιχνών τους πάνω στο

υλικό (χρήση μνήμης, ποσοστού χρόνου υπολογιστικής ισχύος, ευστοχίες/αστοχίες cache κλπ).

6 *Συμπεράσματα και μελλοντική εργασία*

Στο συγκεκριμένο κεφάλαιο περιλαμβάνεται η σύνοψη της διατριβής και τα συμπεράσματα που εξήχθησαν κατά την εκπόνησή της, η συνεισφορά και η καινοτομία που επιδεικνύει στον αντίστοιχο ερευνητικό χώρο, και συζητούνται θέματα μελλοντικής εργασίας και επέκτασης των ερευνητικών αποτελεσμάτων.

6.1 Σύνοψη και Συμπεράσματα

Στην παρούσα διατριβή παρουσιάστηκαν γενικά στοιχεία για τα περιβάλλοντα Υπολογιστικών Νεφών, καθώς και τα συγκεκριμένα χαρακτηριστικά του πολυστρωματικού υπηρεσιοστρεφούς πλαισίου μέσα στο οποίο οφείλει να λειτουργεί ένας μηχανισμός πρόβλεψης απόδοσης εφαρμογών. Πραγματοποιήθηκε ανάλυση των ανοικτών ερευνητικών θεμάτων στην επίμαχη περιοχή και των υφιστάμενων προκλήσεων.

Με βάση αυτό το πλαίσιο και τους προκύπτοντες περιορισμούς αναλύθηκαν υπάρχουσες λύσεις και αναπτύχθηκε ένας καινοτόμος αλγόριθμος δημιουργίας και

βελτιστοποίησης μοντέλων για την πρόβλεψη απόδοσης σε υπηρεσιοστρεφείς υποδομές. Το βασικό συστατικό του αλγόριθμου αυτού, τα ΤΝΔ, ικανοποιούν τους βασικούς περιορισμούς ανίχνευσης της επίδρασης των προγνωστών SLA χωρίς την ύπαρξη γνώσης για την εσωτερική δομή της εφαρμογής ή τον πηγαίο κώδικα. Από την άλλη, οι ΓΑ βοηθούν στην αυτοματοποίηση και βελτιστοποίηση μιας χειροκίνητης και ενστικτώδους διαδικασίας όπως ο καθορισμός των σχεδιαστικών παραμέτρων ενός ΤΝΔ, και επιτρέπουν την χρήση της μεθοδολογίας σε υπηρεσιοστρεφή πλαίσια. Η συνολική μεθοδολογία ελέγχεται μέσω τριών (3) πειραμάτων τα οποία επιβεβαιώνουν την εφαρμογή της μεθόδου σε πραγματικές εφαρμογές, με χρήση ελάχιστης πληροφορίας για τις τελευταίες, και με πολύ ικανοποιητικά επίπεδα ακρίβειας σε προβλέψεις ανά εκτέλεση με συγκεκριμένες παραμέτρους (φόρτου και ποιότητας υπηρεσίας). Έτσι, επιτρέπει την απευθείας μετάφραση των όρων υψηλού επιπέδου βάσει των οποίων συντάσσονται τα Συμβόλαια Επιπέδου Υπηρεσιών σε όρους φυσικών πόρων.

Παράλληλα η υπηρεσιοστρεφής υλοποίηση του πλαισίου για την προσφορά της εκτίμησης απόδοσης ακολούθησε μια διαστρωματική μορφή, που επιτρέπει την αποσύζευξη του πλαισίου από τον χρησιμοποιούμενο αλγόριθμο πρόβλεψης. Έτσι την καθιστά ευέλικτη, με τη δυνατότητα αντικατάστασης της ελλοχεύουσας μεθοδολογίας με διαφορετικές προσεγγίσεις. Επιπλέον, η δυνατότητα συγγραφής αυτής σε μια γλώσσα υψηλού επιπέδου (όπως είναι η scripting γλώσσα εργαλείων όπως το Matlab και το Octave) ελαχιστοποιεί το χρόνο δημιουργίας της και επιτρέπει την εκμετάλλευση πλήθους υφιστάμενων βιβλιοθηκών και εργαλείων.

Επιπλέον, η ανάλυση της χρονικής συμπεριφοράς και των τμηματικών καθυστερήσεων του προαναφερθέντος πλαισίου οδήγησε σε αλλαγή/εμπλουτισμό του

σχεδιασμού της υπηρεσίας ώστε να παρέχεται η δυνατότητα ελαχιστοποίησης των απαιτούμενων πόρων από την τελευταία για την παροχή εκτιμήσεων σε υποδομές μεγάλης κλίμακας. Διαφορετικά πρωτόκολλα επικοινωνίας μελετήθηκαν (SOAP και REST) και πραγματοποιήθηκε σύγκριση με βάση την απόδοση και την αξιοπιστία τους. Το REST πρωτόκολλο εμφάνιζε βελτίωση μιας τάξης μεγέθους όσον αφορά την ταχύτητα, αλλά και ποσοστό αποτυχίας κλήσεων το οποίο αύξανε ανάλογα με το πλήθος των συνολικών αιτήσεων.

Για την πλήρη αυτοματοποίηση του πλαισίου πρόβλεψης, ο μηχανισμός επεκτάθηκε σε 2 επίπεδα, συμπληρώνοντας τα ΤΝΔ προσέγγισης συνάρτησης με ΤΝΔ πρόβλεψης χρονοσειράς, ώστε να ανιχνεύεται εκ των προτέρων η αναμενόμενη κίνηση. Με αυτό τον τρόπο ο ιδιοκτήτης της υπηρεσίας αρκεί να προσδιορίσει την απαιτούμενη ποιότητα υπηρεσίας και ο μηχανισμός αναλαμβάνει όλες τις υπόλοιπες εργασίες. Ο συνδυασμένος μηχανισμός αξιολογήθηκε σε πραγματικά δεδομένα και με βάση δειγματοστρεφή προσομοίωση, της οποίας το σφάλμα ήταν πολύ ικανοποιητικό (<10%), δείχνοντας ότι μπορεί να χρησιμοποιηθεί σε ρεαλιστικά υπηρεσιοστρεφή συστήματα για την αυτοματοποίηση της διαχείρισης.

Όσον αφορά τους παρόχους υποδομής (IaaS Providers), στα πλαίσια της διατριβής μελετήθηκαν πειραματικά παράγοντες επίδρασης στην πραγματική απόδοση εικονικών μηχανών, όταν αυτές εκτελούνται ταυτόχρονα σε διαμοιραζόμενους πόρους. Η επίδραση αυτή μπορεί να είναι σημαντική και να υποβαθμίζει την απόδοσή τους σε μεγάλο βαθμό, ανάλογα με το είδος της υπολογιστικής εργασίας που εκτελείται μέσα στις VMs. Οι καλύτεροι συνδυασμοί που την ελαχιστοποιούν περιλαμβάνουν κυρίως εργασίες γραφικής επεξεργασίας. Επιπλέον η περίοδος του χρονοπρογραμματιστή, μέσα στην

οποία δίνεται ένα συγκεκριμένο ποσοστό της CPU σε κάθε εργασία, επηρεάζει επίσης σημαντικά την απόδοση των εφαρμογών. Για τις καθαρά υπολογιστικές εφαρμογές η περίοδος αυτή πρέπει να είναι όσο το δυνατόν μεγαλύτερη, ενώ για τις διαδραστικές εφαρμογές μια μικρότερη τιμή είναι βέλτιστη. Ανάλογα με την εφαρμογή, αυτή η τιμή μπορεί να διαφέρει. Η μοντελοποίηση αυτών των παραγόντων με ΤΝΔ μπορεί να προσφέρει ασφαλή πρόβλεψη της επίδρασης (με μέσο σφάλμα μικρότερο του 5%), ώστε ο πάροχος υποδομών να τη λαμβάνει υπόψη στη διαδικασία απόφασης ανάθεσης εργασιών σε κόμβους.

6.2 Μελλοντική Εργασία

Οι μεθοδολογίες και υλοποιήσεις που αναπτύχθηκαν στο πλαίσιο αυτής της διατριβής μπορούν να επεκταθούν ή να συνδυαστούν ώστε να προκύψουν νέα ενισχυμένα συστήματα. Στις επόμενες παραγράφους αναφέρονται ενδεικτικά πεδία μελλοντικής εργασίας.

6.2.1 Μηχανισμοί Ανίχνευσης Δραστηριότητας Δεδομένων

Στις σύγχρονες υποδομές Υπολογιστικών Νεφών, η έννοια της ομοσπονδίας Παρόχων (Cloud Federation) αναπτύσσεται τα τελευταία χρόνια. Στο πλαίσιο αυτής διαφορετικοί πάροχοι IaaS διασυνδέονται ώστε να έχουν εναλλακτικές πηγές πόρων, σε περίπτωση εσωτερικής έλλειψης. Όταν διαπιστώνεται η τελευταία, ένα μέρος των εικονικών μηχανών διοχετεύονται στον συνεργαζόμενο πάροχο και εκτελούνται στις αντίστοιχες υποδομές. Το σημαντικό σε αυτή τη περίπτωση είναι η επιλογή των εικονικών μηχανών που θα ομοσπονδοποιηθούν να βασίζεται σε κριτήρια βελτιστοποίησης της απόδοσής τους. Δεδομένου ότι τα δεδομένα εφαρμογής που

χρησιμοποιεί μία εικονική μηχανή μπορεί να είναι αποθηκευμένα στον αρχικό πάροχο, και είναι χρονοβόρο και κοστοβόρο να μεταφερθούν στον ομόσπονδο, η μεταφορά της συγκεκριμένης VM σε μια απομακρυσμένη τοποθεσία θα μειώσει σημαντικά την απόδοσή της λόγω της απομακρυσμένης πρόσβασης.

Για τον λόγο αυτό, είναι ανάγκη να δημιουργηθεί ένα πλαίσιο το οποίο θα κατηγοριοποιεί τους τύπους εικονικών μηχανών ανάλογα με το μέγεθος της δραστηριότητας στα κεντρικά αποθηκευμένα δεδομένα και θα μπορεί να προβλέπει τις μελλοντικές προσπελάσεις. Σε αυτό μπορούν να χρησιμοποιηθούν οι μηχανισμοί πρόβλεψης χρονοσειράς που αναπτύχθηκαν στο Κεφάλαιο 4. Μια αρχική υλοποίηση αυτού του μηχανισμού απεικονίζεται στο [114].

6.2.2 Ανίχνευση και Κατηγοριοποίηση Εφαρμογής

Στο κεφάλαιο 5 αναλύθηκε η μείωση της απόδοσης βασικών δοκιμαστικών εφαρμογών λόγω της ταυτόχρονης εκτέλεσής τους στον ίδιο φυσικό κόμβο. Παρόλαυτα, για να γενικευθεί η χρήση των αποτελεσμάτων της μελέτης χρειάζεται ένας αυτοματοποιημένος τρόπος αντιστοίχισης γενικών εφαρμογών στα συγκεκριμένα δοκίμια (benchmarks).

Ο στόχος της μελλοντικής εργασίας σε αυτόν τον τομέα είναι η απόκτηση του ίχνους της γενικής εφαρμογής που υποδεικνύει τη συμπεριφορά της πάνω στους υπολογιστικούς πόρους και η ταξινόμησή της μέσω της σύγκρισής του με το ίχνος των δοκιμίων.

6.2.3 Συνδυασμός Ελέγχου Εισδοχής Εργασιών και Αλληλεπίδρασης Εικονικών Μηχανών

Πριν την αποδοχή ενός Συμβολαίου Επιπέδου Υπηρεσίας από την πλευρά ενός παρόχου υποδομών είναι απαραίτητη η πραγματοποίηση ενός ελέγχου εισδοχής, που θα επιχειρήσει να αναθέσει τις εικονικές μηχανές της νέας υπηρεσίας στους διαθέσιμους υπολογιστικούς κόμβους. Για να γίνει αυτό ένας τέτοιος μηχανισμός (π.χ. [115]) χρησιμοποιεί ένα μοντέλο ανάθεσης μηχανών σε κόμβους το οποίο προσπαθεί να ελαχιστοποιήσει μέσω κατάλληλων επιλυτών.

Μία ενδιαφέρουσα προσθήκη σε αυτό το μοντέλο είναι η προβλεπόμενη αλληλεπίδραση μεταξύ των εικονικών μηχανών, όπως αναλύθηκε στο Κεφάλαιο 5. Με αυτό τον τρόπο ο πάροχος θα μπορεί να βρει την αντιστοίχιση μηχανών σε πόρους που μεγιστοποιεί την εναπομείνουσα χωρητικότητα των κόμβων αλλά και ελαχιστοποιεί την αλληλεπίδραση μεταξύ των συγκατοικούντων εικονικών μηχανών.

6.2.4 Μηχανισμοί Ανίχνευσης Επιθέσεων Άρνησης Υπηρεσίας και Ελαστικότητα Εφαρμογών

Όπως αναφέρθηκε στην εισαγωγή και στο Κεφάλαιο 4, ένα από τα μεγαλύτερα πλεονεκτήματα των Υπολογιστικών Νεφών είναι οι ελαστικοί πόροι. Ο ιδιοκτήτης μιας υπηρεσίας μπορεί να ρυθμίζει δυναμικά τον αριθμό των πόρων που χρησιμοποιεί η τελευταία ώστε να παρέχει σταθερά επίπεδα υπηρεσίας με βάση τη ζήτηση.

Ένα πρόβλημα που προκύπτει από αυτή τη λειτουργία είναι τι γίνεται στην περίπτωση επίθεσης Άρνησης Υπηρεσίας (Denial of Service Attack). Σε τέτοιες επιθέσεις, ένας κακόβουλος χρήστης εξαπολύει μαζικές αιτήσεις προς τον εξυπηρετητή με σκοπό να τον βγάλει εκτός λειτουργίας λόγω υπερφόρτωσης. Στην περίπτωση όμως

της εφαρμογής της ελαστικότητας, ο εξυπηρετητής θα δεσμεύει όλο και περισσότερους πόρους, καταφέρνωντας να εξυπηρετήσει τις αιτήσεις, χωρίς όμως αυτό να του αποφέρει κέρδος. Αντίθετα θα προκύπτει σημαντική οικονομική ζημία λόγω της υπερδέσμευσης πόρων.

Ο τρόπος επίλυσης που θα επιδιωχθεί είναι η πιθανοτική μοντελοποίηση των μεταβάσεων από διαδοχικά σημεία προόδου της εφαρμογής ώστε να διαπιστώνεται κατά τη διάρκεια της λειτουργίας της αν αυτή η πρόοδος είναι αντίστοιχη των αφιχθεισών αιτήσεων. Αν δεν ισχύει αυτή η συνθήκη, τότε η διαδικασία ελαστικοποίησης θα απενεργοποιείται και θα πραγματοποιείται φιλτράρισμα των πηγών των οποίων οι αιτήσεις δεν προχωρούν στην αλυσίδα δημιουργίας αξίας.

Γλωσσάριο

Στον παρακάτω πίνακα παρατίθενται οι όροι που χρησιμοποιήθηκαν στη Διατριβή:

ANN	Artificial Neural Networks
API	Application Programming Interface
ASCD	Application Service Component Description
CPU	Central Processing Unit
CSV	Comma Separated Values
DoS	Denial of Service
EDF	Earliest Deadline First
FLOPS	Floating point Operations per Second
FPS	Frames Per Second
FTP	File Transfer Protocol
GA	Genetic Algorithms
GAMS	General Algebraic Modelling System
GTv4	Globus Toolkit version 4
HPC	High Performance Computing
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IaaS	Infrastructure as a Service
IRMOS	Interactive Realtime Applications over Service Oriented Infrastructures
KPI	Key Performance Indicator
KVM	Kernel Virtual Machine
MAE	Mean Absolute Error
MIPS	Million Instructions Per Second
MPEG	Moving Picture Experts Group
MPI	Message Passing Interface
MSE	Mean Square Error
NARX	Non-linear Autoregressive Networks with Exogenous Inputs
NIST	National Institute of Science and Technology
OCCI	Open Cloud Computing Interface
OS	Operating System
OVF	Open Virtualization Format
PaaS	Platform as a Service
QoS	Quality of Service
RAM	Random Access Memory
REST	Representational State Transfer
SaaS	Software as a Service
SLA	Service Level Agreements

SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOI	Service Oriented Infrastructure
SPI	Software Platform Infrastructure
SSH	Secure Shell
VM	Virtual Machine
WCET	Worst Case Execution Time
WS	Web Service
WSDL	Web Services Description Language
WSRF	Web Services Resource Framework
XML	Extensible Markup Language

Βιβλιογραφικές Αναφορές

[1] Peter Mell, Timothy Grance, “The NIST Definition of Cloud Computing”, Special Publication 800-145, National Institute of Standards and Technology, September 2011

[2] T. Erl, “Service-oriented Architecture: Concepts, Technology, and Design”, Upper Saddle River: Prentice Hall PTR. ISBN 0-13-185858-0, 2005.

[3] I. Foster, C. Kesselman, S. Tuecke, “*The Anatomy of the Grid: Enabling Scalable Virtual Organizations*” International Journal Supercomputer Applications, Vol. 15, No. 3, 2001.

[4] http://occi-wg.org/

[5] Open Virtualization Format Specification v1.0.0, September 2009, available at http://xml.coverpages.org/DMTF-OVF-v10-DSP0243.pdf

[6] Debusmann, M.; Keller, A., "SLA-driven management of distributed systems using the common information model," Integrated Network Management, 2003. IFIP/IEEE Eighth International Symposium on , vol., no., pp. 563-576, 24-28 March 2003.

[7] G. Wang, A. Chen, C. Wang, C. Fung, and S. Uczekaj, “Integrated Quality of Service (QoS) Management in Service-Oriented Enterprise Architectures”, In Proceedings of the 8th International IEEE Enterprise Distributed Object Computing Conference (EDOC), Monterey, California, September 2004.

[8] Zhengting He, Cheng Peng, Aloysius Mok, "A Performance Estimation Tool for Video Applications," rtas, pp. 267-276, 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06), 2006

[9] Siegfried Benkner, Gerhard Engelbrecht, "A Generic QoS Infrastructure for Grid Web Services," aict-iciw, p. 141, Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06), 2006

[10] Chen, Iyer, Liu, Milojicic, Sahai, "SLA Decomposition: Translating Service Level Objectives to System Level Thresholds," icac, p. 3, Fourth International Conference on Autonomic Computing (ICAC'07), 2007

[11] Jae W. Lee, Krste Asanovic, "METERG: Measurement-Based End-to-End Performance Estimation Technique in QoS-Capable Multiprocessors," rtas, pp. 135-147, 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06), 2006

[12] Peer Hasselmeyer, Bastian Koller, Lutz Schubert, Philipp Wieder: Towards SLA-Supported Resource Management. HPCC 2006: 743-752

[13] Jarvis, S. A., Spooner, D. P., Keung, H. N., Cao, J., Saini, S., and Nudd, G. R. 2006.Performance prediction and its use in parallel and distributed computing systems.Future Gener. Comput. Syst. 22, 7 (Aug. 2006), 745-754.

[14] Oana Florescu, Menno de Hoon, Jeroen Voeten, and Henk Corporaal. Probabilistic modelling and evaluation of soft real-time embedded systems. In Proceedings of SAMOS VI, LNCS 4017,2006.

[15] Singh, K., İpek, E., McKee, S. A., de Supinski, B. R., Schulz, M., and Caruana, R. 2007. Predicting parallel application performance via machine learning approaches: Research Articles. Concurr. Comput. : Pract. Exper. 19, 17 (Dec. 2007), 2219-2235.

[16] A. Kelemen, and Y. Liang, Bayesian Regularized Neural Network for Multiple Gene expression Pattern Classification,Neural Networks, 2003. Proceedings of the International Joint Conference on, Vol. 1, 20-24 July 2003, pp. 654 - 659.

[17] W. Duch and N. Jankowski. Survey of neural transfer functions. Neural Computing Surveys, 2:163–212, 1999.

[18] U. Anders and O. Korn, “Model selection in neural networks,” Neural Networks, vol. 12, pp. 309–323, 2000.

[19] S. Areibi, M. Moussa, H. Abdullah, A comparison of genetic/memetic algorithms and other heuristic search techniques, in: ICAI 2001, Las Vegas, Nevada, 2001.

[20] C. J. Fourie andW. J. Perold, “Comparison of genetic algorithms to other optimization techniques for raising circuit yield in superconducting digital circuits,” IEEE Trans. Appl. Supercond., vol. 13, no. 2, pp. 511–514, Jun. 2003.

[21] Moré, J.J.: The Levenberg-Marquardt algorithm: implementation and theory. In: Watson, G.A. (ed.) Numerical Analysis, Dundee 1977. Lecture Notes in Mathematics, vol. 630, pp. 105–116. Springer, Berlin (1978)

[22] R. Wolski, N. Spring, and J. Hayes, “The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing,” Journal of Future Generation Computing Systems, vol. 15, no. 5-6, Elsevier, 1999

[23] A. Strube, D. Rexachs, E. Luque, “Software probes: Towards a quick method for machine characterization and application performance prediction”, Proceedings of the 7th International Symposium on Parallel and Distributed Computing, 2008.

[24] Martin T. Hagan, Howard B. Demuth, Mark H. Beale, “Neural Network Design”, PWS Publishing, 1996.

[25] Magnus R. Hestenes and Eduard Stiefel Methods of Conjugate Gradients for Solving Linear Systems, Journal of Research of the National Bureau of Standards, 49: 409–436, 1952.

[26] Holland, J. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press.

[27] D.W. Walker, “The design of a standard message passing interface for distributed memory concurrent computers”, Parallel Computing 20 (1994), 657–673.

[28] Krawezik, G. 2003. Performance comparison of MPI and three openMP programming styles on shared memory multiprocessors. In Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures (San Diego, California, USA, June 07 - 09, 2003). SPAA '03. ACM, New York, NY, 118-127.

[29] Fabrice Bellard, FFMPEG multimedia system, http://www.ffmpeg.org, 2005.

[30] IRMOS Project: “Irmos Application Blueprint”, December 2009, available at: http://www.irmosproject.eu/Publications/Download.aspx?ID=128

[31] G. Kousiouris, D. Kyriazis, K. Konstanteli, S. Gogouvitis, G. Katsaros, D. Varvarigou, ”A Service-Oriented Framework for GNU Octave-based Performance Prediction”, in the 2010 IEEE Services Computing Conference, Miami, USA, July 2010.

[32] Checconi, Fabio and Cucinotta, Tommaso and Faggioli, Dario and Lipari, Giuseppe, “Hierarchical Multiprocessor CPU Reservations for the Linux Kernel”, Proceedings of the 5th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT 2009)

[33] FHF Leung, HK Lam, SH Ling, PKS Tam, “Tuning of the Structure and Parameters of a Neural Network Using an Improved Genetic Algorithm”, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 14, NO. 1, JANUARY 2003

[34] J Ilonen, JK Kamarainen, J Lampinen, ” Differential Evolution Training Algorithm for Feed-Forward Neural Networks”, Neural Processing Letters, Springer , Volume 17, Number 1 / February, 2003

[35] IRMOS Project Deliverable D4.1.1 “Definition and implementation of the three scenarios and its real time requirements”, by TSG and other IRMOS partners, February 2009

[36] George Kousiouris, Fabio Checconi, Alessandro Mazzetti, Zlatko Zlatev, Juri Papay, Thomas Voith, Dimosthenis Kyriazis, “Distributed Interactive Real-time Multimedia Applications: A Sampling and Analysis Framework”, under review for the 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems, Belgium, July 2010.

[37] Fiszelew, A., Britos, P., Ochoa, A., Merlino, H.,Fernández, E., García-Martínez, R., “Finding Optimal Neural Network Architecture Using Genetic Algorithms”, Advances in Computer Science and Engineering Research in Computing Science 27, 2007

[38] Giustolisi, Orazio and Simeone, Vincenzo(2006) 'Optimal design of artificial neural networks by a multi-objective strategy: groundwater level predictions / Construction optimale de réseaux de neurones artificiels selon une stratégie multi-objectifs: prévisions de niveau piézométrique', Hydrological Sciences Journal, 51: 3, 502 — 523

[39] http://www.irmosproject.eu

[40] Hey, T., Papay, J. and Surridge, M. (2005) The Role of Performance Engineering Techniques in the Context of the Grid. Concurrency and Computation: Practice and Experience, 17 (2-4). pp. 297-316. ISSN 1532-0626

[41] M. Gerndt, M. Ott, (2009) "Automatic performance analysis with periscope", Concurrency and Computation: Practice and Experience, Volume 22 Issue 6, Pages 736 – 748, 2009, ISSN 1532-0626

[42] Youseff L, Butrico M, Da Silva D (2008) Toward a unified ontology of cloud computing. In: Grid computing environments workshop, 2008. GCE '08grid computing environments workshop,2008, GCE '08, pp 1–10P.

[43] http://aws.amazon.com/ec2/

[44] http://www.mathworks.com/help/toolbox/nnet/function.html

[45] http://www.globus.org/toolkit/

[46] http://www.gnu.org/software/octave/

[47] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A view of cloud computing. Commun. ACM 53, 4 (April 2010), 50-58. DOI=10.1145/1721654.1721672 http://doi.acm.org/10.1145/1721654.1721672

[48] http://collaborate.nist.gov/twiki-cloud-computing/bin/view/CloudComputing/ReferenceArchitectureTaxonomy

[49] Marc N. Haines and Marcus A. Rothenberger. 2010. How a service-oriented architecture may change the software development process. *Commun. ACM* 53, 8 (August 2010), 135-140. DOI=10.1145/1787234.1787269 http://doi.acm.org/10.1145/1787234.1787269

[50] Heiko Koziolek, Performance evaluation of component-based software systems: A survey, Performance Evaluation, Volume 67, Issue 8, Special Issue on Software and Performance.

[51] http://support.gams-software.com/doku.php?id=platform:aws

[52] Roy T. Fielding and Richard N. Taylor. 2002. Principled design of the modern Web architecture. *ACM Trans. Internet Technol.* 2, 2 (May 2002), 115-150. DOI=10.1145/514183.514185 http://doi.acm.org/10.1145/514183.514185

[53] http://jersey.java.net/

[54] http://glassfish.java.net/

[55] E. Smirnova, C.M. So, S.M. Watt, Providing mathematical Web services using Maple in the MONET architecture. In Procs. MONET Workshop (2004)

[56] YarKhan, A., Dongarra J., Seymour K., 2007, in IFIP International Federation for Information Processing, Volume 239. Grid-Based Problem Solving Environments, eds. Gaffney, P. W.. Pool, J.C.T., (Boston: Springer), pp. 215–224.

[57] http://www.hpcgrid.com

[58] http://genss.cs.bath.ac.uk/index.htm

[59] Petcu, D. 2006. BetweenWeb and Grid-based Mathematical Services. In Proceedings of the international Multi-Conference on Computing in the Global information Technology (August 01 - 03, 2006). ICCGI. IEEE Computer Society, Washington, DC, 41. DOI= http://dx.doi.org/10.1109/ICCGI.2006.52

[60] A. Al Zain, K. Hammond, P.W. Trinder, S.A. Linton, H.-W. Loidl, and M. Costanti. SymGrid-Par: Designing a Framework for Executing Computational Algebra Systems on Computational Grids. In Proc. International Conference on Computer Science (Workshop on Practical Aspects of High-level Parallel Programming), Beijing, China, May 27-30, 2007, 2007

[61] http://kenai.com/projects/javaoctave/downloads

[62] Paola Laface, Damiano Carra and Renato Lo Cigno, “A Performance Model for Multimedia Services Provisioning on Network Interfaces”, in Lecture Notes in Computer Science, Volume 3375/2005, Springer Berlin / Heidelberg

[63] Roger J. Castaldo Michael A. McKay Vladimir Tosic, “Exposing GNU Octave Signal Processing Functions as Extensible Markup Language (XML) Web Services”, in Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on

[64] IRMOS Project D3.1.2: “IRMOS Overall Architecture”, NTUA and other partners, February 2009

[65] George Kousiouris, Tommaso Cucinotta, Theodora Varvarigou, "The Effects of Scheduling, Workload Type and Consolidation Scenarios on Virtual Machine Performance and their Prediction through Optimized Artificial Neural Networks , The Journal of Systems and Software (2011),Volume 84, Issue 8, August 2011, pp. 1270-1291, Elsevier, doi:10.1016/j.jss.2011.04.013

[66] Lianzhang Zhu; Xiaowing Liu; , "Technical Target Setting in QFD for Web Service Systems Using an Artificial Neural Network," Services Computing, IEEE Transactions on , vol.3, no.4, pp.338-352, Oct.-Dec. 2010 doi: 10.1109/TSC.2010.45

[67] http://www.gams.com/

[68] Tommaso Cucinotta, Fabio Checconi, George Kousiouris, Dimosthenis Kyriazis, Theodora Varvarigou, Alessandro Mazzetti, Zlatko Zlatev, Jury Papay, Michael Boniface, SA¶ren Berger, Dominik Lamp, Thomas Voith, Manuel Stein, "Virtualised e-Learning with Real-Time Guarantees on the IRMOS Platform ,", in Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2010), Perth, Australia, December 2010.

[69] George Kousiouris, Dimosthenis Kyriazis, Spyridon V. Gogouvitis, Andreas Menychtas, Kleopatra Konstanteli and Theodora A. Varvarigou, "Translation of Application-level Terms to Resource-level attributes across the Cloud Stack Layers ,", in Workshop on Management of Cloud Systems (MoCS 2011), in conjunction with the 16th IEEE Symposium on Computers and Communications (ISCC 2011).

[70] http://www.javaworld.com/javaworld/jw-12-2000/jw-1229-traps.html?page=4

[71] http://www.mathworks.com/help/techdoc/ref/bench.html

[72] http://www.spec.org/benchmarks.html

[73] http://view.eecs.berkeley.edu/wiki/Dwarfs

[74] Ripal Nathuji, Aman Kansal, and Alireza Ghaffarkhah. 2010. Q-clouds: managing performance interference effects for QoS-aware clouds. In Proceedings of the 5th European

conference on Computer systems (EuroSys '10). ACM, New York, NY, USA, 237-250. DOI=10.1145/1755913.1755938 http://doi.acm.org/10.1145/1755913.1755938

[75] J. Happe, D. Westermann, K. Sachs, and L. Kapova, “Statistical inference of software performance models for parametric performance completions,” in Lecture Notes in Computer Science, 2010, Volume 6093/2010, 20-35, DOI: 10.1007/978-3-642-13821-8_4

[76] http://www.vmware.com/

[77] www.xen.org/

[78] http://www.linux-kvm.org/

[79] Padala, X. Zhu, Z. Wanf, S. Singhal, and K. Shin, “Performance evaluation of virtualization technologies for server consolidation,” HP Labs, Tech. Rep. HPL-2007-59, 2007

[80] V. Makhija et al. VMmark: A scalable benchmark for virtualized systems. Technical report, VMWare, 2006.

[81] Y Koh, R. C. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu, An analysis of performance interference effects in virtual environments," in ISPASS. IEEE Computer Society, 2007, pp. 200-209.

[82] Ahmed A. Soror, Umar Farooq Minhas, Ashraf Aboulnaga, Kenneth Salem, Peter Kokosielis, and Sunil Kamath. 2008. Automatic virtual machine configuration for database workloads. ACM Trans. Database Syst. 35, 1, Article 7 (February 2008), 47 pages. DOI=10.1145/1670243.1670250 http://doi.acm.org/10.1145/1670243.1670250

[83] K.T. Moller, Virtual Machine Benchmarking, Diploma Thesis, Universitat Karlsruhe (TH), 2007.

[84] O. Tickoo, R. Iyer, R. Illikkal, and D. Newell, "Modeling virtual machine performance," ACM SIGMETRICS Performance Evaluation Review, vol. 37, 2010, p. 55

[85] Hai Jin, Wenzhi Cao, Pingpeng Yuan, Xia Xie.:VSCBenchmark: benchmark for dynamic server performance of virtualization technology . In: Proceedings of the 1st international forum on Next-generation multicore/manycore technologies (2008), Cairo, Egypt.

[86] Weng, C.; Wang, Z.; Li, M. & Lu, X. The hybrid scheduling framework for virtual machine systems VEE '09: Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, ACM, 2009, 111-120

[87] Antonio García-Guirado, Ricardo Fernández-Pascual, José M. García, Virtual-GEMS: An Infrastructure To Simulate Virtual Machines .fifth annual workshop on modeling,benchmarking and simulation MoBS 2009 held in conjunction with the 36th annual international symposium on computer architecture Sunday,june 21,2009

[88] El-Refaey, M.A.; Rizkaa, M.A.; , "CloudGauge: A Dynamic Cloud and Virtualization Benchmarking Suite," Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), 2010 19th IEEE International Workshop on , vol., no., pp.66-75, 28-30 June 2010, doi: 10.1109/WETICE.2010.17

[89] A. Tikotekar, G. Vall´ee, T. Naughton, H. Ong, C. Engelmann, S. L. Scott, and A. M. Filippi, “Effects of virtualization on a scientific application running a hyperspectral radiative transfer code on virtual machines,” in Proc. of HPCVirt’08. USA: ACM, 2008, pp. 16–23.

[90] White, J. and Pilbeam, A., “A Survey of Virtualization Technologies With Performance Testing”, ArXiv e-prints, 1010.3233, October 2010

[91] IRMOS Project D5.1.1 “Models of real time applications on service oriented infrastructures”, It-Innovation and other partners, February 2009

[92] http://www.mathworks.com/matlabcentral/fileexchange/11984

[93] http://www.irmosproject.eu/Postproduction.aspx

[94] C. L. Liu and J. Layland. Scheduling alghorithms for multiprogramming in a hard real-time environment.

Journal of the ACM, 20(1), 1973.

[95] Tommaso Cucinotta, Dhaval Giani, Dario Faggioli and Fabio Checconi, "Providing Performance Guarantees to Virtual Machines using Real-Time Scheduling," in Proceedings of the 5th Workshop on Virtualization and High-Performance Cloud Computing (VHPC 2010), Ischia (Naples), Italy, August 2010

[96] Tommaso Cucinotta, Gaetano Anastasi, Luca Abeni, "Respecting temporal constraints in virtualised services," in Proceedings of the 2nd IEEE International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA 2009), Seattle, Washington, July 2009

[97] Khalid, O.; Maljevic, I.; Anthony, R.; Petridis, M.; Parrott, K.; Schulz, M.; , "Dynamic Scheduling of Virtual Machines Running HPC Workloads in Scientific Grids," New Technologies, Mobility and Security (NTMS), 2009 3rd International Conference on , vol., no., pp.1-5, 20-23 Dec. 2009 doi: 10.1109/NTMS.2009.5384725

[98] Emeakaroha, Vincent C.; Brandic, Ivona; Maurer, Michael; Dustdar, Schahram; , "Low level Metrics to High level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments," High Performance Computing and Simulation (HPCS), 2010 International Conference on , vol., no., pp.48-54, June 28 2010-July 2 2010

[99] Rui Zhang and Alan J. Bivens. 2007. Comparing the use of bayesian networks and neural networks in response time modeling for service-oriented systems. In Proceedings of the 2007 workshop on Service-oriented computing performance: aspects, issues, and approaches (SOCP '07). ACM, New York, NY, USA, 67-74. DOI=10.1145/1272457.1272467 http://doi.acm.org/10.1145/1272457.1272467

[100] Georgina Gallizo, Roland Kübert, Karsten Oberle, Klaus Satzke, Spyridon V. Gogouvitis, Gregory Katsaros, and Eduardo Oliveros: A service level agreement management framework for real-time applications in cloud computing environments. Proceedings of the 2nd International ICST Conference on Cloud Computing, Barcelona, Spain October 26 – 28, 2010

[101] Alex Guazzelli, Kostantinos Stathatos, and Michael Zeller. 2009. Efficient deployment of predictive analytics through open standards and cloud computing. SIGKDD Explor. Newsl. 11, 1 (November 2009), 32-38. DOI=10.1145/1656274.1656281 http://doi.acm.org/10.1145/1656274.1656281

[102] Jia Rao, Xiangping Bu, Cheng-Zhong Xu, Leyi Wang, George Yin, “VCONF: A Reinforcement Learning Approach to Virtual Machines Auto-configuration”, 6th international conference on Autonomic computing, 2009

[103] Gerald Tesauro, Nicholas K. Jong, Rajarshi Das, Mohamed N. Bennani, “On the use of hybrid reinforcement learning for autonomic resource allocation”, Springer Cluster Computing, 2007

[104] Giovanni Toffetti, Alessio Gambi, Mauro Pezzè and Cesare Pautasso, “Engineering Autonomic Controllers for Virtualized Web Applications”, 10th international Conference on Web engineering, 2010

[105] Jiangtian Li, Xiaosong Ma, Karan Singh, Martin Schulz, Bronis R. de Supinski and Sally A. McKee, “Machine Learning Based Online Performance Prediction for Runtime Parallelization and Task Scheduling”, IEEE International Symposium on Performance Analysis of Systems and Software, 2009

[106] Philipp Leitner, Branimir Wetzstein, Florian Rosenberg, Anton Michlmayr, Schahram Dustdar, and Frank Leymann, “Runtime Prediction of Service Level Agreement Violations for Composite Services”, ServiceWave 2009

[107] Juan M. Tirado, Daniel Higuero, Florin Isaila, Jes´us Carretero, “Predictive Data Grouping and Placement for Cloud-based Elastic Server Infrastructures”, 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2011

[108] Sara Casolari, Mauro Andreolini, Michele Colajanni, "Runtime prediction models for Web-based system resources", IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008

[109] http://www.mathworks.com/help/toolbox/nnet/ug/bss36dv-1.html

[110] Maria P. Menezes, Jr. and Guilherme A. Barreto. 2008. Long-term time series prediction with the NARX network: An empirical evaluation. Neurocomput. 71, 16-18 (October 2008), 3335-3343. DOI=10.1016/j.neucom.2008.01.030

[111] http://www.mediawiki.org/wiki/MediaWiki

[112] http://www.flexiscale.com/

[113] http://www.stevengould.org/software/openforecast/index.shtml

[114] George Kousiouris,George Vafiadis and Theodora Varvarigou, "A Front-end Hadoop based Data Management Service for Efficient Federated Clouds", to appear in 3rd IEEE

International Conference on Cloud Computing Technology and Science (IEEE CloudCom 2011), Athens, Greece, Nov 29- Dec 1, 2011

[115] Kleopatra Konstanteli, Tommaso Cucinotta, and Theodora Varvarigou. “Optimum allocation of distributed service workflows with probabilistic real-time guarantees”. Service Oriented Computing and Applications (Springer), 4:68:229-68:243, December 2010

[116] Dimosthenis Kyriazis, Ralf Einhorn, Lars Furst, Michael Braitmaier, Dominik Lamp, Kleopatra Konstanteli, George Kousiouris, Andreas Menychtas, Eduardo Oliveros, Neil Loughran, Bassem Nasser, "A METHODOLOGY FOR ENGINEERING REAL-TIME INTERACTIVE MULTIMEDIA APPLICATIONS ON SERVICE ORIENTED INFRASTRUCTURES ,", in Proceedings of the IADIS International Conference Applied Computing, Timisoara, Romania 14-16 October 2010.

[117] Luís Costa, “Modeling Interactive Real-time Applications on Service Oriented Infrastructures”, Presentation during 3rd IRMOS Public Seminar, available at: http://irmosproject.eu/Files/02_IRMOS_Oslo_Seminar_Modelling-PartI.pdf

Βιογραφικά Στοιχεία του Διδάκτορα

Ο Γεώργιος Τ. Κουσιουρής γεννήθηκε την 23η Ιουλίου 1980 στην Αθήνα. Αποφοίτησε από το Εκπαιδευτήρια «Ελληνική Παιδεία» το 1998 με βαθμό απολυτηρίου 19.1 («Άριστα»). Την ίδια χρονιά (1998) εισήχθη στο τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών του Πανεπιστημίου Πατρών. Αποφοίτησε από το τελευταίο το 2005 («Λίαν Καλώς») και το θέμα της διπλωματικής εργασίας του ήταν «Ανάπτυξη Υποσυστήματος Δρομολόγησης Πακέτων IP». Η εργασία ολοκληρώθηκε υπό την επίβλεψη του Καθηγητή κ. Δημήτριου Σερπάνου και βαθμολογήθηκε με γενικό χαρακτηρισμό «Άριστα» (10). Επίσης έγινε δεκτή και παρουσιάστηκε στο διεθνές συνέδριο IEEE ISSPIT 2005 με τον τίτλο "IP look-up with time or memory guarantee and low update time".

Τον Δεκέμβριο του 2006 έγινε δεκτός για μεταπτυχιακές σπουδές που οδηγούν στην απόκτηση Διδακτορικού Διπλώματος στη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου. Έλαβε τον τίτλο του διδάκτορα τον Ιούνιο του 2012.

Κατά το παρελθόν εργάστηκε στον ιδιωτικό τομέα (ΟΤΕ) καθώς επίσης και ως Εργαστηριακός Συνεργάτης στην Α.Σ.ΠΑΙ.Τ.Ε. Κατά τη διάρκεια της εκπόνησης της Διδακτορικής του Διατριβής ο Γ. Κουσιουρής εργάστηκε σε Ελληνικά και Ευρωπαϊκά Προγράμματα, στα οποία του δόθηκε η δυνατότητα να εμβαθύνει σε ερευνητικά θέματα, άμεσα συνδεδεμένα με την περιοχή του διδακτορικού του. Έλαβε μέρος σε πολλές επιστημονικές συναντήσεις στην Ελλάδα και στο εξωτερικό και συνεργάστηκε με μηχανικούς και ανώτερα στελέχη διαφόρων οργανισμών, εταιριών υπολογιστικών συστημάτων και ερευνητικών πανεπιστημιακών ομάδων. Έχει εκτελέσει σειρά από παρουσιάσεις και σεμινάρια για ζητήματα πρόβλεψης απόδοσης σε περιβάλλοντα Υπολογιστικών Νεφών. Τα αποτελέσματα της ερευνητικής του εργασίας παρουσιάστηκαν σε διεθνή συνέδρια και δημοσιεύθηκαν στον επιστημονικό Τύπο, σε περιοδικά και βιβλία. Ο Γ. Κουσιουρής είναι μέλος του Τεχνικού Επιμελητηρίου της Ελλάδος (ΤΕΕ) από το 2006.