ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Τεχνικές Αναλυσης και Παρακολούθησης Ορθής Λειτουργίας Πληροφοριακών Συστημάτων Πολλαπλών Επιπέδων

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

ΓΕΩΡΓΙΟΥ Κ. ΧΑΤΖΗΚΩΝΣΤΑΝΤΙΝΟΥ

Διπλωματούχου Ηλεκτρολόγου Μηχανικού και
Μηχανικού Υπολογιστών Ε.Μ.Π. (2008)

Επιβλέπων : Κωνσταντίνος Κοντογιάννης
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2015

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Τεχνικές Αναλυσης και Παρακολούθησης Ορθής Λειτουργίας Πληροφοριακών Συστημάτων Πολλαπλών Επιπέδων

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

ΓΕΩΡΓΙΟΥ Κ. ΧΑΤΖΗΚΩΝΣΤΑΝΤΙΝΟΥ

Διπλωματούχου Ηλεκτρολόγου Μηχανικού και Μηχανικού Υπολογιστών Ε.Μ.Π. (2008)

Επιβλέπων : Κωνσταντίνος Κοντογιάννης
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Η παρούσα έρευνα έχει συγχρηματοδοτηθεί από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο - ΕΚΤ) και από εθνικούς πόρους μέσω του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» του Εθνικού Στρατηγικού Πλαισίου Αναφοράς (ΕΣΠΑ) – Ερευνητικό Χρηματοδοτούμενο Έργο: Ηράκλειτος ΙΙ. Επένδυση στην κοινωνία της γνώσης μέσω του Ευρωπαϊκού Κοινωνικού Ταμείου.

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Τεχνικές Αναλυσης και Παρακολούθησης Ορθής Λειτουργίας Πληροφοριακών Συστημάτων Πολλαπλών Επιπέδων

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

ΓΕΩΡΓΙΟΥ ΧΑΤΖΗΚΩΝΣΤΑΝΤΙΝΟΥ

Διπλωματούχου Ηλεκτρολόγου Μηχανικού και
Μηχανικού Υπολογιστών Ε.Μ.Π. (2008)

Συμβουλευτική επιτροπή: Κωνσταντίνος Κοντογιάννης
Ιωάννης Βασιλείου
Τιμολέων Σελλής

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 30η Οκτωβρίου 2015.

.....	.....	.....
Κ. Κοντογιάννης	Ι. Βασιλείου	Τ. Σελλής
Αν. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής R.M.I.T.
.....	.....	.....
Ν. Παπασπύρου	Δ. Ασκούνης	Ανδρ.-Γ. Σταφυλοπάτης
Αν. Καθηγητής Ε.Μ.Π.	Αν. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

.....
Σ. Χριστοδουλάκης
Ομ. Καθηγητής Πολυτεχνείου Κρήτης

Αθήνα, Οκτώβριος 2015

ΓΕΩΡΓΙΟΣ Κ. ΧΑΤΖΗΚΩΝΣΤΑΝΤΙΝΟΥ

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

Στην 'Αννα

Περίληψη

Αντικείμενο της παρούσας διδακτορικής διατριβής είναι ο σχεδιασμός και η ανάπτυξη ενός περιβάλλοντος-πλαισίου που υποστηρίζει διαδικασίες συμπερασμού σε μοντέλα δέντρων στόχων. Τα μοντέλα δέντρων στόχων χρησιμοποιούνται για την απεικόνιση απαιτήσεων συστήματος, πολιτικών και στόχων που θέτουν οι εμπλεκόμενοι στο σύστημα. Οι διαδικασίες συμπερασμού χρησιμοποιούνται για α) την επαλήθευση συστημάτων λογισμικού κατά τον χρόνο εκτέλεσης, και β) την συγκρότηση πλάνων αποκατάστασης στην περίπτωση που ένας ή περισσότεροι από τους στόχους δεν ικανοποιούνται.

Για την επαλήθευση συστημάτων λογισμικού κατά το χρόνο εκτέλεσης, προτείνεται μια διαδικασία συμπερασμού από τα φύλλα προς τις ρίζες (bottom-up), για ασαφή μοντέλα δέντρων στόχων. Πιο συγκεκριμένα, οι εμπλεκόμενοι του συστήματος ορίζουν σύνολα στόχων οι οποίοι πρέπει να επαληθεύονται κατά τον χρόνο εκτέλεσης. Οι μοντελοποίηση αυτόν των στόχων γίνεται με την χρήση ενός μεταμοντέλου που βασίζεται στα AND/OR μοντέλα δέντρων στόχων. Στιγμιότυπα αυτού του μεταμοντέλου είναι δυνατόν να μετασχηματιστούν σε σταθμισμένους ασαφείς κανόνες, που στη συνέχεια θα αποτελέσουν την βάση γνώσης ενός ασαφούς ελεγκτή. Επιπλέον, το σύνολο των παρατηρήσεων που μπορούν να εξαχθούν από το σύστημα μετατρέπονται σε ασαφή δεδομένα (fuzzy facts) μέσω μίας κατάλληλης διαδικασίας ασαφοποίησης, και χρησιμοποιώντας μία κατάλληλη μηχανή συμπερασμού είναι εφικτό να υπολογιστεί ο βαθμός ικανοποίησης των αρχικών στόχων που έχουν θέσει οι stakeholders του συστήματος.

Για την δημιουργία των πλάνων αποκατάστασης, προτείνεται ένας μηχανισμός συμπερασμού από τις ρίζες του μοντέλου προς τα φύλλα (top-down), που χρησιμοποιεί SAT solvers και γενετικούς αλγορίθμους. Πιο συγκεκριμένα, χρησιμοποιούνται μοντέλα που επιτρέπουν τον ορισμό χρονικών εξαρτήσεων ανάμεσα στους κόμβους που συμβολίζουν ενέργειες (actions). Στην περίπτωση που οι στόχοι του συστήματος δεν επαληθεύονται, είναι δυνατόν τα μοντέλα αυτά να χρησιμοποιηθούν σε συνδυασμό με κατάλληλους μηχανισμούς συμπερασμού, έτσι ώστε να προταθεί ένα πλάνο εκτέλεσης για την αλλαγή παραμέτρων του συστήματος, οι οποίες αλλαγές μπορούν να έχουν θετική επίδραση στους στόχους του συστήματος.

Λέξεις Κλειδιά : Μοντέλα δένδρων στόχων, Ασαφής μηχανισμός συμπερασμού , Επαλήθευση ορθής λειτουργίας χρόνου εκτέλεσης

Abstract

This thesis deals with the design and development of goal model reasoning frameworks. The goal models aim to denote system requirements, system policies, and stakeholder objectives. The reasoning frameworks aim to a) assist on run-time system verification, and b) assist on the formation of remedial plans when one or more requirements are failed to be delivered.

For the run-time system verification, the thesis proposes a bottom-up reasoning framework that utilizes fuzzy goal models proposed as an extension of standard goal models. More specifically, the stakeholders of a software system define sets of goals that the software system under review should comply with. This can be achieved by designing instances of a meta-model proposed as an extension of AND/OR goal trees with well defined formal semantics. These instances (i.e. models) can then be transformed to a set of weighted fuzzy logic programs that constitute the knowledge base of a fuzzy controller. Additionally, values for all observable characteristics of the system are collected by actively monitoring the system, and are transformed to fuzzy facts through a proper fuzzification process. Finally, a fuzzy inference engine is utilized in order to evaluate the degree of satisfaction for the various goals posed by the stakeholders, using run-time system information.

Similarly, for the formation of remedial plans, the thesis proposes a top-down reasoning framework that utilizes SAT solvers and genetic algorithms. More specifically, the thesis proposes temporal dependencies between nodes as extensions to standard goal models, in order to represent sequences of possible remedial actions. The reasoning strategy evaluates these models, in order to devise optimal remedial plans when the system fails to comply with, or deliver its requirements. The objective is for these remedial plans to have a positive impact towards reconfiguring the system to a state that can either satisfy its requirements, or *satisfice* them within an acceptable degree.

Finally, the thesis reports on evaluation experiments on the performance, scalability, and stability of the proposed reasoning frameworks, and identifies areas of future research.

Keywords : Goal models, Fuzzy reasoning, Run-time verification

Ευχαριστίες

Ευχαριστώ τον επιβλέποντα καθηγητή μου κ. Κωνσταντίνο Κοντογιάννη για την πολύτιμη βοήθεια, καθοδήγηση και εξαίρετη συνεργασία του σε όλα τα στάδια εκπόνησης της διατριβής. Ευχαριστώ επίσης τα μέλη της τριμελούς συμβουλευτικής επιτροπής καθηγητές κ. Ιωάννη Βασιλείου και κ. Τιμολέοντα Σελλή, καθώς και τα μέλη της επταμελούς εξεταστικής επιτροπής καθηγητές κ. Νικόλαο Παπασπύρου, κ. Δημήτριο Ασκούνη, κ. Ανδρέα-Γεώργιο Σταφυλοπάτη και τον ομότιμο καθηγητή του Πολυτεχνείου Κρήτης κ. Σταύρο Χριστοδουλάκη για το ενδιαφέρον τους και τη συμμετοχή τους στην αξιολόγηση της παρούσας διατριβής.

Ευχαριστώ, επίσης, τον Δρ. Αλέξανδρο Χορταρά για την παραχώρηση της υλοποίησής του για τα σταθμισμένα ασαφή λογικά προγράμματα, τον υποψήφιο διδάκτορα Μιχάλη Αθανασόπουλο για την εξαιρετική συνεργασία που είχαμε και τον υποψήφιο διδάκτορα Θεόδωρο Καλαματιανό για τις πάντα χρήσιμες ανταλλαγές ιδεών.

Τέλος, αλλά και πάνω από όλα, ευχαριστώ τους γονείς μου και τον αδερφό μου για την συμπαράστασή τους όλα αυτά τα χρόνια, καθώς και την Άννα που ήταν δίπλα μου σε όλες τις δυσκολίες των τελευταίων ετών.

Γιώργος Χατζηκωνσταντίνου
Αθήνα, Οκτώβριος 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Η παρούσα έρευνα έχει συγχρηματοδοτηθεί από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο - ΕΚΤ) και από εθνικούς πόρους μέσω του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» του Εθνικού Στρατηγικού Πλαισίου Αναφοράς (ΕΣΠΑ) – Ερευνητικό Χρηματοδοτούμενο Έργο: Ηράκλειτος ΙΙ. Επένδυση στην κοινωνία της γνώσης μέσω του Ευρωπαϊκού Κοινωνικού Ταμείου.

Εκτεταμένη Περίληψη

Τα σύγχρονα συστήματα λογισμικού, όπως για παράδειγμα τα συστήματα που χρησιμοποιούν τεχνολογίες εικονικοποίησης (virtualization), συχνά μεταβάλλονται δυναμικά ώστε να ικανοποιήσουν ένα μεγάλο σύνολο από λειτουργικές και μη-λειτουργικές απαιτήσεις καθώς οι υπολογιστικές ανάγκες αλλάζουν. Παραδείγματα τέτοιων απαιτήσεων αποτελούν ο ρυθμός εξυπηρέτησης (throughput), η ασφάλεια (security), η ιδιωτικότητα (privacy), η κατανάλωση ενέργειας (energy consumption) και η αποδοτική χρήση πόρων. Η δυναμική προσαρμογή ενός τέτοιου συστήματος μπορεί να παίρνει τη μορφή προσθήκης νέων εικονικών πόρων, μεταφοράς διεργασιών σε διαφορετικά εικονικά μηχανήματα, όπως και μεταφοράς, διαχωρισμού ή διανομής των δεδομένων και της λογικής υπολογισμών σε διαφορετικούς εξυπηρετητές. Παρ' όλ' αυτά, ένα κοινό πρόβλημα που προκύπτει ως συνέπεια της δυναμικής προσαρμογής ενός συστήματος μέσω των παραπάνω αλλαγών είναι η πιθανή παραβίαση μίας ή περισσοτέρων απαιτήσεων.

Για παράδειγμα, ας θεωρήσουμε ένα σύστημα που πραγματοποιεί ανάλυση σε μεγάλο όγκο δεδομένων. Είναι προφανές ότι ένα τέτοιο σύστημα θα πρέπει να πληροί συγκεκριμένους στόχους απόδοσης και να μπορεί να ανταπεξέλθει σε δεδομένα με υψηλή μεταβλητότητα και ποικιλομορφία. Ο σχεδιασμός και η υλοποίηση ενός τέτοιου συστήματος πιθανότατα θα απαιτεί τη χρήση πολλών μονάδων υπολογισμού καθώς και το δυναμικό διαχωρισμό και διανομή των δεδομένων και των υπολογισμών, ώστε να πληροί συγκεκριμένους στόχους απόδοσης. Επιπλέον, θα απαιτεί και την αποδοτική χρήση των μέσων αποθήκευσης δεδομένων, έτσι ώστε να μεγιστοποιεί το ρυθμό διακίνησης δεδομένων και να ελαχιστοποιεί την κατανάλωση ενέργειας.

Σε ένα τέτοιο σύστημα, όταν δεδομένα ή υπολογιστικές μονάδες μεταφέρονται σε διάφορους εξυπηρετητές ή όταν νέα μηχανήματα προστίθενται δυναμικά ή προκύπτουν νέες ροές δεδομένων, είναι δυνατόν να επηρεαστούν μία ή περισσότερες απαιτήσεις. Όταν για παράδειγμα προκύπτουν νέες ροές δεδομένων, μπορεί να αποτύχει ο στόχος απόκρυψης δεδομένων ή όταν τα δεδομένα μεταφέρονται ή διαχωρίζονται μεταξύ δυναμικά προστιθέμενων μηχανημάτων προκειμένου να πληρούν τις απαιτήσεις απόδοσης, ο στόχος ιδιωτικότητας μπορεί να αποτύχει. Επομένως, είναι σημαντικό για τους αναλυτές, τους μηχανικούς λογισμικού και τους διαχειριστές των συστημάτων να υπάρχουν μέθοδοι,

τεχνικές και εργαλεία για την αξιολόγηση του κατά πόσο αλλαγές στη δομή του συστήματος, στο περιβάλλον στο οποίο λειτουργεί το σύστημα και στους πόρους μπορούν να οδηγήσουν στην παραβίαση απαιτήσεων και πολιτικών που πρέπει να ισχύουν στο σύστημα

Σαν απάντηση στο παραπάνω πρόβλημα, έχουν εξεταστεί μοντέλα, περιβάλλοντα πλαίσια και εργαλεία που επιτρέπουν τον προσδιορισμό, την ανάλυση και την αξιολόγηση των απαιτήσεων και στόχων. Σε αυτό το πλαίσιο, η μέχρι τώρα έρευνα αφορά κυρίως στη στατική ανάλυση και στην εφαρμογή διαδικασιών συμπερασμού σε τέτοιου είδους μοντέλα με σκοπό να αξιολογήσει τη συνοχή, την πληρότητα και την εγκυρότητα των απαιτήσεων συστήματος έναντι των στόχων των εμπλεκομένων. Παρ' όλ' αυτά το πρόβλημα της διαχείρισης και της εξαγωγής συμπερασμάτων σε μεγάλα μοντέλα απαιτήσεων, ιδιαίτερα όταν οι εξαρτήσεις ανάμεσα στις σχεδιαστικές αποφάσεις ή τις απαιτήσεις δεν μπορούν να μοντελοποιηθούν πλήρως και ντετερμινιστικά, λόγω της μεγάλης πολυπλοκότητας αυτών των συστημάτων, δεν έχει διερευνηθεί επαρκώς. Συνεπώς, είναι σημαντικό να ερευνήσουμε τεχνικές ανάλυσης οι οποίες μπορούν να εφαρμοστούν στο χρόνο εκτέλεσης και σε μεγάλα μοντέλα απαιτήσεων, ιδιαιτέρως όταν η γνώση που έχουμε στη διάθεσή μας είναι μη πλήρης και ασαφής.

Αντικείμενο της παρούσας διδακτορικής διατριβής είναι η ανάπτυξη τεχνικών για την ανάλυση και την παρακολούθηση ορθής λειτουργίας πληροφοριακών συστημάτων πολλαπλών επιπέδων. Ειδικότερα, στα πλαίσια της παρούσας έρευνας α) προτείνονται τεχνικές μοντελοποίησης για την περιγραφή μοντέλων πολιτικών και περιορισμών λειτουργίας με τη χρήση δέντρων στόχων (goal models), β) εφαρμόζονται τεχνικές πιθανοτικού συμπερασμού (probabilistic inference) και ασαφούς συμπερασμού (fuzzy reasoning) για την επαλήθευση της λειτουργίας του συστήματος σε σχέση με τα μοντέλα δέντρων στόχων και γ) προτείνεται η χρήση ενός γενετικού αλγορίθμου για τον καθορισμό πλάνων αποκατάστασης, η εκτέλεση των οποίων μπορεί να έχει μία συνολικά θετική επίδραση στην ικανοποίηση των απαιτήσεων στο σύστημα που βρίσκεται σε λειτουργία.

Πιο συγκεκριμένα, στα πλαίσια αυτής της εργασίας, προτείνουμε την χρήση ασαφών μοντέλων στόχων (fuzzy goal models) ως έναν τρόπο για την μοντελοποίηση ελλιπούς γνώσης σχετικά με τις απαιτήσεις λογισμικού και τις αλληλεξαρτήσεις τους, και εισάγουμε ένα περιβάλλον πλαίσιο συμπερασμού για ασαφή μοντέλα στόχων που μπορεί να χρησιμοποιηθεί για να αναλύσει και να αξιολογήσει τις απαιτήσεις του συστήματος κατά το χρόνο εκτέλεσης, λαμβάνοντας ως είσοδο δεδομένα που συλλέγονται καθώς το σύστημα λειτουργεί ή μεταβάλλεται.

Τα ασαφή μοντέλα στόχων μοντελοποιούν τις εξαρτήσεις που υπάρχουν μεταξύ των διαφόρων απαιτήσεων καθώς και τις σχέσεις που υπάρχουν μεταξύ των δεδομένων που καταγράφονται καθώς το σύστημα λειτουργεί και τις απαιτήσεις του συστήματος. Δεδομένου ότι αυτά τα μοντέλα μπορεί να αυξηθούν σε μέγεθος καθώς περισσότερα συστήματα συνδέονται μεταξύ τους, προτείνουμε επίσης μια τεχνική που επιτρέπει την ανάλυση των εξαρτήσεων στα μοντέλα στόχων έτσι ώστε να μπορούν να προσδιοριστούν ανεξάρτητες περιοχές (ή υπο-γραφήματα) προκειμένου να μπορεί να παραλληλοποιηθεί η ασαφής συλλογιστική.

Δοθέντος ενός μοντέλου στόχων με fuzzy και crisp κόμβους, εισάγουμε αρχικά μια μέθοδο που επιτρέπει την παραγωγή ασαφών κανόνων από τα ασαφή μοντέλα στόχων και εν συνεχεία, μια διαδικασία μετασχηματισμού μοντέλων που επιτρέπει τη δημιουργία ενός γραφήματος εξαρτήσεων, όπου κάθε κόμβος είναι μια συλλογιστική μονάδα των υπολογισμών που πρέπει να πραγματοποιηθούν, προκειμένου να υπολογιστεί ο βαθμός ικανοποίησης ενός κόμβου.

Ως εκ τούτου, καθώς τα γεγονότα συλλέγονται από το σύστημα που βρίσκεται σε λειτουργία, το πλάνο συμπερασμού (reasoning plan) και οι κανόνες που έχουν εξαχθεί από το μοντέλο κατά τον χρόνο σχεδίασης, μπορούν να χρησιμοποιηθούν προκειμένου να εξαχθούν συμπεράσματα σχετικά με το κατά πόσο συγκεκριμένοι στόχοι συνεχίζουν να ισχύουν καθώς το σύστημα μεταβάλλεται.

Σε περίπτωση που κάποιος από τους στόχους δεν επιτυγχάνεται, ή ο βαθμός ικανοποίησης του είναι μικρότερος από ένα συγκεκριμένο όριο, έχουμε εισαγάγει μια μέθοδο που μπορεί να χρησιμοποιηθεί προκειμένου να δημιουργηθεί ένα πλάνο αποκατάστασης, η εκτέλεση του οποίου μπορεί να οδηγήσει το σύστημα σε μια "καλύτερη" κατάσταση, δηλαδή μια κατάσταση στην οποία περισσότεροι στόχοι επιτυγχάνονται, ή σε περίπτωση που έχουν εκχωρηθεί βάρη στους στόχους που υποδηλώνουν τη σημασία τους, σε μια κατάσταση όπου επιτυγχάνονται οι πιο σημαντικοί στόχοι του συστήματος.

Η αξιολόγηση του προτεινόμενου περιβάλλοντος πλαισίου έγινε με την εκτέλεση μίας σειράς πειραμάτων με τυχαία μοντέλα ποικίλου μεγέθους και πολυπλοκότητας. Χρησιμοποιώντας αυτά τα μοντέλα αξιολογήσαμε την εφαρμογή της προτεινόμενης μεθόδου σε σχέση με το χρόνο εκτέλεσης και το μέγεθος της μνήμης που απαιτείται για μοντέλα διαφορετικών μεγεθών.

Τα πειραματικά αποτελέσματα δείχνουν ότι οι χρόνοι που απαιτούνται για να ολοκληρωθεί η εξαγωγή συμπερασμάτων παραμένουν μικροί ακόμη και για μεγάλα

μοντέλα, και ως εκ τούτου το προτεινόμενο περιβάλλον πλαίσιο μπορεί να χρησιμοποιηθεί κατά το χρόνο εκτέλεσης.

Τέλος, όσον αφορά στη δημιουργία των πλάνων αποκατάστασης, αξιολογήσαμε και συγκρίναμε την ποιότητα των λύσεων που προκύπτουν από τον προτεινόμενο γενετικό αλγόριθμο. Τα αποτελέσματα δείχνουν ότι η προτεινόμενη προσέγγιση επιτρέπει την απόκτηση μιας λύσης που είναι εντός του 90% της βέλτιστης σε πολύ μικρότερο χρόνο από αυτόν που απαιτείται από έναν αλγόριθμο που θα εξέταζε εξαντλητικά όλα τα πιθανά πλάνα αποκατάστασης προκειμένου να εντοπίσει το βέλτιστο πλάνο μεταξύ αυτών.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Τα σύγχρονα συστήματα λογισμικού, όπως αυτά τα οποία χρησιμοποιούν τεχνολογίες εικονικοποίησης (virtualization), συχνά επαναδιαμορφώνονται δυναμικά ώστε να ικανοποιήσουν ένα μεγάλο σύνολο από λειτουργικές και μη-λειτουργικές απαιτήσεις καθώς οι υπολογιστικές ανάγκες αλλάζουν. Παραδείγματα τέτοιων απαιτήσεων μπορεί να είναι ο ρυθμός εξυπηρέτησης (throughput), η ασφάλεια (security), η ιδιωτικότητα (privacy), η κατανάλωση ενέργειας (energy consumption), η αποδοτική χρήση πόρων.

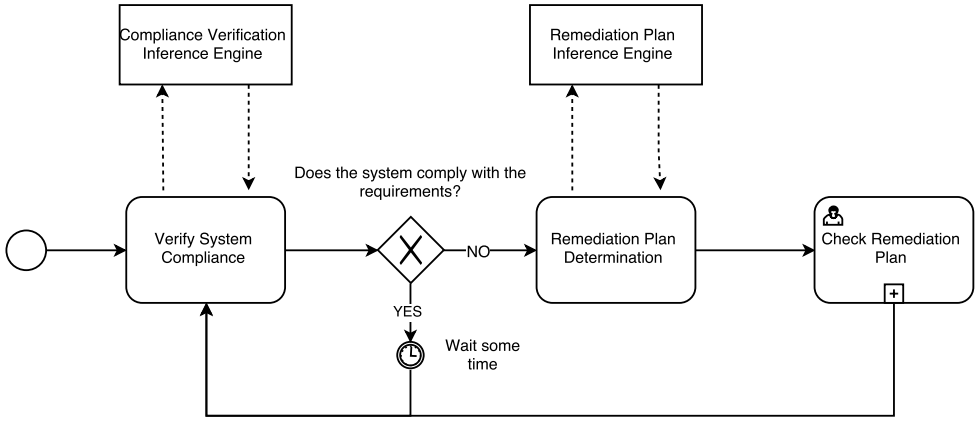
Η δυναμική προσαρμογή ενός συστήματος μπορεί να παίρνει τη μορφή προσθήκης νέων εικονικών πόρων, μεταφοράς διεργασιών σε διαφορετικά εικονικά μηχανήματα, όπως και μεταφορά, διαχωρισμός ή διανομή των δεδομένων και της λογικής υπολογισμών σε διαφορετικούς εξυπηρετητές. Ένα κοινό πρόβλημα που προκύπτει παρ΄ όλ΄ αυτά είναι η πιθανή παραβίαση μίας ή περισσοτέρων απαιτήσεων ως αποτέλεσμα αυτών των αλλαγών. Ως παράδειγμα, ας θεωρήσουμε ένα σύστημα που πραγματοποιεί ανάλυση σε μεγάλο όγκο δεδομένων. Είναι προφανές ότι ένα τέτοιο σύστημα θα πρέπει να πληροί συγκεκριμένους στόχους απόδοσης και να μπορεί να ανταπεξέλθει σε δεδομένα με υψηλή μεταβλητότητα και ποικιλομορφία. Ο σχεδιασμός και η υλοποίηση ενός τέτοιου συστήματος πιθανότατα θα απαιτεί τη χρήση πολλών μονάδων υπολογισμού, το δυναμικό διαχωρισμό και διανομή των δεδομένων και των υπολογισμών, ώστε να πληροί συγκεκριμένους στόχους απόδοσης και την αποδοτική χρήση των μέσων αποθήκευσης δεδομένων, έτσι ώστε να μεγιστοποιεί το ρυθμό διακίνησης δεδομένων και να ελαχιστοποιεί την κατανάλωση ενέργειας. Όταν δεδομένα ή υπολογιστικές μονάδες μεταφέρονται σε διάφορους εξυπηρετητές ή όταν νέα μηχανήματα προστίθενται δυναμικά ή προκύπτουν νέες ροές δεδομένων, είναι δυνατόν να επηρεαστούν μία ή περισσότερες απαιτήσεις. Για παράδειγμα, όταν προκύπτουν νέες ροές δεδομένων, μπορεί να αποτύχει ο στόχος απόκρυψης δεδομένων ή όταν τα δεδομένα μεταφέρονται ή διαχωρίζονται μεταξύ δυναμικά προστιθέμενων μηχανημάτων προκειμένου να πληρούν τις απαιτήσεις απόδοσης, ο στόχος ιδιωτικότητας μπορεί να αποτύχει. *Γι΄ αυτόν το λόγο είναι σημαντικό για*

τους αναλυτές, τους μηχανικούς λογισμικού και τους διαχειριστές των συστημάτων να μπορούν να αναλύσουν και να εξάγουν συμπεράσματα σχετικά με το αν συγκεκριμένοι στόχοι ή απαιτήσεις ισχύουν, όταν τέτοιου τύπου δυναμικές αλλαγές εφαρμόζονται στο σύστημα.

1.1 Κίνητρα

Κατά τη διάρκεια των τελευταίων ετών, έχει παρατηρηθεί μία μεγάλη αύξηση στην ανάπτυξη ισχυρά διασυνδεδεμένων συστημάτων τα οποία χρησιμοποιούνται σε ένα ευρύ φάσμα περιοχών όπως είναι ο τραπεζικός τομέας, το ηλεκτρονικό εμπόριο και η ηλεκτρονική διακυβέρνηση. Τα συστήματα αυτά περιλαμβάνουν πολύπλοκες απαιτήσεις οδηγώντας έτσι στην ύπαρξη πολύ μεγάλων μοντέλων [99]. Σε συνδυασμό με τις πολύπλοκες απαιτήσεις αυτών των συστημάτων, η ανάπτυξη των τεχνολογιών εικονοποίησης έχει καταστήσει πιθανή τη συνεχή εξέλιξη και τη δυναμική προσαρμογή των συστημάτων σε μία προσπάθεια να συνεχίσουν να ικανοποιούν το μεγάλο αριθμό λειτουργικών και μη λειτουργικών απαιτήσεων. Το πρόβλημα αυτό έχει αναγνωριστεί στη σχετική βιβλιογραφία ως ένα σημαντικό πρόβλημα [90], [43], από τη σκοπιά της επαλήθευσης απαιτήσεων καθώς και από τη σκοπιά της συμμόρφωσης με συγκεκριμένες πολιτικές. Έτσι, είναι σημαντικό να αναπτυχθούν μέθοδοι, τεχνικές και εργαλεία για την αξιολόγηση του κατά πόσο αλλαγές στη δομή του συστήματος, στο περιβάλλον στο οποίο λειτουργεί το σύστημα και στους πόρους μπορούν να οδηγήσουν στην παραβίαση απαιτήσεων και πολιτικών που έχουν τεθεί από τους εμπλεκομένους στο σύστημα.

Προς το παρόν, σαν απάντηση στο παραπάνω πρόβλημα, έχουν εξεταστεί μοντέλα, περιβάλλοντα πλαίσια και εργαλεία που επιτρέπουν τον προσδιορισμό, την ανάλυση και την αξιολόγηση των απαιτήσεων και στόχων. Σε αυτό το πλαίσιο, η μέχρι τώρα έρευνα αφορά κυρίως στη στατική ανάλυση και στην εφαρμογή διαδικασιών συμπερασμού σε τέτοιου είδους μοντέλα με σκοπό να αξιολογήσει τη συνοχή, την πληρότητα και την εγκυρότητα των απαιτήσεων συστήματος έναντι των στόχων των εμπλεκομένων [50], [11]. Όμως, καθώς οι επιχειρησιακές διαδικασίες γίνονται όλο και πιο πολύπλοκες απαιτώντας έτσι την ύπαρξη σύνθετων υπηρεσιών (π.χ. Systems-of-Systems), εφαρμογές που σχετίζονται με την επαλήθευση των ιδιοτήτων του συνολικού συστήματος κατά το χρόνο εκτέλεσης αποδεικνύεται ότι είναι ιδιαίτερα χρονοβόρες, σημαντικές και ενδιαφέρουσες [63] [22], [24]. Η έρευνα που έχει γίνει στην περιοχή των προσαρμοζόμενων συστημάτων επιχειρεί να αντιμετωπίσει κάποια από αυτά τα ζητήματα και έχει αρχίσει να υπάρχει ενδιαφέρον για τη μοντελοποίηση και την ανάλυση της συμπεριφοράς αυτών των συστημάτων κατά το χρόνο εκτέλεσης [86], [46]. Παρ΄ όλ΄ αυτά το πρόβλημα της διαχείρισης και της εξαγωγής συμπερασμάτων σε μεγάλα μοντέλα απαιτήσεων, ιδιαίτερα



Σχήμα 1-1: Γραφική απεικόνιση του υπό εξέταση προβλήματος

όταν οι εξαρτήσεις ανάμεσα στις σχεδιαστικές αποφάσεις ή τις απαιτήσεις δεν μπορούν να μοντελοποιηθούν πλήρως και ντετερμινιστικά, λόγω της μεγάλης πολυπλοκότητας αυτών των συστημάτων, δεν έχει διερευνηθεί επαρκώς. Συνεπώς, είναι σημαντικό να ερευνήσουμε τεχνικές ανάλυσης οι οποίες μπορούν να εφαρμοστούν στο χρόνο εκτέλεσης και σε μεγάλα μοντέλα απαιτήσεων, ιδιαιτέρως όταν η γνώση που έχουμε στη διάθεσή μας είναι μη πλήρης και ασαφής.

Στην παρούσα εργασία χρησιμοποιούμε ως βάση έννοιες και θεωρίες από το πεδίο των Μοντέλων Στόχων και ερευνούμε τη δυνατότητα χρήσης σταθμισμένων εξαρτήσεων και ασαφών ελεγκτών για την αξιολόγηση της επίδρασης που μπορεί να έχουν πιθανές μεταβολές του συστήματος στην ικανοποίηση των απαιτήσεων. Τέτοιες τεχνικές ανάλυσης που χρησιμοποιούν δεδομένα από τη λειτουργία του συστήματος θα μπορούσαν να συνεισφέρουν στην ανάπτυξη αυτόνομων και αυτοπροσαρμοζόμενων συστημάτων λογισμικού.

1.2 Περιγραφή και Ορισμός του Προβλήματος

Στην παρούσα διατριβή, εξετάζουμε το πρόβλημα επαλήθευσης ορθής λειτουργίας ενός πληροφοριακού συστήματος κατά τον χρόνο εκτέλεσης σε σχέση με ένα σύνολο απαιτήσεων λογισμικού. Επιπλέον, εξετάζουμε τη δυνατότητα καθορισμού ενός πλάνου αποκατάστασης στην περίπτωση που το υπό εξέταση σύστημα αποτυγχάνει να ικανοποιήσει αυτές τις απαιτήσεις. Μία γραφική αναπαράσταση του προβλήματος δίνεται με την μορφή μίας διεργασίας BPMN στην 1-1. Όπως φαίνεται στην 1-1, πρόκειται για ένα σύνθετο πρόβλημα το οποίο μπορεί να διαχωριστεί στα ακόλουθα υποπροβλήματα:

1. την επαλήθευση ορθής λειτουργίας ενός πληροφοριακού συστήματος κατά τον

χρόνο εκτέλεσης (απεικονίζεται με την διαδικασία "Verify System Compliance" στο Σχήμα 1-1), και

2. τον καθορισμού ενός πλάνου αποκατάστασης των απαιτήσεων που το υπό εξέταση σύστημα αποτυγχάνει να ικανοποιήσει (απεικονίζεται με την διαδικασία "Remediation Plan Determination" στο Σχήμα 1-1).

Προκειμένου να ολοκληρωθούν οι δύο αυτές διαδικασίες θα πρέπει να υπάρχουν οι κατάλληλοι μηχανισμοί συμπερασμού, ένας για κάθε διαδικασία. Ο πρώτος μηχανισμός θα πρέπει να ελέγχει εάν το σύστημα ικανοποιεί τις απαιτήσεις που έχουν τεθεί, ενώ ο δεύτερος πρέπει να συνθέτει ένα πλάνο αποκατάστασης, η εκτέλεση του οποίου μπορεί να οδηγήσει το σύστημα σε μία κατάσταση στην οποία οι απαιτήσεις που έχουν τεθεί για το σύστημα ικανοποιούνται.

1.2.1 Επαλήθευση Ορθής Λειτουργίας

Επαλήθευση στον χρόνο εκτέλεσης είναι η διαδικασία ελέγχου, καθώς το σύστημα βρίσκεται σε λειτουργία, του αν το σύστημα ικανοποιεί την επιθυμητή συμπεριφορά [65, 42]. Στο πλαίσιο της παρούσας διατριβής, υιοθετούμε για το πρόβλημα επαλήθευσης μία οπτική που βασίζεται σε απαιτήσεις, και γι΄ αυτό τον λόγο συμβολίζουμε το πρόβλημα ως *ReqRV* πρόβλημα. Προκειμένου να το ορίσουμε τυπικά, χρησιμοποιούμε ως σημείο έναρξης την εργασία των Zave και Jackson για τις απαιτήσεις λογισμικού [112].

Σύμφωνα με αυτήν την μελέτη, δοθέντος ενός συνόλου απαιτήσεων R, και ενός συνόλου παραδοχών του πεδίου εφαρμογής D, το πρόβλημα επαλήθευσης στοχεύει στον προσδιορισμό των προδιαγραφών S που μπορούν να εξασφαλίσουν την ικανοποίηση των απαιτήσεων R. Τα παραπάνω μπορούν να συνοψιστούν από την ακόλουθη Εξίσωση η οποία παρουσιάζεται στο [112]:

$$D \cup S \vdash R \tag{1.1}$$

Όπως και στο [23], η Εξίσωση 1.1 μπορεί να προσαρμοστεί στο πρόβλημα ReqRV, όπου τώρα το πρόβλημα μπορεί να διατυπωθεί ως εξής. Δοθέντος ενός συνόλου απαιτήσεων R (τις οποίες στο εξής θα αναφέρουμε ως *στόχους συστήματος*), ενός συνόλου παραδοχών του πεδίου εφαρμογής D, και την περιγραφή του συστήματος υπό την μορφή ενός συνόλου S από χαρακτηριστικά λειτουργίας του συστήματος, πρέπει να προσδιοριστεί ο βαθμός στον οποίο το σύστημα ικανοποιεί τους στόχους του συνόλου R.

Παρ΄ όλ΄ αυτά, σε ένα σύστημα λογισμικού, δεν έχουν όλοι οι εμπλεκόμενοι (π.χ. τελικοί χρήστες, διαχειριστές) τους ίδιους στόχους, και ακόμα και ο ίδιος χρήστης μπορεί να έχει διαφορετικούς στόχους από το σύστημα ανάλογα με το περιβάλλον στο οποίο λειτουργεί το σύστημα. Επομένως, θα πρέπει να χρησιμοποιήσουμε μία εξίσωση για κάθε

εμπλεκόμενο, η οποία θα περιγράφει τους στόχους του κάτω από διάφορες συνθήκες.

Πιο τυπικά, θεωρούμε την ακόλουθη Εξίσωση για τον εμπλεκόμενο j ενός συστήματος λογισμικού με n εμπλεκόμενους, το οποίο λειτουργεί στο πλαίσιο C:

$$D_j(C) \cup S \models R_j(C) \tag{1.2}$$

όπου $R_j(C)$ είναι οι στόχοι που θέλει να ικανοποιήσει ο $j^{\text{οστός}}$ εμπλεκόμενος στο πλαίσιο C, $D_j(C)$ είναι οι κανόνες που περιγράφουν την γνώση που σχετίζεται με αυτούς τους στόχους στο πλαίσιο C, και S είναι οι τιμές των χαρακτηριστικών λειτουργίας που αντικατοπτρίζουν την κατάσταση του συστήματος.

Επιπλέον, οι στόχοι των εμπλεκομένων δεν είναι ανεξάρτητοι μεταξύ τους. Καθώς υπάρχουν εξαρτήσεις (ή ακόμα και αντιθέσεις) μεταξύ των στόχων των διαφόρων εμπλεκομένων, θα πρέπει να υπάρχει ένα σύνολο κανόνων,το οποίο συμβολίζεται ως $D^{dep}(C)$, και περιγράφει τις εξαρτήσεις στο πλαίσιο C και μπορεί τυπικά να οριστεί ως:

$$D^{dep}(C) = \bigcup_{i=1}^{n} \bigcup_{j=1, j\neq i}^{n} D_{i\to j}(C) \tag{1.3}$$

όπου $D_{i\to j}(C)$ είναι το σύνολο των κανόνων που περιγράφουν τις εξαρτήσεις μεταξύ των στόχων του $i^{\text{οστού}}$ και του $j^{\text{οστού}}$ εμπλεκομένου στο πλαίσιο C.

Συνοψίζοντας, το ReqRV πρόβλημα για ένα δοθέν πλαίσιο C σκοπεύει στην εύρεση του βαθμού ικανοποίησης των στόχων που θέτουν και οι n εμπλεκόμενοι, δηλαδή $\bigcup_{j=1}^{n} R_j(C)$ που συμβολίζεται ως $R(C)$, λαμβάνοντας υπόψη:

a) το σύνολο S των χαρακτηριστικών λειτουργίας του συστήματος,

b) τις αλληλεξαρτήσεις που ενδεχομένως υπάρχουν μεταξύ των στόχων των διαφόρων εμπλεκομένων, δηλαδή το σύνολο$D^{dep}(C)$, και

c) την ένωση των κανόνων εξάρτησης μεταξύ στόχων που ανήκουν σε διαφορετικούς εμπλεκόμενους, δηλαδή το σύνολο $\bigcup_{j=1}^{n} D_j(C)$,

και περιγράφεται από την επόμενη εξίσωση:

$$S \cup D^{dep}(C) \cup \bigcup_{j=1}^{n} D_j(C) \models \bigcup_{j=1}^{n} R_j(C) \tag{1.4}$$

1.2.2 Αποκατάσταση

Αποκατάσταση είναι η διαδικασία προσδιορισμού μίας κατάλληλης ακολουθίας ενεργειών, δηλαδή ενός πλάνου, η εκτέλεση του οποίου μπορεί να έχει μία συνολικά θετική

επίδραση στην ικανοποίηση των απαιτήσεων στο σύστημα που βρίσκεται σε λειτουργία. Προκειμένου να δημιουργηθεί ένα τέτοιο πλάνο πρέπει να λάβουμε υπόψη τις λογικές και χρονικές εξαρτήσεις που υπάρχουν μεταξύ των ενεργειών που συμβολίζονται ως D_t, καθώς και τους κανόνες που ορίζουν το τρόπο με τον οποίο η εκτέλεση μία ενέργειας επηρεάζει τα χαρακτηριστικά του συστήματος $D_{t\rightarrow s}$. Η εκτέλεση των ενεργειών του συνόλου T, θα οδηγήσει στην μεταβολή συγκεκριμένων χαρακτηριστικών S του συστήματος, το οποίο μπορεί να γραφεί με την ακόλουθη εξίσωση:

$$D_t \cup D_{t\rightarrow s} \cup T \models S \tag{1.5}$$

Στη συνέχεια, χρησιμοποιώντας την Εξίσωση 1.4, μπορούμε να ελέγξουμε ποια θα είναι η επίδραση της εκτέλεσης των ενεργειών στην ικανοποίηση των στόχων του συστήματος.

Έτσι, η διαδικασία της αποκατάστασης περιλαμβάνει δύο επιπλέον βήματα:

1. την εύρεση ενός συνόλου ενεργειών T η εκτέλεση των οποίων μπορεί να ικανοποιήσει τους στόχους που θέτουν όλοι οι εμπλεκόμενοι για το σύστημα λογισμικού, δηλαδή $\bigcup_{j=1}^{n} R_j(C)$, λαμβάνοντας υπόψη τους περιορισμούς που περιγράφονται από τις Εξισώσεις 1.4 και 1.5, και

2. δοθέντος του συνόλου των ενεργειών που πρέπει να εκτελεστούν και των χρονικών εξαρτήσεων που υπάρχουν μεταξύ τους, την δημιουργία ενός πλάνου εκτέλεσης το οποίο σέβεται όλες τις εξαρτήσεις που υπάρχουν μεταξύ των ενεργειών.

1.3 Συνεισφορές της Διατριβής

Σκοπός της παρούσας διατριβής είναι να εισάγει ένα περιβάλλον πλαίσιο το οποίο μπορεί να χρησιμοποιηθεί για τις διαδικασίες της επαλήθευσης των απαιτήσεων λογισμικού σε ένα σύστημα που βρίσκεται σε λειτουργία και της δημιουργίας πλάνων αποκατάστασης. Κατά τη διάρκεια της διαδικασίας επαλήθευσης, τα μοντέλα που περιγράφουν με ένα βαθμό εμπιστοσύνης τις απαιτήσεις που θα πρέπει να ισχύουν σε ένα σύστημα λογισμικού το οποίο βρίσκεται σε λειτουργία, μαζί με τις αλληλεξαρτήσεις τους, και τα δεδομένα που συλλέγονται από το σύστημα καθώς λειτουργεί θα πρέπει να συνδυάζονται προκειμένου να υπολογιστεί ο βαθμός ικανοποίησης των απαιτήσεων καθώς το σύστημα μεταβάλλεται ή εξελίσσεται. Στην περίπτωση που κάποια από τις απαιτήσεις δεν ικανοποιείται ή ικανοποιείται σε έναν βαθμό που δεν ανήκει στο διάστημα των αποδεκτών τιμών, το περιβάλλον πλαίσιο πρέπει να μπορεί να προτείνει μία συγκεκριμένη ακολουθία εργασιών, η εκτέλεση της οποίας μπορεί να οδηγήσει το σύστημα σε μία κατάσταση στην οποία οι

απαιτήσεις θα ικανοποιούνται πλήρως ή τουλάχιστον θα ικανοποιούνται σε μεγαλύτερο βαθμό από ό,τι πριν. Οι συνεισφορές της διατριβής συνοψίζονται ως ακολούθως:

1. Προτείνεται η χρήση AND/OR δένδρων στόχων με σταθμισμένες ακμές συνεισφοράς, στοιχεία υπό συνθήκη, και κόμβους που μπορούν να ισχύουν κατά έναν βαθμό αλήθειας ως ένας τρόπος για την μοντελοποίηση απαιτήσεων που θα πρέπει να ισχύουν σε ένα σύστημα λογισμικού που βρίσκεται σε λειτουργία κάτω από διάφορες συνθήκες Τα μοντέλα αυτά, τα οποία αναφέρονται ως ασαφή μοντέλα στόχων, θα μας επιτρέψουν να μοντελοποιήσουμε την ασάφεια που εγγενώς υπάρχει στον ανθρώπινο τρόπο σκέψης, συνδυάζοντας στοιχεία από τα μοντέλα στόχων και την ασαφή λογική.

2. Ορίζονται και υλοποιούνται μετασχηματισμοί μεταξύ μοντέλων και μετασχηματισμοί από μοντέλα σε κείμενο έτσι ώστε να παράγονται ασαφείς βάσεις γνώσης από ασαφή μοντέλα στόχων. Οι ασαφείς βάσεις γνώσης μπορούν να χρησιμοποιηθούν σε συνδυασμό με τα δεδομένα που συλλέγονται από το σύστημα που βρίσκεται σε λειτουργία, προκειμένου να ελεγχθεί κατά πόσο ορισμένες απαιτήσεις ικανοποιούνται από το σύστημα.

3. Ορίζεται και υλοποιείται ένα μεταμοντέλο το οποίο δανείζεται στοιχεία από την θεωρία των μοντέλων δένδρων στόχων και μπορεί να μοντελοποιήσει τις λογικές και χρονικές εξαρτήσεις που υπάρχουν μεταξύ εργασιών και ενεργειών. Το μεταμοντέλο διατηρεί την AND/OR-διάσπαση των κόμβων, παράλληλα όμως χρησιμοποιούνται επιπλέον δομικά στοιχεία τα οποία μπορούν να απεικονίσουν τις λογικές και χρονικές εξαρτήσεις, αυξάνοντας έτσι την εκφραστικότητα της προτεινόμενης μεθόδου μοντελοποίησης.

4. Σχεδιάζονται και υλοποιούνται δύο μηχανισμοί συμπερασμού για τα ασαφή μοντέλα στόχων, ένας με κατεύθυνση από τα φύλλα προς τις ρίζες και ένας δεύτερος με κατεύθυνση από τις ρίζες προς τα φύλλα. Ο πρώτος (από τα φύλλα προς τις ρίζες) επιτρέπει την επαλήθευση των στόχων και απαιτήσεων που πρέπει να ικανοποιούνται στο σύστημα που βρίσκεται σε λειτουργία και χρησιμοποιεί ασαφείς μηχανισμούς συμπερασμού και τεχνικές ασαφών ελεγκτών. Ο δεύτερος (από τις ρίζες προς τα φύλλα) επιτρέπει τον προσδιορισμό πλάνων αποκατάστασης τα οποία μπορούν να οδηγήσουν το σύστημα σε μία κατάσταση όπου οι στόχοι ικανοποιούνται και χρησιμοποιεί γενετικούς αλγορίθμους και SAT solvers.

5. Εκτελούνται πειράματα που σκοπό έχουν να αξιολογήσουν την απόδοση, την επεκτασιμότητα, και τη σταθερότητα των προτεινόμενων μηχανισμών συμπερασμού.

1.4 Διάρθρωση του Κειμένου

Το κείμενο της διατριβής είναι οργανωμένο στα εξής κεφάλαια:

- Το Κεφάλαιο 2 περιγράφει υπάρχουσες μεθόδους που χρησιμοποιούνται για την μοντελοποίηση πολιτικών και την εκτέλεση διαδικασιών συμπερασμού σε πολιτικές, και συνοψίζει κύρια χαρακτηριστικά των ασαφών ελεγκτών και των σταθμισμένων ασαφών κανόνων.

- Το Κεφάλαιο 3 περιγράφει την έρευνα που έχει ήδη γίνει στις περιοχή των μοντέλων στόχων και των μεθόδων συμπερασμού για τα μοντέλα στόχων.

- Το Κεφάλαιο 4 εισάγει τα μοντέλα που χρησιμοποιούνται για την απεικόνιση των στόχων που πρέπει να ικανοποιεί το σύστημα και των ενεργειών που μπορούν να συμμετέχουν σε ένα πλάνο αποκατάστασης της επιθυμητής συμπεριφοράς του συστήματος.

- Το Κεφάλαιο 5 παρουσιάζει τα βασικά σημεία των δύο κύριων διεργασιών του προτεινόμενου περιβάλλοντος πλαισίου, δηλαδή των διαδικασιών της επαλήθευσης και της δημιουργίας πλάνων αποκατάστασης.

- Το Κεφάλαιο 6 περιγράφει τον μετασχηματισμό των μοντέλων που χρησιμοποιούνται για την μοντελοποίηση των στόχων και των ενεργειών σε σύνολα τυπικών κανόνων , έτσι ώστε να μπορούν να δημιουργηθούν κατάλληλες βάσεις γνώσης οι οποίες στη συνέχεια μπορούν να χρησιμοποιηθούν από μηχανές συμπερασμού.

- Το Κεφάλαιο 7 εισάγει τους μηχανισμούς συμπερασμού που έχουν αναπτυχθεί στα πλαίσια της παρούσας διατριβής για τις διαδικασίες της επαλήθευσης και της δημιουργίας πλάνων αποκατάστασης.

- Το Κεφάλαιο 8 παρουσιάζει τα αποτελέσματα σχετικά με την επίδοση του προτεινόμενου περιβάλλοντος πλαισίου. Η αξιολόγηση γίνεται με την πραγματοποίηση μίας σειρά πειραμάτων πάνω σε τυχαία μοντέλα διαφόρων μεγεθών και πολυπλοκοτήτων.

- Το Κεφάλαιο 9 περιγράφει τα συμπεράσματα και μελλοντικές επεκτάσεις και κατευθύνσεις της έρευνας.

Κεφάλαιο 2

Σχετικές Εργασίες

2.1 Πολιτικές σε Επιχειρησιακά Συστήματα Λογισμικού

Καθώς πολλά χαρακτηριστικά των επιχειρησιακών συστημάτων λογισμικού περιγράφονται από τον όρο *πολιτικές*, θα πρέπει αρχικά να ξεκαθαρίσουμε ποια είναι η σημασία του όρου στα πλαίσια της παρούσας εργασίας.

Ίσως ο πιο απλός ορισμός του όρου πολιτικές και την ίδια στιγμή αυτός ο οποίος ταιριάζει καλύτερα στην παρούσα εργασία είναι αυτός που δίνεται από τους Damianou et al. στο [37]. Σύμφωνα με αυτόν τον ορισμό, *πολιτικές είναι κανόνες που ορίζουν μία επιλογή στη συμπεριφορά ενός συστήματος*. Όπως περιγράφεται αναλυτικότερα στο [91], η λέξη *επιλογή* αναφέρεται στο γεγονός ότι οι πολιτικές περιγράφουν μία διεργασία η οποία είναι ήδη διαθέσιμη στο σύστημα και η οποία μπορεί να κληθεί κάτω από συγκεκριμένες συνθήκες και όχι σε μία λειτουργικότητα που πρέπει να αλλάξει. Αυτό επιτρέπει την αλλαγή της συμπεριφοράς του συστήματος μέσω της επιλογής κατάλληλων πολιτικών ασφαλείας ή ελέγχου προσβασιμότητας.

Καθώς μας ενδιαφέρει να συμπεριλάβουμε και πολιτικές που δεν αναφέρονται στην ασφάλεια του συστήματος, προσαρμόζουμε τον παραπάνω ορισμό και ορίζουμε ως *πολιτικές ένα σύνολο κανόνων που ορίζουν σκόπιμες, αναμενόμενες ή υποχρεωτικές συμπεριφορές ή ιδιότητες ενός συστήματος*. Η ανάγκη αυτής της επέκτασης επιβάλλεται από τους τύπους των πολιτικών που μελετώνται στην παρούσα εργασία και οι οποίες εισάγονται στις επόμενες ενότητες.

2.1.1 Τύποι Πολιτικών

Όπως έχει ήδη αναφερθεί, δεν επικεντρώνουμε την ανάλυση μόνο σε πολιτικές που σχετίζονται με την ασφάλεια ή τον έλεγχο προσβασιμότητας, αλλά επίσης και με πολιτικές που αφορούν σε επιχειρηματικές διαδικασίες, κανονιστικές διατάξεις και αρχιτεκτονική συστήματος. Είναι σημαντικό να διευκρινίσουμε ότι σκοπός μας δεν είναι να καταγράψουμε όλους τους πιθανούς τύπους πολιτικών που υπάρχουν σε ένα επιχειρησιακό σύστημα λογισμικού, αλλά περισσότερο να κατηγοριοποιήσουμε τους βασικότερους τύπους πολιτικών. Θεωρούμε τους παρακάτω τύπους πολιτικών:

2.1.1.1 Πολιτικές Ασφαλείας

Οι πολιτικές αυτής της κατηγορίας μοντελοποιούν τις συνθήκες κάτω από τις οποίες ένα σύστημα λογισμικού θεωρείται ότι είναι ασφαλές. Για να είναι ένα σύστημα λογισμικού ασφαλές, όλες οι υπηρεσίες που προσφέρονται από αυτό και όλες οι πληροφορίες που χρησιμοποιούνται δεν πρέπει να είναι προσβάσιμες από μη εξουσιοδοτημένους ή επικίνδυνους χρήστες. Ταυτόχρονα, το σύστημα πρέπει να είναι διαθέσιμο στους χρήστες και η επίδοση του συστήματος δε θα πρέπει να επηρεάζεται από την παρουσία κανόνων ασφαλείας και περιορισμών. Η ανάθεση ρόλων και οι πολιτικές ελέγχου προσβασιμότητας είναι παραδείγματα πολιτικών ασφαλείας.

2.1.1.2 Πολιτικές Επιχειρησιακών Διεργασιών

Οι πολιτικές που σχετίζονται με επιχειρησιακές διεργασίες καθορίζουν τη σειρά με την οποία πρέπει να εκτελεστούν συγκεκριμένες ενέργειες προκειμένου το σύστημα να πετύχει ένα συγκεκριμένο οργανωτικό στόχο. Οι καλά ορισμένες επιχειρησιακές διεργασίες είναι σημείο κλειδί για τη βελτίωση της οργανωτικής αποτελεσματικότητας, καθώς μη καλά ορισμένες και πολύπλοκες επιχειρησιακές διεργασίες επιδρούν αρνητικά στην ποιότητα των παρεχόμενων υπηρεσιών. Αξίζει να σημειωθεί ότι οι πολιτικές επιχειρησιακών διεργασιών μπορούν να χρησιμοποιηθούν για να συνθέσουν πολύπλοκες προδιαγραφές πολιτικών όπως Service Level Agreements (SLAs) [87]. Σε αυτήν την περίπτωση, η λειτουργικότητα της υπηρεσίας μπορεί να οριστεί με τη μορφή αλληλεπιδράσεων μεταξύ πελάτη και παρόχου, η οποία με τη σειρά της μπορεί να περιγραφεί με τη χρήση διαγραμμάτων ροής.

2.1.1.3 Κανονιστικές Πολιτικές

Αυτές οι πολιτικές προέρχονται από νομικούς περιορισμούς ή ενδοεταιρικούς κανόνες. Ένα παράδειγμα κανονιστικής πολιτικής που επιβάλλει την ιδιωτικότητα για την προστασία των δεδομένων και την εμπιστευτικότητα σε ιατρικά συστήματα λογισμικού είναι το

US Health Insurance Portability and Accountability Act (HIPAA) [4]. Η συμμόρφωση με τις κανονιστικές πολιτικές είναι πολύ σημαντική γαι τις εταιρείες καθώς σε αντίθετη περίπτωση μπορεί να επιβληθούν μεγάλα πρόστιμα.

2.1.1.4 Πολιτικές Σχεδίασης

Αυτή η κατηγορία πολιτικών περιλαμβάνει μοντέλα που καθορίζουν τον τρόπο που πρέπει να είναι δομημένο το σύστημα σε όρους σχεδιαστικών αποφάσεων και περιορισμών [85]. Κανόνες που σχετίζονται με την τοπολογία και τη διασύνδεση των δομικών στοιχείων ενός συστήματος είναι μοντέλα που ανήκουν σε αυτήν την κατηγορία. Οι πολιτικές σχεδίασης μπορεί ακόμα να περιγράφουν σχεδιαστικά πρότυπα σαν αυτά που ορίζονται για τις Service Oriented Architectures (SOA) που περιγράφονται στο [41]. Τα πρότυπα αυτά είναι χρήσιμα καθώς αποτελούν κατάλληλες λύσεις σε γνωστά προβλήματα.

2.1.2 Ζητήματα σχετικά με την Ανάλυση Πολιτικών

Επικεντρώνουμε κυρίως σε δύο σημαντικά, κατά τη γνώμη μας, ζητήματα, τη μοντελοποίηση πολιτικών και την επαλήθευση συμμόρφωσης με συγκεκριμένες πολιτικές.

2.1.2.1 Μοντελοποίηση Πολιτικών

Η μοντελοποίηση πολιτικών αναφέρεται στη διαδικασία δημιουργίας μίας αφηρημένης και ενδεχομένως αυστηρά ορισμένης αναπαράστασης των πολιτικών. Φυσιολογικά οι πολιτικές καταγράφονται σε φυσική γλώσσα έτσι ώστε να είναι εύκολα κατανοητές από όλους όσους παίρνουν μέρος στην ανάπτυξη ενός συστήματος λογισμικού. Παρ΄ όλ΄ αυτά, ενώ η φυσική γλώσσα μπορεί εύκολα να διαβαστεί από τους ανθρώπους, έχει ορισμένα μειονεκτήματα όταν χρησιμοποιείται για την καταγραφή πολιτικών. Πιο συγκεκριμένα, όταν δύο άνθρωποι διαβάσουν ένα κείμενο το οποίο είναι γραμμένο σε φυσική γλώσσα δεν υπάρχει κάτι που να μας εξασφαλίζει ότι θα γίνει κατανοητό και από τους δύο με τον ίδιο τρόπο. Αυτό γίνεται ακόμα χειρότερο όταν οι άνθρωποι αυτοί έχουν διαφορετικό εκπαιδευτικό ή επιστημονικό υπόβαθρο ή μερικές φορές όταν το έγγραφο δεν είναι γραμμένο στη μητρική τους γλώσσα. Εξαρτάται επίσης από το πόσο καλά γραμμένο είναι το κείμενο. Έτσι, η χρήση φυσικής γλώσσας για την καταγραφή πολιτικών αφήνει χώρο για παρερμηνείες.

Επιπλέον, η επεξεργασία εγγράφων τα οποία είναι γραμμένα σε φυσική γλώσσα με τη χρήση εργαλείων CASE δεν είναι μία εύκολη διαδικασία. Αυτό οφείλεται κυρίως στο ότι η φυσική γλώσσα δεν είναι ακριβής και σαφής όπως οι τυπικές γλώσσες.

Λαμβάνοντας υπόψη όλα τα παραπάνω, προκειμένου να μειώσουμε τις παρερμηνείες και να καταστήσουμε δυνατή τη χρήση εργαλείων CASE θα πρέπει να χρησιμοποιήσουμε

τυπικές γλώσσες. Παρ΄ όλ΄ αυτά ο ορισμός μίας τυπικής γλώσσας εμπεριέχει πάντοτε ένα συμβιβασμό μεταξύ εκφραστικότητας και πολυπλοκότητας. Στο εξής θα προσπαθήσουμε να καλύψουμε φορμαλισμούς που μπορούν να τους επεξεργαστούν εργαλεία CASE. Αυτοί οι φορμαλισμοί ποικίλλουν από γραφικούς και λεκτικούς μέχρι μαθηματικούς.

2.1.2.2 Επαλήθευση Συμμόρφωσης

Εξακρίβωση συμμόρφωσης είναι η διαδικασία εκτίμησης του εάν ένα σύστημα λογισμικού ικανοποιεί μια συγκεκριμένη πολιτική ή όχι. Ενδιαφερόμαστε κυρίως για αυτόματες και ημι-αυτόματες μεθόδους οι οποίες απαιτούν τη δήλωση των πολιτικών σε μια τυπική γλώσσα, ή τουλάχιστον σε μια γλώσσα που έχει καλά ορισμένη σημειολογία. Δοσμένης μίας πολιτικής εκφρασμένης σε μία τέτοια γλώσσα μαζί με μοντέλα τα οποία περιγράφουν δομικά και λειτουργικά χαρακτηριστικά του συστήματος λογισμικού, ένα εργαλείο ή ένας αλγόριθμος επαλήθευσης πολιτικών πρέπει να παράγει μία έξοδο που δείχνει το βαθμό συμμόρφωσης του συστήματος με τις πολιτικές. Σύμφωνα με αυτόν τον βαθμό συμμόρφωσης, οι ενδιαφερόμενοι μπορούν να λάβουν ορισμένες σχεδιαστικές αποφάσεις όταν αυτός ο έλεγχος εκτελείται κατά τη διάρκεια ανάπτυξης του συστήματος.

Πέρα από αυτό, μια άλλη σημαντική εφαρμογή των ελέγχων συμμόρφωσης είναι η χρήση τέτοιων εργαλείων και αλγορίθμων κατά το χρόνο εκτέλεσης. Στην περίπτωση αυτή, ένα μοντέλο του συστήματος ενημερώνεται ή δημιουργείται από τα δεδομένα που συλλέγονται με τη χρήση ορισμένων τεχνικών παρακολούθησης. Το παραγόμενο μοντέλο αντικατοπτρίζει την τρέχουσα κατάσταση του συστήματος και μαζί με τα εργαλεία επαλήθευσης επιτρέπει τον έλεγχο συντήρησης και εξέλιξης του συστήματος. Πηγαίνοντας ένα βήμα παραπέρα, μπορούν να αναπτυχθούν αυτο-προσαρμοζόμενα συστήματα, στην περίπτωση που ο έλεγχος συμμόρφωσης συνδυαστεί με τεχνικές αυτόματης επιβολής πολιτικών.

2.1.3 Μοντελοποίηση Πολιτικών

Όταν χρίζουμε ένα σύστημα λογισμικού είναι σημαντικό να μπορούμε να περιγράψουμε πολιτικές που ορίζουν χαρακτηριστικά της αναμενόμενης συμπεριφοράς του συστήματος. Επιπλέον, ακόμα και όταν το σύστημα βρίσκεται σε λειτουργία είναι σημαντικό να μπορούν οι εμπλεκόμενοι να επαληθεύσουν ότι όλες οι απαιτούμενες πολιτικές έχουν καλυφθεί από την υλοποίηση και συνεχίζουν να ικανοποιούνται καθώς το σύστημα μεταβάλλεται με τον χρόνο. Για να πραγματοποιηθούν αυτές οι διαδικασίες, οι πολιτικές πρέπει να περιγράφονται με έναν τρόπο ο οποίος: *a*) δεν αφήνει χώροι για παρερμηνείες, *b*) είναι αρκετά εκφραστικός ώστε να μπορεί να περιγράψει όποια πολιτική χρειαστεί και *c*) μπορεί εύκολα να γίνει κατανοητός από όλους τους εμπλεκόμενους Στην υπόλοιπη ενότητα

περιγράφουμε τεχνικές οι οποίες ποικίλλουν από γραφικές και λεκτικές μέχρι μαθηματικούς φορμαλισμούς, και που μπορούν να χρησιμοποιηθούν για να μοντελοποιήσουν ένα μεγάλο εύρος πολιτικών.

2.1.3.1 Γραφικές Απεικονίσεις

Μία μεγάλη κατηγορία τεχνικών για την καταγραφή και την μοντελοποίηση πολιτικών είναι οι γραφικές απεικονίσεις. Η δημοτικότητά τους σχετίζεται σε μεγάλο βαθμό με το γεγονός ότι είναι εύκολα κατανοητές ακόμα και για τους μη ειδικούς, ενώ την ίδια στιγμή παρέχουν την δυνατότητα μείωσης των παρερμηνειών . Επιπλέον, για ορισμένες γραφικές απεικονίσεις έχουν προταθεί σαφώς ορισμένες σημειολογίες, όπως οι [38, 19] οι οποίες χρησιμοποιούνται για την γλώσσα Business Process Model and Notation (BPMN). Αυτό είναι σημαντικό, καθώς η ύπαρξη μίας τέτοιας σημειολογίας συνεπάγεται ότι μία αυτόματη ή ημι-αυτόματη διαδικασία μπορεί να χρησιμοποιηθεί για την επαλήθευση συμμόρφωσης.

Πρέπει όμως να σημειωθεί ότι υπάρχουν και κάποιες γραφικές απεικονίσεις οι οποίες αποσκοπούν αποκλειστικά σε μία αφαιρετική αναπαράσταση των πολιτικών όπως η Use Case Maps (UCMs) η οποία παρουσιάζεται στο [20]. Η UCM χρησιμοποιείται για να απεικονίσει σενάρια σε συστήματα με πολύπλοκη αρχιτεκτονική και δεν παρέχει καμία λεπτομέρεια για αυτά τα σενάρια. Αυτό είναι χρήσιμο στην περίπτωση που οι χρήστες ενδιαφέρονται να έχουν μία πολύ γενική άποψη σχετικά με το ποια τμήματα του συστήματος συμμετέχουν σε ένα σενάριο, αλλά δεν προσφέρει μία τυπική περιγραφή και αφήνει χώρο για παρανοήσεις.

Στην υπόλοιπη ενότητα κατηγοριοποιούμε ορισμένες μεθόδους γραφικής απεικόνισης ανάλογα με την τεχνική που χρησιμοποιείται. Οι τεχνικές που συμπεριλαμβάνονται στη συνέχεια προσφέρουν κάποιο είδους δυνατότητα τυπικής γραφής χωρίς όμως να είναι απαραίτητο να συνοδεύονται από μία αντίστοιχη σημασιολογία.

UML Profiles

Ένα UML profile αποτελείται από περιορισμούς και στερεότυπα (stereotypes) και αποτελεί τον κύριο μηχανισμό που μπορεί να χρησιμοποιηθεί για την δημιουργία μοντέλων UML για συγκεκριμένα πεδία χωρίς να χρειαστεί να αλλάξει η γλώσσα μοντελοποίησης [58]. Αυτό παρέχει ένα προφανές πλεονέκτημα, η γλώσσα μοντελοποίησης μπορεί να εμπλουτιστεί με νέα στοιχεία τα οποία απαιτούνται για την περιγραφή του νέου πεδίου συνεχίζοντας να έχει τα πλεονεκτήματα της UML. Πιο συγκεκριμένα, καθώς η πλειονότητα των επαγγελματιών IT είναι εξοικειωμένοι με την UML, η νέα γλώσσα είναι εύκολο να αρχίσει να χρησιμοποιείται. Επιπλέον, υπάρχουν αρκετά εργαλεία CASE για

την UML [2] τα περισσότερα από τα οποία υποστηρίζουν τα UML profiles και κατά συνέπεια δεν υπάρχει η ανάγκη να αναπτυχθούν νέα εργαλεία.

Η SecureUML [73] είναι ένα παράδειγμα UML profile που χρησιμοποιείται για να μοντελοποιήσει πολιτικές ελέγχου πρόσβασης. Το μεταμοντέλο βασίζεται σε πολιτικές ελέγχου πρόσβασης με ρόλους και μαζί με τα στερεότυπα για τους ρόλους και τα δικαιώματα, μπορούν να εκφραστούν σε OCL περιορισμοί εξουσιοδότησης.

Ένα άλλο UML profile, για την μοντελοποίηση επιχειρησιακών διεργασιών εισάγεται στο [71]. Το προτεινόμενο μεταμοντέλο παρέχει δύο συμπληρωματικές όψεις. Η πρώτη λέγεται Business Perspective, και περιγράφει τις διεργασίες με έναν γενικό τρόπο, ενώ η δεύτερη δίνει μία πιο λεπτομερή άποψη του μοντέλου και ονομάζεται Sequence Perspective.

Ένα UML profile για την μοντελοποίηση υπηρεσιοκεντρικών αρχιτεκτονικών προτείνεται στο [18]. Το προτεινόμενο profile, ονομάζεται UML4SOA, έχει τυπική σημασιολογία και χρησιμοποιεί εκτενείς εσωτερικές δομές και διαγράμματα ανάπτυξης.

Όπως αναφέρεται στο [58] όμως, κάθε απόκλιση από την στάνταρ μορφή της UML, μπορεί να οδηγήσει σε προβλήματα και κατά συνέπεια κάθε επέκταση με την χρήση profiles πρέπει να γίνεται με προσοχή.

Ακολουθιακά Διαγράμματα

Τα ακολουθιακά διαγράμματα χρησιμοποιούνται προκειμένου να οπτικοποιηθούν σενάρια και αντι-σενάρια τα οποία συχνά χρησιμοποιούνται για να περιγράψουν πολιτικής ορθής και μη ορθής λειτουργίας. Τυπικές αναπαραστάσεις αυτής της κατηγορίας γραφικών αναπαραστάσεων είναι τα Message Sequence Charts (MSCs) [52] και τα UML Interaction Sequence Diagrams [58], με τα δεύτερα να αποτελούν μία ειδική μορφή των πρώτων.

Μία εφαρμογή των MSC περιγράφεται στο [61], όπου στόχος είναι να εκτελεστεί ένας έλεγχος επαλήθευσης ανάμεσα στην συμπεριφορά του συστήματος που θα αναπτυχθεί και ενός ιδανικού για το πεδίο μοντέλου. Για να μπορεί η διαδικασία να είναι ημι-αυτόματη, αρχικά μετατρέπουν τα σενάρια που δίνονται σε MSC ώστε να αποκτήσουν μία τυπική μοντελοποίηση της συμπεριφοράς. Επιπλέον, ή MSC επιτρέπει τους χρήστες να αρχικοποιήσουν το μοντέλο με περισσότερη πληροφορία από αυτήν που υπάρχει στα έντυπα απαιτήσεων, βελτιώνοντας έτσι τον έλεγχο επαλήθευσης και την εξαγωγή μηνυμάτων ανατροφοδότησης.

Επιπλέον, στο [12] εισάγεται μία νέα γραφική γλώσσα, η Property Sequence Chart (PSC), η οποία είναι ένα υποσύνολο των ακολουθιακών διαγραμμάτων και επιτρέπει την χρήση κατάλληλων περιορισμών και τελεστών. Τα νέα αυτά στοιχεία επιτρέπουν τον ορισμό χρονικών ιδιοτήτων για τα μηνύματα κάτι το οποίο δεν υποστηρίζεται ούτε από τα ακολουθιακά διαγράμματα ούτε από τα MSCs. Επίσης οι συγγραφείς παρέχουν την

σημασιολογία της νέας γλώσσας υπό την μορφή αυτομάτων Büchi.

Παρ΄ όλ΄ αυτά, ο τρόπος με τον οποίο τα ακολουθιακά διαγράμματα απεικονίζουν τα σενάρια εισάγει έναν περιορισμό, οι εμπλεκόμενοι θα πρέπει να μοντελοποιήσουν τις πολιτικές σαν ακολουθίες μηνυμάτων που ανταλλάσσονται μεταξύ των οντοτήτων, κάτι το οποίο μπορεί να αποτελεί μία ευθεία διαδικασία ή και όχι.

Κατευθυνόμενοι Γράφοι

Όταν οι προδιαγραφές πολιτικών μπορούν να μετασχηματιστούν σε ένα σύνολο μεταβάσεων οδηγούμενων από γεγονότα μεταξύ πιθανών καταστάσεων, μπορούμε να χρησιμοποιήσουμε για τη μοντελοποίηση κλάσεις κατευθυνόμενων γράφων (π.χ. αυτόματα, petri-nets). Καθώς οι απεικονίσεις που βασίζονται σε γράφους χρησιμοποιούνται σε διάφορα επιστημονικά πεδία, αυτού του είδους τα μοντέλα μπορούν εύκολα να χρησιμοποιηθούν και να γίνουν κατανοητά από τους περισσότερους εμπλεκομένους ανεξάρτητα από το επιστημονικό τους υπόβαθρο. Στην υπόλοιπη ενότητα επικεντρώνουμε κυρίως σε μεθόδους απεικόνισης που είναι επεκτάσεις των Petri-nets ή των μηχανών κατάστασης.

Οι Alfonso et al. στο [6] προτείνουν μία γραφική γλώσσα η οποία βασίζεται στις μηχανές κατάστασης. Σκοπός είναι να εισαχθεί μία μέθοδος απεικόνισης η οποία μπορεί να μοντελοποιήσει σύνθετα σενάρια για συστήματα πραγματικού χρόνου. Η προτεινόμενη γλώσσα που ονομάζεται Visual Timed event Scenarios (VTS), επιχειρεί να συνδυάσει εκφραστικότητα και απλότητα μαζί με τυπικό συντακτικό και καλά ορισμένη σημασιολογία. Η εργασία αυτή αποδεικνύει μέσω συγκεκριμένων παραδειγμάτων ότι η VTS, παρά το περιορισμένο σύνολο γραφικών στοιχείων μπορεί να μοντελοποιήσει αρκετά σύνθετα σενάρια. Επιπλέον, η VTS επεκτείνεται περαιτέρω στο [16] ώστε να υποστηρίζει και σενάρια υπό συνθήκη.

Η γλώσσα Requirements Capture and Analysis Tool (RCAT) [92] είναι ένα άλλο παράδειγμα γραφικής γλώσσας που αποτελεί επέκταση των μηχανών κατάστασης και πιο συγκεκριμένα μία παραλλαγή των Büchi αυτομάτων. Η γλώσσα αυτή προσπαθεί να περιορίσει τα μειονεκτήματα των Büchi αυτομάτων (δηλαδή την πολυπλοκότητα και την έλλειψη διαίσθησης), διατηρώντας παράλληλα την τυπικότητα του συντακτικού. Το αποτέλεσμα είναι μία γραφική γλώσσα η οποία μπορεί να απεικονίσει και να εκφράσει το ότι ένα γεγονός σταδιακά θα εμφανιστεί καθώς και τη χρονική διάταξη των γεγονότων με έναν απλό και διαισθητικό τρόπο.

Αξίζει να σημειωθεί ότι οι γλώσσες VTS και RCAT εισάγονται προκειμένου να προσφέρουν μία διαισθητική γραφική γλώσσα η οποία έχει τυπικό συντακτικό και καλά ορισμένη σημασιολογία. Και στις δύο περιπτώσεις σκοπός των συγγραφέων είναι να χρησιμοποιήσουν τα παραγόμενα μοντέλα ως εισόδους σε εργαλεία επαλήθευσης.

Όπως έχει ήδη σημειωθεί, τα Petri-nets είναι μία άλλη κλάση κατευθυνόμενων

γράφων η οποία μπορεί να επεκταθεί προκειμένου να δημιουργηθούν γλώσσες μοντελοποίησης για την απεικόνιση πολιτικών συμπεριφοράς. Η χρήση των Petri-nets οφείλεται κυρίως στο γεγονός ότι έχουν τυπική σημασιολογία η οποία συνδυάζεται με εκφραστικότητα και στο ότι υπάρχει γι΄ αυτά μία πληθώρα εργαλείων ανάλυσης. Παρ΄ όλ΄ αυτά οι συγγραφείς στο [98] υποστηρίζουν ότι ακόμα και τα υψηλού επιπέδου Petri-nets, έχουν κάποιους περιορισμούς όσον αφορά στην εκφραστικότητα. Αποδεικνύουν αυτούς τους περιορισμούς χρησιμοποιώντας ένα σύνολο από είκοσι πρότυπα διαγραμμάτων ροής τα οποία εμφανίζονται συχνά και τα οποία θεωρούν ότι κάθε γλώσσα διαγραμμάτων ροής πρέπει να μπορεί να τα μοντελοποιήσει. Καταλήγουν στο συμπέρασμα ότι η μοντελοποίησή τους σε υψηλού επιπέδου Petri-nets, παρά το γεγονός ότι είναι εφικτή, απαιτεί σημαντική προσπάθεια. Η πρότασή τους έναντι αυτών των περιορισμών είναι η ανάπτυξη της γλώσσας YAWL (Yet Another Workflow Language) η οποία εμπλουτίζεται με στοιχεία που υποστηρίζουν τον απευθείας ορισμό αυτών των προτύπων.

Μοντελοποίηση με τη Χρήση Πρακτόρων

Τα διαγράμματα ροής χρησιμοποιούνται κανονικά για να μοντελοποιήσουν τη συμπεριφορά ενός συστήματος λογισμικού. Φαίνεται, όμως, παρ΄ όλ΄ αυτά ότι η χρήση πρακτόρων και όλων των άλλων εννοιών που τους συνοδεύουν (π.χ. εργασίες, στόχοι, ρόλοι) βοηθούν τους εμπλεκομένους στο να αναπτύξουν μία βαθύτερη κατανόηση του συστήματος [17]. Πιο συγκεκριμένα, ένα μοντέλο προσανατολισμένο στους πράκτορες αποκαλύπτει τις αλληλεπιδράσεις μεταξύ των οντοτήτων του συστήματος και παράλληλα μεταξύ του συστήματος και των πρακτόρων. Επίσης, μία τέτοια μοντελοποίηση παρέχει μία καλύτερη οπτική του περιβάλλοντος μέσα στο οποίο λειτουργεί το σύστημα. Μία από τις πρώτες γλώσσες μοντελοποίησης προσανατολισμένη στους πράκτορες είναι η i* η οποία παρουσιάζεται στο [106]. Από τότε, έχουν προταθεί στη βιβλιογραφία πολλές επεκτάσεις της γλώσσας.

Μία πρόσφατη επέκταση της i* είναι η Active-i* [105], η οποία επεκτείνει την i* ενσωματώνοντας σε αυτήν στοιχεία των διαγραμμάτων δραστηριότητας. Ενώ η i* είναι κατάλληλη για την απεικόνιση εξαρτήσεων μεταξύ πρακτόρων, δεν έχει τη δυνατότητα να περιγράψει βήμα προς βήμα την ακολουθία των ενεργειών που συνθέτουν κάθε επιχειρησιακή διεργασία. Αντίθετα, η Active-i* συνδυάζει την ικανότητα της i* να περιγράφει τον τρόπο που αλληλεπιδρούν οι πράκτορες μεταξύ τους με την ικανότητα των διαγραμμάτων δραστηριότητας να περιγράφουν με λεπτομέρεια επιχειρησιακές διεργασίες.

Χρησιμοποιώντας τις έννοιες των πρακτόρων, στόχων και ρόλων, οι Chopra et al. [30] προτείνουν ένα φορμαλισμό που μπορεί να χρησιμοποιηθεί για να εκφράσει τους κανόνες που διέπουν την επικοινωνία μεταξύ πρακτόρων σε ένα σύστημα πρακτόρων. Για να το πετύχουν αυτό, εισάγουν τις έννοιες των Commitment και Capabilities. Συ-

νολικά, η προτεινόμενη μέθοδος απεικόνισης είναι απλή και χάρη στην ύπαρξη τυπικής σημασιολογίας [31] μπορεί να χρησιμοποιηθεί για να αξιολογήσει την ορθότητα του πρωτοκόλλου επικοινωνίας.

2.1.3.2 Τυπικές Γλώσσες

Οι γραφικές απεικονίσεις που περιγράφησαν στην προηγούμενη ενότητα έχουν το πλεονέκτημα ότι γίνονται εύκολα κατανοητές από τους περισσότερους εμπλεκομένους, αλλά μερικές φορές στερούνται τυπικότητας και κατά συνέπεια εισάγουν ένα βαθμό ασάφειας. Όταν η τυπικότητα είναι πιο σημαντική από την ευκολία χρήσης και η κατανόηση των προδιαγραφών για τις πολιτικές ανάμεσα σε ανθρώπους διαφορετικού υποβάθρου δεν είναι πρωταρχικός στόχος, μπορούν να χρησιμοποιηθούν τυπικές γλώσσες οι οποίες βασίζονται είτε στη μαθηματική λογική είτε σε γλώσσες προγραμματισμού υψηλού επιπέδου.

Γλώσσες Βασισμένες στη Λογική

Οι γλώσσες που βασίζονται στη λογική χρησιμοποιούνται κυρίως όταν οι πολιτικές θα πρέπει να υποστούν κάποιου είδους επεξεργασία ή να συμμετέχουν σε κάποια διαδικασία συμπερασμού. Η πολυπλοκότητα αυτών των γλωσσών και το γεγονός ότι απαιτείται ένα μαθηματικό υπόβαθρο τις καθιστά ακατάλληλες για την απεικόνιση πολιτικών με έναν εύκολο κατανοητό τρόπο. Έτσι, στις περισσότερες περιπτώσεις, οι γλώσσες αυτές είτε χρησιμοποιούνται για να εκφράσουν τη σημασιολογία αφηρημένων αναπαραστάσεων είτε χρησιμοποιούνται αποκλειστικά από ειδικούς. Στη σχετική βιβλιογραφία, δύο τύποι λογικών χρησιμοποιούνται για να εκφράσουν πολιτικές, οι χρονικές και οι δεοντικές.

Οι φορμαλισμοί που βασίζονται σε χρονικές λογικές είναι κατάλληλοι για τη μοντελοποίηση διαγραμμάτων ροής, καθώς μπορούν να περιγράψουν τη χρονική ακολουθία γεγονότων που συνθέτουν μία επιχειρησιακή διεργασία. Εκμεταλλευόμενοι αυτήν την ιδιότητα, οι συγγραφείς στο [83] ορίζουν ακολουθίες κλήσεων σε μεθόδους συνθέτοντας έτσι μία υπηρεσία web σε μία γλώσσα παρόμοια με την Linear Temporal Logic (LTL). Οι υπηρεσίες που περιγράφονται με αυτόν το φορμαλισμό μπορούν στη συνέχεια να ελεγχθούν ως προς τη συμμόρφωσή τους με χρονικές μη λειτουργικές πολιτικές.

Σε αντίθεση με τις χρονικές λογικές, οι δεοντικές είναι κατάλληλες για να εκφράσουν υποχρεώσεις και δικαιώματα αντί για διαγράμματα ροής. Επομένως, μπορούν να εφαρμοστούν σε νομικά κείμενα και κανόνες. Η γλώσσα FLAVOR (Formal Language for A posteriori Verification Of legal Rules) [97] είναι μία δεοντική λογική με επιπλέον κατασκευαστές χρονικών ιδιοτήτων που επιτρέπουν τον ορισμό πολιτικών ιδιωτικότητας και κανόνων επιχειρησιακών συμφωνιών με έναν ακριβή και απαλλαγμένο από ασάφειες

τρόπο. Ο ορισμός κανόνων στη FLAVOR επιτρέπει τον αυτόματο έλεγχο του κατά πόσο ένας κανόνας ικανοποιείται ή όχι σε ένα σύστημα το οποίο ελέγχεται για τη συμμόρφωσή του σε σχέση με ένα σύνολο απαιτούμενων πολιτικών.

Γλώσσες Υψηλού Επιπέδου

Φορμαλισμοί οι οποίοι βασίζονται στη λογική δεν μπορούν εύκολα να υιοθετηθούν κυρίως λόγω της πολυπλοκότητάς τους. Μία λύση σε αυτό το πρόβλημα είναι η χρήση γλωσσών υψηλότερης τάξης. Αυτές οι γλώσσες μοιάζουν με γλώσσες προγραμματισμού και κατά συνέπεια είναι πιο εύκολο για τους επαγγελματίες στον τομέα της ανάπτυξης λογισμικού να τις μάθουν και να τις χρησιμοποιήσουν. Οι πολιτικές που γράφονται σε γλώσσες υψηλού επιπέδου είτε μεταφράζονται σε λογικές φόρμουλες είτε συνοδεύονται από αλγορίθμους και εργαλεία που επιτρέπουν την περαιτέρω επεξεργασία τους.

Μία γλώσσα που ορίζεται σαν αφαίρεση υψηλότερου επιπέδου μίας χρονικής λογικής είναι η PROPOLS (Property Specification Pattern Ontology Language for Service Composition) [108] η οποία χρησιμοποιείται για τον προσδιορισμό πολιτικών σε BPEL σχήματα. Η PROPOLS κωδικοποιείται σε OWL, και οι συγγραφείς ορίζουν τη σημασιολογία της με όρους αυτομάτων περιορισμένων καταστάσεων περιγράφοντας στο [107] τον τρόπο που οι προδιαγραφές μπορούν να μετασχηματιστούν σε αυτόματα.

Επιπλέον, υπάρχουν γλώσσες οι οποίες υιοθετούν ή βασίζονται σε δομές δεοντικής λογικής, αλλά δεν έχουν τη μαθηματική αυστηρότητα των γλωσσών που περιγράφησαν στην προηγούμενη ενότητα. Αντίθετα, εκφράζονται σε λογική πρώτης τάξης ορίζοντας ένα κατάλληλο σύνολο κατηγορημάτων το οποίο εξασφαλίζει την απαιτούμενη εκφραστικότητα για τη μετάφραση νομικών κειμένων. Οι συγγραφείς στο [74] δίνουν ένα τέτοιο σύνολο κατηγορημάτων για να μεταφράσουν το HIPAA σε κανόνες Prolog. Μέσω αυτού του μετασχηματισμού, τα νομικά κείμενα μπορούν να δοθούν σε μία μορφή που είναι πιο εύκολα κατανοητή από τους μηχανικούς απαιτήσεων και κατά συνέπεια η παραγόμενη βάση γνώσης προορίζεται να χρησιμοποιηθεί αποκλειστικά από αυτούς.

Σε αντίθεση με τους περιορισμούς χρήσης των προηγούμενων γλωσσών, ο συγγραφέας στο [59], αποσκοπεί στην περιγραφή διαφόρων τύπων πολιτικών σε συστήματα που μεταβάλλονται δυναμικά. Η γλώσσα επιτρέπει τον ορισμό προτεραιοτήτων μεταξύ των κανόνων και την εύρεση πιθανών αντικρουόμενων πολιτικών.

Τέλος, μία γλώσσα η οποία δεν αναπαριστάται σε μία άλλη προϋπάρχουσα γλώσσα είναι η Ponder [37], η οποία είναι μία αντικειμενοστρεφής γλώσσα και χρησιμοποιείται κυρίως για να περιγράψει πολιτικές ελέγχου προσβασιμότητας και υποχρεώσεων για κατανεμημένα συστήματα και δίκτυα. Έχει δοθεί μεγάλη έμφαση στη δυνατότητα επέκτασης έτσι ώστε να μπορεί να υποστηρίξει πιθανούς μελλοντικούς τύπους πολιτικών για κατανεμημένα συστήματα.

2.1.4 Επαλήθευση Συμμόρφωσης Πολιτικών

Στην προηγούμενη ενότητα, παρουσιάστηκε ένα ευρύ φάσμα φορμαλισμών και γλωσσών για τη μοντελοποίση πολιτικών. Οι περισσότερες από αυτές έχουν αυστηρά ορισμένο συντακτικό και συνοδεύονται από μία τυπική σημασιολογία. Οι γλώσσες μοντελοποίησης που χαρακτηρίζονται από τις δύο αυτές ιδιότητες, εκτός από το ότι μειώνουν τον αριθμό των ασαφειών, που υπάρχουν στη φυσική γλώσσα, είναι κατάλληλες για τεχνικές που υποστηρίζουν την αυτόματη ή ημι-αυτόματη επαλήθευση συμμόρφωσης των πολιτικών.

Για να ολοκληρωθεί η διαδικασία της επαλήθευσης, εκτός από τις προδιαγραφές των πολιτικών τις οποίες στο εξής θα αναφέρουμε ως *ιδιότητες*, το σύστημα θα πρέπει να περιγραφεί με κάποιο τυπικό τρόπο, δηλαδή απαιτείται ένα μοντέλο του συστήματος.

Η ενότητα αυτή μελετά την επαλήθευση συμμόρφωσης πολιτικών δηλαδή εργαλεία και αλγορίθμους που μπορούν να χρησιμοποιηθούν για να ελέγξουν τη συμπεριφορά του συστήματος ως προς ένα σύνολο πολιτικών που ορίζεται σε κάποιο φορμαλισμό. Τα εργαλεία και οι αλγόριθμοι που περιγράφονται παρακάτω κατηγοριοποιούνται ανάλογα με τη μέθοδο που χρησιμοποιείται για τη διαδικασία της επαλήθευσης και μπορούν να εφαρμοστούν είτε στο χρόνο σχεδίασης είτε στο χρόνο εκτέλεσης.

2.1.4.1 Ελεγκτές Μοντέλων

Η πλειοψηφία των εργαλείων και των αλγορίθμων για την επαλήθευση συμμόρφωσης πολιτικών προέρχεται από τεχνικές ελέγχου μοντέλων (MCTs). Ο κύριος λόγος γι΄ αυτήν την τάση είναι το γεγονός ότι αυτές οι τεχνικές μπορούν να παρέχουν μία πλήρως αυτόματη διαδικασία επαλήθευσης και την ίδια στιγμή να παράγουν ένα αντιπαράδειγμα στην περίπτωση που η επαλήθευση αποτύχει. Παρ΄ όλ΄ αυτά, λόγω του γνωστού προβλήματος του αυξημένου αριθμού καταστάσεων, οι τεχνικές αυτές είναι σπανίως ιδανικές για την επαλήθευση πολιτικών στο χρόνο εκτέλεσης.

Ένας κατανεμημένος και παράλληλος ελεγκτής μοντέλων προτείνεται στο [82], προκειμένου να ελέγξει τη συμμόρφωση διαγραμμάτων ροής ως προς ιδιότητες που περιγράφονται σε LTL. Τα διαγράμματα ροής αναπαριστώνται γραφικά στη γλώσσα YAWL και έτσι πρέπει να μετασχηματιστούν στη γλώσσα DVE, που είναι και η γλώσσα εισόδου του ελεγκτή. Στην περίπτωση αυτή, η γλώσσα YAWL, η οποία είναι μία γλώσσα μοντελοποίησης πολιτικών χρησιμοποιείται για να μοντελοποιήσει τη συμπεριφορά του συστήματος ως ένα σύνολο διαγραμμάτων ροής.

Αντίθετα, οι συγγραφείς στο [78] χρησιμοποιούν τεχνικές ελέγχου μοντέλων για την επαλήθευση στο χρόνο εκτέλεσης, προκειμένου να επαληθεύσουν ιδιότητες πρωτοκόλλων επικοινωνίας σε συστήματα πολλαπλών πρακτόρων. Το σύστημα μοντελοποιείται σαν ένας συνδυασμός ενός μοντέλου αλληλεπιδράσεων και ενός συνόλου περιορισμών

εκφρασμένων σε δεοντική λογική, ενώ οι ιδιότητες εκφράζονται σε μ-λογισμό. Το ενδιαφέρον κομμάτι αυτής της προσέγγισης είναι η χρήση της τοπικής αναζήτησης η οποία δεν απαιτεί τη δημιουργία όλου του χώρου καταστάσεων επιτρέποντας την εφαρμογή της μεθόδου στο χρόνο εκτέλεσης. (Μία σύγκριση ανάμεσα στην καθολική έναντι της τοπικής αναζήτησης μπορεί να βρεθεί στο [88]).

2.1.4.2 Πιθανοτικοί Ελεγκτές Μοντέλων

Όταν η ποσοτική ανάλυση των πολιτικών είναι πιο σημαντική από το αν ικανοποιείται ή παραβιάζεται μία δοθείσα πολιτική, μπορούν να χρησιμοποιηθούν πιθανοτικοί ή στοχαστικοί ελεγκτές μοντέλων. Παρ΄ όλ΄ αυτά οι πιθανοτικοί ελεγκτές μοντέλων, όπως και οι συμβατικοί, έχουν υψηλές απαιτήσεις χρόνου και μνήμης.

Οι συγγραφείς στο [81] επιλέγουν να πραγματοποιήσουν μία ποσοτική ανάλυση χρησιμοποιώντας τον PRISM, ο οποίος είναι ένας στοχαστικός ελεγκτής μοντέλων. Η χρήση αυτού του ελεγκτή τους επιτρέπει να συνθέσουν ένα εργαλείο το οποίο μπορεί να εκτιμήσει τις πιθανότητες για το κόστος και την απόδοση για κάθε μελλοντική εναλλακτική σχεδιαστική απόφαση. Έτσι, το εργαλείο αυτό μπορεί να βοηθήσει τους σχεδιαστές του συστήματος στο να επιλέξουν μεταξύ των εναλλακτικών. Λόγω της φύσης της μηχανής συμπερασμού, η οποία αποκλίνει από την κλασική χρονική λογική, οι ιδιότητες καταγράφονται σε έναν φορμαλισμό παρόμοιο με τον Continuous Stochastic Logic ο οποίος είναι μία πιθανοτική επέκταση της Computation tree logic (CTL).

Αντίθετα, οι συγγραφείς στο [44] προτείνουν μία τεχνική για τη χρήση πιθανοτικών ελεγκτών μοντέλων για την επαλήθευση κατά το χρόνο εκτέλεσης. Πιο συγκεκριμένα, σκοπός τους είναι να υπολογίσουν με αποδοτικό τρόπο το βαθμό ικανοποίησης απαιτήσεων αξιοπιστίας οι οποίες εκφράζονται στη γλώσσα PCTL, που είναι μία πιθανοτική χρονική λογική. Προκειμένου να ξεπεράσουμε το πρόβλημα της αύξησης του αριθμού των καταστάσεων, το μοντέλο του συστήματος και οι ιδιότητες που πρέπει να ελεγχθούν μετασχηματίζονται σε ένα σύνολο συμβολικών εκφράσεων κατά το χρόνο σχεδίασης. Η επαλήθευση συμμόρφωσης περιορίζεται στην αποτίμηση αυτών των εκφράσεων σύμφωνα με τις τιμές που συλλέγονται από το σύστημα στο χρόνο εκτέλεσης.

2.1.4.3 Μηχανισμοί Απόδειξης Θεωρημάτων

Σαν μία λύση το πρόβλημα της αύξησης των καταστάσεων μπορεί να χρησιμοποιηθεί στη θέση του ελεγκτή μοντέλων ένας μηχανισμός απόδειξης θεωρημάτων. Στην περίπτωση αυτή, το σύστημα πρέπει να μοντελοποιηθεί ως ένα σύνολο κατηγορημάτων και κανόνων συμπερασμού και οι ιδιότητες σαν θεωρήματα τα οποία πρέπει να αποδειχτούν με την εφαρμογή αξιωμάτων και ήδη αποδεδειγμένων θεωρημάτων[102]. Κατά συνέπεια,

απαιτείται ένα δυνατό μαθηματικό υπόβαθρο προκειμένου να εφαρμόσουμε αυτές τις μεθόδους στην επαλήθευση συμμόρφωσης. Επιπλέον, όπως αναφέρεται και στο [102], ένας πλήρως αυτοματοποιημένος μηχανισμός απόδειξης θεωρημάτων στερείται ευελιξίας καθώς υπάρχει πάντα ένας συμβιβασμός μεταξύ ευελιξίας και του βαθμού καθοδήγησης που απαιτείται από τον χρήστη.

Στο [70] οι συγγραφείς χρησιμοποιούν ένα μηχανισμό απόδειξης θεωρημάτων προκειμένου να ελέγξουν αν μία αρχιτεκτονική έχει συγκεκριμένες ιδιότητες για να εντοπίσει πιθανά σφάλματα και ελλείψεις κατά το χρόνο σχεδίασης. Η αρχιτεκτονική δε μοντελοποιείται σε κάποια συγκεκριμένη περιγραφική γλώσσα αρχιτεκτονικών για να εξασφαλίσει τη γενικότητα της μεθόδου. Αντίθετα, χρησιμοποιείται μία λογική μεγαλύτερης τάξης η οποία μοντελοποιεί τις αρχιτεκτονικές υπό τη μορφή κανόνων. Η σημαντικότερη διαφορά ανάμεσα σε αυτήν την προσέγγιση και σε αυτές που περιγράφησαν νωρίτερα είναι ότι ο μηχανισμός απόδειξης θεωρημάτων επιτρέπει τη χρήση των ίδιων φορμαλισμών και για την αρχιτεκτονική και για τις ιδιότητες.

Επιπλέον, στο [113] εισάγεται ένα περιβάλλον πλαίσιο για την επαλήθευση απαιτήσεων συνεχούς λειτουργίας για σύνθετα συστήματα λογισμικού. Στην περίπτωση αυτή, οι απαιτήσεις συνεχούς λειτουργίας αναφέρονται στο σύνολο των πολιτικών τις οποίες το σύστημα πρέπει να ικανοποιεί προκειμένου να συνεχίσει να είναι διαθέσιμο ύστερα από επιθέσεις ή λειτουργικές αποτυχίες. Στόχος του περιβάλλοντος πλαισίου είναι να αποδείξει τη συμμόρφωση του συστήματος με τις απαιτήσεις συνεχούς λειτουργίας που τίθενται από τους χρήστες. Οι απαιτήσεις αυτές κωδικοποιούνται ως ένα θεώρημα σε μία συγκεκριμένη λογική που προτείνεται από τους συγγραφείς και ένα σύνολο από κανόνες συμπερασμού χρησιμοποιείται για την αυτόματη δημιουργία μίας απόδειξης γι΄ αυτό το θεώρημα. Οι χρήστες μπορούν στη συνέχεια να επαληθεύσουν ότι το σύστημα ικανοποιεί τις απαιτήσεις συνεχούς λειτουργίας ελέγχοντας την απόδειξη που έχει παραχθεί.

2.1.4.4 Άλλες Προσεγγίσεις

Τα περισσότερα από τα εργαλεία και τους αλγορίθμους που χρησιμοποιούνται για την επαλήθευση συμμόρφωσης πολιτικών βασίζονται σε ελεγκτές μοντέλων και μηχανισμούς απόδειξης θεωρημάτων. Υπάρχουν, όμως, και ορισμένες ενδιαφέρουσες προσεγγίσεις που δεν ανήκουν σε καμιά από τις προηγούμενες κατηγορίες.

Οι Giorgini et al. [48] διερευνούν την επαλήθευση συμμόρφωσης με απαιτήσεις ασφάλειας και εμπιστοσύνης για συστήματα πρακτόρων. Το σύστημα μοντελοποιείται με τη γλώσσα γραφικής απεικόνισης που παρέχεται από το Secure Tropos και η σημασιολογία του δίνεται σε λογική πρώτης τάξης (FOL). Συνεπώς, το σύστημα μπορεί να μετασχηματιστεί σε ένα σύνολο κανόνων γραμμένων σε Datalog,η οποία είναι μία λογική

γλώσσα προγραμματισμού που βασίζεται στην FOL. Έτσι, ένας μηχανισμός συμπερασμού για Datalog αρκεί ώστε να χειριστεί την επαλήθευση συμμόρφωσης από τη στιγμή που και οι ιδιότητες εκφράζονται επίσης σε λογική πρώτης τάξης.

Μία άλλη ενδιαφέρουσα προσέγγιση είναι αυτή που παρουσιάζεται στο [79], στο οποίο η θεωρία παιγνίων χρησιμοποιείται για την επαλήθευση πολιτικών σχετικών με την ασφάλεια για τοπολογίες δικτύων κατά το χρόνο σχεδίασης. Σύμφωνα με αυτήν την προσέγγιση, οι πολιτικές μοντελοποιούνται σαν παιχνίδια τα οποία παίζονται πάνω σε έναν γράφο και στα οποία οι παίκτες είναι οι πιθανές απειλές ασφαλείας και οι δυνητικοί μηχανισμοί άμυνας.

2.1.4.5 Σχόλια πάνω στην Επαλήθευση Συμμόρφωσης

Οι προαναφερόμενες μέθοδοι, ανεξάρτητα από το αν βασίζονται σε τεχνικές ελέγχου μοντέλων ή όχι, θα πρέπει να μπορούν να ανταπεξέλθουν σε δύο βασικά ζητήματα: *α)* την επιλογή του τρόπου απεικόνισης που θα χρησιμοποιηθεί για τη μοντελοποίηση του συστήματος, και *β')* την επιλογή του φορμαλισμού στην οποία κωδικοποιούνται οι ιδιότητες. Η πρώτη επιβάλλεται πάντα από το πρόβλημα που μελετάται. Με άλλα λόγια, εξαρτάται από το για ποια οπτική του συστήματος ενδιαφέρονται οι εμπλεκόμενοι ή μερικές φορές και από το είδος της πληροφορίας που είναι διαθέσιμη. Έτσι, το μοντέλο του συστήματος μπορεί να είναι ένα SA μοντέλο, στην περίπτωση που χρειάζεται να παρθούν αρχιτεκτονικές αποφάσεις, ή ένα μοντέλο επιχειρησιακών διεργασιών, εάν απαιτείται η βελτίωση των διεργασιών.

Καθώς η κατασκευή μίας αποδοτικής μηχανής συμπερασμού ή μηχανής ελέγχου μοντέλων είναι μία εργασία που απαιτεί σημαντική προσπάθεια, τις περισσότερες φορές επιλέγεται για την επαλήθευση συμμόρφωσης μία υπάρχουσα μηχανή. Αυτό επιβάλλει επιπλέον περιορισμούς ως προς τους φορμαλισμούς που μπορούν να χρησιμοποιηθούν για τη μοντελοποίηση του συστήματος και των ιδιοτήτων του. Αυτός είναι και ο λόγος που τις περισσότερες φορές οι ερευνητές πρέπει να παρέχουν αλγορίθμους για το μετασχηματισμό των μοντέλων του συστήματος στους φορμαλισμούς που απαιτούνται από τη μηχανή συμπερασμού.

Αντίθετα, ο καθορισμός του φορμαλισμού που χρησιμοποιείται για τις ιδιότητες του συστήματος καθορίζεται σχεδόν αποκλειστικά από τη μηχανή συμπερασμού που χρησιμοποιείται και έτσι, σε ορισμένες περιπτώσεις, οι συγγραφείς δεν ορίζουν κάποια νέα γλώσσα για αυτές. Αυτό έχει ως αποτέλεσμα την υιοθέτηση μαθηματικών φορμαλισμών οι οποίοι είναι δύσκολοι στην κατανόηση και ακόμα δυσκολότεροι στη χρήση τους για τον ορισμό των ιδιοτήτων. Μία λύση σε αυτό το πρόβλημα είναι ο ορισμός αφηρημένων γλωσσών (π.χ. BPSL [72]) των οποίων η σημασιολογία δίνεται υπό τη μορφή του φορμαλισμού που απαιτείται από τη μηχανή συμπερασμού.

2.1.5 Συζήτηση

Μερικά από τα ζητήματα που περιγράφονται στις προηγούμενες ενότητες συνοψίζονται στους Πίνακες 2.1 και 2.2. Στις επόμενες παραγράφους εισάγουμε κάποιες παρατηρήσεις σχετικά με τα ζητήματα που αναφέρθηκαν σε αυτήν την εργασία.

Πίνακας 2.1: Μοντελοποίηση Πολιτικών

	Graphical Notations				Formal Languages	
	UML Profiles	Seq. Charts	Directed Graphs	Agent Based	Logic Based	High Level
Security	[73]				[14]	[59], [37]
Business	[71], [89], [92]	[61], [12], [98]	[16]	[105]	[83]	[108]
Regulatory					[97]	[74], [59]
Design	[18]			[105], [30]		

2.1.5.1 Μοντελοποίηση

Ο πίνακας 2.1 δείχνει τους τύπους των πολιτικών που υποστηρίζει κάθε μία από τις μεθόδους που έχουν αναφερθεί στην Ενότητα 2.1.3. Πιο συγκεκριμένα, τις έχουμε χωρίσει σύμφωνα με τον τύπο που υποστηρίζεται στα πλαίσια της εργασίας στην οποία ορίζονται.

Το πρώτο πράγμα που παρατηρεί κάποιος είναι ότι τα ακολουθιακά διαγράμματα και οι κατευθυνόμενοι γράφοι χρησιμοποιούνται περισσότερο για τη μοντελοποίηση πολιτικών που σχετίζονται με επιχειρησιακές διεργασίες. Αυτό μπορεί να δημιουργήσει την εσφαλμένη εντύπωση ότι αυτές οι μέθοδοι απεικόνισης χρησιμοποιούνται αποκλειστικά γαι τη μοντελοποίηση αυτού του τύπου πολιτικών. Η αλήθεια είναι ότι τα ακολουθιακά διαγράμματα, καθώς και οι κατευθυνόμενοι γράφοι, φαίνεται να ταιριάζουν περισσότερο για την απεικόνιση σεναρίων και διαγραμμάτων ροής οι οποίες κατά κανόνα αποτελούν μεθόδους αφαίρεσης για επιχειρησιακές πολιτικές.

Υπάρχουν, όμως, περιπτώσεις στις οποίες τα ακολουθιακά διαγράμματα μπορούν να περιγράψουν περιορισμούς σχετικά με τους τρόπους που πρέπει να ανταλλάσσονται τα μηνύματα και κατά συνέπεια μπορούν να μοντελοποιήσουν πρωτόκολλα ασφαλείας. Για παράδειγμα, στο [51], οι συγγραφείς χρησιμοποιούν την MSC για να οπτικοποιήσουν τη γλώσσα High Level Protocol Specification Language (HLPSL), η οποία είναι μία strongly typed γλώσσα που χρησιμοποιείται για να εκφράσει πρωτόκολλα ασφαλείας με τη χρήση ρόλων, στόχων και πρακτόρων.

Πίνακας 2.2: Επαλήθευση Συμμόρφωσης Πολιτικών

Method	**Pros**	**Cons**
Model Checking	• fully automated verification process	• can not easily apply to runtime analysis
Probabilistic Model Checking	• generate counter example	• demanding in terms of execution time and memory
	• support quantitative analysis	
Theorem Provers	• allow the use of the same notation for the system and the properties	• strong mathematical background is required • low flexibility for fully automated provers

2.1.5.2 Επαλήθευση Συμμόρφωσης

Στον Πίνακα 2.2 συνοψίζονται τα πλεονεκτήματα και μειονεκτήματα των τριών βασικότερων μεθόδων που περιγράφονται στην Ενότητα 2.1.4.

Είναι σημαντικό να σημειωθεί ότι οι τεχνικές που βασίζονται σε ελεγκτές μοντέλων δεν μπορούν εύκολα να εφαρμοστούν για την επαλήθευση κατά το χρόνο εκτέλεσης εξαιτίας των περιορισμών στο χρόνο εκτέλεσης και στη μνήμη. Έτσι, για να ικανοποιήσουμε τους περιορισμούς χρόνου που τίθενται από την ανάλυση πραγματικού χρόνου πρέπει να γίνουν μετατροπές στις παραδοσιακές τεχνικές ελέγχου μοντέλων σαν αυτές που εισάγονται στα [44] και [78].

Ενώ, σε αντίθεση με τους ελεγκτές μοντέλων, οι μηχανές απόδειξης θεωρημάτων λύνουν το πρόβλημα αύξησης του αριθμού των καταστάσεων, η χρήση τους συνοδεύεται από κάποιους περιορισμούς. Έτσι, υπάρχουν κάποιες προσεγγίσεις (π.χ. [62]) οι οποίες επιχειρούν να συνδυάσουν τις τεχνικές ελέγχου μοντέλων με μηχανισμούς απόδειξης θεωρημάτων σε μία προσπάθεια να συνδυάσουν τα πλεονεκτήματα των δύο μεθόδων.

2.2 Θεωρία Ασαφών Συνόλων και Ασαφείς Ελεγκτές

Η μαθηματική μοντελοποίηση ασαφών εννοιών παρουσιάστηκε από τον Zadeh το 1965 με την εργασία σου πάνω στα ασαφή σύνολα [110]. Τα ασαφή σύνολα αποτελούν μία γενίκευση των αυστηρών συνόλων. Πιο συγκεκριμένα, δοθέντος ενός συνόλου U, ένα αυστηρό υποσύνολο A του U καθορίζεται από την χαρακτηριστική συνάρτηση χ_A η

οποία ορίζεται ως εξής:

$$\chi_A(x) = \begin{cases} 1 \text{ αν } x \in A \\ 0 \text{ αν } x \notin A \end{cases}$$

η οποία καθορίζει εάν ένα στοιχείο x ανήκει στο A ή όχι. Παρ΄ όλ΄ αυτά, στα ασαφή σύνολα η έννοια αυτή γενικεύεται επιτρέποντας απεικονίσεις των στοιχείων να ανήκουν στο διάστημα [0,1] αντί να περιορίζονται στο σύνολο των δύο στοιχείων $\{0, 1\}$. Πιο συγκεκριμένα, ένα ασαφές υποσύνολο $\tilde{A}$ του U ορίζεται ως εξής:

$$\tilde{A} = \{(x, \mu_A(x))|x \in U\}$$

όπου η συνάρτηση συμμετοχής $\mu_A : U \mapsto [0, 1]$ καθορίζει τον βαθμό συμμετοχής του κάθε στοιχείου x στο ασαφές σύνολο $\tilde{A}$.

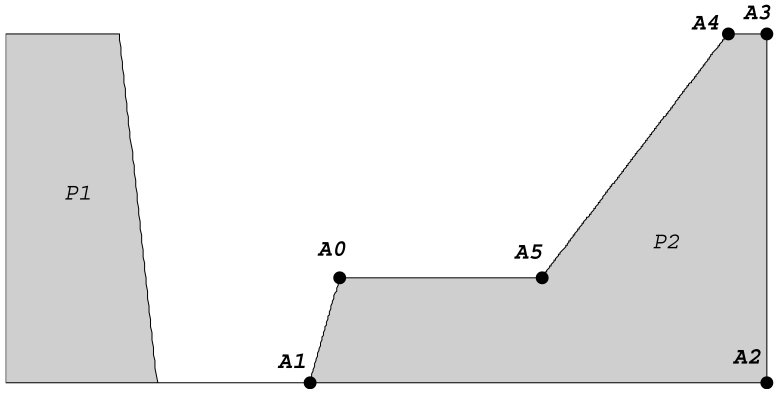
2.2.1 Ελεγκτές Ασαφούς Λογικής

Μία από τις εφαρμογές της θεωρίας ασαφών συνόλων είναι η δημιουργία ελεγκτών ασαφούς λογικής. Οι ασαφείς ελεγκτές είναι κατάλληλοι για εφαρμογές στις οποίες η κλασική λογική πρώτης τάξης δεν μπορεί να δώσει αποδεκτά αποτελέσματα ή όταν είναι δύσκολο να οριστεί ένα τυπικό μοντέλο για την αναπαράσταση την γνώσης του πεδίου εφαρμογής [34]. Καθώς η ασαφής λογική είναι πιο κοντά στον τρόπο σκέψης του ανθρώπου και στην φυσική γλώσσα από ότι τα παραδοσιακά συστήματα λογικής, παρέχει έναν αποδοτικό τρόπο για την μοντελοποίηση της προσεγγιστικής και ανακριβούς φύσης του πραγματικού κόσμου. Στην ουσία, οι ασαφείς ελεγκτές προσφέρουν έναν αλγόριθμο ο οποίος μπορεί να μετατρέψει μία στρατηγική συμπερασμού που δίνεται με λέξεις και βασίζεται στην γνώση των ειδικών σε μία στρατηγική αυτόματου συμπερασμού. Έτσι, αυτή η μεθοδολογία αποδεικνύεται ιδιαίτερα χρήσιμη όταν οι διεργασίες είναι πολύπλοκες για να εφαρμοστούν με τη χρήση παραδοσιακών τεχνικών ή όταν οι διαθέσιμες πηγές πληροφορίας ερμηνεύονται ποσοτικά, χωρίς μεγάλη ακρίβεια ή με ασάφεια.

Όπως αναφέρεται στο [80, 84], οι ασαφείς ελεγκτές συνθέτονται από τέσσερα στοιχεία: **FC-a)** *μία βάση γνώσης*, δηλαδή κανόνες που περιγράφουν την γνώση των ειδικών για την περιοχή εφαρμογής, **FC-b)** *μία διαδικασία ασαφοποίησης* για τον μετασχηματισμό των δεδομένων εισόδου σε επεξεργάσιμη μορφή, **FC-c)** *ένα μηχανισμό συμπερασμού* ο οποίος συνδυάζει την βάση γνώσης με τα δεδομένα εισόδου και εξάγει συμπεράσματα για τις μεταβλητές εξόδου, και **FC-d)** *μία διαδικασία απο-ασαφοποίησης* η οποία μετατρέπει το αποτέλεσμα που εξάγει η μηχανή συμπερασμού σε αποτελέσματα που να μπορούν να χρησιμοποιηθούν από τους χρήστες.

2.2.2 Υπολογισμός Κέντρου Βάρους Πολυγώνων

Περιγράφουμε εδώ τον υπολογισμό της x-συντεταγμένης του 'κέντρου βάρους' [3] για περιοχές που μπορούν να χωριστούν σε ένα πεπερασμένο σύνολο από κυρτά κλειστά πολύγωνα, όπως η περιοχή που φαίνεται στο Σχήμα 2-1. Ο υπολογισμός της x-συντεταγμένης του 'κέντρου βάρους' είναι μέρος της διαδικασίας αποασαφοποίησης η οποία περιγράφεται στην Ενότητα 7.1.1.3, περιγράφουμε όμως στο σημείο αυτό τα βήματα που απαιτούνται για τον υπολογισμό του για λόγους πληρότητας.



Σχήμα 2-1: Δύο κυρτά κλειστά πολύγωνα

Δοθέντος ενός πολυγώνου P με m ακμές, αριθμούμε τις κορυφές στην αντίθετη φορά του ρολογιού κατά σειρά εμφάνισης στην περίμετρό του, π.χ. τις κορυφές $A0$ με $A5$ του πολυγώνου $P2$ στο Σχήμα 2-1. Εν συνεχεία, χρησιμοποιούμε την ακολουθία (x_0, y_0), $\ldots$, (x_{m-1}, y_{m-1}) των m κορυφών για να υπολογίσουμε τον εμβαδόν της περιοχής του P ως εξής:

$$E(P) = \frac{1}{2} \sum_{k=0}^{m-1} (x_k y_{k+1} - x_{k+1} y_k) \tag{2.1}$$

όπου $(x_m, y_m) = (x_0, y_0)$, καθώς το P είναι ένα κλειστό πολύγωνο (δηλαδή η τελευταία και η πρώτη κορυφή ταυτίζονται).

Η x-συντεταγμένη του 'κέντρου βάρους' του πολυγώνου P μπορεί στη συνέχεια να υπολογιστεί από την Εξίσωση:

$$C_x(P) = \frac{1}{6E(P)} \sum_{k=0}^{m-1} (x_k + x_{k+1})(x_k y_{k+1} - x_{k+1} y_k) \tag{2.2}$$

Τέλος, η συνδυασμένη x-συντεταγμένη του 'κέντρου βάρους' μίας περιοχής που α-

ποτελείται από n κυρτά κλειστά πολύγωνα $P_1, \ldots, P_n$, δίνεται από την Εξίσωση:

$$C_x = \frac{\sum_{i=1}^{n} C_x(P_i) \cdot E(P_i)}{\sum_{i=1}^{n} E(P_i)} \tag{2.3}$$

όπου τα $E(P_i)$ και $C_x(P_i)$ υπολογίζονται από τις Εξισώσεις 2.1 και 2.2 αντίστοιχα.

2.2.3 Σταθμισμένοι Ασαφείς Κανόνες

Χτίζοντας πάνω στην θεωρία ασαφούς λογικής, οι Chortaras et al. [33] προτείνουν μία γλώσσα κανόνων η οποία μπορεί να χρησιμοποιηθεί για να εξάγει συμπεράσματα σε μη-ακριβείς ή ασαφείς βάσεις γνώσης.

Σε αυτό το πλαίσιο, μία ασαφής βάση γνώσης ορίζεται ως ένα σύνολο *σταθμισμένων ασαφών κανόνων* (weighted fuzzy rules / $wf-$κανόνες). Δύο παραδείγματα $wf-$κανόνων που μπορούν να χρησιμοποιηθούν για να εξαχθούν συμπεράσματα για τον βαθμό ευτυχίας ενός ανθρώπου p δίνονται στο Σχήμα 2-2. Όπως φαίνεται στο Σχήμα 2-2, κάθε $wf-$κανόνας αποτελείται από:

α) το τμήμα *antecedent* το οποίο δίνεται σαν μία ασαφής σύζευξη από *σταθμισμένα ασαφή άτομα weighted fuzzy atoms*, όπου το βάρος του κάθε ατόμου μοντελοποιεί τη σχετική βαρύτητα το ατόμου ανάμεσα στα άλλα άτομα,

β') το επακόλουθο του κανόνα (*consequent fuzzy atom*), π.χ. το IsHappy και στους δύο κανόνες, και

γ') ένα βάρος (*weight*) στο διάστημα $[0, 1]$ που εκφράζει την σχετική βαρύτητα του κανόνα ανάμεσα στους κανόνες που έχουν το ίδιο επακόλουθο.

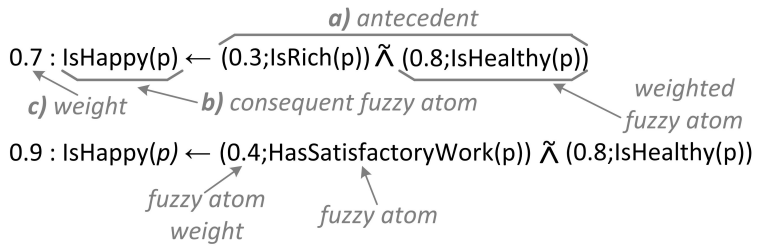


Figure 2-2: Δύο παραδείγματα $wf-$κανόνων

Επιπλέον, μία ειδική περίπτωση $wf-$κανόνων που αναφέρονται ως *ασαφή γεγονότα* (fuzzy facts) επιτρέπουν την απευθείας ανάθεση βαθμός αληθείας στα ασαφή άτομα:

$$0.4 : IsRich(p) \leftarrow (1.0; t)\,, \tag{2.4}$$

όπου 0.4 είναι ο βαθμός αληθείας του ασαφούς ατόμου IsRich για τον άνθρωπο p και t είναι ένα ορισμένο από τη γλώσσα άτομο το οποίο χρησιμοποιείται για να συμβολίσει την απόλυτα αληθή τιμή.

Δοθέντος ενός συνόλου wf-κανόνων και ένα σύνολο από ασαφή γεγονότα, ο μηχανισμός συμπερασμού που εισάγεται στο [33] υπολογίζει τις τιμές αληθείας όλων των ατόμων που εμφανίζονται στο αριστερό μέλος των κανόνων. Αυτό γίνεται χρησιμοποιώντας ένα κατάλληλο τελεστή s-norm προκειμένου να συνδυαστούν οι κανόνες που έχουν το ίδιο consequent, και έναν σταθμισμένο τελεστή t-norm για να συνδυαστούν τα σταθμισμένα ασαφή άτομα που εμφανίζονται στο δεξή μέλος των κανόνων. Στο πλαίσιο της παρούσας διατριβής, ο μηχανισμός χρησιμοποιεί τον τελεστή probabilistic sum s-norm που ορίζεται ως:

$$\perp_{sum}(a, b) = a + b - a \cdot b \tag{2.5}$$

και τον γενικευμένο σταθμισμένο ασαφή τελεστή διάζευξης που ορίζεται στο [33]:

$$\Omega^{prod}_{\langle w_1,\cdots w_n\rangle} = \text{μαξ}\left\{0, \bar{w} - 1 + \prod_{i=1}^{n} a_i^{w_i}\right\} \tag{2.6}$$

όπου a_i είναι οι τιμές αληθείας των ασαφών ατόμων που συμμετέχουν στην διάζευξη, και w_i είναι τα αντίστοιχα βάρη. Όταν όλα τα βάρη είναι ίσα με 1 ο τελεστής της Εξίσωσης 2.6 είναι ο κλασικός product t-norm τελεστή ο οποίος ορίζεται ως:

$$\top_{prod}(a, b) = a \cdot b \tag{2.7}$$

2.2.4 Γλώσσα Ασαφών Ελεγκτών

Μία γλώσσα ειδικού σκοπού η οποία μπορεί για τον ορισμό και την υλοποίηση ασαφών ελεγκτών είναι η Fuzzy Control Language (FCL), η οποία έχει δημοσιευθεί και προτυποποιηθεί από τον οργανισμό International Electrotechnical Commission (IEC 61131-7) [1]. Στο πλαίσιο της παρούσας διατριβής, χρησιμοποιούμε την βιβλιοθήκη jFuzzyLogic [34], η οποία ειναι μία βιβλιοθήκη ανοικτού κώδικα που υλοποιεί το πρότυπο IEC 61131-7.

Πιο συγκεκριμένα, η jFuzzyLogic επιτρέπει τον ορισμό IF-THEN ασαφών κανόνων

τις μορφής:

RULE 1: **IF** a_1 **IS** *low* **OR** a_2 **IS** *low*
THEN c **IS** *high* **WITH** w_1

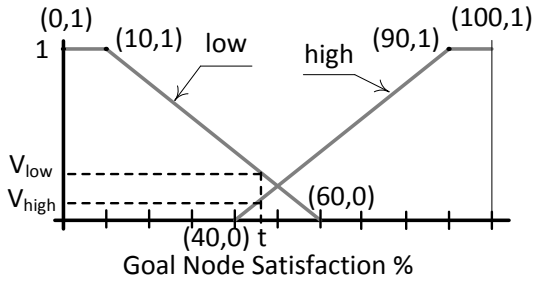
RULE 2: **IF** a_1 **IS** *high* **OR** a_2 **IS** *high*
THEN c **IS** *low* **WITH** w_2

όπου a_1 και a_2 είναι οι μεταβλητές εισόσου, c είναι η μεταβλητή εξόδου και w_1, w_2 είναι τα βάρη των κανόνων. Οι όροι *"low"*και *"high"* που χρησιμοποιούνται στους κανόνες ονομάζονται λεκτικοί όροι *linguistic terms*, και αντιστοιχούν σε ασαφή σύνολα στα οποία οι μεταβλητές μπορεί να ανήκουν με κάποια βαθμό βεβαιότητας. Σκοπός είναι δοθέντων των τιμών για τις μεταβλητές εισόδου a_1 και a_2 να μπορεί κανείς να ορίζει σε ποιο βαθμό αυτές οι δύο μεταβλητές ανήκουν στο ασαφές σύνολο "low", ή στο ασαφές σύνολο "high". Αυτό επιτυγχάνεται ορίζοντας ένα σύνολο από συναρτήσεις συμμετοχής για κάθε μεταβλητή εισόδου μέσω μίας δομής $FUZZIFY$ ως ακολούθως:

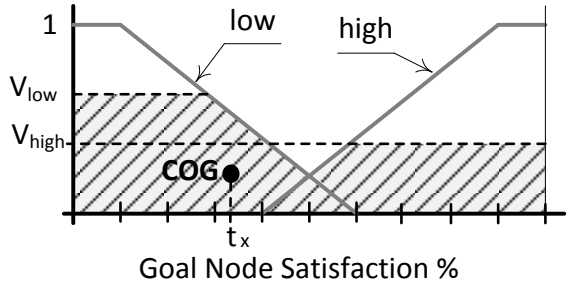
FUZZIFY a_1
TERM low := $(0,1)\,(10,1)\,(60,0)$;
TERM high := $(40,0)\,(90,1)\,(100,1)$;
END_FUZZIFY

η οποία περιγράφει την διαδικασία ασαφοποίησης για την μεταβλητή a_1. Οι δύο συναρτήσεις συμμετοχής και η διαδικασία ασαφοποίησης παρουσιάζονται στο Σχήμα 2-3. Σύμφωνα με αυτό το παράδειγμα, θεωρώντας ότι η τιμή του a_1 είναι ίση με t, η διαδικασία ασαφοποίησης θα επιστρέψει τα V_{low} και V_{high} ως τους βαθμούς κατά τους οποίους η μεταβλητή ανήκει στα ασαφή σύνολα "low" και "high" αντίστοιχα. Στην δομή $FUZZIFY$ που δίνεται παραπάνω οι τιμές (0,1) (10,1) (60,0) αναφέρονται στις συντεταγμένες των σημείων που ορίζουν της συναρτήσεις συμμετοχής για τα "low" και "high" οι οποίες δίνονται στο Σχήμα 2-3.

Μόλις οι τιμές για τις μεταβλητές εισόδου εξαχθούν με την διαδικασία της ασαφοποίησης, η μηχανή συμπερασμού μπορεί να εξάγει συμπεράσματα για τις μεταβλητές εξόδου, όπως είναι η μεταβλητή c, χρησιμοποιώντας τους ασαφείς κανόνες, σαν αυτούς που παρουσιάστηκαν νωρίτερα. Για παράδειγμα, εάν η μεταβλητή a_1 γνωρίζουμε ότι έχει τιμές V_{low}=0.2 και V_{high}=0.1, τότε και οι δύο παραπάνω κανόνες θα ενεργοποιηθούν με τις αντίστοιχες τιμές (0.2, και 0.1) για τα "low" και "high", και μία σύνθετη τιμή θα υπολογιστεί για την μεταβλητή c. Η μηχανή ασαφούς συμπερασμού υπολογίζει τις τιμές "low" και "high" για την μεταβλητή εξόδου c. Αυτές οι τιμές συνδυάζονται στη



Σχήμα 2-3: Χρήση των συναρτήσεων συμμετοχής στην διαδικασία της ασαφοποίησης



Σχήμα 2-4: Χρήση των συναρτησεων συμμετοχής στη διαδικασία της απο-ασαφοποίησης

συνέχεια από μία κατάλληλη διαδικασία απο-ασαφοποίησης η οποία και υπολογίζει το τελικό ποσοστό ικανοποίησης της c. Οι λεπτομέρειες της διαδικασίας απο-ασαφοποίησης ορίζονται από μία δομή *DEFUZZIFY* διατυπωμένη στην παρακάτω μορφή:

DEFUZZIFY c
  TERM low $:=$ $(0,1)\,(10,1)\,(60,0)$;
  TERM high $:=$ $(40,0)\,(90,1)\,(100,1)$;
  METHOD $:=$ **COG**;
END_DEFUZZIFY

όπου εκτός από τις συναρτήσεις συμμετοχής, ορίζεται επιπλέον η μέθοδος που θα χρησιμοποιηθεί, που στην περίπτωση αυτή είναι η μέθοδος κέντρου βάρους. Οι δύο συναρτήσεις συμμετοχής και η διαδικασία απο-ασαφοποίησης φαίνονται στο Σχήμα 2-4. Σύμφωνα με αυτό το παράδειγμα αν V_{low} και V_{high} είναι οι βαθμοί συμμετοχής της μεταβλητής c στα ασαφή σύνολα "low" και "high" υπολογίζεται η x-συντεταγμένη του κέντρου βάρους για την γκρι περιοχή η οποία είναι και η τιμή της μεταβλητής c.

2.3 Γενετικοί Αλγόριθμοι

Οι γενετικοί αλγόριθμοι είναι μία κλάση στοχαστικών αλγορίθμων, οι οποίοι χρησιμοποιούνται συχνά για την επίλυση προβλημάτων βελτιστοποίησης, ειδικά όταν το μέγεθος του προβλήματος βελτιστοποίησης είναι τέτοιο που καθιστά την χρήση παραδοσιακών μεθόδων αναζήτησης μη εφαρμόσιμη. Η σημαντικότερη διαφορά μεταξύ των γενετικών αλγορίθμων και των παραδοσιακών αλγορίθμων βελτιστοποίησης είναι το ότι οι πρώτοι χρησιμοποιούν ένα αρχικό σύνολο πιθανών λύσεων, οι οποίες αξιολογούνται από μία συνάρτηση καταλληλότητας (fitness function). Η αξιολόγηση των λύσεων γίνεται μέσω μίας διαδικασίας η οποία αντιστοιχεί σε κάθε κλήση μία βαθμολογία η οποία δείχνει πόσο κοντά βρίσκεται η λύση αυτή σε μία αποδεκτή λύση του προβλήματος βελτιστοποίησης.

Η γενική δομή ενός γενετικού αλγορίθμου είναι η ακόλουθη:

Algorithm 1 Genetic Algorithm General Structure

1: $t \leftarrow 0$;
2: initialize $P(t)$;
3: evaluate $P(t)$;
4: **repeat**
5: recombine $P(t)$ to yield $C(t)$;
6: evaluate $C(t)$;
7: select $P(t+1)$ from $P(t)$ and $C(t)$;
8: $t \leftarrow t+1$;
9: **until** terminationCondition

Πιο συγκεκριμένα, ένας γενετικός αλγόριθμος διατηρεί έναν πληθυσμό ατόμων ο οποίος συμβολίζεται με $P(t)$, για την γενιά t, όπου το κάθε άτομο αναπαριστά μία πιθανή λύση του προβλήματος που μελετάται. Στη συνέχεια, κάθε άτομο αξιολογείται υπολογίζοντας την τιμή που του αντιστοιχεί μέσω της συνάρτησης καταλληλότητας (γραμμή 3). Κάποια άτομα υφίστανται έναν στοχαστικό μετασχηματισμό με την μορφή γενετικών τελεστών ώστε να σχηματιστούν νέα άτομα (γραμμή 4). Υπάρχουν δύο τελεστές που χρησιμοποιούνται για να παραχθούν νέα άτομα, ο τελεστής μετάλλαξης (mutation), ο οποίος δημιουργεί νέα άτομα κάνοντας αλλαγές σε ένα άτομο που υπάρχει ήδη, και ο τελεστής διασταύρωσης (crossover), ο οποίος παράγει νέα άτομα συνδυάζοντας κομμάτια από δύο ή περισσότερα άτομα. Τα νέα άτομα τα οποία ονομάζονται απόγονοι (offspring) και συμβολίζονται στον αλγόριθμο ως $C(t)$, αξιολογούνται μέσω της συνάρτησης καταλληλότητας (γραμμή 5). Τέλος, ένας νέος πληθυσμός δημιουργείται επιλέγοντας από τους γονείς και τους απογόνους τα άτομα που έχουν την καλύτερη βαθμολογία (γραμμή 6). Μετά από αρκετές γενιές, ο αλγόριθμος συγκλίνει στο καλύτερο άτομο, το οποίο

αντιστοιχεί στην βέλτιστη ή την σχεδόν βέλτιστη λύση του προβλήματος.

Κεφάλαιο 3

Μοντέλα Στόχων και Επεκτάσεις

3.1 Βασικές Αρχές Μοντελοποίησης Στόχων

Τα AND/OR δένδρα στόχων είναι ένας φορμαλισμός μοντελοποίησης που χρησιμοποιείται εκτενώς στο πεδίο της ανάλυσης απαιτήσεων λογισμικού. Η βασική ιδέα στην μοντελοποίηση με δένδρα στόχων είναι η από πάνω προς τα κάτω ανάλυση AND/OR διάσπαση (AND/OR decomposition) των στόχων σε υποστόχους, όπου ένας AND κόμβος μπορεί να ικανοποιηθεί όταν όλοι οι υποστόχοι του είναι αληθείς, ενώ ένας OR κόμβος μπορεί να ικανοποιηθεί όταν τουλάχιστον ένας από τους υποστόχους του ικανοποιείται. Επιπλέον, σε ένα δένδρο στόχων, δύο κόμβοι μπορεί να συνδέονται μέσω μίας ακμής συνεισφοράς (contribution link) , η οποία μοντελοποιεί το γεγονός ότι ένας κόμβος συνεισφέρει θετικά ή αρνητικά σε κάποιον άλλον κόμβο. Υπάρχουν 4 τύποι ακμών συνεισφοράς, οι S^P, S^N, D^P και D^N. Ενώ στην παρούσα διατριβή υιοθετούμε για τους 4 αυτούς τύπους ακμών συνεισφοράς τη σημασιολογία των Chopra et al. [32], θεωρούμε επιπλέον ότι κάθε τέτοια ακμή συνοδεύεται από έναν πραγματικό αριθμό τον οποίο ονομάζουμε βάρος (w), και αντιστοιχεί στον 'βαθμό' στον οποίο ο στόχος αρχής (g_s) (source goal) συνεισφέρει θετικά ή αρνητικά στον στόχο τέλους (g_t) (target goal). Επομένως, μία συνεισφορά ακμής η οποία έχει βάρος w μπορεί να ερμηνευθεί όπως φαίνεται παρακάτω, ανάλογα με τον τύπο:

$$\begin{aligned} S^P/D^P &: \text{η επίτευξη/αποτυχία ικανοποίησης του στόχου } g_s \text{ συνεπάγεται} \\ &\quad \text{την επίτευξη/αποτυχία ικανοποίησης του στόχου } g_t \text{ κατά βαθμό } w \\ S^N/D^N &: \text{η επίτευξη/αποτυχία ικανοποίησης του στόχου } g_s \text{ συνεπάγεται} \\ &\quad \text{την αποτυχία/επίτευξη ικανοποίησης του στόχου } g_t \text{ κατά βαθμό } w \end{aligned} \tag{3.1}$$

Επιπλέον, οι κόμβοι στόχοι μπορούν να διαχωριστούν σε δύο τύπους, τους *hard* και

τους *soft* στόχους [76]. Οι hard στόχοι είναι στόχοι για τους οποιούς υπάρχουν σαφή κριτήρια σχετικά με το αν ικανοποιούνται ή όχι. Αντίθετα, οι soft στόχοι είναι στόχοι οι οποίοι μπορούν να ικανοποιηθούν όταν υπάρχουν επαρκή στοιχεία που συνηγορούν υπέρ της επίτευξης του στόχου και μερικά μόνο στοιχεία που θα δικαιολογούσαν την αποτυχία του.

Εφεξής υιοθετούμε την ορολογία του εισάγεται στο [13], όπου οι hard στόχοι αναφέρονται ως αυστηροί ή *crisp* στόχοι και οι soft ως ασαφείς ή *fuzzy* στόχοι. Επίσης, όταν αναφερόμαστε σε στόχους θα εννοούμε crisp και fuzzy στόχους, εκτός αν αναφέρεται διαφορετικά.

3.2 Επεκτάσεις Μοντέλων Στόχων

Υπάρχουν αρκετές επεκτάσεις των AND/OR δενδρικών μοντέλων που περιγράφησαν νωρίτερα σε αυτήν την ενότητα, οι οποίες έχουν ως στόχο να αυξήσουν την εκφραστικότητα των μοντέλων. Οι πιο σημαντικές από αυτές είναι εκείνες που χρησιμοποιούν στοιχεία υπό συνθήκη (contextual elements) , δηλαδή κόμβους και εξαρτήσεις ανάμεσα στους κόμβους που υπάρχουν/ενεργοποιούνται μόνο όταν ισχύουν συγκεκριμένες συνθήκες, και αυτές που χρησιμοποιούν ακμές οι οποίες εκφράζουν χρονικές εξαρτήσεις ανάμεσα στους κόμβους του μοντέλου.

3.2.1 Μοντέλων Στόχων με Στοιχεία υπό Συνθήκη

Τα μοντέλων στόχων με στοιχεία υπό συνθήκη (contextual goal models) έχουν προταθεί προκειμένου να είναι δυνατή η μοντελοποίηση των απαιτήσεων ενός συστήματος λογισμικού σε περιβάλλοντα που οι συνθήκες μεταβάλλονται. Υπάρχουν διάφορες εργασίες στο πεδίο των απαιτήσεων λογισμικού (Requirements Engineering) που προτείνουν την χρήση συνθηκών στα Μοντέλα Στόχων [8, 64, 10]. Σκοπός αυτών των μοντέλων είναι να καθορίσουν κάτω από ποιο σύνολο συνθηκών συγκεκριμένα τμήματα του Μοντέλου Στόχων αποτελούν μέρος του μοντέλου, και κατά συνέπεια να επιτρέψουν την ταυτόχρονη ύπαρξη πολλαπλών παραλλαγών του ίδιου μοντέλου ανάλογα με τις συνθήκες του περιβάλλοντος μέσα στο οποίο λειτουργεί το σύστημα. Πιο συγκεκριμένα, υπάρχει η δυνατότητα να ανατεθεί σε κάθε δομικό στοιχείο του μοντέλου, όπως κόμβοι, ακμές διάσπασης (decomposition links), ακμές συνεισφοράς (contribution links), ένα σύνολο συνθηκών οι οποίες λειτουργούν ουσιαστικά σαν διακόπτες που ανάλογα με την τιμή τους προσθέτουν ή αφαιρούν το αντίστοιχο στοιχείο από το μοντέλο.

3.2.2 Μοντέλα Στόχων με Χρονικές Εξαρτήσεις

Μία άλλη επέκταση των μοντέλων στόχων που προτείνεται στην βιβλιογραφία και χρησιμοποιείται στην παρούσα διατριβή είναι η χρήση ενός ειδικού τύπου ακμής που μοντελοποιεί χρονικές εξαρτήσεις που ενδεχομένως να υπάρχουν μεταξύ των κόμβων του μοντέλου. Αυτού του είδους οι ακμές, οι οποίες αναφέρονται ως *ακμές προτεραιότητας* (precedence links), εισάγονται στο [69] και χρησιμοποιούνται από τους συγγραφείς σαν ένα μέσο για την μοντελοποίηση της μεταβλητότητας όσον αφορά στην χρονική ακολουθία της ικανοποίησης των στόχων σε μοντέλα πολλαπλών πρακτόρων. Στα πλαίσια της παρούσας διατριβής θα χρησιμοποιήσουμε δύο παραλλαγές των ακμών προτεραιότητας που εισάγονται στο [69]. Περισσότερες λεπτομέρειες για αυτό θα δοθούν στο Κεφάλαιο 4.2.

3.3 Τεχνικές Συμπερασμού για Μοντέλα Στόχων

Τα μοντέλα στόχων χρησιμοποιούνται εκτενώς για ανάλυση πραγματικού χρόνου και αυτοπροσαρμοζόμενα συστήματα (adaptive software systems). Παραδείγματα εφαρμογής τους αποτελούν το [35], όπου γίνεται η διάκριση μεταξύ στόχων χρόνου σχεδίασης και στόχων χρόνου εκτέλεσης, το [75], στο οποίο προτείνεται ένα σύνολο επεκτάσεων για την μεθοδολογία του Tropos που έχει να κάνει με την μοντελοποίηση παραλείψεων και του περιβάλλοντος λειτουργίας, τα [94] και [93], όπου εισάγονται οι έννοιες των απαιτήσεων γνώσης (awareness requirements / AwReqs) και των απαιτήσεων εξέλιξης (evolution requirements / EvoReqs) ως ειδικοί τύποι μετα-απαιτήσεων, και το [13] το οποίο δίνει μία γενική εικόνα του τρόπου με τον οποίο μπορούν να χρησιμοποιηθούν οι ασαφείς στόχοι.

Επιπλέον, στο πεδίο των τεχνικών συμπερασμού για την επαλήθευση ικανοποίησης στόχων, ένας μεγάλος αριθμός προσεγγίσεων χρησιμοποιεί τεχνικές ελέγχου μοντέλων [27]. Αυτές οι τεχνικές παρέχουν μία πλήρως αυτόματη διαδικασία συμπερασμού και την ίδια στιγμή παράγουν ένα αντιπαράδειγμα στην περίπτωση που η επαλήθευση αποτύχει. Παρ΄ όλ΄ αυτά, λόγω του προβλήματος του αυξημένου αριθμού καταστάσεων (state explosion problem), αυτές οι τεχνικές σπάνια είναι κατάλληλες για την επαλήθευση κατά τον χρόνο εκτέλεσης και κατά συνέπεια χρειάζονται μετατροπές προκειμένου να εφαρμοστούν σε περιπτώσεις που υπάρχει ανάγκη ανάλυσης πραγματικού χρόνου [44, 113].

Στο [26] εισάγεται ένας γενετικός αλγόριθμος για την δημιουργία μίας διαδικασίας συμπερασμού που εφαρμόζεται κατά τον χρόνο εκτέλεσης. Τα μοντέλα στόχων που χρησιμοποιούνται στα [26] και [25] μοντελοποιούν εργασίες που αναλύονται σε ενέργειες και οι οποίες αντιστοιχούν σε στοιχειώδεις διεργασίες που μπορούν να εκτελεστούν

αυτόνομα. Δεδομένου του ότι οι κόμβοι συνοδεύονται από έναν θετικό ή αρνητικό αριθμό που αντιστοιχεί σε κέρδος ή κόστος εκτέλεσης αντίστοιχα, η διαδικασία συμπερασμού που εισάγεται στα [26] και [25] είναι μία διαδικασία από πάνω προς τα κάτω, δηλαδή από τις ρίζες προς τα φύλλα, που έχει ως σκοπό την εύρεση ενός βέλτιστου πλάνου που μπορεί να οδηγήσει στην εκτέλεση της ρίζας. Μία άλλη διαδικασία συμπερασμού που εφαρμόζεται από πάνω προς τα κάτω στο μοντέλο είναι και αυτή που εισάγεται στο [10] η οποία στοχεύει στον εντοπισμό αντιφάσεων και αλληλοαναιρούμενων στοιχείων σε μοντέλα στόχων με συνθήκες.

Ένας περιορισμός των παραπάνω προσεγγίσεων είναι ότι δεν μπορούν να μοντελοποιήσουν και να χειριστούν μη λειτουργικές απαιτήσεις. Επιπλέον δεν μπορούν να χρησιμοποιηθούν για ποσοτική ανάλυση των μοντέλων, σε αντίθεση με τον μηχανισμό συμπερασμού που χρησιμοποιείται από το προτεινόμενο περιβάλλον πλαίσιο. Ακόμα και στην περίπτωση του [44] όπου χρησιμοποιείται ένας πιθανοτικός μηχανισμός συμπερασμού, οι απαιτήσεις ελέγχονται ως προς ένα ορισμένο όριο το οποίο είναι κομμάτι των κανόνων και ο βαθμός ικανοποίησης ενός στόχου δεν μπορεί να υπολογιστεί.

Μία παρόμοια μέθοδος με αυτήν που παρουσιάζεται στο επόμενο κεφάλαιο και η οποία επιτρέπει την ποσοτική ανάλυση για δένδρα στόχων είναι η [50]. Οι συγγραφείς δεν χρησιμοποιούν ασαφή λογική για να εκφράσουν τις εξαρτήσεις μεταξύ των στόχων, όμως η σύζευξη δύο στόχων υπολογίζεται μέσω του t-norm τελεστή γινομένου (product t-norm). Επιπλέον, υπολογίζουν δύο τιμές για κάθε στόχο, μία τιμή που εκφράζει την πιθανότητα ικανοποίησης και μία δεύτερη που εκφράζει την πιθανότητα αποτυχίας του στόχου. Οι τιμές όμως αυτές δεν συνδυάζονται σε μία ενιαία τιμή που αντιστοιχεί στο συνολικό βαθμό ικανοποίησης όπως στην περίπτωση της δικής μας προσέγγισης.

Άλλες προσεγγίσεις που χρησιμοποιούν ποσοτικές τεχνικές συμπερασμού για δένδρα στόχων παρουσιάζονται στα [28] και [21]. Στο [28], τα δένδρα στόχων χρησιμοποιούνται για να περιγράψουν τις εξαρτήσεις μεταξύ των μετρικών που χρησιμοποιούνται για την διαχείριση και παρακολούθηση της πορείας ενός έργου λογισμικού όπως κόστος, και ποιότητα του συστήματος που αναπτύσσεται ή συντηρείται. Χρησιμοποιώντας δεδομένα από παλιά παρόμοια έργα, θέτουμε βάρη στις ακμές συνεισφοράς και ένας μηχανισμός συμπερασμού για δίκτυα MLN χρησιμοποιείται για να υπολογίσει τις πιθανότητες επίτευξης των στόχων. Ένα άλλο περιβάλλον πλαίσιο που χρησιμοποιεί πιθανότητες εισάγεται στο [21], στην περίπτωση αυτή όμως οι συγγραφείς προσπαθούν να προσδιορίσουν την πιθανότητα εμφάνισης εμποδίων τα οποία μπορούν να οδηγήσουν στην αποτυχία κάποιων στόχων. Γνωρίζοντας αυτές τις πιθανότητες είναι δυνατόν να επιλέξουμε ένα εναλλακτικό σχέδιο. Ο μηχανισμός συμπερασμού με πιθανότητες είναι διαφορετικός από τον μηχανισμό που χρησιμοποιείται στην παρούσα διατριβή, καθώς εδώ ενδιαφερόμαστε για τον βαθμό ικανοποίησης ενός στόχου και όχι για την πιθανότητα ο στόχος αυτός να

ικανοποιηθεί.

Επιπλέον, στο [95] εισάγεται μία ποσοτική τεχνική συμπερασμού για δένδρα στόχων στα οποία αντιστοιχίζουμε τιμές στις ακμές που συνδέουν τους στόχους μεταξύ τους. Ο μηχανισμός συμπερασμού σε αυτήν την περίπτωση αποσκοπεί στον προσδιορισμό ενός συνολικού βαθμού ο οποίος εκφράζει την επιρροή της διαγραφής ή της προσθήκης ενός κόμβου στον γράφο. Ο υπολογισμός βασίζεται σε μία ανάλυση από πάνω προς τα κάτω, κατά την διάρκεια της οποίας υπολογίζεται ένας βαθμός συνεισφοράς (achievement degree) και ένας βαθμός παρεμπόδισης (obstruction degree) για κάθε κόμβο του μοντέλου, οι οποίοι στη συνέχεια συνδυάζονται για την εξαγωγή ενός συνολικού βαθμού που ονομάζεται impact degree. Η τιμή αυτή μπορεί μετά να χρησιμοποιηθεί για τον εντοπισμό αλληλοαναιρούμενων στοιχείων στο τελικό μοντέλο.

Τέλος, η ανάλυση και η επιλογή μεταξύ εναλλακτικών στρατηγικών ή σχεδιαστικών αποφάσεων μελετάται επίσης στα [100], [39], [67], [36], [54], [15], [9]. Μία πληθώρα τεχνικών και αλγορίθμων έχουν προταθεί σε αυτές τις εργασίες οι οποίες υποστηρίζουν ευριστικές ([39],[67]), ποιοτικές ([67], [36], [54], [9]), ποσοτικές ([36], [54]), ή πιθανοτικές ([100], [54], [15]) τεχνικες συμπερασμού για δένδρα στόχων. Μερικές από αυτές τις εισάγουν τεχνικές και αλγορίθμους που υποστηρίζουν την από κάτω προς τα πάνω ανάλυση των μοντέλων ([36], [15]) όπως και στο προτεινόμενο πλαίσιο για την επαλήθευση των στόχων, παρ΄ όλ΄ αυτά τα τελικά αποτελέσματα μπορούν να χρησιμοποιηθούν κυρίως για την σύγκριση μεταξύ εναλλακτικών λύσεων.

Πίνακας 3.1: Κατηγοριοποίηση τεχνικών συμπερασμού για μοντέλα στόχων

	Ποσοτικές		Ποιοτικές / FOL
	Πιθανοτικές	Άλλες	
Bottom-Up	[100], [54], [28], [15], [21], [44]	[36], [11] [54], [50], [67]	[36], [35], [104], [103], [11], [54], [50], [39], [49]
Top-Down			[104], [103], [10], [9], [26], [25]

Η σύγκριση των παραπάνω μεθόδων παρουσιάζεται συνοπτικά στον Πίνακα 3.1. Πιο συγκεκριμένα, σε αυτόν τον Πίνακα κατηγοριοποιούμε τις εργασίες σύμφωνα με το είδος της διαδικασίας συμπερασμού που χρησιμοποιούν, δηλαδή ποιοτική, ποσοτική, πιθανοτική κ.τ.λ., και σύμφωνα με την κατεύθυνση της διαδικασίας (από πάνω προς τα κάτω (Top-Down) ή από κάτω προς τα πάνω (Bottom-Up). Το γκρι κελί στον Πίνακα αντιστοιχεί στην κατηγορία στην οποία ανήκει η μέθοδος επαλήθευσης που χρησιμοποιείται στην παρούσα διατριβή. Όπως έχει ήδη αναφερθεί, πολλές από τις εργασίες που ανήκουν σε αυτήν την κατηγορία μπορούν να χρησιμοποιηθούν κυρίως για την σύγκριση μεταξύ εναλλακτικών ([36], [54], [67]), ή έχουν κάποιους περιορισμούς σε σύγκριση με την προ-

τεινόμενη μέθοδο ([11], [50]). Μία εκτενέστερη και πιο πλήρης ανάλυση των μεθόδων συμπερασμού που έχουν προταθεί για τα μοντέλα στόχων παρουσιάζεται στα [56] και [55].

Κεφάλαιο 4

Μοντέλα Πεδίου

Σε αυτό το κεφάλαιο παρουσιάζουμε τα μοντέλα πεδίου (Domain Models) που χρησιμοποιούνται από το προτεινόμενο περιβάλλον πλαίσιο. Αρχικά εισάγουμε το μοντέλο που χρησιμοποιείται για την μοντελοποίηση των στόχων που αναμένεται να ικανοποιεί το σύστημα λογισμικού κατά τον χρόνο εκτέλεσης, ενώ στη συνέχεια δίνουμε τις λεπτομέρειες ενός μοντέλου για την καταγραφή ενεργειών και εργασιών. Το τελευταίο μπορεί να χρησιμοποιηθεί προκειμένου να μοντελοποιήσουμε το τρόπο με τον οποίο συγκεκριμένες ενέργειες μπορούν να συμβάλουν στην αποκατάσταση των απαιτήσεων, όταν εντοπιστούν στο σύστημα που βρίσκεται σε λειτουργία στόχοι οι οποίοι δεν ικανοποιούνται.

4.1 Μοντελοποίηση Πολιτικών και Όψεων με Ασαφή Μοντέλα Στόχων

Το πρώτο βήμα για τη δημιουργία ενός περιβάλλοντος πλαισίου για το ReqRV πρόβλημα, είναι ο προσδιορισμός της αναμενόμενης συμπεριφοράς του συστήματος σε διάφορα πλαίσια λειτουργίας. Στο πλαίσιο της παρούσας διατριβής, χρησιμοποιούμε μοντέλα στόχων για την καταγραφή της επιθυμητής συμπεριφοράς του συστήματος όπως αυτή ορίζεται από τους εμπλεκόμενους (stakeholders) στο σύστημα.

Στο υπόλοιπο κεφάλαιο, αρχικά εισάγουμε ένα παράδειγμα ώστε να γίνει καλύτερα κατανοητός ο τρόπος μοντελοποίησης και τα επιμέρους χαρακτηριστικά του, και στη συνέχεια περιγράφουμε το αντίστοιχο μοντέλο πεδίου και δίνουμε κάποιους πιο αυστηρούς ορισμούς των μοντέλων.

4.1.1 Παράδειγμα Μοντέλου Στόχων και Όψεων

Προκειμένου να περιγράψουμε καλύτερα τη διαδικασία που ακολουθείται για το ReqRV πρόβλημα, συνθέτουμε ένα παράδειγμα που βασίζεται εν μέρη σε παραδείγματα που παρουσιάζονται στα [7, 47, 57]. Το παράδειγμα δίνεται στο Σχήμα 4-1, και περιγράφει ένα τμήμα ενός μοντέλου στόχων ενός απλοποιημένου αλλά ρεαλιστικού συστήματος διαχείρισης δεδομένων, το οποίο μπορεί να αντιστοιχεί σε μία εφαρμογή διαχείρισης εγγραφών χρηστών.

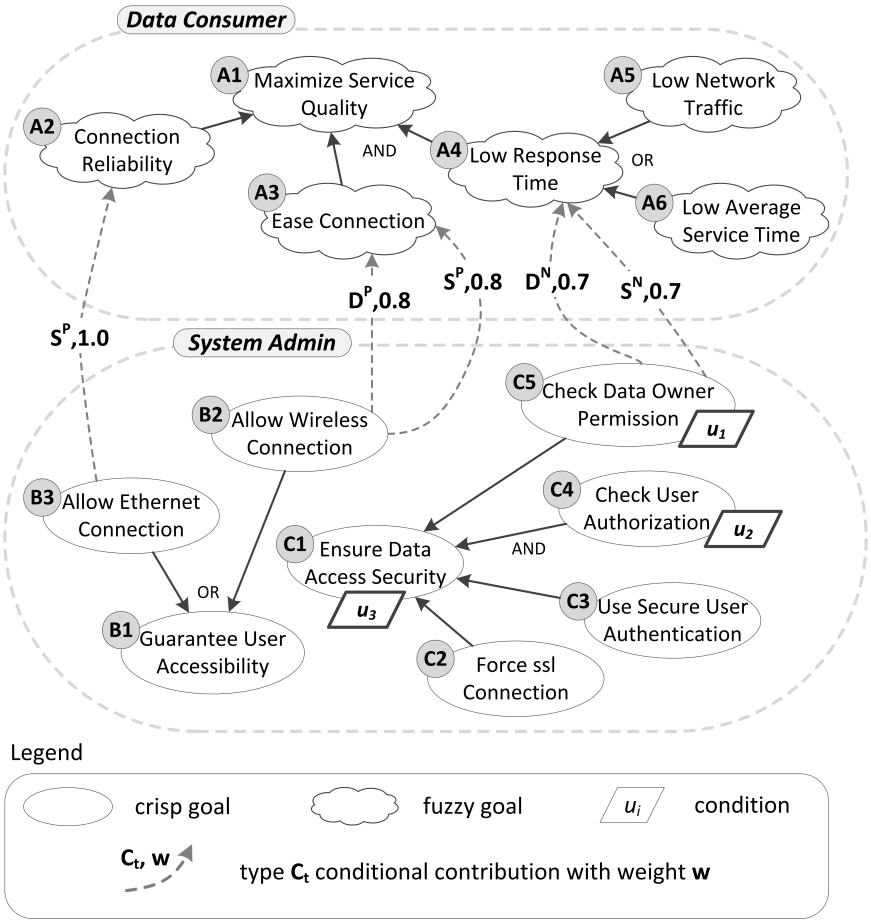
Για το παράδειγμα αυτό θεωρούμε ότι οι στόχοι προέρχονται από δύο εμπλεκομένους, τον "Data Consumer" και τον "System Admin". Ο πρώτος μπορεί να αντιστοιχεί σε ένα μέλος του βοηθητικού προσωπικού που προσπαθεί να προσπελάσει τα δεδομένα του φακέλου ενός ασθενούς σε ένα σύστημα καταχώρησης ιατρικών δεδομένων, και απαιτεί υψηλή ποιότητα των παρεχόμενων υπηρεσιών. Αυτό μοντελοποιείται από τον fuzzy στόχο "Maximize Service Quality", ο οποίος αναλύεται στους υποστόχους "Connection Reliability", "Ease Connection" και "Low Response Time".

Ομοίως, ο "System Admin" επιθυμεί να εξασφαλίσει ότι η προσπέλαση των δεδομένων θα γίνεται με ασφάλεια ("Ensure Data Access Security") και ότι οι χρήστες θα έχουν πάντα πρόσβαση στο σύστημα ("Guarantee User Accessibility"). Και οι δύο αυτοί στόχοι είναι crisp και μπορούν να αναλυθούν σε απλούστερους υποστόχους.

Επιπλέον, το μοντέλο στόχων του Σχήματος 4-1 περιέχει τρεις κόμβους υπό συνθήκη, τους "Check Data Owner Permission", "Check User Authorization" και "Ensure Data Access Security" οι οποίοι συνδέονται με τις συνθήκες u_1, u_2 και u_3 αντίστοιχα. Η συνθήκη u_1 αντιστοιχεί σε μία πολιτική ασφάλειας η οποία επιβάλλει ένα σχήμα στο οποίο ο άνθρωπος στον οποίο ανήκουν τα δεδομένα πρέπει να δώσει την συγκατάθεσή του ώστε οι υπόλοιποι χρήστες να μπορούν να έχουν πρόσβαση στα δεδομένα αυτά. Αυτή η πολιτική ασφάλειας θα πρέπει για παράδειγμα να ισχύει σε ένα σύστημα διαχείρισης ιατρικών φακέλων, όπου οι ασθενείς θα πρέπει να είναι σε θέση να περιορίζουν την πρόσβαση σε συγκεκριμένες πληροφορίες του φακέλου τους. Η πολιτική αυτή μπορεί να γραφεί πιο επίσημα σε μορφή κανόνα OCL2.0 [5] ως εξής:

```
context ContextHelper::u1Holds()
derive: security.dataPermRequired() = true
```

Από την άλλη, η συνθήκη u_2 καθορίζει το κατά πόσο το σύστημα επιτρέπει την εκτέλεση συγκεκριμένων ενεργειών σε ορισμένους χρήστες (η u_2 είναι true) ή όλοι οι χρήστες του συστήματος έχουν τα ίδια δικαιώματα στο σύστημα (η u_2 είναι false). Οι συνθήκες που συνδέονται με τα διάφορα στοιχεία των μοντέλων στόχων μπορούν να χρησιμοποιηθούν προκειμένου να αλλάξουν την ανάλυση. Για παράδειγμα, ο κόμβος "Ensure Data Access Security" υπάρχει στο μοντέλο ανάλογα με την τιμή της συνθήκης



Σχήμα 4-1: Παράδειγμα μοντέλου στόχων και όψεων

u_3, και επιπλέον αναλύεται σε ένα διαφορετικό σύνολο υποστόχων ανάλογα με την τιμή των συνθηκών u_1 και u_2.

Επιπλέον, οι ακμές συνεισφοράς χρησιμοποιούνται για να μοντελοποιήσουν τις εξαρτήσεις που υπάρχουν μεταξύ των στόχων ενός μοντέλου. Για παράδειγμα, το γεγονός ότι οι χρήστες μπορούν να συνδεθούν στο σύστημα μέσω ασύρματης σύνδεσης, δηλαδή ο κόμβος "Allow Wireless Connection" είναι true, συνεισφέρει θετικά στον fuzzy στόχο "Ease Connection" με βαθμό 0.8. Στις επόμενες ενότητες θα αναφερόμαστε στους κόμβους του μοντέλου χρησιμοποιώντας τους κωδικούς που υπάρχουν πάνω σε κάθε έναν από αυτούς, π.χ. ο κωδικός $A2$ αντιστοιχεί στον κόμβο "Connection Reliability".

Τέλος, πρέπει να σημειωθεί ότι ενώ στο παράδειγμα του Σχήματος 4-1 οι δύο εμπλεκόμενοι φαίνεται να έχουν διαφορετικά σύνολα στόχων, δεν υπάρχει κάποιος περιορισμός σχετικά με τους στόχους που μπορεί να ορίσει ο κάθε εμπλεκόμενος. Αντίθετα, διαφορετικοί εμπλεκόμενοι μπορούν να ορίσουν παρόμοιους ή και ίδιους στόχους οι οποίοι αναλύονται με διαφορετικό τρόπο σε υποστόχους ανάλογα με το τι ταιριάζει καλύτερα στις ανάγκες του κάθε εμπλεκόμενου.

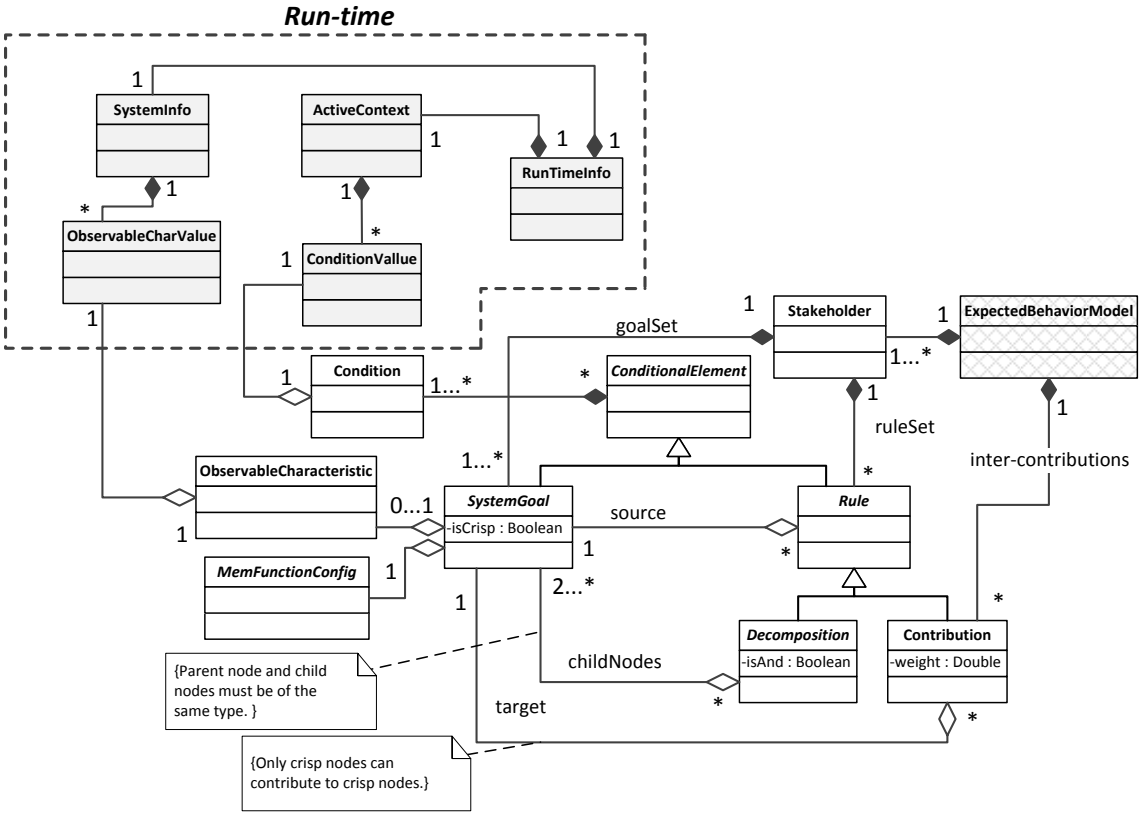
4.1.2 Μοντέλο Στόχων Συστήματος

Το προτεινόμενο μοντέλο πεδίου δίνεται στο Σχήμα 4-2. Σύμφωνα με αυτό το μοντέλο, κάθε εμπλεκόμενος (κλάση "Stakeholder") του συστήματος ορίζει έναν ή περισσότερους στόχους (σχέση "goalSet" στο Σχήμα 4-2), οι οποίοι περιμένει να ισχύουν στο υπό εξέταση σύστημα λογισμικού. Αυτοί οι στόχοι μπορεί να είναι είτε crisp είτε fuzzy και αναλύονται σε απλούστερους υποστόχους με την χρήση κανόνων διάσπασης, ενώ οι κανόνες συνεισφοράς μπορούν να χρησιμοποιηθούν για τον ορισμό σχέσεων ανάμεσα στους στόχους του μοντέλου. Έτσι, κάθε στιγμιότυπο της κλάσης "Stakeholder" ορίζεται σαν ένα σύνολο στόχων ριζών και ένα σύνολο κανόνων διάσπασης και συνεισφοράς. Οι στόχοι όπως και οι κανόνες μπορεί να συνδέονται με μία οι περισσότερες συνθήκες όπως και στα [8, 64]. Αυτό φαίνεται από το γεγονός ότι οι κλάσεις "Goal" και "Rule" κληρονομούν από την αφηρημένη κλάση "ConditionalElement", δηλαδή συγκεκριμένοι στόχοι και σχέσεις μπορούν να ενεργοποιηθούν, και έτσι να αποτελούν τμήμα του μοντέλου, μόνο όταν συγκεκριμένες συνθήκες ισχύουν. Με άλλα λόγια, οι συνθήκες λειτουργούν σαν διακόπτες οι οποίοι ενεργοποιούν ή απενεργοποιούν στόχους και σχέσεις μεταξύ των στόχων, κάτι το οποίο δίνει τη δυνατότητα να μοντελοποιούμε πολλαπλές όψεις σε ένα ενιαίο μοντέλο.

Το σύνολο όλων των στόχων, μαζί με έναν αριθμό κανόνων συνεισφοράς μεταξύ των στόχων που ανήκουν σε διαφορετικούς εμπλεκομένους, αποτελούν το ενιαίο μοντέλο αναμενόμενης συμπεριφοράς του συστήματος (κλάση "ExpectedBehaviorModel").

Παράλληλα, κάθε κόμβος του μοντέλου που δεν είναι ούτε στόχος κάποιας ακμής συνεισφοράς, αλλά ούτε αναλύεται σε υποστόχους ονομάζεται *φύλλο* (leaf goal node) και συνδέεται με ένα συγκεκριμένο χαρακτηριστικό λειτουργίας του συστήματος (κλάση "ObservableCharacteristic"). Καθώς το σύστημα λειτουργεί, δηλαδή στον χρόνο εκτέλεσης, εξάγονται τιμές για τα χαρακτηριστικά λειτουργίας (κλάση "ObservableCharValue") μέσω τεχνικών παρακολούθησης του συστήματος και κατάλληλες συναρτήσεις συμμετοχής που περιγράφονται με στιγμιότυπα της κλάσης "MemFunctionConfig" χρησιμοποιούνται ώστε οι τιμές αυτές να μετατραπούν σε βαθμούς συμμετοχής για τους κόμβους (βλέπε Ενότητες 7.1.1.3 και 6.3.1 για τις συναρτήσεις συμμετοχής και την χρήση τους).

Το ενεργό πλαίσιο, δηλαδή η ανάθεση τιμών ("ConditionValue") στις συνθήκες, επιτρέπουν τον καθορισμό των κανόνων που πρέπει να χρησιμοποιηθούν έτσι ώστε να υπολογιστούν οι τιμές των ριζών ξεκινώντας από τις αρχικές αναθέσεις τιμών στα φύλλα. Τέλος, οι αληθοτιμές των στόχων μπορούν να μετατραπούν με μετρικές μέσω μίας κατάλληλης διαδικασίας αποασαφοποίησης, η οποία για να εκτελεστεί κάθε κόμβος του μοντέλου πρέπει να συνδέεται με ένα στιγμιότυπο της κλάσης "MemFunctionConfig".



Σχήμα 4-2: Μεταμοντέλο αναμενόμενης συμπεριφοράς συστήματος

Table 4.1: Constraints for gm-rules

Rule Type	Constraints
AND, OR	***(1a)*** $w = 1$ ***(1b)*** **if** g_t is fuzzy/crisp **then** all nodes in G_s must be fuzzy/crisp
S^P, D^P, S^N, D^N	***(2a)*** $\|G_s\| = 1$ ***(2b)*** **if** g_t is crisp **then** the source node must be crisp and $w = 1$

4.1.3 Ορισμοί

Έχοντας υπόψη την περιγραφή που δόθηκε στην προηγούμενη ενότητα για τα μοντέλα στόχων, στην ενότητα αυτή εισάγουμε ένα σύνολο ορισμών που θα μας επιτρέψουν να προσδιορίσουμε τα επιμέρους μοντέλα στόχων των εμπλεκομένων, και στη συνέχεια να ορίσουμε το ενιαίο μοντέλο στόχων του συστήματος. Πιο συγκεκριμένα:

Ορισμός 1 *Ένας κανόνας μοντέλου στόχων (gm-rule) είναι μία έκφραση της μορφής* $\langle R_t, g_t, G_s, w \rangle$*, όπου* $R_t \in \{AND, OR, S^P, D^P, S^N, D^N\}$ *είναι ο τύπος του κανόνα,* g_t *είναι ο στόχος τέλους του κανόνα,* G_s *είναι το μη κενό σύνολο των κόμβων (crisp ή*

fuzzy) που δεν περιέχει τον κόμβο g_t, και $w \in (0, 1]$ είναι το βάρος του κανόνα.

Για παράδειγμα ο κανόνας $AND-$διάσπασης του fuzzy κόμβου A_1 στους κόμβους A_2, A_3, και A_4 στο Σχήμα 4-1 μπορεί να γραφεί ως εξής:

$$r_{A1} \quad = \quad \langle \text{AND}, A1, \{A2, A3, A4\}, 1.0 \rangle$$

Ανάλογα με τον τύπο του $gm-$κανόνα, ισχύουν οι περιορισμοί που δίνονται συνοπτικά στον Πίνακα 4.1 και εξασφαλίζουν ότι:

(1a) οι κανόνες διάσπασης έχουν βάρος πάντα ίσο με ένα,

(1b) ένας κόμβος μπορεί να αναλυθεί/διασπαστεί μόνο σε κόμβους του ίδιου τύπου,

(2a) υπάρχει μόνο ένας κόμβος αρχής σε έναν κανόνα συνεισφοράς και συμβολίζεται με g_s, και

(2b) μόνο crisp κόμβοι μπορούν να συνεισφέρουν σε έναν crisp κόμβο και για αυτούς τους κανόνες το βάρος είναι πάντα ίσο με 1.

Οι κόμβοι καθώς και οι gm-κανόνες μπορεί να είναι υπό συνθήκη. Για να μπορούν να ορίζονται τα στοιχεία υπό συνθήκη, ορίζουμε το σύνολο $\mathcal{U}$ όλων των δυνατών συνθηκών που μπορούν να συνδεθούν με του κανόνες ή τους κόμβους, όπου η κάθε συνθήκη είναι μία μεταβλητή η οποία μπορεί να είναι είτε αληθής είτε ψευδής και μπορεί να προσθέτει ή να αφαιρεί στοιχεία από το μοντέλο. Για παράδειγμα, για το μοντέλο του Σχήματος 4-1 το σύνολο $\mathcal{U}$ ορίζεται ως $\mathcal{U} = \{u_1, u_2, u_3\}$, με την τιμή της συνθήκης u_1 να καθορίζει την ύπαρξη ή την απουσία του κόμβου $C5$ στο μοντέλο. Αντίθετα, τα στοιχεία που είναι πάντα παρόντα στο μοντέλο (π.χ. ο κόμβος $A1$) ονομάζονται *σταθερά* και συνδέονται με την συνθήκη $\top$ που είναι πάντα true.

Λαμβάνοντας υπόψη τα προηγούμενα δίνουμε τον παρακάτω ορισμό για το μοντέλο στέχων ενός εμπλεκομένου:

Ορισμός 2 *Το μοντέλο στόχου ενός εμπλεκομένου (Stakeholder Goal Model / SGM) είναι μία έκφραση της μορφής $\langle id, G_{id}, R_{id}, Cond_{id} \rangle$, όπου το id είναι ένα μοναδικό αναγνωριστικό, G_{id} είναι ένα μη κενό σύνολο (crisp ή fuzzy) κόμβων, R_{id} είναι ένα σύνολο από $gm-$κανόνες που περιγράφουν τις σχέσεις που υπάρχουν μεταξύ των κόμβων του συνόλου G_{id}, και $\eth\text{νδ}_{id} : G_{id} \cup R_{id} \mapsto Pow(\mathcal{U} \cup \{\top\})$, είναι μία συνάρτηση που επιστρέφει το σύνολο των συνθηκών που συνδέονται με κάθε στόχο ή $gm-$κανόνα του SGM.*

Πιο συγκεκριμένα, ορίζουμε τη συνάρτηση $\text{Cond}_{id}(a)$ η οποία επιστρέφει για έναν κόμβο ή για έναν $gm-$κανόναa, τις συνθήκες $U^a \subseteq \mathcal{U} \cup \{\top\}$ για τις οποίες το στοιχείο a υπάρχει στο μοντέλο, δηλαδή $U^a = \eth\text{νδ}_{id}(a)$. Έτσι, το στοιχείο a υπάρχει στο μοντέλο μόνο όταν τουλάχιστον μία από τις συνθήκες του συνόλου U^a είναι αληθής.

Ένα απλό παράδειγμα ενός SGM φαίνεται στο κάτω μέρος του Σχήματος 4-1. Το SGM ονομάζεται "System Admin", είναι τμήμα ενός μοντέλου 2 εμπλεκομένων και μπορεί επίσημα να οριστεί ως $\langle \text{SGM}_{SA}, G_{\text{SA}}, R_{\text{SA}}, Cond_{\text{SA}} \rangle$, όπου:

$$G_{\text{SA}} = \{B1, B2, B3, C1, C2, C3, C4, C5\}$$

$$R_{\text{SA}} = \begin{cases} r_{B1} = \langle \text{OR}, B1, \{B2, B3\}, 1 \rangle \ , \\ r_{C1} = \langle \text{AND}, C1, \{C2, C3, C4, C5\}, 1 \rangle \end{cases}$$

$$Cond_{\text{SA}}(a) = \begin{cases} \{U_1\} \ , & \text{if } \ a = C5 \\ \{U_2\} \ , & \text{if } \ a = C4 \\ \{U_3\} \ , & \text{if } \ a = C1 \\ \{\top\} \ , & \text{otherwise} \end{cases}$$

Αναθέτοντας αληθοτιμές στις συνθήκες του συνόλου $\mathcal{U}$, μπορούμε να παράξουμε πολλαπλά *στιγμιότυπα/όψης* (views) για το ίδιο SGM , όπου κάθε στιγμιότυπο περιέχει διαφορετικούς στόχους και $gm-$κανόνες. Έτσι, ορίζοντας ένα σύνολο $U \subseteq \mathcal{U} \cup \{\top\}$ το οποίο περιέχει μόνο τις συνθήκες που είναι αληθείς, το οποίο στο εξής θα αναφέρεται ως *ενεργό πλαίσιο* (active context), μπορούμε να παράξουμε ένα συγκεκριμένο στιγμιότυπο του μοντέλου SGM. Εξ ορισμού, το ενεργό πλαίσιο περιέχει τουλάχιστον την μεταβλητή $\top$, καθώς αναπαριστά μία συνθήκη η οποία είναι πάντα αληθής. Δοθέντος ενός ενεργού πλαισίου U και ενός SGM $m = \langle id, G_{id}, R_{id}, Cond_{id} \rangle$, λέμε ότι παράγουμε το στιγμιότυπο του m ως προς το U, το οποίο συμβολίζουμε ως $m \restriction_U$ και είναι ένα μοντέλο στόχων το οποίο περιέχει:

1. Όλους τους κόμβους $g \in G_{id}$ για τους οποίους $Cond_{id}(g) \cap U \neq \emptyset$, και
2. Όλους τους κανόνες $r = \langle R_t, g_t, G_s, w \rangle \in R_{id}$ για τους οποίους:

   (α′) $Cond_{id}(r) \cap U \neq \emptyset$, και

   (β′) $Cond_{id}(g_t) \cap U \neq \emptyset$, και

   (γ′) $\exists g_s \in G_s : Cond_{id}(g_s) \cap U \neq \emptyset$

Με άλλα λόγια, το $m \restriction_U$ είναι ένα μοντέλο στόχων το οποίο περιέχει όλους τους στόχους που συνδέονται με συνθήκες που ισχύουν (δηλαδή το ενεργό πλαίσιο), και επιπλέον τους $gm-$κανόνες του συνδέονται με συνθήκες που ισχύουν (2a στην παραπάνω λίστα) και για

τους οποίους ο κόμβος τέλους (2b) και τουλάχιστον ένας κόμβος αρχής (2c) υπάρχουν στο μοντέλο.

Εφόσον οι στόχοι όλων των εμπλεκομένων έχουν μοντελοποιηθεί σας ένα σύνολο από SGM (ένα για κάθε εμπλεκόμενο), μπορούμε να ορίσουμε εξαρτήσεις στόχων ανάμεσα στους κόμβους των διαφόρων SGM. Αυτές οι εξαρτήσεις μοντελοποιούνται σαν σταθμισμένες ακμές συνεισφοράς και μπορούν να είναι υπό συνθήκη. Για παράδειγμα, η σταθμισμένη S^N ακμή συνεισφοράς από τον κόμβο $C5$ στον κόμβο $A4$ περιγράφει μία αλληλεξάρτηση μεταξύ των εμπλεκομένων "System Admin" (SGM_{SA}) και "Data Consumer" (SGM_{DC}) και γράφεται υπό την μορφή κανόνα ως:

$$r_1 \;=\; \langle S^P, A4, \{C5\}, 0.7\rangle$$

Ορίζοντας SGM μοντέλα με στοιχεία υπό συνθήκη και ακμές συνεισφοράς μεταξύ αυτών οι οποίες και πάλι μπορεί να είναι υπό συνθήκη, μπορούμε να συνθέσουμε μοντέλα στόχων τα οποία περιγράφουν την αναμενόμενη συμπεριφορά του συστήματος λαμβάνοντας υπόψη τις οπτικές όλων των εμπλεκομένων χάρη στην χρήση των στοιχείων υπό συνθήκη. Ονομάζουμε τα μοντέλα αυτά Μοντέλα Αναμενόμενης Συμπεριφοράς (System Expected Bahavior / SEB) και ένα παράδειγμα δίνεται στο Σχήμα 4-1. Υπό την μορφή ορισμού:

Ορισμός 3 *Ένα Μοντέλο Αναμενόμενης Συμπεριφοράς (SEB) είναι μία έκφραση της μορφής* $\langle M, F, Cond_{SEB}\rangle$ *όπου το* M *είναι ένα σύνολο από n SGM,* F *είναι ένα σύνολο κανόνων συνεισφοράς μεταξύ ζευγών αυτών των SGM και* $Cond_{SEB} : F \mapsto Pow(\mathcal{U} \cup \{\top\})$ *μία συνάρτηση η οποία επιστρέφει το σύνολο των συνθηκών που συνδέονται με κάθε κανόνα συνεισφοράς του* F.

Για παράδειγμα, για το μοντέλο SEB που φαίνεται στο Σχήμα 4-1 τα σύνολα M, F και η συνάρτηση Cond_{SEB} ορίζονται ως:

$$M = \{\text{SGM}_{SA}, \text{SGM}_{DC}\}\ ,\ Cond_{\text{SEB}}(a) = \{\top\}\ ,\ \forall a \in F$$

$$F = \begin{cases} r_1 = \langle S^N, A4, \{C5\}, 0.7\rangle\ ,\ r_2 = \langle D^N, A4, \{C5\}, 0.7\rangle\ , \\ r_3 = \langle S^P, A3, \{B2\}, 0.8\rangle\ ,\ r_4 = \langle D^P, A3, \{B2\}, 0.8\rangle\ , \\ r_5 = \langle S^P, A2, \{B3\}, 1.0\rangle \end{cases}$$

Για να απλοποιήσουμε την ανάλυση των στοιχείων υπό συνθήκη σε ένα SEB μοντέλο με $M = \{\ \langle s1, G_{s1}, R_{s1}, Cond_{s1}\rangle, \cdots, \langle sn, G_{sn}, R_{sn}, Cond_{sn}\rangle\}$ θα χρησιμοποιήσουμε

την συνάρτηση $Cond$ η οποία ορίζεται ως:

$$Cond(a) = \begin{cases} Cond_{s1}(a) & \textbf{if } a \in G_{s1} \cup R_{s1} \\ \vdots & \\ Cond_{sn}(a) & \textbf{if } a \in G_{sn} \cup R_{sn} \\ Cond_{SEB}(a) & \textbf{if } a \in F \end{cases}$$

Επιπλέον, επεκτείνουμε την συνάρτηση $Cond$ για σύνολα στοιχείων του μοντέλου ως ακολούθως: δοθέντος ενός συνόλου $A = \{a_1, \cdots a_n\}$ στόχων ή $gm-$κανόνων $Cond(A) = \bigcup_{i=1}^{n} Cond(a_i)$.

Καθώς κάθε SGM σε ένα μοντέλο SEB μπορεί να έχει πολλές όψεις, και επειδή οι ακμές συνεισφοράς μεταξύ των SGM μπορούν να είναι υπό συνθήκη, πρέπει να επεκτείνουμε το ορισμό του στιγμιότυπου ενός SGM για τα μοντέλα SEB.

Ορισμός 4 *Έστω m ένα SEB που ορίζεται ως $\langle \{\lambda_1, \lambda_2, \ldots \lambda_n\}, F, Cond_{SEB} \rangle$ και U ένα δοθέν ενεργό πλαίσιο. Έστω επίσης F_c ένα σύνολο το οποίο περιέχει όλους τους κανόνες $r = \langle R_t, g_t, G_s, w \rangle \in F$ για τους οποίους:*

1. *$Cond(r) \cap U \neq \emptyset$, και*
2. *$Cond(g_t) \cap U \neq \emptyset$, και*
3. *$Cond(G_s) \cap U \neq \emptyset$.*

Το στιγμιότυπο του m ως προς το U, το οποίο συμβολίζεται ως $m \restriction_U$, είναι το μοντέλο SEB $m' = \langle \{\lambda_1 \restriction_U, \lambda_2 \restriction_U \ldots \lambda_n \restriction_U\}, F_c, Cond_{SEB} \rangle$.

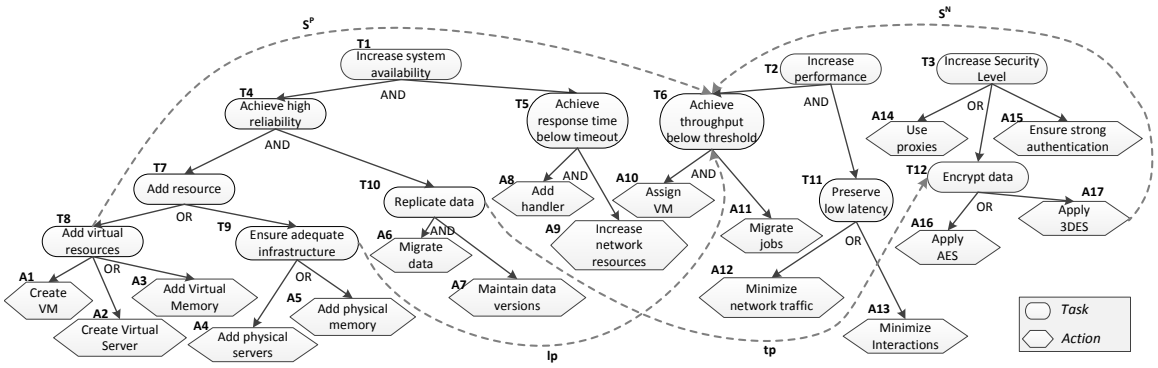
Για παράδειγμα, για το μοντέλο του Σχήματος 4-1, το στιγμιότυπο ως προς το ενεργό πλαίσιο $U1 = \{u_2, \top\}$ θα είναι ένα μοντέλο το οποίο δεν θα περιέχει το κόμβο $C5$, καθώς:

$$Cond(C5) \cap u_1 = \{u_1\} \cap \{u_2, \top\} = \emptyset$$

Επιπλέον, δε θα περιέχει τις S^N και D^N ακμές συνεισφοράς που ξεκινούν από τον κόμβο $C5$, καθώς ο κόμβος αρχής τους, δηλαδή ο κόμβος $C5$, δε θα υπάρχει στο μοντέλο.

4.2 Μοντελοποίηση Εργασιών και Ενεργειών

Προκειμένου να μοντελοποιήσουμε τις ενέργειες που υλοποιούν συγκεκριμένες εργασίες, και να εκφράσουμε τις λογικές και χρονικές εξαρτήσεις που υπάρχουν μεταξύ των



Σχήμα 4-3: Παράδειγμα Μοντέλου Εργασιών και Ενεργειών

ενεργειών και των εργασιών, προτείνουμε ένα μεταμοντέλο το οποίο δανείζεται στοιχεία από την θεωρία των μοντέλων στόχων, και παράλληλα χρησιμοποιεί επιπλέον στοιχεία για την μοντελοποίηση τον λογικών και χρονικών εξαρτήσεων τα οποία αυξάνουν την εκφραστικότητά του. Το προτεινόμενο μεταμοντέλο, το οποίο δίνεται στο Σχήμα 4-4, διατηρεί το σχήμα των AND/OR κανόνων διάσπασης που συναντάται στα μοντέλα στόχων. Στην υπόλοιπη ενότητα δίνουμε αρχικά ένα παράδειγμα στιγμιοτύπου του μεταμοντέλου, ενώ στη συνέχεια περιγράφουμε την σημασιολογία των επιμέρους στοιχείων μοντελοποίησης.

4.2.1 Παράδειγμα Μοντέλου Εργασιών και Ενεργειών

Ένα παράδειγμα μοντέλου εργασιών και ενεργειών δίνεται στο Σχήμα 4-3. Το παράδειγμα περιλαμβάνει τρεις κύριες εργασίες, τις "Increase system availability", "Increase performance", "Increase security level", οι οποίες αναλύονται στη συνέχεια σε απλούστερες εργασίες. Μέσω AND/OR κανόνων διάσπασης, οι σύνθετες εργασίες καταλήγουν να αναλυθούν σε ενέργειες ή οποίες αντιστοιχούν σε στοιχειώδεις και αυτόνομες διεργασίες. Στο παράδειγμα του Σχήματος 4-3, οι εργασίες σχεδιάζονται σαν ελλείψεις ενώ οι ενέργειες σαν εξάγωνα.

Εκτός από τους AND/OR κανόνες διάσπασης, οι κόμβοι εργασιών και ενεργειών μπορούν να συνδέονται μέσω ακμών συνεισφοράς. Οι ακμές αυτές χρησιμοποιούνται για να μοντελοποιήσουν εναλλακτικούς τρόπους για την επίτευξη εργασιών, ή συνθήκες κατά τις οποίες η εκτέλεση μίας εργασίας δεν είναι δυνατή, ως συνέπεια της εκτέλεσης άλλων εργασιών ή ενεργειών Στην προτεινόμενη μοντελοποίηση, μόνο κόμβοι εργασιών μπορούν να εμφανίζονται ως κόμβοι τέλους μίας ακμής συνεισφοράς, καθώς οι ενέργειες αντιστοιχούν σε στοιχειώδεις διαδικασίες για τις οποίες ο κόμβος τους στο μοντέλο μπορεί να είναι είτε αληθής μόνο όταν η ενέργεια έχει εκτελεστεί, σε κάθε άλλη περίπτωση θεωρείται ψευδής. Παραδείγματα ακμών συνεισφοράς είναι η ακμή S^P από τον κόμβο

"Add virtual resources" στον κόμβο "Achieve throughput below threshold" όπου η επίτευξη του πρώτου έχει ως θετική συνέπεια την επίτευξη του δεύτερου, και η ακμή S^N από τον κόμβο "Apply 3DES" encryption στον κόμβο "Achieve throughput below threshold", όπου η εκτέλεση της ενέργειας επιδρά αρνητικά στην επίτευξη της εργασίας.

Τέλος, υπάρχουν δύο επιπλέον τύποι ακμών στο παράδειγμα: *a)* η ακμή $\xrightarrow{lp}$ η οποία μοντελοποιεί την ύπαρξη μίας λογικής προϋπόθεσης από την εργασία "Ensure adequate infrastructure" στην εργασία "Achieve throughput below threshold", υπονοώντας ότι προκειμένου να επιτευχθεί το ζητούμενο throughput, θα πρέπει πρώτα να εξασφαλίσουμε ένα επαρκές επίπεδο πόρων, και *b)* η ακμή $\xrightarrow{tp}$ η οποία μοντελοποιεί την ύπαρξη μίας χρονικής προϋπόθεσης από την εργασία "Replicate Data" στην εργασία "Encrypt Data", υπονοώντας ότι έαν και οι δύο εργασίες πρέπει να επιτευχθούν, τότε η αντιγραφή των δεδομένων θα πρέπει να ολοκληρωθεί πριν την κρυπτογράφηση των δεδομένων.

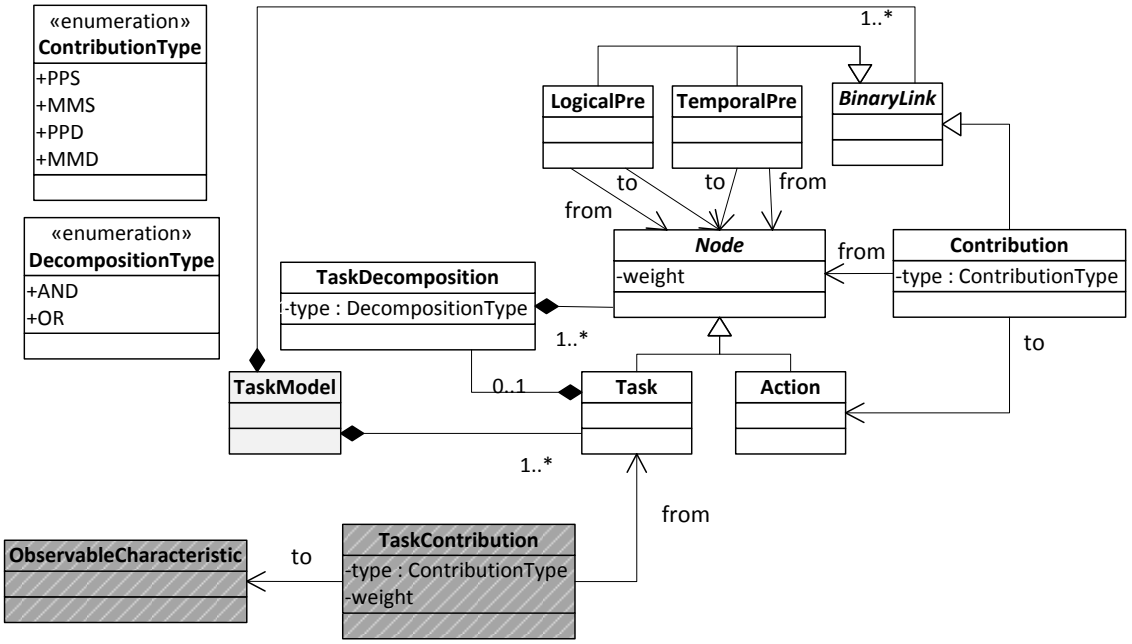
4.2.2 Μεταμοντέλο Εργασιών και Ενεργειών

Το μεταμοντέλο για την μοντελοποίηση των εργασιών και των ενεργειών καθώς και των λογικών και χρονικών εξαρτήσεων που υπάρχουν μεταξύ τους δίνεται στο Σχήμα 4-4. Πιο συγκεκριμένα, τα βασικά στοιχεία ενός μοντέλου εργασιών και ενεργειών είναι:

Κόμβοι: Οι κόμβοι μπορεί να είναι είτε *Εργασίες* (Tasks) είτε *Ενέργειες* (Actions). Οι πρώτοι μπορούν να αναλύονται σε απλούστερες εργασίες ή σε ενέργειες οι οποίες αντιστοιχούν είτε σε εναλλακτικούς τρόπους με τους οποίους μπορεί να επιτευχθεί η αντίστοιχη εργασία ή σε σύνολα εργασιών και ενεργειών που θα πρέπει να εκτελεστούν ταυτόχρονα ώστε να επιτευχθεί η αντίστοιχη εργασία. Αντίθετα, οι κόμβοι ενεργειών αντιστοιχούν σε στοιχειώδεις διεργασίες.

BinaryLinks: Το μεταμοντέλο περιέχει τρεις τύπους ακμών, τις *Logical Preconditions* ($\xrightarrow{lp}$), *Temporal Preconditions* ($\xrightarrow{tp}$), και τις *ακμές συνεισφοράς*. Τα δύο πρώτα είδη ακμών συνδέουν μεταξύ τους εργασίες και ενέργειες και εκφράζουν χρονικές εξαρτήσεις. Πιο συγκεκριμένα, οι ακμές $\xrightarrow{lp}$ μοιάζουν με τις ακμές προτεραιότητας (precedence links) οι οποίες εισάγονται στο [68], και μοντελοποιούν το γεγονός ότι ο κόμβος τέλους μπορεί να εκτελεστεί μόνο όταν ο κόμβος αρχής έχει ήδη εκτελεστεί. Αντίθετα, οι ακμές $\xrightarrow{tp}$ μοντελοποιούν μία ασθενέστερη σχέση προτεραιότητας η οποία υπονοεί ότι στην περίπτωση που και ο κόμβος αρχής και ο κόμβος τέλους συμμετέχουν σε ένα πλάνο, τότε ο κόμβος τέλους πρέπει να εκτελεστεί μετά από τον κόμβο αρχής.

Επιπλέον, όπως και στο [32], θεωρούμε τέσσερις τύπους ακμών συνεισφοράς, τους S^P/S^N που μοντελοποιούν το γεγονός ότι ο κόμβος τέλους ικανοποιείται/αποτυγχάνει όταν ο κόμβος αρχής ικανοποιείται, και τους D^P/D^N που μοντελοποιούν το γεγονός ότι ο κόμβος τέλους αποτυγχάνει/ικανοποιείται όταν ο κόμβος αρχής αποτυγχάνει Παρ΄ όλ΄ αυτά, στο πλαίσιο της παρούσας διατριβής, οι ακμές συνεισφοράς μπορούν να τερματίζουν



Σχήμα 4-4: Μεταμοντέλο Εργασιών και Ενεργειών

μόνο σε ακμές εργασιών καθώς ένας κόμβος ενέργειας μπορεί είτε να εκτελεστεί, οπότε και είναι αληθής, είτε να μην εκτελεστεί, οπότε και είναι ψευδής.

Τέλος, κάποιες εργασίες μπορούν να επηρεάσουν την κατάσταση του συστήματος. Αυτό μοντελοποιείται μέσω των "TaskContribution" ακμών, οι οποίες μοντελοποιούν το τρόπο με τον οποίο η εκτέλεση μίας εργασίας μεταβάλλει την τιμή κάποιου χαρακτηριστικού λειτουργίας του συστήματος.. Και πάλι αυτές οι ακμές συνεισφοράς μπορεί να ανήκουν σε έναν από τους 4 τύπους S^P/S^N ή D^P/D^N, και σε αντίθεση με τις ακμές συνεισφοράς που υπάρχουν μεταξύ εργασιών και ενεργειών, συνοδεύονται από έναν αριθμό στο διάστημα [0,1], τον οποίο αποκαλούμε βάρος και αντιστοιχεί στον βαθμό που η αντίστοιχη ενέργεια επηρεάζει την τιμή του χαρακτηριστικού λειτουργίας. Είναι σημαντικό να σημειωθεί ότι αυτές οι ακμές συνεισφοράς είναι ο τρόπος με τον οποίο τα μοντέλα εργασιών και ενεργειών συνδέονται με τα ασαφή μοντέλα στόχων που περιγράφηκαν στις προηγούμενες ενότητες.

4.2.3 Ορισμοί

Ένα μοντέλο Εργασιών και Ενεργειών (Task and Action / TA model) περιέχει ένα σύνολο από κόμβους ενεργειών και εργασιών οι οποίοι συνδέονται μεταξύ τους μέσω κανόνων διάσπασης ή κανόνων που περιγράφουν δυαδικές σχέσεις. Έτσι, διατυπώνουμε τον παρακάτω ορισμό για τα μοντέλα TA:

Ορισμός 5 *Ένα μοντέλο Εργασιών και Ενεργειών είναι μία έκφραση της μορφής*

$\langle N, R_d, R\rangle$, όπου $N = N_t \cup N_a$ με τα N_t και N_a να συμβολίζουν τα σύνολα των εργασιών και των ενεργειών αντίστοιχα, R_d είναι το σύνολο των κανόνων διάσπασης, και R το σύνολο των δυαδικών κανόνων.

Στον προηγούμενο ορισμό, ένας κανόνας διάσπασης $r_d \in R_d$ περιγράφει τον τρόπο που μία εργασία πατέρας p αναλύεται σε ένα σύνολο σετ $\{c_1, c_2, \cdots c_n\}$ κόμβων εργασιών ή ενεργειών. Πρέπει να υπάρχει ένας κανόνας διάσπασης για κάθε κόμβο εργασίας p, ο οποίος μπορεί να διατυπωθεί ως εξής:

$$r_d = \langle T, p, \{c_1, c_2, \cdots, c_n\}, 1.0\rangle \quad \text{ωηερε} \quad T \in \{\text{AN}\Delta, \text{OP}\}.$$

Για παράδειγμα, ο κόμβος $T6$ ("Achieve throughput below threshold") στο Σχήμα 4-3 αναλύεται στους κόμβους $A10$ ("Assign VM") και $A11$ ("Migrate Jobs"), και επομένως ο αντίστοιχος κανόνας διάσπασης είναι $\langle \text{AND}, T6, \{A10, A11\}, 1.0\rangle$.

Ένας δυαδικός κανόνας $r \in R$ ανάμεσα σε έναν κόμβο αρχής s και έναν κόμβο τέλους t μπορεί να γραφεί στην μορφή:

$$r = \langle T, s, \{t\}, 1.0\rangle \quad \text{ωηερε} \quad T \in \{\lambda\pi, \tau\pi, S^P, S^N, D^P, D^N\}.$$

όπου όπως έχει αναφερθεί παραπάνω, για τους κανόνες συνεισφοράς ο κόμβος αρχής $t \in N_t$. Στο παράδειγμα του Σχήματος 4-3, ο κόμβος $T6$ είναι ο κόμβος τέλους τριών δυαδικών κανόνων, των $\langle S^P, T6, \{T8\}, 1.0\rangle$, $\langle S^N, T6, \{A17\}, 1.0\rangle$ και $\langle \text{lp}, T6, \{T9\}, 1.0\rangle$.

Πρέπει να τονίσουμε ότι το βάρος 1.0 στους παραπάνω κανόνες έχει προστεθεί έτσι ώστε να χρησιμοποιείται για τους κανόνες των μοντέλων TA η ίδια ακριβώς έκφραση που χρησιμοποιείται για τα ασαφή μοντέλα στόχων.

4.3 Μοντέλο Επαλήθευσης και Αποκατάστασης

Ο ορισμός των μοντέλων SEB επιτρέπει στους διάφορους εμπλεκόμενους να ορίσουν τους στόχους που θέλουν να ικανοποιεί το υπό εξέταση σύστημα. Επιπλέον, τα μοντέλα TA επιτρέπουν την μοντελοποίηση των εργασιών που μπορούν να εφαρμοστούν στο σύστημα έτσι ώστε να μεταβληθεί η κατάστασή του. Επίσης, όσο αφορά στις εργασίες και τις ενέργειες, είναι επίσης σημαντικό να μοντελοποιήσουμε τον τρόπο με τον οποίο οι εργασίες και οι ενέργειες επηρεάζουν τους στόχους του συστήματος. Αυτοί οι τύποι εξαρτήσεων μοντελοποιούνται με την χρήση ακμών συνεισφοράς μεταξύ των κόμβων των μοντέλων TA και των κόμβων των μοντέλων SEB και μπορούν να γραφούν ως εξής:

$$d = \langle T, n_{\text{leaf}}, \{n_{\text{task}}\}, 1.0\rangle$$

όπου $T \in \{S^P, S^N, D^P, D^N\}$, n_{task} είναι ένας κόμβος εργασίας του μοντέλου TA , και n_{leaf} είναι ένας κόμβος του μοντέλου SEB. Υπό μορφή ορισμού:

Ορισμός 6 *Ένα μοντέλο Επαλήθευσης και Αποκατάστασης (Verification and Remediation Model / VR model) είναι μία έκφραση της μορφής* $\langle M_{SEB}, M_{TA}, D_{t \to s} \rangle$, *όπου* M_{SEB} *είναι ένα μοντέλο SEB,* M_{TA} *είναι ένα μοντέλο TA , και* $D_{t \to s}$ *είναι ένα σύνολο δυαδικών κανόνων τύπου* S^P, S^N, D^P, *ή* D^N *από τους κόμβους εργασιών του* M_{TA} *στους κόμβους του* M_{SEB}.

Κεφάλαιο 5

Αρχιτεκτονική του Περιβάλλοντος Πλαισίου

Σε αυτό το κεφάλαιο παρουσιάζουμε μία επισκόπηση των δύο βασικότερων διαδικασιών του προτεινόμενου περιβάλλοντος πλαισίου, τις διαδικασίες της επαλήθευσης και της αποκατάστασης.

5.1 Διαδικασία Επαλήθευσης

Σκοπεύουμε να αντιμετωπίσουμε το πρόβλημα ReqRV εισάγοντας ένα περιβάλλον πλαίσιο το οποίο βασίζεται στις αρχές των ασαφών ελεγκτών που παρουσιάζονται στην Ενότητα 2.2.1. Τα βασικά συστατικά μέρη του περιβάλλοντος πλαισίου χωρίζονται σε μέρη χρόνου σχεδίασης (Design-time components) και μέρη χρόνου εκτέλεσης (Run-time components) ανάλογα με την φάση στην οποία χρησιμοποιούνται, και φαίνονται στο Σχήμα 5-1 μαζί με τις αντιστοιχίσεις που υπάρχουν μεταξύ των συνόλων της Εξίσωσης 1.4, και τα στοιχεία ενός ασαφούς ελεγκτή.

Συστατικά Μέρη Χρόνου Σχεδίασης Αρχικά οι εμπλεκόμενοι στο σύστημα ορίζουν ένα σύνολο στόχων που πρέπει να ισχύουν στο σύστημα, μαζί με τις αλληλεξαρτήσεις τους. Αυτό επιτυγχάνεται χρησιμοποιώντας έναν κατάλληλο γραφικό τρόπο απεικόνισης, η σημασιολογία του οποίου περιγράφεται στην Ενότητα 4.1. Το τελικό μοντέλο θα περιέχει ένα σύνολο από (υπό)μοντέλα στόχων, ένα για κάθε εμπλεκόμενο στο σύστημα, μαζί με ένα σύνολο εξαρτήσεων ανάμεσα στα (υπο)μοντέλα. Κάθε υπομοντέλο αντιστοιχεί στο σύνολο D_j της Εξίσωσης 1.4, καθώς αντιπροσωπεύει τους κανόνες που αφορούν στους στόχους του j εμπλεκομένου για διάφορα πλαίσια λειτουργίας.

Το τελικό μοντέλο ("System Expected Behavior Model") μετασχηματίζεται στη συνέχεια σε ένα σύνολο ασαφών κανόνων υπό συνθήκη μέσω μίας συγκεκριμένης διαδικα-

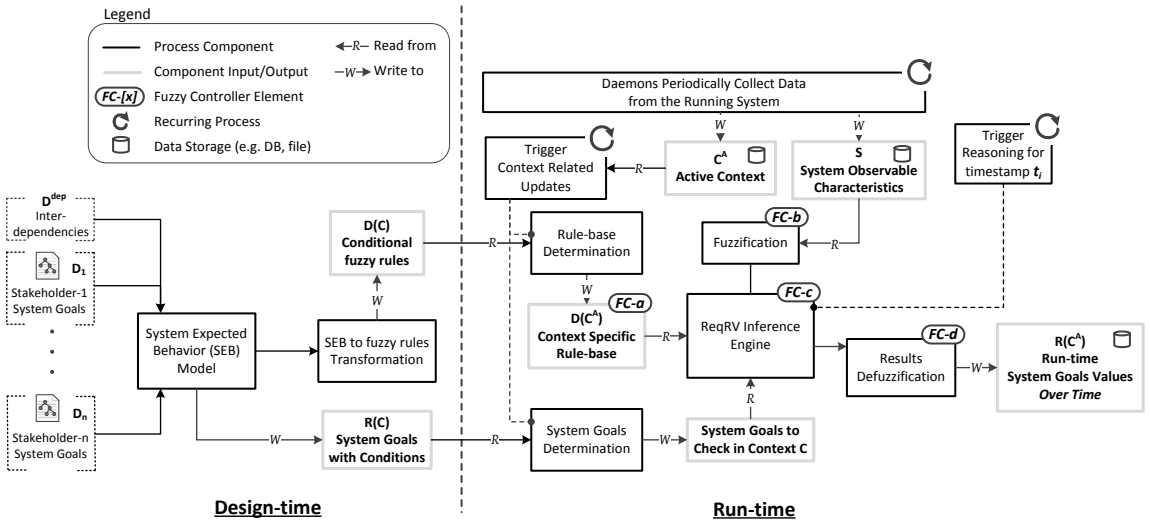
σίας μετασχηματισμού ("SEB to fuzzy rules Transformation" στο Σχήμα 5-1), η οποία περιγράφεται στην Ενότητα 6.3.2. Οι παραγόμενοι κανόνες αντιστοιχούν στο σύνολο D της Εξίσωσης 1.2, και αποκαλούνται κανόνες υπό συνθήκη καθώς ο κάθε κανόνας συνοδεύεται από μια CNF έκφραση η οποία καθορίζει εάν ο κανόνας θα πρέπει να χρησιμοποιηθεί κατά την διαδικασία συμπερασμού. Κατά παρόμοιο τρόπο, εξάγουμε και το πλήρες σύνολο των στόχων για τους οποίους το περιβάλλον πλαίσιο θα πρέπει να υπολογίσει τους βαθμούς ικανοποίησής τους καθώς το σύστημα λειτουργεί σε διάφορα πλαίσια συνθηκών. Αυτό το σύνολο των στόχων αντιστοιχεί στο σύνολο $R(C)$ της Εξίσωσης 1.2, και κάθε στόχος του συνόλου συνοδεύεται από μία συνθήκη η οποία καθορίζει εάν ο στόχος θα πρέπει να ελεγχθεί στο τρέχον πλαίσιο λειτουργίας ή όχι.

Συστατικά Μέρη Χρόνου Εκτέλεσης Κατά τον χρόνο εκτέλεσης παρακολουθούμε το σύστημα και περιοδικά συλλέγουμε δεδομένα τα οποία αντικατοπτρίζουν την κατάσταση του συστήματος που βρίσκεται σε λειτουργία ("System Observable Characteristics"), και το ενεργό πλαίσιο μέσα στο οποίο το σύστημα λειτουργεί ("Active Context"). Η εξαγωγή και συλλογή αυτών των τιμών μπορεί να γίνει χρησιμοποιώντας μία τεχνική ανάλυσης των logs του συστήματος όπως αυτές που περιγράφονται στα [60, 111] και δεν αποτελεί αντικείμενο της παρούσας διατριβής.

Επιπλέον, είτε περιοδικά είτε όταν ένα συγκεκριμένο ποσοστό των δεδομένων που αντικατοπτρίζει της αλλαγές στο ενεργό πλαίσιο μεταβληθεί, δύο διαδικασίες τίθενται σε λειτουργία προκειμένου να πραγματοποιηθούν ενημερώσεις σχετικές με τις αλλαγές στο ενεργό πλαίσιο ("Trigger Context Related Updates"). Η πρώτη διαδικασία ("Rule Base Determination") αποτιμά τις CNF εκφράσεις και καθορίζει το υποσύνολο των ασαφών κανόνων που θα πρέπει να χρησιμοποιηθούν από τον μηχανισμό συμπερασμού ("Context Specific Rule-base"). Αυτή η διαδικασία απαιτεί γραμμικό χρόνο ως προς τον συνολικό αριθμό των CNF εκφράσεων, και το παραγόμενο υποσύνολο των ασαφών κανόνων αποτελεί την βάση γνώσης για τον μηχανισμό συμπερασμού του προβλήματος ReqRV. Η δεύτερη διαδικασία ("System Goals Determination") καθορίζει το σύνολο των στόχων που πρέπει να ελεγχθούν για το ενεργό πλαίσιο, και κατά συνέπεια τους στόχους για τους οποίους θα πρέπει να υπολογιστούν οι τιμές τους. Αυτή η διαδικασία απαιτεί γραμμικό χρόνο ως προς τον αριθμό των στόχων στο αρχικό μοντέλο.

Τέλος, σε τακτά χρονικά διαστήματα ξεκινά η διαδικασία της αποτίμησης των στόχων του συστήματος ("Trigger Reasoning for timestamp t_i"), επιτρέποντας έτσι στον τελικό χρήστη να μπορεί να ελέγχει εάν το σύστημα ικανοποιεί τους επιθυμητούς στόχους.

Αρχικά τα χαρακτηριστικά λειτουργίας του συστήματος ("System Observable Characteristics") δηλαδή το σύνολο S της Εξίσωσης 1.4, μετασχηματίζονται σε ασαφή γεγονότα μέσω μίας κατάλληλης διαδικασίας ασαφοποίησης ("Fuzzification"). Τα ασαφή γεγονότα και οι κανόνες αποτελούν τη βάση γνώσης η οποία στη συνέχεια θα χρησιμο-



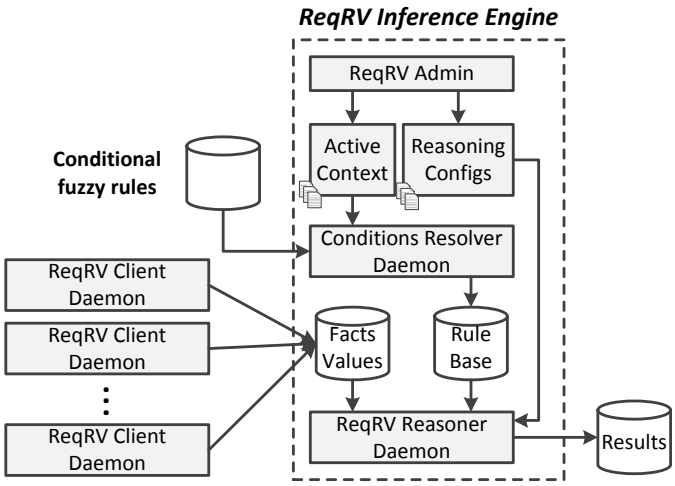
Σχήμα 5-1: Περιγραφή της προτεινόμενης διαδικασίας για το πρόβλημα ReqRV

ποιηθεί από την μηχανή συμπερασμού ("ReqRV Inference Engine") προκειμένου να υπολογίσει τους βαθμούς συμμετοχής για όλους τους κόμβους. Μία κατάλληλη διαδικασία απο-ασαφοποίησης ("Results Defuzzification") μπορεί να χρησιμοποιηθεί στη συνέχεια ώστε να υπολογίζει για κάθε χρονική στιγμή t_i, και για κάθε στόχο του συστήματος που ανήκει στο σύνολο $R(C^A)$ μία τιμή αληθείας (δηλαδή τον βαθμό ικανοποίησης για κάθε στόχο του συστήματος).

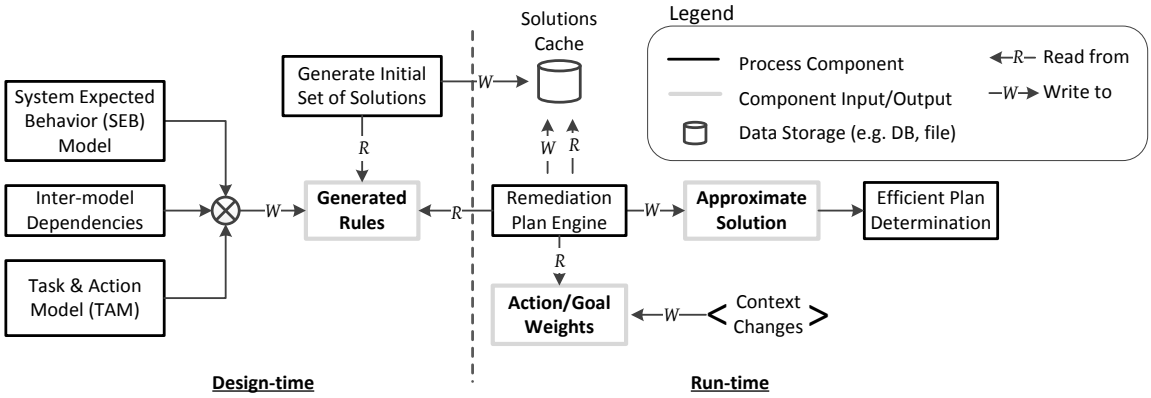
5.1.1 Αρχιτεκτονική της Μηχανής Συμπερασμού

Ένα διάγραμμα της αρχιτεκτονική της μηχανής συμπερασμού που χρησιμοποιείται για την επαλήθευση ορθής λειτουργίας του συστήματος δίνεται στο Σχήμα 5-2. Όπως φαίνεται στην εικόνα, τα γεγονότα που αντιστοιχούν στα χαρακτηριστικά λειτουργίας του συστήματος συλλέγονται από ένα σύνολο δαιμόνων ("Client Daemons") οι οποίοι είναι υπεύθυνοι να καταγράφουν περιοδικά τις τιμές των χαρακτηριστικών λειτουργίας από τα διάφορα συστατικά μέρη του συστήματος (δηλαδή βάσεις δεδομένων, εξυπηρετητές). Εν συνεχεία οι τιμές που έχουν συλλεχθεί αποθηκεύονται σε μία βάση δεδομένων ("Events DB") η οποία χρησιμοποιείται για αυτόν τον λόγο.

Τέλος, ο "Reasoner Daemon" χρησιμοποιεί τις πιο πρόσφατες τιμές για κάθε χαρακτηριστικό λειτουργίας από την "Events DB" ώστε να μπορεί να έπειτα να εξάγει τους βαθμούς ικανοποίησης των απαιτούμενων στόχων. Είναι δηλαδή υπεύθυνος να τρέχει *περιοδικά* τη διαδικασία συμπερασμού και να αποθηκεύει τα αποτελέσματα για κάθε χρονική στιγμή στην "Results" DB. Μπορούμε στη συνέχεια να τρέξουμε ερωτήματα σε αυτήν την βάση, π.χ. μέσω μίας κατάλληλης δικτυακής εφαρμογής, ώστε να ελέγξουμε εάν το σύστημα ικανοποιεί του απαιτούμενους στόχους.



Σχήμα 5-2: Γραφική αναπαράσταση της μηχανής συμπερασμού που χρησιμοποιείται για την διαδικασία της επαλήθευσης



Σχήμα 5-3: Γραφική απεικόνιση της διαδικασίας αποκατάστασης

5.2 Διαδικασία Αποκατάστασης

Παρόμοια με την διαδικασία της επαλήθευσης, υπάρχουν κομμάτια της διαδικασίας αποκατάστασης που ολοκληρώνονται στη φάση της σχεδίασης και άλλα κομμάτια που ολοκληρώνονται στη φάση της εκτέλεσης, όταν εντοπιστούν μέσω της διαδικασίας επαλήθευσης στόχοι του συστήματος που δεν ικανοποιούνται. Τα βασικά μέρη της διαδικασίας αποκατάστασης φαίνονται στο Σχήμα 5-3 και κατηγοριοποιούνται και σε αυτήν την περίπτωση σε συστατικά μέρη χρόνου σχεδίασης και συστατικά μέρη χρόνου εκτέλεσης ανάλογα με τη χρονική στιγμή που χρησιμοποιούνται.

Συστατικά Μέρη Χρόνου Σχεδίασης Αρχικά τα μοντέλα SEB και TA μαζί με τις εξαρτήσεις που υπάρχουν μεταξύ τους, δηλαδή οι ακμές που μοντελοποιούν το πώς επηρεάζεται ή επίτευξη ή η αποτυχία ικανοποίησης ενός στόχου δεδομένου ότι συγ-

κεκριμένες ενέργειες έχουν εκτελεστεί, μετασχηματίζονται σε ένα κατάλληλο σύνολο κανόνων το οποίο θα επιτρέψει την εφαρμογή μίας αυτόματης διαδικασίας συμπερασμού. Στο σημείο αυτό, μπορούμε να χρησιμοποιήσουμε τους κανόνες προκειμένου να εξάγουμε ένα σύνολο από ενέργειες οι οποίες όταν εκτελεστούν στο σύστημα μπορούν να το οδηγήσουν σε μία κατάσταση στην οποία ιδανικά όλοι οι στόχοι ικανοποιούνται. Αυτοί οι συνδυασμοί ενεργειών αποτελούν υποψήφιες λύσεις για το πρόβλημα της αποκατάστασης, και θα χρησιμοποιηθούν κατά τον χρόνο εκτέλεσης από την μηχανή συμπερασμού ("Remediation Plan Engine") ώστε να δημιουργηθεί ένας αρχικός πληθυσμός λύσεων για τον γενετικό αλγόριθμο που χρησιμοποιείται για την δημιουργία των πλάνων αποκατάστασης.

Συστατικά Μέρη Χρόνου Εκτέλεσης Κατά τον χρόνο εκτέλεσης, θεωρούμε ότι κάθε κόμβος στόχου και ενέργειας συνοδεύεται από έναν ακέραιο αριθμό, ο οποίος στην περίπτωση των στόχων είναι θετικός και αντιστοιχεί στον βαθμό σημαντικότητας του στόχου, ενώ στην περίπτωση των ενεργειών είναι αρνητικός και αντιστοιχεί στο κόστος εκτέλεσης της ενέργειας. Λαμβάνοντας υπόψη αυτές τις τιμές, και χρησιμοποιώντας ένα αρχικό σύνολο πιθανών λύσεων, η μηχανή συμπερασμού ("Remediation Plan Engine") πρέπει να μπορεί να προτείνει πλάνα αποκατάστασης, δηλαδή σύνολα ενεργειών οι οποίες όταν εκτελεστούν σειριακά ή παράλληλα μπορούν να οδηγήσουν το σύστημα σε μία κατάσταση στην οποία ικανοποιούνται οι πιο σημαντικοί στόχοι (όπως αυτή σηματοδοτείται από τα αντίστοιχα βάρη) ή τουλάχιστον σε μία κατάσταση που ο βαθμός ικανοποίησης των στόχων είναι μεγαλύτερος από ό,τι στην τρέχουσα κατάσταση.

Το πλάνο αποκατάστασης που προτείνεται από την μηχανή αποκατάστασης ("Remediation Plan Engine") δημιουργείται σε δύο βήματα. Αρχικά η μηχανή πρέπει να εντοπίσει το συνδυασμό των ενεργειών που μπορούν να οδηγήσουν το σύστημα σε μία καλύτερη κατάσταση ("Approximate Solution" στο Σχήμα 5-3). Εν συνεχεία, επιλύοντας τις χρονικές και λογικές εξαρτήσεις που μπορεί να υπάρχουν μεταξύ των ενεργειών είναι δυνατό να δημιουργηθεί το πλάνο αποκατάστασης. Το πλάνο καθορίζει ποιες ενέργειες πρέπει να εκτελεστούν είτε σειριακά είτε παράλληλα, και επίσης με ποια σειρά, έτσι ώστε να μην παραβιάζεται κανένας από τους περιορισμούς.

Κεφάλαιο 6

Εξαγωγή Βάσης Γνώσης από Μοντέλα

Τα μοντέλα SEB και TA που παρουσιάστηκαν στο προηγούμενο κεφάλαιο προσφέρουν το θεωρητικό υπόβαθρο για το πρόβλημα ReqRV και το πρόβλημα αποκατάστασης αντίστοιχα. Τα μοντέλα SEB επιτρέπουν στους εμπλεκόμενους στο σύστημα να καταγράψουν τους στόχους τους κάτω από διάφορες συνθήκες , ενώ τα μοντέλα TA επιτρέπουν τον ορισμό των ενεργειών και των εργασιών που μπορούν να εκτελεστούν στο σύστημα έτσι ώστε να το οδηγήσουν σε μία πιο σταθερή κατάσταση όσον αφορά στην επίτευξη των στόχων που ορίζονται στο μοντέλο SEB. Και τα δύο μοντέλα στοχεύουν στο να παρέχουν ένα μέσο για την καταγραφή την γνώσης που σχετίζεται με τους στόχους του συστήματος και με την αποκατάστασή τους, έτσι ώστε να μπορούν να χρησιμοποιηθούν προκειμένου να αυτοματοποιήσουν τις διαδικασίες της επαλήθευσης και της αποκατάστασης. Και οι δύο αυτέ διαδικασίες μπορούν να αναχθούν σε διαδικασίες συμπερασμού στα αντίστοιχα μοντέλα στόχων, είτε χρησιμοποιώντας μία τεχνική για την διάδοση των τιμών από τα φύλλα προς τις ρίζες (στην περίπτωση της επαλήθευσης), είτε μία τεχνική συμπερασμού με κατεύθυνση από τις ρίζες προς τα φύλλα (στην περίπτωση της αποκατάστασης). Παρ΄ όλ΄ αυτά, πριν προχωρήσουμε με την περιγραφή των μηχανισμών συμπερασμού που χρησιμοποιούνται στο πλαίσιο αυτής της διατριβής, θα περιγράψουμε το σύνολο των κανόνων (δηλαδή τη βάση γνώσης) που μπορούν να παραχθούν από ένα μοντέλο SEB ή TA. Στο εξής όταν αναφερόμαστε σε μοντέλα εννοούμε ένα από τα SEB, TA, ή VR εκτός αν αναφέρεται διαφορετικά.

6.1 Κανόνες Μοντέλων Στόχων

Στα μοντέλα SEB οι εξαρτήσεις ανάμεσα στους κόμβους του μοντέλου γράφονται επίσημα στην μορφή κανόνων μοντέλων στόχων (βλέπε Ορισμό 2). Ο ίδιος συμβολισμός χρησιμοποιείται επίσης για τους κανόνες διάσπασης και τους δυαδικούς κανόνες στα μοντέλα TA , και τους κανόνες εξαρτήσεων μεταξύ των μοντέλων SEB και των μοντέλων TA όταν ορίζεται ένα μοντέλο VR. Έτσι, όλες οι εξαρτήσεις μεταξύ κόμβων του ιδίου μοντέλου SEB ή του ιδίου μοντέλου TA , όπως και εξαρτήσεις μεταξύ των κόμβων ενός SEB και ενός TA μοντέλου μπορούν να γραφούν σαν:

$$\langle T, g_t, G_s, w \rangle \tag{6.1}$$

όπου $T \in \{\text{AND}, \text{OR}, S^P, D^P, S^N, D^N, \text{lp}, \text{tp}\}$, g_t είναι ο κόμβος στόχος, και G_s το σύνολο των κόμβων αρχής, το οποίο στην περίπτωση των κανόνων τύπου S^P, D^P, S^N, D^N, lp, και tp περιέχει ένα μόνο στοιχείο, τον μοναδικό κόμβο αρχής.

Στην υπόλοιπη ενότητα θα παρουσιάσουμε ένα σύνολο από τελεστές για τα μοντέλα του Κεφαλαίου 4 και τους κόμβους τους, οι οποίοι θα χρειαστούν αργότερα προκειμένου να περιγράψουμε την παραγωγή των ασαφών κανόνων και των εκφράσεων CNF από τα μοντέλα.

Σύνολα Κανόνων Μοντέλων Στόχων Ο τελεστής $\mathcal{R}$ πάνω σε ένα μοντέλο χρησιμοποιείται για να ορίσει όλους τους κανόνες που υπάρχουν σε ένα μοντέλο γραμμένους στην μορφή της Εξίσωσης 6.1. Ανάλογα με τον τύπο του μοντέλου, ο τελεστής επιστρέφει ένα σύνολο κανόνων όπως αυτό ορίζεται στη συνέχεια.

- Για ένα μοντέλο SEB που αποτελείται από n μοντέλα SGM και δίνεται στην μορφή $m = \langle \{\lambda_1, \lambda_2, \ldots \lambda_n\}, F, \text{δνδ} \rangle$, όπου $\lambda_i = \langle i, G_i, R_i, Cond \rangle$, το σύνολο όλων των κανόνων ορίζεται ως:

$$\mathcal{R}(m) = \bigcup_{i=1}^{n} R_i \cup F \tag{6.2}$$

- Για ένα μοντέλο TA που δίνεται στη μορφή $m = \langle N, R_d, R \rangle$, το σύνολο όλων των κανόνων ορίζεται ως:

$$\mathcal{R}(m) = R_d \cup R \tag{6.3}$$

- Για ένα μοντέλο VR που δίνεται στη μορφή $m = \langle M_{SEB}, M_{TA}, D_{t \to s} \rangle$, το σύνολο

όλων των κανόνων ορίζεται ως:

$$\mathcal{R}(m) = \mathcal{R}(M_{SEB}) \cup \mathcal{R}(M_{TA}) \cup D_{t \to s} \tag{6.4}$$

Σύνολο Κόμβων Ο τελεστής $\mathcal{G}$ πάνω σε ένα μοντέλο χρησιμοποιείται για να ορίσει το σύνολο όλων των κόμβων στο μοντέλο. Ανάλογα με τον τύπο του μοντέλου, ο τελεστής επιστρέφει ένα σύνολο κόμβων όπως ορίζεται στη συνέχεια.

- Για ένα μοντέλο SEB που αποτελείται από n μοντέλα SGM και δίνεται στην μορφή $m = \langle \{\lambda_1, \lambda_2, \ldots \lambda_n\}, F, \text{Cond} \rangle$, όπου $\lambda_i = \langle i, G_i, R_i, Cond \rangle$, το σύνολο όλων των κόμβων ορίζεται ως:

$$\mathcal{G}(m) = \bigcup_{i=1}^{n} G_i \tag{6.5}$$

- Για ένα μοντέλο TA που δίνεται στη μορφή $m = \langle N, R_d, R \rangle$, το σύνολο όλων των κόμβων ορίζεται ως:

$$\mathcal{G}(m) = N \tag{6.6}$$

- Για ένα μοντέλο VR που δίνεται στη μορφή $m = \langle M_{SEB}, M_{TA}, D_{t \to s} \rangle$, το σύνολο όλων των κόμβων ορίζεται ως:

$$\mathcal{G}(m) = \mathcal{G}(M_{SEB}) \cup \mathcal{G}(M_{TA}) \tag{6.7}$$

Σύνολα κόμβων από τα οποία εξαρτάται ένας κόμβος Δοθέντος ενός μοντέλου m και ενός κόμβου b σε αυτό το μοντέλο, χρησιμοποιούμε τον τελεστή $\mathcal{N}^{[T]}$ για να ορίσουμε το σύνολο των κόμβων από τους οποίους εξαρτάται ο κόμβος b μέσω ενός κανόνα τύπου T, όπου $T \in \{\text{AND}, \text{OR}, S^P, D^P, S^N, D^N, \text{lp}, \text{tp}\}$. Πιο συγκεκριμένα:

$$\mathcal{N}^{[T]}(m, b) = \{s \in \mathcal{G}(m) | \exists r = \langle T, s, \{b\}, 1.0 \rangle \in \mathcal{R}(m)\} \tag{6.8}$$

που περιέχει τους κόμβους αρχής όλων των κανόνων τύπου T για τους οποίους ο κόμβος b είναι ο κόμβος τέλους.

Επιπλέον, χρησιμοποιούμε τον τελεστή $\mathcal{N}$ για να ορίσουμε το σύνολο όλων των

κόμβων από τους οποίους εξαρτάται η τιμή του κόμβου b, που ορίζεται ως:

$$\mathcal{N}(m,b) = \mathcal{N}^{[\mathrm{AN}\Delta]}(m,b) \cup \mathcal{N}^{[\mathrm{OP}]}(m,b) \cup \mathcal{N}^{[S^P]}(m,b) \cup \mathcal{N}^{[S^N]}(m,b) \cup \\ \cup \mathcal{N}^{[D^P]}(m,b) \cup \mathcal{N}^{[D^N]}(m,b) \cup \mathcal{N}^{[\lambda\pi]}(m,b) \tag{6.9}$$

Είναι σημαντικό να τονίσουμε ότι οι κανόνες tp στους οποίους ο κόμβος b εμφανίζεται σαν ο κόμβος τέλους δεν περιέχονται στο σύνολο που δίνεται παραπάνω, καθώς αυτός ο τύπος κανόνα απλά αναπαριστά μία χρονική εξάρτηση και δεν επηρεάζει την τιμή του κόμβου b.

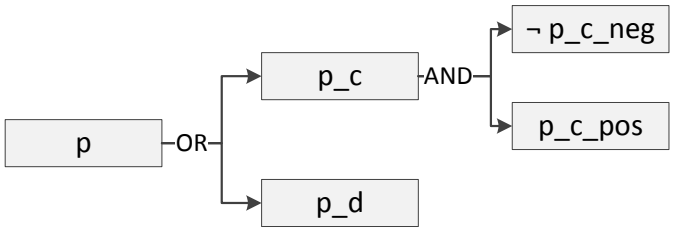
Κόμβοι Φύλλα ενός Μοντέλου Ο τελεστής $\mathcal{L}$ πάνω σε ένα μοντέλο m χρησιμοποιείται για να ορίσει το σύνολο όλων των φύλλων σε ένα δοθέν μοντέλο. Πιο συγκεκριμένα:

$$\mathcal{L}(m) = \{s \in \mathcal{G}(m) | \mathcal{N}(m,s) = \emptyset\}$$

6.2 Κανόνες Δυαδικής Λογικής

Σε αυτήν την ενότητα περιγράφουμε το τρόπο με τον οποίο εξάγεται ένα σύνολο κανόνων δυαδικής λογικής από ένα μοντέλο. Πιο συγκεκριμένα, για ένα μοντέλο m το οποίο περιέχει μόνο crisp κόμβους μπορεί να εξαχθεί ένα σύνολο κανόνων δυαδικής λογικής (Boolean rules) το οποίο εμπεριέχει όλους τους περιορισμούς και τις εξαρτήσεις που περιγράφονται από το μοντέλο. Η σημασιολογία αυτών των κανόνων καθώς και η αντιστοίχισή τους σε εκφράσεις CNF παρουσιάζονται στο [101], και για λόγους πληρότητας περιγράφονται συνοπτικά στον Πίνακα 6.1. Μία ενιαία έκφραση CNF για όλο το μοντέλο μπορεί στη συνέχεια εύκολα να εξαχθεί θεωρώντας την σύζευξη των επιμέρους CNF εκφράσεων που αντιστοιχούν σε κάθε κανόνα δυαδικής λογικής.

Το πρώτο βήμα της διαδικασίας εξαγωγής της CNF έκφρασης περιλαμβάνει τον υπολογισμό δύο συνόλων για κάθε εσωτερικό crisp κόμβο p του μοντέλου. Αυτά τα σύνολα είναι τα$N^{\mathrm{pos}}(p)$και $N^{\mathrm{neg}}(p)$ με την ερμηνεία ότι το $N^{\mathrm{pos}}(p)$ είναι το σύνολο όλων των κόμβων που συνεισφέρουν θετικά στον κόμβο p, ενώ αντίθετα το $N^{\mathrm{neg}}(p)$ είναι το σύνολο όλων των κόμβων οι οποίοι συνεισφέρουν αρνητικά στον κόμβο p. Τα σύνολα



Σχήμα 6-1: Γραφική αναπαράσταση των AND/OR κανόνων στην περίπτωση που $N^{pos}(p) \neq \emptyset$, $N^{neg}(p) \neq \emptyset$, $N^{[\text{lp}]}(p) = \emptyset$, και υπάρχει ένας κανόνας διάσπασης r_d.

Πίνακας 6.1: Αντιστοίχιση των κανόνων δυαδικής λογικής σε εκφράσεις CNF

Boolean Rule	Equivalent Constraints	CNF Clauses
$o = \text{AND}(i_1, i_2, \cdots, i_n)$	$\neg i_1 \Rightarrow \neg o, \cdots, \neg i_n \Rightarrow \neg o,$ $i_1 \wedge i_2 \wedge \cdots \wedge i_n \Rightarrow o$	$(i_1 \vee \neg o) \wedge \cdots \wedge (i_n \vee \neg o) \wedge$ $(\neg i_1 \vee \neg i_2 \vee \cdots \vee \neg i_n \vee o)$
$o = \text{OR}(i_1, i_2, \cdots, i_n)$	$i_1 \Rightarrow o, \cdots, i_n \Rightarrow o,$ $i_1 \wedge i_2 \wedge \cdots \wedge i_n \Rightarrow o$	$(\neg i_1 \vee o) \wedge \cdots \wedge (\neg i_n \vee o) \wedge$ $(i_1 \vee i_2 \vee \cdots \vee i_n \vee \neg o)$

αυτά ορίζονται ως εξής:

$$\begin{aligned} N^{\text{pos}}(p) &= N^{[S^P]}(p) \cup N^{[D^N]}(p) = \\ &= \{e_1, \cdots, e_k\} \cup \{f_1, \cdots, f_l\} \qquad (6.10) \\ N^{\text{neg}}(p) &= N^{[S^N]}(p) \cup N^{[D^P]}(p) = \\ &= \{g_1, \cdots, g_m\} \cup \{h_1, \cdots, h_o\} \qquad (6.11) \end{aligned}$$

όπου $e_1, \cdots, e_k$ είναι οι κόμβοι αρχής όλων των S^P ακμών συνεισφοράς προς τον κόμβο p, $f_1, \cdots, f_l$ είναι οι κόμβοι αρχής όλων των D^N ακμών συνεισφοράς προς τον κόμβο p, $g_1, \cdots, g_m$ είναι οι κόμβοι αρχής όλων των S^N ακμών συνεισφοράς προς τον κόμβο p, και $h_1, \cdots, h_o$ είναι οι κόμβοι αρχής όλων των D^P ακμών συνεισφοράς προς τον κόμβο p. Για παράδειγμα, για τον κόμβο $T6$ του Σχήματος 4-3 $N^{\text{pos}}(T6) = \{T8\}$, και $N^{\text{neg}}(T6) = \{A17\}$.

Αυτά τα δύο σύνολα μαζί με τον κανόνα διάσπασης $r_d = \langle AND, p, \{c_1, c_2, \cdots, c_n\}\rangle$ ή $r_d = \langle OR, p, \{c_1, c_2, \cdots, c_n\}\rangle$ (στην περίπτωση που ο κόμβος p είναι ένας σύνθετος κόμβος με παιδιά τους κόμβους $\{c_1, c_2, \cdots, c_n\}$), και το σύνολο $N^{[lp]}(p)$ καθορίζουν την CNF έκφραση που πρέπει να δημιουργηθεί για τον κόμβο p.

Ανάλογα με το αν υπάρχει ένας κανόνας διάσπασης και εάν τα σύνολα $N^{\text{pos}}(p)$, $N^{\text{neg}}(p)$, και $N^{[\text{lp}]}(p)$ είναι κενά, εξάγεται ένα διαφορετικό σύνολο κανόνων όπως αυτό φαίνεται στους Πίνακες 6.2 και 6.3. Επιπλέον, για τις ψευδο-μεταβλητές p_c_pos (η οποία

Πίνακας 6.2: Εξαγωγή κανόνων AND/OR για έναν crisp κόμβο p ($r_d = \langle T, p, \{c_1, c_2, \cdots, c_n\}\rangle$) όταν $N^{[\text{lp}]}(p) = \emptyset$

$N^{pos}(p)$	$N^{neg}(p)$	Generated AND/OR Rules	
		r_d exists	NO r_d exists
$= \emptyset$	$= \emptyset$	$p \leftarrow \text{T}(c_1, c_2, \cdots, c_n)$	-
$= \emptyset$	$\neq \emptyset$	$p \leftarrow \text{AND}(\text{p\_d}, \neg\text{p\_c\_neg})$	$p \leftarrow \text{AND}(\text{True}, \neg\text{p\_c\_neg})$
$\neq \emptyset$	$= \emptyset$	$p \leftarrow \text{OR}(\text{p\_d}, \text{p\_c\_pos})$	$p \leftarrow \text{OR}(\text{False}, \text{p\_c\_pos})$
$\neq \emptyset$	$\neq \emptyset$	$p \leftarrow \text{OR}(\text{p\_d}, \text{p\_c})$ $\text{p\_c} \leftarrow \text{AND}(\text{p\_c\_pos}, \neg\text{p\_c\_neg})$	$p \leftarrow \text{AND}(\text{p\_c\_pos}, \neg\text{p\_c\_neg})$

Πίνακας 6.3: Εξαγωγή κανόνων AND/OR για έναν crisp κόμβο p ($r_d = \langle T, p, \{c_1, c_2, \cdots, c_n\}\rangle$) όταν $N^{[\text{lp}]}(p) = \{b_1 \cdots b_q\}$

$N^{pos}(p)$	$N^{neg}(p)$	Generated AND/OR Rules	
		r_d exists	NO r_d exists
$= \emptyset$	$= \emptyset$	$p \leftarrow \text{AND}(b_1, b_2, \cdots, b_q, p\_d)$	$p \leftarrow \text{AND}(b_1, b_2, \cdots, b_q, p\_leaf)$
$= \emptyset$	$\neq \emptyset$	$p \leftarrow \text{AND}(\text{p\_}\ell, \neg\text{p\_c\_neg})$ $\text{p\_}\ell \leftarrow \text{AND}(b_1 \cdots b_q, \text{p\_d})$	-
$\neq \emptyset$	$= \emptyset$	$p \leftarrow \text{OR}(\text{p\_}\ell, \text{p\_c\_pos})$ $\text{p\_}\ell \leftarrow \text{AND}(b_1 \cdots b_q, \text{p\_d})$	-
$\neq \emptyset$	$\neq \emptyset$	$p \leftarrow \text{OR}(\text{p\_}\ell, \text{p\_c})$ $\text{p\_c} \leftarrow \text{AND}(\text{p\_c\_pos}, \neg\text{p\_c\_neg})$ $\text{p\_}\ell \leftarrow \text{AND}(b_1 \cdots b_q, \text{p\_d})$	-

παίρνει την τιμή true εάν υπάρχει τουλάχιστον ένας κόμβος που συνεισφέρει θετικά στον κόμβο p), p_c_neg (η οποία παίρνει την τιμή true εάν υπάρχει τουλάχιστον ένας κόμβος που συνεισφέρει αρνητικά στον κόμβο p), και p_d (η οποία παίρνει την τιμή true εάν ο κόμβος p υποστηρίζεται από τους κόμβους παιδιά του) που εμφανίζονται στους Πίνακες 6.2 και 6.3, ισχύουν οι ακόλουθες σχέσεις:

$$\text{p\_c\_pos} = \text{OR}(e_1, \cdots, e_k, \neg f_1, \cdots, \neg f_l) \tag{6.12}$$

$$\text{p\_c\_neg} = \text{OR}(g_1, \cdots, g_m, \neg h_1, \cdots, \neg h_o) \tag{6.13}$$

$$\text{p\_d} = \text{T}(c_1, c_2, \cdots, c_n) \tag{6.14}$$

όπου το T είναι ο τύπος του κανόνα διάσπασης και είναι είτε AND είτε OR, $c_1, c_2, \cdots, c_n$ είναι οι κόμβοι παιδιά του κόμβου p, και οι κόμβοι e_i, f_i, g_i και h_i αντιστοιχούν σε αυτούς που ορίζονται στις Εξισώσεις 6.10 και 6.11.

Είναι σημαντικό να τονίσουμε ότι το σύνολο p_c_pos αντικαθίσταται από την μεταβλητή e_1 στην περίπτωση που $k = 1$ και $l = 0$, ή με την μεταβλητή $\neg f_1$ στην περίπτωση που $k = 0$ και $l = 1$. Κατά παρόμοιο τρόπο, το σύνολο p_c_neg αντικαθίσταται από την μεταβλητή g_1 στην περίπτωση που $m = 1$ και $o = 0$, ή με την μεταβλητή $\neg h_1$ στην περίπτωση που $m = 0$ και $o = 1$. Επιπλέον, εάν και οι δύο δείκτες k και l (ή στην δεύτερη περίπτωση οι m και o) είναι ίσοι με μηδέν, το σύνολο N^{pos} (το σύνολο N^{neg}) είναι κενό, και η ψευδο-μεταβλητή p_c_pos (p_c_neg) δεν εμφανίζεται στους κανόνες.

Αρχικά παρουσιάζουμε στον Πίνακα 6.2 τους κανόνες που εξάγονται στην περίπτωση που $N^{[\text{lp}]}(p) = \emptyset$. Επίσης, μία γραφική αναπαράσταση των εξαρτήσεων μεταξύ των p, p_c_pos, p_c_neg, και p_d στην περίπτωση που $N^{pos}(p) \neq \emptyset$, $N^{neg}(p) \neq \emptyset$, $N^{[\text{lp}]}(p) = \emptyset$, και υπάρχει ένας κανόνας διάσπασης r_d φαίνεται στο Σχήμα 6-1. Το Σχήμα 6-1, δείχνει ότι ένας κόμβος p μπορεί να αποτιμηθεί μέσω της λογικής διάσπασής του, ή μέσω των των κανόνων συνεισφοράς ή και μέσω και των δύο. Επιπλέον, για να χρησιμοποιηθούν οι ακμές συνεισφοράς για την αποτίμηση, πρέπει να ληφθούν υπόψη οι παράγοντες που συνεισφέρουν θετικά και αρνητικά (υπό την μορφή ψευδο-κόμβων). Στη συνέχεια, παρουσιάζουμε στον Πίνακα 6.3 τους κανόνες που εξάγονται στην περίπτωση που $N^{[\text{lp}]}(p) = \{b_1 \cdots b_q\}$.

Για παράδειγμα, για τον κόμβο $T6$ του Σχήματος 4-3 για τον οποίον $N^{[\text{lp}]}(T6) = \{T9\}$, $N^{\text{pos}}(T6) = N^{[S^P]}(T6) = \{T8\}$, και $N^{\text{neg}}(T6) = N^{[S^N]}(T6) = \{A17\}$, εξάγονται οι ακόλουθοι κανόνες δυαδικής λογικής σύμφωνα με την τελευταία περίπτωση του Πίνακα 6.3:

$$T6 \leftarrow \text{OR}(\text{T6\_}\ell, \text{T6\_c}) \quad (6.15) \qquad \text{T6\_c} \leftarrow \text{AND}(T8, \neg A17) \quad (6.17)$$

$$\text{T6\_}\ell \leftarrow \text{AND}(T9, \text{T6\_d}) \quad (6.16) \qquad \text{T6\_d} \leftarrow \text{AND}(A10, A11) \quad (6.18)$$

οι οποίοι μπορούν απευθείας να αντιστοιχηθούν σε CNF εκφράσεις όπως αυτό φαίνεται στον Πίνακα 6.1. Για παράδειγμα η επόμενη CNF έκφραση αντιστοιχεί στον κανόνα της Εξίσωσης (6.17):

$$(T8 \vee \neg\text{T6\_c}) \wedge (\neg A17 \vee \neg\text{T6\_c}) \wedge (\neg T8 \vee A17 \vee \text{T6\_c})$$

Ανακεφαλαιώνοντας, με την διαδικασία που περιγράφεται σε αυτήν την ενότητα μπορούμε να εξάγουμε μία ενιαία CNF έκφραση για όλο το μοντέλο στην περίπτωση που αυτό περιέχει μόνο crisp κόμβους. Επιπλέον, εξάγουμε ένα σύνολο από δυαδικούς κανόνες οι οποίοι μπορούν να χρησιμοποιηθούν προκειμένου να υπολογιστεί η τιμή αληθείας ενός

κόμβου του μοντέλου όπως στην περίπτωση των κανόνων (6.15) - (6.18), γνωρίζοντας τις τιμές αληθείας όλων των μεταβλητών που εμφανίζονται στους κανόνες. Περισσότερες λεπτομέρειες για το πώς οι δυαδικοί κανόνες που εξάγονται για όλους τους κόμβους του μοντέλου μπορούν να χρησιμοποιηθούν προκειμένου να υπολογιστούν οι αληθοτιμές των ριζών όταν γνωρίζουμε τις τιμές των φύλλων θα δοθούν στο επόμενο κεφάλαιο.

6.3 Κανόνες Ασαφούς Λογικής

Σε αυτήν την ενότητα θα περιγράψουμε πώς μπορεί να εξαχθεί ένα σύνολο κανόνων ασαφούς λογικής από ένα μοντέλο που περιέχει όχι μόνο crisp αλλά και fuzzy κόμβους. Στο πλαίσιο της παρούσας διατριβής χρησιμοποιούμε δύο τύπους κανόνων ασαφούς λογικής, τους σταθμισμένους ασαφείς κανόνες (weighted fuzzy rules) 2.2.3 και τους κανόνες ασαφών ελεγκτών (Fuzzy Control Language rules) 2.2.4.

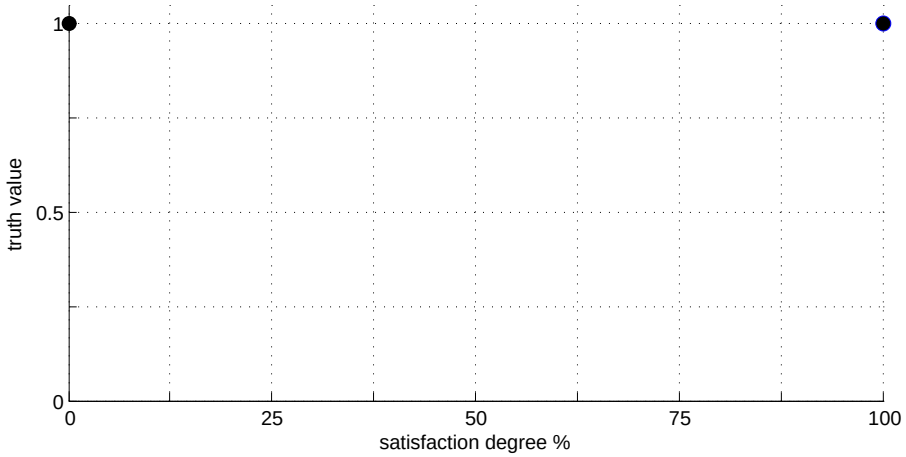
6.3.1 Ορισμός Γλωσσικών Μεταβλητών

Στο πλαίσιο της παρούσας διατριβής θα χρησιμοποιήσουμε δύο γλωσσικές μεταβλητές (linguistic variables), τις LowSat και HighSat. Η τιμή συμμετοχής της LowSat αντιστοιχεί στην αληθοτιμή της έκφρασης 'Ο στόχος g ικανοποιείται λίγο', ενώ η τιμή συμμετοχής της HighSat αντιστοιχεί στην αληθοτιμή της έκφρασης 'Ο στόχος g ικανοποιείται πολύ'. Ο μετασχηματισμός του βαθμού ικανοποίησης ενός κόμβου-στόχου g στις δύο τιμές συμμετοχής γίνεται με την χρήση δύο κατάλληλων συναρτήσεων συμμετοχής. Ο τύπος των συναρτήσεων συμμετοχής για τις δύο γλωσσικές μεταβλητές εξαρτάται από το εάν ο στόχος g είναι crisp ή fuzzy.

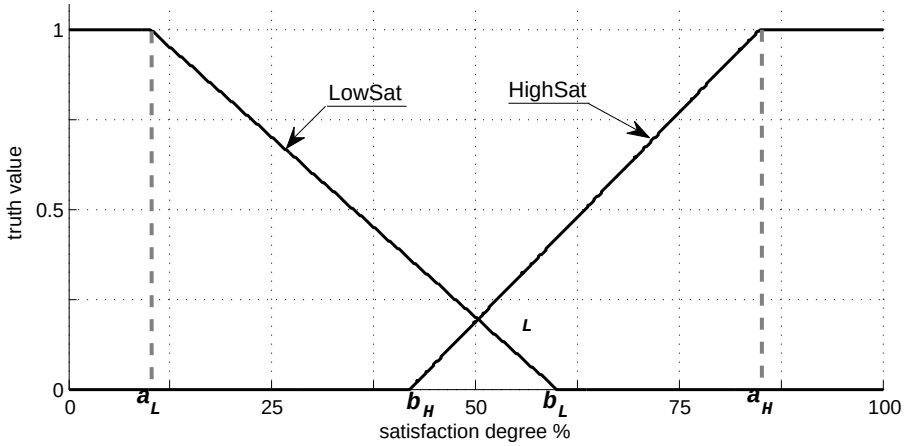
Πιο συγκεκριμένα, για έναν crisp κόμβο g_c με βαθμό ικανοποίησης s ίσο με 100% (εάν ο g_c ικανοποιείται) ή 0% (εάν ο g_c δεν ικανοποιείται), ορίζουμε τις τιμές συμμετοχής w_L και w_H για τα LowSat(g_c) και HighSat(g_c) αντίστοιχα χρησιμοποιώντας τις ακόλουθες συναρτήσεις συμμετοχής:

$$w_L(s) = \begin{cases} 1 & , s = 0 \\ 0 & , s \neq 0 \end{cases} \qquad w_H(s) = \begin{cases} 0 & , s \neq 100 \\ 1 & , s = 100 \end{cases} \tag{6.19}$$

Αντίθετα, για έναν fuzzy στόχο g_f με βαθμό ικανοποίησης ίσο με s, υπολογίζουμε τις τιμές συμμετοχής w_L και w_H για τις LowSat(g_f) και HighSat(g_f) αντίστοιχα



Σχήμα 6-2: Συναρτήσεις συμμετοχής των μεταβλητών LowSat/HighSat για τους *crisp* στόχους



Σχήμα 6-3: Συναρτήσεις συμμετοχής των μεταβλητών LowSat/HighSat για τους *fuzzy* στόχους

χρησιμοποιώντας τις ακόλουθες συναρτήσεις συμμετοχής που φαίνονται στο Σχήμα 6-3:

$$w_L(s; a_L, b_L) = \begin{cases} 1 & , s \leq a_L \\ \frac{b_L - s}{b_L - a_L} & , a_L \leq s \leq b_L \\ 0 & , b_L \leq s \end{cases} \tag{6.20}$$

$$w_H(s; a_H, b_H) = \begin{cases} 0 & , s \leq b_H \\ \frac{s - b_H}{a_H - b_H} & , b_H \leq s \leq a_H \\ 1 & , a_H \leq s \end{cases} \tag{6.21}$$

όπου a_H, a_L, b_H και b_L είναι παράμετροι του περιβάλλοντος πλαισίου οι οποίες πρέπει να οριστούν από τον χρήστη και επιτρέπουν τον ορισμό των συναρτήσεων συμμετοχής ανάλογα με τις ανάγκες της ανάλυσης.

Ανακεφαλαιώνοντας, για κάθε χαρακτηριστικό λειτουργίας με βαθμό ικανοποίησης s εξάγουμε δύο ασαφή γεγονότα (fuzzy facts):

$$\mathbf{w_H} : \text{HighSat}(p) \leftarrow (1.0; t) \text{ and } \mathbf{w_L} : \text{LowSat}(p) \leftarrow (1.0; t)$$

όπου τα w_H και w_L δίνονται από την Εξίσωση 6.19 για τους crisp κόμβους, και από τις Εξισώσεις 6.20 - 6.21 για τους fuzzy κόμβους. Με την χρήση τον δύο ασαφών κατηγορημάτων LowSat και HighSat, επιτρέπουμε την χρήση ενός μηχανισμού ασαφούς συμπερασμού χρησιμοποιώντας την ίδια στιγμή τη σημασιολογία που συνήθως ισχύει για τα μοντέλα στόχων εκφρασμένη σε μορφή ασαφών κανόνων. Παρ΄ όλ΄ αυτά, αυτό δεν σημαίνει ότι αυτά είναι τα μόνα ασαφή κατηγορήματα που μπορούν να χρησιμοποιηθούν. Αντίθετα μπορούμε να ορίσουμε συναρτήσεις συμμετοχής για επιπλέον κατηγορήματα (π.χ. MediumSat) , κάτι που φυσικά απαιτεί την εξαγωγή πιο σύνθετων ασαφών κανόνων, και κατά συνέπεια συνεπάγεται μία πιο πολύπλοκη ανάλυση.

6.3.2 Εξαγωγή Σταθμισμένων Ασαφών Κανόνων

Δοθέντος ενός μοντέλου m και του συνόλου των κανόνων στο μοντέλο, δηλαδή το σύνολο $\mathcal{R}(m)$, εξάγουμε ένα σύνολο σταθμισμένων ασαφών κανόνων για κάθε κόμβο χρησιμοποιώντας τις αντιστοιχίσεις που φαίνονται στον Πίνακα 6.4. Έτσι, μπορούμε να δομήσουμε ένα ασαφές λογικό πρόγραμμα για το μοντέλο m, και εν συνεχεία να χρησιμοποιήσουμε τη μηχανή συμπερασμού που εισάγεται στο [33].

Για παράδειγμα, για τον κόμβο $A4$ στο Σχήμα 4-1 θα εξαχθούν οι ακόλουθοι σταθμισμένοι ασαφείς λογικοί κανόνες για τον υπολογισμό των τιμών συμμετοχής του κόμβου:

$$\begin{aligned}
&1.0 : \text{LowSat}(A4) \leftarrow (1.0; \text{LowSat}(A5)) \,\tilde{\wedge}\, (1.0; \text{LowSat}(A6)) \\
&1.0 : \text{HighSat}(A4) \leftarrow (1.0; \text{HighSat}(A5)) \\
&1.0 : \text{HighSat}(A4) \leftarrow (1.0; \text{HighSat}(A6)) \\
&1.0 : \text{LowSat}(A4) \leftarrow (0.7; \text{HighSat}(C5)) \\
&1.0 : \text{HighSat}(A4) \leftarrow (0.7; \text{LowSat}(C5))
\end{aligned}$$

Είναι σημαντικό να σημειωθεί ότι στα πλαίσια της παρούσας διατριβής τα ασαφή λογικά προγράμματα θα χρησιμοποιηθούν μόνο για την διαδικασία της επαλήθευσης η οποία εφαρμόζεται μόνο στο υπο-μοντέλο SEB ενός μοντέλου VR. Αυτός είναι και ο λόγος για τον οποίο δεν δίνεται κανένας κανόνας αντιστοίχισης για τις ακμές *lp*, καθώς

Πίνακας 6.4: Μετασχηματισμός $gm-$κανόνων σε $wf-$κανόνες

Goal Model Rule	Weighted Fuzzy Rules
$\langle \text{AND}, g, \{g_1, \cdots g_n\}, 1\rangle$	$1.0 : \text{HighSat}(g) \leftarrow (1.0; \text{HighSat}(g_1)) \,\tilde{\wedge} \cdots \tilde{\wedge}\, (1.0; \text{HighSat}(g_n))$ $1.0 : \text{LowSat}(g) \leftarrow (1.0; \text{LowSat}(g_1))$ $\vdots$ $1.0 : \text{LowSat}(g) \leftarrow (1.0; \text{LowSat}(g_n))$
$\langle \text{OR}, g, \{g_1, \cdots g_n\}, 1\rangle$	$1.0 : \text{LowSat}(g) \leftarrow (1.0; \text{LowSat}(g_1)) \,\tilde{\wedge} \cdots \tilde{\wedge}\, (1.0; \text{LowSat}(g_n))$ $1.0 : \text{HighSat}(g) \leftarrow (1.0; \text{HighSat}(g_1))$ $\vdots$ $1.0 : \text{HighSat}(g) \leftarrow (1.0; \text{HighSat}(g_n))$
$\langle S^P, g_t, \{g_s\}, w\rangle$	$1.0 : \text{HighSat}(g_t) \leftarrow (w; \text{HighSat}(g_s))$
$\langle S^N, g_t, \{g_s\}, w\rangle$	$1.0 : \text{LowSat}(g_t) \leftarrow (w; \text{HighSat}(g_s))$
$\langle D^N, g_t, \{g_s\}, w\rangle$	$1.0 : \text{HighSat}(g_t) \leftarrow (w; \text{LowSat}(g_s))$
$\langle D^P, g_t, \{g_s\}, w\rangle$	$1.0 : \text{LowSat}(g_t) \leftarrow (w; \text{LowSat}(g_s))$

οι ακμές αυτές δεν εμφανίζονται στα μοντέλα SEB.

6.3.3 Εξαγωγή Κανόνων Ασαφών Ελεγκτών

Η Γλώσσα Ασαφών Ελεγκτών (Fuzzy Control Language / FCL), η οποία χρησιμοποιείται για τον ορισμό και την υλοποίηση ασαφών ελεγκτών, μπορεί να χρησιμοποιηθεί για την καταγραφή των εξαρτήσεων που υπάρχουν σε ένα μοντέλο με fuzzy κόμβους. Στα πλαίσια της παρούσας διατριβής εξάγουμε ένα σύνολο FCL κανόνων, το οποίο αναφέρουμε ως *ασαφή μονάδα* (fuzzy module), μόνο για τους fuzzy κόμβους του μοντέλου, επομένως, όπως και στην περίπτωση των σταθμισμένων ασαφών ελεγκτών, δεν θα υπάρχει αντιστοίχιση για τις ακμές *lp* .

Πιο συγκεκριμένα, μία FCL μονάδα είναι πρακτικά ένα μπλοκ υπολογισμού το οποίο δημιουργείται χρησιμοποιώντας το πρότυπο του Σχήματος 6-4. Όλοι οι όροι που εμφανίζονται με πλάγια γραφή και ξεκινούν με τον ειδικό χαρακτήρα $ στο πρότυπο του Σχήματος 6-4, είναι όροι που πρέπει να αντικατασταθούν με τιμές που αντιστοιχούν στον δοθέντα κόμβο. Για κάθε κόμβο p του δοθέντος μοντέλου m:

- *goal_ name* είναι το όνομα του στόχου εξόδου που είναι το όνομα του κόμβου p και πρέπει να είναι μοναδικό,

- *Vin_ 1* $\cdots$ *Vin_ n* είναι τα ονόματα των κόμβων εισόδου, δηλαδή των κόμβων του συνόλου $\mathcal{N}(m, p)$, και
- *rule_ 1* $\cdots$ *rule_ k* είναι FCL κανόνες που εξάγονται για τον κόμβο p όπως αυτό περιγράφεται στον Πίνακα 6.5.

Για παράδειγμα, για τον fuzzy κόμβο $A4$ του Σχήματος 4-1, η μεταβλητή *$goal_ name* θα αντικατασταθεί από την τιμή $A4$, και θα υπάρχουν τρεις μεταβλητές εισόδου, οι $A5$, $A6$ και $C5$. Επομένως, θα υπάρχουν τρία FUZZIFY μπλοκ, ένα για κάθε μεταβλητή εισόδου.

Τέλος, εξάγονται οι ακόλουθοι κανόνες, τρεις από τον κανόνα διάσπασης και ένας για την ακμή συνεισφοράς που καταλήγει στον κόμβο $A4$ του μοντέλου του Σχήματος 4-1:

RULE 1: **IF** $A5$ **IS** *low* **AND** $A6$ **IS** *low* **THEN** $A4$ **IS** *low* **WITH** 1.0
RULE 2: **IF** $A5$ **IS** *high* **THEN** $A4$ **IS** *high* **WITH** 1.0
RULE 3: **IF** $A6$ **IS** *high* **THEN** $A4$ **IS** *high* **WITH** 1.0
RULE 4: **IF** $C5$ **IS** *high* **THEN** $A4$ **IS** *low* **WITH** 0.7
RULE 5: **IF** $C5$ **IS** *low* **THEN** $A4$ **IS** *high* **WITH** 0.7

6.4 Ανάθεση Συνθηκών στους Κανόνες

Στην προηγούμενη ενότητα περιγράψαμε τον μετασχηματισμό των κανόνων του μοντέλου στόχων σε δυαδικούς και ασαφείς κανόνες. Παρ΄ όλ΄ αυτά, καθώς στα SEB μοντέλα, και κατά συνέπεια στα VR μοντέλα, επιτρέπουμε τον ορισμό στοιχείων τα οποία ισχύουν υπό συνθήκη, θα πρέπει επιπλέον να παρέχουμε έναν τρόπο ώστε οι συνθήκες που υπάρχουν στους κανόνες να μεταφέρονται και στους κανόνες που εξάγονται για τα μοντέλα. Αυτό επιτυγχάνεται μέσω μίας διαδικασίας δύο βημάτων η οποία περιγράφεται σε αυτήν την ενότητα, και η οποία στοχεύει στο να εξάγει μία CNF έκφραση για κάθε δυαδικό ή ασαφή κανόνα. Δοθέντος του ενεργού πλαισίου μέσα στο οποίο λειτουργεί το σύστημα, δηλαδή των αληθοτιμών για τις μεταβλητές πλαισίου (context variables), μπορούν να αποτιμηθούν οι αληθοτιμές όλων των CNF εκφράσεων, επιτρέποντας μας έτσι να ελέγξουμε εάν ο αντίστοιχος κανόνας θα πρέπει να χρησιμοποιηθεί κατά τη διαδικασία συμπερασμού ή όχι.

Η εξαγωγή των CNF εκφράσεων ολοκληρώνεται σε δύο φάσεις. Κατά την πρώτη φάση ("goal model rules preprocessing") ένας αριθμός ψευδο-κόμβων προστίθεται στο μοντέλο όπου είναι απαραίτητο. Στη συνέχεια, κατά την δεύτερη φάση ("conditional

Πίνακας 6.5: Μετασχηματισμός $gm-$κανόνων σε κανόνες FCL

gm-rule	FCL rules
$\langle \text{AND}, p, \{c_1, c_2, \cdots c_n\}\rangle$	**IF** c_1 **IS** *high* **AND** $\cdots$ **AND** c_n **IS** *high* **THEN** p **IS** *high* **IF** c_1 **IS** *low* **THEN** p **IS** *low* $\vdots$ **IF** c_n **IS** *low* **THEN** p **IS** *low*
$\langle \text{OR}, p, \{c_1, c_2, \cdots c_n\}\rangle$	**IF** c_1 **IS** *low* **AND** $\cdots$ **AND** c_n **IS** *low* **THEN** p **IS** *low* **IF** c_1 **IS** *high* **THEN** p **IS** *high* $\vdots$ **IF** c_n **IS** *high* **THEN** p **IS** *high*
$\langle S^P, g_s, g_t, w\rangle$	**IF** g_s **IS** *high* **THEN** g_t **IS** *high* **WITH** w
$\langle S^N, g_s, g_t, w\rangle$	**IF** g_s **IS** *high* **THEN** g_t **IS** *low* **WITH** w
$\langle D^N, g_s, g_t, w\rangle$	**IF** g_s **IS** *low* **THEN** g_t **IS** *high* **WITH** w
$\langle D^P, g_s, g_t, w\rangle$	**IF** g_s **IS** *low* **THEN** g_t **IS** *low* **WITH** w

rules generation"), το μοντέλο της προηγούμενης φάσης χρησιμοποιείται για να εξαχθεί το τελικό σύνολο δυαδικών και ασαφών κανόνων υπό συνθήκη.

Φάση 1 [goal model rules preprocessing] Η πρώτη φάση στοχεύει στην αντικατάσταση κόμβων που συμμετέχουν ταυτόχρονα ως AND και OR παιδιά σε κανόνες διάσπασης από δύο ψευδο-κόμβους, έναν OR-ψευδοκόμβο και έναν AND-ψευδοκόμβο. Η αναγκαιότητα αυτής της αντικατάστασης θα δικαιολογηθεί στην δεύτερη φάση του αλγορίθμου.

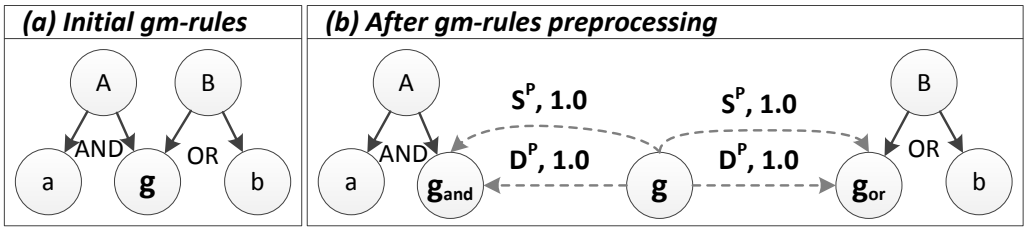
Κατά τη διάρκεια αυτής της φάσης, διατρέχουμε κάθε υπό συνθήκη κόμβο $g \in \mathcal{G}(m)$ του μοντέλου (γραμμές 2-3 του Αλγορίθμου 2), και παίρνουμε όλους τους κανόνες $AND-$διάσπασης (γραμμή 4), και όλους τους κανόνες $OR-$διάσπασης (γραμμή 5) στους οποίους ο κόμβος g συμμετέχει σαν κόμβος παιδί. Είναι σημαντικό να σημειώσουμε ότι προκειμένου να ελέγξουμε εάν ένας κόμβος υπάρχει υπό συνθήκη ή όχι, χρησιμοποιούμε την συνάρτηση Cond η οποία ορίζεται για τα μοντέλα SEB στην παρούσα ανάλυση, καθώς μόνο σε αυτά τα μοντέλα επιτρέπουμε την ύπαρξη στοιχείων υπό συνθήκη.

Στη συνέχεια, εάν ο κόμβος g συμμετέχει σαν κόμβος παιδί ταυτόχρονα σε AND και OR κανόνες διάσπασης (γραμμή 6),αντικαθιστούμε τον κόμβο g με έναν AND-ψευδο-

```
FUNCTION_BLOCK ${goal_name}

      VAR_INPUT
          ${Vin_1} : REAL;
          ...
          ${Vin_n} : REAL;
      END_VAR

      VAR_OUTPUT
            ${goal_name} : REAL;
      END_VAR

      FUZZIFY ${Vin_1}
            TERM low := (0,1) (10,1) (60,0);
            TERM high := (40,0) (90,1) (100,1);
      END_FUZZIFY
      ...
      FUZZIFY ${Vin_n}
            TERM low := (0,1) (10,1) (60,0);
            TERM high := (40,0) (90,1) (100,1);
      END_FUZZIFY

      DEFUZZIFY ${goal_name}
            TERM low := (0,1) (10,1) (60,0);
            TERM high := (40,0) (90,1) (100,1);
            METHOD : COG;
            DEFAULT := 50;
      END_DEFUZZIFY

      RULEBLOCK No1
            AND : PROD;
            ACT : PROD;
            ACCU : NSUM;

          RULE 1 : ${rule_1};
          ...
          RULE k : ${rule_k};
      END_RULEBLOCK

END_FUNCTION_BLOCK
```

Σχήμα 6-4: Πρότυπο FCL Μονάδας

κόμβο (g_{and}) στους AND κανόνες διάσπασης (γραμμή 8), και με έναν OR-ψευδοκόμβο (g_{or}) στους OR κανόνες διάσπασης (γραμμή 11). Αυτό φαίνεται και στο Σχήμα 6-5, όπου ο κόμβος G συμμετέχει σε δύο κανόνες διάσπασης διαφορετικού τύπου.



Σχήμα 6-5: Προσθήκη ψευδοκόμβων και ακμών συνεισφοράς

Έχοντας αντικαταστήσει τον g με τους g_{or} και g_{and} θα πρέπει να εξασφαλίσουμε ότι οι ίδιοι κανόνες που πριν χρησιμοποιούνταν για τον υπολογισμό της αληθοτιμής του g, θα χρησιμοποιηθούν τώρα για τον υπολογισμό της αληθοτιμής των νέων ψευδοκόμβων που έχουν προστεθεί στο μοντέλο. Για να το εξασφαλίσουμε αυτό, προσθέτουμε 4 ακμές συνεισφοράς (γραμμές 13-16) οι οποίες πρακτικά αναθέτουν την τιμή του g, όπως αυτή έχει υπολογιστεί από τους υπάρχοντες κανόνες στους ψευδοκόμβους του. Πιο συγκεκριμένα, οι δύο S^P ακμές συνεισφοράς εξασφαλίζουν ότι οι g_{or} και g_{and} ικανοποιούνται όταν ο κόμβος g ικανοποιείται, ενώ οι δύο D^P ακμές συνεισφοράς, εξασφαλίζουν ότι οι g_{or} και g_{and} δεν ικανοποιούνται όταν ο κόμβος g δεν ικανοποιείται (βλέπε Εξίσωση 3.1).

Ανακεφαλαιώνοντας, μέσω αυτής της φάσης ένα αρχικό μοντέλο m μετασχηματίζεται σε ένα μοντέλο m' το οποίο έχει όλους τους κανόνες του συνόλου $\mathcal{R}(m)$ και επιπλέον τις πρόσθετες ακμές συνεισφοράς, και ένα σύνολο κόμβων που περιέχει όλους τους κόμβους του συνόλου $\mathcal{G}(m)$ και επιπλέον τους ψευδοκόμβους. Σκοπός του παραγόμενου μοντέλου m' είναι να χρησιμοποιηθεί για την εξαγωγή των υπό συνθήκη κανόνων και όχι να αντικαταστήσει το αρχικό μοντέλο.

Φάση 2 [conditional rules generation] Το μοντέλο m' που παράγεται από την προηγούμενη φάση μπορεί τώρα να χρησιμοποιηθεί προκειμένου να εξαχθούν οι απαιτούμενοι δυαδικοί ή ασαφείς κανόνες όπως αυτό περιγράφεται στον Αλγόριθμο 3. Καθώς κάποιοι από τους κόμβους και τις ακμές υπάρχουν στο μοντέλο υπό συνθήκη, θα πρέπει να εξάγουμε μία έκφραση η οποία καθορίζει το αν ο κάθε κανόνας πρέπει να υπάρχει και κατά συνέπεια να χρησιμοποιηθεί στην ανάλυση δοθέντος του ενεργού πλαισίου C^A μέσα στο οποίο λειτουργεί το σύστημα.

Διατρέχουμε κάθε κανόνα του μοντέλου $gm\_\ r = \langle R_t, g_t, G_s, w\rangle$ στο σύνολο των κανόνων που επιστρέφει η πρώτη φάση (γραμμή 1), και δημιουργούμε μία CNF έκφραση (γραμμή 2) την οποία ορίζουμε ως:

$$\mathbf{CNF}(\mathrm{Cond}(gm\_\ r),\ \mathrm{Cond}(g_t),\ \mathrm{Cond}(G_s))$$

η οποία αντιστοιχεί στην σύζευξη 3 εκφράσεων, ενός για κάθε όρο,όπου η κάθε έκφραση είναι μία διάζευξη των μεταβλητών που περιέχονται στον αντίστοιχο όρο. Αυτή η

Algorithm 2 Φάση 1 - goal model rules preprocessing

Input : m : initial model
Output : R_{full} : new goal model rules

1: $R_{pre} \leftarrow$ emptySet
2: **for all** $g \in \mathcal{G}(m)$ **do**
3: **if** $Cond(g) - \{\top\} \neq \emptyset$ **then**
4: andRules $\leftarrow$ getDecompRules("AND", R, g)
5: orRules $\leftarrow$ getDecompRules("OR", R, g)
6: **if** andRules $\neq \emptyset$ **and** orRules $\neq \emptyset$ **then**
7: **for all** $r \in$ andRules **do**
8: substituteInRule(r, g, g_{and})
9: **end for**
10: **for all** $r \in$ orRules **do**
11: substituteInRule(r, g, g_{or})
12: **end for**
13: add rule $\langle S^P, g_{\text{or}}, \{g\}, 1.0\rangle$ in R_{pre}
14: add rule $\langle S^P, g_{\text{and}}, \{g\}, 1.0\rangle$ in R_{pre}
15: add rule $\langle D^P, g_{\text{or}}, \{g\}, 1.0\rangle$ in R_{pre}
16: add rule $\langle D^P, g_{\text{and}}, \{g\}, 1.0\rangle$ in R_{pre}
17: **end if**
18: **end if**
19: **end for**
20: $R_{full} \leftarrow \mathcal{R}(m) \cup R_{pre}$
21: **return** R_{full}

έκφραση περιγράφει το γεγονός ότι για να υπάρχει ένας κανόνας μοντέλου στόχων τουλάχιστον ένας κόμβος αρχής και ο κόμβος τέλους θα πρέπει να υπάρχουν στο μοντέλο, και ο ίδιος ο κανόνας θα πρέπει να εξαρτάται από μία συνθήκη που ανήκει στο ενεργό πλαίσιο. Για παράδειγμα για τον κόμβο $C1$ στο Σχήμα 4-1:

$$\mathbf{CNF}(\{\top\}, \{u_3\}, \{u_1, u_2, \top\}) = \top \wedge u_3 \wedge (u_1 \vee u_2 \vee \top) = \top$$

Παρ΄ όλ΄ αυτά, εάν ο $C1$ διαιρούνταν μόνο στους $C5$ και $C4$, η αντίστοιχη CNF έκφραση θα ήταν:

$$\mathbf{CNF}(\{\top\}, \{u_3\}, \{\mathbf{u_1}, \mathbf{u_2}\}) = \top \wedge u_3 \wedge (u_1 \vee u_2) = u_3 \wedge (u_1 \vee u_2)$$

που σημαίνει ότι στην περίπτωση αυτή, εάν και οι δύο συνθήκες u_1 και u_2 ήταν false, ο κανόνας της $AND-$διάσπασης δεν θα υπήρχε στο μοντέλο.

Στη συνέχεια, ένα σύνολο δυαδικών ή ασαφών κανόνων εξάγεται για κάθε κανόνα στο μοντέλο (γραμμή 3), χρησιμοποιώντας τους κανόνες μετασχηματισμού που δίνονται

Algorithm 3 Δημιουργία δυαδικών/ασαφών κανόνων υπό συνθήκη

Input : G': $\mathcal{G}(m')$, D': $\mathcal{R}(m')$
Output : CondRB : conditional Boolean/Fuzzy rules

```
for all gm_ r= ⟨R_t, g_t, G_s, w⟩ ∈ D' do
    f ← CNF(Cond(gm_r), Cond(g_t), Cond(G_s))
    BF ← produceBFRules(gm_r)
    for all bf_r ∈ BF do
        add ⟨f, bf_r⟩ in CondRB
    end for
end for
for all g ∈ G' do
    if Cond(g) − {⊤} ≠ ∅ then
        if isAndChild(g) then
            f ← ¬CNF(Cond(g))
            addANDFacts(f,g,CondRB)
        end if
        if isOrChild(g) then
            f ← ¬CNF(Cond(g))
            addORFacts(f,g,CondRB)
        end if
    end if
end for
return CondRB
```

στους Πίνακες 6.2, 6.3, 6.4, και 6.5, ανάλογα με το εάν το μοντέλο περιέχει μόνο crisp, μόνο fuzzy ή ταυτόχρονα crisp και fuzzy κόμβους, και με το είδος της ανάλυσης που θέλουμε να εφαρμόσουμε.

Για παράδειγμα, για τον κανόνα $AND-$διάσπασης $r_{C1} = \langle \text{AND}, C1, \{C2, C3, C4, C5\}, 1\rangle$, του παραδείγματος του Σχήματος 4-1, οι ακόλουθοι κανόνες να εξαχθούν (ανάλογα με το είδος της διαδικασίας συμπερασμού που θέλουμε να χρησιμοποιήσουμε):

- **Boolean Rules**

$$C1 \leftarrow \text{AND}(C2, C3, C4, C5)$$

- **Weighted Fuzzy Rules**

$$
\begin{aligned}
1.0 : \mathrm{HighSat}(C1) \leftarrow\ & (1.0; \mathrm{HighSat}(C2)) \,\tilde{\wedge}\, (1.0; \mathrm{HighSat}(C3)) \\
& \tilde{\wedge}\, (1.0; \mathrm{HighSat}(C4)) \,\tilde{\wedge}\, (1.0; \mathrm{HighSat}(C5)) \\
1.0 : \mathrm{LowSat}(C1) \leftarrow\ & (1.0; \mathrm{LowSat}(C2)) \\
1.0 : \mathrm{LowSat}(C1) \leftarrow\ & (1.0; \mathrm{LowSat}(C3)) \\
1.0 : \mathrm{LowSat}(C1) \leftarrow\ & (1.0; \mathrm{LowSat}(C4)) \\
1.0 : \mathrm{LowSat}(C1) \leftarrow\ & (1.0; \mathrm{LowSat}(C5))
\end{aligned}
$$

- **Fuzzy Control Language Rules**

RULE 1: **IF** $C2$ **IS** *high* **AND** $C3$ **IS** *high*
AND $C4$ **IS** *high* **AND** $C5$ **IS** *high* **THEN** $C1$ **IS** *high* **WITH** 1.0
RULE 2: **IF** $C2$ **IS** *low* **THEN** $C1$ **IS** *low* **WITH** 1.0
RULE 3: **IF** $C3$ **IS** *high* **THEN** $C1$ **IS** *high* **WITH** 1.0
RULE 4: **IF** $C4$ **IS** *high* **THEN** $C1$ **IS** *low* **WITH** 1.0
RULE 5: **IF** $C5$ **IS** *low* **THEN** $C1$ **IS** *high* **WITH** 1.0

Οι κανόνες που εξάγονται συνδυάζονται με τις CNF εκφράσεις που παράχθηκαν προηγουμένως προκειμένου να προκύψουν οι απαιτούμενοι υπό συνθήκη κανόνες (γραμμές 4 - 6).

Κάποιοι κόμβοι παιδιά όμως μπορεί να εμφανίζονται υπό συνθήκη και κατά συνέπεια να μην υπάρχουν στο μοντέλο κατά τον χρόνο εκτέλεσης, παρ΄ όλο που εμφανίζονται στους κανόνες, π.χ. οι κόμβοι $C4$ και $C5$ του προηγούμενου παραδείγματος.

Έστω ότι το ενεργό πλαίσιο δεν συμπεριλαμβάνει την συνθήκη u_1 γεγονός που συνεπάγεται ότι ο κόμβος $C5$ δεν υπάρχει στο μοντέλο κατά τον χρόνο εκτέλεσης. Επίσης κάθε κανόνας που μπορεί να χρησιμοποιηθεί για τον υπολογισμό της τιμής του κόμβου $C5$ δε θα είναι ενεργός καθώς η αντίστοιχη CNF έκφραση περιέχει την μεταβλητή u_1 η οποία είναι false. Έτσι, καθώς η τιμή αληθείας του κόμβου $C5$ θα χρησιμοποιηθεί για τον υπολογισμό τις τιμής του κόμβου $C1$, θα πρέπει να εξασφαλίσουμε ότι η τιμή του $C5$, και εν συνεχεία οι τιμές των $\mathrm{HighSat}(C5)$ και $\mathrm{LowSat}(C5)$ στην περίπτωση των σταθμισμένων ασαφών κανόνων, τίθενται σε μία τιμή η οποία δεν θα επηρεάσει τον υπολογισμό της τιμής του $C1$ όταν εκτελείται η διαδικασία του συμπερασμού.

Προκειμένου να ξεπεράσουμε αυτό το πρόβλημα, διατρέχουμε όλους τους κόμβους g του μοντέλου, και αν ο κόμβος είναι ένας κόμβος υπό συνθήκη (γραμμή 9) προσθέτουμε ένα γεγονός υπό συνθήκη στο σύνολο των κανόνων (σύνολο CondRB), μόνο όταν ο κόμβος είναι κόμβος παιδί κάποιου σύνθετου κόμβου (γραμμές 10 και 14). Τα γεγο-

νότα που πρέπει να προστεθούν εξαρτώνται από τον τύπο των κανόνων που παράγονται (δηλαδή από αν χρησιμοποιούνται δυαδικοί κανόνες, σταθμισμένοι ασαφείς κανόνες ή κανόνες ασαφών ελεγκτών) και είναι τα ακόλουθα:

- **Boolean Rules** Στην περίπτωση που ο κόμβος g είναι παιδί ενός $AND-$σύνθετου κόμβου, θέτουμε την τιμή του ίση με true, ενώ αντίθετα εάν είναι παιδί ενός $OR-$σύνθετου κόμβου θέτουμε την τιμή του ίση με false.

- **Weighted Fuzzy Rules** Στην περίπτωση που ο κόμβος g είναι παιδί ενός $AND-$σύνθετου κόμβου προσθέτουμε τα ασαφή γεγονότα:

  $$\mathbf{1.0} : \mathrm{HighSat}(g) \leftarrow (1.0; t) \text{ and } \mathbf{0.0} : \mathrm{LowSat}(g) \leftarrow (1.0; t)$$

  ενώ στην περίπτωση που ο κόμβος g είναι παιδί ενός $OR-$σύνθετου κόμβου προσθέτουμε τα ασαφή γεγονότα:

  $$\mathbf{0.0} : \mathrm{HighSat}(g) \leftarrow (1.0; t) \text{ and } \mathbf{1.0} : \mathrm{LowSat}(g) \leftarrow (1.0; t)$$

- **Fuzzy Control Language Rules** Στην περίπτωση που ο κόμβος g είναι παιδί ενός $AND-$σύνθετου κόμβου θέτουμε την τιμή του βαθμού ικανοποίησής του ίση με 100% , ενώ αντίθετα στην περίπτωση που ο κόμβος g είναι παιδί ενός $OR-$σύνθετου κόμβου θέτουμε την τιμή του βαθμού ικανοποίησής του ίση με 0%

Αυτά τα γεγονότα θα αποτελούν μέρος της βάσης γνώσης μόνο στην περίπτωση που η έκφραση $\neg\mathbf{CNF}(Cond(g))$ είναι αληθής (γραμμές 11 και 15), όπου :

$$\mathbf{CNF}(\{a_1, a_2, \cdots a_n\}) = a_1 \vee a_2 \vee \cdots \vee a_n$$

που είναι ίση με u_1 για τον κόμβο $C3$. Αυτό συνεπάγεται ότι τα γεγονότα συμμετέχουν στην βάση γνώσης μόνο όταν ο κόμβος g δεν υπάρχει στο μοντέλο, και έτσι όλοι οι κανόνες που μπορούν να χρησιμοποιηθούν δεν υπάρχουν στην βάση γνώσης. Αυτός είναι και ο λόγος που αναθέτουμε τιμές στα γεγονότα με τον τροπο που περιγράφεται παραπάνω.

Τέλος, καθώς ο τύπος των κανόνων διάσπασης αλλάζει τα γεγονότα που προστίθενται στην βάση γνώσης, έχοντας εισάγει κατά την φάση 1 στο μοντέλο τους ψευδοκόμβους, εξασφαλίζουμε ότι δεν υπάρχει κανένας κόμβος ο οποίος να συμμετέχει ταυτόχρονα σε AND και OR κανόνες διάσπασης.

Κεφάλαιο 7

Μηχανισμοί Συμπερασμού Επαλήθευσης και Αποκατάστασης

Στο προηγούμενο κεφάλαιο περιγράψαμε πώς τα μοντέλα που απεικονίζουν στόχους που πρέπει να ικανοποιούνται σε ένα σύστημα λογισμικού που βρίσκεται σε λειτουργία καθώς και ενέργειες οι οποίες μπορούν να συμβάλουν προς την αποκατάσταση της ορθής λειτουργίας ενός συστήματος μπορούν να μετασχηματιστούν σε ένα σύνολο κανόνων δυαδικής ή ασαφούς λογικής. Σε αυτό το κεφάλαιο θα παρουσιάσουμε τις λεπτομέρειες χρήσης αυτών των κανόνων για την αυτοματοποίηση των μηχανισμών επαλήθευσης και αποκατάστασης. Πιο συγκεκριμένα περιγράφουμε πώς οι κανόνες αυτοί μπορούν να χρησιμοποιηθούν από μία μηχανή συμπερασμού προκειμένου να εκτελεστεί είτε μία διαδικασία υπολογισμού από κάτω προς τα πάνω (bottom-up) στην περίπτωση της επαλήθευσης, είτε μία διαδικασία υπολογισμού από πάνω προς τα κάτω (top-down) στην περίπτωση της επαλήθευσης.

7.1 Επαλήθευση

Προκειμένου να επαληθεύσουμε ότι ένα σύστημα ικανοποιεί ένα σύνολο στόχων οι οποίοι μοντελοποιούνται από ένα μοντέλο SEB θα πρέπει να υλοποιήσουμε μία διαδικασία συμπερασμού από κάτω προς τα πάνω, δηλαδή θα πρέπει να υλοποιήσουμε έναν μηχανισμό για την διάδοση των αληθοτιμών από τα φύλλα που αντιστοιχούν στα χαρακτηριστικά του συστήματος που καταγράφουμε προς τις ρίζες του μοντέλου. Στην παρούσα διατριβή, αυτή η διαδικασία συμπερασμού εφαρμόζεται είτε πάνω σε μία βάση γνώσης που περιέχει σταθμισμένους ασαφείς κανόνες χρησιμοποιώντας την μηχανή συμπερασμού που εισάγεται στο [33], είτε μέσω μίας υβριδικής προσέγγισης στην οποία χρησιμοποιούνται ταυτόχρονα κανόνες δυαδικής λογικής και κανόνες ασαφών ελεγκτών για τους crisp

και fuzzy κόμβους αντίστοιχα. Στο υπόλοιπο αυτής της ενότητας αρχικά περιγράφουμε τη διαδικασία συμπερασμού με την χρήση σταθμισμένων ασαφών κανόνων, και στη συνέχεια την υβριδική προσέγγιση.

7.1.1 Σταθμισμένοι Ασαφείς Κανόνες

Στον χρόνο σχεδίασης παράγουμε τους σταθμισμένους ασαφείς κανόνες όπως αυτό έχει περιγραφεί στο προηγούμενο κεφάλαιο. Αυτοί οι κανόνες είναι δυνατό να χρησιμοποιηθούν σε πραγματικό χρόνο (όσο το σύστημα λειτουργεί) προκειμένου να ελέγξουμε εάν το σύστημα ικανοποιεί συγκεκριμένους στόχους λαμβάνοντας υπόψη χαρακτηριστικά της λειτουργίας του συστήματος. Πιο συγκεκριμένα, για να ολοκληρωθεί η διαδικασία συμπερασμού σε πραγματικό χρόνο οι ακόλουθες ενέργειες πρέπει να ολοκληρωθούν (βλέπε Ενότητα 5.1):

1. *προσδιορισμός της βάσης γνώσης* και *ασαφοποίηση* των χαρακτηριστικών λειτουργίας του συστήματος τα οποία εξάγονται με τη χρήση τεχνικών παρακολούθησης,
2. *διαδικασία συμπερασμού*, δηλαδή υπολογισμός των τιμών αληθείας όλων των στόχων χρησιμοποιώντας έναν κατάλληλο μηχανισμό συμπερασμού,
3. η *απο-ασαφοποίηση* των υπολογιζόμενων τιμών προκειμένου να εξαχθούν οι βαθμοί ικανοποίησης όλων των στόχων.

Τα βήματα αυτά εκτελούνται σε τακτά χρονικά διαστήματα καθώς το σύστημα βρίσκεται σε λειτουργία. Αυτό είναι σημαντικό καθώς τόσο οι συνθήκες μέσα στις οποίες λειτουργεί το σύστημα όσο και τα χαρακτηριστικά λειτουργίας του συστήματος μεταβάλλονται με τον χρόνο. Σκοπός του προτεινόμενου μηχανισμού συμπερασμού είναι να παρακολουθεί την επίδραση αυτών των αλλαγών στον βαθμό συμμόρφωσης του συστήματος με τους στόχους που έχουν θέσει οι εμπλεκόμενοι στο σύστημα. Στη συνέχεια αυτής της ενότητας περιγράφουμε λεπτομερώς τις παραπάνω τρεις ενέργειες.

7.1.1.1 Προσδιορισμός Βάσης Γνώσης και Ασαφοποίηση

Το στάδιο του προσδιορισμού της βάσης γνώσης περιλαμβάνει την αποτίμηση των CNF εκφράσεων που συνδέονται με κάθε υπό συνθήκη $wf-$κανόνα, λαμβάνοντας υπόψη το πλαίσιο μέσα στο οποίο λειτουργεί το σύστημα. Από την άλλη, η διαδικασία της ασαφοποίησης, που είναι μία στάνταρ διαδικασία στους ασαφείς ελεγκτές, περιλαμβάνει τον μετασχηματισμό των χαρακτηριστικών λειτουργίας του συστήματος σε ασαφή γεγονότα (fuzzy facts) με την βοήθεια μίας συνάρτησης συμμετοχής για κάθε ασαφές κατηγόρημα, δηλαδή για τα κατηγορήματα HighSat και LowSat. Η μορφή των συναρτήσεων συμμε-

τοχής εξαρτάται από το αν ο αντίστοιχος κόμβος είναι fuzzy ή crisp και περιγράφονται λεπτομερώς στην ενότητα 6.3.1.

Πιο συγκεκριμένα, γνωρίζοντας ο βαθμός ικανοποίησης για ένα χαρακτηριστικό λειτουργίας είναι ίσος με s, θα παραχθούν τα ακόλουθα ασαφή γεγονότα:

$$\mathbf{w_H} : \text{HighSat}(p) \leftarrow (1.0; t) \text{ and } \mathbf{w_L} : \text{LowSat}(p) \leftarrow (1.0; t)$$

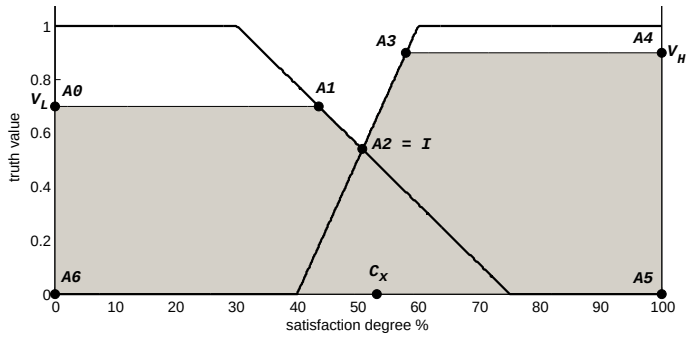
όπου w_H και w_L δίνονται από την Εξίσωση 6.19 για τους crisp κόμβους, και από τις Εξισώσεις 6.20 - 6.21 για τους fuzzy κόμβους.

7.1.1.2 Μηχανισμός Συμπερασμού

Τέλος, η μηχανή συμπερασμού που εισάγεται στο [33] χρησιμοποιείται ως το βασικό δομικό στοιχείο του μηχανισμού συμπερασμού για την διαδικασία της επαλήθευσης. Η βάση γνώσης και τα ασαφή γεγονότα που έχουν προκύψει από την διαδικασία της ασαφοποίησης (δηλαδή η ανάθεση των αληθοτιμών w_H και w_L σε κάθε κόμβο φύλλο) χρησιμοποιούνται από την μηχανή συμπερασμού για να υπολογιστούν οι αληθοτιμές για τα ασαφή κατηγορήματα HighSat και LowSat για όλους τους στόχους. Είναι σημαντικό να αναφερθεί ότι οι τιμές των HighSat και LowSat που υπολογίζονται για τους crisp κόμβους μπορεί να είναι αποκλειστικά ίσες με μηδέν ή ένα εξαιτίας των περιορισμών που αναφέρονται στον Πίνακα 4.1, από τους οποίους συνεπάγεται ότι όλοι οι κανόνες που αναφέρονται σε ένα crisp κόμβο έχουν βάρος ίσο με 1. Οι τιμές των HighSat και LowSat που υπολογίζονται τόσο για τους crisp όσο και για τους fuzzy κόμβους μπορούν στην συνέχεια να χρησιμοποιηθούν για την εξαγωγή συμπερασμάτων, δηλαδή για τον υπολογισμό των βαθμών ικανοποίησης όλων των στόχων του συστήματος, εφαρμόζοντας μία κατάλληλη διαδικασία απο-ασαφοποίησης.

7.1.1.3 Αποασαφοποίηση

Η διαδικασία απο-ασαφοποίησης που χρησιμοποιείται για τον κάθε κόμβο εξαρτάται από τον τύπο του κόμβου. Πιο συγκεκριμένα, για τους crisp κόμβους χρησιμοποιείται μία σχετικά απλή διαδικασία κατηγοριοποιώντας τους κόμβους με LowSat = 1.0 και HighSat = 0.0 ως false (βαθμός ικανοποίησης 0%), και τους κόμβους με LowSat = 0.0 και HighSat = 1.0 ως true (βαθμός ικανοποίησης 100%). Στην περίπτωση που οι τιμές των LowSat και HighSat είναι ταυτόχρονα 1.0 ή 0.0 για έναν στόχο, τότε δεν μπορούμε να καταλήξουμε σε κάποιο συμπέρασμα σχετικά με την τιμή του στόχου (δηλαδή η αληθοτιμή του αντίστοιχου crisp κόμβου είναι άγνωστη). Όσον αφορά στις τιμές στο διάστημα (0,1), δεν είναι δυνατόν να υπάρξουν τέτοιες τιμές για crisp κόμβους εξαιτίας των περιορισμών του Πίνακα 4.1, σύμφωνα με τους οποίους όλοι οι κανόνες που αφορούν



Σχήμα 7-1: Παράδειγμα αποασαφοποίησης ($a_L = 30$, $b_L = 75$, $b_H = 40$, $a_H = 60$, $V_L = 0.7$, $V_H = 0.9$)

σε crisp κόμβους έχουν βάρος 1.

Αντίθετα, για τους fuzzy στόχους ακολουθείται μία πιο πολύπλοκη διαδικασία αποασαφοποίησης, η αποασαφοποίηση κέντρου βάρους, που είναι μία από τις πιο συχνά χρησιμοποιούμενες τεχνικές αποασαφοποίησης [84]. Η μέθοδος αυτή υπολογίζει το κέντρο βάρους της περιοχής που βρίσκεται κάτω από τις συναρτήσεις συμμετοχής, και η $x-$συντεταγμένη του σημείου αυτού είναι η τιμή αποασαφοποίησης.

Σαν παράδειγμα, ας θεωρήσουμε τις συναρτήσεις συμμετοχής που απεικονίζονται στο Σχήμα 7-1 για τις οποίες $a_L = 30$, $b_L = 75$, $b_H = 40$, $a_H = 60$. Αν οι τιμές που υπολογίζονται για τα LowSat και HighSat ενός fuzzy στόχου (δηλαδή ενός κόμβου ρίζα του μοντέλου) είναι $V_L = 0.7$ και $V_H = 0.9$ αντίστοιχα, η τιμή αποασαφοποίησης είναι η $x-$συντεταγμένη του κέντρου βάρους (C_x) για την σκιαγραφημένη περιοχή του σχήματος. Η περιοχή αυτή είναι ένα μη-κυρτό κλειστό πολύγωνο που ορίζεται από την ακολουθία ακμών A0, ..., A6, και έτσι χρησιμοποιούμε την Εξίσωση 2.2 για να υπολογίσουμε την τιμή αποασαφοποίησης του fuzzy κόμβου, που σε αυτήν την περίπτωση είναι $C_x = 53.17$ %.

Ανάλογα με τις παραμέτρους των συναρτήσεων συμμετοχής και τις τιμές LowSat και HighSat για έναν fuzzy στόχο, η περιοχή κάτω από τις συναρτήσεις συμμετοχής μπορεί να πάρει διάφορες μορφές. Σε κάθε περίπτωση πάντως, η περιοχή αυτή θα αποτελείται από μη κυρτά κλειστά πολύγωνα, επιτρέποντας την χρήση της Εξίσωσης 2.2.

7.1.1.4 Παράδειγμα Εξαγωγής Συμπερασμάτων

Σαν παράδειγμα εφαρμογής της μεθόδου, αναπτύξαμε ένα πρωτότυπο της ReqRV μηχανής συμπερασμού, και έναν προσομοιωτή για μία εφαρμογή διαχείρισης δεδομένων που χρησιμοποιεί το μοντέλο SEB του Σχήματος 4-1. Σχεδιάσαμε το μοντέλο SEB χρησιμοποιώντας τα εργαλεία του Eclipse *ecore*, και εφαρμόσαμε μετασχηματισμούς μεταξύ μοντέλων και μετασχηματισμούς μοντέλων σε κείμενο προκειμένου να παραχθούν οι

Πίνακας 7.1: Αποτελέσματα για το SEB μοντέλο του Σχήματος 4-1

		Input Values								Output Values		
Seq.	Active Context	$A5$	$A6$	$B2$	$B3$	$C2$	$C3$	$C4$	$C5$	$A1$	$B1$	$C1$
1	$\{u_3, \top\}$	50%	30%	F	T	T	T	-	-	19.3%	T	T
2	$\{u_1, u_3, \top\}$	50%	30%	F	T	T	T	-	T	18.7%	T	T
3	$\{u_1, u_2, u_3, \top\}$	50%	30%	F	T	T	T	F	T	18.7%	T	F
4	$\{u_1, u_2, u_3, \top\}$	50%	30%	F	T	T	T	F	F	51.3%	T	F
5	$\{u_1, u_2, u_3, \top\}$	50%	30%	T	T	T	T	F	F	73.9%	T	F
6	$\{u_1, u_2, u_3, \top\}$	50%	70%	T	T	T	T	F	F	79.8%	T	F

$wf-$κανόνες υπό συνθήκη. Οι τιμές των παραμέτρων για τις συναρτήσεις συμμετοχής είναι a_L=5, b_L=55, a_H=95 και b_H=45, οι οποίες έχουν ως συνέπεια οι συναρτήσεις συμμετοχής των HighSat και LowSat να είναι συμμετρικές, έτσι ώστε να μην υπερισχύει κάποιο κατηγόρημα έναντι του άλλου. Περισσότερες λεπτομέρειες για την επιλογή των παραμέτρων θα δοθούν στην Ενότητα 8.1.3. Το πρόγραμμα προσομοίωσης παράγει μία ακολουθία αληθοτιμών (βαθμών ικανοποίησης) για τους κόμβους φύλλα $B2$, $B3$, $C2$, $C3$, $C4$, $C5$, και $A6$ του μοντέλου του Σχήματος 4-1. Οι εναπομείναντες κόμβοι φύλλα , δηλαδή ο κόμβος $A5$, θεωρούμε ότι είναι άγνωστοι, και αντιστοιχούν σε κόμβους που είτε είναι μη μετρήσιμοι είτε ο βαθμός ικανοποίησής τους δεν είναι δυνατόν να υπολογιστεί από τα χαρακτηριστικά λειτουργίας του συστήματος. Επιπλέον, για κάθε ακολουθία θεωρούμε ότι είναι ενεργό ένα διαφορετικό πλαίσιο. Για κάθε συνδυασμό τιμών για τα φύλλα και ενεργού πλαισίου υπολογίζουμε τους βαθμούς ικανοποίησης των τριών ριζών $A1$, $B1$ και $C1$. Ένα υποσύνολο της παραγόμενης ακολουθίας που θεωρούμε ότι δείχνει καλύτερα πώς μεταβάλλονται οι βαθμοί ικανοποίησης ως αποτέλεσμα των αλλαγών στις αληθοτιμές των φύλλων και στο ενεργό πλαίσιο δίνονται στον Πίνακα 7.1. Για κάθε ακολουθία ο Πίνακας 7.1 περιέχει τις τιμές που έχουν ανατεθεί στα φύλλα ("Input Values"), το ενεργό πλαίσιο μέσα στο οποίο λειτουργεί το σύστημα, και τους υπολογιζόμενους βαθμούς ικανοποίησης για τις ρίζες ("Output Values").

Αρχικά (Seq. 1),κανένας από τους κόμβους $C5$ και $C4$ δεν υπάρχουν στο μοντέλο καθώς οι συνθήκες u_1 και u_2 δεν ανήκουν στο ενεργό πλαίσιο λειτουργίας. Παρ΄ όλ΄ αυτά, η ρίζα $C1$ ικανοποιείται, καθώς τα δύο εναπομείναντα παιδιά που υπάρχουν στο μοντέλο είναι αληθείς. Επιπλέον, και η ρίζα $B1$ ικανοποιείται καθώς ένα από τα παιδιά της (ο κόμβος $B3$) είναι αληθής. Όσον αφορά στους fuzzy κόμβους του παραδείγματος, ο βαθμός ικανοποίησης του $A5$ είναι ίσος με 50% (LowSat(A5) = 0.1, HighSat(A1) = 0.1) και του $A6$ ίσος με 30% (LowSat(A5) = 0.5, HighSat(A1) = 0.0). Καθώς ο κόμβος $A4$ είναι OR κόμβος με παιδιά τους κόμβους $A5$ και $A6$, οι ακόλουθοι κανόνες

θα χρησιμοποιηθούν για τον υπολογισμό των βαθμών συμμετοχής του:

$$1.0 : \text{LowSat}(A4) \leftarrow (1.0; \text{LowSat}(A5)) \tilde{\wedge} (1.0; \text{LowSat}(A6))$$
$$1.0 : \text{HighSat}(A4) \leftarrow (1.0; \text{HighSat}(A5))$$
$$1.0 : \text{HighSat}(A4) \leftarrow (1.0; \text{HighSat}(A6))$$

από τους οποίους υπολογίζουμε τις τιμές $\text{LowSat(A4)} = \top_{prod}(0.1, 0.5) = 0.05$, και $\text{HighSat(A4)} = \bot_{sum}(0.1, 0.0) = 0$ (Εξισώσεις 2.6 και 2.5). Αντίστοιχα, οι βαθμοί συμμετοχής του AND κόμβου $A1$ υπολογίζονται από τους ακόλουθους κανόνες:

$$1.0 : \text{HighSat}(A1) \leftarrow (1.0; \text{HighSat}(A2)) \tilde{\wedge} (1.0; \text{HighSat}(A3)) \tilde{\wedge} (1.0; \text{HighSat}(A4))$$
$$1.0 : \text{LowSat}(A1) \leftarrow (1.0; \text{LowSat}(A2))$$
$$1.0 : \text{LowSat}(A1) \leftarrow (1.0; \text{LowSat}(A3))$$
$$1.0 : \text{LowSat}(A1) \leftarrow (1.0; \text{LowSat}(A4))$$

από τους οποίους υπολογίζουμε τις τιμές $\text{LowSat}(A1) = \bot_{sum}(0.0, 0.8, 0.05) = 0.81$, καθώς $\text{LowSat}(A3)$=0.8 εξαιτίας της D^P ακμής συνεισφοράς από τον κόμβο $B2$ και $\text{LowSat}(A4)$=0.05, και $\text{HighSat}(A1) = \top_{prod}(1.0, 0.0, 0.0)$=0, καθώς $\text{HighSat}(A2)$=1.0 εξαιτίας της S^P ακμής συνεισφοράς από τον κόμβο $B3$. Συνδυάζοντας τους βαθμούς συμμετοχής του κόμβου $A1$ υπολογίζουμε τον συνολικό βαθμό ικανοποίησης 19.3%.

Εν συνεχεία, (Seq. 2 and 3), οι κόμβοι $C5$ και $C4$ προστίθενται σταδιακά στο μοντέλο καθώς οι συνθήκες u_1 και u_2 γίνονται αληθείς. Και στις δύο περιπτώσεις ο υπολογιζόμενος βαθμός ικανοποίησης για τον κόμβο $A1$ μεταβάλλεται ελάχιστα από την προηγούμενη περίπτωση ως συνέπεια της προσθήκης του κόμβου $C5$ που είναι αληθής στο μοντέλο. Η S^N ακμή συνεισφοράς από τον κόμβο $C5$ στον κόμβο $A4$ έχει ως αποτέλεσμα την αύξηση της τιμής του LowSat(A4) από 0.05 σε 0.715, που με την σειρά του αυξάνει την τιμή του LowSat(A1) από 0.81 σε 0.943, και μειώνει τον συνολικό βαθμό ικανοποίησης του κόμβου $A1$ σε 18.7%. Επιπλέον, στην ακολουθία Seq 3 ο κόμβος $C1$ γίνεται false καθώς ο κόμβος $C4$, που τώρα υπάρχει στο μοντέλο, είναι false.

Αντίθετα, όταν ο κόμβος $C5$ γίνει false στην επόμενη ακολουθία (Seq. 4), ο βαθμός ικανοποίησης του fuzzy κόμβου $A1$ αυξάνεται σε 51.3% σαν συνέπεια της D^N ακμής συνεισφοράς από τον κόμβο $C5$ στον κόμβο $A4$ που έχει ως συνέπεια την αύξηση του $\text{HighSat}(A4)$ στην τιμή 0.73. Έτσι, η τιμή του HighSat(A1) γίνεται ίση με 0.584, η οποία σε συνδυασμό με το γεγονός ότι $\text{LowSat}(A1) = 0.525$ δίνει έναν βαθμό ικανοποίησης ίσο με 51.3%.

Τέλος, όταν ο κόμβος $B2$ αλλάξει από false σε true (Seq. 5), ο βαθμός ικανοποίησης του κόμβου $A1$ αλλάζει σε 73.9% καθώς ο κόμβος $B2$ συνεισφέρει θετικά σε ένα από

τα παιδιά του κόμβου $A1$, δηλαδή στον κόμβο $A3$. Κατά παρόμοιο τρόπο όταν η τιμή του κόμβου $A6$ αυξηθεί σε 70% (LowSat($A6$) = 0.0, HighSat($A6$) = 0.5) ο βαθμός ικανοποίησης του κόμβου $A1$ αυξάνεται σε 79.8%

7.1.2 Υβριδική Διαδικασία Συμπερασμού

Αντί να εξάγουμε ένα σύνολο σταθμισμένων ασαφών κανόνων για ολόκληρο το μοντέλο SEB και να τρέξουμε απευθείας τη διαδικασία συμπερασμού στο σύνολο αυτών των κανόνων, σε αυτήν την ενότητα εισάγουμε έναν εναλλακτική διαδικασία συμπερασμού. Στην περίπτωση αυτή ένα ξεχωριστό σύνολο κανόνων εξάγεται για τον κάθε κόμβο του μοντέλου και στη συνέχεια η αληθοτιμή του κόμβου υπολογίζεται χρησιμοποιώντας αυτούς τους κανόνες εφαρμόζοντας ξεχωριστά μία διαδικασία συμπερασμού για τον κάθε κόμβο. Αυτή η προσέγγιση μας επιτρέπει να εξάγουμε διαφορετικού τύπου κανόνες για κάθε κόμβο (π.χ. Boolean ή Fuzzy) ανάλογα με το είδος του κόμβου. Ονομάζουμε το σύνολο των κανόνων που αντιστοιχούν σε κάθε κόμβο *μονάδα συμπερασμού* (reasoning unit) και η κάθε μονάδα σχετίζεται με μία *λογική συμπερασμού* (reasoning logic) ανάλογα με τον τύπο των κανόνων που χρησιμοποιούνται. Για την υβριδική διαδικασία συμπερασμού απαιτούνται τα ακόλουθα βήματα:

- δημιουργία των μονάδων συμπερασμού και αρχικοποίηση των *In* και *Out* στοιχείων τους,
- καθορισμός της λογικής συμπερασμού για την κάθε μονάδα, και
- καθορισμός του πλάνου εκτέλεσης που περιλαμβάνει τον προσδιορισμό της σειράς με την οποία πρέπει να αποτιμηθούν οι μονάδες συμπερασμού έτσι ώστε να μην παραβιάζονται οι εξαρτήσεις που υπάρχουν μεταξύ των μονάδων

7.1.2.1 Δημιουργία των Μονάδων Συμπερασμού

Προκειμένου να εξάγουμε τις μονάδες συμπερασμού για ένα δοθέν μοντέλο m, διατρέχουμε διαδοχικά κάθε στόχο του μοντέλου $p \in \mathcal{G}(m)$ - $\mathcal{L}(m)$, δηλαδή όλους τους κόμβους που δεν είναι φύλλα, και για κάθε έναν από αυτούς δημιουργούμε μία μονάδα συμπερασμού U_p. Για παράδειγμα για το μοντέλο του Σχήματος 4-1, δημιουργούνται οι έξι μονάδες U_{A1}, U_{A2}, U_{A3}, U_{A4}, U_{B1} και U_{C1}, μία για κάθε έναν από τους εσωτερικούς κόμβους $A1$, $A2$, $A3$, $A4$, $B1$ και $C1$, αντίστοιχα. Για να δείξουμε καλύτερα την εξάρτηση που υπάρχει μεταξύ ενός κόμβου p και της μονάδας συμπερασμού του U_p, ονομάζουμε τον κόμβο p *έξοδο* (output) της μονάδας συμπερασμού U_p. Επιπλέον, το σύνολο των κόμβων από την τιμή των οποίων εξαρτάται η τιμή του κόμβου p, δηλαδή το σύνολο $\mathcal{N}(m, p)$ που περιέχει τους κόμβους αρχής των ακμών συνεισφοράς που

καταλήγουν στον κόμβο p, και τα παιδιά του κόμβου p εάν ο κόμβος p είναι σύνθετος κόμβος, αναφέρεται ως η *είσοδος* (input) της μονάδας συμπερασμού. Στο εξής θα χρησιμοποιούμε τα σύμβολα $In[U_p]$ και $Out[U_p]$ για την είσοδο και την έξοδο μίας μονάδας συμπερασμού αντίστοιχα. Για παράδειγμα, για τις μονάδες συμπερασμού που εξάγονται για το μοντέλο του Σχήματος 4-1 οι είσοδοι και έξοδοι είναι:

$$
\begin{aligned}
Out[U_{A1}] &= A1 \ , \ In[U_{A1}] = \{A2, A3, A4\} \\
Out[U_{A2}] &= A2 \ , \ In[U_{A2}] = \{B3\} \\
Out[U_{A3}] &= A3 \ , \ In[U_{A3}] = \{B2\} \\
Out[U_{A4}] &= A4 \ , \ In[U_{A4}] = \{A5, A6, C5\} \\
Out[U_{B1}] &= B1 \ , \ In[U_{B1}] = \{B2, B3\} \\
Out[U_{C1}] &= C1 \ , \ In[U_{C1}] = \{C2, C3, C4, C5\}
\end{aligned}
$$

7.1.2.2 Προσδιορισμός της Λογικής Συμπερασμού

Μόλις δημιουργηθούν όλες οι μονάδες συμπερασμού, συλλέγουμε για κάθε μονάδα U_p το σύνολο των κανόνων που εξάγονται από τις ακμές συνεισφοράς που καταλήγουν στον κόμβο p, και τον κανόνα διάσπασης του p, εφόσον υπάρχει. Στη συνέχεια, χρησιμοποιούμε τους μετασχηματισμούς που εισάγονται στην Ενότητα 6.2 και στην Ενότητα 6.3.3, ώστε να εξάγουμε είτε ένα σύνολο κανόνων δυαδικής λογικής (Boolean rules), είτε ένα σύνολο κανόνων ασαφών ελεγκτών (Fuzzy Control Language Rules) ανάλογα με τον τύπο του κόμβου. Πιο συγκεκριμένα, για έναν crisp κόμβο μπορούμε να χρησιμοποιήσουμε είτε κανόνων δυαδικής λογικής είτε κανόνες ασαφών ελεγκτών, ενώ για έναν fuzzy κόμβο μπορούμε να χρησιμοποιήσουμε μόνο κανόνες ασαφών ελεγκτών. Αυτό μας δίνει την δυνατότητα να εφαρμόσουμε είτε μία υβριδική διαδικασία συμπερασμού είτε μία διαδικασία που χρησιμοποιεί αποκλειστικά ασαφείς κανόνες.

Για παράδειγμα, για την μονάδα συμπερασμού U_{A4} (βλέπε το μοντέλο του Σχήματος 4-1) η έξοδος θα υπολογιστεί με την χρήση των παρακάτω κανόνων ασαφών ελεγκτών:

RULE 1: **IF** $A5$ **IS** *low* **AND** $A6$ **IS** *low* **THEN** $A4$ **IS** *low* **WITH** 1.0

RULE 2: **IF** $A5$ **IS** *high* **THEN** $A4$ **IS** *high* **WITH** 1.0

RULE 3: **IF** $A6$ **IS** *high* **THEN** $A4$ **IS** *high* **WITH** 1.0

RULE 4: **IF** $C5$ **IS** *high* **THEN** $A4$ **IS** *low* **WITH** 0.7

RULE 5: **IF** $C5$ **IS** *low* **THEN** $A4$ **IS** *high* **WITH** 0.7

ενώ για την μονάδα συμπερασμού U_{C1} μπορούν να χρησιμοποιηθούν είτε οι κανόνες δυαδικής λογικής:

$$C1 \leftarrow \text{AND}(C2, C3, C4, C5)$$

είτε οι ακόλουθοι κανόνες ασαφών ελεγκτών:

RULE 1: **IF** $C2$ **IS** *high* **AND** $C3$ **IS** *high*
AND $C4$ **IS** *high* **AND** $C5$ **IS** *high* **THEN** $C1$ **IS** *high* **WITH** 1.0
RULE 2: **IF** $C2$ **IS** *low* **THEN** $C1$ **IS** *low* **WITH** 1.0
RULE 3: **IF** $C3$ **IS** *high* **THEN** $C1$ **IS** *high* **WITH** 1.0
RULE 4: **IF** $C4$ **IS** *high* **THEN** $C1$ **IS** *low* **WITH** 1.0
RULE 5: **IF** $C5$ **IS** *low* **THEN** $C1$ **IS** *high* **WITH** 1.0

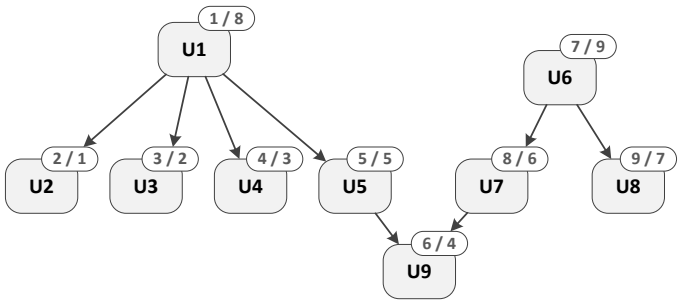
7.1.2.3 Προσδιορισμός Σειριακού Πλάνου Εκτέλεσης

Κάθε μονάδα συμπερασμού περιέχει την λογική συμπερασμού που θα χρησιμοποιηθεί προκειμένου να υπολογιστεί η αληθοτιμή του κόμβου εξόδου. Παρ΄ όλ΄ αυτά, για να γίνει αυτό χρειάζονται οι τιμές αληθείας όλων των κόμβων εισόδου. Με τη σειρά τους και αυτοί οι κόμβοι μπορεί να πρέπει να υπολογιστούν από άλλες μονάδες συμπερασμού οι οποίες επίσης χρειάζονται τις τιμές των κόμβων εισόδου για να υπολογιστούν. Επομένως, υπάρχουν εξαρτήσεις ανάμεσα στις μονάδες συμπερασμού που εξάγονται από ένα μοντέλο SEB, γεγονός που συνεπάγεται ότι η αποτίμηση των μονάδων θα πρέπει να γίνει με μία σειρά που θα σέβεται αυτές τις εξαρτήσεις. Στην ενότητα αυτή παρουσιάζουμε τον μηχανισμό δημιουργίας σειριακών πλάνων εκτέλεσης των μονάδων συμπερασμού.

Ορισμός 7 *Λέμε ότι μία μονάδα συμπερασμού U_a απαιτεί άμεσα την μονάδα συμπερασμού U_b, και το συμβολίζουμε ως $U_a \xrightarrow{req} U_b$ ανν $Out[U_b] \in In[U_a]$.*

Ορισμός 8 *Λέμε ότι μία μονάδα συμπερασμού U_a απαιτεί την μονάδα συμπερασμού U_b, και το συμβολίζουμε ως $U_a \xrightarrow{req*} U_b$ ανν $U_a \xrightarrow{req} U_b$ ή υπάρχει μία μονάδα συμπερασμού U_k τέτοια ώστε $U_a \xrightarrow{req} U_k$ και $U_k \xrightarrow{req*} U_b$.*

Ο τελεστής $\xrightarrow{req*}$ αποτελεί ουσιαστικά ένα τρόπο για την καταγραφή των εξαρτήσεων, οι οποίες σχηματίζουν έναν κατευθυνόμενο γράφο από μονάδες συμπερασμού στον οποίο μία μονάδα εξαρτάται *μόνο* από τις μονάδες που βρίσκονται πριν από αυτήν στα μονοπάτια που συμμετέχει αυτή η μονάδα. Ονομάζουμε αυτόν τον κατευθυνόμενο γράφο



Σχήμα 7-2: Παράδειγμα γράφου εξάρτησης μονάδων συμπερασμού

Γράφο Εξάρτησης Μονάδων Συμπερασμού (Reasoning Unit Dependency Graph) και ένα παράδειγμα δίνεται στο Σχήμα 7-2, στο οποίο υπάρχουν οι παρακάτω εξαρτήσεις:

$$U_1 \xrightarrow{req} U_2 \ , \ U_1 \xrightarrow{req} U_3 \ , \ U_1 \xrightarrow{req} U_4 \ , \ U_1 \xrightarrow{req} U_5$$
$$U_6 \xrightarrow{req} U_7 \ , \ U_6 \xrightarrow{req} U_8 \ , \ U_5 \xrightarrow{req} U_9 \ , \ U_7 \xrightarrow{req} U_9$$

Οι δύο αριθμοί που βρίσκονται πάνω σε κάθε κόμβο στον γράφο του Σχήματος 7-2) είναι οι αριθμοί που ανατίθενται σε κάθε μονάδα όταν διασχίζουμε τον γράφο χρησιμοποιώντας: *α)* έναν αλγόριθμο προ-διατεταγμένης διάσχισης, και *β')* έναν αλγόριθμο τοπολογικής ταξινόμησης. Για παράδειγμα, για τον κόμβο U_1, ο πρώτος αριθμός (1) αντιστοιχεί στην σειρά επίσκεψης του κόμβου όταν χρησιμοποιείται προ-διατεταγμένη διάσχιση, ενώ ο δεύτερος αριθμός (8) αντιστοιχεί στην σειρά επίσκεψης του κόμβου όταν χρησιμοποιείται κάποιος αλγόριθμος τοπολογικής διάταξης.

Ορισμός 9 *Λέμε ότι ένα μονοπάτι μονάδων συμπερασμού U_1, U_2, $\cdots$, U_n είναι ένα 'κατάλληλο' μονοπάτι, εάν για κάθε ζεύγος U_i, U_j μονάδων συμπερασμού στην ακολουθία, το U_i εμφανίζεται πριν το U_j στην περίπτωση που $U_j \xrightarrow{req*} U_i$.*

Εάν υπάρχουν κυκλικές εξαρτήσεις σε ένα σύνολο μονάδων συμπερασμού, δεν υπάρχει κατάλληλο μονοπάτι αυτών των μονάδων στον γράφο. Παρ' όλ' αυτά, στο πλαίσιο της παρούσας διατριβής θεωρούμε ότι στα μοντέλα SEB οι ακμές συνεισφοράς δεν δημιουργούν κυκλικές. Έτσι, δοθέντος ενός κατευθυνόμενου γράφου όλων των μονάδων συμπερασμού μπορούμε να εξάγουμε μία ακολουθία στην οποία κάθε μονάδα εμφανίζεται μετά από όλες τις άλλες μονάδες από τις οποίες εξαρτάται. Αυτό το σειριακό πλάνο εκτέλεσης μπορεί να υπολογιστεί με την χρήση ενός αλγορίθμου τοπολογικής διάταξης (π.χ. ο αλγόριθμος του Tarjan [96]). Για παράδειγμα, για τον γράφο του Σχήματος 7-2, ένα σειριακό πλάνο μπορεί να είναι το παρακάτω:

$$U_2 \ , \ U_3 \ , \ U_4 \ , \ U_9 \ , \ U_5 \ , \ U_7 \ , \ U_8 \ , \ U_1 \ , \ U_6$$

Επιπλέον, δοθέντος ενός συνόλου $A = \{U_1, U_2, \cdots, U_n\}$ από μονάδες συμπερασμού ορίζουμε τα παρακάτω δύο σύνολα κόμβων (μεταβλητών):

$$I[A] = \bigcup_{i=1}^{n} In[U_i] - \bigcup_{i=1}^{n} Out[U_i] \tag{7.1}$$

$$O[A] = \bigcup_{i=1}^{n} Out[U_i] - \bigcup_{i=1}^{n} In[U_i] \tag{7.2}$$

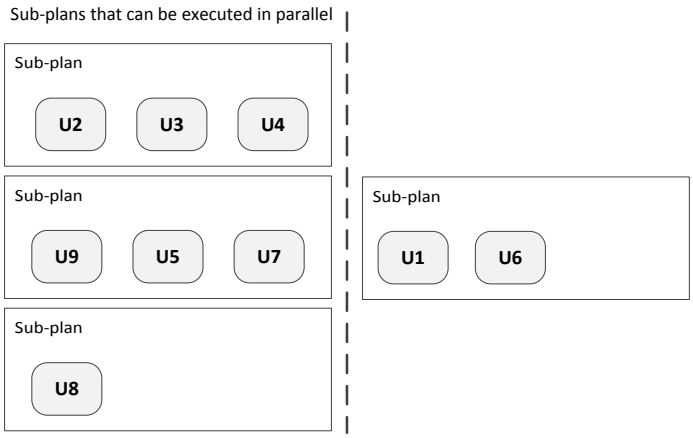
όπου το $I[A]$ περιέχει τους κόμβους του μοντέλου που εμφανίζονται μόνο ως είσοδος στις μονάδες συμπερασμού του συνόλου A (δηλαδή το $I[A]$ είναι το σύνολο των φύλλων), ενώ το $O[A]$ περιέχει τους κόμβους που εμφανίζονται μόνο ως έξοδος στις μονάδες του συνόλου A (δηλαδή το $O[A]$ περιέχει τις ρίζες). Επομένως, χρησιμοποιώντας μία ακολουθία από μονάδες συμπερασμού και μία ανάθεση αληθοτιμών για τα στοιχεία του συνόλου $I[A]$, μπορούμε να υπολογίσουμε τις αληθοτιμές των στοιχείων του συνόλου $O[A]$, δηλαδή των ριζών του μοντέλου.

7.1.2.4 Προσδιορισμός Παράλληλου Πλάνου Εκτέλεσης

Εκτός από τον προσδιορισμό μία ακολουθίας μονάδων συμπερασμού, ο σειριακός υπολογισμός των οποίων εξασφαλίζει την αποτίμηση των ριζών, προτείνουμε σε αυτήν την ενότητα έναν αλγόριθμο ο οποίος επιτρέπει τον προσδιορισμό ενός πλάνου εκτέλεσης που υποστηρίζει την παράλληλη εκτέλεση των μονάδων συμπερασμού.

Μετρικές Μονάδων Συμπερασμού Στο σημείο αυτό θα ορίσουμε ένα σύνολο μετρικών για τις μονάδες συμπερασμού οι οποίες θα χρησιμοποιηθούν στη συνέχεια για τον προσδιορισμό επιμέρους πλάνων τα οποία μπορούν να εκτελεστούν παράλληλα. Οι ακόλουθες μετρικές αποτελούν χαρακτηριστικό κάθε μονάδας στο μοντέλο συμπερασμού:

- DFS Label (dfs-label), είναι ο αριθμός που ανατίθεται στην μονάδα συμπερασμού όταν διασχίζουμε τον γράφο με την χρήση του αλγορίθμου προ-διατεταγμένης διάσχισης,
- topological sorting label (to-label), είναι ο αριθμός που ανατίθεται στην μονάδα συμπερασμού όταν διασχίζουμε τον γράφο με την χρήση ενός αλγορίθμου τοπολογικής διάταξης,
- συνολικός αριθμός παιδιών (child-all), είναι το πλήθος των μονάδων συμπερασμού από τις οποίες μία μονάδα εξαρτάται και συνδέονται με ένα μονοπάτι μήκους 1 (τις οποίες στο εξής θα αναφέρουμε ως *άμεσες εξαρτήσεις*),



Σχήμα 7-3: Το πλάνο εκτέλεσης που επιστρέφει ο αλγόριθμος για $M = 3$

- αριθμός παιδιών τα οποία έχουμε επισκεφθεί (child-proc), είναι το πλήθος των άμεσων εξαρτήσεων τις οποίες έχει ήδη επισκεφθεί ο αλγόριθμος, που αρχικοποιείται στην τιμή 0,

- πρώτος κόμβος εξάρτησης του τρέχοντος παράλληλου επιπέδου (level-first-dep), είναι το όνομα του πρώτου κόμβου που επισκέπτεται ο αλγόριθμος και από τον οποίο εξαρτάται η τρέχουσα μονάδα, που αρχικοποιείται στην τιμή null,

- εξάρτηση παράλληλου επιπέδου (dep-level) που αρχικοποιείται στην τιμή -1.

Οι τρεις τελευταίες μετρικές χρησιμοποιούνται από τον προτεινόμενο αλγόριθμο και οι τιμές τους ενημερώνεται καθώς ο αλγόριθμος επισκέπτεται τις μονάδες συμπερασμού. Ο αλγόριθμος προσδιορισμού πλάνου χρησιμοποιεί επίσης και την παράμετρο M, η οποία ονομάζεται *μήκος επιμέρους πλάνου* και καθορίζει ποιο είναι το μέγιστο μήκος για τα επιμέρους πλάνα.

Αλγόριθμος Προσδιορισμού Παράλληλου Πλάνου Ο Αλγόριθμος 4 στοχεύει στο να δημιουργήσει ένα σύνολο ανεξάρτητων επιμέρους πλάνων από έναν γράφο εξαρτήσεων μονάδων συμπερασμού. Κάθε επιμέρους πλάνο είναι μία ακολουθία από το πολύ M μονάδες συμπερασμού, όπου η τιμή του M μπορεί να οριστεί από τον χρήστη. Πιο συγκεκριμένα, κάθε επιμέρους πλάνο αποτελείται από ακολουθίες που εξαρτώνται μεταξύ τους και κατά συνέπεια μπορούν να εκτελεστούν παράλληλα. Για παράδειγμα, στο Σχήμα 7-3 δίνεται το πλάνο εκτέλεσης που επιστρέφει ο προτεινόμενος αλγόριθμος για $M = 3$ για τον γράφο εξαρτήσεων του Σχήματος 7-2. Σύμφωνα με αυτό το πλάνο, ο υπολογισμός θα ολοκληρωθεί σε δύο φάσεις, όπου κατά την πρώτη φάση υπάρχουν 3 επιμέρους πλάνα που μπορούν να εκτελεστούν παράλληλα, και κατά την δεύτερη φάση υπολογίζονται όλα τα υπόλοιπα επιμέρους πλάνα.

Η σχετική διαδικασία δίνεται στον Αλγόριθμο 4 (αλγόριθμος *Προσδιορισμού Πλάνου*). Ο αλγόριθμος διατρέχει όλους τους κόμβους που περιέχονται στην λίστα "currentNodesList" (γραμμή 10), η οποία αρχικά περιέχει όλους τους κόμβους φύλλα. Σταδιακά κόμβοι προστίθενται ή αφαιρούνται από τη λίστα, και η διαδικασία ολοκληρώνεται όταν η λίστα αδειάσει (γραμμή 6). Αρχικά αυτή η λίστα περιέχει όλους τους κόμβους φύλλα (γραμμή 4), δηλαδή τις μονάδες που δεν εξαρτώνται σε καμία άλλη μονάδα. Κάθε φορά που επισκεπτόμαστε έναν κόμβο, ο αλγόριθμος ελέγχει εάν κάποιος από τους κόμβους που εξαρτάται από αυτόν είναι τώρα 'ελεύθερος'. Ένας κόμβος γίνεται ελεύθερος όταν *όλες* οι μονάδες από τις οποίες εξαρτάται έχουν προστεθεί σε κάποιο επιμέρους πλάνο. Αυτό γίνεται καλώντας τη συνάρτηση "processParentNodes" (γραμμή 12) που δίνεται στον Αλγόριθμο 5. Αξίζει να σημειωθεί ότι σε αυτήν τη συνάρτηση όλες οι μονάδες που εξαρτώνται από την τρέχουσα μονάδα ελέγχονται, και αν είναι ελεύθερες είτε προστίθενται στο σύνολο "freeNodesSet" ώστε να ληφθούν υπόψη στην επόμενη επανάληψη του αλγορίθμου, είτε προστίθενται στην λίστα "loopNodesList" έτσι ώστε να αποτελέσει μέρος του τρέχοντος συνόλου επιμέρους πλάνων. Μόλις το μέγεθος της "loopNodesList" γίνει ίσο με την παράμετρο M (γραμμή 13) ένα νέο επιμέρους πλάνο προστίθεται στο σύνολο επιμέρους πλάνων (γραμμή 14) και η "loopNodesList" τίθεται ίση με την κενή λίστα (γραμμή 15). Είναι σημαντικό να σημειωθεί ότι όλες οι μονάδες στην "loopNodesList" είναι ταξινομημένες ως προς την μετρική to-label, και έτσι λαμβάνονται υπόψη όλες οι εξαρτήσεις ανάμεσα στις μονάδες που περιέχονται στην "loopNodesList".

Εάν έχουμε επισκεφθεί όλους τους κόμβους στην "currentNodesList" δημιουργείται ένα νέο επιμέρους πλάνο το οποίο περιέχει όλες τις μονάδες της λίστας "loopNodesList" (γραμμές 18-20) και τότε η "currentNodesList", που τώρα είναι άδεια, γεμίζει με όλες τις ελεύθερες μονάδες που περιέχονται στο σύνολο "freeNodesSet" (γραμμή 23). Η διαδικασία τερματίζει όταν έχουμε επισκεφθεί όλες τις μονάδες του γράφου εξαρτήσεων.

Έχοντας περιγράψει την λειτουργία του Αλγορίθμου 4, συνεχίζουμε με την περιγραφή των βημάτων του Αλγορίθμου 5, ο οποίος εκτελεί τον εντοπισμό των κόμβων που είναι ελεύθεροι μετά από κάθε επανάληψη του κύριου αλγορίθμου (Αλγόριθμος 4). Ο Αλγόριθμος 5 συνεχίζει διατρέχοντας όλους τους κόμβους των μονάδων που συνδέονται με την μονάδα *cn* μέσω ενός μονοπατιού μήκους 1 (γραμμή 1). Για κάθε έναν από αυτούς, αρχικά ενημερώνει την τιμή των μετρικών "level-first-dep" και "dev-level"(γραμμές 3 -6), εάν αυτή η μονάδα είναι η πρώτη μονάδα που επισκεπτόμαστε και ανήκει στις άμεσες εξαρτήσεις της. Στη συνέχεια αυξάνει των αριθμό των άμεσων εξαρτήσεων που έχει επισκεφθεί ο αλγόριθμος κατά ένα (γραμμή 7). Στη συνέχεια, ελέγχει εάν *α*) όλα τα παιδιά είναι ήδη ελεύθερα (γραμμή 8), *και β*') η πρώτη ελεύθερη μονάδα από τα παιδιά της τρέχουσας μονάδας περιέχεται στην λίστα "loopNodesList" (γραμμή 9), *και γ*') το

Algorithm 4 Plan Formulation Algorithm

Input : L: leaf nodes, M sub-pal size
Output : *plan*: List[Set[List[Node]]]

```
1:  plan ← emptyList
2:  freeNodesSet ← emptySet
3:  currentNodesList ← emptyList[dfs-label-comparator]
4:  currentNodesList.addAll(L)
5:  currentParLevel ← 0
6:  while currentNodesList IS NOT EMPTY do
7:    currentParLevel++
8:    subPlanSet ← emptySet
9:    loopNodesList ← emptyList[to-label-comparator]
10:   for all currentNode ∈ currentNodesList do
11:     loopNodesList.add(currentNode)
12:     processParentNodes(currentNode, freeNodesSet,
            loopNodesList, M, currentParLevel)
13:     if loopNodesList.size == M then
14:       subPlanSet.add(loopNodesList)
15:       loopNodesList.removeAll()
16:     end if
17:   end for
18:   if loopNodesList IS NOT EMPTY then
19:     subPlanSet.add(loopNodesList)
20:     loopNodesList.removeAll()
21:   end if
22:   currentNodesList.removeAll()
23:   currentNodesList.addAll(freeNodesSet)
24:   freeNodesSet ← emptySet
25:   plan.add(subPlan)
26: end while
27: return plan
```

μέγεθος του επιμέρους πλάνου δεν έχει υπερβεί το μέγιστο επιτρεπτό μέγεθος (γραμμή 9) , και αν αυτό ισχύει τότε καλεί αναδρομικά την "processParentNodes" για τον κόμβο p (γραμμή 11). Αλλιώς, εάν ο τρέχον κόμβος πατέρας είναι ελεύθερος, αλλά είτε το μέγεθος του επιμέρους πλάνου είναι ίσο με M, είτε εάν το πρώτο ελεύθερο παιδί ανήκει σε άλλο επιμέρους πλάνο, απλά προσθέτει τον κόμβο p στο σύνολο των ήδη ελεύθερων κόμβων (γραμμή 13).

Παράδειγμα Δημιουργίας Επιμέρους Πλάνου Η σειρά με την οποία επισκεπτόμαστε τους κόμβους του γράφου του Σχήματος 7-2 με την χρήση του προτεινόμενου αλγορίθμου, και κατά συνέπεια τα βήματα για την δημιουργία του πλάνου εκτέλεσης του

Algorithm 5 processParentNodes

Input : *cn*: unit node, $freeNodesSet$: set of units
$loopNodeList$: List of units, M: sub-plan size
$currentParLevel$

```
1: parentNodes ← cn.parentNodes
2: for all p ∈ parentNodes do
3:    if p.dev-level ≠ currentParLevel then
4:       p.level-first-dep = cn
5:       p.dev-level = currentParLevel
6:    end if
7:    p.child-proc++
8:    if p.child-proc == p.child-all then
9:       if p.level-first-dep in loopNodesList AND
             loopNodesList.size < M then
10:         loopNodesList.add(p)
11:         processParentNodes(p, freeNodesSet,
                 loopNodesList, M, currentParLevel)
12:      else
13:         freeNodesSet.add(p)
14:      end if
15:   end if
16: end for
17: return
```

Σχήματος 7-3, δίνονται στα Σχήματα 7-4 - 7-8. Στη συνέχεια περιγράφουμε αναλυτικά τα βήματα όταν ο αλγόριθμος εφαρμοστεί στον γράφο του Σχήματος 7-2:

Βήμα 1

Αρχικά, όταν ο βρόχος που ξεκινά στην γραμμή 6 (Αλγόριθμος 4) εκτελείται για πρώτη φορά, η λίστα "currentNodesList" περιέχει τους κόμβους U_2, U_3, U_4, U_9, U_8 που είναι φύλλα, και η μεταβλητή "currentParLevel" αυξάνεται κατά ένα (γραμμή 7 στον Αλγόριθμο 4). Έπειτα, (βλέπε Σχήμα 7-4) επισκεπτόμαστε τον κόμβο U_2 και τον προσθέτουμε στη λίστα "loopNodesList", ενώ οι τιμές για τις μεταβλητές "child-proc", "level-first-dep", και "dep-level" του κόμβου πατέρα παίρνουν τις τιμές '1', 'U_2', και '1' αντίστοιχα.

Βήμα 2

Στη συνέχεια, (Σχήμα 7-5) και συνεχίζοντας την επεξεργασία των κόμβων στη λίστα "currentNodesList", επισκεπτόμαστε τους κόμβους U_3 και U_4 και τους προσθέτουμε στην λίστα "loopNodesList",αυξάνοντας την τιμή της "child-proc" για τον κόμβο U_1 σε '2' και στη συνέχεια σε '3'. Στο σημείο αυτό, καθώς το μέγεθος της λίστας "loopN-

odesList"είναι ίσο με το μέγιστο μέγεθος όπως αυτό ορίζεται από την μεταβλητή M που είναι ίση με 3 στο παράδειγμά μας, το πρώτο επιμέρους πλάνο δημιουργείται και περιέχει τους κόμβους $[U_2, U_3, U_4]$, και αδειάζουμε τη λίστα "loopNodesList".

Βήμα 3

Ο αλγόριθμος επισκέπτεται στη συνέχεια τον κόμβο U_9 (Σχήμα 7-6) και τον προσθέτει στην λίστα "loopNodesList", θέτοντας τις τιμές των "child-proc", "level-first-dep", και "dep-level" για τον κόμβο πατέρα U_5 σε '1', 'U_9' και '1', αντίστοιχα. Καθώς τώρα η "child-proc" είναι ίση με την "child-all" για τον κόμβο U_5, και η "level-first-dep" είναι ίση με U_9 η οποία περιέχεται στην λίστα "loopNodesList", ο κόμβος U_5 προστίθεται και πάλι στη λίστα "loopNodesList". Επιπλέον, ο κόμβος U_1 που είναι ο κόμβος πατέρας του κόμβου U_5, προστίθεται στο σύνολο "freeNodesSet" καθώς τώρα όλες οι άμεσες εξαρτήσεις του έχουν προστεθεί σε κάποιο επιμέρους πλάνο.

Βήμα 4

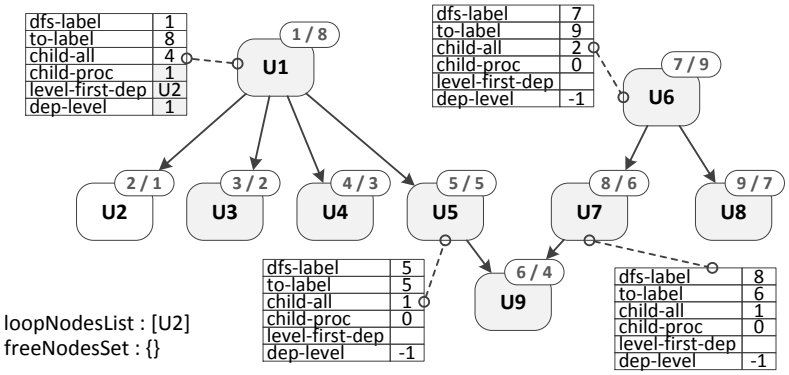
Κατά παρόμοιο τρόπο, οι τιμές των "child-proc", "level-first-dep", και "dep-level" για τον κόμβο U_7 γίνονται '1', 'U_9' και '1' αντίστοιχα (Σχήμα 7-7). Καθώς όμως η "child-proc" είναι ίση με την "child-all" για τον κόμβο U_7, ο κόμβος προστίθεται στη λίστα "loopNodesList". Για άλλη μία φορά, καθώς το μέγεθος της "loopNodesList" είναι ίσο με την M που είναι 3 καθώς περιέχει τους κόμβους U_9, U_5, U_7, δημιουργείται το δεύτερο επιμέρους πλάνο $[U_9, U_5, U_7]$ και αδειάζουμε τη λίστα "loopNodesList".

Βήμα 5

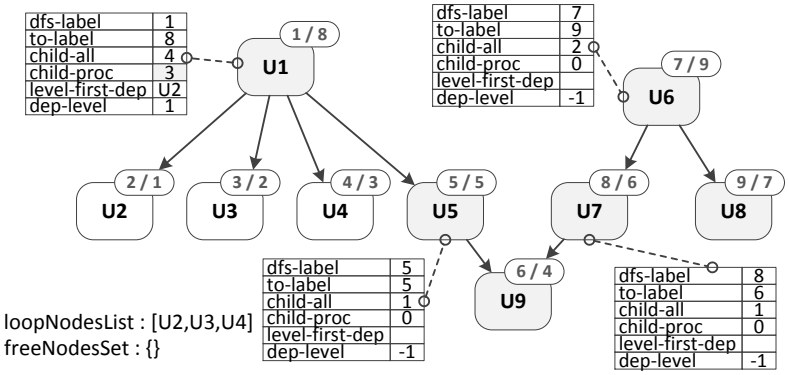
Στο βήμα 5, οι μετρικές του κόμβου U_6 ενημερώνονται (Σχήμα 7-8), επισκεπτόμαστε τον κόμβο U_8 και τον προσθέτουμε στη λίστα "loopNodesList", και θέτουμε την τιμή της "child-proc" σε '2' για τον κόμβο U_6. Καθώς έχουμε επεξεργαστεί όλα τα παιδιά του κόμβου U_6, ο U_6 προστίθεται στο σύνολο "freeNodesSet".

Βήμα 6

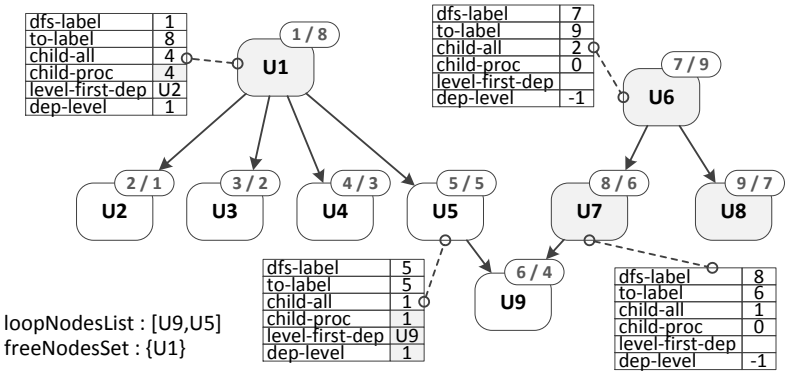
Τέλος, αφού έχουμε επισκεφτεί όλους τους κόμβους της λίστας "currentNodesList", η πρώτη εκτέλεση του βρόχου που ξεκινά στη γραμμή 6 του Αλγορίθμου 4 ολοκληρώνεται, και αδειάζει η λίστα "currentNodesList". Οι κόμβοι τις λίστας "loopNodesList", π.χ. ο κόμβος U_8, χρησιμοποιούνται για να δημιουργήσουν ένα νέο επιμέρους πλάνο (γραμμές 18-21 του Αλγορίθμου 4), και οι κόμβοι του συνόλου "freeNodesSet" προστίθενται στη λίστα "currentNodesList" η οποία είναι άδεια στο σημείο αυτό (γραμμές 22-23 του Αλγορίθμου 4). Στο σημείο αυτό, εάν ο γράφος ήταν μεγαλύτερος, ο αλγόριθμος να



Σχήμα 7-4: Βήμα 1 - Επισκεπτόμαστε τον κόμβο U_2

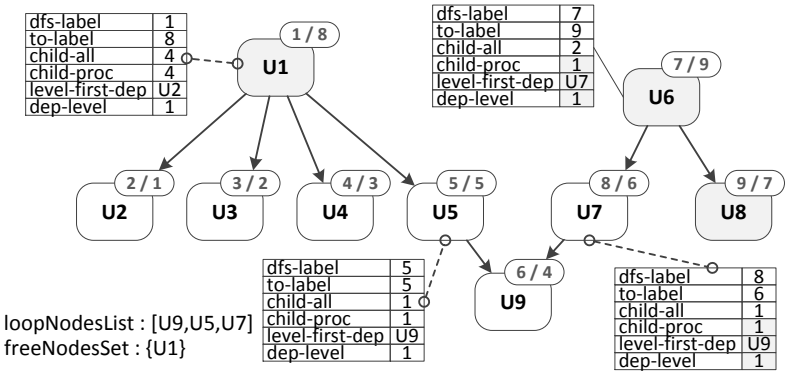


Σχήμα 7-5: Βήμα 2 - Επισκεπτόμαστε τους κόμβος U_3 και U_4

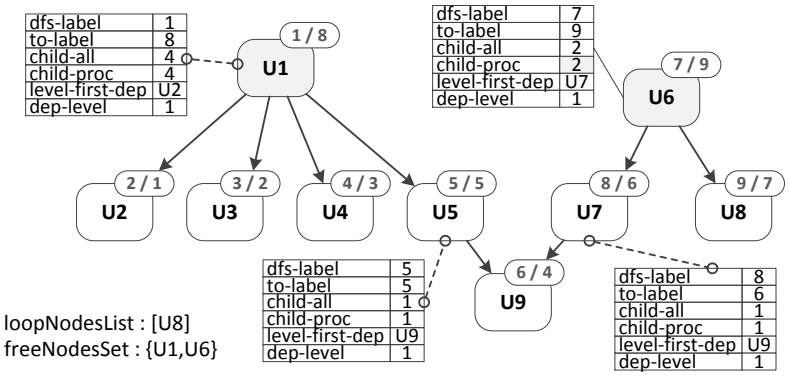


Σχήμα 7-6: Βήμα 3 - Επισκεπτόμαστε τους κόμβους U_9 και U_5

ξαναεφαρμοζόταν με την λίστα "currentNodesList" να περιέχει τα στοιχεία του συνόλου "freeNodesSet". Ο αλγόριθμος θα τερμάτιζε όταν θα είχαμε επισκεφτεί όλους τους κόμβους του γράφου.



Σχήμα 7-7: Βήμα 4 - Επισκεπτόμαστε τον κόμβο U_7



Σχήμα 7-8: Βήμα 5 - Επισκεπτόμαστε τον κόμβο U_8

7.2 Αποκατάσταση

Στην περίπτωση που το σύστημα δεν καταφέρει να πετύχει τους στόχους του, θα πρέπει να εξάγουμε ένα πλάνο αποκατάστασης το οποίο θα μπορέσει να οδηγήσει το σύστημα σε μία 'καλύτερη' κατάσταση, δηλαδή σε μία κατάσταση στην οποία περισσότεροι στόχοι ικανοποιούνται, ή στην περίπτωση που ο κάθε στόχος έχει ένα βαθμό βαρύτητας, σε μία κατάσταση στην οποία οι σημαντικότεροι στόχοι ικανοποιούνται. Αυτό μπορεί να επιτευχθεί μέσω μίας ανάλυσης από τις ρίζα προς τα φύλλα (top-down analysis) σε ένα δοθέν μοντέλο VR, καθώς μας ενδιαφέρει να βρούμε έναν συνδυασμό ενεργειών, που είναι τα φύλλα του μοντέλου VR, που είναι δυνατόν να ικανοποιήσουν τους στόχους-ρίζες. Στην υπόλοιπη ενότητα, αρχικά εισάγουμε μία μέθοδο για τον προσδιορισμό ενός βέλτιστου πλάνου αποκατάστασης στην περίπτωση που ο γράφος περιέχει μόνο crisp κόμβους. Στη συνέχεια εισάγουμε έναν γενετικό αλγόριθμο ο οποίος μπορεί να χρησιμοποιηθεί για τον προσδιορισμό μίας προσεγγιστικής λύσης στο πρόβλημα βελτιστοποίησης που μελετάμε, και επίσης μπορεί να χρησιμοποιηθεί στην περίπτωση που στο μοντέλο υπάρχουν fuzzy κόμβοι.

7.2.1 Προσδιορισμός Βέλτιστου Πλάνου

Όπως έχει ήδη περιγραφεί στην Ενότητα 6.2, δοθέντος ενός μοντέλου που περιέχει μόνο crisp κόμβους, μπορούμε να εξάγουμε μία ενιαία CNF έκφραση για όλο το μοντέλο η οποία αποτελεί μία λογική αναπαράσταση των σχέσεων που υπάρχουν ανάμεσα στους κόμβους του μοντέλου. Αυτή η CNF έκφραση μπορει στη συνέχεια να χρησιμοποιηθεί προκειμένου να εφαρμόσουμε διαδικασίες συμπερασμού προκειμένου να εντοπίσουμε πιθανούς συνδυασμούς ενεργειών, οι οποίες όταν εφαρμοστούν στο σύστημα, μπορούν να οδηγήσουν στην εκπλήρωση των στόχων ριζών του μοντέλου VR.

Η εύρεση ενός τέτοιου συνδυασμού ενεργειών ισοδυναμεί με την επίλυση του προβλήματος SAT για την CNF έκφραση, η οποία θα αναθέσει αληθοτιμές στις μεταβλητές της έκφρασης, συμπεριλαμβανομένων και των μεταβλητών που αντιστοιχούν σε ενέργειες. Η εκτέλεση των ενεργειών για τις οποίες η αληθοτιμή είναι αληθής αποτελεί και την λύση στο πρόβλημα αποκατάστασης. Παρ΄ όλ΄ αυτά, στα πλαίσια της παρούσας διατριβής θεωρούμε ότι οι κόμβοι μπορούν να έχουν βάρη τα οποία αντιστοιχούν είτε στον βαθμό προτίμησης (για του στόχους) είτε στο κόστος (για τους κόμβους εργασιών). Επομένως, δεν μας ενδιαφέρει απλά να βρούμε μία ανάθεση τιμών που ικανοποιούν την CNF έκφραση, αλλά επιπλέον μία ανάθεση που έχει τη μέγιστη δυνατή βαθμολογία λαμβάνοντας υπόψη το κόστος/την προτίμηση του κάθε κόμβου που είναι αληθής.

Το πρόβλημα προσδιορισμού του καλύτερου πλάνου είναι ένα πρόβλημα βελτιστοποίησης το οποίο μπορεί να αναχθεί σε ένα στιγμιότυπο του προβλήματος Max-SAT το οποίο ονομάζεται *Επιμερισμένο Max-SAT με βάρη* (Weighted Partial Max-SAT / WP-MAXSAT). Στο WP-MAXSAT έχουμε δύο σύνολα εκφράσεων, τις hard και soft εκφράσεις. Οι πρώτες είναι οι εκφράσεις τις οποίες μία λύση πρέπει να ικανοποιεί, ενώ οι δεύτερες είναι εκφράσεις που έχουν βάρη και η λύση θα πρέπει να ικανοποιεί μόνο εκείνες που εξασφαλίζουν το μέγιστο δυνατό άθροισμα βαρών.

Στην περίπτωση μας, η CNF έκφραση που εξάγεται από το μοντέλο αντιστοιχεί στην hard έκφραση του WP-MAXSAT, ενώ τα ζεύγη βάρος-κόμβος χρησιμοποιούνται για την δημιουργία της soft έκφρασης του προβλήματος.

7.2.2 Προσδιορισμός Προσεγγιστικού Πλάνου

Ένας εξαντλητικός αλγόριθμος για την επίλυση του WP-MAXSAT μπορεί να χρησιμοποιηθεί προκειμένου να πάρουμε το βέλτιστο πλάνο, παρ΄ όλ΄ αυτά, η δυνατότητα εφαρμογής μίας τέτοιες προσέγγισης μπορεί να είναι περιορισμένη ειδικά όταν κάποιος δουλεύει με μοντέλα που έχουν αρκετά μεγάλο μέγεθος και μεγάλη πολυπλοκότητα. Σε αυτές τις περιπτώσεις ο υπολογισμός μίας προσεγγιστικής λύσης μπορεί να είναι προτιμότερος αντί για τον υπολογισμό της βέλτιστης λύσης, καθώς η εύρεση της τελευταίας

μπορεί να συνεπάγεται μεγάλο κόστος υπολογισμού το οποίο δεν αξίζει να δαπανηθεί στην περίπτωση που μπορούμε να χρησιμοποιήσουμε μία προσεγγιστική λύση καλής ποιότητας.

Έτσι, σε αυτήν την ενότητα προτείνουμε έναν αλγόριθμο τοπικής αναζήτησης που μπορεί να χρησιμοποιηθεί κάθε φορά που χρειάζεται να προσδιοριστεί ένα πλάνο αποκατάστασης, και παράγει λύσεις των οποίων η ποιότητα συγκλίνει στην ποιότητα της βέλτιστης λύσης μετά από έναν αριθμό επαναλήψεων. Ο αριθμός των επαναλήψεων του αλγορίθμου αναζήτησης μπορεί να καθοριστεί μέσω παραμέτρων εισόδου επιτρέποντας έτσι την προσαρμογή στους χρονικούς περιορισμούς που μπορεί να υπάρχουν για την εύρεση μίας λύσης.

Πιο συγκεκριμένα, για ένα σύνολο μονάδων συμπερασμού A, ο αλγόριθμος αναζήτησης ψάχνει για λύσεις στον χώρο που ορίζεται από το σύνολο $I[A]$ και επιχειρεί να βρει γρήγορα όσο το δυνατόν καλύτερης ποιότητας λύσεις. Θα πρέπει να σημειωθεί ότι ενώ ο αλγόριθμος WP-MAXSAT πρέπει να εφαρμοστεί πάνω σε όλο το μοντέλο (δηλαδή στις μεταβλητές που αντιστοιχούν και στους εσωτερικούς κόμβους αλλά και στα φύλλα) προκειμένου να βρει την βέλτιστη λύση, ο προτεινόμενος αλγόριθμος δημιουργεί αναθέσεις αληθοτιμών μόνο για τα φύλλα μεταλλάσσοντας προηγούμενες λύσεις. Με τον τρόπο αυτό, ο χώρος αναζήτησης είναι σημαντικά μικρότερος, ειδικά για μοντέλα VR που έχουν μεγάλο βάθος και πολυπλοκότητα (δηλαδή μεγάλο λόγο εσωτερικών κόμβων προς κόμβους φύλλα).

Η διαδικασία αναζήτησης ξεκινά ελέγχοντας ένα σταθερού μεγέθους σύνολο αποθηκευμένων λύσεων, και επιλέγει την καλύτερη μεταξύ αυτών. Για να ξεκινήσει η διαδικασία ακόμα και μία μόνο λύση αρκεί, και η λύση αυτή μπορεί να υπολογιστεί με μικρό υπολογιστικό κόστος χρησιμοποιώντας έναν απλό SAT solver μία φορά κατά τον χρόνο σχεδίασης, θεωρώντας ότι όλοι οι κόμβοι είναι crisp . Το σύνολο των λύσεων εμπλουτίζεται σταδιακά καθώς νέες λύσεις εντοπίζονται κατά τον χρόνο εκτέλεσης. Όταν μία λύση επιλεγεί, ο αλγόριθμος τοπικής αναζήτησης ψάχνει στον χώρο των λύσεων $I[A]$ προκειμένου να εντοπίσει μία καλύτερη λύση. Είναι σημαντικό να σημειωθεί ότι ο χώρος $I[A]$ θα περιέχει μόνο κόμβους που αντιστοιχούν σε εργασίες και που μπορούν να είναι είτε αληθείς είτε ψευδείς, αφού οι κόμβοι εργασιών είναι πάντα crisp κόμβοι.

Ο προτεινόμενος αλγόριθμος Αναζήτησης Φύλλων (Leaves Search / LS) ξεκινά θέτοντας ως τρέχουσα λύση αυτήν που δίνεται και συνεχίζει υπολογίζοντας μια αρχική ανάθεση αληθοτιμών με βάση την δοθείσα λύση (γραμμή 1). Στη συνέχεια ο αλγόριθμος εφαρμόζει μία σειρά από μεταλλάξεις για έναν προκαθορισμένο αριθμό φορών που είναι ίσος με την παράμετρο FF επί τον αριθμό των φύλλων. Κατά τη διάρκεια των μεταλλάξεων, κάποιες από τις αναθέσεις αληθοτιμών επιλέγονται τυχαία και αντιστρέφονται από αληθείς σε ψευδείς και το ανάποδο. Ο αριθμός των αναθέσεων που αντιστρέφονται

Algorithm 6 Leaves Search Algorithm

Input: X: Initial solution, A: Reasoning Units, $I[A]$: Leaves of B, FF: flips-factor, NF: neighborhood-factor

```
1: S ← X
2: for i = 1 to FF * |L[A]| do
3:    Y ← X
4:    d ← random integer ∈ [1, NF * |L[B]|]
5:    for j = 1 to d do
6:       Y[n] ← ¬Y[n], where n is a random leaf
7:    end for
8:    evaluate(Y)
9:    if weight(Y)≥weight(X) then
10:      S ← Y
11:   end if
12: end for
13: return S
```

είναι ίσος με την παράμετρο NF πολλαπλασιασμένη με τον αριθμό των φύλλων (γραμμές 5-7). Οι νέες αναθέσεις των φύλλων αξιολογούνται χρησιμοποιώντας μία τεχνική συμπερασμού από τα φύλλα προς τις ρίζες, προκειμένου να ελέγξουμε εάν δίνουν καλύτερες λύσεις για το μοντέλο VR (γραμμή 8). Στη συνέχεια, συγκρίνουμε το συνολικό βάρος των δύο λύσεων, και εάν η νέα λύση είναι καλύτερη από της τρέχουσα, τότε αυτή γίνεται η επιλεγμένη λύση και χρησιμοποιείται ως η αρχική λύση της επόμενης επανάληψης (γραμμές 9-10). Τέλος, ο αλγόριθμος επιστρέφει την καλύτερη λύση που έχει εντοπίσει κατά τη διάρκεια της αναζήτησης.

Κεφάλαιο 8

Αξιολόγηση

Η επίδοση του προτεινόμενου περιβάλλοντος πλαισίου αξιολογήθηκε με μία σειρά πειραμάτων που χρησιμοποιούν τυχαία μοντέλα διαφορών μεγεθών και πολυπλοκοτήτων.

Πιο συγκεκριμένα, αξιολογούμε τους μηχανισμούς επαλήθευσης και αποκατάστασης ως προς τον χρόνο εκτέλεσης και το μέγεθος της μνήμης που απαιτείται για μοντέλα διαφόρων μεγεθών και δείχνουμε ότι οι απαιτήσεις χρόνου και μνήμης παραμένουν μικρές για μεγάλα μοντέλα, επιτρέποντας έτσι την εφαρμογή τους για μεγάλα μοντέλα στον χρόνο εκτέλεσης.

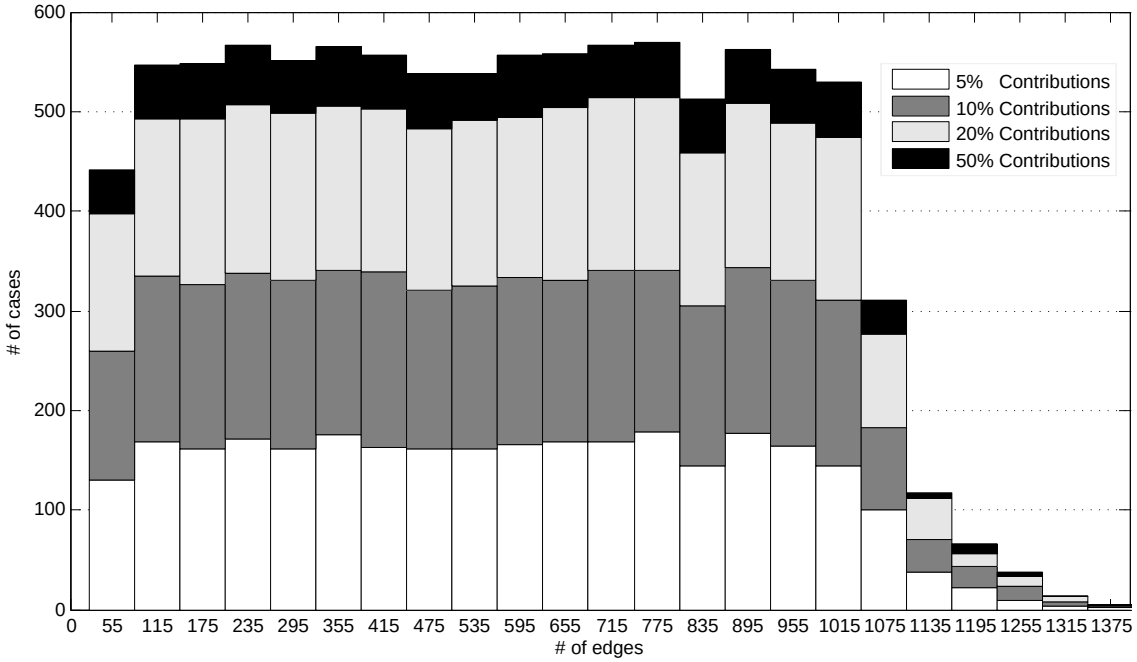
8.1 Επαλήθευση με την Χρήση Σταθμισμένων ασαφών Κανόνων

Στην περίπτωση των σταθμισμένων ασαφών κανόνων, επιπλέον του χρόνου εκτέλεσης και της μνήμης, εξετάζουμε την επίδραση των παραμέτρων των συναρτήσεων συμμετοχής στην διαδικασία συμπερασμού και δίνουμε κατευθύνσεις σχετικά με την επιλογή των τιμών τους. Επιπλέον, εξετάζουμε την ευαισθησία των αποτελεσμάτων στα βάρη που ανατίθενται στις ακμές συνεισφοράς και δείχνουμε ότι ο μηχανισμός συμπερασμού δίνει σταθερά αποτελέσματα για μικρές διακυμάνσεις των βαρών.

8.1.1 Διαμόρφωση των Πειραμάτων

Προκειμένου να εκτελεστούν τα πειράματα, σχεδιάσαμε και υλοποιήσαμε μία εφαρμογή η οποία δοθέντων συγκεκριμένων παραμέτρων μπορεί να παράξει τυχαία SEB μοντέλα. Στη συνέχεια παράγουμε μοντέλα για τις παρακάτω αναθέσεις τιμών στις παραμέτρους:

- ελάχιστος συνολικός αριθμός κόμβων στο μοντέλο: 40 - 1000, με μέγεθος διαστημάτων 20 κόμβους



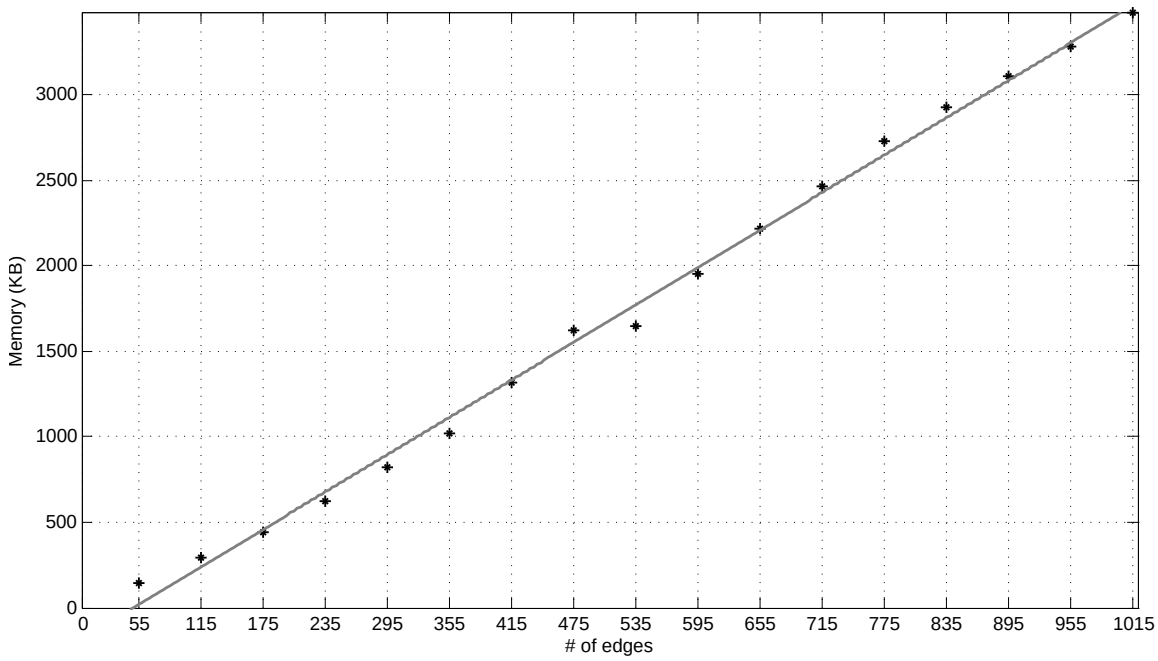
Σχήμα 8-1: Πλήθος SEB μοντέλων σύμφωνα με τον συνολικό αριθμό ακμών και το ποσοστό των ακμών συνεισφοράς

- μέγιστος αριθμός παιδιών ανά κόμβο: 4, 8, 12 ,16, 20
- λόγος AND προς OR κόμβων: 1
- ποσοστό ακμών συνεισφοράς στο μοντέλο: 5% , 10%, 20%, 50%
- πιθανότητα επιλογής συγκεκριμένου τύπου συνεισφοράς (S^P, S^N, D^P, D^P): 25 % ανά τύπο

και επίσης παράγουμε μία αρχική ανάθεση τιμών για τα φύλλα των μοντέλων, έτσι ώστε να είναι δυνατή η εξαγωγή των ασαφών γεγονότων που χρειάζονται ως είσοδος στην μηχανή συμπερασμού.

Παρήχθησαν συνολικά 9800 μοντέλα, τα οποία στη συνέχεια χωρίστηκαν σε 23 ομάδες ανάλογα με τον συνολικό αριθμό ακμών. Χρησιμοποιήσαμε τον συνολικό αριθμό ακμών στο μοντέλο ως δείκτη της πολυπλοκότητας του μοντέλου, καθώς οι ακμές περιγράφουν εξαρτήσεις ανάμεσα στους κόμβους του μοντέλου, και επομένως προσθέτουν $wf-$κανόνες στην βάση γνώσης, αυξάνοντας έτσι το μέγεθός της.

Οι ομάδες έχουν επιπλέον χωριστεί σε 4 υπό-ομάδες ανάλογα με το ποσοστό των ακμών που αντιστοιχούν σε ακμές συνεισφοράς. Το πλήθος των μοντέλων που ανήκουν σε κάθε ομάδα και υπό-ομάδα φαίνονται στο Σχήμα 8-1. Πιο συγκεκριμένα, η πρώτη ομάδα (δηλαδή η πρώτη μπάρα στο διάγραμμα) περιέχει όλα τα μοντέλα τα οποία έχουν συνολικό αριθμό ακμών στο διάστημα $[25, 85]$. Κάθε ομάδα αντιπροσωπεύεται από το



Σχήμα 8-2: Μέγεθος μνήμης που απαιτείται για την ολοκλήρωση της διαδικασίας συμπερασμού ως προς τον συνολικό αριθμό ακμών στα μοντέλα SEB

μέσο του αντίστοιχου διαστήματος, π.χ. για την πρώτη ομάδα το μέσο είναι 55. Από τα 441 μοντέλα που ανήκουν σε αυτήν την ομάδα, τα 130 έχουν 5% ακμές συνεισφοράς, τα 130 έχουν 10% ακμές συνεισφοράς, τα 138 έχουν 20% ακμές συνεισφοράς, και τέλος τα 43 έχουν 50% ακμές συνεισφοράς.

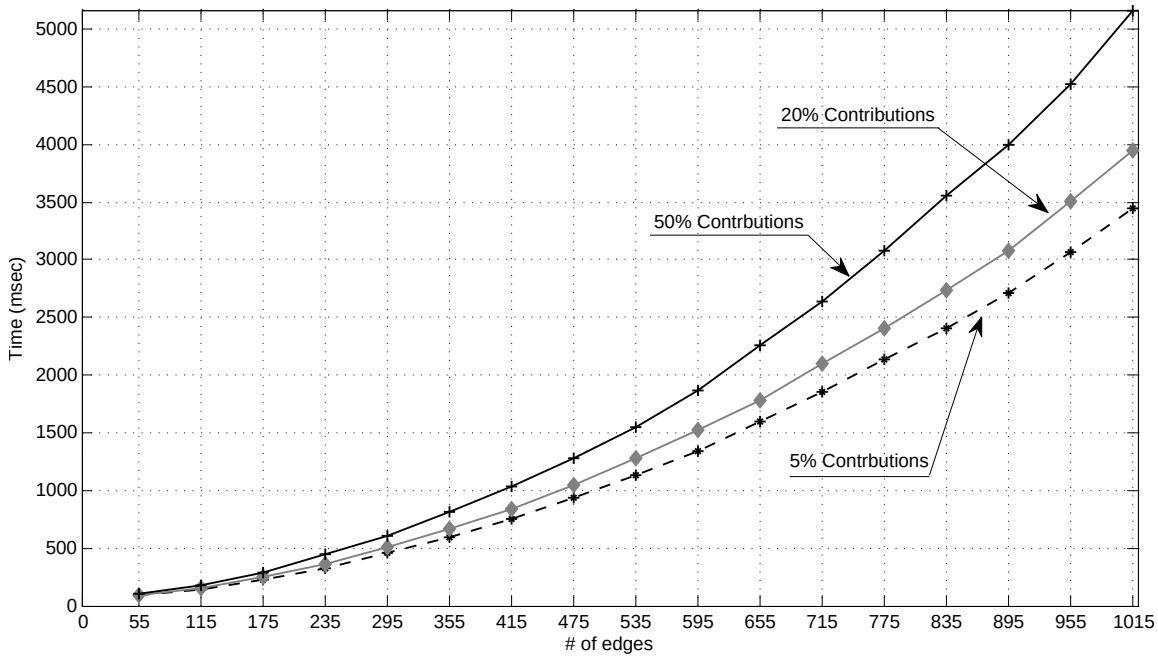
Στο Σχήμα 8-1 φαίνεται ότι η κατανομή των μοντέλων στις διάφορες ομάδες είναι σχεδόν ίδια για τις πρώτες 17 ομάδες, ενώ μόνο ένα μικρό ποσοστό των μοντέλων ανήκουν στις 6 τελευταίες. Έτσι, χρησιμοποιούμε μόνο τα μοντέλα που ανήκουν στις πρώτες 17 ομάδες για την αξιολόγηση του προτεινόμενου περιβάλλοντος πλαισίου.

8.1.2 Κλιμάκωση

Για κάθε μοντέλο SEB που ανήκει σε μία από τις πρώτες 17 ομάδες του Σχήματος 8-1 παράγουμε τους $wf-$κανόνες, εκτελούμε την διαδικασία συμπερασμού και μετράμε a) την μνήμη που χρησιμοποιείται από τον μηχανισμό συμπερασμού και b) τον χρόνο που απαιτείται για την ολοκλήρωση της διαδικασίας συμπερασμού.

Όπως φαίνεται στο Σχήμα 8-2, η μνήμη που απαιτείται για την ολοκλήρωση της διαδικασίας συμπερασμού αυξάνεται γραμμικά ως προς το μέγεθος των μοντέλων, με τιμή ίση με 125 KB για μοντέλα με μέσο αριθμό κόμβων ίσο με 55, και με τιμή ίση με 4 MB για μοντέλα με μέσο αριθμό κόμβων ίσο με 1015.

Αντίθετα, ο χρόνος αυξάνεται πολυωνυμικά ως προς τον αριθμό των ακμών όπως

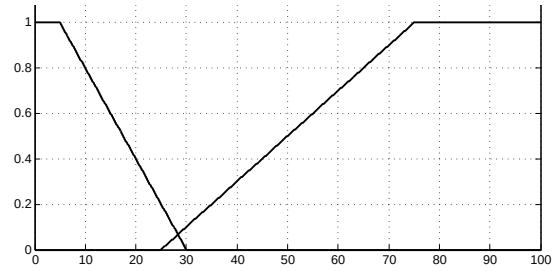


Σχήμα 8-3: Ο χρόνος που απαιτείται για την ολοκλήρωση της διαδικασίας συμπερασμού ως προς τον συνολικό αριθμό ακμών στα μοντέλα SEB

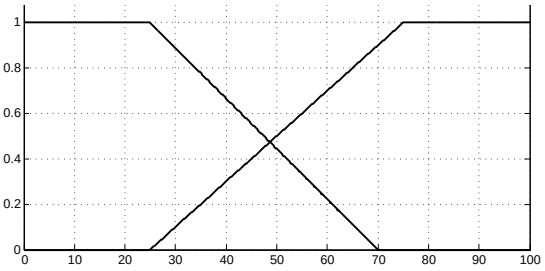
αυτό φαίνεται και στο Σχήμα 8-3. Επιπλέον, για μοντέλα με τον ίδιο αριθμό ακμών, καθώς το ποσοστό των ακμών συνεισφοράς αυξάνεται, ο χρόνος που απαιτείται για την ολοκλήρωση της διαδικασίας συμπερασμού αυξάνεται επίσης. Το γεγονός αυτό δείχνει ότι η προσθήκη ακμών συνεισφοράς αυξάνει την πολυπλοκότητα της βάσης γνώσης που χρησιμοποιείται από την μηχανή συμπερασμού. Σε κάθε περίπτωση πάντως, ο χρόνος και η μνήμη που απαιτείται παραμένουν σε χαμηλά επίπεδα ακόμα και για μεγάλα μοντέλα.

8.1.3 Επίδραση των Παραμέτρων των Συναρτήσεων Συμμετοχής

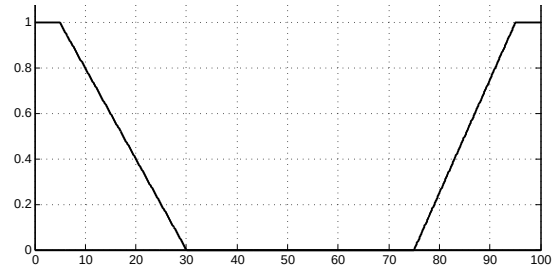
Στην ενότητα αυτή μελετάμε την επίδραση των παραμέτρων a_L, b_L, b_H, a_H στις τελικές τιμές που υπολογίζονται μέσω της διαδικασίας απο-ασαφοποίησης που περιγράφεται στην Ενότητα 7.1.1.3. Όπως αναφέρεται στην Ενότητα 7.1.1.3 και φαίνεται στο Σχήμα 7-1, οι τιμές των τεσσάρων παραμέτρων a_L, b_L, b_H, a_H καθορίζουν την περιοχή κάτω από τις συναρτήσεις συμμετοχής, η οποία εν συνεχεία χρησιμοποιείται για τον υπολογισμό της $x-$συντεταγμένης του κέντρου βάρους C_x, για δοθείσες τιμές των LowSat(g) και HighSat(g) για έναν κόμβο g. Για να αξιολογήσουμε την επίδραση που έχουν στα τελικά αποτελέσματα διαφορετικές συναρτήσεις συμμετοχής (δηλαδή οι παράμετροι a_L, b_L, b_H, a_H), θεωρούμε τις τέσσερις αντιπροσωπευτικές κλάσεις συναρτήσεων συμμετο-



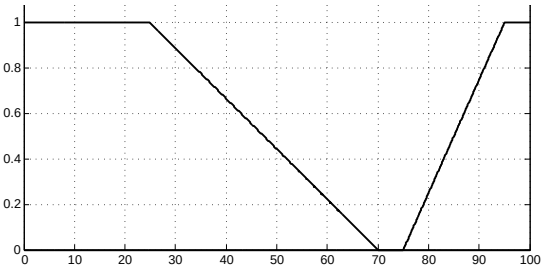
(α′) $a_L = 5$, $b_L = 30$, $b_H = 25$, $a_H = 75$



(β′) $a_L = 25$, $b_L = 70$, $b_H = 25$, $a_H = 75$



(γ′) $a_L = 5$, $b_L = 30$, $b_H = 75$, $a_H = 95$

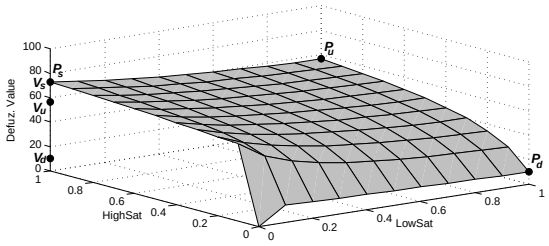


(δ′) $a_L = 25$, $b_L = 70$, $b_H = 75$, $a_H = 95$

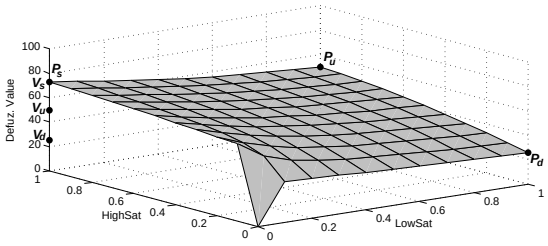
Σχήμα 8-4: Συνδυασμοί συναρτήσεων συμμετοχής για τις LowSat/HishSat που χρησιμοποιούνται για την αξιολόγηση της επίδρασης των παραμέτρων a_L, b_L, b_H, a_H στις υπολογιζόμενες τιμές

χής που φαίνονται στο Σχήμα 8-4. Για κάθε έναν από αυτούς τους τέσσερις τύπους, υπολογίζουμε την τιμή του βαθμού ικανοποίησης που προκύπτει μέσω της διαδικασίας απο-ασαφοποίησης για διάφορους συνδυασμούς τιμών των LowSat και HighSat. Τα αποτελέσματα παρουσιάζονται στο Σχήμα 8-5, όπου κάθε επιμέρους σχήμα δείχνει τους βαθμούς ικανοποίησης για τις αντίστοιχες συναρτήσεις συμμετοχής (π.χ. στο Σχήμα 8-5(α′) φαίνονται τα αποτελέσματα για τις συναρτήσεις συμμετοχής του Σχήματος 8-4(α′)).

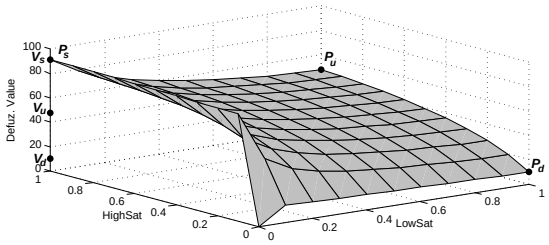
Ας θεωρήσουμε αρχικά την τιμή της μεταβλητής C_x (δηλαδή τον βαθμό ικανοποίησης) που υπολογίζεται για τις παρακάτω ακραίες περιπτώσεις: *a*) $\text{LowSat}(g) = 0$ και $\text{HighSat}(g) = 1$ (πλήρης επιτυχία επίτευξης του στόχου) που συμβολίζεται ως V_s, *b*) $\text{LowSat}(g) = 1$ και $\text{HighSat}(g) = 1$ (άγνωστη τιμή) που συμβολίζεται ως V_u, και *c*) $\text{LowSat}(g) = 1$ και $\text{HighSat}(g) = 0$ (πλήρης αποτυχία επίτευξης του στόχου) που συμβολίζεται ως V_d. Οι τιμές των V_s, V_u ανδ V_d είναι οι $z-$συντεταγμένες των σημείων P_s, P_u και P_d αντίστοιχα τα οποία φαίνονται στο Σχήμα 8-5. Αυτές οι τιμές ορίζουν δύο διαστήματα στον άξονα των z (δηλαδή στην διάσταση "Defuz. Value"), τα διαστήματα $[V_d, V_u)$ και $(V_u, V_s]$. Στόχοι με βαθμό ικανοποίησης στο διάστημα $[V_d, V_u)$ είναι στόχοι που δεν ικανοποιούνται, ενώ στόχοι με βαθμό ικανοποίησης στο διάστημα $(V_u, V_s]$ είναι στόχοι που ικανοποιούνται. Μόλις οριστούν τα διαστήματα, μπορούν να επιλεγούν υποδιαστήματα τα οποία αντιστοιχούν σε κλάσεις αποτυχίας ή επιτυχίας (π.χ. 'Ικανοποιείται



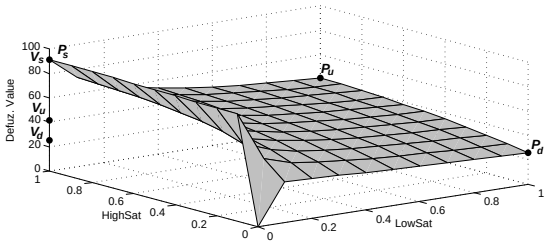
(α′) $V_s = 73\%$, $V_u = 57\%$, $V_d = 10\%$



(β′) $V_s = 73\%$, $V_u = 50\%$, $V_d = 26\%$



(γ′) $V_s = 92\%$, $V_u = 48\%$, $V_d = 10\%$



(δ′) $V_s = 92\%$, $V_u = 41\%$, $V_d = 26\%$

Σχήμα 8-5: Βαθμοί ικανοποίησης που υπολογίζονται μέσω της διαδικασίας απο-ασαφοποίησης για τέσσερις διαφορετικούς συνδυασμούς συναρτήσεων συμμετοχής των LowSat/HighSat

ελάχιστα', 'Ικανοποιείται Αρκετά') ανάλογα με τις ανάγκες της ανάλυσης. Έτσι, οι τιμές των τεσσάρων παραμέτρων ουσιαστικά ορίζουν το μήκος των δύο διαστημάτων $[V_d, V_u)$, και $(V_u, V_s]$, και κατά συνέπεια την ικανότητα να ορίζει κανείς υποκλάσεις εντός των διαστημάτων όταν χρειάζεται.

Πιο συγκεκριμένα, όταν η περιοχή κάτω από την συνάρτηση συμμετοχής LowSat είναι σχεδόν ίση με την περιοχή κάτω από τη συνάρτηση συμμετοχής HishSat (Σχήμα 8-4(β′) και Σχήμα 8-4(γ′)) η τιμή της V_u είναι σχεδόν ίση με 50% (Σχήμα 8-5(β′) και Σχήμα 8-5(γ′)). Αντίθετα, όταν η περιοχή κάτω από την HishSat είναι μεγαλύτερη από την περιοχή κάτω από την LowSat (Σχήμα 8-4(α′)) ο βαθμός ικανοποίησης που υπολογίζεται τείνει να αυξάνεται και να απομακρύνεται από το 50% ($V_u = 57\%$ στο Σχήμα 8-5(α′)). Στην αντίθετη περίπτωση, δηλαδή όταν η περιοχή κάτω από την LowSat είναι μεγαλύτερη από την περιοχή κάτω από την HighSat (Σχήμα 8-4(δ′)), ο βαθμός ικανοποίησης που υπολογίζεται τείνει να μειώνεται και να απομακρύνεται από το 50% ($V_u = 41\%$ στο Σχήμα 8-5(α′))

Επιπλέον, μία μικρή περιοχή κάτω από την συνάρτηση LowSat έχει ως συνέπεια την αύξηση του μήκους του διαστήματος $[V_d, V_u)$ καθώς οδηγεί στην μείωση της τιμής της V_d. Για παράδειγμα, $V_d = 10\%$ για το Σχήμα 8-4(α′) και το Σχήμα 8-4(γ′), ενώ $V_d = 26\%$ για το Σχήμα 8-4(β′) και το Σχήμα 8-4(δ′). Κατά παρόμοιο τρόπο, όσο μικρότερη είναι η περιοχή κάτω από την HighSat τόσο μεγαλύτερη η τιμή της μεταβλητής V_s, και κατά συνέπεια τόσο μεγαλύτερο το μήκος του διαστήματος $(V_u, V_s]$. Για παράδειγμα, V_s

$= 92\%$ για το Σχήμα 8-4(γ′) και το Σχήμα 8-4(δ′), ενώ $V_s = 73\%$ για το Σχήμα 8-4(α′) και το Σχήμα 8-4(β′).

8.1.4 Σταθερότητα

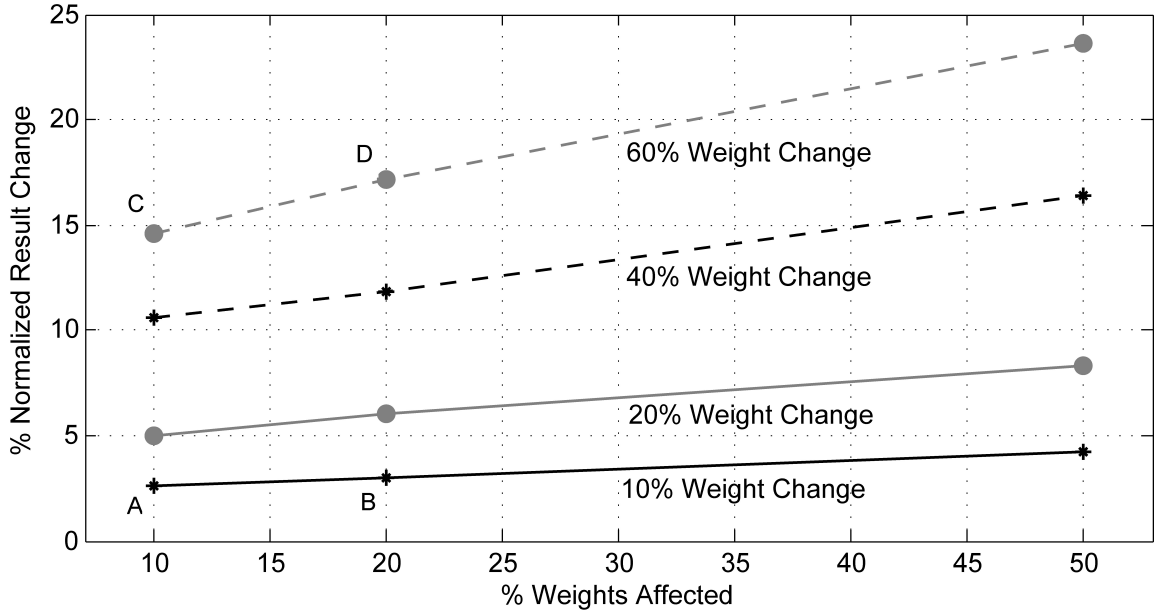
Προκειμένου να δείξουμε ότι τα αποτελέσματα που υπολογίζονται από τον μηχανισμό συμπερασμού είναι σταθερά σε μικρές μεταβολές των βαρών που μπορούν να υπάρξουν στις ακμές συνεισφοράς λόγω τις υποκειμενικότητας των εμπλεκομένων, εκτελούμε μία σειρά πειραμάτων για να διερευνήσουμε τον τρόπο που επηρεάζονται τα αποτελέσματα όταν ένα ποσοστό των βαρών μεταβάλλεται.

Χρησιμοποιούμε 16 δυαδικά δένδρα διαφόρων μεγεθών στα οποία επιλέγουμε τυχαία κόμβους οι οποίοι θα είναι οι στόχοι ενός πλήθους S^P ακμών συνεισφοράς. Το πλήθος των ακμών συνεισφοράς που προστίθεται σε ένα μοντέλο είναι το 50% του συνολικού αριθμού των ακμών σε αυτό το μοντέλο και τα βάρη επιλέγονται τυχαία στο διάστημα [1] [0.0, 0.625].

Για κάθε ακμή συνεισφοράς στο μοντέλο προσθέτουμε έναν επιπλέον κόμβο αρχής με αληθή αρχική τιμή (HighSat = 1.0, LowSat = 0.0), ενώ οι αρχικές τιμές όλων των φύλλων είναι ψευδείς, έτσι ώστε να καταγραφούν μόνο οι αλλαγές που οφείλονται στις αλλαγές των βαρών που έχουν ανατεθεί στις ακμές συνεισφοράς. Για κάθε περίπτωση μεταβάλλουμε το 10%, 20% και 50% των βαρών στο $\pm 10\%$, $\pm 20\%$, $\pm 40\%$ και $\pm 60\%$ της αρχικής τιμής, και καταγράφουμε την κανονικοποιημένη τιμή της μεταβολής για την τιμή που υπολογίζεται για το κατηγόρημα HighSat του κάθε κόμβου.

Από τα αποτελέσματα που φαίνονται στο Σχήμα 8-6 μπορούμε να παρατηρήσουμε ότι ο ρυθμός αύξησης της % κανονικοποιημένης τιμής μεταβολής είναι μικρότερη στην περίπτωση που μεταβάλλονται τα βάρη κατά 10% ("10% Weight Value Change") από ότι στην περίπτωση που μεταβάλλονται κατά 60% ("60% Weight Change"). Για παράδειγμα, όταν η τιμή του 20% των βαρών μεταβάλλεται κατά 10% (το σημείο B στο Σχήμα 8-6), η κανονικοποιημένη τιμή μεταβολής αυξάνεται στο 3% από 2.61% που είναι η αντίστοιχη τιμή στην περίπτωση που το 10% των βαρών μεταβάλλεται κατά 10% (το σημείο A στο Σχήμα 8-6), προκαλώντας μία αύξηση της τάξης του 0.40%. Αντίθετα, για την περίπτωση που οι τιμές των βαρών μεταβάλλονται κατά 60% ("60% Weight Value Change") η αντίστοιχη αύξηση (η αύξηση μεταξύ των σημείων D και C στο Σχήμα 8-6) είναι περίπου ίση με 3%. Έτσι, όσο μικρότερη η τιμή της % μεταβολής της τιμής των βαρών τόσο μικρότερος ο ρυθμός αύξησης του % της κανονικοποιημένης τιμής μεταβολής.

[1] Το άνω όριο είναι ίσο με 0.625 προκειμένου να εξασφαλίσουμε ότι όταν τα βάρη μεταβάλλονται στο $+60\%$ της αρχικής τιμής $w_{\text{ινιτ}}$, η νέα τιμή θα είναι και πάλι μικρότερη ή ίση με 1 ($1.6 w_{\text{ινιτ}} \leq 1 \Rightarrow w_{\text{ινιτ}} \leq 1/1.6 \Rightarrow w_{\text{ινιτ}} \leq 0.625$)



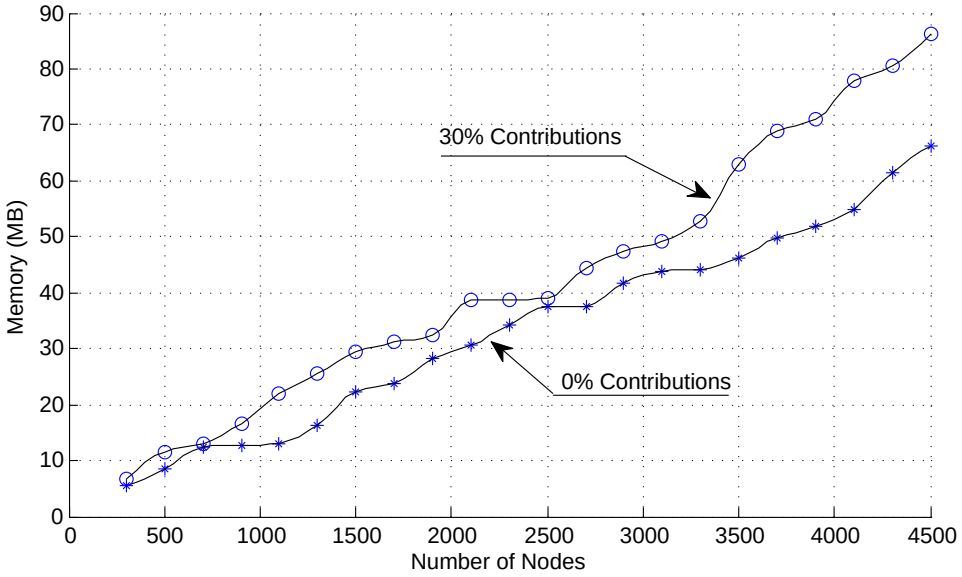
Σχήμα 8-6: % Κανονικοποιημένη Τιμή της Μεταβολής συναρτήσει του % των βαρών που μεταβάλλονται κατά ±10%, ±20%, ±40% και ±60% της αρχικής τιμής.

Επιπλέον, και στις τέσσερις περιπτώσεις το % της κανονικοποιημένης τιμής μεταβολής, που είναι ένας δείκτης του πόσο γρήγορα μεταβάλλονται τα αποτελέσματα που υπολογίζονται από τον μηχανισμό συμπερασμού συναρτήσει του ποσοστού % των βαρών που επηρεάζονται, μεταβάλλεται γραμμικά ως προς το ποσοστό % των βαρών που επηρεάζονται. Συνεπώς, τα υπολογιζόμενα αποτελέσματα μεταβάλλονται με σταθερό τρόπο καθώς το ποσοστό των βαρών που αλλάζουν αυξάνεται.

8.2 Επαλήθευση με Χρήση Μονάδων Συμπερασμού

Σε αυτήν την ενότητα παρουσιάζουμε πειραματικά αποτελέσματα σχετικά με την επίδοση του προτεινόμενου περιβάλλοντος πλαισίου. Η επίδοση του περιβάλλοντος πλαισίου αξιολογήθηκε με την χρήση τυχαίων μοντέλων τα οποία δημιουργήθηκαν για τις παρακάτω τιμές των παραμέτρων:

- συνολικός αριθμός κόμβων στο μοντέλο: 300 - 4500, με βήμα 200 κόμβων
- μέγιστος αριθμός παιδιών ανά κόμβο: 5, 10, 20
- λόγος AND προς OR κόμβων: 1
- ποσοστό ακμών συνεισφοράς στο μοντέλο: 0% , 30%

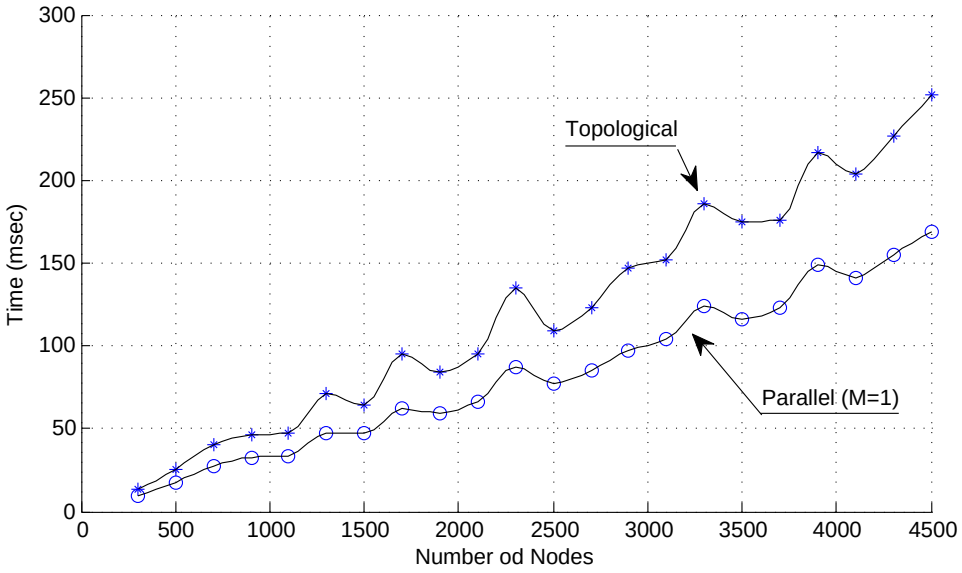


Σχήμα 8-7: Απαιτήσεις μνήμης για την ολοκλήρωση της διαδικασίας συμπερασμού όταν δεν υπάρχει καμία ακμή συνεισφοράς ή το ποσοστό των ακμών συνεισφοράς είναι 30%

- πιθανότητα επιλογής συγκεκριμένου τύπου συνεισφοράς (S^P, S^N, D^P, D^P): 25 % ανά τύπο

Επίσης δημιουργήσαμε μία τυχαία αρχική ανάθεση τιμών για τα φύλλα των μοντέλων, έτσι ώστε να είναι δυνατή η εξαγωγή των ασαφών γεγονότων που χρειάζονται ως είσοδος στην μηχανή συμπερασμού. Εν συνεχεία, χρησιμοποιήσαμε τα μοντέλα που προέκυψαν για να υπολογίσουμε την τιμή αληθείας για τους κόμβους ρίζες και καταγράψαμε τον χρόνο και τον χώρο που χρειάστηκε συναρτήσει του μεγέθους και της πολυπλοκότητας των μοντέλων (δηλαδή το πλήθος των κόμβων και των ακμών συνεισφοράς). Αρχικά χρησιμοποιήσαμε ένα πλάνο συμπερασμού το οποίο προέκυψε από την τοπολογική ταξινόμηση του γράφου εξαρτήσεων των μονάδων συμπερασμού, και στη συνέχεια επαναλάβαμε τους υπολογισμούς χρησιμοποιώντας ένα παράλληλο πλάνο συμπερασμού το οποίο προέκυψε από την χρήση του προτεινόμενου αλγορίθμου για $M = 1$. Σκοπός μας είναι να εξετάσουμε πώς επηρεάζονται ο χρόνος και το μέγεθος μνήμης που απαιτούνται για την ολοκλήρωση των υπολογισμών όταν είτε αυξάνεται η πολυπλοκότητα του μοντέλου με την προσθήκη επιπλέον ακμών συνεισφοράς, είτε χρησιμοποιείται μία διαφορετική προσέγγιση (σειριακός ή παράλληλος τρόπος υπολογισμού) για την δημιουργία του πλάνου συμπερασμού.

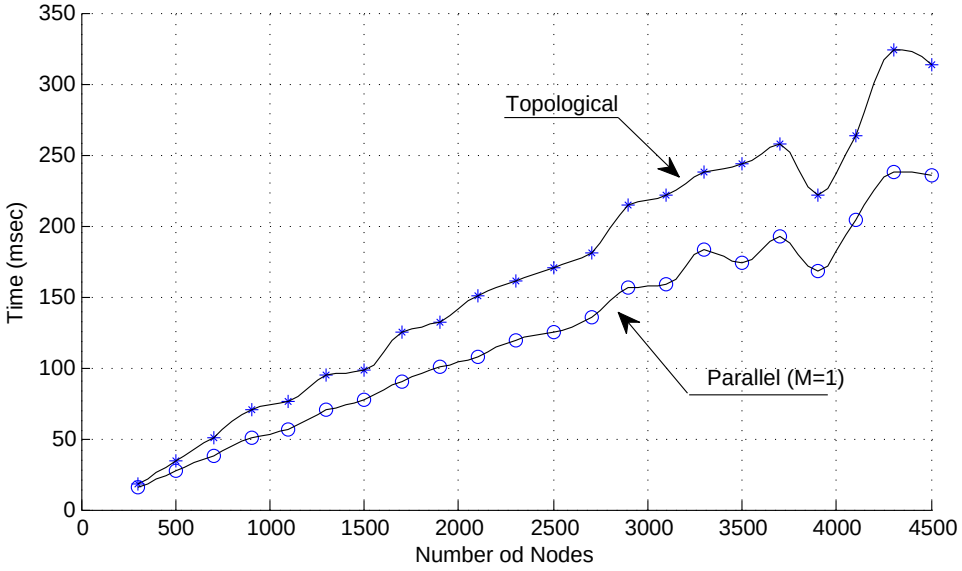
Όπως φαίνεται στο Σχήμα 8-7, το μέγεθος της μνήμης που απαιτείται όταν χρησιμοποιείται είτε ένα σειριακό είτε ένα παράλληλο πλάνο εκτέλεσης αυξάνεται σχεδόν γραμμικά ως προς το μέγεθος των μοντέλων, με μία τιμή ίση με 5.5 MB μια μοντέλα με μέσο αριθμό κόμβων ίσο με 500, και μία τιμή ίση με 66 MB για μοντέλα με μέσο



Σχήμα 8-8: Απαιτήσεις χρόνου για την ολοκλήρωση της διαδικασίας συμπερασμού όταν δεν υπάρχουν ακμές συνεισφοράς στο μοντέλο

αριθμό κόμβων ίσο με 4500 όταν δεν υπάρχουν ακμές συνεισφοράς. Ομοίως, όταν το ποσοστό των ακμών συνεισφοράς είναι ίσο με 30% η απαιτούμενη μνήμη αυξάνεται και πάλι γραμμικά ως προς το μέγεθος των μοντέλων, παρ΄ ολ΄ αυτά στην περίπτωση αυτή ξεκινά από τα 6.5 MB για μοντέλα με μέσο αριθμό κόμβων ίσο με 500, και φτάνει σε μία τιμή ίση με 88 MB για μοντέλα με μέσο αριθμό κόμβων ίσο με 4500.

Όσον αφορά στις απαιτήσεις χρόνου, τα αποτελέσματα παρουσιάζονται στο Σχήμα 8-8 και 8-9. Παρά το γεγονός ότι ο χρόνος που απαιτείται προκειμένου να ολοκληρωθεί η διαδικασία συμπερασμού είναι μικρός, το πλάνο συμπερασμού που προκύπτει από τον προτεινόμενο αλγόριθμο εξασφαλίζει ακόμα μικρότερους χρόνους εκτέλεσης. Πιο συγκεκριμένα, για μοντέλα με 4500 κόμβους ο χρόνος που απαιτείται είναι σχεδόν 85% μικρότερος από τον χρόνο που απαιτείται όταν το πλάνο αποκατάστασης παράγεται με την χρήση τοπολογικής ταξινόμησης των μονάδων συμπερασμού. Επιπλέον, καθώς ο αριθμός των ακμών συνεισφοράς αυξάνεται (Σχήμα 8-9), και κατά συνέπεια η πολυπλοκότητα των μοντέλων αυξάνεται, για μοντέλα του ίδιου μεγέθους απαιτείται περισσότερος χρόνος για την ολοκλήρωση της διαδικασίας συμπερασμού.



Σχήμα 8-9: Απαιτήσεις χρόνου για την ολοκλήρωση της διαδικασίας συμπερασμού όταν το ποσοστό των ακμών συνεισφοράς είναι 30%

8.3 Αλγόριθμοι Ορισμού Πλάνων Αποκατάστασης

Προκειμένου να αξιολογήσουμε την δυνατότητα εφαρμογής και την επίδοση του αλγορίθμου LS που παρουσιάζεται στην Ενότητα 7.2.2 πραγματοποιούμε μία σειρά πειραμάτων με την χρήση τυχαίων VR μοντέλων διαφόρων μεγεθών και πολυπλοκοτήτων. Τα μοντέλα που χρησιμοποιούνται για την αξιολόγηση περιέχουν μόνο crisp κομβους και ακμές συνεισφοράς με βάρη ίσα με 1.0, έτσι ώστε να είναι δυνατό να χρησιμοποιήσουμε τον αλγόριθμο WP-MAXSAT προκειμένου να μπορούμε να ελέγξουμε τις λύσεις που παράγονται από τον αλγόριθμο LS. Χρησιμοποιώντας αυτά τα μοντέλα αξιολογήσαμε την επίδοση του αλγορίθμου WP-MAXSAT ως προς τον χρόνο που απαιτείται για μοντέλα διαφορετικών μεγεθών. Επιπλέον, χρησιμοποιώντας τα ίδια μοντέλα αξιολογήσαμε και συγκρίναμε την επίδοση του αλγορίθμου LS ως προς τον χρόνο εκτέλεσης αλλά και την ποιότητα της λύσης για διάφορες τιμές της παραμέτρου FF.

8.3.1 Κατασκευή Μοντέλων και Μετρικές Αξιολόγησης

Προκειμένου να αξιολογήσουμε τον αλγόριθμο LS υλοποιήσαμε έναν μηχανισμό παραγωγής τυχαίων μοντέλων ενεργειών, ο οποίος για δοθείσες τιμές συγκεκριμένων παραμέτρων όπως το μέγεθος του μοντέλου, ο λόγος του πλήθους των κόμβων ενεργειών προς το πλήθος των κόμβων εργασιών κτλ. επιστρέφει ένα τυχαίο μοντέλο. Τα απο-

τελέσματα παρουσιάζονται και περιγράφονται σε αυτήν την ενότητα, ενώ οι τιμές των παραμέτρων ορίστηκαν ως εξής: *α*) μέγεθος μοντέλου ($|N|$): 20 - 300, με βήμα 20 κόμβους *β*′) λόγος AND προς OR κόμβων: 1 *γ*′) ποσοστό κόμβων που συμμετέχουν σε δυαδικούς κανόνες: 30%. Χρησιμοποιώντας τις παραπάνω τιμές παράγουμε 10 τυχαία μοντέλα για κάθε μέγεθος μοντέλου για 15 διαφορετικά μεγέθη μοντέλων. Για κάθε μοντέλο θεωρούμε 5 παραλλαγές σε κάθε μία από τις οποίες αναθέτουμε διαφορετικά βάρη στο 10% των κόμβων. Στη συνέχεια χρησιμοποιούμε τους αλγορίθμους WP-MAXSAT και WP-MINSAT ώστε να υπολογίσουμε την καλύτερη και την χειρότερη λύση αντίστοιχα. Προκειμένου να αξιολογήσουμε και να συγκρίνουμε την επίδοση της διαδικασίας αναζήτησης ως προς τα αποτελέσματα του αλγορίθμου WP-MAXSAT χρησιμοποιούμε τις παρακάτω μετρικές:

- $\mathsf{t}_{\mathsf{wp-maxsat}}$: χρόνος εκτέλεσης του αλγορίθμου WP-MAXSAT,
- $\mathsf{t}_{\mathsf{ls}}^{\mathsf{FF}}$: χρόνος εκτέλεσης του αλγορίθμου LS με παράγοντα εναλλαγών (flips factor) FF, και
- $\mathsf{Q_S} = \frac{\mathsf{W_S} - \mathsf{W_{worst}}}{\mathsf{W_{optimal}} - \mathsf{W_{worst}}}$: ποιότητα λύσης,

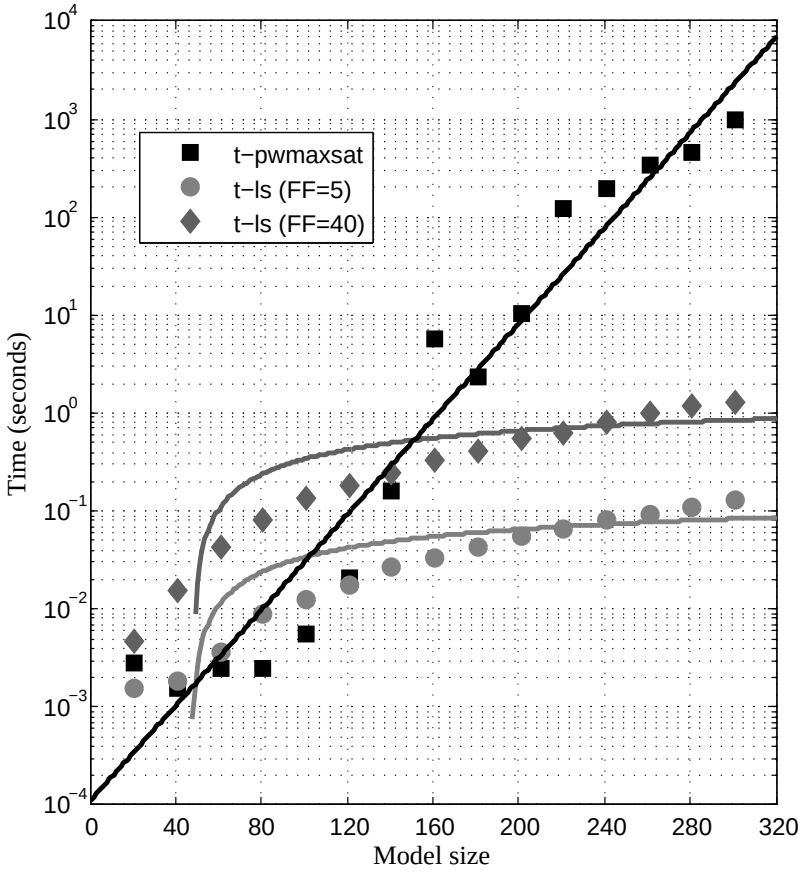
όπου $\mathsf{W_S}$ είναι το συνολικό βάρος της λύσης, και $\mathsf{W_{optimal}}$ και $\mathsf{W_{worst}}$ είναι τα συνολικά βάρη των λύσεων που υπολογίζονται με τους αλγορίθμους WP-MAXSAT και WP-MINSAT αντίστοιχα. Για κάθε παραγόμενο μοντέλο, η διαδικασία αναζήτησης εκτελείται για τις ακόλουθες τιμές παραμέτρων: $FF \in \{5, 10, 15, 20, 25, 30, 35, 40\}$, $NF = 0.2$ και μέσες τιμές υπολογίστηκαν για τις παραπάνω μετρικές.

8.3.2 Πειραματικά Αποτελέσματα

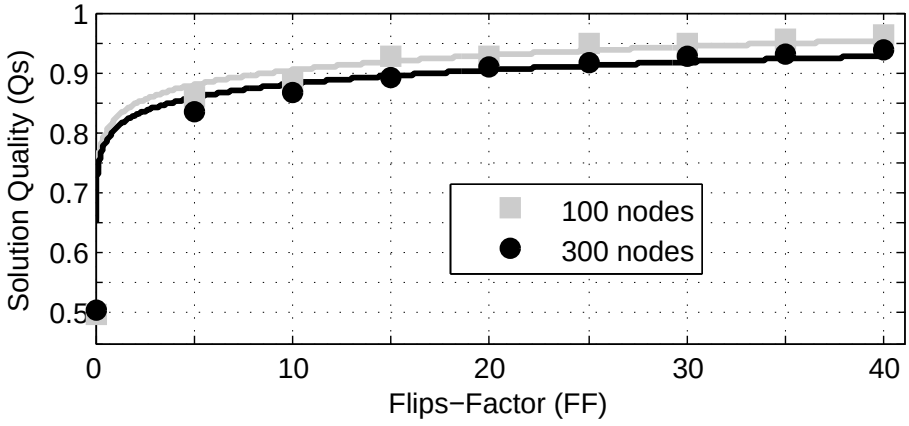
Η παρουσίαση των αποτελεσμάτων χωρίζεται σε:

- αξιολόγηση του χρόνου εκτέλεσης λαμβάνοντας υπόψη τα $\mathsf{t}_{\mathsf{wp-maxsat}}$ και $\mathsf{t}_{\mathsf{ls}}^{\mathsf{FF}}$, και
- αξιολόγηση της ποιότητας της λύσης υπολογίζοντας την τιμή του $\mathsf{Q_S}$ για μοντέλα διαφόρου μεγέθους και τιμές της παραμέτρου FF.

Χρόνος Εκτέλεσης. Στο Σχήμα 8-10 παρουσιάζεται η μέση τιμή του $\mathsf{t}_{\mathsf{wp-maxsat}}$ συναρτήσει του μεγέθους του μοντέλου. Ο χρόνος που απαιτείται από τον αλγόριθμο WP-MAXSAT αυξάνεται εκθετικά ως προς το μέγεθος του μοντέλου, κάτι που είναι αναμενόμενο λόγω της φύσης του προβλήματος. Επιπλέον, στο Σχήμα 8-10 παρουσιάζονται οι μέσες τιμές του $\mathsf{t}_{\mathsf{ls}}^{5}$ καθώς και οι μέσες τιμές του $\mathsf{t}_{\mathsf{ls}}^{40}$ για όλα τα μεγέθη μοντέλων που μελετάμε $FF = 5$ και $FF = 40$ είναι η ελάχιστη και μέγιστη αντίστοιχα



Σχήμα 8-10: Χρόνος Εκτέλεσης



Σχήμα 8-11: $\mathsf{Q_S}$ συναρτήσει της παραμέτρου FF

τιμή της παραμέτρου FF, ενώ τα αποτελέσματα για όλες τις ενδιάμεσες τιμές της παραμέτρου FF βρίσκονται ανάμεσα στις τιμές που προκύπτουν για τις δύο αυτές ακραίες περιπτώσεις. Οι τιμές για το $\mathsf{t}_{\mathsf{ls}}^{5}$ είναι σε γενικές γραμμές ανάλογες της FF, παρ' ολ' αυτά ανεξάρτητα από την τιμή της FF, η μέση τιμή του $mathsft_{ls}$ μεταβάλλεται σχεδόν

γραμμικά ως προς τον αριθμό των κόμβων. Για μεγάλα μοντέλα, ο αλγόριθμος LS είναι σημαντικά πιο γρήγορος από τον αλγόριθμο WP-MAXSAT . Για παράδειγμα, για μοντέλα με 300 κόμβους και παράγοντα εναλλαγής ίσο με 40, ο LS χρειάζεται κατά μέσο όρο $\frac{1}{1000}$ του χρόνου που χρειάζεται ο αλγόριθμος WP-MAXSAT και υπολογίζει λύσης που έχουν καλή ποιότητα ($\mathsf{Q_S} = 0.938$) όπως θα περιγραφεί στη συνέχεια.

Ποιότητα Λύσης. Στο Σχήμα 8-11 παρουσιάζεται ο τρόπος που μεταβάλλεται η τιμή του $\mathsf{Q_S}$ συναρτήσει της παραμέτρου FF για μοντέλα με μέγεθος 100 και 300 κόμβους. Η ποιότητα της αρχικής λύσης δίνεται για $FF = 0$ και είναι περίπου ίση με 0.5 και για τα δύο μεγέθη μοντέλων. Από το σημείο αυτό και μετά, καθώς η FF αυξάνεται, η ποιότητα μεταβάλλεται ασυμπτωτικά προς την βέλτιστη λύση και παίρνει την μέγιστη τιμή της για $FF = 40$ (0.961 για μέγεθος ίσο με 100 και 0.938 για μέγεθος ίσο με 300).

Κεφάλαιο 9

Επίλογος

9.1 Συζήτηση

Υπάρχουν κυρίως δύο απειλές για την εγκυρότητα της προτεινόμενης μεθόδου, το μέγεθος των μοντέλων που χρησιμοποιούνται και ο προσδιορισμός των βαρών για τις ακμές συνεισφοράς. Το πρώτο σχετίζεται με το πώς κατασκευάζονται και συντηρούνται τα μοντέλα στόχων, ειδικά αυτά που αποτελούνται από πολλές ακμές και κόμβους σαν αυτά που χρησιμοποιούνται στην αξιολόγηση του περιβάλλοντος πλαισίου. Στα πλαίσιο αυτής της διατριβής θεωρούμε ότι οι εμπλεκόμενοι στο σύστημα λογισμικού μπορούν όχι μόνο να ορίσουν τα δικά τους δένδρα στόχων, αλλά ότι επίσης μπορούν να χρησιμοποιούν προκαθορισμένα δένδρα στόχων από αποθετήρια (repositories) όπου οι απαιτήσεις μοντελοποιούνται από ειδικούς ή εξάγονται από έγγραφα προδιαγραφών και απαιτήσεων λογισμικού χρησιμοποιώντας τεχνικές εξόρυξης κειμένου [77]. Με την χρήση τέτοιων αποθετηρίων μοντέλων είναι δυνατή η σύνθεση μοντέλων στόχων ποικίλου μεγέθους και πολυπλοκότητας.

Επιπλέον, ενώ στις τεχνικές που χρησιμοποιούν *ποιοτική* συλλογιστική (qualitative reasoning), η διάδοση τιμών καθώς κινούμαστε προς τα πάνω στο δένδρο καταλήγει σχετικά γρήγορα να δίνει ασαφή αποτελέσματα [100], ένα κοινό πρόβλημα που συναντάται σχεδόν σε κάθε προσέγγιση *ποσοτικής* συλλογιστικής (quantitative reasoning) είναι το πώς κανείς αναθέτει τα βάρη στους κόμβους συνεισφοράς. Αυτό το πρόβλημα, καθώς και το κατά πόσο είναι ευκολότερο να κατανοήσει κανείς σε μεγαλύτερο βαθμό βάρη με ποιοτικά παρά με ποσοτικά χαρακτηριστικά, είναι το θέμα των [66, 67]. Στις εργασίες αυτές οι συγγραφείς παρουσιάζουν μία σειρά αποτελεσμάτων που επαληθεύουν ότι η χρήση των αριθμητικών τιμών αυξάνει την κατανόηση των μοντέλων ακόμα και για τους μη-ειδικούς, ενώ ταυτόχρονα προτείνουν μια μέθοδο για τον προσδιορισμό αυτών των τιμών.

Παράλληλα, ο μηχανισμός ασαφούς συμπερασμού που χρησιμοποιείται από το πε-

ριβάλλον πλαίσιο επιτρέπει των υπολογισμό των βαρών στους $wf-$κανόνες μέσω μίας διαδικασίας εκπαίδευσης. Η διαδικασία αυτή μπορεί επίσης να χρησιμοποιηθεί στην περίπτωσή μας προκειμένου να υπολογιστούν τιμές για τα βάρη των $wf-$κανόνων που παράγονται από το μοντέλο στόχων. Πιο συγκεκριμένα, δοθέντων των $wf-$κανόνων που παράγονται από το μοντέλο, και ενός συνόλου δεδομένων που περιέχει έγκυρους συνδυασμούς τιμών για τις συναρτήσεις μέλη HighSat και LowSat των στόχων που εμφανίζονται στους κανόνες, μπορούμε να εξάγουμε αριθμητικές τιμές για τα άγνωστα βάρη. Η προαναφερθείσα διαδικασία εκπαίδευσης για σταθμισμένους ασαφείς κανόνες περιγράφεται λεπτομερώς στο [33, σελ. 114], όπου οι συγγραφείς ορίζουν το πρόβλημα της προσαρμογής των βαρών για σταθμισμένα ασαφή λογικά προγράμματα ως ένα πρόβλημα βελτιστοποίησης που έχει ως στόχο τον καθορισμό των βαρών για τα κατηγορήματα που εμφανίζονται σε ένα σύνολο $wf-$κανόνων.

Τέλος, είναι σημαντικό να σημειωθεί ότι η διαδικασία συμπερασμού που παρουσιάζεται σε αυτή την εργασία μπορεί να εφαρμοστεί ακόμη και αν ορισμένοι από τους κόμβους φύλλα είναι μη μετρήσιμοι. Στην περίπτωση αυτή, οι κόμβοι φύλλα αντιμετωπίζονται ως άγνωστοι. Ωστόσο, στην περίπτωση αυτή θα πρέπει να ληφθεί υπόψη το ποσοστό των κόμβων που είναι άγνωστοι και χρησιμοποιούνται ως είσοδος στον μηχανισμό συμπερασμού κατά τον υπολογισμό του βαθμού ικανοποίησης. Πιο συγκεκριμένα, όσο υψηλότερο είναι το ποσοστό των άγνωστων κόμβων, τόσο μικρότερος είναι ο βαθμός εμπιστοσύνης για τα αποτελέσματα που παράγονται από τον μηχανισμό συμπερασμού.

Σύγκριση με Πιθανοθεωρητικές Προσεγγίσεις: Προσεγγίσεις που χρησιμοποιούν πιθανότητες όπως η [50] δεν είναι ισοδύναμες με εκείνες που χρησιμοποιούν ασαφή λογική και λειτουργούν περισσότερο επικουρικά καθώς μπορούν να χρησιμοποιηθούν για τον προσδιορισμό των βαρών και των σχέσεων ανάμεσα στους στόχους [109]. Κατά την γνώμη μας, υπάρχουν κάποια χαρακτηριστικά της ασαφούς λογικής τα οποία σχετίζονται κυρίως με την εκφραστικότητα και την ερμηνεία των αποτελεσμάτων [109], τα οποία καθιστούν τις τεχνικές που βασίζονται στην ασαφή λογική περισσότερο κατάλληλες για το υπό εξέταση πρόβλημα καθώς: *a*) θέλουμε να μοντελοποιήσουμε την ασάφεια που υπάρχει στην ανθρώπινη διαδικασία σκέψης και εξαγωγής συμπερασμάτων και όχι την αβεβαιότητα, και *b*) γνωρίζουμε ότι ένα γεγονός έχει συμβεί (και όχι ποια είναι η πιθανότητα να συμβεί το γεγονός αυτό) και θέλουμε να μελετήσουμε τον τρόπο που αυτό επηρεάζει τον βαθμό ικανοποίησης στόχων που βρίσκονται ψηλότερα στο δένδρο (και όχι την πιθανότητα οι στόχοι που βρίσκονται ψηλότερα στο δένδρο να ικανοποιηθούν)

Συναρτήσεις Συμμετοχής και Σημασιολογία Ασαφών Τελεστών: Η επιλογή των καταλληλότερων συναρτήσεων συμμετοχής (membership functions) είναι ένα από τα πιο σημαντικά προβλήματα στην ανάπτυξη ασαφών ελεγκτών. Ενώ υπάρχουν κάποιες γενικές κατευθυντήριες γραμμές που προτείνονται στη βιβλιογραφία, και

έχουν προταθεί μέθοδοι για την εξαγωγή των συναρτήσεων συμμετοχής με την χρήση γενετικών αλγορίθμων [53], η επιλογή των παραμέτρων για τις συναρτήσεις συμμετοχής βασίζεται κυρίως στην εμπειρία των χρηστών και τις ανάγκες της ανάλυσης [80]. Στην ανάλυση που παρουσιάζεται στην παρούσα εργασία, έχουμε δώσει κάποιες γενικές κατευθυντήριες γραμμές όσον αφορά στον τρόπο επιλογής των παραμέτρων για τις συναρτήσεις συμμετοχής στην Ενότητα 8.1.3, ωστόσο η προτεινόμενη προσέγγιση δεν αφορά μόνο συναρτήσεις συμμετοχής σαν αυτές που απεικονίζονται στο Σχήμα 6-3, και η ίδια τεχνική μπορεί να εφαρμοστεί για συναρτήσεις συμμετοχής που έχουν σιγμοειδή ή τραπεζοειδή μορφή. Το μόνο που χρειάζεται να γίνει είναι να αντικατασταθούν οι Σχέσεις 2.2 και 2.1 με τις αντίστοιχες σχέσεις με ολοκληρώματα, στην περίπτωση που η περιοχή κάτω από τη συνδυασμένη επιφάνεια των συναρτήσεων συμμετοχής δεν μπορεί να διαιρεθεί σε ένα σύνολο απλών κλειστών πολυγώνων. Επιπλέον, η μηχανή συμπερασμού που χρησιμοποιείται υποστηρίζει τρεις σταθμισμένους ασαφείς τελεστές σύζευξης (weighted fuzzy conjunction operators) οι οποίοι παράγουν αποτελέσματα που δεν αλλάζουν δραματικά για μικρές παραλλαγές των βαρών όπως φαίνεται και στο [33, σελ. 112].

9.2 Περίληψη Διατριβής

Καθώς τα συστήματα λογισμικού γίνονται όλο και πιο πολύπλοκα και γίνονται διαθέσιμα μέσω τεχνολογιών εικονοποίησης (virtualization) ή τεχνολογιών που επιτρέπουν την δυναμική παραχώρηση πόρων είναι δυνατό να προκύπτουν συστήματα Εξαιρετικά Μεγάλης Κλίμακας (Ultra Large Scale systems). Ένα βασικό ερώτημα που προκύπτει σε σύνθετα συστήματα μεγάλης κλίμακας είναι το κατά πόσο συγκεκριμένες απαιτήσεις εξακολουθούν να ισχύουν όταν τα συστήματα αλληλεπιδρούν με νέους και απρόβλεπτες τρόπους, ή όταν νέοι πόροι προστίθενται στα επιμέρους συστήματα προκειμένου να καλύψουν τις ανάγκες των εμπλεκομένων ή να βελτιώσουν την συνολική απόδοση του συστήματος. Το ερώτημα αυτό περιλαμβάνει δύο βασικά ζητήματα. Το πρώτο σχετίζεται με το κατά πόσο είναι δυνατό να εξάγει κανείς συμπεράσματα κατά τον χρόνο εκτέλεσης και εντός ενός αποδεκτού βαθμού εμπιστοσύνης ως προς το αν συγκεκριμένες απαιτήσεις συνεχίζουν να ισχύουν στο σύστημα, όταν αυτό μεταβάλλεται ως αποτέλεσμα της δυναμικής προσαρμογής του σε αλλαγές στο περιβάλλον. Για να γίνει αυτό, θα πρέπει κανείς να μπορεί να εφαρμόσει μεθόδους εξαγωγής συμπερασμάτων χρησιμοποιώντας τεχνικές που επιτρέπουν την μοντελοποίηση των εξαρτήσεων μεταξύ των απαιτήσεων λογισμικού με έναν βαθμό εμπιστοσύνης. Το δεύτερο ζήτημα αφορά στην ύπαρξη μίας μεθόδους εξαγωγής συμπερασμάτων η οποία είναι δυνατό να παράξει αποτελέσματα σε πραγματικό χρόνο ή τουλάχιστον σε σχεδόν πραγματικό χρόνο. Για να γίνει αυτό θα

πρέπει να μπορούμε να παραλληλοποιήσουμε τους υπολογισμούς που απαιτούνται από την μηχανή συμπερασμού ώστε να είναι δυνατό να κατανεμηθεί το υπολογιστικό φορτίο σε διαφορετικούς εξυπηρετητές επιτρέποντας έτσι την εκτέλεση των υπολογισμών εντός αποδεκτών χρονικών ορίων καθώς το φορτίο, τα δεδομένα ή το μέγεθος του συστήματος αυξάνονται.

Στα πλαίσια αυτής της εργασίας, προτείνουμε την χρήση ασαφών μοντέλων στόχων (fuzzy goal models) ως έναν τρόπο για την μοντελοποίηση ελλιπούς γνώσης σχετικά με τις απαιτήσεις λογισμικού και τις αλληλεξαρτήσεις τους, και εισάγουμε ένα περιβάλλον πλαίσιο συμπερασμού για ασαφή μοντέλα στόχων που μπορεί να χρησιμοποιηθεί για να αναλύσει και να αξιολογήσει τις απαιτήσεις του συστήματος κατά το χρόνο εκτέλεσης, λαμβάνοντας ως είσοδο δεδομένα που συλλέγονται καθώς το σύστημα λειτουργεί ή μεταβάλλεται. Τα ασαφή μοντέλα στόχων μοντελοποιούν τις εξαρτήσεις που υπάρχουν μεταξύ των διαφόρων απαιτήσεων καθώς και τις σχέσεις που υπάρχουν μεταξύ των δεδομένων που καταγράφονται καθώς το σύστημα λειτουργεί και τις απαιτήσεις του συστήματος. Δεδομένου ότι αυτά τα μοντέλα μπορεί να αυξηθούν σε μέγεθος καθώς περισσότερα συστήματα συνδέονται μεταξύ τους, προτείνουμε επίσης μια τεχνική που επιτρέπει την ανάλυση των εξαρτήσεων στα μοντέλα στόχων έτσι ώστε να μπορούν να προσδιοριστούν ανεξάρτητες περιοχές (ή υπο-γραφήματα) προκειμένου να μπορεί να παραλληλοποιηθεί η ασαφής συλλογιστική.

Πιο συγκεκριμένα, δοθέντος ενός μοντέλου στόχων με fuzzy και crisp κόμβους, εισάγουμε αρχικά μια μέθοδο που επιτρέπει την παραγωγή ασαφών κανόνων από τα ασαφή μοντέλα στόχων και εν συνεχεία, μια διαδικασία μετασχηματισμού μοντέλων που επιτρέπει τη δημιουργία ενός γραφήματος εξαρτήσεων, όπου κάθε κόμβος είναι μια συλλογιστική μονάδα των υπολογισμών που πρέπει να πραγματοποιηθούν, προκειμένου να υπολογιστεί ο βαθμός ικανοποίησης ενός κόμβου. Ως εκ τούτου, καθώς τα γεγονότα συλλέγονται από το σύστημα που βρίσκεται σε λειτουργία, το πλάνο συμπερασμού (reasoning plan) και οι κανόνες που έχουν εξαχθεί από το μοντέλο κατά τον χρόνο σχεδίασης, μπορούν να χρησιμοποιηθούν προκειμένου να εξαχθούν συμπεράσματα σχετικά με το κατά πόσο συγκεκριμένοι στόχοι συνεχίζουν να ισχύουν καθώς το σύστημα μεταβάλλεται Σε περίπτωση που κάποιος από τους στόχους δεν επιτυγχάνεται, ή ο βαθμός ικανοποίησης του είναι μικρότερος από ένα συγκεκριμένο όριο, έχουμε εισαγάγει μια μέθοδο που μπορεί να χρησιμοποιηθεί προκειμένου να δημιουργηθεί ένα πλάνο αποκατάστασης, η εκτέλεση του οποίου μπορεί να οδηγήσει το σύστημα σε μια 'καλύτερη' κατάσταση, δηλαδή μια κατάσταση στην οποία περισσότεροι στόχοι επιτυγχάνονται, ή σε περίπτωση που έχουν εκχωρηθεί βάρη στους στόχους που υποδηλώνουν τη σημασία τους, σε μια κατάσταση όπου επιτυγχάνονται οι πιο σημαντικοί στόχοι του συστήματος.

Προκειμένου να αξιολογηθεί η απόδοση του προτεινόμενου περιβάλλοντος πλαισίου,

πραγματοποιήσαμε μια σειρά πειραμάτων με τυχαία μοντέλα ποικίλου μεγέθους και πολυπλοκότητας. Χρησιμοποιώντας αυτά τα μοντέλα αξιολογήσαμε την εφαρμογή της προτεινόμενης μεθόδου σε σχέση με το χρόνο εκτέλεσης και το μέγεθος της μνήμης που απαιτείται για μοντέλα διαφορετικών μεγεθών. Τα πειραματικά αποτελέσματα δείχνουν ότι οι χρόνοι που απαιτούνται για να ολοκληρωθεί η εξαγωγή συμπερασμάτων παραμένουν μικροί ακόμη και για μεγάλα μοντέλα, και ως εκ τούτου το προτεινόμενο περιβάλλον πλαίσιο μπορεί να χρησιμοποιηθεί κατά το χρόνο εκτέλεσης. Τέλος, αξιολογήσαμε και συγκρίναμε την ποιότητα των λύσεων που προκύπτουν από τον αλγόριθμο LS που χρησιμοποιείται για τον προσδιορισμό των πλάνων αποκατάστασης. Τα αποτελέσματα δείχνουν ότι η προτεινόμενη προσέγγιση επιτρέπει την απόκτηση μιας λύσης που είναι εντός του 90% της βέλτιστης σε πολύ μικρότερο χρόνο από αυτόν που απαιτείται από τον Weighted Partial MAX-SAT αλγόριθμο για να υπολογίσει τη βέλτιστη λύση.

9.3 Συνεισφορά της Διατριβής

Λαμβάνοντας υπόψη την λίστα με τα σημεία συνεισφοράς της διδακτορικής διατριβής που παρουσιάζεται στην Εισαγωγή (Ενότητα 1.3), περιγράφουμε εδώ τον τρόπο που κάθε ένα από τα σημεία αυτά έχει επιτευχθεί. Όπως έχει ήδη αναφερθεί, έχουμε εισάγει μία σειρά από επεκτάσεις στα μοντέλα δένδρων στόχων και τη σημασιολογία τους, έχουμε προτείνει μετασχηματισμούς για την εξαγωγή κανόνων ασαφούς λογικής από τα μοντέλα στόχων, έχουμε εισάγει ένα μεταμοντέλο για την μοντελοποίηση λογικών και χρονικών εξαρτήσεων μεταξύ των κόμβων στα μοντέλα, και έχουμε σχεδιάσει και υλοποιήσει ένα περιβάλλον πλαίσιο που υποστηρίζει διαδικασίες συμπερασμού τόσο από κάτω προς τα πάνω (δηλαδή από τα φύλλα προς τις ρίζες των δένδρων) όσο και από πάνω προς τα κάτω (δηλαδή από τις ρίζες προς τα φύλλα). Πιο συγκεκριμένα, τα σημεία συνεισφοράς της διδακτορικής διατριβής σε σχέση με το πρόβλημα που μελετάται είναι συνοπτικά τα εξής:

1. *Προτείνεται η χρήση AND/OR δένδρων στόχων με σταθμισμένες ακμές συνεισφοράς, στοιχεία υπό συνθήκη, και κόμβους που μπορούν να ισχύουν κατά έναν βαθμό αλήθειας ως ένας τρόπος για την μοντελοποίηση απαιτήσεων που θα πρέπει να ισχύουν σε ένα σύστημα λογισμικού που βρίσκεται σε λειτουργία κάτω από διάφορες συνθήκες Τα μοντέλα αυτά, τα οποία αναφέρονται ως ασαφή μοντέλα στόχων, θα μας επιτρέψουν να μοντελοποιήσουμε την ασάφεια που εγγενώς υπάρχει στον ανθρώπινο τρόπο σκέψης, συνδυάζοντας στοιχεία από τα μοντέλα στόχων και την ασαφή λογική.*

Εισάγουμε τα μοντέλα Αναμενόμενης Συμπεριφοράς (System Expected Behavior models) τα οποία επιτρέπουν τον ορισμό fuzzy και crisp στόχων οι οποίοι πρέπει να ικανοποιούνται σε ένα σύστημα λογισμικού που βρίσκεται σε λειτουργία, και μπορούν να μοντελοποιήσουν τις εξαρτήσεις που υπάρχουν μεταξύ των διαφόρων στόχων με την χρήση σταθμισμένων ακμών συνεισφοράς (Κεφάλαιο 4). Η ύπαρξη σταθμισμένων και στοιχείων υπό συνθήκη στα μοντέλα αυξάνει την εκφραστικότητα τον μοντέλων και επιτρέπει την αναπαράσταση ασαφούς και μη πλήρους γνώσης.

2. *Ορίζονται και υλοποιούνται μετασχηματισμοί μεταξύ μοντέλων και μετασχηματισμοί από μοντέλα σε κείμενο έτσι ώστε να παράγονται ασαφείς βάσεις γνώσης από ασαφή μοντέλα στόχων. Οι ασαφείς βάσεις γνώσης μπορούν να χρησιμοποιηθούν σε συνδυασμό με τα δεδομένα που συλλέγονται από το σύστημα που βρίσκεται σε λειτουργία, προκειμένου να ελεγχθεί κατά πόσο ορισμένες απαιτήσεις ικανοποιούνται από το σύστημα.*

   Ορίζουμε μετασχηματισμούς για την εξαγωγή σταθμισμένων ασαφών κανόνων και κανόνων ασαφών ελεγκτών που μπορούν να κωδικοποιήσουν τους περιορισμούς και τις εξαρτήσεις που ορίζονται στα μοντέλα (Κεφάλαιο 6). Εισάγουμε επίσης ένα περιβάλλον πλαίσιο το οποίο χρησιμοποιεί τους κανόνες αυτούς σε συνδυασμό με δεδομένα που συλλέγονται από το εν λειτουργία σύστημα έτσι ώστε να επαληθεύσουμε ότι το σύστημα ικανοποιεί τους στόχους που έχουν οριστεί στα αντίστοιχα μοντέλα (Κεφάλαιο 5).

3. *Ορίζεται και υλοποιείται ένα μεταμοντέλο το οποίο δανείζεται στοιχεία από την θεωρία των μοντέλων δένδρων στόχων και μπορεί να μοντελοποιήσει τις λογικές και χρονικές εξαρτήσεις που υπάρχουν μεταξύ εργασιών και ενεργειών. Το μεταμοντέλο διατηρεί την AND/OR-διάσπαση των κόμβων, παράλληλα όμως χρησιμοποιούνται επιπλέον δομικά στοιχεία τα οποία μπορούν να απεικονίσουν τις λογικές και χρονικές εξαρτήσεις, αυξάνοντας έτσι την εκφραστικότητα της προτεινόμενης μεθόδου μοντελοποίησης.*

   Εισάγουμε τα μοντέλα Εργασιών και Ενεργειών (Task and Actions models), στα οποία οι χρονικές εξαρτήσεις ανάμεσα στους κόμβους συμβολίζονται χρησιμοποιώντας είτε $\xrightarrow{lp}$ ακμές είτε $\xrightarrow{tp}$ ακμές (Κεφάλαιο 4). Οι πρώτες συμβολίζουν το γεγονός ότι η εργασία ή η ενέργεια αρχής πρέπει να εκτελεστεί πριν από την εργασία ή ενέργεια τέλους. Οι δεύτερες εισάγουν μία ασθενέστερη μορφή χρονικής εξάρτησης, μοντελοποιώντας το γεγονός ότι όταν τόσο ο κόμβος αρχής όσο και ο τερματικός κόμβος πρέπει να εκτελεστούν, τότε ο τερματικός κόμβος πρέπει να εκτελεστεί μετά την ολοκλήρωση εκτέλεσης της εργασίας ή ενέργειας αρχής.

4. *Σχεδιάζονται και υλοποιούνται δύο μηχανισμοί συμπερασμού για τα ασαφή μοντέλα στόχων, ένας με κατεύθυνση από τα φύλλα προς τις ρίζες και ένας δεύτερος με κατεύθυνση από τις ρίζες προς τα φύλλα. Ο πρώτος (από τα φύλλα προς τις ρίζες) επιτρέπει την επαλήθευση των στόχων και απαιτήσεων που πρέπει να ικανοποιούνται στο σύστημα που βρίσκεται σε λειτουργία και χρησιμοποιεί ασαφείς μηχανισμούς συμπερασμού και τεχνικές ασαφών ελεγκτών. Ο δεύτερος (από τις ρίζες προς τα φύλλα) επιτρέπει τον προσδιορισμό πλάνων αποκατάστασης τα οποία μπορούν να οδηγήσουν το σύστημα σε μία κατάσταση όπου οι στόχοι ικανοποιούνται και χρησιμοποιεί γενετικούς αλγορίθμους και SAT solvers*

   Δεδομένου ότι τα μοντέλα Αναμενόμενης Συμπεριφοράς και τα μοντέλα Εργασιών και Ενεργειών μπορούν να μετασχηματιστούν σε σύνολα κανόνων ασαφούς και δυαδικής λογικής, εισάγουμε δύο περιβάλλοντα πλαίσια, ένα για την επαλήθευση των απαιτήσεων λογισμικού και ένα για την δημιουργία πλάνων αποκατάστασης. Πιο συγκεκριμένα, χρησιμοποιούμαι μία βιβλιοθήκη για σταθμισμένους ασαφείς κανόνες [33] προκειμένου να εκτελέσουμε μία διαδικασία εξαγωγής συμπερασμάτων με κατεύθυνση προς τις ρίζες των μοντέλων Αναμενόμενης Συμπεριφοράς, και έτσι να επαληθεύσουμε τον βαθμό που οι απαιτήσεις και οι στόχοι ικανοποιούνται στο σύστημα. Επιπλέον, συνδυάζουμε κανόνες δυαδικής και ασαφούς λογικής και αναπτύσσεται μία υβριδική διαδικασία εξαγωγής συμπερασμάτων με κατεύθυνση προς τις ρίζες. Τέλος, τα VR μοντέλα μετασχηματίζονται σε κανόνες δυαδικής λογικής και κανόνες ασαφών ελεγκτών και αναπτύσσεται ένας γενετικός αλγόριθμος για την υλοποίηση μίας διαδικασία εξαγωγής συμπερασμάτων με κατεύθυνση προς τα φύλλα (Κεφάλαιο 7).

5. *Εκτελούνται πειράματα που σκοπό έχουν να αξιολογήσουν την απόδοση, την επεκτασιμότητα, και τη σταθερότητα των προτεινόμενων μηχανισμών συμπερασμού.*

   Προκειμένου να εκτελεστούν τα πειράματα, σχεδιάσαμε και υλοποιήσαμε μία εφαρμογή η οποία δοθέντων συγκεκριμένων παραμέτρων μπορεί να παράξει τυχαία μοντέλα. Χρησιμοποιώντας αυτήν την εφαρμογή, παράγουμε μοντέλα διαφόρων μεγεθών και πολυπλοκοτήτων και πραγματοποιούμε μία σειρά πειραμάτων προκειμένου να αξιολογήσουμε την απόδοση των προτεινόμενων μηχανισμών συμπερασμού (Κεφάλαιο 8).

9.4 Μελλοντικές Επεκτάσεις

Η παρούσα εργασία μπορεί να επεκταθεί προς διάφορες κατευθύνσεις.

- Να εξεταστεί η δυνατότητα χρήσης γενετικών ή πιθανοτικών αλγορίθμων προκειμένου να προσδιοριστούν με έναν αυτόματο τρόπο οι παράμετροι των συναρτήσεων συμμετοχής. Καθώς η επιλογή των καταλληλότερων τιμών για τις παραμέτρους αυτές είναι ένα σημαντικό στοιχείο της προτεινόμενης μεθόδου, αυτό θα μπορέσει να αυξήσει την δυνατότητα εφαρμογής της μεθόδου σε μεγάλα συστήματα λογισμικού και επιπλέον θα επιτρέψει την χρήση πιο σύνθετων συναρτήσεων συμμετοχής για τις μεθόδους ασαφοποίησης και από-ασαφοποίησης

- Να επεκταθεί ο αλγόριθμος που χρησιμοποιείται για τον προσδιορισμό των πλάνων αποκατάστασης και να εξερευνηθεί η δυνατότητα χρήσης εναλλακτικών γενετικών τελεστών. Επιπλέον, να ελεγχθεί η δυνατότητα χρήσης τεχνικών για την προσαρμογή των τιμών των παραμέτρων του γενετικού αλγορίθμου κατά τον χρόνο εκτέλεσης, έτσι ώστε να γίνεται πιο αποδοτική αναζήτηση στον χώρο των λύσεων.

- Να προστεθούν στους κόμβους ενεργειών και εργασιών επιπλέον πληροφορίες σχετικά με το πώς μπορούν να εκτελεστούν, επιτρέποντας έτσι την δημιουργία ενός αυτο-προσαρμοζόμενου συστήματος. Στην περίπτωση αυτή, το περιβάλλον πλαίσιο όχι μόνο θα προτείνει ένα πλάνο αποκατάστασης που θα μπορεί να οδηγήσει το σύστημα σε μία κατάσταση όπου περισσότεροι ή οι περισσότερο σημαντικοί στόχοι θα ικανοποιούνται, αλλά θα είναι δυνατό να εκτελεστούν οι αντίστοιχες ενέργειες ώστε το σύστημα να οδηγηθεί στην επιθυμητή κατάσταση.

Bibliography

[1] Programmable controllers - part 7: Fuzzy control programming. Technical Report IEC 61131-7:2000, International Electrotechnical Commission, August 2000.

[2] Case tools - computer-aided software engineering tools community:uml, Apr. 2012.

[3] Centroid, jan 2015.

[4] Hipaa, oct 2015.

[5] Ocl2.0, jan 2015.

[6] A. Alfonso, V. Braberman, and N. Kicillof. Visual timed event scenarios. In *Proceedings of the 26th International Conference on Software Engineering*, 2004.

[7] Raian Ali, Fabiano Dalpiaz, and Paolo Giorgini. Location-based software modeling and analysis: Tropos-based approach. In Qing Li, Stefano Spaccapietra, Eric S. K. Yu, and Antoni Olivé, editors, *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 169–182. Springer, 2008.

[8] Raian Ali, Fabiano Dalpiaz, and Paolo Giorgini. A goal-based framework for contextual requirements modeling and analysis. *Requir. Eng.*, 15(4):439–458, 2010.

[9] Raian Ali, Fabiano Dalpiaz, and Paolo Giorgini. A goal-based framework for contextual requirements modeling and analysis. *Requir. Eng.*, 15(4):439–458, 2010.

[10] Raian Ali, Fabiano Dalpiaz, and Paolo Giorgini. Reasoning with contextual requirements: Detecting inconsistency and conflicts. *Information & Software Technology*, 55(1):35–57, 2013.

[11] Daniel Amyot, Sepideh Ghanavati, Jennifer Horkoff, Gunter Mussbacher, Liam Peyton, and Eric S. K. Yu. Evaluating goal models within the goal-oriented requirement language. *Int. J. Intell. Syst.*, 25(8):841–877, 2010.

[12] M. Autili, P. Inverardi, and P. Pelliccione. A scenario based notation for specifying temporal properties. In *Proceedings of the 2006 international workshop on Scenarios and state machines: models, algorithms, and tools (SCESM '06)*, pages 21–28, New York, NY, USA, 2006. ACM.

[13] Luciano Baresi, Liliana Pasquale, and Paola Spoletini. Fuzzy goals for requirements-driven adaptation. In *RE*, pages 125–134. IEEE Computer Society, 2010.

[14] David A. Basin, Felix Klaedtke, and Samuel Müller. Policy monitoring in first-order temporal logic. In Tayssir Touili, Byron Cook, and Paul Jackson, editors, *Computer Aided Verification, 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings*, volume 6174 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2010.

[15] Nelly Bencomo and Amel Belaggoun. Supporting decision-making for self-adaptive systems: From goal models to dynamic decision networks. In Joerg Doerr and Andreas L. Opdahl, editors, *Requirements Engineering: Foundation for Software Quality - 19th International Working Conference, REFSQ 2013, Essen, Germany, April 8-11, 2013. Proceedings*, volume 7830 of *Lecture Notes in Computer Science*, pages 221–236. Springer, 2013.

[16] V. Braberman, N. Kicillof, and A. Olivero. A scenario-matching approach to the description and model checking of real-time properties. *IEEE Transactions on Software Engineering*, 31(12):1028 – 1041, dec. 2005.

[17] Paolo Bresciani, Paolo Giorgini, Fausto Giunchiglia, John Mylopoulos, and Anna Perini. Tropos: An agent-oriented software development methodology, 2003.

[18] Roberto Bruni, Matthias Hölzl, Nora Koch, Alberto Lluch Lafuente, Philip Mayer, Ugo Montanari, Andreas Schroeder, and Martin Wirsing. A service-oriented uml profile with formal support. In Luciano Baresi, Chi-Hung Chi, and Jun Suzuki, editors, *Service-Oriented Computing*, volume 5900 of *Lecture Notes in Computer Science*, pages 455–469. Springer Berlin / Heidelberg, 2009.

[19] Jeremy W. Bryans and Wei Wei. Formal analysis of bpmn models using event-b. In *Proceedings of the 15th international conference on Formal methods for industrial critical systems (FMICS'10)*, pages 33–49. Springer-Verlag, 2010.

[20] R.J.A. Buhr. Use case maps as architectural entities for complex systems. *Software Engineering, IEEE Transactions on*, 24(12):1131–1155, Dec 1998.

[21] Antoine Cailliau and Axel van Lamsweerde. Assessing requirements-related risks through probabilistic goals and obstacles. *Requir. Eng.*, 18(2):129–146, 2013.

[22] Radu Calinescu. When the requirements for adaptation and high integrity meet. In *Proceedings of the 8th Workshop on Assurances for Self-adaptive Systems*, ASAS '11, pages 1–4, 2011.

[23] Radu Calinescu, Carlo Ghezzi, Marta Z. Kwiatkowska, and Raffaela Mirandola. Self-adaptive software needs quantitative verification at runtime. *Commun. ACM*, 55(9):69–77, 2012.

[24] Javier Cámara, Rogério de Lemos, Carlo Ghezzi, and Antónia Lopes, editors. *Assurances for Self-Adaptive Systems - Principles, Models, and Techniques*, volume 7740 of *Lecture Notes in Computer Science*. Springer, 2013.

[25] George Chatzikonstantinou, Michael Athanasopoulos, and Kostas Kontogiannis. Towards a goal driven task personalization specification framework. In *IEEE Ninth World Congress on Services, SERVICES 2013, Santa Clara, CA, USA, June 28 - July 3, 2013*, pages 180–184. IEEE Computer Society, 2013.

[26] George Chatzikonstantinou, Michael Athanasopoulos, and Kostas Kontogiannis. Task specification and reasoning in dynamically altered contexts. In Matthias Jarke, John Mylopoulos, Christoph Quix, Colette Rolland, Yannis Manolopoulos, Haralambos Mouratidis, and Jennifer Horkoff, editors, *Advanced Information Systems Engineering - 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings*, volume 8484 of *Lecture Notes in Computer Science*, pages 625–639. Springer, 2014.

[27] George Chatzikonstantinou and Kostas Kontogiannis. Policy modeling and compliance verification in enterprise software systems: A survey. In *MESOCA*, pages 27–36. IEEE, 2012.

[28] George Chatzikonstantinou, Kostas Kontogiannis, and Ioanna-Maria Attarian. A goal driven framework for software project data analytics. In Camille Salinesi, Moira C. Norrie, and Oscar Pastor, editors, *Advanced Information Systems Engineering - 25th International Conference, CAiSE 2013, Valencia, Spain, June 17-21, 2013. Proceedings*, volume 7908 of *Lecture Notes in Computer Science*, pages 546–561. Springer, 2013.

[29] Liang Cheng and Yang Zhang. Model checking security policy model using both uml static and dynamic diagrams. In *Proceedings of the 4th international conference on Security of information and networks*, SIN '11, pages 159–166, New York, NY, USA, 2011. ACM.

[30] Amit K. Chopra, Fabiano Dalpiaz, Paolo Giorgini, and John Mylopoulos. Modeling and reasoning about service-oriented applications via goals and commitments. In *CAiSE*, pages 113–128, 2010.

[31] Amit K. Chopra, Fabiano Dalpiaz, Paolo Giorgini, and John Mylopoulos. Reasoning about agents and protocols via goals and commitments. In *Proceedings*

of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '10), pages 457–464, 2010.

[32] Amit K. Chopra, Fabiano Dalpiaz, Paolo Giorgini, and John Mylopoulos. Reasoning about agents and protocols via goals and commitments. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *AAMAS*, pages 457–464. IFAAMAS, 2010.

[33] Alexandros Chortaras, Giorgos B. Stamou, and Andreas Stafylopatis. Definition and adaptation of weighted fuzzy logic programs. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 17(1):85–135, 2009.

[34] Pablo Cingolani and Jesús Alcalá-Fdez. jfuzzylogic: a robust and flexible fuzzy-logic inference system language implementation. In *FUZZ-IEEE 2012, IEEE International Conference on Fuzzy Systems, Brisbane, Australia, June 10-15, 2012, Proceedings.*, pages 1–8. IEEE, 2012.

[35] Fabiano Dalpiaz, Alexander Borgida, Jennifer Horkoff, and John Mylopoulos. Runtime goal models: Keynote. In Roel Wieringa, Selmin Nurcan, Colette Rolland, and Jean-Louis Cavarero, editors, *IEEE 7th International Conference on Research Challenges in Information Science, RCIS 2013, Paris, France, May 29-31, 2013*, pages 1–11. IEEE, 2013.

[36] Fabiano Dalpiaz, Paolo Giorgini, and John Mylopoulos. Adaptive socio-technical systems: a requirements-based approach. *Requir. Eng.*, 18(1):1–24, 2013.

[37] Nicodemos Damianou, Naranker Dulay, Emil Lupu, and Morris Sloman. The ponder policy specification language. In *Proceedings of the International Workshop on Policies for Distributed Systems and Networks (POLICY '01)*, pages 18–38. Springer-Verlag, 2001.

[38] Remco M. Dijkman, Marlon Dumas, and Chun Ouyang. Semantics and analysis of business process models in bpmn. *Information and Software Technology*, 50(12):1281 – 1294, 2008.

[39] Golnaz Elahi and Eric S. K. Yu. Comparing alternatives for analyzing requirements trade-offs - in the absence of numerical data. *Information & Software Technology*, 54(6):517–530, 2012.

[40] Herbert B Enderton. *A mathematical introduction to logic*. Academic press, 2001.

[41] T. Erl. *SOA Design Patterns*. Pearson Education, Inc, Boston, MA, 2009.

[42] Yliès Falcone, Jean-Claude Fernandez, and Laurent Mounier. Runtime verification of safety-progress properties. In Saddek Bensalem and Doron Peled, editors, *RV*, volume 5779 of *Lecture Notes in Computer Science*, pages 40–59. Springer, 2009.

[43] Yliès Falcone, Mohamad Jaber, Thanh-Hung Nguyen, Marius Bozga, and Saddek Bensalem. Runtime verification of component-based systems. In Gilles Barthe, Alberto Pardo, and Gerardo Schneider, editors, *Software Engineering and Formal Methods - 9th International Conference, SEFM 2011, Montevideo, Uruguay, November 14-18, 2011. Proceedings*, volume 7041 of *Lecture Notes in Computer Science*, pages 204–220. Springer, 2011.

[44] Antonio Filieri, Carlo Ghezzi, and Giordano Tamburrelli. Run-time efficient probabilistic model checking. In Richard N. Taylor, Harald Gall, and Nenad Medvidovic, editors, *ICSE*, pages 341–350. ACM, 2011.

[45] Mitsuo Gen and Runwei Cheng. *Genetic algorithms and engineering optimization*, volume 7. John Wiley & Sons, 2000.

[46] Carlo Ghezzi. Engineering evolving and self-adaptive systems: An overview. In Manfred Broy, Christian Leuxner, and Tony Hoare, editors, *Software and Systems Safety - Specification and Verification*, volume 30 of *NATO Science for Peace and Security Series - D: Information and Communication Security*, pages 88–102. IOS Press, 2011.

[47] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone. Modeling security requirements through ownership, permission and delegation. In *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, pages 167–176, Aug 2005.

[48] Paolo Giorgini, Fabio Massacci, John Mylopoulos, and Nicola Zannone. Requirements engineering for trust management: Model, methodology, and reasoning. In *of the 3rd International i* Workshop - istar08*, page 2006, 2006.

[49] Paolo Giorgini, Fabio Massacci, John Mylopoulos, and Nicola Zannone. Requirements engineering for trust management: model, methodology, and reasoning. *Int. J. Inf. Sec.*, 5(4):257–274, 2006.

[50] Paolo Giorgini, John Mylopoulos, Eleonora Nicchiarelli, and Roberto Sebastiani. Reasoning with goal models. In Stefano Spaccapietra, Salvatore T. March, and Yahiko Kambayashi, editors, *ER*, volume 2503 of *Lecture Notes in Computer Science*, pages 167–181. Springer, 2002.

[51] Yann Glouche, Thomas Genet, Olivier Heen, and Olivier Courtay. A security protocol animator tool for avispa. In *In ARTIST-2 workshop*, 2006.

[52] David Harel and P. S. Thiagarajan. Message sequence charts. In *In UML for Real: Design of Embedded Real-Time Systems*, pages 77–105. Kluwer Academic Publishers, 2003.

[53] Abdollah Homaifar and Ed McCormick. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE T. Fuzzy Systems*, 3(2):129–139, 1995.

[54] Jennifer Horkoff, Daniele Barone, Lei Jiang, Eric S. K. Yu, Daniel Amyot, Alexander Borgida, and John Mylopoulos. Strategic business modeling: representation and reasoning. *Software and System Modeling*, 13(3):1015–1041, 2014.

[55] Jennifer Horkoff and Eric S. K. Yu. Analyzing goal models: different approaches and how to choose among them. In William C. Chu, W. Eric Wong, Mathew J. Palakal, and Chih-Cheng Hung, editors, *Proceedings of the 2011 ACM Symposium on Applied Computing (SAC), TaiChung, Taiwan, March 21 - 24, 2011*, pages 675–682. ACM, 2011.

[56] Jennifer Horkoff and Eric S. K. Yu. Comparison and evaluation of goal-oriented satisfaction analysis techniques. *Requir. Eng.*, 18(3):199–222, 2013.

[57] Silvia Ingolfo, Alberto Siena, and John Mylopoulos. Establishing regulatory compliance for software requirements. In Manfred A. Jeusfeld, Lois M. L. Delcambre, and Tok Wang Ling, editors, *Conceptual Modeling - ER 2011, 30th International Conference, ER 2011, Brussels, Belgium, October 31 - November 3, 2011. Proceedings*, volume 6998 of *Lecture Notes in Computer Science*, pages 47–61. Springer, 2011.

[58] G. Booch J. Rumbaugh, I. Jacobson. *The Unified Modeling Language Reference Manual*. Addison-Wesley, Reading, MA, 1999.

[59] L. Kagal, T. Finin, and Anupam Joshi. A policy language for a pervasive computing environment. In *IEEE 4th International Workshop on Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003.*, pages 63 – 74, june 2003.

[60] Theodoros Kalamatianos, Kostas Kontogiannis, and Peter Matthews. Domain independent event analysis for log data reduction. In Xiaoying Bai, Fevzi Belli, Elisa Bertino, Carl K. Chang, Atilla Elçi, Cristina Cerschi Seceleanu, Haihua Xie, and Mohammad Zulkernine, editors, *COMPSAC*, pages 225–232. IEEE Computer Society, 2012.

[61] L. Kof, R. Gacitua, M. Rouncefield, and P. Sawyer. Ontology and model alignment as a means for requirements validation. In *2010 IEEE Fourth International Conference on Semantic Computing (ICSC)*, pages 46 –51, Sept. 2010.

[62] W. Kong, K. Ogata, T. Seino, and K. Futatsugi. A lightweight integration of theorem proving and model checking for system verification. In *Software Engineering Conference, 2005. APSEC '05. 12th Asia-Pacific*, dec. 2005.

[63] Ingolf H. Krüger, Michael Meisinger, and Massimiliano Menarini. Interaction-based runtime verification for systems of systems integration. *J. Log. Comput.*, 20(3):725–742, 2010.

[64] Alexei Lapouchnian and John Mylopoulos. Capturing contextual variability in *i** models. In Jaelson Brelaz de Castro, Xavier Franch, John Mylopoulos, and Eric S. K. Yu, editors, *iStar*, volume 766 of *CEUR Workshop Proceedings*, pages 96–101. CEUR-WS.org, 2011.

[65] Martin Leucker and Christian Schallhart. A brief account of runtime verification. *J. Log. Algebr. Program.*, 78(5):293–303, 2009.

[66] Sotirios Liaskos, Saeideh Hamidi, and Rina Jalman. Qualitative vs. quantitative contribution labels in goal models: Setting an experimental agenda. In Jaelson Castro, Jennifer Horkoff, Neil A. M. Maiden, and Eric S. K. Yu, editors, *Proceedings of the 6th International* i* *Workshop 2013, Valencia, Spain, June 17-18, 2013*, volume 978 of *CEUR Workshop Proceedings*, pages 37–42. CEUR-WS.org, 2013.

[67] Sotirios Liaskos, Rina Jalman, and Jorge Aranda. On eliciting contribution measures in goal models. In Mats Per Erik Heimdahl and Pete Sawyer, editors, *2012 20th IEEE International Requirements Engineering Conference (RE), Chicago, IL, USA, September 24-28, 2012*, pages 221–230. IEEE Computer Society, 2012.

[68] Sotirios Liaskos, Shakil M. Khan, Marin Litoiu, Marina Daoud Jungblut, Vyacheslav Rogozhkin, and John Mylopoulos. Behavioral adaptation of information systems through goal models. *Inf. Syst.*, 37(8):767–783, 2012.

[69] Sotirios Liaskos and John Mylopoulos. On temporally annotating goal models. In Jaelson Brelaz de Castro, Xavier Franch, John Mylopoulos, and Eric S. K. Yu, editors, *Proceedings of the* 4^{th} *International* i* *Workshop, Hammamet, Tunisia, June 07-08, 2010*, volume 586 of *CEUR Workshop Proceedings*, pages 62–66. CEUR-WS.org, 2010.

[70] K. Lichtner, P. Alencar, and D. Cowan. A framework for software architecture verification. In *Australian Software Engineering Conference, 2000. Proceedings. 2000*, pages 149 –157, 2000.

[71] Beate List and Birgit Korherr. A uml 2 profile for business process modelling. In *Proceedings of the 1st International Workshop on Best Practices of UML at the 24th International Conference on Conceptual Modeling (ER 2005*, pages 24–28. Springer Verlag, 2005.

[72] Y. Liu, S. Muller, and K. Xu. A static compliance-checking framework for business process models. *IBM Systems Journal*, 46(2):335 –361, 2007.

[73] Torsten Lodderstedt, David Basin, and Jürgen Doser. Secureuml: A uml-based modeling language for model-driven security. pages 426–441. Springer, 2002.

[74] J.C. Maxwell and A.I. Antón. Developing production rule models to aid in acquiring requirements from legal texts. In *17th IEEE International Requirements Engineering Conference (RE '09)*, pages 101 –110, 31 2009-sept. 4 2009.

[75] Mirko Morandini, Loris Penserini, and Anna Perini. Towards goal-oriented development of self-adaptive systems. In *2008 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2008, Leipzig, Germany, May 12-13, 2008*, pages 9–16, 2008.

[76] John Mylopoulos, Lawrence Chung, and Eric S. K. Yu. From object-oriented to goal-oriented requirements analysis. *Commun. ACM*, 42(1):31–37, 1999.

[77] Md. Rashed Iqbal Nekvi and Nazim H. Madhavji. Impediments to regulatory compliance of requirements in contractual systems engineering projects: A case study. *ACM Trans. Management Inf. Syst.*, 5(3):15:1–15:35, 2015.

[78] Nardine Osman, David Robertson, and Christopher Walton. Run-time model checking of interaction and deontic models for multi-agent systems. In Hideyuki Nakashima, Michael P. Wellman, Gerhard Weiss, and Peter Stone, editors, *AAMAS*, pages 238–240. ACM, 2006.

[79] V. Papadopoulou and A. Gregoriades. Nonfunctional requirements validation - a game theoretic approach. In *Management and Service Science, 2009. MASS '09. International Conference on*, pages 1 –4, sept. 2009.

[80] Kevin M. Passino and Stephen Yurkovich. *Fuzzy Control*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1997.

[81] Davide Prandi, Paola Quaglia, and Nicola Zannone. Formal analysis of bpmn via a translation into cows. In *Proceedings of the 10th international conference on Coordination models and languages (COORDINATION'08)*, pages 249–263. Springer-Verlag, 2008.

[82] F. Rabbi, Hao Wang, and W. MacCaull. Yawl2dve: An automated translator for workflow verification. In *Secure Software Integration and Reliability Improvement (SSIRI), 2010 Fourth International Conference on*, june 2010.

[83] Hridesh Rajan, Jia Tao, Steve Shaner, and Gary Leavens. Tisa: A language design and modular verification technique for temporal policies in web services. In *Programming Languages and Systems*, pages 333–347. Springer Berlin / Heidelberg, 2009.

[84] Timothy J. Ross. *Fuzzy Logic with Engineering Applications*. John Wiley & Sons, August 2004.

[85] Kamran Sartipi and Kostas Kontogiannis. On modeling software architecture recovery as graph matching. In *ICSM*, pages 224–234, 2003.

[86] Peter Sawyer, Nelly Bencomo, Jon Whittle, Emmanuel Letier, and Anthony Finkelstein. Requirements-aware systems: A research agenda for RE for self-adaptive systems. In *RE 2010, 18th IEEE International Requirements Engineering Conference, Sydney, New South Wales, Australia, September 27 - October 1, 2010*, pages 95–103, 2010.

[87] Holger Schmidt. Service level agreements based on business process modeling, 2000.

[88] Tobias Schuele and Klaus Schneider. Global vs. local model checking: A comparison of verification techniques for infinite state systems. In *Proceedings of the Software Engineering and Formal Methods, Second International Conference (SEFM 04)*, pages 67–76, Washington, DC, USA, 2004. IEEE Computer Society.

[89] Bikram Sengupta and R. Cleaveland. Triggered message sequence charts. *IEEE Transactions on Software Engineering*, 32(8):587 –607, aug. 2006.

[90] Amir Molzam Sharifloo and Paola Spoletini. LOVER: light-weight formal verification of adaptive systems at run time. In Corina S. Pasareanu and Gwen Salaün, editors, *Formal Aspects of Component Software, 9th International Symposium, FACS 2012, Mountain View, CA, USA, September 12-14, 2012. Revised Selected Papers*, volume 7684 of *Lecture Notes in Computer Science*, pages 170–187. Springer, 2012.

[91] Morris Sloman and Emil Lupu. Security and management policy specification. *IEEE Network*, 16:10–19, 2002.

[92] M.H. Smith and K. Havelund. Requirements capture with rcat. In *International Requirements Engineering, 2008. RE '08. 16th IEEE*, pages 183 –192, sept. 2008.

[93] Vítor E. Silva Souza, Alexei Lapouchnian, Konstantinos Angelopoulos, and John Mylopoulos. Requirements-driven software evolution. *Computer Science - R&D*, 28(4):311–329, 2013.

[94] Vítor Estêvão Silva Souza, Alexei Lapouchnian, William N. Robinson, and John Mylopoulos. Awareness requirements for adaptive systems. In *2011 ICSE Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS 2011, Waikiki, Honolulu , HI, USA, May 23-24, 2011*, pages 60–69, 2011.

[95] Daisuke Tanabe, Kohei Uno, Kinji Akemine, Takashi Yoshikawa, Haruhiko Kaiya, and Motoshi Saeki. Supporting requirements change management in goal oriented analysis. In *RE*, pages 3–12. IEEE Computer Society, 2008.

[96] Robert Endre Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.

[97] R. Thion and D. Le Metayer. Flavor: A formal language for a posteriori verification of legal rules. In *Policies for Distributed Systems and Networks (POLICY), 2011 IEEE International Symposium on*, pages 1–8, June 2011.

[98] W.M.P. van der Aalst and A. H. M. Ter Hofstede. Yawl: Yet another workflow language. *Information Systems*, 30:245–275, 2003.

[99] Axel van Lamsweerde. Goal-oriented requirements enginering: A roundtrip from research to practice. In *12th IEEE International Conference on Requirements Engineering (RE 2004), 6-10 September 2004, Kyoto, Japan*, pages 4–7. IEEE Computer Society, 2004.

[100] Axel van Lamsweerde. Reasoning about alternative requirements options. In Alexander Borgida, Vinay K. Chaudhri, Paolo Giorgini, and Eric S. K. Yu, editors, *Conceptual Modeling: Foundations and Applications - Essays in Honor of John Mylopoulos*, volume 5600 of *Lecture Notes in Computer Science*, pages 380–397. Springer, 2009.

[101] Miroslav N. Velev. Efficient translation of boolean formulas to cnf in formal verification of microprocessors. In *Proceedings of the 2004 Asia and South Pacific Design Automation Conference*, ASP-DAC '04, pages 310–315, Piscataway, NJ, USA, 2004. IEEE Press.

[102] Jianhua Wang, Jingbo Shao, Yingmei Li, and Jinfeng Ding. Survey on formal verification methods for digital ic. In *Proceedings of the 2009 Fourth International Conference on Internet Computing for Science and Engineering (ICICSE '09)*, pages 164–168, Washington, DC, USA, 2009. IEEE Computer Society.

[103] Yiqiao Wang, Sheila A. McIlraith, Yijun Yu, and John Mylopoulos. An automated approach to monitoring and diagnosing requirements. In R. E. Kurt Stirewalt, Alexander Egyed, and Bernd Fischer, editors, *22nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2007), November 5-9, 2007, Atlanta, Georgia, USA*, pages 293–302. ACM, 2007.

[104] Yiqiao Wang, Sheila A. McIlraith, Yijun Yu, and John Mylopoulos. Monitoring and diagnosing software requirements. *Autom. Softw. Eng.*, 16(1):3–35, 2009.

[105] Tuoye Xu, Wenting Ma, Lin Liu, and D. Karagiannis. Synthesizing enterprise strategic model and business processes in active-i*. In *14th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2010*, pages 345 –354, oct. 2010.

[106] E. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*, pages 226–235, 1997.

[107] Jian Yu, Tan Phan Manh, Jun Han, and Yan Jin. Pattern based property specification and verification for service composition. Technical Report SUT.CeCSES-TR010, Swinburne University of Technology, 2006.

[108] Jian Yu, Tan Phan Manh, Jun Han, Yan Jin, Yanbo Han, and Jianwu Wang. Pattern based property specification and verification for service composition. In *In: Proceedings of 7th International Conference on Web Information Systems Engineering (WISE)*, pages 156–168, 2006.

[109] Lotfi Zadeh. Discussion: Probability theory and fuzzy logic are complementary rather than competitive. *Technometrics*, 37(3):271–276, 1995.

[110] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[111] Safaa Zaman and Fakhri Karray. Features selection for intrusion detection systems based on support vector machines. In *Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE*, pages 1–8. IEEE, 2009.

[112] Pamela Zave and Michael Jackson. Four dark corners of requirements engineering. *ACM Trans. Softw. Eng. Methodol.*, 6(1):1–30, 1997.

[113] Yanjun Zuo and S. Lande. A logical framework of proof-carrying survivability. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, pages 472–481, 2011.

Βιογραφικό Σημείωμα

Προσωπικές Πληροφορίες

Ονοματεπώνυμο: Γεώργιος Χατζηκωνσταντίνου
Ημερομηνία Γέννησης: 19/02/1985
E-mail: gchatzi85@gmail.com

Εκπαίδευση

10/2009 – 10/2015	**Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ ΕΜΠ** Εκπόνηση διδακτορικής διατριβής Επιβλέπων: Αν. Καθηγητής Κωνσταντίνος Κοντογιάννης Θέμα: Τεχνικές Ανάλυσης και Παρακολούθησης Ορθής Λειτουργίας Πληροφοριακών Συστημάτων Πολλαπλών Επιπέδων
09/2002 – 02/2008	**Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ ΕΜΠ** Βαθμός Διπλώματος: 8.14/10 Θέμα Διπλωματικής: Αποδοτική και Αξιόπιστη Επικοινωνία με Κατανεμημένα Συστήματα Αποθήκευσης μέσω Δικτυακής Συσκευής Αποθήκευσης
2002	**1ο Ενιαίο Λύκειο Καβάλας** Βαθμός Απολυτηρίου: 19.5/20

Εργασιακή Εμπειρία

12/2013 – 03/2016	**Openbet Hellas – Τμήμα Business Information** Project RMS (πλατφόρμα διαχείρισης ανθρωπίνων πόρων) Ανάπτυξη reports στην πλατφορμα Jaspersoft Θέση: Software Engineer (Java, REST, ExtJS, AngularJS)
11/2010 – 07/2011	**SingularLogic** Project Orbi (ERP πλατφόρμα για ελεύθερους επαγγελματίες) Θέση: Application Developer (PHP, Javascript/ExtJS, Databases)
02/2008 – 08/2010	**EFG Eurobank Ergasias** Project EDDIE (εφαρμογή διαχείρισης στεγαστικών δανείων) Θέση: Application Developer (Java, SOAP/AXIS, Databases)

Υποτροφίες

02/2011 – 02/2014	**Ηράκλειτος ΙΙ –** Ενίσχυση του ανθρώπινου ερευνητικού δυναμικού μέσω της υλοποίησης διδακτορικής έρευνας

Ξένες Γλώσσες

Αγγλικά	Cambridge Proficiency of English (C2) Michigan Certificate of Proficiency in English (C2)
Γερμανικά	Goethe Zertifikat Deutsch (B1)

Δημοσιεύσεις

1. George Chatzikonstantinou, Kostas Kontogiannis: Policy modeling and compliance verification in enterprise software systems: A survey. MESOCA 2012: 27-36

2. George Chatzikonstantinou, Kostas Kontogiannis: Model Contextual Variability for Agents Using Goals and Commitments. IStar 2013: 103-108

3. George Chatzikonstantinou, Kostas Kontogiannis, Ioanna-Maria Attarian: A Goal Driven Framework for Software Project Data Analytics. CAiSE 2013: 546-561

4. George Chatzikonstantinou, Michael Athanasopoulos, Kostas Kontogiannis: Towards a Goal Driven Task Personalization Specification Framework. SERVICES 2013: 180-184

5. George Chatzikonstantinou, Michael Athanasopoulos , Kostas Kontogiannis: Task Specification and Reasoning in Dynamically Altered Contexts. CAiSE 2014: 546-561

6. George Chatzikonstantinou, Kostas Kontogiannis: Run-Time Requirements Verification for Reconfigurable Systems, Information and Software Technology, Volume 75, July 2016, Pages 105-121