Ατομική Διπλωματική Εργασία

ΕΥΡΕΤΙΚΕΣ ΜΕΘΟΔΟΙ ΠΡΟΤΑΣΙΑΚΩΝ ΕΠΙΛΥΤΩΝ ΓΙΑ ΠΡΟΒΛΗΜΑΤΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΔΡΑΣΗΣ

Ελένη Προξένου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2012

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Ευρετικές μέθοδοι Προτασιακών Επιλυτών για Προβλήματα Προγραμματισμού Δράσης

Ελένη Προξένου

Επιβλέπων Καθηγητής
Δρ. Γιάννης Δημόπουλος

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2012

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής μου εργασίας Δρ. Γιάννη Δημόπουλο, για τις πολύτιμες συμβουλές, την υπομονή και την καθοδήγησή του.

Θα ήθελα επίσης να ευχαριστήσω όλους τους καθηγητές που μου μεταλαμπάδευσαν τις γνώσεις και εμπειρίες τους κατά τη διάρκεια των σπουδών μου, συμβάλλοντας στην επιτυχή ολοκλήρωση αυτής της εργασίας.

Ακόμη, θα ήθελα να εκφράσω την ευγνωμοσύνη μου στην οικογένειά μου και ιδιαίτερα στους γονείς μου Νίκο και Άννα, για την αμέριστη συμπαράσταση και βοήθειά τους καθ' όλη τη διάρκεια των σπουδών και της ζωής μου.

Περίληψη

Η παρούσα διπλωματική εργασία στοχεύει στην ανάπτυξη και υλοποίηση τεχνικών που μπορούν να χρησιμοποιηθούν σε προτασιακούς επιλυτές (SAT solvers), για την αποδοτικότερη επίλυση των προβλημάτων Προγραμματισμού Δράσης (Planning).

Πιο συγκεκριμένα, σε αυτή την εργασία επικεντρωθήκαμε στη μελέτη τεχνικών που μπορούν να χρησιμοποιηθούν κατά τη διαδικασία της επιλογής μεταβλητής απόφασης και κατά τη διαδικασία της επιλογής τιμής για τη μεταβλητή απόφασης, οι οποίες αξιοποιούν τις ιδιότητες που έχουν τα προβλήματα Προγραμματισμού Δράσης, για να οδηγήσουν τους προτασιακούς επιλυτές σε καλύτερες επιλογές μεταβλητών απόφασης και τελικά σε γρηγορότερες και αποδοτικότερες λύσεις.

Οι τεχνικές που εξετάσαμε υλοποιήθηκαν και ενσωματώθηκαν στο ευρετικό απόφασης που χρησιμοποιεί ο προτασιακός επιλυτής PrecoSAT 570 [19] και αξιολογήθηκαν πειραματικά. Για την ολοκλήρωση της εργασίας χρησιμοποιήθηκε επίσης το σύστημα SATPLAN, ένα ολοκληρωμένο σύστημα επίλυσης προβλημάτων Προγραμματισμού Δράσης που στηρίζεται στην ιδέα μετασχηματισμού του προβλήματος Προγραμματισμού Δράσης σε πρόβλημα Προτασιακής Ικανοποιησιμότητας [20].

Τα πειραματικά αποτελέσματα γενικότερα έδειξαν ότι η εφαρμογή των τεχνικών που υλοποιήθηκαν αυξάνει το χρόνο επίλυσης των προβλημάτων. Φαίνεται ότι κάποιες τεχνικές μειώνουν τον αριθμό αποφάσεων και αυξάνουν τον αριθμό μεταδόσεων ανά απόφαση. Παρόλα αυτά, η εστίαση μόνο στις τεχνικές που φαίνεται ότι αποδίδουν και η τροποποίηση της υλοποίησης ώστε να μη χρειάζεται η χρονοβόρα προεργασία, θα μπορούσαν να βελτιώσουν δραματικά το χρόνο επίλυσης.

Συμπερασματικά, φαίνεται ότι καμία από τις τεχνικές που εξετάσαμε δε βελτιώνει το χρόνο επίλυσης των προβλημάτων. Το γεγονός όμως ότι κάποιες από τις τεχνικές οδηγούν σε μείωση των αποφάσεων για την επίλυση πολλών προβλημάτων φανερώνει ότι η εφαρμογή εξειδικευμένων τεχνικών, είτε για τα προβλήματα Προγραμματισμού Δράσης είτε για άλλα είδη προβλημάτων, μπορεί να συμβάλει στην ανάπτυξη αποδοτικότερων επιλυτών, εξειδικευμένων για συγκεκριμένα είδη προβλημάτων.

Περιεχόμενα

Κεφάλαιο **1** **Εισαγωγή**…… **1**

1.1 Ιστορικό υπόβαθρο 1

1.2 Στόχος της διπλωματικής εργασίας 3

1.3 Περιεχόμενο της διπλωματικής εργασίας 3

Κεφάλαιο **2** **Προγραμματισμός Δράσης**……. **5**

2.1 Εισαγωγή στον Προγραμματισμό Δράσης 5

2.2 Η γλώσσα STRIPS 6

2.2.1 Αναπαράσταση προβλημάτων με τη γλώσσα STRIPS 6

2.2.2 Παράδειγμα αναπαράστασης προβλήματος με τη γλώσσα STRIPS 7

2.3 Το σύστημα SATPLAN 10

2.4 Κωδικοποίηση SAT-MAX-PLAN 11

Κεφάλαιο **3** **Ικανοποιησιμότητα και Προτασιακή Λογική**……. **13**

3.1 Το Πρόβλημα της Ικανοποιησιμότητας 13

3.2 Προτασιακή Λογική 15

3.2.1 Εισαγωγή 15

3.2.2 Σύνταξη και σημασιολογία 16

3.2.3 Προτασιακή Επίλυση 19

Κεφάλαιο **4** **Προτασιακοί επιλυτές**…….……. **20**

4.1 Εισαγωγή 20

4.2 Τροποποιήσεις της αρχικής φόρμουλας 21

4.3 Δέντρα Αναζήτησης 22

4.4 Αλγόριθμος DPLL 23

4.4.1 Εισαγωγή 23

4.4.2 Περιγραφή του αλγόριθμου DPLL 24

4.4.3 Παράδειγμα εκτέλεσης του αλγόριθμου DPLL 26

4.5 Αλγόριθμος CDCL 29
4.5.1 Εισαγωγή 29
4.5.2 Περιγραφή του αλγόριθμου CDCL 30
4.6 Τεχνικές των μοντέρνων προτασιακών επιλυτών 34
4.6.1 Εισαγωγή 34
4.6.2 Προεπεξεργαστής 35
4.6.3 Μοναδιαία Μετάδοση 35
4.6.4 Ευρετικό Απόφασης 36
4.6.5 Ανάλυση σύγκρουσης 37
4.6.6 Διαγραφή εκμαθημένων προτάσεων 38
4.6.7 Επανεκκίνηση 39
4.6.8 Αλληλεπιδράσεις μεταξύ των κομματιών ενός προτασιακού επιλυτή 40

Κεφάλαιο **5** **Ο προτασιακός επιλυτής PrecoSAT**…….. **42**
5.1 Εισαγωγή 42
5.2 Μορφοποίηση DIMACS 43
5.3 Ευρετικό απόφασης του PrecoSAT 44
5.4 Ευρετικό για την επιλογή μεταβλητής απόφασης 44
5.4.1. Δομές δεδομένων 45
5.4.2. Αλγόριθμοι θερμότητας 47
5.4.2.1. Αλγόριθμος αρχικοποίησης θερμότητας 48
5.4.2.2. Αλγόριθμος υπολογισμού θερμότητας 48
5.5 Ευρετικό για την επιλογή τιμής της μεταβλητής απόφασης 50
5.5.1 Δομές δεδομένων 50
5.5.2 Ευρετικό Jeroslow-Wang 51
5.5.3 Ευρετικό Rsat 52
5.6 Σύνοψη 52

Κεφάλαιο **6** **Υλοποίηση**…….. **54**
6.1 Εισαγωγή 54
6.2 Βασική ιδέα της υλοποίησης 57
6.3 Εντολές εκτέλεσης 59

6.4 Περιγραφή της λειτουργίας των παραμέτρων 59
6.4.1 Παράμετροι για τον καθορισμό αρχείων εισόδου 60
6.4.2 Παράμετροι για εκτύπωση στατιστικών στοιχείων σε αρχεία εξόδου 60
6.4.3 Παράμετροι χρονικών επιπέδων 62
6.4.4 Παράμετροι είδους μεταβλητής 63
6.4.5 Παράμετροι «θορύβου» 64
6.4.6 Παράμετρος αύξησης θερμότητας 64
6.4.7 Παράμετροι κατάταξης μεταβλητών 65
6.4.8 Παράμετροι επιλογής τιμής 67
6.4.9 Συνδυασμός των παραμέτρων 68
6.5 Παραδείγματα εκτέλεσης 70

Κεφάλαιο **7** **Πειραματική Ανάλυση**..... **73**
7.1 Εισαγωγή 73
7.2 Περιγραφή των συστημάτων 74
7.3 Πίνακες αποτελεσμάτων 75

Κεφάλαιο **8** **Συμπεράσματα** **88**
8.1 Σύνοψη και τελικά συμπεράσματα 88
8.2 Επεκτάσεις και μελλοντική εργασία 92

Β ι β λ ι ο γ ρ α φ ί α **9 6**

Κεφάλαιο 1

Εισαγωγή

1.1 Ιστορικό υπόβαθρο 1

1.2 Στόχος της διπλωματικής εργασίας 3

1.3 Περιεχόμενο της διπλωματικής εργασίας 3

1.1 Ιστορικό υπόβαθρο

Ο Προγραμματισμός Δράσης ορίζεται ως η εύρεση μιας ακολουθίας δράσεων οι οποίες, όταν εκτελεστούν, οδηγούν στην επίτευξη κάποιου στόχου [17]. Το πρόβλημα του Προγραμματισμού Δράσης είναι ένα από τα σημαντικότερα πεδία της Τεχνητής Νοημοσύνης και έχει προσελκύσει το ενδιαφέρον πολλών ερευνητών, διότι η μελέτη του συμβάλλει στην κατανόηση της ευφυούς συμπεριφοράς και στην ανάπτυξη αυτόνομων ευφυών συστημάτων.

Γενικά, η πολυπλοκότητα του κλασικού προβλήματος Προγραμματισμού Δράσης (Classical Planning) είναι PSPACE [5]. Παρόλα αυτά, συνήθως τα προβλήματα του πραγματικού κόσμου μπορούν να λυθούν σε σχετικά σύντομο χρονικό διάστημα. Το γεγονός αυτό έδωσε κίνητρα στους επιστήμονες για την ανάπτυξη και μελέτη πλήθους αλγορίθμων, σειριακών και παράλληλων, και τη δημιουργία διάφορων εργαλείων που στοχεύουν στη γρήγορη και αποδοτική επίλυση των προβλημάτων Προγραμματισμού Δράσης.

Μια πρωτότυπη ιδέα επίλυσης προβλημάτων Προγραμματισμού Δράσης προτάθηκε το 1992 από τους Henry Kautz και Bart Selman, οι οποίοι πρότειναν τη μέθοδο μετασχηματισμού του προβλήματος Προγραμματισμού Δράσης σε πρόβλημα Προτασιακής Ικανοποιησιμότητας [10]. Μετά από 4 χρόνια, οι Kautz και Selman

πρότειναν το σύστημα SATPLAN, ένα ολοκληρωμένο σύστημα επίλυσης προβλημάτων Προγραμματισμού Δράσης που στηρίζεται στην προτασιακή λογική και θεωρείται ως ένα από τα πιο ανταγωνιστικά συστήματα επίλυσης προβλημάτων Προγραμματισμού Δράσης [11]. Το 2004, το σύστημα SATPLAN πήρε την πρώτη θέση στο συνέδριο ICAPS 2004 (International Conference on Automated Planning and Scheduling) [9] και το 2006 δημιουργήθηκε μια πιο αποδοτική έκδοση του SATPLAN [20], η οποία χρησιμοποιείται στην παρούσα διπλωματική εργασία.

Για την επίλυση των προβλημάτων Προγραμματισμού Δράσης, το σύστημα SATPLAN χρησιμοποιεί γενικούς προτασιακούς επιλυτές (SAT solvers) όπως MiniSat, PrecoSAT, Siege. Οι προτασιακοί επιλυτές (SAT solvers) είναι συστήματα που εφαρμόζουν διάφορες τεχνικές, για να λύσουν το πρόβλημα της Προτασιακής Ικανοποιησιμότητας, δηλαδή της εύρεσης κάποιας ανάθεσης τιμών στις μεταβλητές μιας φόρμουλας (CNF theory/ formula), ώστε η φόρμουλα να ικανοποιείται. Στην παρούσα διπλωματική χρησιμοποιείται η νεότερη έκδοση του προτασιακού επιλυτή PrecoSAT (έκδοση 570), ο οποίος πήρε το χρυσό μετάλλιο στο διαγωνισμό SAT 2009 (έκδοση 236) και θεωρείται ένας από τους καλύτερους αντιπροσώπους των μοντέρνων προτασιακών επιλυτών [3].

Η αποδοτικότητα της μεθόδου επίλυσης των προβλημάτων Προγραμματισμού Δράσης χρησιμοποιώντας το σύστημα SATPLAN εξαρτάται σε μεγάλο βαθμό από την αποδοτικότητα του προτασιακού επιλυτή. Για το λόγο αυτό, είναι σημαντικό να αναπτυχθούν εξειδικευμένες τεχνικές, για την αποδοτικότερη λειτουργία των προτασιακών επιλυτών.

Τα τελευταία 10 χρόνια υπήρξε αξιοσημείωτη πρόοδος στις τεχνικές που χρησιμοποιούνται στους προτασιακούς επιλυτές, η οποία συνέβαλε στην αποδοτικότερη επίλυση πολλών προβλημάτων, συμπεριλαμβανομένων των προβλημάτων Προγραμματισμού Δράσης. Εκτός από τις τεχνικές που δεν εστιάζονται σε συγκεκριμένο είδος προβλημάτων και ενσωματώνονται σε γενικούς προτασιακούς επιλυτές, έχουν αναπτυχθεί εξειδικευμένες τεχνικές που επικεντρώνονται σε συγκεκριμένα είδη προβλημάτων, όπως είναι τα προβλήματα Προγραμματισμού Δράσης, και προσπαθούν να χρησιμοποιήσουν τα χαρακτηριστικά και τις ιδιότητες

αυτών των προβλημάτων, για την αποδοτικότερη και ταχύτερη επίλυσή τους [13, 14, 15].

1.2 Στόχος της διπλωματικής εργασίας

Στόχος της παρούσας διπλωματικής εργασίας είναι η μελέτη των ευρετικών μεθόδων που χρησιμοποιούνται στους προτασιακούς επιλυτές και η ανάπτυξη και μελέτη εξειδικευμένων καινοτόμων τεχνικών για την αποδοτικότερη επίλυση των προβλημάτων Προγραμματισμού Δράσης. Οι τεχνικές που αναπτύχθηκαν για σκοπούς της διπλωματικής, ενσωματώθηκαν στον κώδικα του επιλυτή PrecoSAT, και οι επιδράσεις τους στην επίλυση των προβλημάτων Προγραμματισμού Δράσης μελετήθηκαν πειραματικά.

1.3 Περιεχόμενο της διπλωματικής εργασίας

Η εκπόνηση της παρούσας διπλωματικής εργασίας διήρκεσε περίπου εννέα μήνες (Σεπτέμβριο 2011 μέχρι Μάιο 2012) και μπορεί να διαχωριστεί σε τρεις φάσεις: Απόκτηση του θεωρητικού υπόβαθρου, Υλοποίηση και Πειραματική Αξιολόγηση.

Στην πρώτη φάση, ο κυριότερος στόχος ήταν η απόκτηση του σχετικού θεωρητικού υπόβαθρου. Για τον σκοπό αυτό, μελετήθηκαν επιστημονικά άρθρα και σχετικά κεφάλαια βιβλίων για την κατανόηση του προβλήματος του Προγραμματισμού Δράσης και της σχέσης του με το Πρόβλημα της Ικανοποιησιμότητας. Ο δεύτερος στόχος αυτής της φάσης ήταν η εξοικείωση με τον κώδικα του προτασιακού επιλυτή PrecoSAT και ο εντοπισμός των σημείων του κώδικα που θα τροποποιούνταν κατά τη φάση της Υλοποίησης. Η φάση αυτή είχε χρονική διάρκεια δύο μηνών.

Η δεύτερη φάση στόχευε στην ανάπτυξη και υλοποίηση των τεχνικών που περιγράφονται αναλυτικότερα στο Κεφάλαιο 6. Οι τεχνικές αυτές επικεντρώνονται στην αποδοτικότερη επιλογή μεταβλητής απόφασης (decision variable selection) και επιλογή τιμής για τη μεταβλητή απόφασης (phase selection). Κατά τη φάση υλοποίησης, τροποποιήθηκε ο κώδικας του PrecoSAT 570 προκειμένου να παρέχεται η

δυνατότητα ενσωμάτωσης των νέων τεχνικών στην αρχική λειτουργία του PrecoSAT, όταν στην εντολή εκτέλεσης του επιλυτή, δοθεί η αντίστοιχη παράμετρος (command line argument) για κάθε τεχνική που ενδιαφέρει το χρήστη. Η φάση αυτή διήρκεσε περίπου πέντε μήνες.

Τέλος, στην τρίτη φάση έγινε πειραματική αξιολόγηση των τεχνικών που υλοποιήθηκαν, σε μεγάλο σύνολο προβλημάτων (benchmark domains). Έγιναν συγκρίσεις σχετικά με την αποδοτικότητα των τεχνικών, οι οποίες οδήγησαν σε διάφορα συμπεράσματα, και ολοκληρώθηκε η συγγραφή της διπλωματικής εργασίας. Η διάρκεια αυτής της φάσης ήταν περίπου δύο μήνες.

Κεφάλαιο 2

Προγραμματισμός Δράσης

2.1 Εισαγωγή στον Προγραμματισμό Δράσης 5
2.2 Η γλώσσα STRIPS 6
2.2.1 Αναπαράσταση προβλημάτων με τη γλώσσα STRIPS 6
2.2.2 Παράδειγμα αναπαράστασης προβλήματος με τη γλώσσα STRIPS 7
2.3 Το σύστημα SATPLAN 10
2.4 Κωδικοποίηση SAT-MAX-PLAN 11

2.1 Εισαγωγή στον Προγραμματισμό Δράσης

Ο Προγραμματισμός Δράσης, όπως αναφέρθηκε στην εισαγωγή, αφορά την εύρεση μιας ακολουθίας από δράσεις για την επίτευξη συγκεκριμένου στόχου. Δηλαδή, δεδομένης μιας αρχικής κατάστασης, ψάχνουμε μια ακολουθία από δράσεις που θα μας οδηγήσει σε μια τελική κατάσταση. Οι δράσεις πρέπει να ανήκουν στο προκαθορισμένο σύνολο επιτρεπτών δράσεων του προβλήματος.

Πιο συγκεκριμένα, σε ένα πρόβλημα Προγραμματισμού Δράσης, τα δεδομένα είναι τα εξής:

- Η περιγραφή του κόσμου (domain) του προβλήματος που επιθυμούμε να επιλύσουμε και το σύνολο των πιθανών δράσεων (actions) που μπορούμε να εφαρμόσουμε στο συγκεκριμένο κόσμο για την επίτευξη του στόχου
- Η αρχική κατάσταση (initial state)
- Η τελική κατάσταση (goal state) στην οποία θέλουμε να φτάσουμε

Το ζητούμενο είναι μια ακολουθία δράσεων από το σύνολο των πιθανών δράσεων, οι οποίες, όταν εκτελεστούν, ξεκινώντας από την αρχική κατάσταση καταλήγουμε στην τελική κατάσταση - στόχο.

Για την επίλυση προβλημάτων Προγραμματισμού Δράσης, η εφαρμογή αλγόριθμων αναζήτησης, όπως ο A*, δεν είναι αποδοτική, διότι τέτοιοι αλγόριθμοι οδηγούνται στην εξερεύνηση κάθε πιθανής λύσης για το πρόβλημα. Το γεγονός αυτό δημιουργεί πολλά προβλήματα στην επίλυση δύσκολων προβλημάτων, αφού η διαδικασία εξερεύνησης κάθε πιθανής λύσης είναι πολύ χρονοβόρα και ελάχιστα αποδοτική.

Για την επίτευξη αποδοτικότερων λύσεων στα προβλήματα Προγραμματισμού Δράσης, αναπτύχθηκαν εμπρόσθιοι (αναζήτηση ξεκινώντας από την αρχική προς την τελική κατάσταση) και οπίσθιοι (αναζήτηση ξεκινώντας από την τελική προς την αρχική κατάσταση) αλγόριθμοι. Οι αλγόριθμοι αυτοί στοχεύουν στη μείωση του χώρου αναζήτησης και στην αποδοτικότερη επίλυση των προβλημάτων Προγραμματισμού Δράσης, εκμεταλλευόμενοι τη δομή αναπαράστασης των προβλημάτων αυτών.

2.2 Η γλώσσα STRIPS

2.2.1 Αναπαράσταση προβλημάτων με τη γλώσσα STRIPS

Η γλώσσα STRIPS (Stanford Research Institute Problem Solver) αποτελεί μια μέθοδο αναπαράστασης των προβλημάτων Προγραμματισμού Δράσης και σκοπός της είναι η διευκόλυνση της επίλυσης αυτών των προβλημάτων. Μέσω της STRIPS, μπορούμε να αναπαραστήσουμε τους στόχους, τις καταστάσεις και τις δράσεις του προβλήματος χωρίς ασάφειες. Η STRIPS είναι αρκετά εκφραστική, ώστε να επιτρέπεται η περιγραφή πολλών προβλημάτων μέσω αυτής. Ταυτόχρονα όμως, είναι και αρκετά περιοριστική, για να μπορεί να χρησιμοποιηθεί από αποδοτικούς αλγορίθμους.

Στον Προγραμματισμό Δράσης, οι δράσεις είναι ντετερμινιστικές. Αυτό σημαίνει ότι η τρέχουσα κατάσταση και μια δράση καθορίζουν την επόμενη κατάσταση με μοναδικό τρόπο. Κάθε δράση αποτελείται από προϋποθέσεις (preconditions) και συνέπειες (effects). Οι προϋποθέσεις πρέπει να ισχύουν πριν από την εφαρμογή της δράσης, ενώ οι συνέπειες πρέπει να ισχύουν μετά την εφαρμογή της δράσης.

Οι καταστάσεις (states) είναι στιγμιότυπα του κόσμου (domain) του προβλήματος που προσπαθούμε να λύσουμε. Κάθε κατάσταση αποτελείται από συζεύξεις κατηγορημάτων (literals - facts). Ο τρόπος με τον οποίο μεταβαίνουμε από τη μια

κατάσταση στην άλλη είναι ο εξής: Από την τρέχουσα κατάσταση, με την εφαρμογή μιας δράσης της οποίας ικανοποιούνται οι προϋποθέσεις, μεταβαίνουμε σε μια άλλη (ή την ίδια) κατάσταση, στην οποία πρέπει να ικανοποιούνται οι συνέπειες της δράσης που εφαρμόστηκε.

Οι στόχοι (goals) θεωρούνται μερικές καταστάσεις. Ένας στόχος αποτελείται από συζεύξεις κατηγορημάτων. Μια κατάσταση ικανοποιεί το στόχο, αν όλα τα κατηγορήματα του στόχου και πιθανόν επιπρόσθετα κατηγορήματα, ικανοποιούνται.

Οι αδρανείς δράσεις (actionNOOPs) είναι δράσεις που, όταν εφαρμοστούν, δεν προκαλούν καμία αλλαγή στην κατάσταση (πχ. have(cake)).

Αμοιβαία αποκλειόμενοι (mutex – mutually exclusive actions) είναι οι όροι που δεν μπορούν να ισχύουν ταυτόχρονα (πχ. eat(cake) και have(cake)) και αποτρέπουν την εφαρμογή δράσεων που αλληλοαναιρούνται.

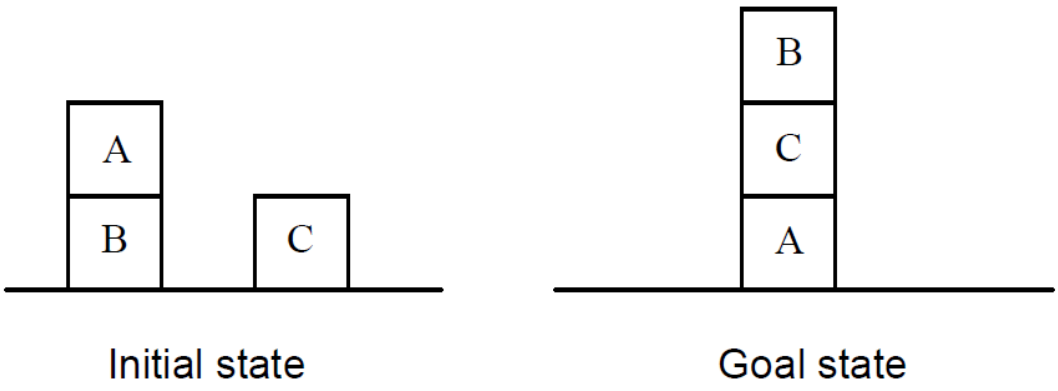
Στον Προγραμματισμό Δράσης ισχύει η υπόθεση του κλειστού κόσμου (closed world assumption). Δηλαδή, αν στην αναπαράσταση μια κατάστασης δεν εμφανίζονται κάποια κατηγορήματα που υπάρχουν στον κόσμο (domain) τότε υποθέτουμε ότι είναι ψευδή στη συγκεκριμένη κατάσταση.

2.2.2 Παράδειγμα αναπαράστασης προβλήματος με τη γλώσσα STRIPS

Ο κόσμος των κύβων (Blocks World) είναι ένας από τους πιο δημοφιλείς κόσμους προβλημάτων Προγραμματισμού Δράσης. Ο κόσμος αυτός αποτελείται από κύβους, οι οποίοι είναι τοποθετημένοι πάνω σε ένα τραπέζι. Οι κύβοι μπορούν να τοποθετηθούν ο ένας πάνω στον άλλον σχηματίζοντας στοίβα, αλλά το πολύ ένας κύβος μπορεί να τοποθετηθεί πάνω σε οποιονδήποτε κύβο. Ένα ρομποτικό χέρι μπορεί να πάρει έναν κύβο και να τον μετακινήσει σε μια άλλη θέση, είτε πάνω στο τραπέζι είτε πάνω σε κάποιον άλλο κύβο. Θεωρητικά, το τραπέζι είναι αρκετά μεγάλο, για να μπορούν να τοποθετηθούν όλοι οι κύβοι πάνω σε αυτό (όταν δηλαδή κανένας κύβος δεν είναι πάνω σε άλλον κύβο). Ο στόχος είναι ο σχηματισμός μιας στοίβας ή περισσότερων, για τις οποίες προκαθορίζεται ποιος κύβος θα είναι πάνω σε ποιον. [17]

Αξίζει να σημειωθεί ότι όλα τα προβλήματα του κόσμου των κύβων (Blocks World) έχουν ως λύση μια ακολουθία δράσεων, όπου ο αριθμός των δράσεων είναι μικρότερος ή ίσος με το διπλάσιο του συνολικού αριθμού των κύβων του προβλήματος. Δηλαδή, αν N είναι ο αριθμός των κύβων, στη χειρότερη περίπτωση πρώτα θα τοποθετηθούν όλοι οι κύβοι πάνω στο τραπέζι (το πολύ N δράσεις) και μετά θα σχηματιστούν οι στοίβες (και πάλι το πολύ N δράσεις). Επομένως, θα χρειαστούν συνολικά το πολύ 2*N δράσεις.

Ακολουθεί ένα παράδειγμα αναπαράστασης του προβλήματος Blocks World στη γλώσσα STRIPS.



Σχήμα 2.1 Η αρχική (initial state) και τελική (goal state) κατάσταση για το παράδειγμα του προβλήματος Blocks World.

Ένα παράδειγμα σχηματικής απεικόνισης της αρχικής και τελικής κατάστασης ενός στιγμιοτύπου του προβλήματος Blocks World φαίνεται στο πιο πάνω σχήμα. Τα κατηγορήματα και οι δράσεις που περιγράφονται θα μπορούσαν να χρησιμοποιηθούν για οποιεσδήποτε αρχικές και τελικές καταστάσεις του προβλήματος Blocks World.

Κατηγορήματα:

on(B, O): Ο κύβος Β είναι πάνω στο Ο

clear(B): Ο κύβος Β δεν έχει κανένα κύβο πάνω του

table(O): Το Ο είναι το τραπέζι

block(B): Το Β είναι κύβος

equal(B1, B2): Το Β1 ισούται με το Β2 (τα Β1 και Β2 είναι ο ίδιος κύβος)

Δράσεις:

1) puton(B1, B2, O): Τοποθέτησε τον κύβο Β1 από τον Ο πάνω στον Β2.

- Προϋποθέσεις (preconditions): on(B1,O) ∧ clear(B1) ∧ clear(B2) ∧ ¬equal(B1, O) ∧ ¬equal(B2, B1) ∧ ¬equal(B2, O) ∧ block(B1) ∧ block(B2)
- Συνέπειες (effects): on(B1, B2) ∧ clear(O) ∧ ¬on(B1,O) ∧ ¬clear(B2)

2) putTable(B1, T, B2): Τοποθέτησε τον κύβο Β1 από το Β2 πάνω στο τραπέζι

- Προϋποθέσεις (preconditions): on(B1, B2) ∧ clear(B1) ∧ ¬equal(B1, B2) ∧ block(B2) ∧ table(T)
- Συνέπειες (effects): on(B1, T) ∧ clear(B2) ∧ ¬on(B1,B2)

Αρχική κατάσταση (initial state):

block(a) ∧ block(b) ∧ block(c) ∧ table(t) ∧ on(a, b) ∧ on(b, t) ∧ on(c, t) ∧ clear(a) ∧ clear(c)

Τελική κατάσταση (goal state):

on(c, a) ∧ on (b, c)

2.3 Το σύστημα SATPLAN

Το σύστημα SATPLAN αποτελεί ένα από τα κορυφαία συστήματα επίλυσης προβλημάτων Προγραμματισμού Δράσης. Όπως αναφέρθηκε στην εισαγωγή, το σύστημα SATPLAN προτάθηκε το 1996 από τους Henry Kautz και Bart Selman και πήρε την πρώτη θέση στο συνέδριο ICAPS 2004 (International Conference on Automated Planning and Scheduling) [7]. Το 2006 δημιουργήθηκε μια πιο αποδοτική έκδοση του SATPLAN, η οποία χρησιμοποιείται στην παρούσα διπλωματική εργασία.

Το σύστημα SATPLAN 2006 δέχεται ως είσοδο τα προβλήματα που ανήκουν στο υποσύνολο STRIPS της γλώσσας PDDL (Planning Domain Definition Language) και προσπαθεί να βρει λύσεις με το μικρότερο σύνολο χρονικών βημάτων, ανάγοντας το πρόβλημα Προγραμματισμού Δράσης στο πρόβλημα της Ικανοποιησιμότητας (SAT) [11]. Αρκετοί ερευνητές προσπαθούν να τροποποιήσουν τον κώδικά του και να εφαρμόσουν νέες μεθόδους, για να βελτιώσουν την αποδοτικότητα του συστήματος, κάνοντας το σύστημα εξίσου ανταγωνιστικό με άλλες δημοφιλείς τεχνολογίες επίλυσης προβλημάτων προγραμματισμού δράσης.

Πιο κάτω φαίνεται ο αλγόριθμος που χρησιμοποιεί το σύστημα SATPLAN, για να λύσει ένα πρόβλημα Προγραμματισμού Δράσης.

```
1.      SATPLAN(problem, T_maximum){
2.          //inputs: a) problem is a planning problem,
3.          //          b) T_maximum is an upper limit for plan length
4.          for T = 0 to T_maximum{
5.                  cnf = translate_to_sat(problem, T);
6.                  assignment = call_sat_solver( cnf );
7.                  if ( assignment<>null ) {
8.                          return extract_solution( assignment, mapping );
9.                  }
10.         }
11.         return failure;
12.     }
```

Περιγραφή του αλγόριθμου SATPLAN

Αρχικά, το σύστημα SATPLAN κατασκευάζει το γράφημα δράσεων μέχρι κάποιο χρονικό επίπεδο k και μεταφράζει τους περιορισμούς που προκύπτουν σε Συζευκτική Κανονική Μορφή (Conjunctive Normal Form – CNF). Στην προτασιακή λογική, η Συζευκτική Κανονική Μορφή είναι σύζευξη προτάσεων (clauses) που αποτελούνται από διαζεύξεις μεταβλητών (variables).

Το αρχείο που περιέχει τις προτάσεις σε Συζευκτική Κανονική Μορφή, δίνεται ως είσοδος στον προτασιακό επιλυτή (SAT solver), ο οποίος καλείται να βρει μια ανάθεση τιμών στις μεταβλητές που ικανοποιεί όλες οι προτάσεις.

Αν υπάρχει τέτοια ανάθεση τιμών που ικανοποιεί όλες τις προτάσεις τότε το σύστημα SATPLAN παράγει ως έξοδο τη λύση του αρχικού προβλήματος. Αν, αντίθετα, δεν υπάρχει τέτοια ανάθεση τιμών, τότε ο SATPLAN δημιουργεί το γράφημα δράσεων μέχρι το επόμενο επίπεδο (k+1) και η διαδικασία επαναλαμβάνεται.

2.4 Κωδικοποίηση SAT-MAX-PLAN

Υπάρχουν διάφορες κωδικοποιήσεις των περιορισμών του προβλήματος Προγραμματισμού Δράσης σε συζευκτική κανονική μορφή. Στην παρούσα διπλωματική εργασία χρησιμοποιούμε την κωδικοποίηση SAT-MAX-PLAN (SMP), η οποία θεωρείται καλύτερη κωδικοποίηση, διότι επιτυγχάνει μεγαλύτερη διάδοση περιορισμών από άλλες γνωστές κωδικοποιήσεις όπως Blackbox και SatPlan06 [18].

Οι προτάσεις σε συζευκτική κανονική μορφή, χρησιμοποιώντας την κωδικοποίηση SMP δημιουργούνται, όπως φαίνεται πιο κάτω [18]:

1. Μοναδιαίες διαζεύξεις για την αρχική και την τελική κατάσταση
2. $A(T) \rightarrow f(T)$, για κάθε δράση A και για κάθε όρο f, όπου $f \in pre(A)$.
3. $A(T) \rightarrow f(T + 1)$, για κάθε δράση A και για κάθε όρο f, όπου $f \in add(A)$.
4. $A(T) \rightarrow \neg f(T + 1)$, για κάθε δράση A και για κάθε όρο f, όπου $f \in del(A)$.

5. f(T) → A1(T − 1) ∨ . . . ∨ Am(T − 1), για κάθε όρο f και για κάθε δράση Ai, όπου 1 ≤ i ≤ m (συμπεριλαμβανομένων των αδρανών (noop) δράσεων) και f ∈ add(Ai).

6. ¬f(T) → A1(T − 1) ∨ . . . ∨ Am(T − 1) ∨ ¬f(T − 1), για κάθε όρο f και για όλες τις δράσεις Ai, όπου 1 ≤ i ≤ m και f ∈del(Ai).

7. ¬A1(T)∨¬A2(T), για κάθε ζεύγος δράσεων A1,A2 τέτοιο ώστε το σύνολο del(A1)∩pre(A2) ≠∅.

8. ¬ f1(T) ∨ ¬ f2(T), για κάθε ζεύγος όρων f1,f2 που είναι αμοιβαία αποκλειόμενοι τη χρονική στιγμή T.

Επεξηγήσεις συμβολισμών :

pre (A) = preconditions (προϋποθέσεις) της δράσης A

add (A) = θετικές συνέπειες της δράσης A

del (A) = αρνητικές συνέπειες της δράσης A

Η μετατροπή του προβλήματος σε Συζευκτική Κανονική Μορφή (CNF) έχει ως στόχο την αναγωγή του προβλήματος σε πρόβλημα Ικανοποιησιμότητας και την επίλυσή του με τη χρήση ενός προτασιακού επιλυτή (πχ. PrecoSAT).

Κεφάλαιο 3

Ικανοποιησιμότητα και Προτασιακή Λογική

3.1 Το Πρόβλημα της Ικανοποιησιμότητας 13
3.2 Προτασιακή Λογική 15
3.2.1 Εισαγωγή 15
3.2.2 Σύνταξη και σημασιολογία 16
3.2.3 Προτασιακή Επίλυση 19

3.1 Το Πρόβλημα της Ικανοποιησιμότητας

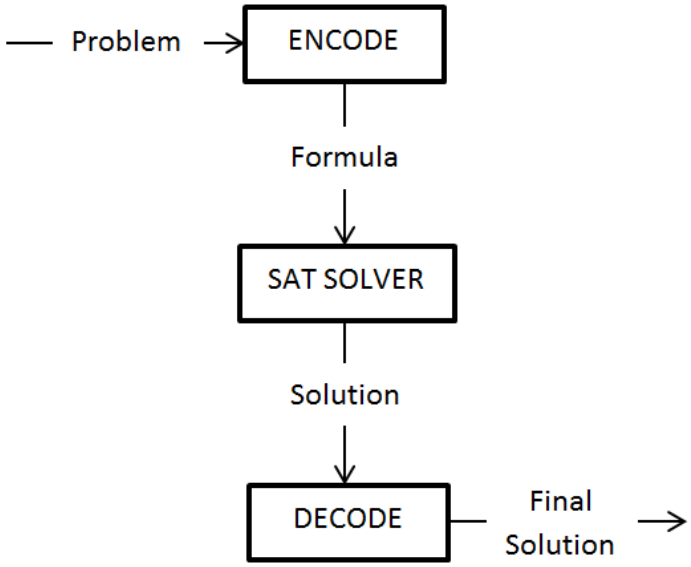
Το πρόβλημα της Προτασιακής Ικανοποιησιμότητας (SAT – Propositional Satisfiability) είναι το πρόβλημα της εύρεσης κάποιας ανάθεσης τιμών στις μεταβλητές μιας φόρμουλας (CNF formula), ώστε η φόρμουλα να ικανοποιείται, δηλαδή να γίνεται αληθής (TRUE). Αν δεν υπάρχει καμία τέτοια ανάθεση, σημαίνει ότι για όλες τις δυνατές αναθέσεις η πρόταση γίνεται ψευδής (FALSE). Σε αυτή την περίπτωση, λέμε ότι η πρόταση είναι μη ικανοποιήσιμη.

Το πρόβλημα SAT ήταν το πρώτο γνωστό NP-πλήρες πρόβλημα. Η απόδειξη έγινε από τον Stephen Cook το 1971 [7]. Δεν έχει βρεθεί αλγόριθμος που λύνει όλα τα στιγμιότυπα του SAT και οι επιστήμονες τείνουν να πιστεύουν ότι δεν υπάρχει τέτοιος αλγόριθμος, αν και δεν έχει αποδειχθεί ακόμα (πρόβλημα P versus NP). Η πολυπλοκότητα όμως αυτή αφορά τη χειρότερη περίπτωση. Στην πραγματικότητα, πολλά προβλήματα που αντιμετωπίζει η βιομηχανία μπορούν να λυθούν σε σύντομο χρονικό διάστημα. Το γεγονός αυτό έδωσε κίνητρα στους επιστήμονες για την ανάπτυξη συστημάτων που επιχειρούν να λύσουν το πρόβλημα της Ικανοποιησιμότητας εφαρμόζοντας εξειδικευμένες τεχνικές και ευρετικά, για να μειώσουν το χρόνο επίλυσης. Επίσης, υπάρχουν ειδικές περιπτώσεις του προβλήματος SAT, όπως το 2-SAT, που μπορούν να λυθούν σε πολυωνυμικό χρόνο.

Για την επίλυση στιγμιοτύπων του προβλήματος της Ικανοποιησιμότητας, αναπτύχθηκαν διάφορα συστήματα που είναι γνωστά ως προτασιακοί επιλυτές (SAT solvers). Αρχικά, οι τεχνικές που χρησιμοποιούσαν αυτά τα συστήματα δεν ήταν αρκετά αποδοτικές. Όμως, τα τελευταία 10 περίπου χρόνια, αναπτύχθηκαν καινοτόμες τεχνικές που βελτίωσαν δραματικά την αποδοτικότητα των προτασιακών επιλυτών. Μοντέρνες τεχνικές, όπως η εκμάθηση προτάσεων μετά από σύγκρουση (Conflict-Directed Clause Learning, CDCL), οι επανεκκινήσεις της αναζήτησης ανά χρονικά διαστήματα (Restarts), τα εξελιγμένα ευρετικά επιλογής μεταβλητών απόφασης (Decision heuristics) και οι βελτιωμένες δομές δεδομένων (Data structures) που χρησιμοποιούνται, συνέβαλαν στη μείωση του χρόνου επίλυσης των προβλημάτων, με αποτέλεσμα τις σημερινές εντυπωσιακές επιδόσεις των προτασιακών επιλυτών.

Για πολλά προβλήματα από διάφορα πεδία, όπως χρονοδρομολόγηση (scheduling), προγραμματισμό δράσης (planning) και επαλήθευση λογισμικού (software verification), οι προτασιακοί επιλυτές μπορούν να λειτουργήσουν σαν «μαύρο κουτί» και να συμβάλουν σε μεγάλο βαθμό στην επίλυση των προβλημάτων. Αυτό μπορεί να επιτευχθεί μέσω ενός μετασχηματισμού του αρχικού προβλήματος σε πρόβλημα Ικανοποιησιμότητας. Δηλαδή, ο μετασχηματισμός έχει ως στόχο την αναπαράσταση του προβλήματος με μια φόρμουλα σε Συζευκτική Κανονική Μορφή (CNF). Η επίλυση του προβλήματος Ικανοποιησιμότητας, δηλαδή η εύρεση ανάθεσης τιμών για τις μεταβλητές της φόρμουλας, ώστε η πρόταση να ικανοποιείται, μπορεί να γίνει μέσω των προτασιακών επιλυτών (SAT solvers). Αν μια τέτοια ανάθεση βρεθεί, η ανάθεση αυτή μπορεί να μετασχηματιστεί πίσω στο αρχικό πρόβλημα και να δώσει τη λύση του αρχικού προβλήματος. Η ιδιότητα αυτή παρέχει τη δυνατότητα χρήσης των προτασιακών επιλυτών (SAT solvers), όπως ο PrecoSAT, σε άλλα συστήματα, όπως ο SATPLAN.

Το σχήμα 3.1 παρουσιάζει πώς χρησιμοποιούνται οι προτασιακοί επιλυτές, για να λύσουν τα διάφορα προβλήματα. Το πρόβλημα μέσω του κωδικοποιητή (encoder) μετατρέπεται σε προτασιακή φόρμουλα, η οποία δίνεται ως είσοδος στον προτασιακό επιλυτή. Η έξοδος του προτασιακού επιλυτή δίνεται ως είσοδος στον αποκωδικοποιητή (decoder), ο οποίος παράγει την τελική λύση, δηλαδή τη λύση στο αρχικό πρόβλημα σε μορφή που μπορεί να κατανοήσει ο χρήστης.



Σχήμα 3.1 Τα στάδια επίλυσης ενός προβλήματος με τη βοήθεια κάποιου προτασιακού επιλυτή.

3.2 Προτασιακή Λογική

3.2.1 Εισαγωγή

Η Προτασιακή Λογική (Propositional Logic) είναι ένας φορμαλισμός κάποιων απλών μορφών συλλογιστικής. Είναι το απλούστερο λογικό σύστημα. Οι ισχυρισμοί αποτελούνται από προτάσεις οι οποίες είναι είτε αληθείς είτε ψευδείς αλλά όχι και τα δύο.

Οι προτασιακοί επιλυτές χειρίζονται φόρμουλες σε Συζευκτική Κανονική Μορφή (CNF). Στην προτασιακή λογική, μια φόρμουλα είναι σε Συζευκτική Κανονική Μορφή όταν είναι σύζευξη προτάσεων που αποτελούνται από διαζεύξεις κατηγορημάτων. Η ανάθεση τιμών στις μεταβλητές, δηλαδή η αντιστοίχιση των μεταβλητών σε μια από τις τιμές true ή false, καθορίζει αν η φόρμουλα θα είναι αληθής ή ψευδής.

3.2.2 Σύνταξη και σημασιολογία

Το αλφάβητο της προτασιακής λογικής αποτελείται από τα επόμενα σύμβολα:

- 'Ατομα, πχ. P, Q, ή P_1, P_2, ή 1, 2, 3 κτλ.
- Πέντε λογικούς συνδέσμους:

¬ (άρνηση), ∧ (σύζευξη), ∨ (διάζευξη), → (συνεπαγωγή), ↔ (ισοδυναμία).

- Δύο σύμβολα στίξης: "(" και ")".

Η άρνηση είναι μοναδιαίος τελεστής, ενώ οι υπόλοιποι τελεστές είναι δυαδικοί.

Πιο κάτω φαίνεται ο Πίνακας Αληθείας για τους λογικούς συνδέσμους ¬ (άρνηση), ∧ (σύζευξη), ∨ (διάζευξη), → (συνεπαγωγή), ↔ (ισοδυναμία).

Πίνακας 3.1 Ο Πίνακας Αληθείας των λογικών τελεστών της Προτασιακής Λογικής.

P	**Q**	**¬P**	**P ∨Q**	**P ∧Q**	**P → Q**	**P ↔ Q**
0	0	1	0	0	1	1
0	1	1	1	0	1	0
1	0	0	1	0	0	0
1	1	0	1	1	1	1

Ορισμοί:

1) Ένα άτομο (atom) είναι μια μεταβλητή V ή η άρνησή της, ¬V.

2) Ένα κατηγόρημα (literal) είναι ένα άτομο L ή η άρνησή του, ¬L.

3) Η πολικότητα (polarity) ενός κατηγορήματος είναι αρνητική (negative), αν το κατηγόρημα είναι η άρνηση ενός ατόμου ¬L. Αλλιώς, η πολικότητα του κατηγορήματος είναι θετική (positive).

4) Μια πρόταση (clause) είναι διάζευξη (disjunction) κατηγορημάτων (literals).

5) Μια φόρμουλα (formula) είναι σύζευξη (conjunction) προτάσεων (clauses).

6) Ο αριθμός κατηγορημάτων σε μια πρόταση λέγεται μέγεθος της πρότασης (size of the clause).

7) Μοναδιαία πρόταση (unit clause) είναι μια πρόταση που αποτελείται από μόνο ένα κατηγόρημα.

8) Δυαδική πρόταση (binary clause) είναι μια πρόταση που αποτελείται από μόνο δύο κατηγορήματα.

9) Αγνό κατηγόρημα (pure literal) είναι μια μεταβλητή που εμφανίζεται μόνο θετικά ή μόνο αρνητικά.

10) Κενή πρόταση (Empty Clause) είναι μια πρόταση που δεν περιέχει κανένα κατηγόρημα.

11) Μια φόρμουλα ανήκει στο k-SAT αν κάθε πρόταση της φόρμουλας περιέχει το πολύ k κατηγορήματα (πχ. 2-SAT, 3-SAT).

12) Μια συνεπαγωγή της μορφής $(L1 \vee L2 \vee L3\ldots \vee Ln) \rightarrow L'$, η οποία δεν είναι σε Συζευκτική Κανονική Μορφή (CNF), ισοδυναμεί με την πρόταση $(\neg L1 \vee \neg L2 \vee \neg L3 \vee \ldots \neg Ln \vee L')$, η οποία είναι Συζευκτική Κανονική Μορφή.

Γενικά, οποιαδήποτε φόρμουλα της προτασιακής λογικής μπορεί να μετατραπεί σε Συζευκτική Κανονική Μορφή. Η μέθοδος που ακολουθείται για τη μετατροπή μιας προτασιακής φόρμουλας σε Συζευκτική Κανονική Μορφή παραλείπεται, διότι δε χρησιμοποιείται στην παρούσα διπλωματική. Περιγραφή της μεθόδου αυτής υπάρχει στο [16].

Για σκοπούς απλοποίησης, στα παραδείγματα που ακολουθούν, οι προτάσεις θα αναπαρίστανται, όπως περιγράφεται πιο κάτω:

Μια πρόταση που αντιπροσωπεύει διάζευξη κατηγορημάτων (L1 ∨ L2 ∨ (L3 ∨ L4) ∨ L5) θα αναπαρίσταται ως [L1 + L2 + L3 + L4 + L5]. Τα κατηγορήματα μιας πρότασης μπορούν να αλλάξουν θέσεις μεταξύ τους, επειδή ο τελεστής της διάζευξης έχει την αντιμεταθετική ιδιότητα.

Μια φόρμουλα που αποτελεί σύζευξη προτάσεων, για παράδειγμα (C1 ∧ (C2 ∧ C3)), θα αναπαρίσταται με τα σύμβολα «<» και «>», δηλαδή <C1, C2, C3>.

Για παράδειγμα, η αναπαράσταση της φόρμουλας F = (1 ∧ (2 ∨ 3 ∨ 4) ∧ (5 ∨ 6) ∧ 7), η οποία είναι σε Συζευκτική Κανονική Μορφή (CNF) είναι η εξής:

F= < [1], [2 + 3 + 4], [5 + 6], [7]>

Ακολουθούν οι κανόνες σχετικά με την ικανοποιησιμότητα των προτάσεων και της φόρμουλας, καθώς και κάποιες παρατηρήσεις που προκύπτουν:

Οι προτασιακές μεταβλητές (propositional variables) μπορεί να είναι είτε αληθείς (true) είτε ψευδείς (false) αλλά όχι και τα δύο. Επομένως, μια φόρμουλα μπορεί να είναι είτε αληθής είτε ψευδής, ανάλογα με τις τιμές των μεταβλητών από τις οποίες αποτελείται, καθώς και τους τελεστές που έχουν μεταξύ τους. Με άλλα λόγια, η ανάθεση τιμών στις μεταβλητές, δηλαδή η αντιστοίχιση των μεταβλητών σε μια από τις τιμές true ή false, καθορίζει αν η φόρμουλα θα είναι αληθής ή ψευδής.

Μια πρόταση ικανοποιείται, όταν ικανοποιείται τουλάχιστον ένα από τα κατηγορήματα που την αποτελούν. Ένα κατηγόρημα ικανοποιείται, αν είναι άτομο που πήρε την τιμή «αληθές» (true) ή αν είναι άρνηση ατόμου που πήρε την τιμή «ψευδές» (false). Η κενή πρόταση είναι πάντα μη ικανοποιήσιμη.

Μια φόρμουλα ικανοποιείται, αν ικανοποιούνται όλες οι προτάσεις από τις οποίες αποτελείται. Αν υπάρχει ανάθεση με την οποία η φόρμουλα ικανοποιείται, τότε λέμε ότι η φόρμουλα είναι ικανοποιήσιμη (satisfiable). Διαφορετικά, η φόρμουλα είναι μη ικανοποιήσιμη (unsatisfiable).

3.2.3 Προτασιακή επίλυση

Η προτασιακή επίλυση (resolution) είναι μια τεχνική μετατροπής της φόρμουλας σε μια ισοδύναμη φόρμουλα και χρησιμοποιείται στη διαδικασία επίλυσης των SAT προβλημάτων.

Έστω οι προτάσεις C1 = [x + La1 + La2 + La3… + Lan] και C2 = [¬x + Lb1 + Lb2 + Lb3… + Lbm]. Οι προτάσεις C1 και C2 έχουν κοινή μεταβλητή το x, με διαφορετικές πολυκότητες (στη C1 το x έχει θετική πολικότητα, ενώ στη C2 το x έχει αρνητική πολικότητα).

Η επιλύουσα (resolvent) των C1 και C2 είναι η πρόταση C = [La1+ La2+ La3… +Lan+ Lb1+ Lb2 + Lb3… + Lbm], η οποία περιέχει όλα τα κατηγορήματα καθεμίας από τις C1 και C2, εκτός από τα x και ¬x, και αντικαθιστά τις προτάσεις C1 και C2, μειώνοντας τον συνολικό αριθμό προτάσεων της φόρμουλας.

Η διαδικασία της επίλυσης μειώνει τον αριθμό των προτάσεων της φόρμουλας, αλλά, συνήθως, αυξάνει πολύ το μέγεθος των προτάσεων. Το γεγονός αυτό αποτελεί ένα από τα κυριότερα μειονεκτήματα της μεθόδου αυτής, όταν χρησιμοποιείται ως μέρος της λειτουργίας των προτασιακών επιλυτών.

Κεφάλαιο 4

Προτασιακοί Επιλυτές

4.1 Εισαγωγή 20
4.2 Τροποποιήσεις της αρχικής φόρμουλας 21
4.3 Δέντρα Αναζήτησης 22
4.4 Αλγόριθμος DPLL 23
4.4.1 Εισαγωγή 23
4.4.2 Περιγραφή του αλγόριθμου DPLL 24
4.4.3 Παράδειγμα εκτέλεσης του αλγόριθμου DPLL 26
4.5 Αλγόριθμος CDCL 29
4.5.1 Εισαγωγή 29
4.5.2 Περιγραφή του αλγόριθμου CDCL 30
4.6 Τεχνικές των μοντέρνων προτασιακών επιλυτών 34
4.6.1 Εισαγωγή 34
4.6.2 Προεπεξεργαστής 35
4.6.3 Μοναδιαία Μετάδοση 35
4.6.4 Ευρετικό Απόφασης 36
4.6.5 Ανάλυση σύγκρουσης 37
4.6.6 Διαγραφή εκμαθημένων προτάσεων 38
4.6.7 Επανεκκίνηση 39
4.6.8 Αλληλεπιδράσεις μεταξύ των κομματιών ενός προτασιακού επιλυτή 40

4.1 Εισαγωγή

Οι προτασιακοί επιλυτές είναι συστήματα που προσπαθούν να επιλύσουν Προβλήματα Ικανοποιησιμότητας, δηλαδή να αναθέσουν τιμές στις μεταβλητές μιας φόρμουλας σε Συζευκτική Κανονική Μορφή, ώστε η φόρμουλα να ικανοποιείται.

Οι μοντέρνοι προτασιακοί επιλυτές καλούνται να λύσουν προβλήματα με εκατομμύρια μεταβλητές και ακόμα περισσότερες προτάσεις (clauses). Ο χώρος αναζήτησης τέτοιων προβλημάτων είναι τεράστιος και ο έλεγχος όλων των δυνατών αναθέσεων μέσα σε λογικά χρονικά πλαίσια είναι ανέφικτος. Πιο συγκεκριμένα, αν N είναι ο αριθμός των μεταβλητών μιας φόρμουλας F, τότε πρέπει να ελεγχθούν 2^N αναθέσεις, κάτι πολύ χρονοβόρο και καθόλου αποδοτικό. Για το λόγο αυτό, επινοήθηκαν έξυπνοι αλγόριθμοι και τεχνικές που μειώνουν το χώρο αναζήτησης, συνεπώς και το χρόνο επίλυσης των Προβλημάτων Ικανοποιησιμότητας.

Οι πιο γνωστοί αλγόριθμοι που επινοήθηκαν είναι οι DPLL (Davis Putman Logemann Loveland) και CDCL (Conflict Driven Clause Learning). Και οι δύο αλγόριθμοι στηρίζονται στα δέντρα αναζήτησης (search trees) που περιγράφονται παρακάτω. Ο αλγόριθμος DPLL είναι ο παλαιότερος από τους δύο αλγορίθμους και δε χρησιμοποιείται στους μοντέρνους προτασιακούς επιλυτές. Οι σύγχρονοι προτασιακοί επιλυτές στηρίζονται στον αλγόριθμο CDCL, που είναι μια εξελιγμένη μορφή του αλγόριθμου DPLL. Εκτενής περιγραφή των αλγόριθμων DPLL και CDCL ακολουθεί στη συνέχεια.

4.2 Τροποποιήσεις της αρχικής φόρμουλας

Για την αποδοτικότερη λειτουργία των προτασιακών επιλυτών (SAT solvers), γίνονται κάποιες τροποποιήσεις της φόρμουλας που δίνεται ως είσοδος στον προτασιακό επιλυτή. Οι τροποποιήσεις αυτές δεν επηρεάζουν την ικανοποιησιμότητα της φόρμουλας ή το πρόβλημα το οποίο αντιπροσωπεύει. Αυτό που επιτυγχάνουν είναι την αποφυγή άσκοπων ενεργειών.

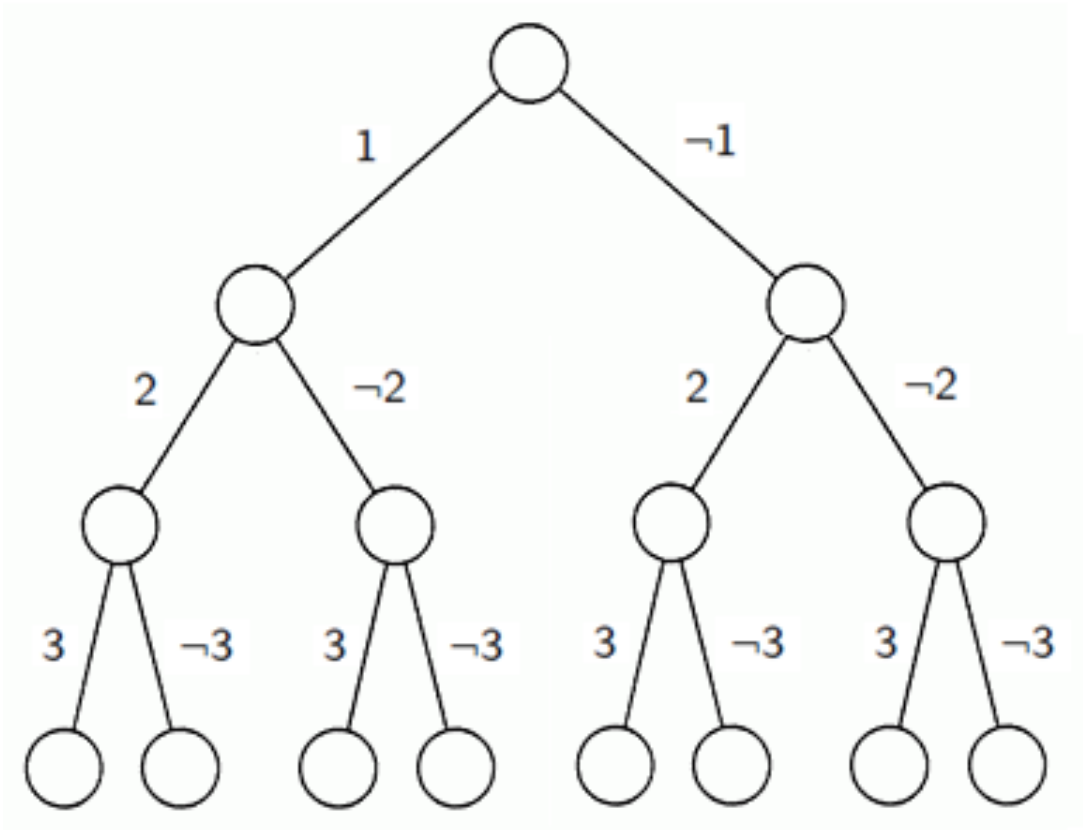
Πιο συγκεκριμένα, στην τροποποιημένη φόρμουλα ισχύουν τα εξής:

1) Κάθε κατηγόρημα (literal) εμφανίζεται το πολύ μια φορά μέσα σε κάποια πρόταση (clause). Εάν εμφανίζεται περισσότερες από μία φορές, οι επανεμφανίσεις του κατηγορήματος στην ίδια πρόταση διαγράφονται. Αυτό δεν επηρεάζει την ικανοποιησιμότητα της πρότασης, συνεπώς ούτε και της φόρμουλας.

2) Καμία πρόταση δεν περιέχει κάποιο κατηγόρημα και την άρνησή του. Αυτό πρέπει να ισχύει, διότι μια τέτοια πρόταση θα ικανοποιείται πάντοτε, με αποτέλεσμα να μην περιορίζει το χώρο αναζήτησης. Γι αυτό, αν μια πρόταση περιέχει κάποιο κατηγόρημα και την άρνησή του, αφαιρείται πριν από την έναρξη της αναζήτησης.

4.3 Δέντρα Αναζήτησης

Τα δέντρα αναζήτησης (search trees) στο πρόβλημα της Ικανοποιησιμότητας (SAT) είναι δυαδικά δέντρα αναζήτησης. Δηλαδή, κάθε κόμβος Κ έχει δύο παιδιά, Κ1 και Κ2. Η ακμή από τον κόμβο-πατέρα Κ προς το αριστερό παιδί, Κ1, αντιπροσωπεύει ένα κατηγόρημα, έστω L, ενώ η ακμή από τον κόμβο-πατέρα προς το δεξί παιδί, Κ2, αντιπροσωπεύει το κατηγόρημα ¬L. Τα κατηγορήματα L και ¬L πρέπει να αντιστοιχούν στην ίδια μεταβλητή, η οποία δεν πρέπει να εμφανίζεται σε καμία ακμή κανενός μονοπατιού που ξεκινά από τη ρίζα και φτάνει στον κόμβο-πατέρα Κ.



Σχήμα 4.1 Παράδειγμα Δέντρου Αναζήτησης

Κάθε κόμβος έχει κάποιο ύψος, το οποίο ισούται με τον αριθμό των διακλαδώσεων (branches) στο μονοπάτι που φτάνει στον κόμβο. Το μονοπάτι αυτό αντιστοιχεί στη μερική ανάθεση (δηλαδή ανάθεση τιμών σε μόνο μερικές από τις μεταβλητές) που περιέχει όλα τα κατηγορήματα (literals) που αντιστοιχούν στις ακμές του μονοπατιού. Για την επίλυση ενός προβλήματος ικανοποιησιμότητας, χρειάζεται να βρεθεί ένα μονοπάτι που ξεκινά από τη ρίζα του δέντρου αναζήτησης, το οποίο δίνει τιμή σε όλες τις μεταβλητές του προβλήματος (πλήρης ανάθεση) και ικανοποιεί όλες τις προτάσεις (clauses) του προβλήματος.

Μπορούμε να σταματήσουμε να ακολουθούμε ένα μονοπάτι μόλις η αντίστοιχη μερική ανάθεση των μεταβλητών κάνει κάποια πρόταση ψευδή (false). Στην περίπτωση αυτή, το μονοπάτι λέγεται κλειστό μονοπάτι. Αντίθετα, αν κάποιο μονοπάτι περιέχει όλες τις μεταβλητές της φόρμουλας και δεν κάνει ψευδή καμία πρόταση της φόρμουλας, τότε η αντίστοιχη ανάθεση των μεταβλητών αποτελεί λύση του προβλήματος και η φόρμουλα είναι ικανοποιήσιμη. Το δέντρο αναζήτησης πρέπει να επεκτείνεται με νέους κόμβους-παιδιά μέχρι να βρεθεί μονοπάτι που περιέχει όλες τις μεταβλητές ή μέχρι όλα τα μονοπάτια να κλείσουν. Στη δεύτερη περίπτωση, η πρόταση είναι μη ικανοποιήσιμη.

Η έγκαιρη εύρεση των κλειστών μονοπατιών είναι καίριας σημασίας, αφού μειώνει κατά πολύ το χώρο αναζήτησης. Πιο συγκεκριμένα, μια πρόταση μεγέθους N μπορεί να αφαιρέσει μέχρι και το 2^N-ο κομμάτι του χώρου αναζήτησης. Οι μικρότερες προτάσεις οδηγούν σε γρηγορότερη αναζήτηση, επειδή «κλαδεύουν» το δέντρο αναζήτησης περισσότερο από τις μεγαλύτερες προτάσεις. Επομένως, είναι πολύ σημαντικό να «κλαδέψουμε» το δέντρο αναζήτησης στην αρχή της αναζήτησης, τότε που οι προτάσεις οι οποίες μπορεί να γίνουν ψευδείς και να κλείσουν το μονοπάτι, έχουν μικρό μέγεθος.

4.4 Αλγόριθμος DPLL

4.4.1 Εισαγωγή

Το 1960 οι M. Davis και H. Putnam πρότειναν τον αλγόριθμο DP. Ο αλγόριθμος αυτός στηριζόταν στην Προτασιακή Επίλυση (resolution) και δεν ήταν αρκετά αποδοτικός. Το

1962, οι M. Davis, G. Logemann και D. Loveland πρότειναν έναν εναλλακτικό αλγόριθμο, πιο αποδοτικό, που είναι γνωστός ως DPLL αλγόριθμος.

Ο αλγόριθμος DPLL είναι ένας αναδρομικός αλγόριθμος, ο οποίος στηρίζει τη λειτουργία του στο δέντρο αναζήτησης και στη μέθοδο οπισθοχώρησης (backtracking).

4.4.2 Περιγραφή του αλγόριθμου DPLL

Πιο κάτω παρουσιάζεται ο ψευδοκώδικας για τον αλγόριθμο DPLL και στη συνέχεια γίνεται αναλυτική περιγραφή των βημάτων από τα οποία αποτελείται.

Αλγόριθμος DPLL

```
1.      boolean DPLL ( ClauseSet S )
2.      {
3.              while (S contains a unit clause {L}) {
4.                      delete from S all clauses containing L;
5.                      delete ¬L from all clauses in S;
6.              }
7.              if (∅ ∈ S) return UNSATISFIABLE;
8.              while (S contains a pure literal L ) {
9.                      delete from S all clauses containing L;
10.             }
11.             if (S = ∅) return SATISFIABLE;
12.             choose a literal L occurring in S;
13.             if (DP(S ∪ {{L}}) return SATISFIABLE;
14.             else if(DP(S ∪ {{¬L}}) ) return SATISFIABLE;
15.             else return UNSATISFIABLE;
16.     }
```

Περιγραφή του αλγόριθμου DPLL

Γραμμές 3-6: Σε αυτό το σημείο, ο αλγόριθμος ψάχνει για μοναδιαίες προτάσεις (unit clauses) και τις ικανοποιεί αναθέτοντας την τιμή TRUE στα κατηγορήματα των μοναδιαίων προτάσεων (κάθε μοναδιαία πρόταση έχει ακριβώς ένα κατηγόρημα). Επειδή ο κανόνας αυτός «κόβει» στα δύο το κλαδί στο οποίο βρίσκεται η αναζήτηση εκείνη τη στιγμή, θεωρείται πολύ σημαντικός και έχει μεγάλη θετική επίδραση στην

αποδοτικότητα των προτασιακών επιλυτών. Ο κανόνας αυτός λέγεται μοναδιαίος (unit) και η αντίστοιχη μοναδιαία πρόταση λέγεται αιτιολογική πρόταση (reason clause) του κατηγορήματος της μοναδιαίας πρότασης, όταν ανατεθεί τιμή σε αυτό.

Κατά τη λειτουργία αυτού του κανόνα, ο κόμβος στον οποίο βρίσκεται η αναζήτηση, δεν αποκτά δύο κόμβους-παιδιά, αλλά επεκτείνεται κατά έναν κόμβο, και δεν αυξάνεται το επίπεδο στο δέντρο αναζήτησης. Η ακμή μεταξύ του νέου κόμβου και του πατέρα του αντιπροσωπεύεται με το κατηγόρημα της μοναδιαίας πρότασης (unit clause) στο οποίο έγινε η τελευταία ανάθεση τιμής.

Με την εφαρμογή αυτού του κανόνα, για το κατηγόρημα L κάθε μοναδιαίας πρότασης, το οποίο παίρνει την τιμή TRUE, διαγράφονται από τη φόρμουλα F όλες οι προτάσεις που το περιέχουν, διότι ικανοποιούνται. Επίσης, διαγράφεται το κατηγόρημα ¬L, το οποίο παίρνει την τιμή FALSE, από όλες τις προτάσεις της φόρμουλας F που το περιέχουν, αφού η ικανοποιησιμότητά τους δεν εξαρτάται πλέον από εκείνο το κατηγόρημα.

Γραμμή 7: Αν η τρέχουσα ανάθεση τιμών στις μεταβλητές δημιουργεί μια κενή πρόταση (empty clause), σημαίνει ότι η φόρμουλα δεν ικανοποιείται με την τρέχουσα ανάθεση τιμών. Η κενή πρόταση λέγεται πρόταση σύγκρουσης (conflict clause). Στην περίπτωση αυτή, ο αλγόριθμος επιστρέφει UNSATISFIABLE (μη ικανοποιήσιμη).

Γραμμές 8-10: Σύμφωνα με τον κανόνα αυτό, γίνεται αναζήτηση για αγνά κατηγορήματα (pure literals), δηλαδή για μεταβλητές που εμφανίζονται μόνο με μία πολικότητα, και ανατίθεται σε αυτές η τιμή TRUE. Οι προτάσεις που περιέχουν αγνά κατηγορήματα διαγράφονται από τη φόρμουλα F, διότι ικανοποιούνται.

Γραμμή 11: Αν η φόρμουλα F είναι άδεια, δηλαδή δεν περιέχει προτάσεις που δεν ικανοποιήθηκαν, τότε ο αλγόριθμος επιστρέφει SATISFIABLE (ικανοποιήσιμη).

Γραμμές 12-15: Στο βήμα αυτό, ο αλγόριθμος επεκτείνει την ανάθεση κατά μια μεταβλητή και καλεί αναδρομικά τον αλγόριθμο με παράμετρο τη νέα ανάθεση. Αν προκύψει σύγκρουση και επιστραφεί UNSATISFIABLE, τότε γίνεται η λεγόμενη οπισθοχώρηση (backtracking) και η μεταβλητή παίρνει την αντίθετη τιμή (δηλαδή αν αρχικά πήρε την τιμή true και προέκυψε σύγκρουση, η μεταβλητή θα πάρει την τιμή

false). Ο κανόνας αυτός ονομάζεται κανόνας διαχωρισμού (split), διότι διαχωρίζει το δέντρο αναζήτησης και προσθέτει νέους κόμβους στο δέντρο αναζήτησης σε μεγαλύτερα επίπεδα.

4.4.3 Παράδειγμα εκτέλεσης του αλγόριθμου DPLL

Αρχική φόρμουλα:

F= <[x1+x2+x3], [x1+¬x2+x3], [x1+¬x2+¬x3], [¬x1+x2+x3], [¬x1+x2+¬x3], [¬x1+¬x2+¬x3], [¬x1+¬x2+x3] >

Πιο κάτω φαίνεται η εκτέλεση του αλγόριθμου DPLL για την εύρεση ανάθεσης τιμών στις μεταβλητές της φόρμουλας F που να ικανοποιεί την F, δηλαδή να κάνει την F αληθή (TRUE).

Φόρμουλα (επίπεδο 0):

F= <[x1+x2+x3], [x1+¬x2+x3], [x1+¬x2+¬x3], [¬x1+x2+x3], [¬x1+x2+¬x3], [¬x1+¬x2+¬x3], [¬x1+¬x2+x3] >

Ανάθεση 1: x1= True.

Η μεταβλητή x1 παίρνει την τιμή True. Οι προτάσεις που περιέχουν το κατηγόρημα x1 ικανοποιούνται και για το λόγο αυτό διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος ¬x1 διαγράφονται από τις προτάσεις, αφού το ¬x1 είναι false και, επομένως, η ικανοποιησιμότητα των προτάσεων που περιέχουν το ¬x1 εξαρτάται πλέον μόνο από τα υπόλοιπα κατηγορήματα των προτάσεων.

Φόρμουλα (επίπεδο 1):

F= <[x2+x3], [x2+¬x3], [¬x2+¬x3], [¬x2+x3] >

Ανάθεση 2: x2= True.

Όπως για την πρώτη ανάθεση, οι προτάσεις που περιέχουν το κατηγόρημα x2 ικανοποιούνται και για το λόγο αυτό διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος ¬x2 διαγράφονται από τις προτάσεις, αφού το ¬x2 είναι false και, επομένως, η ικανοποιησιμότητα των προτάσεων που περιέχουν το ¬x2 εξαρτάται πλέον μόνο από τα υπόλοιπα κατηγορήματα των προτάσεων.

Φόρμουλα (επίπεδο 2):

F= <[x3], [¬x3] > → Σύγκρουση

ΟΠΙΣΘΟΧΩΡΗΣΗ (πίσω στο επίπεδο 1)

Φόρμουλα (επίπεδο 1):

F= <[x2+x3], [x2+¬x3], [¬x2+¬x3], [¬x2+x3] >

Ανάθεση 3: x2= False.

Παρόμοια με τις πιο πάνω αναθέσεις, οι προτάσεις που περιέχουν το κατηγόρημα ¬x2 ικανοποιούνται και διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος x2 διαγράφονται από τις προτάσεις.

Φόρμουλα (επίπεδο 2):

F= <[x3], [¬x3] > → Σύγκρουση

ΟΠΙΣΘΟΧΩΡΗΣΗ (πίσω στο επίπεδο 0)

Φόρμουλα (επίπεδο 0):

F= <[x1+x2+x3], [x1+¬x2+x3], [x1+¬x2+¬x3], [¬x1+x2+x3], [¬x1+x2+¬x3], [¬x1+¬x2+¬x3], [¬x1+¬x2+x3] >

Ανάθεση 4: x1= False.

Παρόμοια με τις πιο πάνω αναθέσεις, , οι προτάσεις που περιέχουν το κατηγόρημα ¬x1 ικανοποιούνται και διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος x1 διαγράφονται από τις προτάσεις.

Φόρμουλα (επίπεδο 1):

F= <[x2+x3], [¬x2+x3], [¬x2+¬x3]] >

Ανάθεση 5: x2= True.

Σε αυτό το σημείο, οι προτάσεις που περιέχουν το κατηγόρημα x2 ικανοποιούνται και διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος ¬x2 διαγράφονται από τις προτάσεις.

Φόρμουλα (επίπεδο 2):

F= <[x3], [¬x3] > → Σύγκρουση

ΟΠΙΣΘΟΧΩΡΗΣΗ (πίσω στο επίπεδο 1)

Φόρμουλα (επίπεδο 1):

F= <[x2+x3], [¬x2+x3], [¬x2+¬x3]] >

Ανάθεση 6: x2= False.

Όπως πιο πάνω, οι προτάσεις που περιέχουν το κατηγόρημα ¬x2 ικανοποιούνται και διαγράφονται από τη φόρμουλα. Επίσης, οι εμφανίσεις του κατηγορήματος x2 διαγράφονται από τις προτάσεις.

Φόρμουλα (επίπεδο 2):

F= <[x3] >

Το κατηγόρημα x3 ανήκει σε μοναδιαία πρόταση (unit clause). Σύμφωνα με τον αλγόριθμο, το x3 θα πάρει την τιμή True και η λειτουργία του αλγορίθμου θα συνεχίσει αναδρομικά. Σε αυτή την αναδρομική κλήση, η φόρμουλα F θα είναι κενή και ο αλγόριθμος DPLL θα επιστρέψει SATISFIABLE.

Αποτέλεσμα:

Η φόρμουλα είναι ικανοποιήσιμη (SATISFIABLE). Η ανάθεση τιμών που εντοπίστηκε, η οποία κάνει τη φόρμουλα F αληθή, είναι x1=False, x2=False και x3=True.

Στο σημείο αυτό, αξίζει να σημειωθεί ότι ενώ τα κατηγορήματα που γίνονται ψευδή (false) θεωρητικά διαγράφονται από τις προτάσεις, όπως φαινεται στο πιο πάνω παράδειγμα, στην πραγματικότητα τα κατηγορήματα αυτά δε διαγράφονται, επειδή μπορεί να χρησιμοποιηθούν ξανά μετά από μικρό χρονικό διάστημα λόγω κάποιας οπισθοχώρησης, αλλά φαινομενικά μπαίνουν σε κατάσταση διαγραφής.

4.5 Αλγόριθμος CDCL

4.5.1 Εισαγωγή

Ο αλγόριθμος CDCL είναι μια επέκταση του αλγόριθμου DPLL και αναπτύχθηκε το 1962, από τους M. Davis, G. Logemann και D. Loveland. Χρησιμοποιείται στους σημερινούς, μοντέρνους προτασιακούς επιλυτές και οι τεχνικές που ενσωματώθηκαν στον αλγόριθμο αυτό συνέβαλαν στη δραματική βελτίωση των χρόνων επίλυσης των προβλημάτων ικανοποιησιμότητας από τους προτασιακούς επιλυτές.

Οι κυριότερες βελτιώσεις που έχουν γίνει στον αλγόριθμο CDCL είναι οι εξής:

1) Ο αλγόριθμος CDCL αξιοποιεί καλύτερα τη γνώση από τις συγκρούσεις (conflicts) που προκύπτουν κατά την αναζήτηση. Από μια σύγκρουση παράγεται η λεγόμενη «εκμαθημένη πρόταση» (learned clause), η οποία συμβάλλει στην αποφυγή παρόμοιων μονοπατιών που δεν οδηγούν σε λύση.

2) Η οπισθοδρόμηση είναι δυναμική. Δηλαδή, μετά από μια σύγκρουση, η αναζήτηση δεν επιστρέφει απαραίτητα στο αμέσως προηγούμενο επίπεδο, αλλά μπορεί να γίνει οπισθοδρόμηση σε πολύ μικρότερο επίπεδο, ακυρώνοντας πολλές αποφάσεις. Έτσι, η αναζήτηση διαφεύγει γρηγορότερα από «κλαδιά» του δέντρου αναζήτησης που δεν οδηγούν σε λύση.

3) Επειδή η οπισθοδρόμηση είναι δυναμική, η σειρά με την οποία επιλέγονται οι μεταβλητές απόφασης (decision variables) μπορεί να αλλάζει σε κάθε μονοπάτι. Αντίθετα, στον αλγόριθμο DPLL, η σειρά με την οποία επιλέγονταν οι μεταβλητές ήταν η ίδια σε όλα τα μονοπάτια. Η αλλαγή της σειράς με την οποία επιλέγονται οι μεταβλητές έχει μεγάλη σημασία, διότι κάποιες αποφάσεις μπορούν να δημιουργήσουν περισσότερες μοναδιαίες προτάσεις (unit clauses), μειώνοντας με αυτόν τον τρόπο το χώρο αναζήτησης, και να λύσουν γρηγορότερα τα προβλήματα.

4.5.2 Περιγραφή του αλγόριθμου CDCL

Πιο κάτω φαίνεται ο αλγόριθμος CDCL και ακολουθεί μια αναλυτική περιγραφή του.

Αλγόριθμος CDCL

```
1.  Algorithm CDCL( ClauseSet S )
2.  {
3.      while(1){
4.          while (S contains a unit clause {L}) {
5.              delete from S all clauses containing L;
6.              delete ¬L from all clauses in S;
7.          }
8.          while (S contains a pure literal L ) {
9.              delete from S all clauses containing L;
10.         }
11.         if (S contains no conflicting clause){
12.             if (all clauses of S are satisfied) return SATISFIABLE;
13.             l := choose_literal(S);
14.             assign(S,l);
15.         }
16.         else{
17.             if (conflict at current_level==0) return UNSATISFIABLE;
18.             analyze_conflict();
19.             undo_assignments();
20.             S := add_conflict_clause(S);
21.         }
22.     }
23. }
```

Περιγραφή του αλγόριθμου CDCL

Κατά την έναρξη της εκτέλεσης του αλγορίθμου, δεν έχει γίνει ακόμα καμία ανάθεση τιμών, απόφαση (decision), ή σύγκρουση (conflict), και βρισκόμαστε στη ρίζα του δέντρου αναζήτησης, η οποία είναι στο επίπεδο 0.

Στις γραμμές 3-22 εκτελείται ένας βρόγχος while. Ακολουθεί αναλυτικότερη περιγραφή του βρόγχου.

Γραμμές 4-7: Η γραμμές αυτές αντιστοιχούν στον μοναδιαίο κανόνα του αλγόριθμου DPLL. Στον αλγόριθμο CDCL, η διαδικασία αυτή λέγεται Μοναδιαία Μετάδοση (Unit

Propagation). Κατά τη διαδικασία αυτή, γίνεται ανάθεση τιμών στα κατηγορήματα όλων των μοναδιαίων προτάσεων (unit clauses) της φόρμουλας. Τα κατηγορήματα αυτά ανήκουν στο επίπεδο (level), στο οποίο έγινε η ανάθεση τους. Επίσης, η μοναδιαία πρόταση στην οποία ανήκουν τίθεται ως αιτιολογική πρόταση (reason clause).

Τα κατηγορήματα που παίρνουν τιμή αποθηκεύονται σε μια δομή που λέγεται ατραπός (trail) και έχει τη μορφή στοίβας. Στη δομή trail, τα κατηγορήματα στα οποία έχει ανατεθεί τιμή, είναι αποθηκευμένα με τη σειρά με την οποία τους έχει ανατεθεί τιμή (όσο πιο πρόσφατα έχει πάρει τιμή ένα κατηγόρημα, τόσο πιο κοντά στην κορυφή της στοίβας θα βρίσκεται). Στον πιο πάνω αλγόριθμο, για σκοπούς απλοποίησης, δεν αναφέρουμε τις δομές που χρησιμοποιούνται. Συνήθως η δομή trail υλοποιείται ξεχωριστά και γίνονται πολλές βελτιώσεις σε αυτήν, ώστε η πρόσβαση στα κατηγορήματα που περιέχει να είναι όσο πιο γρήγορη γίνεται.

Αν προκύψει σύγκρουση κατά την ανάθεση τιμής στο κατηγόρημα κάποιας μοναδιαίας πρότασης, ο προτασιακός επιλυτής θυμάται τη σύγκρουση, για να μπορεί αργότερα (από τη γραμμή 11 και μετά) να κάνει τους σχετικούς ελέγχους και την ανάλυση της σύγκρουσης.

Γραμμές 8-10: Σύμφωνα με τον κανόνα αυτό, γίνεται αναζήτηση για αγνά κατηγορήματα (pure literals), δηλαδή για μεταβλητές που εμφανίζονται μόνο με μία πολικότητα και, όπως ακριβώς στον αλγόριθμο DPLL, ανατίθεται σε αυτές η τιμή TRUE. Οι προτάσεις που περιέχουν αγνά κατηγορήματα διαγράφονται από τη φόρμουλα F, διότι ικανοποιούνται.

Οι γραμμές 11-15 αφορούν την περίπτωση που δεν υπάρχει σύγκρουση κατά τη διαδικασία μοναδιαίας μετάδοσης, ενώ οι γραμμές 16-21 αφορούν την περίπτωση που υπάρχει σύγκρουση κατά τη διαδικασία μοναδιαίας μετάδοσης.

Γραμμές 11-12: Στην περίπτωση που δε βρεθεί σύγκρουση κατά τη διαδικασία της μοναδιαίας μετάδοσης και έχουν ικανοποιηθεί όλες οι προτάσεις, τότε ο αλγόριθμος επιστρέφει SATISFIABLE.

Γραμμές 13-15: Στην περίπτωση που δε βρεθεί σύγκρουση κατά τη διαδικασία της μοναδιαίας μετάδοσης αλλά δεν έχουν ικανοποιηθεί ακόμη όλες οι προτάσεις, τότε επιλέγεται μια μεταβλητή, στην οποία δεν έχει ανατεθεί ακόμα τιμή, ως μεταβλητή απόφασης μέσω της διαδικασίας choose_literal. Επίσης, επιλέγεται τιμή για τη μεταβλητή απόφασης. Υπάρχουν διάφορα ευρετικά που μπορούν να χρησιμοποιηθούν σε αυτή τη διαδικασία. Ένα από τα πιο γνωστά μοντέρνα ευρετικά είναι το ευρετικό VSIDS, μια παραλλαγή του οποίου χρησιμοποιεί ο προτασιακός επιλυτής PrecoSAT (η εκτενέστερη ανάλυση των ευρετικών του PrecoSAT υπάρχει στο επόμενο κεφάλαιο).

Η μεταβλητή απόφασης που επιλέγεται, δημιουργεί δύο νέα μονοπάτια στον κόμβο όπου βρίσκεται η αναζήτηση. Ένα χαρακτηριστικό των μεταβλητών απόφασης είναι ότι δεν έχουν αιτιολογική πρόταση, επειδή η ανάθεσή τους δεν προέκυψε από την εμφάνισή τους σε μοναδιαίες προτάσεις. Μετά την επιλογή της μεταβλητής απόφασης, το επίπεδο του δέντρου αναζήτησης αυξάνεται κατά ένα και η μεταβλητή απόφασης προστίθεται στην τρέχουσα ανάθεση. Επίσης, το επίπεδο στο οποίο ανήκει η μεταβλητή απόφασης ισούται με το τρέχον επίπεδο του δέντρου αναζήτησης (current level). Μετά την εκτέλεση της γραμμής 14, ο αλγόριθμος επιστρέφει στην αρχή του βρόγχου while (γραμμή 3) και η διαδικασία επαναλαμβάνεται.

Στην περίπτωση που υπάρξει σύγκρουση κατά τη διαδικασία της μοναδιαίας μετάδοσης, δεν εκτελούνται οι γραμμές 11-15 και η εκτέλεση του προγράμματος συνεχίζει στη γραμμή 16.

Γραμμή 17: Αν υπάρξει σύγκρουση κατά τη διαδικασία μοναδιαίας μετάδοσης και η αναζήτηση βρίσκεται στο επίπεδο 0, δηλαδή στη ρίζα του δέντρου αναζήτησης, τότε δεν μπορεί να γίνει οποιουδήποτε είδους οπισθοδρόμηση. Στην περίπτωση αυτή, η φόρμουλα δεν μπορεί να ικανοποιηθεί και ο αλγόριθμος επιστρέφει UNSATISFIABLE (μη ικανοποιήσιμη).

Γραμμή 18: Στη γραμμή 18 γίνεται η διαδικασία ανάλυσης της σύγκρουσης (conflict analysis), η οποία έχει ως σκοπό την παραγωγή της εκμαθημένης πρότασης (learned clause). Μια ιδιότητα της εκμαθημένης πρότασης (learned clause) είναι ότι περιέχει μόνο ένα κατηγόρημα από το τρέχον επίπεδο. Η εκμαθημένη πρόταση δημιουργείται με

τρόπο που να υπάρχει ένα κομμάτι της τρέχουσας μερικής ανάθεσης το οποίο την κάνει μοναδιαία πρόταση (unit clause). Αυτό μπορεί να γίνει, όταν όλα τα κατηγορήματα εκτός ενός, που υπάρχουν στην εκμαθημένη πρόταση, γίνονται ψευδή σε προηγούμενα επίπεδα και μόνο ένα κατηγόρημα παίρνει τιμή στο τελευταίο επίπεδο. Στις υλοποιήσεις των προτασιακών επιλυτών γίνεται μεγάλη προσπάθεια για τη βελτιστοποίηση αυτής της διαδικασίας.

Γραμμή 19: Στη γραμμή αυτή, εκτελείται η διαδικασία δυναμικής οπισθοδρόμησης (backjumping) και η αναζήτηση επιστρέφει στο επίπεδο όπου η εκμαθημένη πρόταση γίνεται μοναδιαία πρόταση (όχι απαραίτητα το αμέσως προηγούμενο επίπεδο). Οι αναθέσεις των μεταβλητών που έγιναν μετά από εκείνο το επίπεδο, καθώς και οι πληροφορίες για τις αιτιολογικές προτάσεις, διαγράφονται.

Γραμμή 20: Η εκμαθημένη πρόταση προστίθεται στη φόρμουλα. Στο σημείο αυτό, ο αλγόριθμος επιστρέφει στην αρχή του βρόγχου while (γραμμή 3) και η διαδικασία επαναλαμβάνεται.

4.6 Τεχνικές των μοντέρνων προτασιακών επιλυτών

4.6.1 Εισαγωγή

Στο σημείο αυτό, θα γίνει μια σύντομη αναφορά στις τεχνικές που χρησιμοποιούνται στους μοντέρνους προτασιακούς επιλυτές.

Η λειτουργίες των κομματιών από τα οποία αποτελείται ο αλγόριθμος CDCL είναι περίπου οι ίδιες σε όλους τους μοντέρνους προτασιακούς επιλυτές. Οι διαφορές που υπάρχουν μεταξύ των προτασιακών επιλυτών αφορούν κυρίως τα ευρετικά, τις δομές δεδομένων και τις τεχνικές υλοποίησης που χρησιμοποιούν.

Οι τεχνικές που έχουν αναπτυχθεί τα τελευταία χρόνια και έχουν μειώσει δραματικά το χρόνο επίλυσης των προβλημάτων ικανοποιησιμότητας μπορούν να διαχωριστούν ανάλογα με το κομμάτι του αλγορίθμου CDCL στο οποίο ανήκουν, όπως αυτά περιγράφονται πιο κάτω.

4.6.2 Προεπεξεργαστής

Το πρώτο κομμάτι από το οποίο αποτελείται ο αλγόριθμος CDCL είναι ο Προεπεξεργαστής (Preprocessor). Ο Προεπεξεργαστής είναι υπεύθυνος για την απλοποίηση της αρχικής φόρμουλας στο ξεκίνημα της λειτουργίας του προτασιακού επιλυτή, πριν από την έναρξη της αναζήτησης.

Η λειτουργία του Προεπεξεργαστή είναι σημαντική, διότι γίνεται αφαίρεση μεταβλητών και προτάσεων που δεν είναι απαραίτητες για την επίλυση του προβλήματος, με αποτέλεσμα τη μείωση του συνολικού αριθμού μεταβλητών και προτάσεων της φόρμουλας, συνεπώς και το χρόνο επίλυσης. Παρόλο που η λειτουργία του Προεπεξεργαστή γενικά μειώνει το χρόνο επίλυσης, είναι πολύ σημαντικό να υπάρχει ισορροπία μεταξύ του χρόνου επίλυσης που αφιερώνεται στον Προεπεξεργαστή και του χρόνου επίλυσης που αφιερώνεται στα υπόλοιπα μέρη του αλγορίθμου, διότι η υπερβολική προεπεξεργασία μπορεί τελικά να έχει αρνητικές συνέπειες στο χρόνο επίλυσης του προβλήματος.

4.6.3 Μοναδιαία Μετάδοση

Το δεύτερο κομμάτι του αλγορίθμου CDCL είναι η Μοναδιαία Μετάδοση (Unit Propagation). Στο κομμάτι αυτό αφιερώνεται το μεγαλύτερο ποσοστό χρόνου για την επίλυση του προβλήματος. Ακριβώς γι' αυτό το λόγο είναι πολύ σημαντική η βελτιστοποίηση του κώδικα, των δομών δεδομένων και των τεχνικών που χρησιμοποιούνται σε αυτό το κομμάτι.

Σκοπός της Μοναδιαίας Μετάδοσης είναι η εύρεση όλων των μοναδιαίων προτάσεων που προέκυψαν με την τρέχουσα ανάθεση τιμών και η πρόσθεση του κατηγορήματος κάθε μοναδιαίας πρότασης στην τρέχουσα ανάθεση. Εάν εντοπιστεί σύγκρουση κατά τη διάρκεια εκτέλεσης αυτής της διαδικασίας, η διαδικασία σταματά και επιστρέφει την πρόταση που προκάλεσε τη σύγκρουση (conflict clause). Διαφορετικά, δηλαδή όσο δεν υπάρχει σύγκρουση, και συνεχίζουν να υπάρχουν μοναδιαίες προτάσεις στη φόρμουλα, η διαδικασία επαναλαμβάνεται. Όσο περισσότερες προτάσεις έχει μια φόρμουλα, τόσο πιο χρονοβόρα γίνεται η διαδικασία της Μοναδιαίας Μετάδοσης.

Σε αυτό το κομμάτι, είναι πολύ σημαντικός ο γρήγορος εντοπισμός των μοναδιαίων προτάσεων, διότι η διαδικασία της Μοναδιαίας Μετάδοσης μπορεί να εκτελεστεί εκατομμύρια φορές με αποτέλεσμα μια μικρή βελτίωση να μπορεί στην πραγματικότητα να βελτιώσει δραματικά το συνολικό χρόνο. Μια από τις πιο διάσημες μοντέρνες τεχνικές για τη διαδικασία της Μοναδιαίας Μετάδοσης είναι η τεχνική Two-Watched-Literal, η οποία προσπαθεί να εντοπίσει πολύ γρήγορα τις μοναδιαίες προτάσεις χρησιμοποιώντας ειδικά διαμορφωμένες δομές δεδομένων.

4.6.4 Ευρετικό απόφασης

Το τρίτο κομμάτι του αλγόριθμου CDCL είναι το Ευρετικό απόφασης (Decision Heuristic).

Το Ευρετικό Απόφασης εφαρμόζεται, όταν δεν μπορεί να εφαρμοστεί κανένας άλλος κανόνας του αλγόριθμου CDCL (δηλαδή όταν δεν υπάρχουν άλλες μοναδιαίες προτάσεις ή αγνά κατηγορήματα). Στην περίπτωση αυτή, ο επιλυτής πρέπει αναγκαστικά να επιλέξει μια μη ανατεθειμένη μεταβλητή, με την οποία θα συνεχίσει η αναζήτηση.

Μέσω του Ευρετικού Απόφασης πρέπει να παρθούν δύο αποφάσεις: Πρέπει να αποφασιστεί η μεταβλητή (η οποία δεν πρέπει να έχει ήδη πάρει τιμή) στην οποία θα ανατεθεί τιμή (variable selection heuristic) και πρέπει να αποφασιστεί η τιμή (polarity) που θα δοθεί στη μεταβλητή (phase selection heuristic).

Υπάρχουν διάφορες τεχνικές που χρησιμοποιούνται στο Ευρετικό Απόφασης. Για παράδειγμα, μία τεχνική που χρησιμοποιείται κυρίως για σκοπούς σύγκρισης είναι η τυχαία επιλογή μεταβλητής απόφασης και η τυχαία επιλογή της τιμής της. Μία άλλη, πιο αποδοτική, τεχνική επιλογής μεταβλητής είναι η τεχνική VSIDS (Variable State Independent Decaying Sum), σύμφωνα με την οποία επιλέγεται η μεταβλητή που πήρε μέρος στις περισσότερες και πιο πρόσφατες συγκρούσεις [2]. Μια άλλη γνωστή τεχνική επιλογής μεταβλητής ή τιμής είναι η τεχνική Jeroslow-Wang [6, 8], σύμφωνα με την οποία επιλέγουμε τη μεταβλητή/κατηγόρημα με το μεγαλύτερο βάρος. Το βάρος κάθε κατηγορήματος είναι ανάλογο του αριθμού εμφανίσεών του στις προτάσεις της

φόρμουλας και αντιστρόφως ανάλογο του μεγέθους εκείνων των προτάσεων. Δηλαδή, όσο περισσότερες και μικρότερες είναι οι προτάσεις στις οποίες συμμετέχει το κατηγόρημα, τόσο μεγαλύτερο θα είναι το βάρος του. Η λογική που κρύβεται πίσω από αυτή την τεχνική είναι ότι οι μικρότερες προτάσεις «κόβουν» μεγαλύτερο κομμάτι του δέντρου αναζήτησης από τις μεγαλύτερες προτάσεις και, επομένως, θεωρούνται πιο σημαντικές.

Το Ευρετικό Απόφασης είναι πολύ σημαντικό, διότι καθορίζει το «κλαδί» του δέντρου αναζήτησης που θα εξεταστεί. Όταν ο επιλυτής παίρνει καλές αποφάσεις, τότε η αναζήτηση οδηγείται γρήγορα σε κάποιο μονοπάτι που αντιστοιχεί στη λύση του προβλήματος ή, σε περίπτωση που δεν υπάρχει ανάθεση που ικανοποιεί τη φόρμουλα, αποδεικνύεται με αποδοτικό τρόπο ότι η φόρμουλα είναι μη ικανοποιήσιμη. Από την άλλη, όταν ο επιλυτής αναζητεί τη λύση σε «κλαδιά» του δέντρου αναζήτησης που δεν οδηγούν σε λύση, τότε οι συνέπειες είναι αρνητικές, αφού ο χρόνος επίλυσης αυξάνεται. Η τεχνική (ή ο συνδυασμός τεχνικών) που εφαρμόζεται για το Ευρετικό Απόφασης του προτασιακού επιλυτή, καθώς και οι δομές δεδομένων που χρησιμοποιούνται, καθορίζουν την αποδοτικότητα του Ευρετικού Απόφασης.

4.6.5 Ανάλυση σύγκρουσης

Το τέταρτο κομμάτι του αλγόριθμου CDCL είναι η Ανάλυση σύγκρουσης (Conflict Analysis). Η Ανάλυση σύγκρουσης αποτελεί μία από τις σημαντικότερες διαφορές των αλγόριθμων DPLL και CDCL και είναι ένας από κυριότερους λόγους επιτυχίας του αλγόριθμου CDCL.

Σκοπός της φάσης αυτής είναι η ανάλυση της σύγκρουσης που προέκυψε κατά τη Μοναδιαία Μετάδοση (Unit Propagation) και η παραγωγή της εκμαθημένης πρότασης (learned clause). Η εκμαθημένη πρόταση παράγεται μέσω προτασιακής επίλυσης (resolution) της πρότασης σύγκρουσης (conflict clause) και των αιτιολογικών προτάσεων (reason clauses) των κατηγορημάτων της. Μετά τη δημιουργία της, η εκμαθημένη πρόταση προστίθεται στη φόρμουλα και συμβάλλει στην αποφυγή παρόμοιων – αποτυχημένων – μονοπατιών κατά τη διάρκεια της αναζήτησης.

Αν στο Ευρετικό Απόφασης χρησιμοποιείται η τεχνική VSIDS (Variable State Independent Decaying Sum) τότε, κατά την Ανάλυση Σύγκρουσης, γίνεται και η ανανέωση του σκορ σχετικά με τη συμμετοχή των κατηγορημάτων σε πρόσφατες συγκρούσεις.

Με την Ανάλυση Σύγκρουσης, ο χρόνος επίλυσης μειώνεται δραματικά λόγω της δυναμικής οπισθοδρόμησης (backjumping) σε προηγούμενο επίπεδο, όπου, συνήθως, η εκμαθημένη πρόταση γίνεται μοναδιαία πρόταση (unit clause) και ο αλγόριθμος συνεχίζει εκτελώντας τον κανόνα της Μοναδιαίας Μετάδοσης (Unit Propagation).

4.6.6 Διαγραφή εκμαθημένων προτάσεων

Το πέμπτο κομμάτι του αλγορίθμου CDCL είναι η Διαγραφή εκμαθημένων προτάσεων (Removal). Σκοπός αυτού του κομματιού είναι η διαγραφή εκμαθημένων προτάσεων (learned clauses) που προστέθηκαν στη φόρμουλα σε παλαιότερο σημείο της αναζήτησης, κατά τη διαδικασία Ανάλυσης Σύγκρουσης (Conflict Analysis), αλλά φαίνεται ότι δε χρησιμοποιούνται πια.

Οι εκμαθημένες προτάσεις που δε χρησιμοποιούνται είναι σημαντικό να διαγράφονται, διότι προκαλούν καθυστερήσεις σε άλλες διαδικασίες (κυρίως στη διαδικασία Μοναδιαίας Μετάδοσης). Συνήθως, οι εκμαθημένες προτάσεις μεγέθους δύο δε διαγράφονται, διότι «κόβουν» μεγάλα κλαδιά του δέντρου αναζήτησης και δεν είναι χρονοβόρα η διαδικασία της Μοναδιαίας Μετάδοσης σε προτάσεις που αποτελούνται από μόνο δύο κατηγορήματα. Επίσης, δε διαγράφονται εκμαθημένες προτάσεις που είναι αιτιολογικές προτάσεις μεταβλητών στις οποίες έχει ανατεθεί τιμή, διότι η διαγραφή τέτοιων προτάσεων οδηγεί σε προβλήματα στη λειτουργία της Ανάλυσης Σύγκρουσης.

Από την άλλη όμως, η διαγραφή των εκμαθημένων προτάσεων επιτρέπει στην αναζήτηση να επισκεφτεί μονοπάτια τα οποία ήδη επισκέφτηκε και να οδηγηθεί ξανά στις ίδιες συγκρούσεις, χάνοντας πολύτιμο χρόνο. Επίσης, με τη διαγραφή χρήσιμων εκμαθημένων προτάσεων, ελλοχεύει ο κίνδυνος μη τερματισμού του αλγορίθμου. Τέλος, επειδή η Ανάλυση Σύγκρουσης και η παραγωγή της εκμαθημένης πρότασης

απαιτεί χρόνο, αν το θετικό αποτέλεσμα που θα είχαν, δηλαδή η αποφυγή αποτυχημένων μονοπατιών που ήδη επισκέφθηκε ο αλγόριθμος, ακυρωθεί, τότε ο χρόνος επίλυσης τελικά θα είναι χειρότερος από το χρόνο επίλυσης του προβλήματος χωρίς να γίνεται καμία επεξεργασία των συγκρούσεων.

Επομένως, χρειάζεται ιδιαίτερη προσοχή στην εύρεση των χρονικών στιγμών που θα γίνεται η διαγραφή των εκμαθημένων προτάσεων οι οποίες φαίνεται ότι δε χρησιμοποιούνται, αλλά και στα κριτήρια που καθορίζουν ποιες εκμαθημένες προτάσεις θα διαγραφούν.

4.6.7 Επανεκκίνηση

Το έκτο και τελευταίο κομμάτι του αλγορίθμου CDCL είναι η Επανεκκίνηση (Restart).

Σκοπός των επανεκκινήσεων είναι η ακύρωση όλων των αναθέσεων τιμών στις μεταβλητές και η επανεκκίνηση της αναζήτησης. Η αναζήτηση ξαναρχίζει από το επίπεδο μηδέν. Μετά την επανεκκίνηση, επειδή οι μεταβλητές που επιλέγονται ως μεταβλητές απόφασης θα επιλεγούν με διαφορετική σειρά, θα δημιουργηθεί ένα δέντρο αναζήτησης που πιθανότατα θα είναι διαφορετικό από τα προηγούμενα. Συνεπώς, οι επανεκκινήσεις βοηθούν την αναζήτηση να δραπετεύσει από μονοπάτια από τα οποία είναι δύσκολο να ξεφύγει χρησιμοποιώντας μόνο τη δυναμική οπισθοχώρηση. Αν μετά από μια επανεκκίνηση προκύψει μονοπάτι παρόμοιο με τα προηγούμενα, επειδή η ανάθεση των μεταβλητών θα γίνει με διαφορετική σειρά, η εκμαθημένη πρόταση που θα προκύψει σε περίπτωση σύγκρουσης θα είναι διαφορετική και θα οδηγήσει σε διαφορετικό μονοπάτι.

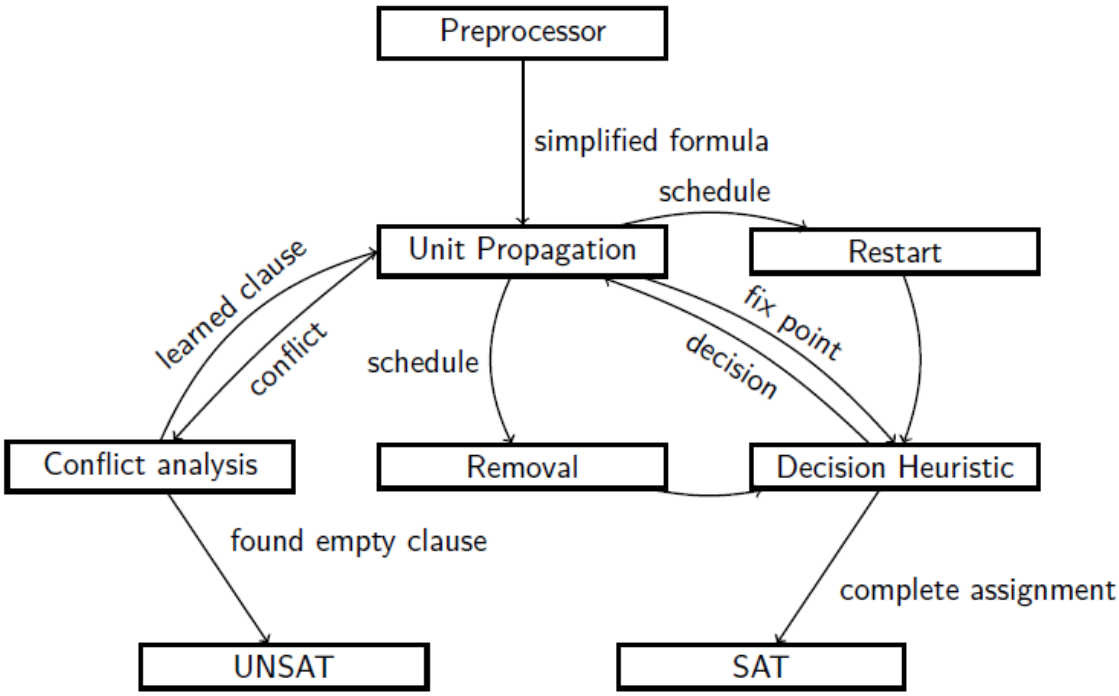
Υπάρχουν διαφορετικές τεχνικές καθορισμού των χρονικών στιγμών στις οποίες θα γίνουν οι επανεκκινήσεις. Συνήθως, οι τεχνικές αυτές αποφασίζουν πότε θα γίνει η επόμενη επανεκκίνηση χρησιμοποιώντας τον αριθμό συγκρούσεων και αποφάσεων και αυξάνουν δυναμικά το διάστημα μεταξύ της προτελευταίας και τελευταίας επανεκκίνησης (dynamic scheduling) [1].

Είναι σημαντικό οι επανεκκινήσεις να γίνονται την κατάλληλη στιγμή. Δεν πρέπει να γίνονται περισσότερες επανεκκινήσεις από όσο είναι απαραίτητο, διότι αν γίνονται

πολύ συχνά τότε ακυρώνουμε τη δουλειά που έκανε ο επιλυτής, παρόλο που μπορεί να πλησίασε υπερβολικά στην εξεύρεση λύσης, και τον υποχρεώνουμε να ξαναρχίσει από την αρχή, αυξάνοντας το χρόνο επίλυσης. Από την άλλη όμως, αν δε γίνονται οι επανεκκινήσεις αρκετά συχνά τότε ο επιλυτής κινδυνεύει να χαθεί σε μονοπάτια που δεν έχουν λύση και να σπαταλήσει άσκοπα χρόνο. Πειραματικά, φαίνεται ότι οι προτασιακοί επιλυτές στους οποίους υπάρχει η λειτουργία της Επανεκκίνησης αποδίδουν καλύτερα [12].

4.6.8 Αλληλεπιδράσεις μεταξύ των κομματιών ενός προτασιακού επιλυτή

Στο πιο κάτω σχήμα φαίνονται τα μέρη από τα οποία αποτελείται ένας μοντέρνος προτασιακός επιλυτής, καθώς και οι αλληλεπιδράσεις μεταξύ τους.



Σχήμα 4.2 Τα μέρη ενός προτασιακού επιλυτή και οι αλληλεπιδράσεις μεταξύ τους [12].

Όπως φαίνεται στο σχήμα 4.2, πριν την έναρξη της αναζήτησης, εκτελείται ο Προεπεξεργαστής, ο οποίος απλοποιεί τη φόρμουλα (simplified formula). Στη συνέχεια, εκτελείται η διαδικασία της Μοναδιαίας Μετάδοσης (Unit Propagation), η οποία συνεχίζεται με το Ευρετικό Απόφασης (Decision Heuristic) ή με Επανεκκίνηση (Restart) ή με Διαγραφή των εκμαθημένων προτάσεων (Removal). Οι τελευταίες δύο συνεχίζονται με το Ευρετικό Απόφασης. Μετά το Ευρετικό απόφασης, αν έχει βρεθεί μια ανάθεση που ικανοποιεί τη φόρμουλα, ο αλγόριθμος τερματίζει και επιστρέφει SAT (satisfiable). Διαφορετικά, ο αλγόριθμος συνεχίζει με τη διαδικασία της Μοναδιαίας Μετάδοσης. Αν υπάρξει σύγκρουση κατά τη Μοναδιαία Μετάδοση, γίνεται Ανάλυση της σύγκρουσης (Conflict analysis), παράγεται η εκμαθημένη πρόταση (learned clause) και ο αλγόριθμος συνεχίζει με τη διαδικασία της Μοναδιαίας Μετάδοσης. Αν βρεθεί κενή πρόταση (empty clause), δηλαδή πρόταση χωρίς κατηγορήματα, τότε ο αλγόριθμος τερματίζει και επιστρέφει UNSAT (unsatisfiable).

Στο επόμενο κεφάλαιο θα επικεντρωθούμε στην εκτενέστερη ανάλυση των ευρετικών απόφασης που χρησιμοποιούνται στον προτασιακό επιλυτή PrecoSAT.

Κεφάλαιο 5

Ο προτασιακός επιλυτής PrecoSAT

5.1 Εισαγωγή 42
5.2 Μορφοποίηση DIMACS 43
5.3 Ευρετικό απόφασης του PrecoSAT 44
5.4 Ευρετικό για την επιλογή μεταβλητής απόφασης 44
5.4.1 Δομές δεδομένων 45
5.4.2 Αλγόριθμοι θερμότητας 47
5.4.2.1 Αλγόριθμος αρχικοποίησης θερμότητας 48
5.4.2.2 Αλγόριθμος υπολογισμού θερμότητας 48
5.5 Ευρετικό για την επιλογή τιμής της μεταβλητής απόφασης 50
5.5.1 Δομές δεδομένων 50
5.5.2 Ευρετικό Jeroslow-Wang 51
5.5.3 Ευρετικό Rsat 52
5.6 Σύνοψη 52

5.1 Εισαγωγή

Ο PrecoSAT είναι ένας σύγχρονος προτασιακός επιλυτής και ενσωματώνει μοντέρνες τεχνικές για να λύσει Προβλήματα Ικανοποιησιμότητας. Δηλαδή, προσπαθεί να βρει κάποια ανάθεση τιμών στις μεταβλητές μιας φόρμουλας (CNF theory/ formula) που δέχεται ως είσοδο, έτσι ώστε η φόρμουλα να ικανοποιείται. Στην παρούσα διπλωματική χρησιμοποιείται η νεότερη έκδοση του προτασιακού επιλυτή PrecoSAT (έκδοση 570), ο οποίος πήρε το χρυσό μετάλλιο στο διαγωνισμό SAT 2009 (έκδοση 236) και είναι ένας από τους καλύτερους αντιπροσώπους των μοντέρνων προτασιακών επιλυτών [3]. Ο κώδικας του προτασιακού επιλυτή PrecoSAT είναι γραμμένος στη γλώσσα προγραμματισμού C++.

Οι τεχνικές που χρησιμοποιεί ο PrecoSAT στοχεύουν στη βελτίωση του χρόνου επίλυσης όλων των ειδών προβλημάτων. Με άλλα λόγια, δεν είναι εξειδικευμένες τεχνικές για συγκεκριμένα είδη προβλημάτων, όπως είναι τα προβλήματα Προγραμματισμού Δράσης (Planning). Για σκοπούς της παρούσας διπλωματικής, επικεντρωθήκαμε στην τροποποίηση του Ευρετικού Απόφασης του PrecoSAT με στόχο την αξιοποίηση κάποιων χαρακτηριστικών ιδιοτήτων των προβλημάτων Προγραμματισμού Δράσης για την αύξηση της αποδοτικότητας του επιλυτή στην επίλυση αυτών των προβλημάτων.

5.2 Μορφοποίηση DIMACS

Η φόρμουλα που δίνεται ως είσοδος στον προτασιακό επιλυτή PrecoSAT πρέπει να είναι σε Συζευκτική Κανονική Μορφή (CNF) και το αρχείο που την περιέχει πρέπει να ακολουθεί τη μορφοποίηση DIMACS (DIMACS format).

Παράδειγμα CNF φόρμουλας σε μορφοποίηση DIMACS:

```
c
c this is a comment
c
p cnf 5 4
1 2 0
-1 5 3 4 0
-3 -4 0
2 0
```

Το αρχείο που περιέχει τη φόρμουλα μπορεί να αρχίζει με σχόλια. Οι γραμμές με σχόλια πρέπει να αρχίζουν με τον χαρακτήρα “c”.

Αμέσως μετά τα σχόλια, πρέπει να υπάρχει μια γραμμή της μορφής “p cnf <ακέραιος_1> <ακέραιος_2> ”, όπου ο <ακέραιος_1> είναι ο συνολικός αριθμός των μεταβλητών της φόρμουλας και ο <ακέραιος_2> είναι ο συνολικός αριθμός προτάσεων της φόρμουλας.

Στις επόμενες γραμμές, ακολουθούν οι προτάσεις της φόρμουλας. Κάθε γραμμή εκπροσωπεί μια πρόταση και είναι μια ακολουθία διαφορετικών, μη μηδενικών ακεραίων στο διάστημα [– <ακέραιος_1>, <ακέραιος_1>], χωρισμένων με το χαρακτήρα SPACE. Ο αριθμός 0 στο τέλος μιας γραμμής συμβολίζει το τέλος της πρότασης. Στις προτάσεις δε γίνεται να υπάρχουν αντίθετα κατηγορήματα (πχ. 4 και –4). Οι θετικοί αριθμοί εκπροσωπούν τις αντίστοιχες μεταβλητές, ενώ οι αρνητικοί αριθμοί εκπροσωπούν τις αρνήσεις των αντίστοιχων μεταβλητών.

5.3 Ευρετικό απόφασης του PrecoSAT

Σε αυτό το υποκεφάλαιο θα γίνει μια αναλυτική περιγραφή του Ευρετικού Απόφασης (Decision Heuristic) που χρησιμοποιεί ο προτασιακός επιλυτής PrecoSAT. Το ευρετικό απόφασης του PrecoSAT χωρίζεται σε δύο μέρη:

- Ευρετικό επιλογής μεταβλητής απόφασης (υποκεφάλαιο 5.4)
- Ευρετικό επιλογής τιμής για τη μεταβλητή απόφασης (υποκεφάλαιο 5.5)

Ακολουθεί εκτενής περιγραφή των δύο πιο πάνω ευρετικών.

5.4 Ευρετικό για την επιλογή μεταβλητής απόφασης

Για την επιλογή μεταβλητής απόφασης (variable selection), ο PrecoSAT χρησιμοποιεί μια παραλλαγή του ευρετικού VSIDS (Variable State Independent Decaying Sum). Στόχος του ευρετικού επιλογής μεταβλητής απόφασης που χρησιμοποιεί ο PrecoSAT είναι η επιλογή της θερμότερης μη ανατεθειμένης μεταβλητής.

Σύμφωνα με την περιγραφή του PrecoSAT [4]: “For the decision heuristic we use a low-pass filter on the number of times a variable is involved in producing a conflict, implemented as an infinite response filter of order 3”. Η πρόταση αυτή θα επεξηγηθεί σταδιακά στα υποκεφάλαια που ακολουθούν.

5.4.1 Δομές δεδομένων

Οι υποψήφιες μεταβλητές απόφασης (decision variables), σύμφωνα με τον αλγόριθμο του PrecoSAT, είναι αποθηκευμένες σε ειδικά προσαρμοσμένη δομή, ώστε η πρόσβαση σε αυτές να είναι εύκολη και γρήγορη.

Η δομή δεδομένων που χρησιμοποιεί ο PrecoSAT για να αποθηκεύει τις υποψήφιες μεταβλητές απόφασης είναι η schedule.decide, η οποία έχει τύπο Heap<Rnk, Hotter>. Η schedule.decide περιέχει τις υποψήφιες μεταβλητές απόφασης ταξινομημένες βάσει της θερμότητάς τους (heat), η οποία εξαρτάται από τον αριθμό των πρόσφατων συγκρούσεων (conflicts), στα οποία συμμετείχε κάθε μεταβλητή. Με τη χρήση αυτής της δομής, διευκολύνεται η πρόσβαση στη μεταβλητή με τη μεγαλύτερη θερμότητα.

Το Rnk είναι μια απλή δομή - struct που περιέχει τη θερμότητα (heat) και τη θέση (position) μιας μεταβλητής. Το Hotter είναι επίσης μια δομή - struct που χρησιμεύει για τη σύγκριση της θερμότητας δύο μεταβλητών, λειτουργεί δηλαδή σαν συγκριτής (comparator), και επιτρέπει την ταξινόμηση των μεταβλητών βάσει της θερμότητάς τους.

Η δομή σωρός (heap) του PrecoSAT χρησιμοποιείται για να επιτρέπει την εφαρμογή του αλγόριθμου ταξινόμησης heap sort στις μεταβλητές βάσει της θερμότητάς τους και ενσωματώνει μια δομή τύπου στοίβα (stack). Η στοίβα χρησιμεύει για τη γρήγορη ανάληψη της μεταβλητής με τη μεγαλύτερη θερμότητα μέσω της μεθόδου schedule.decide.max() (σε O(1)) . Η αλλαγή των θέσεων των μεταβλητών στο heap συνεπάγεται την αλλαγή της θέσης τους στο stack, ώστε στην κορυφή του stack (θέση stack[0]) να βρίσκεται η μεταβλητή με τη μεγαλύτερη θερμότητα.

Όταν ανατεθεί τιμή σε μια μεταβλητή, τότε παύει να είναι υποψήφια μεταβλητή απόφασης (αφού έχει ήδη πάρει τιμή) και αφαιρείται από τη δομή schedule.decide. Αντίστοιχα, όταν μια μεταβλητή χάσει την τιμή της (πχ. μετά από κάποια σύγκρουση) τότε η μεταβλητή αυτή προστίθεται στη δομή schedule.decide και θεωρείται υποψήφια μεταβλητή απόφασης. Η πρόσθεση μιας μεταβλητής στη δομή schedule.decide γίνεται στις μεθόδους αρχικοποίησης των δομών (solver::init, solver::resize), πριν αρχίσει η διαδικασία ανάθεσης τιμών στις μεταβλητές, και στη μέθοδο solver::unassign, μόλις η μεταβλητή χάσει την τιμή της. Αντίθετα, η αφαίρεση μιας μεταβλητής από τη δομή

schedule.decide γίνεται στη μέθοδο solver::decide, στην οποία γίνεται η επιλογή της μεταβλητής απόφασης, μόνο αν η μεταβλητή στην οποία έχει ανατεθεί τιμή, έχει μεγαλύτερη θερμότητα από τη θερμότερη μη ανατεθειμένη μεταβλητή. Η χρήση της δομής schedule.decide με αυτό τον τρόπο έχει ως αποτέλεσμα την εξοικονόμηση χρόνου και την αύξηση της αποδοτικότητας του επιλυτή.

Μια άλλη δομή δεδομένων που χρησιμοποιεί ο PrecoSAT, η οποία σχετίζεται με το ευρετικό που εφαρμόζει ο PrecoSAT για την επιλογή της επόμενης μεταβλητής απόφασης, είναι η δομή iirf (infinite impulse response filter). Η iirf είναι μια δομή δεδομένων στην οποία αποθηκεύεται ο αριθμός της σύγκρουσης (conflict) στην οποία συμμετείχε κάποια μεταβλητή. Το iirf(v,κ) είναι ο αριθμός της σύγκρουσης στην οποία συμμετείχε η μεταβλητή v, στην κ-οστή, μετρώντας από το τέλος, σύγκρουση στην οποία συμμετείχε. Για παράδειγμα, iirf(v,1) είναι ο αριθμός της τελευταίας σύγκρουσης (conflict) στην οποία συμμετείχε η μεταβλητή v.

Ο αριθμός των προηγούμενων συγκρούσεων που λαμβάνονται υπόψη για τον υπολογισμό της νέας θερμότητας μιας μεταβλητής που συμμετέχει σε σύγκρουση, καθορίζει την τάξη (order) του φιλτραρίσματος (filter). Ο PrecoSAT αποθηκεύει στη δομή iirf τους αριθμούς των δύο τελευταίων συγκρούσεων στα οποία συμμετείχε μια μεταβλητή. Για τον υπολογισμό της νέας θερμότητας κάποιας μεταβλητής που συμμετέχει σε σύγκρουση, ο PrecoSAT λαμβάνει υπόψη τις δύο προηγούμενες συγκρούσεις στις οποίες συμμετείχε η μεταβλητή, καθώς και την τρέχουσα σύγκρουση (δηλαδή, filter order=3). Η θερμότητα κάποιας μεταβλητής αυξάνεται αν η μεταβλητή συμμετείχε στη σύγκρουση με αριθμό ίσο με τον αριθμό της τελευταίας σύγκρουσης μείον κ (όπου κ είναι το κ στο iirf(v,κ)). Όσο πιο «παλιά» είναι η σύγκρουση (δηλαδή όσο μεγαλύτερο είναι το κ στο iirf(v,κ)), τόσο μικρότερη είναι η αύξηση της θερμότητας (heat) της μεταβλητής. Στον PrecoSAT, το κ παίρνει τις τιμές 1 (για το τελευταίο conflict πριν το τρέχον conflict) και 2 (για το προτελευταίο conflict πριν το τρέχον conflict).

Το ευρετικό επιλογής μεταβλητής απόφασης του PrecoSAT αναφέρεται ως low pass filter, διότι είναι κατασκευασμένο να απαγορεύει τις υψηλές συχνότητες – θερμότητες (απαγορεύει τις θερμότητες πάνω από 2^{24}) και να μειώνει τις θερμότητες των μεταβλητών, προκειμένου οι μεγαλύτερες θερμότητες να έχουν μεγάλη μείωση ενώ οι

μικρές θερμότητες να μειώνονται ελάχιστα. Η αύξηση της θερμότητας περιορίζεται μόνο στις μεταβλητές που συμμετέχουν σε συγκρούσεις, μέχρι που η θερμότητα κάποιας μεταβλητής, η οποία συμμετέχει σε σύγκρουση, ξεπεράσει το κατώφλι (cutoff frequency), το οποίο ισούται με 2^{24}. Τότε, εκτελείται η μέθοδος rescore(), η οποία μειώνει τη θερμότητα όλων των μεταβλητών, καθώς και την τιμή της μεταβλητής hinc, η οποία καθορίζει το ποσό της αύξησης της θερμότητας των μεταβλητών που συμμετέχουν σε συγκρούσεις.

5.4.2 Αλγόριθμοι θερμότητας

Ο PrecoSAT επιλέγει συνήθως ως μεταβλητή απόφασης την πρώτη διαθέσιμη μεταβλητή που έχει τη μεγαλύτερη θερμότητα, μέσω της μεθόδου schedule.decide.max. Μια μεταβλητή θεωρείται διαθέσιμη όταν ο τύπος της (Variable type, Vrt) είναι ΕΛΕΥΘΕΡΗ (FREE) και δεν της έχει ανατεθεί τιμή (vals[v] = 0).

Περίπου μία στις 2000 φορές εκτέλεσης του αλγόριθμου επιλογής μεταβλητής απόφασης (μέθοδος solver::decide στο αρχείο precosat.cc), η επιλογή της μεταβλητής απόφασης γίνεται με τυχαιοποιημένο τρόπο.

Πιο κάτω περιγράφονται οι αλγόριθμοι με τους οποίους υπολογίζεται η θερμότητα των μεταβλητών.

Ορισμοί:

- Το hinc (heat increament) είναι η τιμή με την οποία αυξάνεται η θερμότητα (heat) μιας μεταβλητής και εκπροσωπεί την ομώνυμη μεταβλητή του κώδικα.
- Μια μεταβλητή συμμετέχει σε σύγκρουση (conflict) αν επιλύεται (resolved) κατά τη διάρκεια ανάλυσης της σύγκρουσης (conflict analysis) για την παραγωγή της εκμαθημένης πρότασης (learned clause) ή αν περιέχεται στην εκμαθημένη πρόταση.

5.4.2.1 Αλγόριθμος αρχικοποίησης θερμότητας

Ο αλγόριθμος αρχικοποίησης θερμότητας, HeatInitialization, δείχνει τα βήματα για την αρχικοποίηση της θερμότητας των μεταβλητών και την αρχικοποίηση της δομής iirf, όπου αποθηκεύεται ο αριθμός της προτελευταίας και τελευταίας σύγκρουσης στις οποίες συμμετείχε μια μεταβλητή. Ο αλγόριθμος αυτός εκτελείται μόνο μια φορά, κατά τη διαδικασία αρχικοποίησης όλων των δομών δεδομένων.

```
1.      HeatInitialization ( VariableSet V, OrderList O ) {
2.          for each variable v in V {
3.              v→heat = 0 ;
4.          }
5.          hinc = 128 ;
6.          for each variable v in V {
7.              for each order o in O {
8.                  iirf(v,o)= -1 ;
9.              }
10.         }
11.     }
```

Όπως φαίνεται από τον αλγόριθμο HeatInitialization, η θερμότητα (heat) κάθε μεταβλητής αρχικοποιείται με 0 και κάθε θέση της δομής iirf αρχικά ισούται με -1.

5.4.2.2 Αλγόριθμος υπολογισμού θερμότητας

Ο αλγόριθμος υπολογισμού θερμότητας, HeatCalculation, περιγράφει πώς ακριβώς μεταβάλλεται η θερμότητα των μεταβλητών. Ο αλγόριθμος αυτός εκτελείται μετά την Ανάλυση Σύγκρουσης για την εύρεση της εκμαθημένης πρότασης, όταν υπάρξει σύγκρουση.

Σύντομη επεξήγηση κάθε γραμμής του αλγορίθμου φαίνεται σε σχόλια, στα δεξιά των γραμμών. Μετά, γίνεται αναλυτικότερη περιγραφή των κομματιών από τα οποία αποτελείται ο αλγόριθμος, με παραπομπές στις αντίστοιχες γραμμές.

Αλγόριθμος HeatCalculation

```
1.    HeatCalculation (ConflictClause cls, ConflictNumber c, ConflictVariables Q,
      VariableSet V, OrderList O) {
2.        if (cls<>null){ //if there is a conflict
3.            for each variable v in Q { //for each conflict variable
4.                if ( iirf(v,1) == c-1 ){ //if v participated in conflict c-1
5.                    v→heat = v→heat + hinc/4 ; //(last conflict)
6.                }
7.                if ( iirf(v,2) == c-2 ){ //if v participated in conflict c-2
8.                    v→heat = v→heat + hinc/8 ; //(penultimate conflict)
9.                }
10.               v→heat = v→heat + hinc/2 ; //(current conflict)
11.               iirf(v,2) = c-1 ; //update the number of the penultimate conflict
12.               iirf(v,1) = c ; //update the number of the last conflict
13.               if ( v→heat >= 2^24){ //2^24 is the cutoff frequency
14.                   hinc = hinc / 2^14 ;  //decrease hinc
15.                   For each var in V { //for all the variables of the formula
16.                       var→heat = var→heat / 2^14 ; //decrease the heat
17.                    }
18.               }
19.           }
20.           hinc = (hinc*100 + 10) /100 ; //increase hinc
21.       }
22.   }
```

Περιγραφή του αλγόριθμου HeatCalculation

Γραμμές 4-10: Συνοπτικά, σε αυτές τις γραμμές, για κάθε μεταβλητή που συμμετέχει στη σύγκρουση (conflict variable), αυξάνεται η θερμότητα της κατά hinc/2, επειδή συμμετέχει στην τρέχουσα σύγκρουση. Η θερμότητα της μεταβλητής αυξάνεται επιπρόσθετα κατά hinc/4, εάν συμμετείχε στην τελευταία σύγκρουση και κατά hinc/8, εάν συμμετείχε στην προτελευταία σύγκρουση.

Γραμμές 11-12: Στις γραμμές 11-12 ανανεώνονται οι αριθμοί της τελευταίας και προτελευταίας σύγκρουσης στις οποίες συμμετείχε η μεταβλητή.

Γραμμές 13-19: Η θερμότητα (heat) αλλάζει μόνο για τις μεταβλητές που συμμετέχουν σε σύγκρουση (conflict variables) μέχρι που η θερμότητα κάποιας από αυτές τις μεταβλητές ξεπεράσει το κατώφλι 2^{24} (cutoff frequency). Τότε, καλείται η μέθοδος

rescore, όπου μειώνεται η θερμότητα όλων των μεταβλητών, καθώς και το hinc. Σε αυτή τη διαδικασία, γίνεται το φιλτράρισμα των υψηλών θερμοτήτων. Δηλαδή, μειώνονται όλες οι θερμότητες, αλλά όσο μεγαλύτερη είναι η θερμότητα μιας μεταβλητής, τόσο περισσότερο μειώνεται. Οι γραμμές 14-17 περιγράφουν τη λειτουργία της μεθόδου rescore στο αρχείο precosat.cc.

Γραμμή 20: Μετά από κάθε σύγκρουση, γίνεται μικρή αύξηση του hinc. Όπως φαίνεται από τον αλγόριθμο υπολογισμού θερμότητας, το hinc αυξάνεται με μικρό ρυθμό σε κάθε ανάλυση σύγκρουσης, και μειώνεται αρκετά σε κάθε επαναβαθμολόγηση – rescore (γραμμές 14-17).

Στο επόμενο υποκεφάλαιο περιγράφονται οι αλγόριθμοι που χρησιμοποιεί ο προτασιακός επιλυτής PrecoSAT για την ανάθεση τιμής στη μεταβλητή απόφασης.

5.5 Ευρετικό για την επιλογή τιμής της μεταβλητής απόφασης

Για την επιλογή τιμής (phase selection) της μεταβλητής απόφασης, ο PrecoSAT συνδυάζει τις πιο κάτω τεχνικές:

- Jeroslow- Wang heuristic
- RSAT heuristic

Η ιδέα του ευρετικού επιλογής τιμής που χρησιμοποιεί ο PrecoSAT είναι η εξής: Αν δεν είναι αποθηκευμένη η τελευταία τιμή που ανατέθηκε στη μεταβλητή απόφασης κατά τη διάρκεια λειτουργίας του επιλυτή, τότε χρησιμοποιείται το ευρετικό Jeroslow-Wang. Αλλιώς, χρησιμοποιείται το ευρετικό RSAT και η μεταβλητή απόφασης παίρνει την τιμή που είχε στην τελευταία της ανάθεση.

5.5.1 Δομές δεδομένων

Για το ευρετικό (heuristic) Jeroslow – Wang, χρησιμοποιείται η δομή jwhs, στην οποία αποθηκεύεται το βάρος κάθε κατηγορήματος (literal). Η διαδικασία εύρεσης του βάρους ενός κατηγορήματος περιγράφεται στο υποκεφάλαιο 5.5.2.

Για το ευρετικό RSAT, η τιμή της μεταβλητής αποθηκεύεται στο πεδίο phase (τύπου Val) κάθε μεταβλητής. Οι μεταβλητές είναι τύπου Var. Τα Var και Val είναι δομές - structs και ορίζονται στο αρχείο precosat.hh.

5.5.2 Ευρετικό Jeroslow-Wang

Έστω ότι N(v,k,s) είναι ο αριθμός των προτάσεων (clauses) μεγέθους k στις οποίες εμφανίζεται μια μεταβλητή v. Η μεταβλητή εμφανίζεται θετικά αν s=1 ή αρνητικά αν s=0.

Για παράδειγμα, N(2,4,1) δείχνει τον αριθμό των θετικών εμφανίσεων της μεταβλητής 2 στις προτάσεις μεγέθους 4.

Το βάρος (Weight) κάθε κατηγορήματος (literal) υπολογίζεται με τον εξής τρόπο:

$W(v,s) = SUM\ [k=1..n] \quad (N(v,k,s) * [\ (128 * 2^{-k}) +1])$

Αν W(v,1) > W(v,0) τότε το ευρετικό Jeroslow-Wang επιλέγει την θετική πολικότητα (positive phase), δηλαδή αναθέτει στη μεταβλητή v την τιμή true. Αλλιώς, αναθέτει στη μεταβλητή v την τιμή false [6]. Ακολουθεί ο αλγόριθμος υπολογισμού των βαρών για το ευρετικό Jeroslow-Wang.

Αλγόριθμος JWH

```
1.      JWH (LiteralSet L, ClauseSet S) {
2.          For each lit in L {    //Initialize
3.              Weight (lit) = 0 ;
4.          }
5.          For each clause cls in S { //Calculate Weight
6.              k = size of cls ;
7.              For each lit in cls {
8.                  Weight (lit) = Weight (lit) + [ (128 * 2^-k)  +1 ] ;
9.              }
10.         }
11.     }
```

Τα βάρη των κατηγορημάτων ανανεώνονται κάθε φορά που παράγεται μια εκμαθημένη πρόταση, μόνο για τα κατηγορήματα που εμπεριέχονται στην εκμαθημένη πρόταση. Για κάθε κατηγόρημα lit της εκμαθημένης πρότασης μεγέθους k, το βάρος του αυξάνεται: Weight (lit) = Weight (lit) + [(128 * 2^{-k}) +1] ;

5.5.3 Ευρετικό Rsat

Το ευρετικό Rsat πήρε το όνομά του από τον επιλυτή RSAT, στον οποίο χρησιμοποιήθηκε. Σύμφωνα με τη στρατηγική που ακολουθείται στο ευρετικό Rsat, η μεταβλητή απόφασης παίρνει την τιμή που είχε στην τελευταία της ανάθεση [6].

Οι προηγούμενες τιμές των μεταβλητών "ξεχνιούνται" όταν εκτελεστεί η διαδικασία rebias. Η διαδικασία rebias εκτελείται, όταν ξεπεράσουμε ένα όριο συγκρούσεων (limit.rebias.conflicts στον κώδικα). Αρχικά, ο οριακός αριθμός συγκρούσεων που οδηγούν σε rebias (limit.rebias.conflicts) ισούται με 1000. Ο αριθμός αυτός αυξάνεται σταδιακά. Ο ρυθμός αύξησης του οριακού αριθμού των συγκρούσεων που πρέπει να γίνουν πριν εκτελεστεί η διαδικασία rebias, ακολουθεί την ακολουθία Luby (Luby series). Κάθε φορά που εκτελείται η διαδικασία rebias, το όριο μεγαλώνει κατά δ, όπου δ = ο επόμενος αριθμός της ακολουθίας Luby επί 1000.

Η ακολουθία Luby είναι η ακολουθία 1 1 2 1 1 2 4 1 1 2 4 8 1 1 2 4 8 16 …

5.6 Σύνοψη

Συνοπτικά, ο PrecoSAT χρησιμοποιεί μια παραλλαγή του αλγόριθμου VSIDS (Variable State Independent Decaying Sum) για την επιλογή της μεταβλητής απόφασης, και το συνδυασμό των αλγόριθμων Rsat και Jeroslow-Wang για την επιλογή της τιμής ανάθεσης για τη μεταβλητή απόφασης.

Αξίζει να σημειωθεί ότι η θερμότητα των μεταβλητών (heat), καθώς και οι προηγούμενες τιμές των μεταβλητών διατηρούνται μετά από τις επανεκκινήσεις (restarts) της αναζήτησης. Έτσι, ο επιλυτής μετά από κάθε επανεκκίνηση επιλέγει

διαφορετικές μεταβλητές, ανάλογα με τη θερμότητα τους, και ακολουθεί διαφορετικά μονοπάτια.

Κεφάλαιο 6

Υλοποίηση

6.1 Εισαγωγή 54
6.2 Βασική ιδέα της υλοποίησης 57
6.3 Εντολές εκτέλεσης 59
6.4 Περιγραφή της λειτουργίας των παραμέτρων 59
6.4.1 Παράμετροι για τον καθορισμό αρχείων εισόδου 60
6.4.2 Παράμετροι για εκτύπωση στατιστικών στοιχείων σε αρχεία εξόδου 60
6.4.3 Παράμετροι χρονικών επιπέδων 62
6.4.4 Παράμετροι είδους μεταβλητής 63
6.4.5 Παράμετροι «θορύβου» 64
6.4.6 Παράμετρος αύξησης θερμότητας 64
6.4.7 Παράμετροι κατάταξης μεταβλητών 65
6.4.8 Παράμετροι επιλογής τιμής 67
6.4.9 Συνδυασμός των παραμέτρων 68
6.5 Παραδείγματα εκτέλεσης 70

6.1 Εισαγωγή

Στόχος της φάσης υλοποίησης ήταν η ενσωμάτωση διάφορων τεχνικών στη λειτουργία του προτασιακού επιλυτή PrecoSAT, για την αποδοτικότερη επίλυση των προβλημάτων Προγραμματισμού Δράσης. Στη φάση αυτή, τροποποιήθηκε ο κώδικας του προτασιακού επιλυτή PrecoSAT και υλοποιήθηκαν οι λειτουργίες διάφορων ευρετικών μεθόδων, οι οποίες ενσωματώνονται στη λειτουργία του PrecoSAT, όταν δοθούν οι αντίστοιχες παράμετροι (command line arguments). Επίσης, έγιναν μερικές αλλαγές στον κώδικα του συστήματος SATPLAN για την επίτευξη της επικοινωνίας μεταξύ του SATPLAN και του τροποποιημένου PrecoSAT. Σκοπός των ευρετικών που

υλοποιήθηκαν είναι η αξιοποίηση των χαρακτηριστικών που έχουν τα προβλήματα Προγραμματισμού Δράσης για την αποδοτικότερη επίλυσή τους από τον προτασιακό επιλυτή PrecoSAT.

Για τη βελτίωση της απόδοσης του PrecoSAT στην επίλυση προβλημάτων Προγραμματισμού Δράσης, επικεντρωθήκαμε σε τροποποιήσεις του ευρετικού απόφασης που χρησιμοποιείται στον PrecoSAT. Πιο συγκεκριμένα, προσπαθήσαμε να εξετάσουμε πώς μπορούμε να συνδυάσουμε το ευρετικό που χρησιμοποιεί ο PrecoSAT, το οποίο είναι μια παραλλαγή του ευρετικού VSIDS (Variable State Independent Decaying Sum), με κάποιες άλλες εξειδικευμένες τεχνικές, οι οποίες αξιοποιούν το γεγονός ότι το πρόβλημα εισόδου είναι ένα Πρόβλημα Προγραμματισμού Δράσης, για να βελτιώσουμε το χρόνο επίλυσης των προβλημάτων Προγραμματισμού Δράσης. Οι εξειδικευμένες τεχνικές που υλοποιήθηκαν, χρησιμοποιούν τις επιπρόσθετες πληροφορίες που παρέχει το σύστημα SATPLAN σχετικά με το πρόβλημα που επιλύεται, και προσπαθούν να τις αξιοποιήσουν για την επίτευξη πιο στοχευμένης επιλογής μεταβλητής απόφασης (variable selection), καθώς και την επίτευξη πιο στοχευμένης επιλογής τιμής για τη μεταβλητή απόφασης (phase selection).

Όπως αναφέρθηκε σε προηγούμενα κεφάλαια, οι καλές αποφάσεις του προτασιακού επιλυτή οδηγούν σε καλύτερα μονοπάτια, εξοικονομώντας πολύτιμο χρόνο. Επομένως, τροποποιώντας το ευρετικό απόφασης του προτασιακού επιλυτή, μπορούμε να βελτιώσουμε τον συνολικό χρόνο επίλυσης των προβλημάτων. Για την τροποποίηση του ευρετικού απόφασης, μια προσέγγιση ήταν η ενσωμάτωση διάφορων τεχνικών στο ευρετικό που χρησιμοποιεί ο PrecoSAT. Μια διαφορετική προσέγγιση ήταν η υλοποίηση νέου ευρετικού απόφασης από την αρχή, πλήρως ανεξάρτητου από το ευρετικό που χρησιμοποιεί ο PrecoSAT. Στην παρούσα διπλωματική ακολουθήσαμε την πρώτη προσέγγιση, διότι το ευρετικο VSIDS θεωρείται ένα από τα καλύτερα ευρετικά επιλογής μεταβλητής στους γενικούς – μη εξειδικευμένους – μοντέρνους επιλυτές προβλημάτων, και θα ήταν ενδιαφέρον να εξετάσουμε πώς μπορουμε να αξιοποιήσουμε αυτή την τεχνική, για να βελτιώσουμε το χρόνο επίλυσης σε συγκεκριμένο είδος προβλημάτων, όπως είναι τα προβλήματα Προγραμματισμού Δράσης.

Τα χαρακτηριστικά των προβλημάτων Προγραμματισμού Δράσης στα οποία βασιστήκαμε για την επινόηση των εξειδικευμένων τεχνικών που υλοποιήθηκαν ήταν το είδος των μεταβλητών (fact, action, actionNOOP), το χρονικό επίπεδο στο οποίο ανήκουν, δηλαδή το time level όπως καθορίζεται από το σύστημα SATPLAN και, στην περίπτωση που οι μεταβλητές είναι facts, κατά πόσο ανήκουν στην αρχική ή στην τελική κατάσταση (goalFacts, initFacts).

Τα ευρετικά που υλοποιήθηκαν και σχετίζονται με τα χρονικά επίπεδα στα οποία ανήκουν οι μεταβλητές, χρησιμοποιούν τις ιδέες στις οποίες στηρίζονται οι εμπρόσθιοι (αναζήτηση ξεκινώντας από την αρχική προς την τελική κατάσταση - δηλαδή από μικρότερα χρονικά επίπεδα προς μεγαλύτερα χρονικά επίπεδα) και οπίσθιοι (αναζήτηση ξεκινώντας από την τελική προς την αρχική κατάσταση – δηλαδή από μεγαλύτερα χρονικά επίπεδα προς μικρότερα χρονικά επίπεδα) αλγόριθμοι, με τους οποίους επιλύονται τα προβλήματα Προγραμματισμού Δράσης, καθώς και τροποποιήσεις των ιδεών αυτών (πχ. επιλογή μεταβλητών από τα πρώτα ή τελευταία k χρονικά επίπεδα). Επίσης, γνωρίζοντας ότι τα προβλήματα Προγραμματισμού Δράσης έχουν συνήθως λιγότερες μεταβλητές από κάποιο είδος σε σχέση με τον αριθμό των μεταβλητών άλλου είδους (πχ. έχουν συνήθως πολύ λιγότερα facts σε σχέση με actions), μέσω των ευρετικών που σχετίζονται με το είδος της μεταβλητής γίνεται προσπάθεια μείωσης του χώρου αναζήτησης, περιορίζοντας την αναζήτηση σε μεταβλητές συγκεκριμένου είδους (πχ. facts).

Για την υλοποίηση των νέων τεχνικών στον PrecoSAT, έγιναν αλλαγές στα αρχεία precomain.cc, precosat.hh και precosat.cc, ενώ για την επίτευξη της επικοινωνίας μεταξύ του SATPLAN και του τροποποιημένου PrecoSAT, έγιναν μικρές αλλαγές στα αρχεία input.cpp, satplan.cpp, SolverInterface.h και precosatSolver.cpp.

Ακολουθεί η περιγραφή της βασικής ιδέας στην οποία στηρίζεται η υλοποίηση των τεχνικών, μια σύντομη αναφορά για τις εντολές εκτέλεσης του προτασιακού επιλυτή PrecoSAT και η αναλυτική περιγραφή των παραμέτρων που υλοποιήθηκαν. Στη συνέχεια, παρουσιάζονται κάποια παραδείγματα εκτέλεσης του προτασιακού επιλυτή PrecoSAT με τη χρήση των νέων παραμέτρων και δίνεται μια σύντομη σχετική επεξήγηση.

Υπενθύμιση των όρων που χρησιμοποιούνται:

- Facts: Κατηγορήματα που ισχύουν σε μια κατάσταση.
- InitFacts: Facts που ισχύουν στην αρχική κατάσταση.
- GoalFacts: Facts που ισχύουν στην τελική κατάσταση.
- Actions: Δράσεις που, όταν εφαρμοστούν, προκαλούν αλλαγή στην κατάσταση.
- ActionNOOPs: Δράσεις που, όταν εφαρμοστούν, δεν προκαλούν καμία αλλαγή στην κατάσταση (πχ. have(cake)).

6.2 Βασική ιδέα της υλοποίησης

Η βασική ιδέα στην οποία στηρίζεται η υλοποίηση των τεχνικών είναι η χρηση μιας δομής ίδιου τύπου με τη δομή schedule.decide, για την προσωρινή αποθήκευση των υποψήφιων μεταβλητών απόφασης οι οποίες απορρίπτονται, διότι δεν ικανοποιούν τα κριτήρια των τεχνικών που εφαρμόζονται. Η schedule.decide, όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, είναι μια δομή που χρησιμοποιείται στον προτασιακό επιλυτή PrecoSAT και λειτουργεί σαν στοίβα. Στη schedule.decide αποθηκεύονται οι υποψήφιες μεταβλητές απόφασης, ταξινομημένες βάσει της θερμότητάς τους.

Πριν αρχίσει η αναζήτηση μεταβλητής απόφασης, αφαιρούνται (γίνονται pop) από τη δομή schedule.decide όλες οι μεταβλητές στις οποίες ανατέθηκε τιμή, και ανανεώνονται οι μετρητές για τις διαθέσιμες μεταβλητές κάθε είδους και κάθε χρονικού επιπέδου. Επίσης, κατά τη διάρκεια αυτής της προεργασίας, για κάθε είδος μεταβλητής, υπολογίζεται το ελάχιστο και μέγιστο χρονικό επίπεδο στο οποίο εμφανίζονται οι μεταβλητές εκείνου του είδους. Στο ευρετικό που χρησιμοποιεί ο PrecoSAT, δεν αφαιρούνται όλες οι μεταβλητές στις οποίες ανατέθηκε τιμή, αλλά αφαιρούνται μόνο οι ανατεθειμένες μεταβλητές που είναι θερμότερες από τη θερμότερη μη ανατεθειμένη μεταβλητή. Στην περίπτωση όμως των νέων ευρετικών, η διαδικασία αυτή ήταν απαραίτητη για την ανανέωση των μετρητών και τη μετέπειτα αναζήτηση της κατάλληλης μεταβλητής.

Στη συνέχεια, εκτελείται η αναζήτηση μεταβλητής απόφασης που ικανοποιεί τα κριτήρια, όπως προκύπτουν από τις παραμέτρους που δόθηκαν. Πιο συγκεκριμένα, κατά τη διαδικασία εκτέλεσης του ευρετικού απόφασης του PrecoSAT, η αναζήτηση

μεταβλητής απόφασης ξεκινά από τη θερμότερη διαθέσιμη μεταβλητή απόφασης, η οποία βρίσκεται στην κορυφή (top) της δομής schedule.decide. Αρχικά, γίνεται έλεγχος, για να διαπιστωθεί κατά πόσο η θερμότερη μεταβλητή του schedule.decide ικανοποιεί τα κριτήρια που θέτουν οι νέες τεχνικές (πχ. αν δοθεί η παράμετρος –fonly, τότε η υποψήφια μεταβλητή απόφασης πρέπει να είναι fact). Αν η μεταβλητή ικανοποιεί τα κριτήρια, τότε επιλέγεται ως μεταβλητή απόφασης. Διαφορετικά, η μεταβλητή αποθηκεύεται στην προσωρινή δομή για τις μεταβλητές που απορρίπτονται και η αναζήτηση συνεχίζεται με τον έλεγχο της νέας θερμότερης μεταβλητής στη schedule.decide, δηλαδή της θερμότερης από αυτές που έμειναν, η οποία βρίσκεται στην κορυφή της schedule.decide.

Όταν βρεθεί η κατάλληλη μεταβλητή απόφασης, ανατίθεται τιμή στη μεταβλητή απόφασης σύμφωνα με τις παραμέτρους που δόθηκαν ή, αν δε δόθηκαν παράμετροι που σχετίζονται με την επιλογή τιμής της μεταβλητής απόφασης, η ανάθεση τιμής γίνεται σύμφωνα με την προκαθορισμένη μέθοδο του PrecoSAT για την επιλογή τιμής της μεταβλητής απόφασης (JW, Rsat).

Όταν τελειώσει η διαδικασία επιλογής μεταβλητής απόφασης, οι μεταβλητές που απορρίφθηκαν κατά την αναζήτηση, αποθηκεύονται πίσω στη δομή schedule.decide, διατηρώντας τη σειρά που είχαν πριν απορριφθούν (δηλαδή την κατάταξη βάσει της θερμότητάς τους). Αυτό επιτυγχάνεται εύκολα με τις μεθόδους push και pop που έχει η δομή δεδομένων τύπου στοίβα (stack). Αναλυτικότερα, για κάθε μεταβλητή που απορρίφθηκε, η μεταβλητή γίνεται pop από τη δομή με τις μεταβλητές που απορρίφθηκαν και μετά γίνεται push στη δομή schedule.decide. Με τον τρόπο αυτό, γίνεται προσπάθεια μείωσης των σφαλμάτων που μπορεί να προκύψουν στον προϋπάρχοντα κώδικα του PrecoSAT λόγω των τροποποιήσεων που υφίσταται κατά την υλοποίηση των νέων τεχνικών.

Παρόλο που υπήρχαν και άλλες (ίσως αποδοτικότερες) ιδέες, τελικά επικράτησε η ιδέα αυτή, διότι, λόγω της απλότητάς της, περιόριζε σε μεγάλο βαθμό τα σφάλματα που προέκυπταν σε άλλα σημεία του κώδικα εξαιτίας της υλοποίησης των νέων τεχνικών.

6.3 Εντολές εκτέλεσης

Οι εντολές για την παραγωγή του εκτελέσιμου αρχείου του προτασιακού επιλυτή PrecoSAT είναι οι εξής:

./configure

make

Η εκτέλεση του PrecoSAT γίνεται με την εξής εντολή:

./precosat <input_filename.cnf> ....arguments...

Το <input_filename.cnf> είναι το αρχείο που περιέχει τη φόρμουλα σε Συζευκτική Κανονική Μορφή (CNF), σε DIMACS format.

6.4 Περιγραφή της λειτουργίας των παραμέτρων

Για τους σκοπούς της παρούσας διπλωματικής, υλοποιήθηκε η λειτουργία των παραμέτρων (command line arguments) που ακολουθούν. Εκτός από τις παραμέτρους για τον καθορισμό των αρχείων εισόδου και τις παραμέτρους για την εκτύπωση στατιστικών σε αρχεία εξόδου, οι υπόλοιπες παράμετροι αφορούν τις λειτουργίες των ευρετικών και χωρίζονται σε παραμέτρους επιλογής μεταβλητής απόφασης και παραμέτρους επιλογής τιμής για τη μεταβλητή απόφασης. Οι παράμετροι για την επιλογή μεταβλητής απόφασης αποτελούνται από τις παραμέτρους χρονικών επιπέδων, τις παραμέτρους είδους μεταβλητής, τις παραμέτρους «θορύβου», την παράμετρο αύξησης θερμότητας και τις παραμέτρους κατάταξης μεταβλητών. Οι παράμετροι επιλογής τιμής της μεταβλητής απόφασης είναι λιγότερες και αποτελούνται από την ομώνυμη κατηγορία παραμέτρων.

6.4.1 Παράμετροι για τον καθορισμό αρχείων εισόδου

-varf <fn>: Μέσω της παραμέτρου –varf καθορίζεται το όνομα του αρχείου VARFILE (<fn>) που δίνεται ως είσοδος στον PrecoSAT. Το αρχείο VARFILE παράγεται από τον SATPLAN και, για κάθε μεταβλητή, περιέχει το είδος (fact ή action ή actionNOOP), την περιγραφή (πχ. LOAD HOIST0 CRATE1 TRUCK0 DEPOT0), και το χρονικό επίπεδο στο οποίο ανήκει η μεταβλητή (πχ. 10).

-pddl <file>: Με την παράμετρο –pddl ο χρήστης καθορίζει το όνομα του αρχείου που περιέχει την περιγραφή του προβλήματος (αρχική και τελική κατάσταση) στη γλώσσα STRIPS. Αυτά τα είδη αρχείων συνήθως τελειώνουν σε .pddl.

6.4.2 Παράμετροι για εκτύπωση στατιστικών στοιχείων σε αρχεία εξόδου

-rnk <k> <file>: Αν δοθεί η παράμετρος –rnk (rank), ανά <k> αποφάσεις (decisions), τυπώνονται στο αρχείο με το όνομα <file>, η μέση θέση στη στοίβα (schedule.decide) και μέση θερμότητα των μεταβλητών απόφασης που επιλέχθηκαν, καθώς και οι μετρητές των διαθέσιμων μεταβλητών απόφασης για κάθε είδος (initFact=3, goalFact=2, otherFact=1, actionNOOP=0, action=-1). Επίσης, τυπώνεται το ύψος του δέντρου αναζήτησης (decision level) στην τελευταία απόφαση (decision) που έγινε, η τελευταία μεταβλητή που επιλέχθηκε ως μεταβλητή απόφασης, το είδος της μεταβλητής (fact ή action ή actionNOOP), το ποσοστό κατά το οποίο πρέπει να αυξηθεί η θερμότητα αυτής της μεταβλητής, για να φτάσει τη θερμότητα της πιο θερμής μεταβλητής, και ο συνολικός αριθμός των συγκρούσεων που έγιναν μέχρι εκείνη τη στιγμή.

-lrnk <p> <d>: Η παράμετρος –lrnk (limited rank) λειτουργεί σε συνδυασμό με την παράμετρο –rnk. Αν η θέση της μεταβλητής απόφασης είναι μεγαλύτερη από <p>, τότε, για να βρεθεί ο μέσος όρος θέσης που επιλέγει το ευρετικό απόφασης, προστίθεται ο αριθμός <d> αντί η πραγματική θέση της μεταβλητής στη στοίβα της δομής schedule.decide.

-pfx <k> <N> <file>: Αν δοθεί η παράμετρος –pfx (print fixed variables), τότε εκτυπώνονται, ανά <N> αποφάσεις, στο αρχείο με το όνομα <file>, στατιστικά στοιχεία

σχετικά με το ποσοστό των μεταβλητών που παίρνουν τιμή (fixed variables) κατά τη διαδικασία της Μοναδιαίας Μετάδοσης (Unit Propagation), μετά από κάθε απόφαση που γίνεται στο επίπεδο απόφασης (=ύψος του δέντρου αναζήτησης) <k>.

Αναλυτικότερα, στο αρχείο εξόδου της παραμέτρου –pfx φαίνονται τα εξής:

1) Ο αριθμός απόφασης (decision).

2) Ο αριθμός των επισκέψεων της αναζήτησης στο επίπεδο απόφασης <k>, στις τελευταίες <N> αποφάσεις (levelVisits).

3) Ο μέσος όρος των μεταβλητών που πήραν τιμή (fixed variables) κατά τη διαδικασία της Μοναδιαίας Μετάδοσης, μετά από τις αποφάσεις στο επίπεδο <k>, για τις τελευταίες <N> αποφάσεις.

4) Ο μικρότερος αριθμός αναθέσεων που έγιναν σε μεταβλητές στη διαδικασία της Μοναδιαίας Μετάδοσης, μετά από τις αποφάσεις στο επίπεδο <k>, για τις τελευταίες <N> αποφάσεις.

5) Ο μεγαλύτερος αριθμός αναθέσεων που έγιναν σε μεταβλητές στη διαδικασία της Μοναδιαίας Μετάδοσης, μετά από τις αποφάσεις στο επίπεδο <k>, για τις τελευταίες <N> αποφάσεις.

6) Ο συνολικός αριθμός των μεταβλητών στις οποίες έχει ανατεθεί τιμή από την έναρξη της λειτουργίας του προτασιακού επιλυτή μέχρι το τρέχον σημείο της αναζήτησης.

7) Το ποσοστό των μεταβλητών που έχουν τιμή στο τρέχον σημείο της αναζήτησης.

-binfo <N> <K> <file>: Αν δοθεί η παράμετρος –binfo (branching information), τότε, κάθε <K> αποφάσεις, στο αρχείο με όνομα <file>, εκτυπώνονται πληροφορίες σχετικά με τις μεταβλητές που επιλέγονται ως μεταβλητές απόφασης. Πιο συγκεκριμένα, εκτυπώνεται ο αριθμός των μεταβλητών απόφασης για κάθε χρονικό επίπεδο (time level), όπως αυτό καθορίζεται από τον SATPLAN, ανά είδος μεταβλητής (fact, initFact,

goalFact, action, actionNOOP), για κάθε <N> επίπεδα απόφασης. Τα στατιστικά αυτά εκτυπώνονται και μετά την τελευταία απόφαση, πριν τερματιστεί η λειτουργία του επιλυτή. Επίσης, σε ένα άλλο αρχείο, το οποίο βρίσκεται στον ίδιο φάκελο με το <file> και έχει το ίδιο όνομα με το <file> με τη διαφορά ότι η ονομασία του ξεκινά με τα γράμματα «ZZLR_», τυπώνονται τα στατιστικά στην τελευταία επανεκκίνηση (restart). Σε κάθε επανεκκίνηση, γίνεται αντικατάσταση αυτού του αρχείου με τα στατιστικά της τελευταίας, μέχρι στιγμής, επανεκκίνησης. Συγκρίνοντας αυτά τα δύο αρχεία (<file> και ZZLR_<file>), ο χρήστης μπορεί να κάνει παρατηρήσεις και να εξάγει συμπεράσματα σχετικά με τις αποφάσεις που έγιναν μετά την τελευταία επανεκκίνηση, μέχρι να τερματιστεί η λειτουργία του PrecoSAT.

6.4.3 Παράμετροι χρονικών επιπέδων

Με τις πιο κάτω παραμέτρους καθορίζεται το χρονικό επίπεδο της μεταβλητής που επιλέγεται ως μεταβλητή απόφασης από τον PrecoSAT.

-tfvar <k> <v>: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που ανήκει στα πρώτα <k> χρονικά επίπεδα. Αν στα πρώτα <k> επίπεδα μείνουν λιγότερες από <v> μεταβλητές, το ευρετικό «χαλαρώνει» τους περιορισμούς επιτρέποντας την επιλογή μεταβλητής από περισσότερα χρονικά επίπεδα, όσο γίνεται πιο κοντά στα αρχικά όρια, επιδιώκοντας να υπάρχουν τουλάχιστον <v> μεταβλητές μέσα στα χρονικά όρια.

-tlvar <k> <v>: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που ανήκει στα τελευταία <k> χρονικά επίπεδα. Αν στα τελευταία <k> επίπεδα μείνουν λιγότερες από <v> μεταβλητές, το ευρετικό «χαλαρώνει» τους περιορισμούς επιτρέποντας την επιλογή μεταβλητής από περισσότερα χρονικά επίπεδα, όσο γίνεται πιο κοντά στα αρχικά όρια, επιδιώκοντας να υπάρχουν τουλάχιστον <v> μεταβλητές μέσα στα χρονικά όρια.

-tflvar <k> <v>: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που ανήκει στα πρώτα ή τελευταία <k> χρονικά επίπεδα. Αν συνολικά στα πρώτα και τελευταία <k> επίπεδα μείνουν λιγότερες από <v> μεταβλητές, το ευρετικό «χαλαρώνει» τους

περιορισμούς επιτρέποντας την επιλογή μεταβλητής από περισσότερα χρονικά επίπεδα, όσο γίνεται πιο κοντά στα αρχικά όρια, επιδιώκοντας να υπάρχουν τουλάχιστον <v> μεταβλητές μέσα στα χρονικά όρια.

-tnflvar <k> <v>: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που δεν ανήκει στα πρώτα ή τελευταία <k> χρονικά επίπεδα. Αν συνολικά στα ενδιάμεσα <k> επίπεδα μείνουν λιγότερες από <v> μεταβλητές, το ευρετικό «χαλαρώνει» τους περιορισμούς επιτρέποντας την επιλογή μεταβλητής από περισσότερα χρονικά επίπεδα, όσο γίνεται πιο κοντά στα αρχικά όρια, επιδιώκοντας να υπάρχουν τουλάχιστον <v> μεταβλητές μέσα στα χρονικά όρια.

6.4.4 Παράμετροι είδους μεταβλητής

Με τις πιο κάτω παραμέτρους καθορίζεται το είδος της μεταβλητής που επιλέγεται ως μεταβλητή απόφασης από τον PrecoSAT.

-fonly: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact.

-initFacts: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που είναι initFact (δηλαδή fact της αρχικής κατάστασης).

-goalFacts: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που είναι goalFact (δηλαδή fact της τελικής κατάστασης).

-noopy: Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που είναι actionNOOP.

-noopn: Αποφεύγεται η επιλογή μεταβλητών που είναι actionNOOP.

Αν δεν υπάρχουν άλλες υποψήφιες μεταβλητές απόφασης του είδους που σχετίζεται με την παράμετρο (πχ. αν δόθηκε η παράμετρος –fonly και δεν υπάρχουν άλλες υποψήφιες μεταβλητές απόφασης που είναι facts), τότε εφαρμόζεται το προϋπάρχον ευρετικό του PrecoSAT, δηλαδή επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή ως μεταβλητή απόφασης.

6.4.5 Παράμετροι «θορύβου»

Οι παράμετροι «θορύβου» είναι οι παράμετροι με τις οποίες, αν ικανοποιούνται κάποια κριτήρια, εφαρμόζεται το αρχικό ευρετικό του PrecoSAT (δηλαδή η επιλογή της θερμότερης μη ανατεθειμένης μεταβλητής) αντί το ευρετικό βάσει των νέων παραμέτρων. Ακολουθεί η περιγραφή αυτών των παραμέτρων.

-noise<p>: Οι νέοι κανόνες του ευρετικού, βάσει των υπόλοιπων παραμέτρων που δίνονται, εφαρμόζονται με πιθανότητα (100 - <p>)/100, όπου p ένας ακέραιος αριθμός. Για παράδειγμα, αν ο χρήστης δώσει την παράμετρο –noise 20, τότε οι νέοι κανόνες εφαρμόζονται με πιθανότητα 80%. Διαφορετικά, εφαρμόζεται το προϋπάρχον ευρετικό του PrecoSAT, δηλαδή επιλέγεται η πιο θερμή μεταβλητή.

-tdlevel <k>: Οι τροποποιήσεις στο ευρετικό απόφασης του PrecoSAT εφαρμόζονται μέχρι το <k>-επίπεδο του δέντρου αναζήτησης (k decision level).

-fdlevel <k>: Οι τροποποιήσεις στο ευρετικό απόφασης του PrecoSAT εφαρμόζονται από το <k>-επίπεδο του δέντρου αναζήτησης (k decision level) και μετά.

-trlevel <k>: Οι τροποποιήσεις στο ευρετικό απόφασης του PrecoSAT εφαρμόζονται μέχρι την <k>-οστή επανεκκίνηση (restart).

-frlevel <k>: Οι τροποποιήσεις στο ευρετικό απόφασης του PrecoSAT εφαρμόζονται από την <k>-οστή επανεκκίνηση (restart) και μετά.

6.4.6 Παράμετρος αύξησης θερμότητας

-fheat <N>: Αν δοθεί με άλλες παραμέτρους, γίνεται επιπλέον αύξηση της θερμότητας των μεταβλητών που είναι facts, όταν συμμετέχουν σε σύγκρουση, κατά το <N>% της θερμότητάς τους (βλ. Αλγόριθμο HeatCalculation κεφ.5). Αν δε δοθούν άλλες παράμετροι, τότε, εκτός την αύξηση θερμότητας, γίνεται προσπάθεια επιλογής μεταβλητής που είναι fact, αν αυτή έχει την ίδια θερμότητα με τη θερμότερη μη ανατεθειμένη μεταβλητή που βρίσκεται στην κορυφή της δομής schedule.decide. Στην περίπτωση αυτή, αν δεν υπάρχει μεταβλητή του είδους fact που έχει θερμότητα όση η θερμότερη μη ανατεθειμένη μεταβλητή, τότε επιλέγεται η θερμότερη μη ανατεθειμένη

μεταβλητή που βρίσκεται στην κορυφή της schedule.decide, όπως γίνεται στον αρχικό αλγόριθμο του PrecoSAT. Ο λόγος για τον οποίο γίνεται αυτή η προσπάθεια φαίνεται κυρίως στο διάστημα που μεσολαβεί από την έναρξη της λειτουργίας του προτασιακού επιλυτή μέχρι την πρώτη σύγκρουση (conflict), διότι μέχρι εκείνη τη στιγμή οι θερμότητες όλων των μεταβλητών ισούνται με μηδέν και είναι πολύ πιθανόν οι πρώτες μεταβλητές της στοίβας schedule.decide να είναι actions. Αν η αναζήτηση αρχίσει με μεταβλητές απόφασης που είναι actions, τότε ίσως οι μεταβλητές είδους fact να μη συμμετέχουν στις συγκρούσεις που θα συμβούν και, συνεπώς, να μην αυξηθεί η θερμότητά τους. Στην περίπτωση αυτή, οι πιο θερμές μεταβλητές πιθανότατα θα είναι μεταβλητές είδους action, κάτι που δε συμβαδίζει με το στόχο λειτουργίας της παραμέτρου.

6.4.7 Παράμετροι κατάταξης μεταβλητών

Οι παράμετροι κατάταξης μεταβλητών λειτουργούν σε συνδυασμό με τις υπόλοιπες παραμέτρους και στοχεύουν στον περιορισμό της αναζήτησης στις θερμότερες μεταβλητές, δηλαδή στις μεταβλητές που κατατάσσονται στις ψηλότερες θέσεις της δομής schedule.decide, βάσει του αρχικού ευρετικού (VSIDS) που χρησιμοποιεί ο PrecoSAT. Ακολουθεί η περιγραφή τους.

-dtop <p>: Επιλέγεται η θερμότερη μεταβλητή που ικανοποιεί τα κριτήρια βάσει των υπόλοιπων παραμέτρων που δόθηκαν, και ανήκει στις θερμότερες <p> μεταβλητές. Αν δεν υπάρχει τέτοια μεταβλητή, επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή.

-jw <N> <k> <file>: Επιλέγεται η θερμότερη μεταβλητή που έχει το μεγαλύτερο βάρος Jeroslow-Wang (two-sided Jeroslow-Wang), ανάμεσα στις θερμότερες <N> μη ανατεθειμένες μεταβλητές γενικά και στις θερμότερες <N> μη ανατεθειμένες μεταβλητές που ικανοποιούν όλους τους περιορισμούς χρονικών επιπέδων και περιορισμούς είδους μεταβλητής. Ως τιμή της μεταβλητής επιλέγεται η πολικότητα του κατηγορήματος με το μεγαλύτερο βάρος. Αναλυτικότερα, αρχικά υπολογίζεται το βάρος (weight) κάθε μεταβλητής, το οποίο ισούται με το άθροισμα των βαρών των κατηγορημάτων (literals) που αντιστοιχούν στη μεταβλητή αυτή. Δηλαδή, προστίθενται

τα βάρη των κατηγορημάτων που αντιστοιχούν στις θετικές και αρνητικές εμφανίσεις της μεταβλητής στη φόρμουλα, όπως υπολογίζονται από τον αλγόριθμο Jeroslow-Wang. Έπειτα, γίνεται σύγκριση των βαρών των <N> πιο θερμών μεταβλητών όπως καθορίζονται από το αρχικό ευρετικό του PrecoSAT, με τα βάρη των πιο θερμών μεταβλητών από αυτές που θεωρούνται αποδεκτές σύμφωνα με τις υπόλοιπες παραμέτρους που δόθηκαν (πχ. –fonly). Ως μεταβλητή απόφασης (decision variable) επιλέγεται η μεταβλητή με το μεγαλύτερο βάρος. Αν το βάρος του θετικού κατηγορήματος που αντιστοιχεί στη μεταβλητή απόφασης είναι μεγαλύτερο από το βάρος του αρνητικού κατηγορήματος, τότε η μεταβλητή απόφασης παίρνει την τιμή TRUE. Διαφορετικά, παίρνει την τιμή FALSE. Στο αρχείο <file> τυπώνονται πληροφορίες σχετικά με τις μεταβλητές απόφασης και τα βάρη τους, ανά <k> αποφάσεις.

-jwg: Η παράμετρος –jwg δίνεται σε συνδυασμό με την παράμετρο –jw. Αν δοθεί η παράμετρος –jwg, τότε εφαρμόζεται το ευρετικό που περιγράφεται για την παράμετρο –jw, με τη διαφορά ότι για την επιλογή μεταβλητής λαμβάνονται υπόψη μόνο οι πιο «καλές» θερμές μεταβλητές, δηλαδή οι θερμές μεταβλητές που ικανοποιούν και τα κριτήρια των υπόλοιπων παραμέτρων (πχ. –fonly). Επομένως, δε γίνεται σύγκριση των βαρών των <N> πιο θερμών μεταβλητών όπως καθορίζονται από το αρχικό ευρετικό του PrecoSAT, και των βαρών των πιο θερμών μεταβλητών από αυτές που θεωρούνται αποδεκτές σύμφωνα με τις υπόλοιπες παραμέτρους που δόθηκαν (πχ. –fonly). Η συγκριση των βαρών των μεταβλητών γίνεται μόνο για τις μεταβλητές που θεωρούνται αποδεκτές σύμφωνα με τις υπόλοιπες παραμέτρους που δόθηκαν.

-fmaxleveldtop: Δίνεται προαιρετικά μαζί με την παράμετρο –dtop. Επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή του πιο «κοινού» χρονικού επιπέδου των υποψήφιων μεταβλητών απόφασης που ικανοποιούν όλους τους περιορισμούς χρονικών επιπέδων και περιορισμούς είδους μεταβλητής. Το πιο «κοινό» χρονικό επίπεδο είναι το χρονικό επίπεδο με το μεγαλύτερο άθροισμα πηλίκων της θέσης κατάταξης θερμότητας (=θέση στο schedule.decide) κάθε μεταβλητής εκείνου του χρονικού επιπέδου προς τη μέγιστη επιτρεπτή θέση στην κατάταξη θερμότητας (=παράμετρος του –dtop).

-varlevel <k>: Επιλέγεται η <k>-οστή θερμότερη μη ανατεθειμένη μεταβλητή. Η παράμετρος αυτή χρησιμεύει κυρίως για τη σύγκριση των πειραματικών αποτελεσμάτων μεταξύ της επιλογής μιας τυχαίας μεταβλητής, η οποία βρίσκεται σε συγκεκριμένη θέση της κατάταξης των μεταβλητών βάσει της θερμότητάς τους, και μιας μεταβλητής που μπορεί να είναι λιγότερο θερμή αλλά πληροί ορισμένα κριτήρια (πχ. είναι fact).

6.4.8 Παράμετροι επιλογής τιμής

Οι παράμετροι επιλογής τιμής σχετίζονται με την επιλογή της κατάλληλης τιμής για τη μεταβλητή απόφασης (phase selection). Ακολουθεί σύντομη περιγραφή τους.

-noopval1: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος actionNOOP και δεν υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-noopval2: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος actionNOOP και υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-noopval3: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος actionNOOP είτε υπάρχει είτε δεν υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-actionval1: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος action και δεν υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-actionval2: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος action και υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-actionval2: Ανατίθεται η τιμή TRUE στις μεταβλητές απόφασης που ανήκουν στο είδος action είτε υπάρχει είτε δεν υπάρχει αποθηκευμένη η τιμή της τελευταίας τους ανάθεσης.

-rsat: Η παράμετρος –rsat δίνεται σε συνδυασμό με την παράμετρο –jw, προαιρετικά. Αν δοθεί η παράμετρος –rsat μαζί με την παράμετρο –jw, τότε εφαρμόζεται το ευρετικό επιλογής μεταβλητής όπως περιγράφεται για την παράμετρο –jw, με τη διαφορά ότι χρησιμοποιείται ο αρχικός αλγόριθμος που χρησιμοποιεί ο PrecoSAT για την επιλογή

τιμής της μεταβλητής απόφασης (συνδυασμός Jeroslow-Wang και Rsat). Δηλαδή, η τιμή της μεταβλητής απόφασης ισούται με την τελευταία τιμή που της είχε ανατεθεί, αν υπάρχει αποθηκευμένη αυτή η τιμή. Διαφορετικά, η τιμή της μεταβλητής απόφασης επιλέγεται χρησιμοποιώντας τον αλγόριθμο Jeroslow-Wang. Δηλαδή, αν το βάρος του θετικού κατηγορήματος (literal) που αντιστοιχεί στη μεταβλητή απόφασης είναι μεγαλύτερο από το βάρος του αρνητικού κατηγορήματος τότε η μεταβλητή απόφασης παίρνει την τιμή TRUE, αλλιώς παίρνει την τιμή FALSE.

Σημείωση: Η παράμετρος –jw ανήκει εν μέρει και στην κατηγορία παραμέτρων επιλογής τιμής, διότι η τιμή της μεταβλητής απόφασης επιλέγεται χρησιμοποιώντας τον αλγόριθμο Jeroslow-Wang, αντί την προκαθορισμένη μέθοδο του PrecoSAT για την επιλογή τιμής της μεταβλητής απόφασης.

6.4.9 Συνδυασμός των παραμέτρων

Οι παράμετροι που υλοποιήθηκαν μπορούν να συνδυαστούν βάσει των πιο κάτω κανόνων:

1) Αν δοθεί μια παράμετρος για την οποία χρειάζονται πληροφορίες όπως το είδος ή το χρονικό επίπεδο κάθε μεταβλητής, τότε είναι απαραίτητο να καθοριστούν τα αρχεία εισόδου μέσω των αντίστοιχων παραμέτρων.

2) Τα -jw και –fmaxleveldtop δεν μπορούν να δοθούν ταυτόχρονα.

3) Όταν δοθεί η παράμετρος –varlevel, οποιαδήποτε άλλη νέα παράμετρος που αφορά τεχνική (δηλαδή όλες οι παράμετροι εκτός από τις παραμέτρους αρχείων εισόδου και τις παραμέτρους εκτύπωσης στατιστικών), αγνοείται.

4) Σχετικά με τις παραμέτρους χρονικών επιπέδων, μόνο μια από τις παραμέτρους χρονικών επιπέδων μπορεί να δοθεί σε κάθε εκτέλεση του PrecoSAT. Αν, εκτός από την παράμετρο χρονικών επιπέδων, δοθούν και παράμετροι σχετικές με το είδος της μεταβλητής (πχ. -fonly), τότε επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή εκείνου του είδους που ανήκει όσο πιο κοντά στα χρονικά όρια

γίνεται (ή μέσα στα χρονικά όρια). Αν δεν υπάρχουν άλλες υποψήφιες μεταβλητές απόφασης εκείνων των ειδών, τότε εφαρμόζεται η λειτουργία της παραμέτρου χρονικών επιπέδων σαν να μη δόθηκε καμία παράμετρος σχετική με το είδος της μεταβλητής.

5) Σχετικά με τις παραμέτρους είδους μεταβλητής, τα -initFacts και -goalFacts έχουν την ίδια προτεραιότητα. Δηλαδή, αν δοθούν μαζί, ο αλγόριθμος επιλέγει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact είτε της αρχικής είτε της τελικής κατάστασης. Επίσης, τα –fonly και –noopy έχουν την ίδια προτεραιότητα. Δηλαδή, αν δοθούν μαζί, επιλέγεται είτε fact είτε actionNOOP.

6) Αν δοθεί η παράμετρος –fonly μαζί με τις παραμέτρους –initFacts ή –goalFacts, τότε δίνεται προτεραιότητα στις παραμέτρους -initFacts και –goalFacts. Δηλαδή, μόνο αν δεν υπάρχουν άλλα facts της αρχικής ή τελικής κατάστασης, επιλέγεται ως μεταβλητή απόφασης κάποια μεταβλητή από τα υπόλοιπα facts.

7) Αν δοθεί παράμετρος χρονικών επιπέδων μαζί με παράμετρο είδους μεταβλητών και την παράμετρο –dtop, τότε επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή μέσα στα όρια της –dtop, η οποία ικανοποιεί τα κριτήρια σχετικά με το είδος της μεταβλητής και βρίσκεται μέσα στα χρονικά όρια ή όσο πιο κοντά στα χρονικά όρια γίνεται (δηλαδή ανήκει μέσα στα χρονικά όρια, ή στο ελάχιστο ή μέγιστο χρονικό επίπεδο στο οποίο υπάρχουν οι μεταβλητές του είδους της, αναλόγως με το ποιο από τα δύο είναι πιο κοντά στα χρονικά όρια όπως καθορίζονται από την παράμετρο χρονικών ορίων που δόθηκε).

8) Αν δοθούν ταυτόχρονα οι παράμετροι –dtop –fonly –goalFacts (ή –dtop –fonly –initFacts), μόνες τους ή με περισσότερες παραμέτρους, τότε, αν υπάρχουν μη ανατεθειμένα goalFacts (initFacts) στα όρια της –dtop, επιλέγεται το θερμότερο από αυτά. Διαφορετικά, επιλέγεται το θερμότερο μη ανατεθειμένο fact μέσα στα όρια της dtop. Αν δεν υπάρχει ούτε fact στα όρια της –dtop, τότε επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή.

9) Σχετικά με τις παραμέτρους επιλογής τιμής, μόνο μία από τις –noopval1, -noopval2 και -noopval3 μπορεί να δοθεί σε κάθε εκτέλεση του PrecoSAT. Το ίδιο ισχύει για τις παραμέτρους –actionval1, -actionval2 και -actionval3.

Στο σχήμα που ακολουθεί, φαίνονται όλες οι παράμετροι των οποίων οι λειτουργίες υλοποιήθηκαν κατά τη φάση της υλοποίησης.

-varf <file> -pddl <file>	-rnk <N> <file> -lrnk <N> <K> -pfx <N> <K> <file> -bInfo <N> <K> <file>	-tfvar <t> <v> -tlvar <t> <v> -tflvar <t> <v> -tnflvar <t> <v>
-noopval1 -noopval2 -noopval3 -actionval1 -actionval2 -actionval3	-noise <N> -tdlevel <N> -fdlevel <N> -trlevel <N> -frlevel <N>	-fonly -initFacts -goalFacts -noopy -noopn
-dtop <N> -varlevel <N> -fmaxleveldtop	-jw <N> <K> <file> -rsat -jwg	-fheat <N>

Σχήμα 6.1 Όλες οι παράμετροι των οποίων οι λειτουργίες υλοποιήθηκαν και ενσωματώθηκαν στον κώδικα του προτασιακού επιλυτή PrecoSAT.

6.5 Παραδείγματα εκτέλεσης

Ακολουθούν μερικά παραδείγματα εκτέλεσης του PrecoSAT με τις νέες παραμέτρους:

- ./precosat CNF -fonly -initFacts -goalFacts -varf VARFILE -pddl p01.pddl

Με την πιο πάνω εντολή, το ευρετικό απόφασης του PrecoSAT θα επιλέγει τη θερμότερη μη ανατεθειμένη μεταβλητη που είναι fact (-fonly) και ανήκει στην τελική (-goalFacts) ή αρχική κατάσταση (-initFacts). Αν δεν υπάρχουν άλλες τέτοιες μεταβλητές, τότε το ευρετικό απόφασης θα επιλέξει τη θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact. Όταν μείνουν μόνο μεταβλητές του είδους action ή actionNOOP, το ευρετικό απόφασης θα επιλέγει ως μεταβλητή απόφασης την πιο θερμή μη ανατεθειμένη μεταβλητή.

- ./precosat CNF –fonly dtop 1000 –noise 30 -varf VARFILE –pddl p01.pddl

Με την πιο πάνω εντολή, το ευρετικό απόφασης του PrecoSAT θα επιλέγει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact (-fonly), από τις 1000 πιο θερμές μεταβλητές (-dtop 1000). Αν καμία από τις 1000 πιο θερμές μεταβλητές δεν είναι fact, τότε το ευρετικό απόφασης θα επιλέξει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή, όπως θα έκανε το αρχικό ευρετικό του PrecoSAT. Η εφαρμογή του κανόνα επιλογής μεταβλητής fact από τις 1000 πιο θερμές μεταβλητές, θα εφαρμόζεται με πιθανότητα 70% (-noise 30). Με πιθανότητα περίπου 30% θα εφαρμόζεται το ευρετικό που προϋπήρχε στον PrecoSAT, δηλαδή θα επιλέγεται η πιο θερμή μη ανατεθειμένη μεταβλητή.

- ./precosat CNF -v -varf VARFILE –pddl p01.pddl

Με την πιο πάνω εντολή, το ευρετικό απόφασης του PrecoSAT δε θα ενσωματώσει καμία νέα τεχνική, διότι δε δίνεται καμία παράμετρος που αντιστοιχεί σε τεχνική. Στην περίπτωση αυτή, το ευρετικό απόφασης θα είναι το προϋπάρχον ευρετικό του

PrecoSAT. Δηλαδή, ως μεταβλητή απόφασης θα επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή. Όταν δοθεί η παράμετρος –v, η υλοποίηση της οποίας προϋπήρχε στον κώδικα του PrecoSAT, τυπώνονται στατιστικά στην οθόνη σχετικά με τη λειτουργία του PrecoSAT, ανά τακτά χρονικά διαστήματα. Με τη χρήση της παραμέτρου αυτής, ο χρήστης μπορεί να δει χρήσιμα στατιστικά στοιχεία όπως είναι ο αριθμός των αποφάσεων που έγιναν, ο αριθμός των συγκρούσεων, ο αριθμός των μεταβλητών στις οποίες ανατέθηκε τιμή και αριθμός των εκμαθημένων προτάσεων.

- ./precosat CNF -fonly -varf VARFILE –pddl p01.pddl

Με την πιο πάνω εντολή, το ευρετικό απόφασης επιλέγει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή είδους fact. Αν δεν υπάρχει καμία διαθέσιμη μεταβλητή του είδους fact, τότε το ευρετικό απόφασης επιλέγει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή, δηλαδή εφαρμόζεται το αρχικό ευρετικό του προτασιακού επιλυτή PrecoSAT.

- ./precosat CNF -varf VARFILE -pddl pfile12 -fonly -fdlevel 20 -tdlevel 50

Με την πιο πάνω εντολή, το ευρετικό απόφασης επιλέγει ως μεταβλητή απόφασης τη θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact, μόνο στα επίπεδα απόφασης (decision levels) 20 μέχρι 50. Στα υπόλοιπα επίπεδα απόφασης, εφαρμόζεται το ευρετικό απόφασης του αρχικού PrecoSAT, δηλαδή επιλέγεται ως μεταβλητή απόφασης η θερμότερη μη ανατεθειμένη μεταβλητή.

Κεφάλαιο 7

Πειραματική Ανάλυση

7.1 Εισαγωγή 73
7.2 Περιγραφή των συστημάτων 74
7.3 Πίνακες αποτελεσμάτων 75

7.1 Εισαγωγή

Σε αυτό το κεφάλαιο παρουσιάζονται τα αποτελέσματα των εκτελέσεων του προτασιακού επιλυτή PrecoSAT μέσω του SATPLAN, για διάφορα συστήματα (συνδυασμό τεχνικών), τα οποία περιγράφονται πιο κάτω. Από τα δοκιμαστικά πειράματα που γίνονταν κατά τη φάση της υλοποίησης, παρατηρήσαμε βελτίωση στην αποδοτικότητα της επίλυσης των προβλημάτων, όταν ως μεταβλητές απόφασης επιλέγονταν μεταβλητές του είδους fact. Για το λόγο αυτό, τα συστήματα που παρουσιάζονται πιο κάτω επικεντρώνονται σε συγκρίσεις εκτελέσεων στις οποίες χρησιμοποιούνται παράμετροι είδους μεταβλητής (πχ. –fonly).

Το χρονικό όριο (TIMEOUT) για τις εκτελέσεις είναι 3800 δευτερόλεπτα. Δηλαδή, αν χρησιμοποιώντας κάποιο συγκεκριμένο σύστημα, δεν επιλυθεί κάποιο πρόβλημα το πολύ σε 3800 δευτερόλεπτα, τότε θεωρούμε ότι το σύστημα δεν μπόρεσε να λύσει το πρόβλημα.

Ακολουθεί η περιγραφή των συστημάτων και οι πίνακες με τα αποτελέσματα των εκτελέσεων για κάθε σύστημα.

7.2 Περιγραφή των συστημάτων

1) Σύστημα PLAIN

   Το σύστημα PLAIN είναι το σύστημα που δεν ενσωματώνει καμία νέα τεχνική, δηλαδή οι μεταβλητές απόφασης επιλέγονται χρησιμοποιώντας το αρχικό ευρετικό του προτασιακού επιλυτή PrecoSAT.

2) Σύστημα FONLY-TOP1000

   Για το σύστημα FONLY-TOP1000, επιχειρούμε να επιλύσουμε τα προβλήματα εκτελώντας τον PrecoSAT με τις παραμέτρους [-fonly –dtop 1000]. Σύμφωνα με το σύστημα αυτό, ως μεταβλητή απόφασης επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή του είδους fact, η οποία ανήκει στις θερμότερες 1000 μεταβλητές. Αν δεν υπάρχει μεταβλητή είδους fact στις θερμότερες 1000 μεταβλητές, τότε επιλέγεται ως μεταβλητή απόφασης η θερμότερη μη ανατεθειμένη μεταβλητή, δηλαδή εφαρμόζεται το αρχικό ευρετικό του PrecoSAT.

3) Σύστημα FONLY-TOP3000

   Για το σύστημα FONLY-TOP3000, επιχειρούμε να επιλύσουμε τα προβλήματα εκτελώντας τον PrecoSAT με τις παραμέτρους [-fonly –dtop 3000]. Σύμφωνα με το σύστημα αυτό, ως μεταβλητή απόφασης επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή του είδους fact, η οποία ανήκει στις θερμότερες 3000 μεταβλητές. Αν δεν υπάρχει μεταβλητή είδους fact στις θερμότερες 3000 μεταβλητές, τότε επιλέγεται ως μεταβλητή απόφασης η θερμότερη μη ανατεθειμένη μεταβλητή, δηλαδή εφαρμόζεται το αρχικό ευρετικό του PrecoSAT.

4) Σύστημα FONLY-UNL

   Για το σύστημα FONLY-UNL, επιχειρούμε να επιλύσουμε τα προβλήματα εκτελώντας τον PrecoSAT με τις παραμέτρους [-fonly]. Σύμφωνα με το σύστημα αυτό, ως μεταβλητή απόφασης επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή του είδους fact. Αν δεν υπάρχει μεταβλητή είδους fact,

τότε επιλέγεται ως μεταβλητή απόφασης η θερμότερη μη ανατεθειμένη μεταβλητή, δηλαδή εφαρμόζεται το αρχικό ευρετικό του PrecoSAT.

5) Σύστημα GOAL-TOP3000

Για το σύστημα GOAL-TOP3000, επιχειρούμε να επιλύσουμε τα προβλήματα εκτελώντας τον PrecoSAT με τις παραμέτρους [-goalFacts –dtop 3000]. Σύμφωνα με το σύστημα αυτό, ως μεταβλητή απόφασης επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή του είδους goalFact, η οποία ανήκει στις θερμότερες 3000 μεταβλητές. Αν δεν υπάρχει μεταβλητή είδους goalFact στις θερμότερες 3000 μεταβλητές, τότε επιλέγεται ως μεταβλητή απόφασης η θερμότερη μη ανατεθειμένη μεταβλητή, δηλαδή εφαρμόζεται το αρχικό ευρετικό του PrecoSAT.

7.3 Πίνακες αποτελεσμάτων

Για την πειραματική ανάλυση χρησιμοποιήθηκαν 5 τύποι πινάκων, οι οποίοι επεξηγούνται πιο κάτω:

1) Πίνακας τύπου Α

Στον πίνακα τύπου Α παρουσιάζεται ο συνολικός αριθμός των μεταβλητών του προβλήματος, καθώς και ο αριθμός και το ποσοστό εμφάνισης κάθε είδους μεταβλητών (facts, actions, actionNOOPs) για τα συστήματα FONLY-TOP1000 και PLAIN, στο δυσκολότερο πρόβλημα ανά domain που λύθηκε και από τα δύο συστήματα, για το επίπεδο πριν το επίπεδο λύσης (solution_level -1). Οι πληροφορίες αυτές προέρχονται από το αρχείο CNF που παράγει το σύστημα SATPLAN. Τα ποσοστά περικλείονται με παρενθέσεις.

2) Πίνακας τύπου Β

Στον πίνακα τύπου Β παρουσιάζεται ο αριθμός προβλημάτων ανά domain που λύθηκαν από κάθε σύστημα.

3) Πίνακας τύπου Γ

Στον πίνακα τύπου Γ παρουσιάζεται το άθροισμα των χρόνων που χρειάστηκαν, για να λυθούν όλα τα προβλήματα ανά domain, μόνο για τα προβλήματα που λύθηκαν και από τα δύο συστήματα που συγκρίνονται.

4) Πίνακας τύπου Δ

Στον πίνακα τύπου Δ παρουσιάζεται ο χρόνος και ο αριθμός των χρονικών επιπέδων (levels) που χρειάστηκαν για την επίλυση του πιο δύσκολου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000 και το σύστημα PLAIN (και από τα δύο συστήματα).

5) Πίνακας τύπου Ε

Στον πίνακα τύπου Ε1 παρουσιάζεται ο αριθμός των συγκρούσεων (conflicts), μεταδόσεων (propagations), αποφάσεων (decisions), μεταδόσεων ανά απόφαση (propagations per decision) και μεταδόσεων ανά απόφαση προς τον αριθμό των μεταβλητών στις οποίες δεν ανατέθηκε τιμή κατά την προεπεξεργασία της φόρμουλας από τον PrecoSAT ((propagations per decision) /variables), τα οποία έγιναν κατά την επίλυση του δυσκολότερου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000 και το σύστημα PLAIN (και από τα δύο συστήματα), για το επίπεδο πριν το επίπεδο λύσης (solution_level -1).

6) Πίνακας τύπου Ζ

Στον πίνακα τύπου Ζ παρουσιάζεται το ποσοστό των μεταβλητών, γενικά και ανά είδος (facts, actions, actionNOOPs), που ήταν μεταβλητές απόφασης στα επίπεδα 1 μέχρι 50 του δέντρου αναζήτησης, κατά την επίλυση του δυσκολότερου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000 και το σύστημα PLAIN (και από τα δύο συστήματα), για το επίπεδο πριν το επίπεδο λύσης (solution_level -1).

7) Πίνακας τύπου Η

Στον πίνακα τύπου Η παρουσιάζεται το ποσοστό των μεταβλητών, γενικά και ανά είδος (facts, actions, actionNOOPs), που ήταν μεταβλητές απόφασης στα επίπεδα 51 μέχρι 100 του δέντρου αναζήτησης, κατά την επίλυση του δυσκολότερου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000

και το σύστημα PLAIN (και από τα δύο συστήματα), για το επίπεδο πριν το επίπεδο λύσης (solution_level -1).

8) Πίνακας τύπου Θ

Στον πίνακα τύπου Θ παρουσιάζεται το ποσοστό των μεταβλητών, γενικά και ανά είδος (facts, actions, actionNOOPs), που ήταν μεταβλητές απόφασης στα επίπεδα 1 μέχρι 50 του δέντρου αναζήτησης, μέχρι την τελευταία επανεκκίνηση, κατά την επίλυση του δυσκολότερου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000 και το σύστημα PLAIN (και από τα δύο συστήματα), για το επίπεδο πριν το επίπεδο λύσης (solution_level -1).

9) Πίνακας τύπου Ι

Στον πίνακα τύπου Ι παρουσιάζεται το ποσοστό των μεταβλητών, γενικά και ανά είδος (facts, actions, actionNOOPs), που ήταν μεταβλητές απόφασης στα επίπεδα 51 μέχρι 100 του δέντρου αναζήτησης, μέχρι την τελευταία επανεκκίνηση, κατά την επίλυση του δυσκολότερου προβλήματος που λύθηκε από το σύστημα FONLY-TOP1000 και το σύστημα PLAIN (και από τα δύο συστήματα), για το επίπεδο πριν το επίπεδο λύσης (solution_level -1).

10) Πίνακας τύπου Κ

Στον πίνακα τύπου Κ παρουσιάζεται το μέσο ύψος (average height) του δέντρου αναζήτησης, για το δυσκολότερο πρόβλημα που λύθηκε από τα συστήματα FONLY-TOP1000 και PLAIN (και από τα δύο συστήματα).

11) Πίνακας τύπου Λ

Στον πίνακα τύπου Λ παρουσιάζεται το μέγιστο ύψος (maximum height) του δέντρου αναζήτησης, για το δυσκολότερο πρόβλημα που λύθηκε από τα συστήματα FONLY-TOP1000 και PLAIN (και από τα δύο συστήματα).

Σημείωση: Το σύμβολο «*» σε μια θέση του πίνακα σημαίνει ότι η τιμή για εκείνη τη θέση δεν ορίζεται ή δεν υπάρχει.

Ακολουθούν οι πίνακες.

Πίνακας 7.1 Πίνακας τύπου Α για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Hardest problem (number)	Επίπεδο (Level)	Αριθμός μεταβλητών	Αριθμός Facts, (%)	Αριθμός Actions, (%)	Αριθμός ActionNOOPs, (%)
BAR	3	29	11,356	2,034 (17.91)	7,362 (64.83)	1,960 (17.26)
ELEV	13	18	41,625	4,438 (10.66)	33,047 (79.39)	4,140 (9.95)
PATH	7	12	9,323	2,500 (26.82)	4,585 (49.18)	2,238 (24.01)
PIPES	15	19	32,647	3,747 (11.48)	25,402 (77.81)	3,498 (10.71)
SATEL	13	12	61,389	3,439 (5.6)	54,834 (89.32)	3,116 (5.08)
SCAN	27	11	59,097	692 (1.17)	57,785 (97.78)	620 (1.05)
STACKS	5	22	3,520	756 (21.48)	2,045 (58.1)	719 (20.43)
STORE	16	10	10,729	1,721 (16.04)	7,515 (70.04)	1,493 (13.92)
TRANS	14	18	52,896	3,739 (7.07)	45,721 (86.44)	3,436 (6.5)
TRUCKS	13	23	124,836	16,145 (12.93)	93,363 (74.79)	15,328 (12.28)
VISIT	7	35	11,555	3,156 (27.31)	5,340 (46.21)	3,059 (26.47)

Πίνακας 7.2 Πίνακας τύπου Β για τα συστήματα PLAIN, FONLY-TOP1000, FONLY-TOP3000, FONLY-UNL και GOAL-TOP3000.

Domain	Αριθμός προβλημάτων που λύθηκαν				
	FONLY-TOP1000	FONLY-TOP3000	FONLY-UNL	GOAL-TOP3000	PLAIN
BAR	4	4	4	4	8
ELEV	11	11	11	9	14
PATH	8	8	8	8	16
PIPES	21	21	21	17	23
SATEL	17	16	16	16	18
SCAN	18	18	18	16	18
STACKS	5	5	5	5	5
STORE	16	16	16	15	16
TRANS	12	12	12	9	13
TRUCKS	10	10	10	8	11
VISIT	10	10	10	10	12

Πίνακας 7.3 Πίνακας τύπου Γ για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Άθροισμα χρόνων (sec.)	
	FONLY-TOP1000	PLAIN
BAR	1,396.25	136.14
ELEV	4,448.96	281.22
PATH	1,110.94	9.95
PIPES	5,910.25	1,552.32
SATEL	5,265.40	380.06
SCAN	308.83	251.11
STACKS	1,386.52	60.04
STORE	630.43	354.86
TRANS	5,633.77	394.02
TRUCKS	6,020.70	1,767.13
VISIT	219.82	54.54

Πίνακας 7.4 Πίνακας τύπου Γ για τα συστήματα FONLY-TOP3000 και PLAIN.

Domain	Άθροισμα χρόνων (sec.)	
	FONLY-TOP3000	PLAIN
BAR	1,403.64	136.14
ELEV	4,565.85	281.22
PATH	1,109.62	9.95
PIPES	5,786.63	1,552.32
SATEL	2,687.18	279.00
SCAN	304.93	251.11
STACKS	1,374.03	60.04
STORE	633.87	354.86
TRANS	4,656.95	394.02
TRUCKS	6,021.52	1,767.13
VISIT	221.08	54.54

Πίνακας 7.5 Πίνακας τύπου Γ για τα συστήματα FONLY-UNL και PLAIN.

Domain	Άθροισμα χρόνων (sec.)	
	FONLY-UNL	PLAIN
BAR	1,404.63	136.14
ELEV	4,599.76	281.22
PATH	695.56	9.95
PIPES	5,892.11	1,552.32
SATEL	3,452.27	279.00
SCAN	304.51	251.11
STACKS	1,271.64	60.04
STORE	640.74	354.86
TRANS	4,657.17	394.02
TRUCKS	6,002.73	1,767.13
VISIT	219.92	54.54

Πίνακας 7.6 Πίνακας τύπου Γ για τα συστήματα GOAL-TOP3000 και PLAIN.

Domain	Άθροισμα χρόνων (sec.)	
	GOAL-TOP3000	PLAIN
BAR	1,450.58	136.14
ELEV	2,476.91	53.45
PATH	888.06	9.95
PIPES	3,513.42	907.05
SATEL	3,872.86	279.00
SCAN	229.33	214.29
STACKS	1,916.38	60.04
STORE	84.23	16.10
TRANS	152.03	27.92
TRUCKS	3,557.73	390.55
VISIT	813.48	54.54

Πίνακας 7.7 Πίνακας τύπου Δ για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Πρόβλημα	Επίπεδο λύσης	Χρόνος (sec.)	
			FONLY–TOP1000	PLAIN
BAR	3	30	306.05	32.28
ELEV	13	19	2,389.83	116.24
PATH	7	13	979.44	4.39
PIPES	15	20	2,081.81	333.31
SATEL	13	13	2,302.54	101.06
SCAN	27	12	98.95	94.78
STACKS	5	23	249.70	13.16
STORE	16	11	578.46	338.76
TRANS	14	19	3,324.04	153.62
TRUCKS	13	24	3,421.08	1,251.32
VISIT	7-half	36	206.03	45.07

Πίνακας 7.8 Πίνακας τύπου Ε για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Συγκρούσεις	Μεταδόσεις	Αποφάσεις	Μεταδόσεις ανά απόφαση	Μεταδόσεις ανά απόφαση προς αριθμό μεταβλητών
FONLY					
BAR	45,639	25,510,063	99,638	256.03	0.04
ELEV	63,623	117,662,113	164,360	715.88	0.02
PATH	34,524	10,693,193	1,555,115	6.88	0
PIPES	155,656	237,787,069	401,712	591.93	0.02
SATEL	47,789	128,769,014	105,783	1,217.29	0.03
SCAN	1	37,635	0	*	*
STACKS	415,499	76,208,303	742,328	102.66	0.04
STORE	300,500	246,922,190	645,760	382.37	0.05
TRANS	69,615	316,360,260	204,792	1,544.79	0.04
TRUCKS	328,385	355,541,014	734,528	484.04	0.04
VISIT	44,600	35,687,471	151,982	234.81	0.04
PLAIN					
BAR	48,211	29,122,998	222,745	130.75	0.02
ELEV	67,320	128,119,450	506,822	252.79	0.01
PATH	5,737	2,505,794	16,011	156.5	0.04
PIPES	193,295	247,097,832	2,562,859	96.41	0
SATEL	44,324	97,866,223	208,345	469.73	0.01
SCAN	1	37,635	0	*	*
STACKS	68,154	12,344,951	192,522	64.12	0.03
STORE	1,033,149	723,867,264	6,773,985	106.86	0.02
TRANS	60,954	286,955,548	614,557	466.93	0.01
TRUCKS	782,561	854,320,929	5,953,535	143.5	0.01
VISIT	80,747	59,856,846	715,877	83.61	0.02

Πίνακας 7.9 Πίνακας τύπου Z για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Fixed (%)	Facts (%)	Actions (%)	ActionNOOPs (%)
FONLY				
BAR	98.21	99.97	0.03	0
ELEV	95.7	99.96	0.04	0
PATH	16.07	99.96	0.02	0.01
PIPES	89.92	99.83	0.17	0
SATEL	91.94	99.66	0.34	0
SCAN	*	*	*	*
STACKS	99.87	98.53	0.78	0.69
STORE	96.17	99.97	0.03	0
TRANS	90.2	99.84	0.16	0
TRUCKS	96.77	99.96	0.04	0
VISIT	79.27	99.98	0.02	0
PLAIN				
BAR	58.5	28.29	71.49	0.22
ELEV	41.64	32.07	67.93	0
PATH	87.75	17.3	55.61	27.09
PIPES	17.62	14.53	85.41	0.05
SATEL	40.78	12.93	87.07	0
SCAN	*	*	*	*
STACKS	85.91	25.55	51.51	22.94
STORE	26.23	20.77	79.22	0
TRANS	37.03	39.31	60.69	0
TRUCKS	19.9	18.86	81.13	0.02
VISIT	23.06	11.99	88.01	0

Πίνακας 7.10 Πίνακας τύπου Η για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Fixed (%)	Facts (%)	Actions (%)	ActionNOOPs (%)
FONLY				
BAR	1.79	100	0	0
ELEV	3.88	99.97	0.03	0
PATH	21.26	99.96	0.02	0.01
PIPES	8.51	97.61	2.39	0
SATEL	4.77	87.11	12.89	0
SCAN	*	*	*	*
STACKS	0.13	87.54	8.38	4.08
STORE	3.82	99.97	0.03	0
TRANS	6.08	97.38	2.62	0
TRUCKS	3.12	99.94	0.06	0
VISIT	18.4	99.98	0.02	0
PLAIN				
BAR	31.88	27.01	72.81	0.18
ELEV	27.78	20.57	79.43	0
PATH	11.22	11.14	64.7	24.16
PIPES	18	13.17	86.76	0.07
SATEL	43.56	13.08	86.92	0
SCAN	*	*	*	*
STACKS	13.29	25.45	54.25	20.3
STORE	27.71	17.53	82.46	0
TRANS	23.41	22.09	77.91	0
TRUCKS	20.32	20.03	79.95	0.01
VISIT	29.28	11.91	88.09	0

Πίνακας 7.11 Πίνακας τύπου Θ για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Fixed (%)	Facts (%)	Actions (%)	ActionNOOPs (%)
FONLY				
BAR	98.2	99.97	0.03	0
ELEV	95.69	99.96	0.04	0
PATH	16.06	99.96	0.02	0.01
PIPES	89.92	99.83	0.17	0
SATEL	91.93	99.66	0.34	0
SCAN	*	*	*	*
STACKS	99.87	98.53	0.78	0.69
STORE	96.16	99.97	0.03	0
TRANS	90.2	99.84	0.16	0
TRUCKS	96.77	99.96	0.04	0
VISIT	79.25	99.98	0.02	0
PLAIN				
BAR	58.49	28.29	71.49	0.22
ELEV	41.64	32.07	67.93	0
PATH	87.73	17.31	55.61	27.07
PIPES	17.62	14.53	85.42	0.05
SATEL	40.77	12.93	87.07	0
SCAN	*	*	*	*
STACKS	85.9	25.54	51.52	22.94
STORE	26.23	20.77	79.23	0
TRANS	37.01	39.28	60.72	0
TRUCKS	19.9	18.86	81.13	0.02
VISIT	23.04	11.98	88.02	0

Πίνακας 7.12 Πίνακας τύπου Ι για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Fixed (%)	Facts (%)	Actions (%)	ActionNOOPs (%)
FONLY				
BAR	1.8	100	0	0
ELEV	3.88	99.97	0.03	0
PATH	21.26	99.96	0.02	0.01
PIPES	8.51	97.61	2.39	0
SATEL	4.78	87.11	12.89	0
SCAN	*	*	*	*
STACKS	0.13	87.54	8.38	4.08
STORE	3.83	99.97	0.03	0
TRANS	6.08	97.38	2.62	0
TRUCKS	3.12	99.94	0.06	0
VISIT	18.41	99.98	0.02	0
PLAIN				
BAR	31.89	27.01	72.81	0.18
ELEV	27.79	20.57	79.43	0
PATH	11.23	11.14	64.7	24.16
PIPES	18	13.17	86.76	0.07
SATEL	43.57	13.08	86.92	0
SCAN	*	*	*	*
STACKS	13.29	25.45	54.25	20.3
STORE	27.72	17.53	82.46	0
TRANS	23.42	22.09	77.91	0
TRUCKS	20.32	20.03	79.95	0.01
VISIT	29.28	11.91	88.09	0

Πίνακας 7.13 Πίνακας τύπου Κ για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Μέσο ύψος (sec.)	
	FONLY-TOP1000	PLAIN
BAR	15.9	49.5
ELEV	19.8	83.9
PATH	129.3	27.5
PIPES	30.2	218.3
SATEL	26.4	71
SCAN	0	0
STACKS	17.5	29
STORE	21.8	111.3
TRANS	50	133.1
TRUCKS	22.4	157.7
VISIT	34.6	112.9

Πίνακας 7.14 Πίνακας τύπου Λ για τα συστήματα FONLY-TOP1000 και PLAIN.

Domain	Μέγιστο ύψος (sec.)	
	FONLY-TOP1000	PLAIN
BAR	89	271
ELEV	184	846
PATH	319	160
PIPES	879	2,910
SATEL	359	1,128
SCAN	0	0
STACKS	90	170
STORE	118	1,002
TRANS	2,364	2,042
TRUCKS	323	1,227
VISIT	347	710

Κεφάλαιο 8

Συμπεράσματα

8.1 Σύνοψη και τελικά συμπεράσματα 88
8.2 Επεκτάσεις και Μελλοντική Εργασία 92

8.1 Σύνοψη και τελικά συμπεράσματα

Οι προτασιακοί επιλυτές είναι συστήματα που προσπαθούν να επιλύσουν Προβλήματα Ικανοποιησιμότητας, δηλαδή να αναθέσουν τιμές στις μεταβλητές μιας φόρμουλας σε Συζευκτική Κανονική Μορφή, έτσι ώστε η φόρμουλα να ικανοποιείται. Οι προτασιακοί επιλυτές αποτελούν μια δημοφιλή μέθοδο επίλυσης προβλημάτων από διάφορα επιστημονικά πεδία, διότι τα προβλήματα μπορούν να μετατραπούν, μέσω ενός πολυωνυμικού μετασχηματισμού, σε φόρμουλα Συζευκτικής Κανονικής Μορφής, και να λυθούν αποδοτικά με τη βοήθεια ενός προτασιακού επιλυτή. Η λύση κάποιου προβλήματος που λύνεται με αυτό τον τρόπο, είναι η έξοδος του προτασιακού επιλυτή, όταν αποκωδικοποιηθεί.

Η επίλυση προβλημάτων που ανήκουν σε συγκεκριμένα επιστημονικά πεδία, από γενικούς προτασιακούς επιλυτές προβλημάτων, είναι πιθανότατα λιγότερο αποδοτική από την επίλυση προβλημάτων μέσω εξειδικευμένων προτασιακών επιλυτών για κάθε κατηγορία προβλημάτων, οι οποίοι αξιοποιούν τα χαρακτηριστικά του συγκεκριμένου είδους προβλημάτων, για να δώσουν γρηγορότερες και αποδοτικότερες λύσεις. Με αυτό το κίνητρο, στην παρούσα διπλωματική εργασία προσπαθήσαμε να εξειδικεύσουμε ένα σημαντικό κομμάτι των προτασιακών επιλυτών, το ευρετικό απόφασης, για την κατηγορία των προβλημάτων Προγραμματισμού Δράσης.

Ο Προγραμματισμός Δράσης είναι η εύρεση μιας ακολουθίας από δράσεις, από ένα προκαθορισμένο σύνολο επιτρεπτών δράσεων, η εφαρμογή των οποίων οδηγεί στην επίτευξη κάποιου στόχου. Στον Προγραμματισμό Δράσης, γίνεται μεγάλη προσπάθεια για την εύρεση μιας τέτοιας ακολουθίας, αν υπάρχει, χρησιμοποιώντας το μικρότερο δυνατό αριθμό χρονικών επιπέδων, έτσι ώστε να μειώνεται το κόστος της λύσης σε χρόνο και πόρους.

Για την επίλυση προβλημάτων Προγραμματισμού Δράσης έχουν αναπτυχθεί διάφορα συστήματα. Ένα από αυτά είναι το σύστημα SATPLAN, το οποίο αποτελεί ένα ολοκληρωμένο σύστημα επίλυσης προβλημάτων Προγραμματισμού Δράσης που στηρίζεται στην ιδέα μετασχηματισμού του προβλήματος Προγραμματισμού Δράσης σε πρόβλημα Προτασιακής Ικανοποιησιμότητας [20]. Το σύστημα SATPLAN κατασκευάζει το γράφημα δράσεων μέχρι κάποιο συγκεκριμένο χρονικό επίπεδο και μεταφράζει τους περιορισμούς που προκύπτουν σε Συζευκτική Κανονική Μορφή (CNF). Στη συνέχεια, το αρχείο που περιέχει τις προτάσεις σε Συζευκτική Κανονική Μορφή, δίνεται ως είσοδος σε κάποιον προτασιακό επιλυτή, ο οποίος καλείται να βρει μια ανάθεση τιμών στις μεταβλητές, η οποία ικανοποιεί όλες οι προτάσεις. Αν το πρόβλημα μπορεί να ικανοποιηθεί, τότε το σύστημα SATPLAN παράγει ως έξοδο τη λύση του αρχικού προβλήματος. Αν ο προτασιακός επιλυτής δεν μπορέσει να βρει ανάθεση που ικανοποιεί όλες τις προτάσεις, τότε το σύστημα SATPLAN δημιουργεί το γράφημα δράσεων μέχρι το επόμενο επίπεδο (κ+1) και η διαδικασία επαναλαμβάνεται.

Στην παρούσα διπλωματική εργασία, εξετάστηκαν διάφορες ευρετικές μέθοδοι προτασιακών επιλυτών για την επίλυση προβλημάτων Προγραμματισμού Δράσης. Οι ευρετικές μέθοδοι που εξετάστηκαν, υλοποιήθηκαν και ενσωματώθηκαν στον κώδικα του προτασιακού επιλυτή PrecoSAT, και μελετήθηκαν πειραματικά. Για την πειραματική ανάλυση χρησιμοποιήθηκε και το σύστημα SATPLAN, το οποίο καλούσε τον τροποποιημένο επιλυτή PrecoSAT για την επίλυση των προβλημάτων.

Τα χαρακτηριστικά των προβλημάτων Προγραμματισμού Δράσης στα οποία βασίζονται οι εξειδικευμένες τεχνικές που υλοποιήθηκαν είναι το είδος των μεταβλητών (fact, action, actionNOOP), το χρονικό επίπεδο στο οποίο ανήκουν, δηλαδή το time level όπως καθορίζεται από το σύστημα SATPLAN και, στην

περίπτωση που οι μεταβλητές είναι facts, κατά πόσο ανήκουν στην αρχική ή στην τελική κατάσταση (goalFacts, initFacts).

Από τις πειραματικές μετρήσεις που έγιναν, μπορούν να εξαχθούν κάποια συμπεράσματα για την αποδοτικότητα των τεχνικών. Η πειραματική ανάλυση έδειξε ότι η λειτουργία της παραμέτρου –fonly, σε συνδυασμό με την παράμετρο –dtop, γενικά μειώνει τον αριθμό αποφάσεων που γίνονται κατά την επίλυση των περισσότερων προβλημάτων και αυξάνει τον αριθμό μεταδόσεων ανά απόφαση. Τα συστήματα FONLY-TOP1000 και FONLY-TOP3000 φαίνεται ότι είναι τα πιο αποδοτικά από τα συστήματα που εξετάστηκαν, διότι έλυσαν τον μεγαλύτερο αριθμό προβλημάτων, και έχουν μικρότερο άθροισμα χρόνων επίλυσης από άλλα συστήματα που έλυσαν τον ίδιο αριθμό προβλημάτων.

Με εξαίρεση το δυσκολότερο πρόβλημα στα domains PATH και STACK, τα οποία έχουν τον μικρότερο συνολικό αριθμό μεταβλητών, ο αριθμός μεταδόσεων ανά απόφαση είναι μεγαλύτερος, όταν χρησιμοποιείται το σύστημα FONLY-TOP1000. Επιπρόσθετα, για το δυσκολότερο πρόβλημα ανά domain, το μέσο ύψος του δέντρου αναζήτησης είναι μικρότερο σε όλα τα domains, εκτός από το PATH. Το μέγιστο ύψος του δέντρου αναζήτησης είναι επίσης μικρότερο για το δυσκολότερο πρόβλημα των περισσότερων domains. Το γεγονός αυτό δείχνει ότι αν «κρατήσουμε» μόνο τις αποδοτικές τεχνικές, όπως είναι το σύστημα FONLY-TOP1000, και βελτιώσουμε την ιδέα της υλοποίησης αφαιρώντας τη χρονοβόρα προεργασία για την ανανέωση των μετρητών πριν από κάθε απόφαση, ίσως καταφέρουμε να μειώσουμε το χρόνο επίλυσης των προβλημάτων.

Συν τοις άλλοις, τα αποτελέσματα των πειραμάτων δείχνουν ότι στο σύστημα PLAIN το 50-90% των μεταβλητών απόφασης είναι μεταβλητές του είδους action, ενώ είναι πολύ λιγότερες οι μεταβλητές απόφασης που ανήκουν στο είδος actionNOOP (περίπου 0.02%). Εξαίρεση αποτελούν τα δυσκολότερα προβλήματα των domains PATH και STACKS, στα οποία ο αριθμός αποφάσεων αυξάνεται όταν εφαρμόζεται το σύστημα FONLY-TOP1000. Στα προβλήματα αυτών των domains, γίνονται σχετικά πολλές (20% περίπου) αναθέσεις μεταβλητών που ανήκουν στο είδος actionNOOP. Βάσει των

πιο πάνω, φαίνεται ότι η τεχνική FONLY-TOP1000 δεν είναι αποδοτική για την επίλυση μικρών προβλημάτων.

Συμπερασματικά, φαίνεται ότι καμία από τις τεχνικές που εξετάσαμε δε μειώνει το χρόνο επίλυσης των προβλημάτων. Αντίθετα, τα πειραματικά αποτελέσματα έδειξαν ότι η εφαρμογή των τεχνικών που υλοποιήθηκαν, χρησιμοποιώντας την ιδέα υλοποίησης που περιγράφεται στο ομώνυμο κεφάλαιο, αυξάνει σημαντικά το χρόνο επίλυσης των προβλημάτων. Το γεγονός όμως ότι κάποιες από τις τεχνικές μειώνουν τον αριθμό αποφάσεων που απαιτούνται για την επίλυση των περισσότερων προβλημάτων, φανερώνει ότι η εφαρμογή εξειδικευμένων τεχνικών, είτε για τα προβλήματα Προγραμματισμού Δράσης είτε για άλλα είδη προβλημάτων, μπορεί να συμβάλει στην ανάπτυξη αποδοτικότερων επιλυτών, εξειδικευμένων για συγκεκριμένα είδη προβλημάτων.

8.2 Επεκτάσεις και Μελλοντική Εργασία

Η παρούσα διπλωματική εργασία θα μπορούσε να επεκταθεί με ποικίλους τρόπους.

Μια επέκταση θα μπορούσε να είναι η αξιοποίηση της γνώσης που αποκτά ο προτασιακός επιλυτής κατά την (αποτυχημένη) προσπάθεια επίλυσης ενός προβλήματος μέχρι k χρονικά επίπεδα, για την επίλυση του ίδιου προβλήματος σε περισσότερα (πχ. k+1) χρονικά επίπεδα.

Όπως αναφέρθηκε στον αλγόριθμο λειτουργίας του συστήματος SATPLAN, ο προτασιακός επιλυτής καλείται να βρει μια ανάθεση τιμών στις μεταβλητές της φόρμουλας που αντιστοιχεί στο γράφημα δράσεων μέχρι κάποιο χρονικό επίπεδο k, η οποία ικανοποιεί τη φόρμουλα. Αν αυτό δεν είναι εφικτό, το σύστημα SATPLAN κατασκευάζει το γράφημα δράσεων μέχρι το επόμενο χρονικό επίπεδο k+1, και καλεί τον προτασιακό επιλυτή, για να επιλύσει το πρόβλημα.

Ο προτασιακός επιλυτής κάθε φορά αντιμετωπίζει το πρόβλημα σαν ένα εντελώς ανεξάρτητο, καινούριο πρόβλημα, και δε χρησιμοποιεί καθόλου τη γνώση που απέκτησε κατά την προσπάθεια επίλυσης του ίδιου προβλήματος σε λιγότερα χρονικά επίπεδα. Η γνώση αυτή θα μπορούσε να είναι το σύνολο των ενεργειών του

προτασιακού επιλυτή μετά την τελευταία επανεκκίνηση, οι οποίες σχετίζονται με τις μεταβλητές απόφασης που επιλέχθηκαν, τις τιμές των μεταβλητών απόφασης, καθώς και τις εκμαθημένες προτάσεις που παρήχθησαν.

Οι πληροφορίες σχετικά με τις αποφάσεις και τις εκμαθημένες προτάσεις του προβλήματος θα μπορούσαν να χρησιμοποιηθούν με διάφορους τρόπους. Για παράδειγμα, οι εκμαθημένες προτάσεις (ή μέρος αυτών) θα μπορούσαν να αποθηκευτούν σε κατάλληλες δομές δεδομένων πριν αρχίσει η αναζήτηση και να αποτρέψουν την αναζήτηση λύσης σε μονοπάτια που οδηγούν σε σύγκρουση.

Η γνώση σχετικά με τις αποφάσεις θα μπορούσε να κατευθύνει την πορεία του επιλυτή προς το μονοπάτι που ακολούθησε για την απόδειξη της μη ικανοποιησιμότητας του προβλήματος σε λιγότερα χρονικά επίπεδα (το οποίο μπορεί να προέκυψε μετά από εκατομμύρια συγκρούσεις, αποφάσεις και αναθέσεις), εξοικονομώντας πολύτιμο χρόνο. Επίσης, οι μεταβλητές απόφασης θα μπορούσαν να ταξινομηθούν βάσει κάποιου βάρους που είναι ανάλογο του αριθμού αναθέσεων (propagations) που προκάλεσαν, και, με τη χρήση κάποιου φράγματος (threshold) για τον αριθμό των μεταβλητών απόφασης ή για το βάρος τους, να χρησιμοποιηθούν ως μεταβλητές απόφασης του νέου προβλήματος οι μεταβλητές απόφασης με το μεγαλύτερο βάρος.

Για την αποφυγή της πλήρους αντιγραφής της προηγούμενης πορείας του επιλυτή, θα μπορούσε να επιλέγεται μεταβλητή απόφασης του προηγούμενου προβλήματος με κάποια πιθανότητα, ή να επιλέγεται μόνο αν ανήκει και στις θερμότερες k μεταβλητές του νέου προβλήματος. Για το σκοπό αυτό θα μπορούσαν να χρησιμοποιηθούν οι παράμετροι –noise και –dtop. Η πλήρης αντιγραφή της προηγούμενης πορείας του επιλυτή θα μπορούσε να δημιουργήσει προβλήματα, διότι είναι η πορεία που ακολούθησε ο επιλυτής για την απόδειξη της μη ικανοποιησιμότητας του προβλήματος λιγότερων χρονικών επιπέδων και ίσως να μην οδηγεί σε λύση, αν υπάρχει, του προβλήματος περισσότερων χρονικών επιπέδων.

Οι αποφάσεις του προτασιακού επιλυτή και οι εκμαθημένες προτάσεις που παρήχθησαν, μπορούν να αποθηκεύονται μετά από κάθε επανεκκίνηση σε κάποιο αρχείο. Το αρχείο είναι προτιμότερο να αντικαθίσταται μετά από κάθε επανεκκίνηση, για να περιέχει μόνο τις πληροφορίες της τελευταίας επανεκκίνησης, και το όνομά του

να ελέγχεται από το σύστημα SATPLAN και να δίνεται ως - προαιρετική - παράμετρος (command line argument) κατά την κλήση του επιλυτή.

Μια άλλη επέκταση της παρούσας διπλωματικής εργασίας θα μπορούσε να είναι η επιλογή μεταβλητών απόφασης είδους facts και actions εναλλάξ, ξεκινώντας από τα facts της αρχικής κατάστασης (initFacts, χρονικό επίπεδο 0) και προχωρώντας προς τα επόμενα χρονικά επίπεδα, συνδυάζοντας το ευρετικό απόφασης που χρησιμοποιεί ο προτασιακός επιλυτής PrecoSAT. Αυτό θα μπορούσε να επιτευχθεί μέσω της χρήσης δύο δομών δεδομένων του ίδιου τύπου με τη δομή schedule.decide (η οποία είναι υλοποιημένη στον κώδικα του προτασιακού επιλυτή PrecoSAT και περιέχει τις υποψήφιες μεταβλητές απόφασης), εκ των οποίων η μία δομή θα αφορά μόνο τις μεταβλητές που είναι facts και η άλλη δομή θα αφορά μόνο τις μεταβλητές που είναι actions. Οι μεταβλητές απόφασης θα είναι οι θερμότερες μη ανατεθειμένες μεταβλητές κάθε δομής (fact και action εναλλάξ), οι οποίες, κατά προτίμηση, ανήκουν στο αμέσως επόμενο χρονικό επίπεδο. Το αντίστοιχο θα μπορούσε να γίνει με αντίστροφη σειρά, δηλαδή ξεκινώντας από τα facts της τελικής κατάστασης (goalFacts, τελευταίο χρονικό επίπεδο) και προχωρώντας προς τα προηγούμενα χρονικά επίπεδα.

Σχετικά με τον κώδικα της υλοποίησης, επειδή στόχος της διπλωματικής αυτής ήταν η σύγκριση των τεχνικών μεταξύ τους και όχι η βελτιστοποίηση του κώδικα, κατά τη φάση υλοποίησης δεν έγινε οποιαδήποτε στοχευμένη προσπάθεια βελτιστοποίησης του κώδικα. Για τη βελτιστοποίηση του κώδικα θα μπορούσαμε να κρατήσουμε μόνο τα συστήματα (συνδυασμό τεχνικών) που φαίνονται αποδοτικά για τα περισσότερα προβλήματα, όπως το σύστημα [–fonly –dtop 1000], και να γίνει προσπάθεια βελτιστοποίησής τους, καθώς και κατάλληλη τροποποίηση των δομών δεδομένων που χρησιμοποιούνται (πχ. της δομής schedule.decide). Για παράδειγμα, για την παράμετρο -fonly, σύμφωνα με την οποία επιλέγεται η θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact, θα μπορούσαμε να χρησιμοποιούμε ξεχωριστή δομή ίδιου τύπου με την schedule.decide για τις μεταβλητές που είναι facts, ώστε να αποφεύγονται οι υπολογισμοί που γίνονται πριν από κάθε απόφαση για την ανανέωση των μετρητών των διαθέσιμων μεταβλητών απόφασης κάθε είδους (facts, actions, actionNOOPs), και για να υπάρχει άμεση πρόσβαση στη θερμότερη μη ανατεθειμένη μεταβλητή που είναι fact. Με τη χρήση ξεχωριστής δομής δεδομένων για τα facts, αναμένεται μείωση του χρόνου επίλυσης των προβλημάτων.

Εκτός από την εξέταση τεχνικών μέσω πειραμάτων, θα ήταν χρήσιμο να μελετηθεί εκτενέστερα, σε θεωρητικό επίπεδο, η επίδραση του ευρετικού VSIDS στην επίλυση προβλημάτων Προγραμματισμού Δράσης, και να επιτευχθεί πλήρης κατανόηση της σχέσης του ευρετικού VSIDS με το γράφημα δράσεων των προβλημάτων Προγραμματισμού Δράσης. Επίσης, χρειάζεται να εξεταστεί η εφαρμογή εξειδικευμένων τεχνικών και σε άλλα μέρη των προτασιακών επιλυτών (πχ. ανάλυση σύγκρουσης).

Πέραν από αυτά, η μελέτη εξειδικευμένων τεχνικών θα μπορούσε να επεκταθεί και σε άλλες κατηγορίες προβλημάτων, όπως είναι τα προβλήματα βελτιστοποίησης (scheduling problems) και ο έλεγχος μοντέλων (model checking).

Βιβλιογραφία

[1] A. Biere, “Adaptive Restart Control for Conflict Driven SAT Solvers”, In Proc. 11th Intl. Conf. on Theory and Applications of Satisfiability Testing (SAT'08), Lecture Notes in Computer Science (LNCS) vol. 4996, Springer 2008

[2] A. Biere, “Advanced Model Checking”, Lecture Notes, Institute for Formal Models and Verification, Johannes Kepler University, pp. 127-130, 2012

[3] A. Biere, “Lingeling, Plingeling, PicoSAT and PrecoSAT at SAT Race 2010”, Technical Report 10/1, FMV Reports Series, Institute for Formal Models and Verification, Johannes Kepler University, August 2010.

[4] A. Biere, “P{re,i}coSAT@SC'09. Solver description for SAT Competition 2009”, In SAT 2009 Competitive Event Booklet, 2009

[5] T. Bylander, “The Computational Complexity of Propositional STRIPS Planning”, Artificial Intelligence, vol. 69, pp. 165-204, 1994

[6] J. Chen, “Phase Selection Heuristics for Satisfiability Solvers”, CoRR, 2011

[7] S. Cook, “The Complexity of Theorem-Proving Procedures”, Proceedings of the Third Annual ACM Symposium on Theory of Computing, pp. 151–158, 1971

[8] Y.Dimopoulos, “Propositional Satisfiability: Algorithms and Relationship to CSP”, Lecture Notes, Course “Solution Techniques for Constaint Satisfaction Problems”, Department of Computer Science, University of Cyprus, 2012

[9] H. Kautz, “Deconstructing Planning as Satisfiability”, Proceedings of the 21st National Conference on Artificial Intelligence, (AAAI-06), Boston, MA, 2006

[10] H. Kautz, B. Selman, “Planning as satisfiability”, In Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI'92), pp. 359-363, 1992

[11] H. Kautz, B. Selman, J. Hoffmann, “SatPlan: Planning as Satisfiability”, 2006

[12] N. Manthey, “Improving SAT Solvers Using State-of-the-Art Techniques”, Technische Universität Dresden (diploma thesis), 2010

[13] J. Rintanen, “Heuristics for Planning with SAT”, CP 2010, pp. 414-428, 2010

[14] J. Rintanen, “Planning and SAT”, In A. Biere, H. van Maaren, M. Heule and T. Walsh, Eds., Handbook of Satisfiability, pp. 483-504, IOS Press., 2009

[15] J. Rintanen, “Planning with Specialized SAT Solvers”, AAAI, 2011

[16] S. Russel, P. Norvig, Artificial Intelligence: A Modern Approach (Second Edition), Chapter 7, Logical Agents, pp. 215, Prentice Hall, 2002.

[17] S. Russel, P. Norvig, Artificial Intelligence: A Modern Approach (Third Edition), Chapter 10, Classical Planning, Prentice Hall, 2010.

[18] A. Sideris, Y. Dimopoulos, “Constraint Propagation in Propositional Planning”, 20th International Conference on Automated Planning and Scheduling, (ICAPS'10), Toronto, Canada, 2010

[19] http://fmv.jku.at/precosat/

[20] http://www.cs.rochester.edu/u/kautz/satplan/index.htm