Ατομική Διπλωματική Εργασία

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΠΡΟΒΛΗΜΑΤΩΝ ΠΡΟΤΙΜΗΣΕΩΝ ΣΕ ΠΡΟΤΑΣΙΑΚΗ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Παναγιώτα Κωνσταντινίδου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2016

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μοντελοποίηση Προβλημάτων Προτιμήσεων σε Προτασιακή Βελτιστοποίηση

Παναγιώτα Κωνσταντινίδου

Επιβλέπων Καθηγητής
Γιάννης Δημόπουλος

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

Μάιος 2016

Ευχαριστίες

Φτάνοντας στο τέλος των σπουδών μου και της διπλωματικής μου εργασίας, χαίρομαι που έφερα εις πέρας τον στόχο μου αυτό στο Τμήμα Πληροφορικής Πανεπιστήμιου Κύπρου. Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Δημόπουλο Γιάννη, που με τις γνώσεις του, τις συμβουλές και την καθοδήγηση του κατά την διάρκεια της διπλωματικής εργασίας με βοήθησε να εκπληρώσω τον στόχο μου.

Έπειτα θα ήθελα να ευχαριστήσω το οικογενειακό μου περιβάλλον και ειδικότερα τους γονείς μου, που ήταν κοντά μου σε όλη την φοιτητική μου πορεία και με στήριξαν δίνοντας μου τα εφόδια τόσο ψυχολογικά όσο και σε αξίες για την μετέπειτα σταδιοδρομία μου.

Επίσης θα ήθελα να πω ένα μεγάλο ευχαριστώ στους φίλους και συμφοιτητές μου εντός και εκτός του Πανεπιστημίου που ήταν δίπλα μου και μου χάρισαν αξέχαστες αναμνήσεις και εμπειρίες. Όπως ακόμα και για την στήριξη που μου παρείχαν σε αυτή την μεγάλη μου προσπάθεια.

Περίληψη

Για την αυτοματοποιημένη διαδικασία λήψης αποφάσεων διαδραματίζουν βασικό ρόλο οι πληροφορίες σχετικά με τις προτιμήσεις των χρηστών. Στις μέρες μας είναι χρήσιμο να μπορούμε να αξιολογήσουμε με αποδοτικό τρόπο τις προτιμήσεις έτσι ώστε να μπορούμε να πάρουμε την πιο ικανοποιητική και αποδοτικότερη απόφαση. Η διαδικασία όμως για την εξαγωγή πληροφοριών από τις προτιμήσεις των χρηστών είναι πάρα πολύ δύσκολη, και οι μέθοδοι για αυτή την διαδικασία είναι πολύ σημαντικοί.

Στην παρούσα διπλωματική εργασία είχαμε να μελετήσουμε προβλήματα με προτιμήσεις υπό συνθήκη. Πιο συγκεκριμένα μας δινόταν ένα πρόβλημα που απαρτιζόταν από τις μεταβλητές και τα πεδία τιμών των μεταβλητών. Αυτά τα προβλήματα είχαν περιορισμούς και προτιμήσεις πάνω στα πεδία τιμών. Δηλαδή περιορισμούς που συσχέτιζαν τις μεταβλητές τους και περιόριζαν τις επιτρεπτές αναθέσεις, όπως επίσης και προτιμήσεις πάνω στις τιμές. Αυτό έκανε τη λύση τους πιο δύσκολη. Έτσι εμείς μοντελοποιήσαμε αυτά τα προβλήματα σε Προτασιακή Ικανοποιησιμότητα και σε Προγραμματισμό Συνόλου Απαντήσεων για να μπορέσουμε να τα επιλύσουμε.

Για να αναπαραστήσουμε τις προτιμήσεις που μας δόθηκαν χρησιμοποιήσαμε τα Δίκτυα Προτιμήσεων. Τα Δίκτυα αυτά είναι δίκτυα τα οποία βοηθούν σε τέτοιες αναπαραστάσεις και από τα οποία μπορούμε να πάρουμε πληροφορίες για την βέλτιστη λύση ενός προβλήματος. Έχοντας λοιπόν τις προτιμήσεις ως δεδομένο, φτιάξαμε το Δίκτυο που αντιστοιχούσε σε αυτές και εφαρμόσαμε σε αυτό κάποιες τεχνικές. Οι τεχνικές αυτές ήταν ο υπολογισμός των βαρών που θα δίναμε στους κόμβους του δικτύου και έπειτα η μετατροπή των πληροφοριών σε Προτασιακή Ικανοποιησιμότητα.

Όπως είπαμε και πιο πάνω δημιουργήσαμε δύο συστήματα τα οποία έπαιρναν το προβλήματα και το κάθε ένα από αυτά έκανε την αντίστοιχη μοντελοποίηση, σε Προτασιακή Ικανοποιησιμότητα και σε Προγραμματισμό Συνόλου Απαντήσεων. Στην πρώτη μοντελοποίηση δημιουργήσαμε ένα αρχείο όπου τα δεδομένα του προβλήματος είχαν ερμηνευτεί σε Κανονική Συζευκτική Μορφή. Ενώ στην δεύτερη δημιουργήσαμε ένα αρχείο όπου το πρόβλημα μεταφράστηκε σε Λογικό Προγραμματισμό. Αυτές οι

δύο μοντελοποιήσεις δόθηκαν στους αντίστοιχους επιλυτές, Maxino και Clingo. Αυτοί με την σειρά τους μας έδωσα την λύση του προβλήματος και με την κατάλληλη ερμηνεία μπορέσαμε να κατανοήσουμε την λύση.

Για να ελέγξουμε τις μοντελοποιήσεις που δημιουργήσαμε, τις εφαρμόσαμε σε πολλά προβλήματα. Έπειτα τα μελετήσαμε και διαπιστώσαμε ότι τα Δίκτυα Προτιμήσεων είναι πολύ βοηθητικά για την αναπαράσταση των προτιμήσεων των προβλημάτων. Επίσης ότι η κάθε μεταβλητή εξαρτάται από όλες τις προηγούμενες μεταβλητές στο Δίκτυο Προτιμήσεων και για την ανάθεση μιας τιμής παίζει σημαντικό ρόλο η θέση της σε αυτό. Τέλος οι μέθοδοι αυτοί δίνουν τα καλύτερα δυνατά αποτελέσματα αφού οι δύο επιλυτές ικανοποιούν τους αντίστοιχους περιορισμούς.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο

Στις μέρες μας οι πληροφορίες που προκύπτουν από τις προτιμήσεις των χρηστών διαδραματίζουν σημαντικό και βασικό ρόλο στην αυτοματοποίηση της διαδικασίας λήψης αποφάσεων. Είναι πολύ σημαντικό σε πάρα πολλούς τομείς να μπορούμε με αποδοτικό τρόπο να αξιολογήσουμε τις προτιμήσεις έτσι ώστε να μπορούμε να λάβουμε μια αποδοτικότερη και βέλτιστη απόφαση. Το να μπορέσουμε όμως να εξάγουμε τις σωστές αποφάσεις και πληροφορίες για τις προτιμήσεις των χρηστών καθιστά την διαδικασία πολύ περίπλοκη και δύσκολη. Η εύρεση τέτοιων μεθόδων λήψης προτιμήσεων από τους χρήστες θα βοηθήσουν στην αυτοματοποίηση της διαδικασίας.

Τέτοιου είδους προβλήματα με προτιμήσεις συναντάμε συχνά στην καθημερινότητά μας, στα οποία καλούμαστε να πάρουμε μια απόφαση που θα είναι πιο ικανοποιητική για μας. Ένα τέτοιο παράδειγμα μπορεί να είναι η απόφαση ενός συνοδευτικού ποτού για το κυρίως πιάτο σε ένα εστιατόριο. Το άτομο έχει κάποια προτίμηση στο φαγητό του, ας υποθέσουμε ότι του αρέσουν τα ζυμαρικά και δεν προτιμάει το κρέας. Αν όμως πάρει ζυμαρικά θα προτιμούσε να παραγγείλει ένα άσπρο κρασί ως συνοδευτικό ποτό για το πιάτο του. Αν όμως παραγγείλει κρέας τότε η προτίμηση του στο ποτό είναι κόκκινο κρασί και λιγότερο το άσπρο. Ας υποθέσουμε όμως ότι τα χρήματα που μπορεί

να διαθέσει για το γεύμα του μας περιορίζουν στις επιλογές. Τότε καλείται το άτομο να πάρει το καλύτερο συνδυασμό που θα τον αφήσει ικανοποιημένο με βάση το χρηματικό ποσό που μπορεί να διαθέσει

Ένα πιο περίπλοκο πρόβλημα θα ήταν η απόφαση αγοράς ενός καινούργιου αυτοκίνητου. Το άτομο θα επισημάνει στον πωλητή τις προτιμήσεις του για το αυτοκίνητο που θα ήθελε να αγοράσει όπως το χρώμα, η ιπποδύναμη, η χρηματική αξία του, η χρονιά παραγωγής του, και αν το προτιμάει αυτόματο. Πχ μπορεί το άτομο να θέλει ένα ασημένιο αυτοκίνητο 15 αλόγων ιπποδύναμης, αξίας €9 000, το οποίο να είναι πρόσφατης παραγωγής και αυτόματο. Αλλιώς θα προτιμούσε κάποιο με τα ίδια χαρακτηριστικά αλλά με ταχύτητες. Ή ίσως μια άλλη του επιλογή να ήταν κρατώντας και πάλι όλα τα χαρακτηριστικά ίδια αλλά αλλάζοντας το χρώμα σε μαύρο. Συνεχίζοντας με όλα τα χαρακτηριστικά στα οποία θέλει να δώσει προτιμήσεις, μας δίνει τις επιλογές του. Και με βάση όλα αυτά τα χαρακτηριστικά ο πωλητής καλείται να βρει ένα τέτοιο αυτοκίνητο έτσι ώστε να αφήσει τον πελάτη του ικανοποιημένο δίνοντας του την καλύτερη δυνατή λύση.

1.2Σκοπός Διπλωματικής Εργασίας

Σκοπός της παρούσας διπλωματικής εργασίας είναι η μοντελοποίηση προβλημάτων με προτιμήσεις υπό συνθήκη. Το πρόβλημα αυτό ορίζεται από κάποιες μεταβλητές και ένα σύνολο τιμών για την κάθε μία, οι οποίες μπορούν να γίνουν ανάθεση στις μεταβλητές. Από αυτές τις μεταβλητές διαλέγουμε κάποιες, ένα υποσύνολο, στο οποίο θα δώσουμε κάποιες προτιμήσεις, και θα φτιαχτεί ένα Δίκτυο Προτιμήσεων (CP-Net). Αυτό το Δίκτυο Προτιμήσεων μας δείχνει την εξάρτηση μεταξύ των μεταβλητών αυτών και στις προτιμήσεις των τιμών στο πεδίο τιμών τους.

Τα Δίκτυα Προτιμήσεων (CP-Nets) είναι μια γραφική αναπαράσταση προτιμήσεων καθορίζοντας σχέσεις μεταξύ των εμπλεκομένων μεταβλητών και ορίζοντας τις προτιμήσεις τους. Μέσω αυτών των δικτύων μπορούμε να βρούμε με κάποιες τεχνικές το βέλτιστο αποτέλεσμα. Η σύγκριση στα αποτελέσματα γίνεται στο πλαίσιο του

“ceteris paribus”,δηλαδή “όλα τα άλλα ίδια”, όπου τα αποτελέσματα διαφέρουν μεταξύ τους σε ακριβώς μια μεταβλητή.

Στην διπλωματική αυτή, έχουμε ένα σύνολο μεταβλητών όπου το πεδίο τιμών τους είναι αριθμητικό. Καλούμαστε να δώσουμε μια τιμή από το πεδίο τιμών ικανοποιώντας κάποιους περιορισμούς που έχουν κάποια ζεύγη μεταβλητών πάνω σε συγκεκριμένες τιμές τους, αλλά επίσης ικανοποιώντας και προτιμήσεις στα πεδία τιμών που έχουν δοθεί από το Δίκτυο Προτιμήσεων (CP-Net).

Για μια τέτοια ανάθεση τιμών χρησιμοποιήθηκε ο επιλυτής για προβλήματα Σταθμικής Προτασιακής Βελτιστότητας (weighted-MaxSAT), Maxino. Λόγω του ότι ό επιλυτής αυτός δέχεται προβλήματα σε Κανονική Συζευκτική Μορφή (Conjunctive Normal Form (CNF)), έγινε η κατάλληλη αναπαράσταση των μεταβλητών και των πεδίων τιμών σε μία άλλη μορφή και οι προτιμήσεις μαζί με τους περιορισμούς κωδικοποιήθηκαν σε όρους (clauses) και σε CNF μορφή. Ο επιλυτής κάνει ανάθεση τιμών στις μεταβλητές για να ικανοποιήσει το CNF δηλαδή να γίνει αληθής και αυτή η ανάθεση είναι και η λύση του προβλήματος που δόθηκε.

Εκτός από τον πιο πάνω επιλυτή προσπαθήσαμε να μοντελοποιήσουμε το πρόβλημα προτιμήσεων σε Λογικό Προγραμματισμό και με την χρήση του επιλυτή Clingo να πάρουμε δύο λύσεις από δύο διαφορετικές μοντελοποιήσεις του ίδιου προβλήματος. Λόγο της πολυπλοκότητας της μοντελοποίησης αυτής, καταφέραμε και μοντελοποιήσαμε μια απλοποιημένη μορφή του προβλήματος. Δίνοντας το βασικό πρόβλημα με τους περιορισμούς, να δοθεί μια ανάθεση τιμής σε κάθε μεταβλητή.

1.1 Δομή Διπλωματικής Εργασίας

Αρχικά στο πρώτο κεφάλαιο της διπλωματικής εργασίας, Κεφάλαιο 2, επεξηγούνται και ορίζονται τα Προβλήματα Ικανοποίησης Περιορισμών (ΠΙΠ), η Προτασιακή Ικανοποιησιμότητα και η Κανονική Συζευκτική Μορφή. Ορίζουμε τί είναι ένα Πρόβλημα Ικανοποίησης Περιορισμών, και πως μπορούμε ένα πρόβλημα που υπάρχει να το μετατρέψουμε σε ένα τέτοιο πρόβλημα. Γίνεται αναφορά στην Προτασιακή

Ικανοποιησιμότητα συγκρίνοντας τους δύο αυτούς ορισμούς ενός προβλήματος, Επιπρόσθετα η μεθοδολογία που μπορεί να χρησιμοποιηθεί για μετατροπή από ΠΠ σε Ικανοποιησιμότητα. Λόγω του ότι στην Προτασιακή Ικανοποιησιμότητα το πρόβλημα για να επιλυθεί πρέπει να δοθεί στον επιλυτή σε Κανονική Συζευκτική Μορφή, γίνεται λεπτομερής επεξήγηση αυτής της μορφής και των ορισμών που την αφορούν. Στο Κεφάλαιο 3, εξηγείται ο Προγραμματισμός Συνόλου Απαντήσεων (ΠΣΑ), όπου βλέπουμε την διαφορά αυτού του τρόπου μοντελοποίησης, με την Προτασιακή Ικανοποιησιμότητα. Κοιτάζουμε τα βήματα που ακολουθούν οι δύο μοντελοποιήσεις, κυρίως όμως ο ΠΣΑ ,δηλαδή από τον ορισμό του προβλήματος την μετατροπή του στην κατάλληλη μορφή, μετά επιλυτής μας δίνει το αποτέλεσμα σε κάποια μορφή από την οποία εμείς μπορούμε να βγάλουμε το τελικό αποτέλεσμα και την λύση του αρχικού προβλήματος. Λόγω του ότι χρειαζόμαστε κάποιο υπόβαθρο στην γλώσσα αυτής της μοντελοποίησης, προσπαθήσαμε να παραθέσουμε τα πιο σημαντικά έτσι ώστε να γίνει πιο κατανοητή η μοντελοποίηση.

Στο Κεφάλαιο 4 επεξηγούνται τα Δίκτυα Προτιμήσεων, η μορφή την οποία έχουν και πώς αυτή η μορφή μας βοηθάει στην αναπαράσταση των προτιμήσεων. Για καλύτερη κατανόηση τους δίνονται παραδείγματα με απλά προβλήματα προτιμήσεων και τα παραγόμενα Δίκτυα Προτιμήσεων τους. Στις μεθοδολογίες γίνεται χρήση των δικτύων αυτών για αυτό και η κατανόηση τους είναι πολύ σημαντική. Εξηγείται το πώς από αυτά τα δίκτυα μπορούμε να πάρουμε το βέλτιστο επιθυμητό αποτέλεσμα και πώς μπορούμε έχοντας δύο αποτελέσματα να τα συγκρίνουμε και να βρούμε το προτιμότερο.

Στο Κεφάλαια 5, ακολουθούν οι μεθοδολογίες που χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία για την μοντελοποίηση των προβλημάτων. Δημιουργήσαμε τα κατάλληλα συστήματα, όπου έκαναν χρήση αυτών των μεθοδολογιών για την αντίστοιχη μοντελοποίηση. Δηλαδή δίνοντας και στα δύο συστήματα ένα πρόβλημα με περιορισμούς και προτιμήσεις, εξηγείται το πώς μετατρέπονται αυτά τα δεδομένα σε Προτασιακή Ικανοποιησιμότητα και έπειτα σε CNF, και στο δεύτερο σε Λογικό Προγραμματισμό. Κάθε περιορισμό τον μετατρέψαμε σε λογικό τύπο όπου με χρήση κανόνων έπαιρναν την μορφή που χρειαζόταν, δηλαδή είτε σε CNF είτε σε τύπους του Λογικού Προγραμματισμού. Επίσης εξηγείται και η χρήση του Δικτύου Προτιμήσεων.

Περιγράφεται η μορφή του Δικτύου και πώς αυτό έχει δημιουργηθεί. Αφού περιγράψουμε την μορφή περιγράφεται ο υπολογισμός των κατάλληλων βαρών στο Δίκτυο και πώς οι πληροφορίες που περιέχει γίνονται περιορισμοί του προβλήματος. Αφού γίνουν περιορισμοί στο πρόβλημα τότε μοντελοποιούνται και στην αντίστοιχη μορφή.

Στο Κεφάλαιο 6 γίνεται πειραματική αξιολόγηση των μοντελοποιήσεων με σενάρια προβλημάτων και τα αποτελέσματα τα οποία παίρνουμε σε κάθε τέτοιο σενάριο. Για να δοκιμάσουμε το μέγεθος των προβλημάτων που μπορεί να δεχτεί το σύστημα, δώσαμε προβλήματα με μεγάλο αριθμό μεταβλητών και ελέγξαμε τον χρόνο που χρειάζεται. Επίσης παρατηρήσαμε την επιρροή των περιορισμών στις αναθέσεις των τιμών, όπως επίσης και των προτιμήσεων σε συνδυασμό με τους περιορισμούς. Με βάση τα σενάρια αυτά εξάγουμε τα συμπεράσματα για τις μεθοδολογίες αυτές.

Στο Κεφάλαιο 7 γίνονται κάποιες εισηγήσεις για μελλοντικές βελτιώσεις και επεκτάσεις των συστημάτων μοντελοποίησης των Προβλημάτων.

Κεφάλαιο 2

Ικανοποίηση Περιορισμών

2.1 Εισαγωγή

Αρχικά θα πρέπει να εξηγήσουμε κάποιες βασικές έννοιες της Τεχνητής Νοημοσύνης που σχετίζονται με την Ικανοποίηση Περιορισμών. Θα ορίσουμε τι είναι Προβλήματα Ικανοποίησης Περιορισμών και κάποιες βασικές έννοιες που θα πρέπει να γνωρίζουμε για την μορφή των περιορισμών. Για καλύτερη κατανόηση του ορισμού ενός προβλήματος σε ένα Πρόβλημα Ικανοποίησης Περιορισμών θα δώσουμε κάποιο απλό παράδειγμα. Έπειτα θα εξηγήσουμε την Προτασιακή Ικανοποιησιμότητα και θα παραθέσουμε τις διαφορές μεταξύ της Προτασιακής Ικανοποιησιμότητα και των Προβλημάτων Ικανοποίησης Περιορισμών(ΠΙΠ), όπως επίσης και τον τρόπο με τον οποίο μπορούμε να μετατρέψουμε ένα πρόβλημα από ΠΙΠ σε Προτασιακή Ικανοποιησιμότητα. Ακόμα θα πρέπει να ορίσουμε το τί είναι η Κανονική Συζευκτική Μορφή, όπου σε αυτή την μορφή μετατρέπονται στην Προτασιακή Ικανοποιησιμότητα οι περιορισμοί του προβλήματος και δίνονται στους επιλυτές για να δοθεί η λύση. Τέλος θα ολοκληρώσουμε με την επεξήγηση του επιλυτή που χρησιμοποιήθηκε

δείχνοντας την μορφή του αρχείου που του δίνεται σαν είσοδος και με ποια μορφή επιστρέφει την λύση του.

2.2 Προβλήματα Ικανοποίησης Περιορισμών

Ένας τομέας με τον οποίο ασχολείται η Τεχνητή Νοημοσύνη είναι τα Προβλήματα Ικανοποίησης Περιορισμών (ΠΙΠ). Υπάρχουν αρκετές εξειδικευμένες εταιρίες οι οποίες αναπτύσσουν τεχνολογίες Προβλημάτων Ικανοποίησης Περιορισμών, όπως ακόμα και επιχειρήσεις οι οποίες τις χρησιμοποιούν για να δώσουν λύση σε προβλήματα βελτιστοποίησης ή ακόμα και για λήψη αποφάσεων που τις αφορούν.

Τα προβλήματα αυτά έχουν απλή δομή και απλή τυπική αναπαράσταση, αποτελούνται από μια τριάδα για τον ορισμό τους. Η τριάδα που ορίζει ένα πρόβλημα ικανοποίησης περιορισμών είναι (V,D,C) ,όπου :

- το V (Variables) δηλώνει το σύνολο n μεταβλητών που αφορούν το πρόβλημα, V_1 ,V_2 , ... ,V_n
- το D (Domain) δηλώνει το σύνολο των n πεδίων τιμών D_1 , D_2 , ... ,D_n που αντιστοιχούν σε κάθε μεταβλητή έτσι ώστε το Vi ∈ Di ,κάθε μεταβλητή έχει ένα πεπερασμένο σύνολο πιθανών τιμών και
- το C (Constraints) δηλώνει ένα σύνολο σχέσεων ή αλλιώς περιορισμών C_1,C_2,...,C_m, όπου C_i($V_{k,...}$ V_m.) είναι μια σχέση μεταξύ των μεταβλητών του προβλήματος η οποία περιορίζει τις τιμές που μπορούν να πάρουν ταυτόχρονα οι μεταβλητές.

Οι περιορισμοί είναι εκφράσεις που συσχετίζουν τις μεταβλητές του προβλήματος και που επηρεάζονται από τον περιορισμό.

Π.χ.:

- $X_1+X_2> 10$
- $X_1+2*X_2=3*X_3$
- $X_2 \neq X_1$

Οι περιορισμοί μπορούν χαρακτηριστούν με τρείς τρόπους ανάλογα με τον αριθμό των μεταβλητών που περιλαμβάνουν. Ένας περιορισμός χαρακτηρίζεται ως μοναδιαίος

(unary) όταν αφορά μόνο μία μεταβλητή (π.χ. $X_1 \geq 2$). Μπορεί να χαρακτηριστεί ως δυαδικός (binary) όταν στον περιορισμό λαμβάνουν μέρος δύο μεταβλητές του προβλήματος (π.χ. $X_1+X_2 \geq 10$). Τέλος έχουμε τους περιορισμούς ανώτερης τάξης (higher order), οι οποίοι είναι περιορισμοί που περιλαμβάνουν περισσότερες από 2 μεταβλητές (π.χ. $X_1+2*X_2 =3*X_3$).

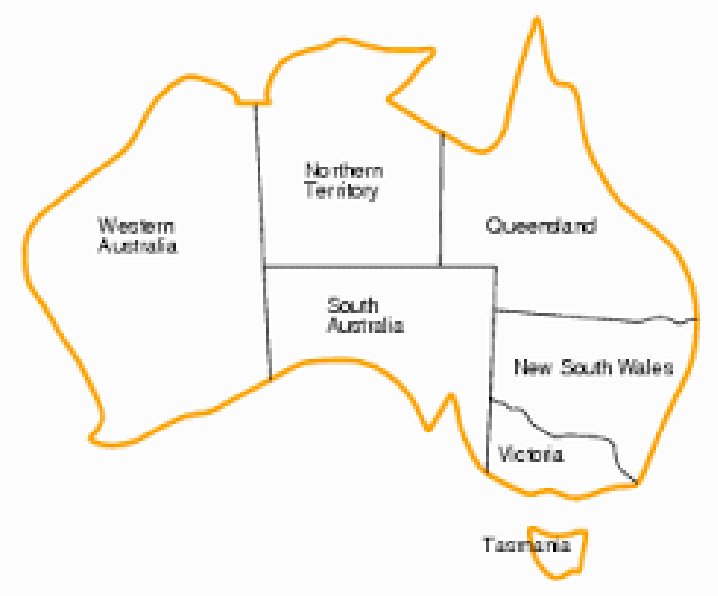
Επίσης έχουμε δύο είδη περιορισμών, τους απόλυτους περιορισμούς (hard constraints) και τους ελαστικούς περιορισμούς (soft constraints). Η διαφορά αυτών των περιορισμών είναι ότι οι ελαστικοί δηλώνουν προτιμήσεις και δεν είναι απαραίτητη η ικανοποίηση τους.[1]

Με το πιο πάνω τρόπο τα προβλήματα μοντελοποιούνται και δίνονται σε ένα επιλυτή. Ο επιλυτής θα επιλύσει το πρόβλημα. Λύση του Προβλήματος Ικανοποίησης Περιορισμών λέγεται η ανάθεση τιμών σε όλες τις μεταβλητές, η οποία ικανοποιεί όλους τους περιορισμούς. Μια ανάθεση τιμών μπορεί να χαρακτηριστεί ολοκληρωμένη όταν περιέχει όλες τις μεταβλητές. Το Πρόβλημα Ικανοποίησης Περιορισμών λέγεται συνεπές αν υπάρχει τουλάχιστον μια λύση, αλλιώς ονομάζεται ασυνεπές.

Για την επίλυση των Προβλημάτων Ικανοποίησης Περιορισμών έχουμε αλγορίθμους αναζήτησης οι οποίοι εκμεταλλεύονται αυτή την απλή αναπαράσταση. Χρησιμοποιούν γενικούς ευρετικούς μηχανισμούς για να επιτύχουν την επίλυση μεγάλων προβλημάτων, ίσως ακόμη και στην εύρεση μιας βέλτιστης λύση τους.

Παράδειγμα Προβλήματος Ικανοποίησης Περιορισμών:

Ένα από τα πιο γνωστά προβλήματα ικανοποίησης περιορισμών είναι ο χρωματισμός του χάρτη της Αυστραλίας. Σκοπός του προβλήματος είναι δύο γειτονικές περιοχές να μην έχουν το ίδιο χρώμα. Θέλουμε να χρωματίσουμε τον χάρτη με τρία διαφορετικά χρώματα: Κόκκινο, Πράσινο και Μπλε.



Σχήμα 2.1 :Χάρτης Αυστραλίας

Το πρόβλημα ορίζεται με βάση την τριάδα που εξηγήθηκε πιο πάνω ως εξής:

- Οι μεταβλητές του προβλήματος είναι οι περιοχές: WA, NT, SA, Q, NSW, V, T.
- Το πεδίο τιμών τους (είναι το ίδιο για όλες τις μεταβλητές στο παρόν πρόβλημα):{red, green,blue}.
- Και τέλος οι περιορισμοί που δηλώνουν την απαίτηση του προβλήματος ότι δύο γείτονες δεν μπορούν να έχουν το ίδιο χρώμα: WA≠SA, WA≠NT, SA≠NT, Q≠SA, Q≠NT, NSW≠SA, NSW≠Q, V≠SA, NSW≠V.

Λύση του προβλήματος αυτού είναι να δοθεί ένα χρώμα για κάθε περιοχή έτσι ώστε να ικανοποιηθούν οι πιο πάνω περιορισμοί. Μια πιθανή λύση θα ήταν η εξής: WA={red}, SA={green}, NT={blue}, Q={red}, NSW={blue}, V={red}, T={blue}.

2.3 Λόγοι επιλογής αναπαράστασης του προβλήματος σε ΠΙΠ

Οι βασικοί λόγοι για την επιλογή αναπαράστασης ενός προβλήματος σε Πρόβλημα Ικανοποίησης Περιορισμών είναι ότι, η μετατροπή ενός προβλήματος είναι σχετικά απλή και δεν διαφέρει πολύ από το αρχικό πρόβλημα, δηλαδή το πρόβλημα το οποίο αντιμετωπίζουμε. Οι μεταβλητές μπορούν να συσχετιστούν πολύ εύκολα με το αρχικό πρόβλημα και οι περιορισμοί να αναπαρασταθούν σε μια πιο απλή μορφή γραμμικών

ανισοτήτων. Επίσης μετατρέποντας το πρόβλημα σε ΠΠ μπορεί να δοθεί λύση σε γρήγορο χρονικό διάστημα αφού υπάρχουν αλγόριθμοί που εξιδανικεύονται στην επίλυση τέτοιων προβλημάτων.

2.4 Προτασιακή Ικανοποιησιμότητα

Το πρόβλημα της Ικανοποιησιμότητας στην Προτασιακή Λογική είναι γνωστό ως **SAT** (Boolean Satisfiability Problem), το οποίο αποτελεί εδώ και χρόνια ένα βασικό θέμα μελέτης όχι μόνο από πλευράς θεωρίας υπολογισμού αλλά και από πλευράς μαθηματικής λογικής. Είναι από τα πρώτα προβλήματα στον κλάδο της Πληροφορικής που αποδείχτηκε ότι ανήκει στα NP-πλήρη (NP-complete) προβλήματα (Cook,1971), δηλαδή προβλήματα τα οποία πιστεύουμε ότι δεν υπάρχει αλγόριθμός ο οποίος μπορεί να τα επιλύσει σε πολυωνυμικό χρόνο.

Ελέγχει αν ένας περίπλοκος λογικός τύπος μπορεί να γίνει αληθής, δηλαδή να ικανοποιηθεί (be satisfiable).Αυτό θα επιτευχθεί αν δοθούν οι τιμές “αληθής” και “ψευδής” σε μεταβλητές στον λογικό τύπο έτσι ώστε με αυτή την ανάθεση όλος ο τύπος να γίνει αληθής άρα και ικανοποιήσιμος. Η μη εύρεση μιας τέτοιας ανάθεσης τιμών κάνει τον τύπο ψευδή και επομένως ονομάζεται μη ικανοποιήσιμος (unsatisfiable). Ο λογικός τύπος που δίνεται είναι σε Κανονική Συζευκτική Μορφή (Conjunctive Normal Form (CNF)).

Επίσης έχουμε και την Προτασιακή Βελτιστότητα αλλιώς MAX-SAT (maximum satisfiability problem), μπορεί να θεωρηθεί γενίκευση του SAT, του οποίου ο σκοπός είναι να ικανοποιηθούν όσο το δυνατόν πιο πολλοί όροι (clauses) της Κανονικής Συζευκτικής Μορφής. Σαν γενίκευση της Προτασιακής Βελτιστότητας (MaxSAT) είναι η Σταθμική Προτασιακή Βελτιστότητα (weighted MaxSAT). Στη Σταθμική Προτασιακή Βελτιστότητα (weighted MaxSAT), σε κάθε όρο (clause) δίνεται και κάποιο βάρος. Αυτό το βάρος υποδεικνύει το ότι αν ικανοποιηθεί κάποιος όρος (clause), αυτό είναι σημαντικότερο από το να ικανοποιηθούν πολλοί άλλοι όροι (clauses). Στόχος είναι η ελαχιστοποίηση του συνολικού βάρους των μη ικανοποιήσιμων όρων (clauses)και όχι μόνο ο αριθμός τους.[4]

2.4.1 Διαφορές Προβλημάτων Ικανοποίησης Περιορισμών και Προτασιακής Ικανοποιησιμότητας

Μεταξύ των Προβλημάτων Ικανοποίησης Περιορισμών και των προβλημάτων Προτασιακής Ικανοποιησιμότητας υπάρχουν κάποιες διαφορές. Μια από αυτές είναι οι τιμές οι οποίες παίρνουν οι μεταβλητές τους, το πεδίο τιμών των μεταβλητών τους. Στα ΠΙΠ το πεδίο τιμών είναι πεπερασμένο, σε αντίθεση με τα SAT όπου το πεδίο τιμών είναι δυαδικό και παίρνουν την τιμή "αληθές" ή "ψευδές" (1 ή 0). Διαφορά επίσης έχουν και στους περιορισμούς όπου στο SAT είναι μη δυαδικοί ενώ στα ΠΙΠ έχουμε δυαδικούς περιορισμούς. [5]

2.4.2 Μετατροπή από ΠΙΠ σε πρόβλημα Προτασιακής Ικανοποιησιμότητας

Ένα πρόβλημα ΠΙΠ μπορεί να κωδικοποιηθεί σε πρόβλημα Προτασιακής Ικανοποιησιμότητας με κάποιες αλλαγές. Μπορούμε να συσχετίσουμε μια μεταβλητή της Προτασιακής Ικανοποιησιμότητας με μια μεταβλητή του ΠΙΠ αν για κάθε πιθανή τιμή της μεταβλητής δημιουργήσουμε μια νέα. Δηλαδή για μια μεταβλητή Xij στην Προτασιακή Ικανοποιησιμότητα, συσχετίζεται με την Xi στο ΠΙΠ εάν της δώσουμε την τιμή j από το πεδίο τιμών της. Επομένως η απόδοση τιμής "αληθές" ή "ψευδές" στην Xij μας δηλώνει ότι στο ΠΙΠ η μεταβλητή Xi έχει την τιμή j ή και όχι αντίστοιχα.

2.5 Κανονική Συζευκτική Μορφή (Conjunctive Normal Form (CNF))

Οι περιορισμοί στη Προτασιακή Ικανοποιησιμότητα είναι σε Κανονική Συζευκτική Μορφή (Conjunctive Normal Form). Οι επιλυτές, δοθέντος ενός προβλήματος με περιορισμούς της μορφής αυτής, καλούνται να δώσουν μια δυαδική αναπαράσταση στα λεκτικά (literals) (Αληθής-Ψευδής) έτσι ώστε να ικανοποιηθεί η πρόταση αυτή.

Στη Κανονική Συζευκτική Μορφή οι προτάσεις αποτελούνται από συζεύξεις διαζεύξεων. Σύζευξη ονομάζουμε το λογικό AND το οποίο στην Προτασιακή Λογική συμβολίζεται με " Λ " , και διάζευξη το λογικό OR και συμβολίζεται με " V " .Επίσης έχουμε την άρνηση η οποία συμβολίζεται με " ┐ ".

Μια πρόταση σε CNF έχει δηλαδή την πιο κάτω μορφή:

$\mathbf{A_1 \wedge A_2 \wedge A_3 \wedge \ldots \wedge A_n}$

Όπου τα A1,…,An λέγονται όροι (clauses) και κάθε ένα από αυτά αποτελείται από λεκτικά (literal) που ενώνονται μεταξύ τους με διαζεύξεις

$\mathbf{A_1=(\beta_1 \vee \beta_2 \vee \ldots \vee \beta_n)}$ **,** όπου τα $\beta_1, \beta_2 \ldots, \beta_n$ ονομάζονται λεκτικά.

Τα λεκτικά είναι οι προτασιακές μεταβλητές και μπορούν να εμφανιστούν με την θετική ή την αρνητική τους μορφή δηλαδή ως β ή ┐ β. Αν ένα λεκτικό εμφανίζεται στον λογικό τύπο μόνο με την μία μορφή του, δηλαδή μόνο με την θετική ή μόνο με την αρνητική του μορφή, τότε το λεκτικό αυτό το χαρακτηρίζουμε ως pure literal (καθαρό λεκτικό).

Επίσης έχουμε 3 ήδη όρων (clauses):

- Μοναδιαίος όρος (Unit clause), ο οποίος αποτελείται από μόνο ένα λεκτικό είτε στην θετική είτε στην αρνητική του μορφή, π.χ. (β_1)/\ (┐ β_2).
- Δυαδικός όρος (Binary clause), ο οποίος αποτελείται από δύο λεκτικά με την θετική είτε με την αρνητική τους μορφή , π.χ. ($\beta_1 \vee$ ┐ β_2)/\ (┐ $\beta_1 \vee \beta_2$).
- Άδειος όρος (Empty clause) είναι ο όρος χωρίς λεκτικό.

Οποιαδήποτε πρόταση μπορεί να μετατραπεί σε Κανονική Συζευκτική Μορφή με την χρήση λογικών κανόνων. [2]

Κανόνες:

Διπλή συνεπαγωγή: $(\varphi \Leftrightarrow \psi)=(\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$ (AN KAI MONO AN)

Συνεπαγωγή: $(\varphi \Rightarrow \psi)=(\neg \varphi \vee \psi)$ (EAN TOTE)

Συζεύξει από διαζεύξεις:

$\varphi \vee (\psi_1 \wedge \psi_2)=(\varphi \vee \psi_1) \wedge (\varphi \vee \psi_2)$

$(\psi_1 \wedge \psi_2) \vee \varphi = (\psi_1 \vee \varphi) \wedge (\psi_2 \vee \varphi)$

De Morgan:

$\neg (\varphi \wedge \psi) = (\neg \varphi \vee \neg \psi)$

$\neg (\varphi \vee \psi) = (\neg \varphi \wedge \neg \psi)$

Διπλή άρνηση: $\neg \neg \varphi = \varphi$

Αλγόριθμος μετατροπής σε Κανονική Συζευκτική Μορφή:

1. Απαλοιφή των συνεπαγωγών.
2. Απαλοιφή των διπλών αρνήσεων και σπρώχνουμε τις αρνήσεις στο επίπεδο των λεκτικών.
3. Εισαγωγή Συζεύξεων από Διαζεύξεις

2.6 Maxino Επιλυτής

Ο Maxino είναι επιλυτής προβλημάτων Προτασιακής Βελτιστοποίησης (MaxSAT), δηλαδή βελτιστοποίησης του απλού προβλήματος Προτασιακής Ικανοποιησιμότητας για προτασιακούς τύπους. Ο Maxino βασίζεται στον αλγόριθμο k-ProcessCore, τον οποίο εξηγούν οι Mario Alviano, Carmine Dodaro, and Francesco Ricca στο άρθρο τους «A MaxSAT Algorithm Using Cardinality Constraints of Bounded Size» [3]

Ο Maxino λύνει προβλήματα Προτασιακής Βελτιστοποίησης (MaxSAT) με βάρη καθώς κάνει χρήση hard και soft clauses (ρητών ή απόλυτων και ελάχιστων όρων). Οι μαλακοί όροι σχετίζονται με το βάρος και δεν είναι αναγκαία η ικανοποίηση τους, ενώ οι ρητοί όροι πρέπει να ικανοποιηθούν αναγκαστικά. Βασικός σκοπός του επιλυτή είναι να δοθεί μια ανάθεση ‘‘αληθές’’-‘‘ψευδές’’ στα λεκτικά έτσι ώστε να ικανοποιηθούν οι ρητοί όροι και να μειωθεί το συνολικό βάρος των μη ικανοποιήσιμων μαλακών όρων.

Στη διπλωματική αυτή έγινε χρήση του επιλυτή αυτού, χρησιμοποιώντας την έκδοση maxino-kdyn, το οποίο χρησιμοποιεί τον αλγόριθμο K με μια δυναμική επιλογή του ορίου λεκτικών ανάλογα με το μέγεθος του επεξεργασμένου πυρήνα.

Στον Maxino δίνεται ένα αρχείο της μορφής [11]

```
p wcnf 4 16 100
100 -1 -2 -3 0
100 -1 -2 -4 0
100 -1 -3 -2 0
100 -1 -3 -4 0
100 -1 -4 -2 0
100 -1 -4 -3 0
100 -2 -3 -1 0
100 -2 -3 -4 0
100 -2 -4 -1 0
100 -2 -4 -3 0
100 -3 -4 -1 0
100 -3 -4 -2 0
1 1 0
1 2 0
2 3 0
2 4 0
```

Στο πιο πάνω παράδειγμα η πρώτη γραμμή δηλώνει ότι είναι weightCNF; όπου κάθε γραμμή του αντιστοιχεί σε ένα όρο (clause), και έχει το δικό του βάρος. Έπειτα αναγράφεται ο αριθμός των διαφορετικών λεκτικών (μεταβλητών), μετά ο αριθμός των γραμμών του αρχείου, εκτός της πρώτης γραμμής, με λίγα λόγια ο αριθμός των όρων (clauses) του προβλήματος. Και τέλος γράφεται το μέγιστο βάρος, το οποίο έχουν κυρίως οι ρητοί όροι (hard clauses). Κάθε γραμμή όπως έχουμε αναφέρει αντιστοιχεί σε ένα όρο (clause), επομένως όλο το αρχείο είναι ένας προτασιακός τύπος της μορφής CNF. Οι γραμμές συνδέονται με το λογικό «/\» (AND) ενώ τα στοιχεία της κάθε γραμμής συνδέονται με το λογικό «\/» (OR).

Οι πρώτοι όροι (clauses) κάνουν χρήση του μέγιστου βάρους επομένως είναι ρητοί όροι (hard clauses), ενώ οι τελευταίοι όροι έχουν πιο μικρό βάρος, που δηλώνει ότι είναι ελάχιστοι όροι (soft clauses). Επίσης, οι ελάχιστοι όροι (soft clauses) μεταξύ τους έχουν διαφορετικό βάρος που δηλώνει την αναγκαιότητα για ικανοποίηση τους. Για την αναπαράσταση της άρνησης ενός λεκτικού χρησιμοποιείται το “-“ και στο τέλος κάθε

γραμμής τοποθετούμε το 0. Όπως ακόμα οι μεταβλητές αναπαριστώνται σε αριθμητική μορφή και όχι σε χαρακτήρες αλφαβήτου όπως συνηθίζεται στην προτασιακή λογική. Το κάθε λεκτικό έχει δυαδικό πεδίο τιμών “αληθές” ή “ψευδές”.

Τέλος ο επιλυτής δίνει το πιο κάτω αποτέλεσμα:

o 0

c ub 2

o 1

o 2

s OPTIMUM FOUND

v -1 -2 3 4

Η τελευταία γραμμή είναι και η λύση του προβλήματος, οι τιμές των λεκτικών που έχουν την θετική τους μορφή δηλώνουν ότι έχουν πάρει την τιμή “αληθές” ενώ τα λεκτικά με την αρνητική(έχουν “–“) τους μορφή δηλώνουν ότι ο επιλυτής έχει δώσει την τιμή “ψευδής”. Αν δεν βρέθηκε λύση στο πρόβλημα τότε θα μας δώσει το μήνυμα «UNSATISFIABLE».

Κεφάλαιο 3

Προγραμματισμός Συνόλου Απαντήσεων

3.1 Εισαγωγή

Εκτός από την Προτασιακή Ικανοποιησιμότητα υπάρχουν και άλλες μεθοδολογίες αναπαράστασης γνώσης βασισμένες στη λογική και μπορούν να χωριστούν σε δύο κατηγορίες. Η μία κατηγορία είναι οι γλώσσες που σχετίζονται με την κλασική λογική(classical logic), ενώ η δεύτερη είναι βασισμένη στις μεθοδολογίες λογικού προγραμματισμού (logic programming).

Μια από αυτές είναι και ο Προγραμματισμός Συνόλου Απαντήσεων (Answer Set Programming (ASP)). Είναι μια μορφή δηλωτικού προγραμματισμού που προσανατολίζεται σε δύσκολα (κυρίως NP-δύσκολα) προβλήματα αναζήτησης. Στο ASP, τα προβλήματα αναζήτησης μειώνονται σε υπολογιστικά σταθερά μοντέλα, και υπάρχουν επιλυτές, προγράμματα για την δημιουργία σταθερών μοντέλων, που χρησιμοποιούνται για την εκτέλεση αναζήτησης.

Ο Προγραμματισμός Συνόλου Απαντήσεων αποτελεί μια δηλωτική προσέγγιση επίλυσης προβλημάτων η οποία συνδυάζει την απλή γλώσσα μοντελοποίησης με ικανότητες επίλυσης υψηλής απόδοσης. Επίσης έχει τις ρίζες του στις Βάσεις Δεδομένων, στον Λογικό Προγραμματισμό, (Logic Based) Αναπαράσταση γνώσης και στην (nonmonotonic) αιτιολογία, όπως ακόμα και στον περιορισμό των προβλημάτων κυρίως σε ελέγχους ικανοποιησιμότητας.

3.2 Σύνταξη Λογικού Προγραμματισμού

Ένα λογικό πρόγραμμα (P) πάνω σε ένα σύνολο από atoms (A) είναι ένα πεπερασμένο σύνολο κανόνων (r). [9]
Ένας κανόνας είναι της μορφής
$a_0 \leftarrow a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n$ όπου $0 \leq m \leq n$ και τα $a_i \in A$, $0 \leq i \leq n$.

Το a_0 λέγεται κεφάλι (head) του κανόνα και τα $\{a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n\}$ είναι το σώμα (body) του κανόνα. Τα $\{a_1,\ldots,a_m\}$είναι το θετικό σώμα και $\{\sim a_{m+1},\ldots,\sim a_n\}$ το αρνητικό σώμα.

Γλωσσικές Δομές:

If: “←“ (“ :-“)
And: “ , “ (“,”)
Or: “ ; ” (“;”)
Default Negation: “~” (“not”)
Classical Negation: “¬” (“ - “)

Μεταβλητές (Variables): p(X) :- q(X)
Λεκτικά υπό Συνθήκης (Conditional literals): p :- q(X) : r(X)
Διάζευξη (Disjunction): p(X) ; q(X) :- r(X)
Περιορισμοί Ακεραιότητας (Integrity constraints): :- q(X), p(X)
Επιλογή (Choice): 2 {p(X,Y) :q(X)}7 :- r(Y)
Συνάθροιση (Aggregates): s(Y) :- r(Y), 2 #sum {X : p(X,Y), q(X)}7

Βελτιστοποίηση (Optimization):

Αδύναμοι Περιορισμοί (Weak Constraints): :~ q(X), p(X,C) [C]

Δηλώσεις (Statements): #minimize{C : q(X), p(X,C)}

Περιορισμοί Ακεραιότητας (Integrity constraints):

Είναι για να περιορίσουμε μη αποδεκτές πιθανές λύσεις οι οποίες θα μπορούσε ο επιλυτής να δώσει.

Όπως δείξαμε πιο πάνω είναι της μορφής : :- q(X), p(X)

που αυτό σε πιο γενική μορφή αναπαριστάται σε:

$\leftarrow a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n$ όπου $0\leq m\leq n$ και τα a_i είναι atoms , $1\leq i\leq n$.

:- edge(V,U), colored(V,C), colored(U,C).

Κανόνες Επιλογής (Choice rule):

Για επιλογή πάνω σε υποσύνολα έχουμε τον πιο κάτω κανόνα:

$\{a_1,\ldots,a_m\}\leftarrow a_{m+1},\ldots,a_n,\sim a_{n+1},\ldots\sim al$ όπου $0\leq m\leq n\leq l$ και τα a_i είναι atoms, $1\leq i\leq l$

Αν το σώμα (body) μπορεί να ικανοποιηθεί από ένα στατικό μοντέλο τότε κάθε υποσύνολο του $\{a_1,\ldots,am\}$ μπορεί να συμπεριληφθεί στο στατικό μοντέλο.

{ buy(pizza); buy(wine); buy(corn) }:- at(grocery).

Κανόνας Πληθικότητας (Cardinality Rule):

Για τον έλεγχο της πληθικότητας του υποσυνόλου στον κανόνα έχουμε τους κανόνες πληθικότητας. Καθορίζει την μικρότερη πληθικότητα.

$a_0\leftarrow k\{a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n\}$ όπου $0\leq m\leq n$ και τα a_i είναι atoms, $1\leq i\leq n$ και το k είναι ένας θετικός ακέραιος αριθμός. Το k είναι αυτό που θα καθορίσει το μικρότερο όριο στο σώμα του κανόνα. Το άτομο στο κεφάλι του κανόνα (head atom) ανήκει στο στατικό μοντέλο. εάν τουλάχιστον τα k στοιχεία από το σώμα μπορούν να ικανοποιηθούν και ανήκουν και αυτά στο στατικό μοντέλο.

Σε αυτού του τύπου κανόνων συμπεριλαμβάνεται και ο κανόνας που καθορίζει ένα άνω όριο, εκτός από αυτόν για το κάτω.

$a_0\leftarrow k\{a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n\}r$ όπου $0\leq m\leq n$ και τα a_i είναι atoms, $1\leq i\leq n$ και τα k ,r είναι θετικοί ακέραιοι αριθμοί. Το k μας δηλώνει το κάτω όριο και το r το άνω όριο.

Επιπρόσθετα έχουμε και κανόνες πληθικότητας που αφορούν το κεφάλι του κανόνα.
$k\{a_1,\ldots,a_m,\sim a_{m+1},\ldots,\sim a_n\}\ r \leftarrow a_{n+1},\ldots,a_o,\ldots,\sim a_{o+1},\ldots,\sim a_p$ όπου $0\leq m\leq n \leq o\leq p$ και τα a_i είναι atoms, $1\leq i\leq p$ και τα k ,r είναι θετικοί ακέραιοι αριθμοί.

Κανόνες με Βάρη:

$a_0 \leftarrow k\{w_1:a_1,\ldots,w_m:a_m,w_{m+1}:\sim a_{m+1},\ldots,w_n:\sim a_n\}$ όπου $0\leq m\leq n$ και τα a_i είναι atoms, το k και τα w_i είναι ακέραιοι όπου $1\leq i\leq n$. Επίσης τα w_i είναι βάρη και τα σχετίζουμε με λεκτικά.

Περιορισμός με Βάρη :

$k\{w_1:a_1,\ldots,w_m:a_m,w_{m+1}:\sim a_{m+1},\ldots,w_n:\sim a_n\}r$ όπου $0\leq m\leq n$ και τα a_i είναι atoms και τα k, r, wi είναι ακέραιοι $1\leq i\leq n$.
Ο περιορισμός ικανοποιείται σε ένα στατικό μοντέλο αν το άθροισμα των βαρών των ατόμων που ανήκουν ή όχι στο στατικό μοντέλο είναι μεταξύ των ορίων αυτών.
10 { 4:course(db); 6:course(ai); 8:course(project); 3:course(xml) } 20

Άρνηση:

Υπάρχουν δύο τύπων αρνήσεις , η κλασική άρνηση("¬") και η default άρνηση ("~").
Η κλασική άρνηση εφαρμόζεται μόνο στα άτομα (atoms) σε αντίθεση με την default άρνηση που εφαρμόζεται στο κεφάλι(head) των κανόνων.

Κανόνας Διάζευξης:

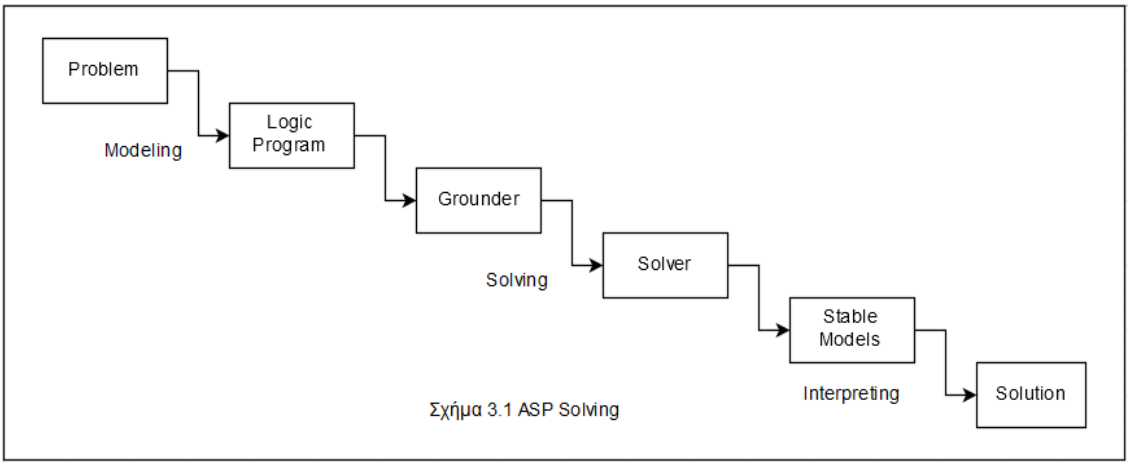
$a1;\ldots;am \leftarrow am+1,\ldots an,\sim an+1,\ldots, \sim ao$ όπου $0\leq m\leq n \leq o$,τα a_i είναι atoms $0\leq i\leq o$
[8]

3.3 Σύγκριση Προτασιακής Ικανοποιησιμότητας με Προγραμματισμό Συνόλου Απαντήσεων

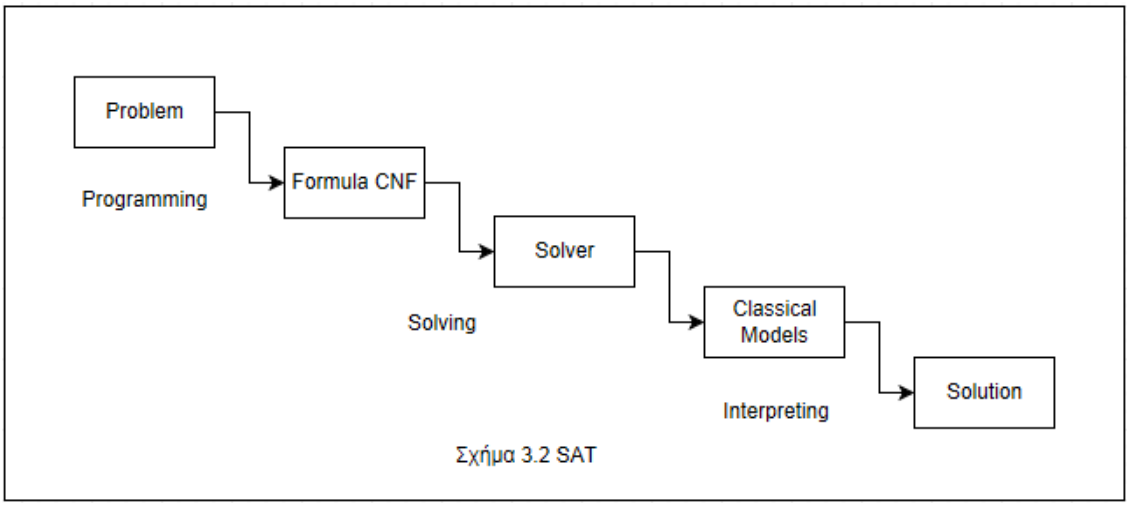
Οι δύο τρόποι μοντελοποίησης προβλημάτων ικανοποίησης έχουν κάποιες διαφορές μεταξύ τους. Μια ομοιότητα τους όμως είναι ότι είναι από κάτω προς τα πάνω υπολογισμός. Αντιθέτως όμως το ASP βασίζεται στην εποικοδομητική λογική (constructive logic), ενώ η Προτασιακή Ικανοποιησιμότητα (SAT) στην κλασική

λογική. Επίσης είναι κλειστού και ανοικτού κόσμου αιτιολογίας ενώ το SAT είναι ανοικτού. Έχει γλώσσα μοντελοποίησης εν αντιθέσει με την προτασιακή που δεν έχει. Επιπρόσθετα έχει πολύπλοκους τρόπους συλλογισμού, αυτοί οι τρόποι είναι ο έλεγχος ικανοποιησιμότητας, της απαρίθμησης/προβολής, της τομής/της ένωσης, κι τέλος της βελτιστοποίησης. Το SAT κάνει μόνο έλεγχο ικανοποιησιμότητας. Τέλος το ASP λύνει προβλήματα NP και NP^{NP} , το SAT είναι σε προβλήματα NP.

..



Σχήμα 3.1 ASP Solving



Σχήμα 3.2 SAT

Πιο πάνω βλέπουμε τα διάφορα βήματα για τους δύο τρόπους μοντελοποίησης ενός προβλήματος και από ποια στάδια θα περάσουν μέχρι να δοθεί λύση στο πρόβλημα. Βλέπουμε καθαρά ότι τα στάδια στο ASP είναι πιο πολλά, το πρόβλημα πρέπει να μοντελοποιηθεί σε λογικό πρόγραμμα το οποίο θα δοθεί στον επιλυτή ο οποίος θα μας επιστρέψει στατικά μοντέλα τα οποία είναι η λύση. Στο SAT το πρόβλημα κωδικοποιείται σε CNF δίνεται στο επιλυτή ο οποίος θα επιστρέψει ένα κλασικό μοντέλο όπου αυτό είναι και η λύση.

3.4 Clingo Επιλυτής

Το Πανεπιστήμιο του Potsdam ανέπτυξε κάποια εργαλεία για τον Προγραμματισμό Συνόλου Απαντήσεων (ASP), το Potassco. Το Potassco είναι το όνομα που δόθηκε σε αυτή την συλλογή και σημαίνει Potsdam Answer Set Solving Collection.

Κάποια εργαλεία του είναι:

- Grounder gringo, lingo,
- Solver clasp, claspfolio, claspar, aspeed
- Grounder+Solver Clingo, Clingcon, ROSoClingo

Για την παρούσα διπλωματική εργασία χρησιμοποιήθηκε το Clingo. Το οποίο συνδυάζει το clasp και το gringo.Το clasp είναι επιλυτής για σύνολα απαντήσεων για κανονικά και διαζευκτικά λογικά προγράμματα, ενώ το gringo για προγράμματα ελεύθερων μεταβλητών.

Το clingo προσφέρει περισσότερο έλεγχο στην διαδικασία του grounding και της επίλυσης από ότι τα gringo και clasp μεμονωμένα. Επίσης υποστηρίζει παράλληλη και διαζευκτική επίλυση. [7][8]

Για τη διαχείριση των προτιμήσεων μεταξύ των στατικών μοντέλων χρησιμοποιήθηκε το asprin. Το asprin είναι ένα λογισμικό στο οποίο συμπεριλαμβάνεται η βιβλιοθήκη η οποία περιέχει όλες τις απαραίτητες λειτουργίες για τις εντολές βελτιστοποίησης και των προτιμήσεων. Βασίζεται σε προηγμένες δυνατότητες ελέγχου για σταδιακή επίλυση, επιτρέποντας την αναζήτηση για συγκεκριμένες λύσεις χωρίς τροποποιήσεις του ASP.

3.5 Προτιμήσεις και Βελτιστοποίηση

3.5.1 Γλώσσα

Η γλώσσα αποτελείται από τα πιο κάτω στοιχεία [12]

- Σταθμικός Τύπος (Weighted formula) $w_1,\ldots,w_m : \phi$, όπου wi είναι όρος (term) και ϕ είναι Boolean τύπος
- Naming atom, name(s). Το name είναι κατηγόρημα (predicate), το s είναι το όνομα της προτίμησης.
- Στοιχείο Προτίμησης (Preference element) $\Phi_1 > \ldots > \Phi_m \parallel \phi$, όπου Φ_r είναι ένα σύνολο από σταθμικούς τύπους (weighted formulas) και ϕ είναι μη σταθμικός τύπος (non-weighted formula).
- Δήλωση Προτίμησης (Preference statement) #preference(s,t){$e_1,\ldots,e_n$}, όπου το s και t αντιπροσωπεύουν το όνομα και τον τύπο αντίστοιχα της δήλωσης προτίμησης, και τα e_i είναι στοιχεία προτίμησης.
- Ντιρεκτίβα Βελτιστοποίησης (Optimization directive) #optimize(s) λέει στον επιλυτή να περιορίσει τον τρόπο επίλυσης του στην σχέση προτίμηση που δηλώνει το s.
- Προδιαγραφή Προτίμησης (Preference specifications) είναι ένα σύνολο S που αποτελείται από preference statements και ένα directive #optimize(s) τέτοιο ώστε το S να είναι ακυκλικό και κλειστό. $s \in S$
    - Κλειστό, αν για κάποιο p όπου το $p \in S$, και id(p)=s όποτε το name(s) εμφανίζεται στο S.
    - Ακυκλικό, αν η σχέση εξάρτησης που προκαλείται μεταξύ των στοιχείων στο S με τη ονομασία των atoms είναι ακυκλικό.

Η δήλωση προτίμησης (preference statement) δηλώνει μια σχέση προτίμησης σχετίζοντας το τύπο της προτίμησης (preference type) και το στοιχείο προτίμησης (preference element).

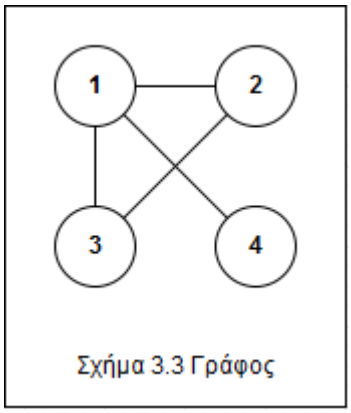
Ο τύπος προτίμησης t είναι μια συνάρτηση αντιστοίχησης ενός συνόλου στοιχείων προτίμησης E σε μια αυστηρή σχέση προτίμησης t(E) στο στατικό μοντέλο (σε σύνολα από atoms).Το πεδίο τιμών t (dom(t)) καθορίζει τα αποδεκτά στοιχεία προτίμησης.

Το #preference(s,t)E λέγεται αποδεκτό αν το $E \in$ dom(t). Επίσης υποδηλώνει την σχέση t(E) με $\succ_s$.

- Ορίζουμε το less(cardinality) με το dom(less(cardinality))= $P(\{a,\neg a|\ a\in A\})$, όπου το P(S) δηλώνει την δύναμη του συνόλου του S και $(X,Y)\in$ less(cardinality)(E) αν και μόνο αν $|\{\ell \in E \mid X|=\ell\}|<|\{\ell \in E \mid Y|=\ell\}|$.
- Ορίζουμε το subset με το dom(subset))= $P(\{a,\neg a|\ a\in A\})$ και $(X,Y)\in$ subset(E) αν και μόνο αν $|\{\ell \in E \mid X|=\ell\}| \subset |\{\ell \in E \mid Y|=\ell\}|$.
- Ορίζουμε το *pareto* με το dom(*pareto*)= $P(\{n|\ n \in N\})$ και $(X,Y) \in$ *pareto*(E) αν και μόνο αν $\bigwedge_{name(s)\in E}(X \succcurlyeq_s Y) \wedge \bigvee_{name(s)\in E}(X \succcurlyeq_s Y)$
- Ορίζουμε το *lexico* με το dom(*lexico*)= $P(\{w:n|n\in N,w\in Z)$ και $(X,Y)\in$*lexico*(E) αν και μόνο αν $\bigvee_{w:name(s)\ \in E}((X \succcurlyeq_s Y) \wedge \bigwedge_{v:name(s')\ \in E, v<w}(X =_{s'} Y))$
- Ορίζουμε το *more(weight)* με το dom(more(weight))= $P(\{\ w:a,w:\neg a|\ a\in A,w\in Z)$ και $(X,Y) \in$ *more(weight)*(E) αν και μόνο αν $\Sigma_{(w:\ell)\ \in E,X|=}\ \ell^w > \Sigma_{(w:\ell)\ \in E,Y|=}\ \ell^w$

3.6 Παράδειγμα Λογικού Προγραμματισμού

Ένα απλό παράδειγμα για τον τρόπο κωδικοποίησης ενός προβλήματος σε λογικό προγραμματισμό είναι ο χρωματισμός ενός γράφου. Η βασική ιδέα του προβλήματος αυτού είναι έχοντας ως δεδομένο ένα γράφο, θέλουμε να χρωματίσουμε τους κόμβους του έτσι ώστε να μην έχουμε δύο γειτονικούς κόμβους με το ίδιο χρώμα.[9]



Σχήμα 3.3 Γράφος

Για την μοντελοποίηση έχουμε το κατηγόρημα (predicate) vertex/1 το οποίο είναι για τους κόμβους του γράφου και παίρνει μία παράμετρο. Επίσης έχουμε ένα κατηγόρημα για τις ακμές, edges/2 το οποίο παίρνει 2 παραμέτρους, τέλος το color/1 για τα χρώματα

που θα χρωματιστούν οι κόμβοι και το colored/2 το οποίο θα δώσει την απάντηση και παίρνει 2 παραμέτρους.

```
vertex(1..4).
edge(1,2).
edge(1,4).
edge(3,1).
edge(3,2).
color(red).
color(green).
color(blue).
```

Δεδομένα (Instances)

```
colored(V,C) :- not colored(V,D),
                not colored(V,E),
                C != D, C != E, D != E,
                color(E), color(C), color(D),
                vertex(V).
:- edge(V,U), colored(V,C), colored(U,C).
```

Κωδικοποίηση Προβλήματος

Οι δύο κανόνες για την κωδικοποίηση του προβλήματος είναι για τον περιορισμό του προβλήματος ότι δύο γειτονικοί κόμβοι δεν μπορούν αν έχουν το ίδιο χρώμα. Ο πρώτος κανόνας μας λέει ότι για να χρωματιστεί ο κόμβος V με το χρώμα C, πρέπει να μην έχει χρωματιστεί με κάποιο άλλο χρώμα D ή E, όπου τα τρία χρώματα D,C,E είναι διαφορετικά μεταξύ τους. Ο δεύτερος κανόνας είναι ένας αυστηρός περιορισμός που δηλώνει ότι απαγορεύεται να υπάρχει ακμή μεταξύ των V και U, και να έχουν χρωματιστεί με το ίδιο χρώμα

.

Το πιο πάνω γίνεται στο βήμα του Logic Program (Σχήμα 3.1) και έπειτα προχωράμε στο βήμα του Grounder όπου με την κατάλληλη εντολή μπορούμε να δούμε τα πιο κάτω.

Ο επιλυτής που χρησιμοποιήθηκε στο παράδειγμα είναι ο Clingo και τρέξαμε την πιο πάνω μοντελοποίηση με την εξής εντολή:

$ clingo --text colored.lp

```
edge(1,2). edge(1,4).  edge(3,1). edge(3,2).
color(red). color(green). color(blue).
vertex(1). vertex(2). vertex(3). vertex(4).
colored(1,blue):-not colored(1,green),not colored(1,red).
colored(2,blue):-not colored(2,green),not colored(2,red).
colored(3,blue):-not colored(3,green),not colored(3,red).
colored(4,blue):-not colored(4,green),not colored(4,red).
colored(1,green):-not colored(1,blue),not colored(1,red).
colored(2,green):-not colored(2,blue),not colored(2,red).
colored(3,green):-not colored(3,blue),not colored(3,red).
colored(4,green):-not colored(4,blue),not colored(4,red).
colored(1,blue):-not colored(1,red),not colored(1,green).
colored(2,blue):-not colored(2,red),not colored(2,green).
colored(3,blue):-not colored(3,red),not colored(3,green).
colored(4,blue):-not colored(4,red),not colored(4,green).

:-colored(2,blue),colored(1,blue).
:-colored(2,green),colored(1,green).
:-colored(2,red),colored(1,red).
:-colored(4,blue),colored(1,blue).
:-colored(4,green),colored(1,green).
:-colored(4,red),colored(1,red).
colored(1,red):-not colored(1,blue),not colored(1,green).
colored(2,red):-not colored(2,blue),not colored(2,green).
colored(3,red):-not colored(3,blue),not colored(3,green).
colored(4,red):-not colored(4,blue),not colored(4,green).
colored(1,green):-not colored(1,red),not colored(1,blue).
colored(2,green):-not colored(2,red),not colored(2,blue).
colored(3,green):-not colored(3,red),not colored(3,blue).
colored(4,green):-not colored(4,red),not colored(4,blue).
colored(1,red):-not colored(1,green),not colored(1,blue).
colored(2,red):-not colored(2,green),not colored(2,blue).
colored(3,red):-not colored(3,green),not colored(3,blue).
colored(4,red):-not colored(4,green),not colored(4,blue).

:-colored(1,blue),colored(3,blue).
:-colored(1,green),colored(3,green).
:-colored(1,red),colored(3,red).
:-colored(2,blue),colored(3,blue).
:-colored(2,green),colored(3,green).
```

Σχήμα 3.4 Παράδειγμα Εκτέλεσης για το βήμα του Grounding

Στην συνέχεια προχωράμε στο βήμα επίλυσης, όπου ο επιλυτής καλείται να δώσει την λύση για το πρόβλημα που μοντελοποιήθηκε.

Εκτελούμε την εξής εντολή $ clingo colored.lp

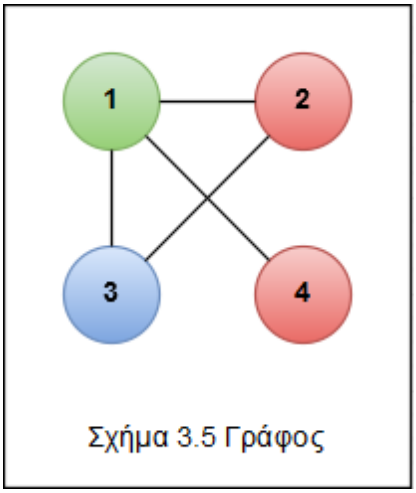
```
clingo version 4.5.4
Reading from test1.lp ...
Solving...
Answer: 1
edge(1,2) edge(1,4) edge(3,1) edge(3,2) color(red) color(green)
color(blue) vertex(1) vertex(2) vertex(3) vertex(4) colored(1,blue)
colored(2,green) colored(3,red) colored(4,red)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.002s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s
```

Σχήμα 3.4 Παράδειγμα Εκτέλεσης για το βήμα της Επίλυσης

Τέλος από το πιο πάνω αποτέλεσμα που μας δίνεται μπορούμε να δούμε ότι η λύση είναι:
colored(1,blue) colored(2,green) colored(3,red) colored(4,red)

Αυτό βρίσκεται στο βήμα Stable Models όπου η λύση εκφράζεται σε αυτή την στατική μορφή. Έπειτα ο γράφος ζωγραφίζεται.



Σχήμα 3.5 Γράφος

Κεφάλαιο 4

Δίκτυα Προτιμήσεων (CP-Networks)

4.1 Εισαγωγή

Το να μπορούμε να πάρουμε πληροφορίες σχετικά με τις προτιμήσεις των χρηστών έχει πολύ σημαντικό ρόλο στην αυτοματοποιημένη διαδικασία λήψης αποφάσεων. Για να γίνει λοιπόν εύκολη αυτή εξαγωγή των πληροφοριών έχουν αναπτυχθεί πολλές τεχνικές, μια από αυτές είναι και τα Δίκτυα Προτιμήσεων. Είναι μια πιο ποιοτική αναπαράσταση των προτιμήσεων των χρηστών που δείχνει και την εξάρτηση ή την ανεξαρτησία μεταξύ των δηλώσεων προτίμησης στο πλαίσιο του ceteris paribus. “Ceteris Paribus” σημαίνει “όλα τα άλλα ίδια”, δηλαδή η σύγκριση γίνεται μεταξύ λύσεων που διαφέρουν στην τιμή ακριβώς μιας μεταβλητής. [6]

4.2 Σχέσεις Προτίμησης

Ας υποθέσουμε ότι είναι ένας κόσμος ο οποίος αποτελείται από ένα σύνολο καταστάσεων S, και σε κάθε κατάσταση s υπάρχουν ορισμένες ενέργειες A_s που μπορούν να εκτελεστούν. Η κάθε δράση που πραγματοποιείται σε μια κατάσταση έχει και ένα συγκεκριμένο αποτέλεσμα, το σύνολο όλων των αποτελεσμάτων το

συμβολίζουμε ως Ο. Μια κατάταξη προτίμησης είναι η σχέση ≥ στο σύνολο των αποτελεσμάτων όπου: o1≥o2, το οποίο σημαίνει ότι το o1 προτιμάται εξίσου ή και περισσότερο από το o2. Επίσης μπορεί να γίνει και χρήση του o1>o2 όταν θέλουμε να δηλώσουμε την αυστηρή προτίμηση στο o1 από το o2.

Στόχος του προβλήματος λήψης αποφάσεων είναι ότι γνωρίζοντας μια συγκεκριμένη κατάσταση, να γίνει επιλογή της ενέργειας που θα έχει το προτιμότερο αποτέλεσμα. Συχνά, για κάποια κατάσταση s, ένα συγκεκριμένο αποτέλεσμα Ο, δεν μπορεί ναι προκύψει από καμία ενέργεια $a \in A_s$.Τα αποτελέσματα τα οποία μπορούν να ληφθούν ονομάζονται εφικτά αποτελέσματα (feasible outcomes).

Το "ceteris paribus" στοιχείο στα Δίκτυα Προτιμήσεων εξασφαλίζει ότι οι δηλώσεις που κάνει κάποιος είναι σχετικά ασθενείς. Συγκεκριμένα αν θεωρήσουμε ότι έχουμε δύο μεταβλητές Α και Β οι οποίες είναι ανεξάρτητες μεταξύ τους, τότε θα αξιολογηθούν και οι προτιμήσεις τους ξεχωριστά. Επομένως για παράδειγμα αν a1>a2 και b1>b2 τότε καθαρά το a1b1 είναι το αποτέλεσμα το οποίο προτιμάται περισσότερο, ενώ το a2b2 είναι το αποτέλεσμα που προτιμάται λιγότερο. Αλλά αν οι περιορισμοί που υπάρχουν θεωρήσουν το a1b1 αδύνατο, τότε πρέπει να ικανοποιηθούμε με ένα από τα a1b2 ή a2b1. Όμως δεν μπορούμε να αποφασίσουμε ποιο από τα δύο προτιμάται περισσότερο με την χρήση αυτών των αξιολογήσεων, παρόλα αυτά αν γίνει χρήση κάποιων ισχυρών συνθηκών, μπορεί να κατασκευαστεί μια εξίσωση η οποία θα προσθέσει βάρη και θα τα αντιστοιχήσει σε διαφορετικές ιδιότητες, Με μια τέτοια αποσύνθεση της συνάρτησης προτιμήσεων θα μας επιτρέψει να αναγνωρίσουμε πιο εύκολα τα προτιμότερα αποτελέσματα.

Οι πράξεις μπορούν να θεωρηθούν ως ένα σύνολο από προτιμήσεις πάνω στο σύνολο των μεταβλητών απόφασης. Ορίζουμε ένα σύνολο μεταβλητών(οι οποίες αντιστοιχούν σε χαρακτηριστικά ή γνωρίσματα) V={X1.,...,Xn} πάνω στις οποίες δηλώνουμε προτιμήσεις. Για κάθε Xi αντιστοιχεί ένα πεδίο τιμών $Dom(Xi)=\{x_{i,1},...x_{i,n}\}$ με τιμές που μπορεί να πάρει. Μια ανάθεση x από το σύνολο των μεταβλητών $X \subseteq V$ είναι μια συνάρτηση η οποία συσχετίζει κάθε μεταβλητή με ένα στοιχείο από το πεδίο τιμών της. Έχουμε δύο ειδών τέτοιες αναθέσεις :

- αν το X=V, τότε το x είναι μια πλήρης ανάθεση (complete assignment) .

- αν το X≠V, τότε το x είναι μια μερική ανάθεση (partial assignment).

Ορίζουμε ως Asst(X) το σύνολο όλων των αναθέσεων στο X ⊆ V.

4.3 Δίκτυα Προτιμήσεων(CP-Networks)

Για κάθε μεταβλητή Xi, ζητάμε από τον χρήστη να ορίσει ένα σύνολο από πατρικές μεταβλητές(parent variables) Pa(Xi), οι οποίες επηρεάζουν την προτίμηση αυτής της μεταβλητής πάνω στις τιμές της Xi. Έτσι αν δοθεί μια τιμή στο Pa(Xi) τότε ο χρήστης θα πρέπει να είναι σε θέση να καθορίσει την σειρά προτίμησης στις τιμές της Xi, παραμένοντας όλα τα άλλα ίδια. (ceteris paribus). Επιπρόσθετα ο χρήστης ζητείται να δώσει τις προτιμήσεις του για τις τιμές της Xi για όλα τα στιγμιότυπα των μεταβλητών που υπάρχουν στο σύνολο Pa(Xi). Χρησιμοποιώντας όλες αυτές τις πληροφορίες δημιουργούμε ένα σχολιασμένο κατευθυνόμενο γράφο όπου στους κόμβους αντιστοιχούμε τις μεταβλητές του προβλήματος και ο άμεσος πρόγονος του κάθε Xi είναι ο Pa(Xi). Ο κάθε κόμβος Xi σχολιάζεται με ένα πίνακα ο οποίος ονομάζεται Conditional Preference Table (CPT). Αυτός ο πίνακας περιγράφει τις προτιμήσεις του χρήστη για τις τιμές της μεταβλητής Xi, που αντιστοιχεί στον κόμβο, για κάθε συνδυασμό των τιμών των πατρικών μεταβλητών.

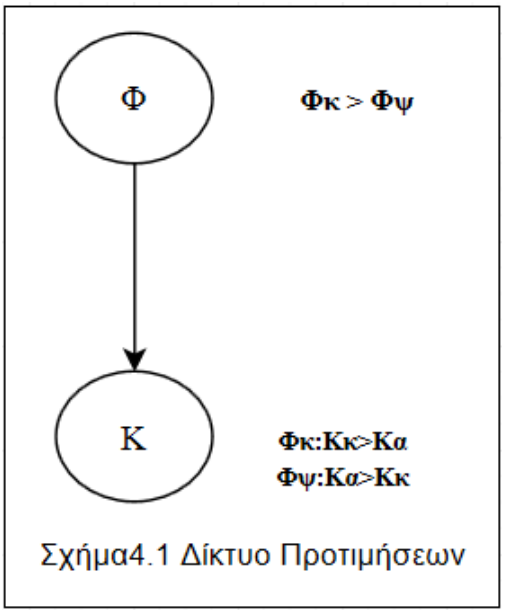
Όλες αυτές τις δομές τις ονομάζουμε Conditional Preference Networks ή CP-Networks(CP-Nets). Ορίζεται ως: ένα CP-Net πάνω στις μεταβλητές V={X1 ,..., Xn} είναι ένας κατευθυνόμενος γράφος G πάνω στα Xi,..,Xn, όπου οι κόμβοι του σχολιάζονται από υπό όρους πίνακες προτιμήσεων (CPT) CPT(Xi) για κάθε Xi ∈ V.

Για παράδειγμα :

Σε ένα δείπνο υπάρχουν δύο επιλογές για το κυρίως πιάτο το κρέας και το ψάρι. Προτιμώ να παραγγείλω κρέας παρά ψάρι. Επίσης μπορώ να συνοδέψω το φαγητό μου με κρασί λευκό ή κόκκινο. Αν παραγγείλω κρέας προτιμώ το κόκκινο κρασί αντί το άσπρο, ενώ αν παραγγείλω ψάρι προτιμώ το άσπρο κρασί από το κόκκινο.

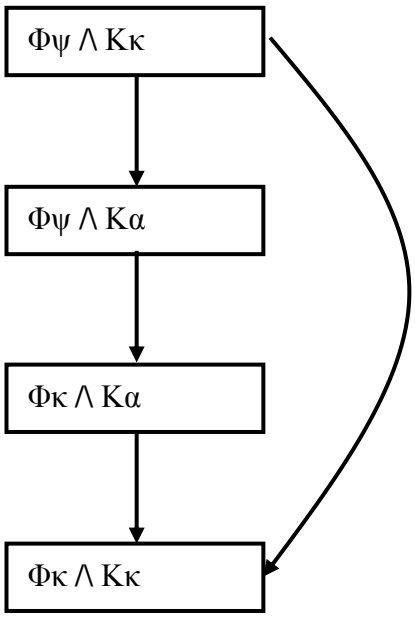
Στο πρόβλημα βλέπουμε ότι έχουμε δύο μεταβλητές το Φ και το Κ που είναι το φαγητό και το κρασί αντίστοιχα. Οι δύο μεταβλητές έχουν πεδίο τιμών το {Κρέας (Φκ), Ψάρι

(Φψ)} και {Κόκκινο(Κκ), Άσπρο(Κα)} αντίστοιχα. Οι πιο πάνω προτιμήσεις μεταφράζονται στα εξής:



Σχήμα4.1 Δίκτυο Προτιμήσεων

Το Δίκτυο Προτιμήσεων μας εκφράζει τις προτιμήσεις όσο αφορά το δείπνο όπως εξηγήσαμε πιο πάνω: Προτιμώ το Φκ από το Φψ και για το Φκ προτιμώ το Κκ ενώ για το Φψ προτιμώ το Κα.

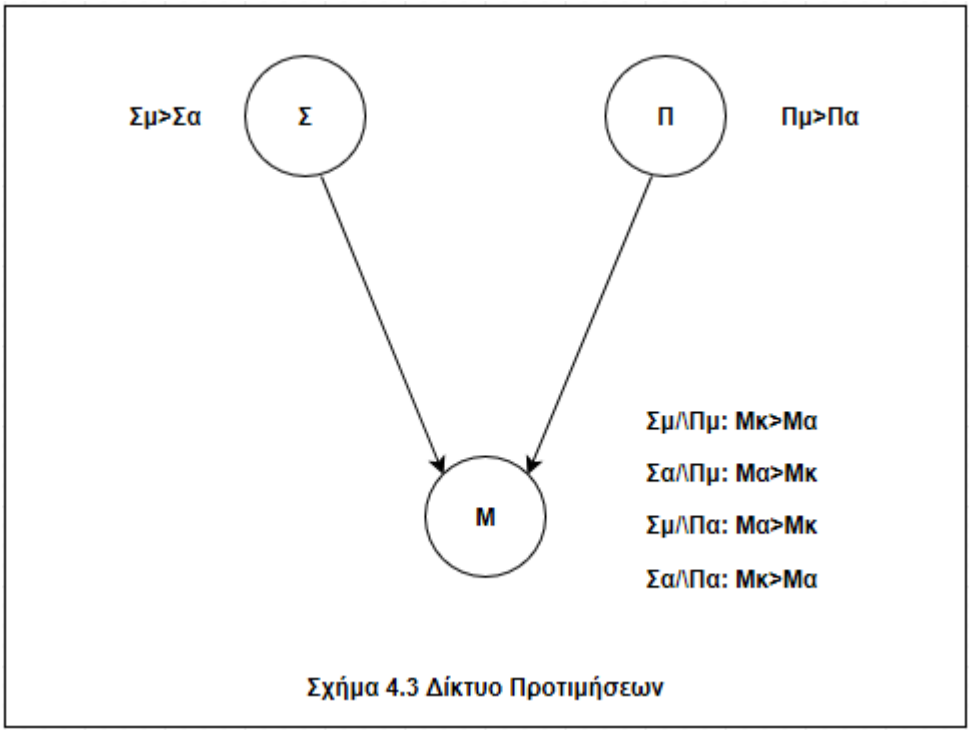


Σχήμα 4.2: Παραγόμενο Γράφημα Προτιμήσεων

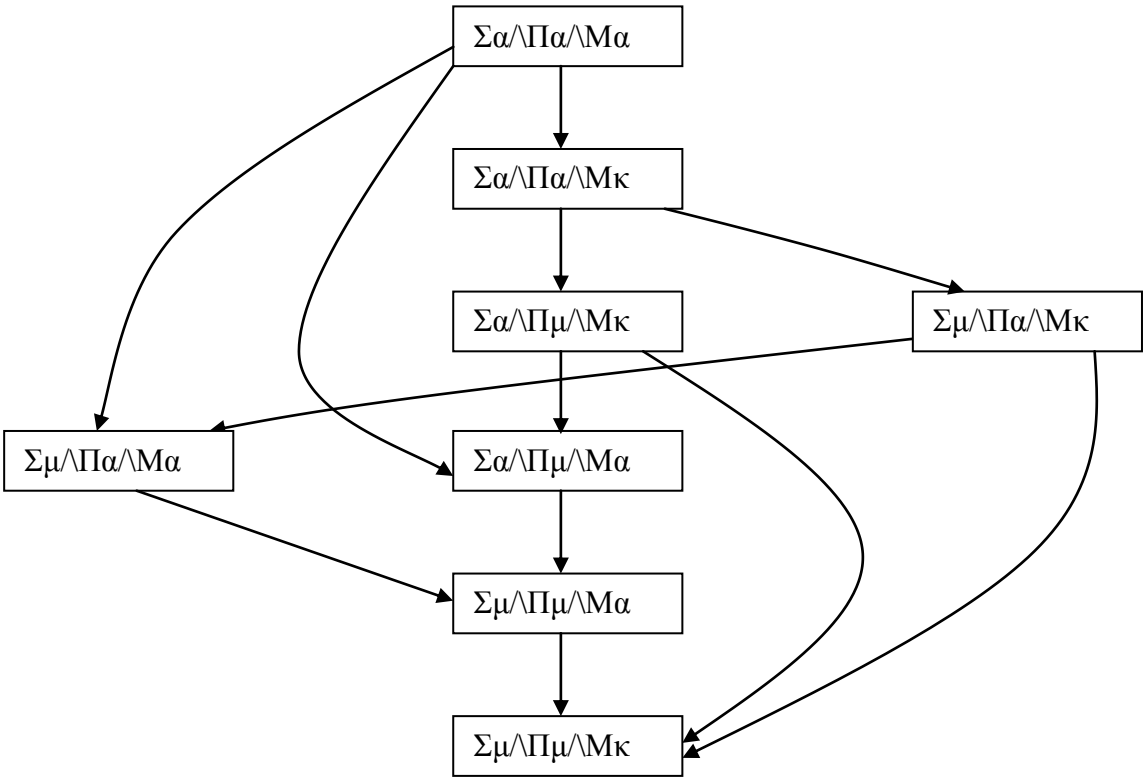
Το γράφημα προτιμήσεων είναι πάνω στα αποτελέσματα (outcomes) που προκύπτουν από το CP-Net. Μια ακμή από ένα αποτέλεσμα o_i στο o_j μας δείχνει την προτίμηση στο o_j από το o_i η οποία προσδιορίζεται άμεσα από τα CPT στο CP-Net. Για παράδειγμα το γεγονός ότι το Φψ /\ Κα προτιμάται από το Φψ /\Κκ είναι μια άμεση συνέπεια της σημασιολογίας του CPT(Κ). (Φψ: Κα>Κκ). Το Φψ /\ Κκ που είναι ο πρώτος κόμβος είναι το χειρότερο αποτέλεσμα ενώ ο τελευταίος κόμβος Φκ /\ Κκ είναι το καλύτερο προτιμότερο αποτέλεσμα.

Ένα πιο περίπλοκο παράδειγμα θα ήταν:

Οι προτιμήσεις σε θέμα ένδυσης για κάποιο δείπνο. Ας υποθέσουμε ότι έχουμε σαν μεταβλητές το σακάκι (Σ), το παντελόνι (Π) και την μπλούζα (Μ), τα οποία έχουν πεδίο τιμών τα χρώματα που μπορούν να πάρουν, {άσπρο(α) ,μαύρο(μ)} για το σακάκι και το παντελόνι και {κόκκινο(κ), άσπρο(α)} για την μπλούζα. Οι προτιμήσεις για το σακάκι και το παντελόνι είναι ίδιες, και προτιμάται το μαύρο από το άσπρο. Οι προτιμήσεις όμως για τα χρώματα της μπλούζας εξαρτώνται από τον συνδυασμό των χρωμάτων στο σακάκι και παντελόνι. Αν έχουν το ίδιο χρώμα τότε προτιμώ μια κόκκινη μπλούζα από μία άσπρη. Αν όμως έχουν διαφορετικά χρώματα θα προτιμούσα μια άσπρη μπλούζα από μια κόκκινη.



Σχήμα 4.3 Δίκτυο Προτιμήσεων



Σχήμα 4.4: Παραγόμενο Γράφημα Προτιμήσεων

4.4Άκυκλικά Δίκτυα Προτιμήσεων

Στα Δίκτυα Προτιμήσεων υπάρχουν περιπτώσεις στις οποίες οι εξαρτήσεις των μεταβλητών δημιουργούν κύκλους. Οι κύκλοι δείχνουν αλληλεξάρτηση μεταξύ των εμπλεκόμενων μεταβλητών. Αυτό όμως κάνει το πρόβλημα πιο περίπλοκο και ενδεχομένως και μη ικανοποιήσιμο αφού οι μεταβλητές θεωρούνται το ίδιο σημαντικές. Για αυτό προτιμούμε ένα ακυκλικό Δίκτυο Προτιμήσεων το οποίο έχει καθαρές εξαρτήσεις μεταξύ των μεταβλητών και προτιμήσεις για αυτές. Έτσι το πρόβλημα καθίσταται πιο απλό και ικανοποιήσιμο.

4.5 Βέλτιστο Αποτέλεσμα

Μια από τις σημαντικές ιδιότητες των Δίκτυο Προτιμήσεων είναι ότι δεδομένου ενός CP-Network N,μπορούμε πολύ εύκολα να προσδιορίσουμε το καλύτερο αποτέλεσμα μεταξύ μιας κατάταξης προτιμήσεων που ικανοποιεί το N.

Για να παράγουμε ένα βέλτιστο αποτέλεσμα σαρώνουμε απλά το δίκτυο, από την αρχή μέχρι το τέλος βάζοντας σε κάθε μεταβλητή την προτιμότερη τιμή λαμβάνοντας υπόψη το στιγμιότυπο των προγόνων του. Το δίκτυο αυτό, καθορίζει ένα μοναδικό βέλτιστο αποτέλεσμα.

Μπορούμε να επισημάνουμε ότι το πρόβλημα, όταν υπάρχουν περιορισμοί, γίνεται πιο πολύπλοκο και η εύρεση του βέλτιστου αποτελέσματος με απλή σάρωση του Δικτύου δεν είναι εφικτή. Θα πρέπει καθώς γίνεται η σάρωση στο δίκτυο να υπολογίζονται και οι αντίστοιχοι περιορισμοί και αυτό επηρεάζει τις τιμές που θα επιλεγούν, επομένως και το αποτέλεσμα δεν είναι το βέλτιστο αλλά το προτιμότερο δυνατό αποτέλεσμα.

4.6 Σύγκριση Αποτελεσμάτων

Μια άλλη ιδιότητα την οποία υποστηρίζει το μοντέλο αναπαράστασης προτιμήσεων είναι η σύγκριση αποτελεσμάτων. Δεδομένου δύο αποτελεσμάτων μπορούμε να ξεχωρίσουμε πιο από τα δύο είναι το προτιμότερο. Για αυτή την σύγκριση υπάρχουν δύο τρόποι:

- Ερωτήσεις/Σχέσεις Κυριαρχίας (Dominance Queries) :Δεδομένου ενός CP-Net N και δύο αποτελεσμάτων ο και ο', ψάχνουμε το αν ισχύει το N |=ο>ο'. Αν αυτή η σχέση υπάρχει, τότε το αποτέλεσμα ο προτιμάται από το ο' και λέμε ότι το ο κυριαρχεί στο ο' ως προς το N.
- Ερωτήσεις/ Σχέσεις Κατάταξης(Ordering Queries): Δεδομένου ενός CP-Net N και δύο αποτελεσμάτων ο και ο', ψάχνουμε το αν ισχύει το N |≠ο>ο'. Αν αυτή η σχέση υπάρχει, τότε υπάρχει μια ταξινόμηση βάση προτιμήσεων συνεπής με το N, στην οποία ο>ο'. Με βάση αυτή τη γνώση που εκφράζεται από το N μπορούμε να πούμε ότι το ο ταξινομείται 'over' (πάνω από το) ο'.

Οι σχέσεις κατάταξης είναι πιο αδύναμες από τις σχέσεις κυριαρχίας. Αν ισχύει το N |=ο>ο' τότε σίγουρα ισχύει και το N |≠ο>ο'.

Κεφάλαιο 5

Μοντελοποίηση Προτιμήσεων σε Προτασιακή Βελτιστότητα

5.1 Εισαγωγή

Σκοπός μας είναι, ότι δεδομένου ενός προβλήματος περιορισμών σε ένα σύνολο μεταβλητών με κάποιο πεδίο τιμών, να γίνει η αναπαράσταση των περιορισμών του προβλήματος αυτού σε Κανονική Συζευκτική Μορφή. Με αυτή την μορφή θα δοθεί στον επιλυτή και αυτός με την σειρά του θα μας δώσει την ανάθεση τιμών από το πεδίου τιμών της κάθε μεταβλητής. Με αυτή την ανάθεση το πρόβλημα θα ικανοποιείται και η λύση του θα πρέπει να είναι η συγκεκριμένη ανάθεση.

Το πρόβλημα το οποίο καλούμαστε να μοντελοποιήσουμε είναι το ακόλουθο :
Δοθέντος ενός συνόλου από μεταβλητές που έχουν αριθμητικό πεδίο τιμών και κάποιους περιορισμούς μεταξύ των μεταβλητών αυτών, όπως επίσης και κάποιες προτιμήσεις που τις αφορούν, να μοντελοποιήσουμε το πρόβλημα σε προτασιακή ικανοποίηση. Με την μοντελοποίηση αυτή θα περιορίσουμε την λύση έτσι ώστε να δοθεί ακριβώς μια τιμή από το πεδίο τιμών κάθε μεταβλητής και αυτές οι τιμές να μπορούν να ικανοποιήσουν τους περιορισμούς που δόθηκαν. Αυτή η ανάθεση θα πρέπει να είναι η προτιμότερη λαμβάνοντας υπόψη τις προτιμήσεις που δόθηκαν και από τις οποίες δημιουργήθηκε το σχετικό Δίκτυο Προτιμήσεων.

5.2 Μετατροπή των Βασικών Περιορισμών του Προβλήματος σε CNF

Το πρόβλημα που καλούμαστε να επιλύσουμε έχει κάποιους περιορισμούς, των οποίων η ικανοποίηση τους είναι απαραίτητη. Οι περιορισμοί αυτοί αναφέρονται σε ζευγάρια μεταβλητών και τιμές από το πεδίο τιμών τους, για τις οποίες δεν θέλουμε την συνύπαρξη τους στην λύση. Πιο συγκεκριμένα δεν θέλουμε τον συνδυασμό της συγκεκριμένης ανάθεσης των τιμών αυτών στις μεταβλητές που συσχετίζονται, αλλά επιτρέπεται η ανάθεση μίας από τις δύο τιμές στις μεταβλητές.

5.2.1 Μορφή Αρχείου Εισόδου των Περιορισμών

Αυτοί οι περιορισμοί δίνονται μέσω ενός αρχείου με συγκεκριμένη μορφή. Οι περιορισμοί αυτοί έχουν την μορφή με την οποία μας επιστρέφονται από το εργαλείο το οποίο χρησιμοποιήσαμε και θα εξηγηθεί στην συνέχεια. Το αρχείο αυτό μας λέει επιπλέον πόσες είναι συνολικά οι μεταβλητές μας και το πεδίο τιμών τους. Στην πρώτη γραμμή του αρχείου αυτού δίνονται αυτά, συν το πόσα ζευγάρια μεταβλητών σχετίζονται σε περιορισμούς και πόσα ζεύγη τιμών από τα πεδία τους δεν πρέπει να συνυπάρχουν στην ανάθεση. Έπειτα δίνονται για κάθε ζευγάρι μεταβλητών τα ζευγάρια τιμών.

Για παράδειγμα ένα αρχείο εισόδου έχει την παρακάτω μορφή:

A B C D

X1 X2: (val val) (val val)

X1 X3: (val val) (val val)

X2 X3: (val val) (val val)

Στην πρώτη γραμμή του αρχείου εισόδου τα A, B, C και D ορίζονται ως εξής:
Το A αναφέρεται στον αριθμό των μεταβλητών του προβλήματος, το B στο μέγεθος του πεδίου τιμών των μεταβλητών, όπου όλες οι μεταβλητές έχουν τον ίδιο μέγεθος. Το C στον αριθμό των ζευγαριών των μεταβλητών, και η τιμή D αναφέρετε στο πόσα ζευγάρια με τιμές του πεδίου τιμών έχουμε περιορισμό για το κάθε ζευγάρι μεταβλητών.

Οι επόμενες γραμμές είναι της μορφής :
$\_X_i \_X_j$:_(val_val)_(val_val)_…._(val_val)
Ο αριθμός των παρενθέσεων είναι ίσος με τον αριθμό των ζευγαριών από το πεδίο τιμών των μεταβλητών που ορίσαμε πιο πριν (η τιμή C). Τα ζευγάρια αυτά ορίζουν την απαγόρευση της να συνύπαρξης της ανάθεσης τους στις μεταβλητές που αντιστοιχούν. Το ‘_’ είναι για να δείξουμε ότι μεταξύ των στοιχείων αυτών έχουμε κενό, δεν περιέχεται στην μορφή του αρχείου, είναι για λόγους επεξήγησης της μορφής. Διαχωρίζουμε την γραμμή με ‘:’.Στο αριστερό μέρος γράφουμε την πρώτη μεταβλητή και μετά την δεύτερη, αφήνοντας μεταξύ τους κενό, και στο δεξί μέρος της πρότασης γράφουμε σε παρενθέσεις τα ζεύγη των τιμών αφήνοντας κενό μεταξύ των τιμών αλλά και των παρενθέσεων.

Έστω ότι έχουμε το πιο κάτω παράδειγμα:

4 3 3 2

0 1: (0 1) (0 2)

1 3: (1 0) (1 2)

2 3: (1 0) (2 0)

Όπως μπορούμε να συμπεράνουμε από το παράδειγμα, το πρόβλημα μας αποτελείται από 4 μεταβλητές οι οποίες είναι οι {0,1,2,3}, το πεδίο τιμών για κάθε μεταβλητή είναι μεγέθους 3, συγκεκριμένα οι τιμές {0,1,2}. Ο αριθμός των ζευγαριών των μεταβλητών που θα ορίσουμε τους περιορισμούς είναι 3 και για κάθε ένα από τα ζευγάρια καθορίζουμε 2 ζεύγη τιμών τους τα οποία δεν μπορούν να ανατεθούν σε μία λύση πχ (0 1),(0,2). για τις μεταβλητές 0 και 1.

5.2.2 Random Uniform CSP Generator

Για την δημιουργία των Περιορισμών Ικανοποίησης χρησιμοποιήθηκε το εργαλείο Random Uniform CSP Generator, το οποίο είναι σε γλώσσα C.[10]

Το εργαλείο αυτό δημιουργήθηκε λόγω του ότι πολλοί ερευνητές που ασχολούνται με τα Προβλήματα Ικανοποίησης Περιορισμών χρησιμοποιούν τυχαίες περιπτώσεις για να αξιολογήσουν τους αλγορίθμους ικανοποίησης περιορισμών. Παρόλο που η αξιολόγηση ενός αλγορίθμου για Προβλήματα Ικανοποίησης Περιορισμών πρέπει να γίνεται σε πραγματικά προβλήματα τα οποία υπάρχουν στον κόσμο, υπάρχει μια ευρεία συναίνεση σχετικά με την αξία του "εργαστηριακού" πειράματος σε τυχαία προβλήματα.

Η χρήση τυχαίων προβλημάτων προσφέρει πολλά πλεονεκτήματα για την αξιολόγηση της απόδοσης των αλγορίθμων. Κάποια από αυτά είναι :

- Η παραγωγή μεγάλων ποσοτήτων έτσι ώστε να μπορούν να αναφερθούν στατιστικά μέσα και διακυμάνσεις.
- Είναι εύκολη η αλλαγή των παραμέτρων του Random Uniform CSP Generator για την δημιουργία των τυχαίων δειγμάτων.
- Είναι εύκολη η εύρεση των παραμέτρων (που δίνονται για την δημιουργία των προβλημάτων) που μπορούν αλλαχθούν έτσι ώστε να παραχθούν προβλήματα για οποία υπάρχει η πιθανότητα 50% να μπορούν να λυθούν. Τέτοια προβλήματα είναι δύσκολα για αυτό και αναδεικνύουν τις διαφορές στην απόδοση του αλγορίθμου.

Η εκδοχή του προγράμματος αυτού δέχεται 4 παραμέτρους :

- Το αριθμό των μεταβλητών του προβλήματος.
- Το αριθμό του πεδίου τιμών των μεταβλητών.
- Τον αριθμό των περιορισμών. Είναι δυαδικοί περιορισμοί. Επιλέγονται τυχαία από μια ομοιόμορφη κατανομή. Ο αριθμός αυτός μπορεί να είναι ακέραιος ή κλάσμα ανάμεσα στο 0 και 1. Αν ένα πρόβλημα έχει 20 μεταβλητές τότε ο μέγιστος αριθμός περιορισμών είναι 20*19/2=190. Ένα συγκεκριμένο πρόβλημα μπορεί να οριστεί με ένα αριθμό περιορισμών ίσο με 95 ή 0.5. Η συνάρτηση C παίρνει την παράμετρο αυτή σαν ακέραιο αριθμό και δεν χρειάζεται στρογγυλοποίηση.
- Η στενότητα κάθε περιορισμού. Όλοι οι περιορισμοί έχουν την ίδια στενότητα. Αναφέρεται στον αριθμό των ζευγαριών από τιμές. Τα ζευγάρια επιλέγονται τυχαία από μια ομοιόμορφη κατανομή. Μπορεί να οριστεί σαν ακέραιος ή κλάσμα μεταξύ 0 και 1. Αν ένα πρόβλημα έχει μεταβλητές με πεδίο τιμών το 5, τότε ο μέγιστος αριθμός ζευγαριών με τιμές που δεν μπορούν να συνυπάρχουν στην λύση του προβλήματος είναι 5*5=25.

Το πρόγραμμα χρησιμοποιεί μια ψευδό-τυχαία γεννήτρια, η οποία κάνει χρήση των συναρτήσεων rand() ή random(). Επίσης είμαστε σίγουροι ότι δεν υπάρχουν διπλές περιπτώσεις .

Παράδειγμα εισόδου και την έξοδο του προγράμματος.
urbcsp 100 10 10 10 100 100

100 Μεταβλητές με πεδίο τιμών μεγέθους 10, 10 περιορισμοί με 10 ζευγάρια από το πεδίο τιμών του κάθε περιορισμού .100 ένας αρνητικός αριθμός, σημαίνει να ξεκινήσει μια τυχαία ακολουθία από ψευδό-τυχαίους περιορισμούς. Ένας θετικός αριθμός σημαίνει συνέχισε με την ίδια ακολουθία. Και τέλος θα δοθούν 100 διαφορετικά παραδείγματα.
Η έξοδος του είναι:

```
Instance 99
 17  19: (6 6) (2 0) (3 8) (5 7) (9 6) (2 7) (5 6) (8 2) (9 9) (4 8)
 57  94: (0 3) (0 1) (8 0) (2 2) (7 6) (9 1) (8 4) (3 0) (9 2) (8 8)
 10  28: (5 0) (4 6) (9 2) (8 2) (1 2) (3 5) (4 8) (1 1) (3 3) (4 0)
  1  90: (1 4) (4 5) (5 3) (7 8) (7 2) (7 1) (0 0) (0 4) (0 5) (1 9)
 55  64: (2 0) (5 9) (0 8) (0 2) (9 0) (5 1) (5 4) (2 7) (1 6) (5 0)
  9  32: (0 5) (1 1) (6 3) (1 8) (2 4) (5 6) (3 5) (2 8) (9 9) (5 3)
  3  12: (0 7) (3 6) (8 8) (0 8) (6 1) (1 4) (2 0) (3 2) (4 1) (3 0)
 52  69: (5 5) (7 8) (8 2) (1 8) (9 7) (9 2) (9 3) (3 1) (9 9) (4 8)
 11  59: (9 0) (0 1) (8 7) (5 8) (7 4) (2 2) (2 1) (8 4) (9 8) (6 9)
 14  44: (9 0) (2 4) (3 3) (5 0) (2 7) (1 4) (3 9) (9 6) (6 8) (7 0)
```

Για διάφορα τυχαία προβλήματα έγινε χρήση του πιο πάνω προγράμματος παίρνοντας τα αρχεία εξόδου του και με κάποια μικρό αλλαγή δημιουργήσαμε τα αρχεία εισόδου που εξηγήθηκαν στο Υποκεφάλαιο 5.2.1. Η αλλαγή που έγινε είναι να προστεθεί η πρώτη γραμμή του αρχείου όπου δίνει τις πληροφορίες του προβλήματος (μεταβλητές, πεδίο τιμών, περιορισμοί, ζευγάρια περιορισμών).

5.2.3 Μετατροπή των Περιορισμών σε CNF

Οι περιορισμοί αυτοί του προβλήματος είναι απαραίτητο να ικανοποιηθούν, είναι βασικοί περιορισμοί του προβλήματος και καθορίζονται από την αρχή. Για αυτό το λόγο οι όροι (clauses) που αφορούν αυτούς τους περιορισμούς, πρέπει να έχουν βάρος μεγαλύτερο από κάθε άλλο όρο (clause).

.

Κάθε παρένθεση μεταφράζεται σε ένα όρο (clause) ο οποίος είναι της μορφής :

w_-valvar1_-valvar2_0

Όπως αναφέραμε και πιο πάνω, ο όρος (clause) αρχίζει με το βάρος που του αντιστοιχεί, το οποίο υπολογίζεται ανάλογα με το πρόβλημα και με κάποιους άλλους υπολογισμούς. Οι υπολογισμοί αυτοί θα εξηγηθούν σε μεταγενέστερο υποκεφάλαιο πιο αναλυτικά. Έπειτα συνεχίζουμε με την άρνηση την πρώτης τιμής που αντιστοιχεί στην πρώτη μεταβλητή και μετά με την άρνηση της τιμής της δεύτερης μεταβλητής.

Η πιο πάνω μορφή προκύπτει από εφαρμογή κανόνων για προτασιακούς τύπους. Ο προτασιακός τύπος για αυτούς τους περιορισμούς είναι:

Not (valvar1/\valvar2) (το οποίο μεταφράζεται ως: δεν ισχύει (valvar1 και valvar2)

Με εφαρμογή του κανόνα De Morgan η πρόταση γίνεται
=>not valvar1\/ not valvar2
Που αυτό εξασφαλίζει ότι ή καμία από τις τιμές αυτές δεν θα γίνει ανάθεση στις μεταβλητές που αντιστοιχούν ή μόνο μία από τις δύο μπορεί να ανατεθεί. Αυτό δηλαδή που ορίζει και ο περιορισμός.

5.3 Περιορισμοί που αφορούν το Πεδίο Τιμών

Εκτός από τους περιορισμούς που δίνονται για το πρόβλημα, προκύπτουν και κάποιοι άλλοι βασικοί περιορισμοί που επίσης η ικανοποίηση τους είναι αναγκαία και απαραίτητη. Οι περιορισμοί αυτοί οφείλονται στο ότι για κάθε μεταβλητή θα πρέπει να της ανατεθεί αποκλειστικά μόνο μια τιμή από το πεδίο τιμών της. Αλλά και σε όλες τις μεταβλητές πρέπει σίγουρα να γίνει μία ανάθεση τιμής, δεν γίνεται μία μεταβλητή να μην πάρει τιμή.

Επομένως δημιουργούμε κάποιους ρητούς όρους (hard clauses), με το ίδιο βάρος το οποίο δώσαμε και πριν στους περιορισμούς που δίνονται με τον ορισμό του προβλήματος. Όπως αναφέραμε και προηγουμένως το βάρος αυτό θα αντιστοιχεί σε μία πολύ μεγάλη τιμή, η οποία θα μας εξασφαλίσει την ικανοποίηση αυτών των περιορισμών.

Η λογική πίσω από αυτούς τους περιορισμούς είναι ότι δεν θέλουμε την συνύπαρξη δύο τιμών μίας μεταβλητής στην λύση. Μετατρέποντας το πρόβλημα από ΠΙΠ σε πρόβλημα Προτασιακής Ικανοποιησιμότητας αναπαριστούμε την ανάθεση μιας τιμής του πεδίου τιμών μιας μεταβλητής στην δημιουργία μιας αντίστοιχης μεταβλητής στο Πρόβλημα Ικανοποιησιμότητας. Δηλαδή αν Xi πάρει την τιμή j τότε η νέα μεταβλητή είναι η Xij. Επομένως συμπεραίνουμε ότι θα πρέπει να δημιουργήσουμε περιορισμούς που να δηλώνουν ότι για παράδειγμα η Xij και Xil,όπου j και l δύο διαφορετικές τιμές του πεδίου ορισμού της Xi να μην μπορούν να γίνουν αληθείς. Η μετατροπή αυτή γίνεται και για τους περιορισμούς του προβλήματος στο Υποκεφάλαιο 5.2.3.

Η μετατροπή στην παρούσα διπλωματική έχει γίνει με συγκεκριμένο τρόπο εφόσον ασχολούμαστε με αριθμητικές μεταβλητές που έχουν ως πεδίο τιμών αριθμούς. Επίσης για κάθε νέα μεταβλητή δόθηκε μια νέα αριθμητική αντιστοίχηση έτσι ώστε να μπορεί να δοθεί στον επιλυτή. Η μετατροπή αυτή γίνεται ως εξής:

Πολλαπλασιάζουμε τον αριθμό της μεταβλητής με το μέγεθος του πεδίου τιμών της και προσθέτουμε την τιμή του πεδίου τιμών που θέλουμε +1. Δηλαδή για την μεταβλητή 0 του προβλήματος, με μέγεθος πεδίου τιμών το 3, για να βρούμε την αντιστοίχιση της για την $1^{η}$ τιμή του πεδίου τιμών της (η 0 εφόσον αρχίζουμε από το 0 τα πεδία τιμών), θα γίνει η πράξη (0*3)+0+1=1. Επομένως στην Προτασιακή Ικανοποίηση η τιμή 1 δηλώνει ότι μιλάμε για την τιμή 0 του πεδίου τιμών της μεταβλητής 0. Προσθέτουμε το +1 στις πράξεις λόγω του ότι ο επιλυτής δέχεται αριθμητικές μεταβλητές αρχίζοντας από το 1. Για την μεταβλητή 2 και την $3^{η}$ τιμή της (=2 αφού τα πεδία τιμών αρχίζουν από το 0), θα υπολογίζαμε το (2*3)+2+1=9 άρα το 9 είναι η νέα μεταβλητή.

Ο προτασιακός τύπος για αυτό είναι:
Not ($X_{ij} \wedge X_{il}$)
Με την χρήση του κανόνα De Morgan
=> (not $X_{ij} \vee$ not X_{il})
Και σε μορφή την οποία δέχεται ο επιλυτής θα γραφτεί ένας όρος (clause) της μορφής :
w_-var1_-var2_0
Όπου w είναι το βάρος του όρου (clause) και τα var1 var2 οι μεταβλητές στο πρόβλημα Προτασιακής Ικανοποιησιμότητας που αντιστοιχούν σε 2 τιμές μιας μεταβλητής στο ΠΙΠ.

Για κάθε μεταβλητή θα δημιουργηθούν n τέτοιες νέες μεταβλητές και θα γραφτούν όροι (clauses) με τα διαφορετικά ζευγάρια που μπορούν να προκύψουν από αυτές. Για κάθε μία θα γραφτεί ένας όρος (clause) για οποιοδήποτε συνδυασμό με τις υπόλοιπες. Αυτό θα γίνει και για τις n μεταβλητές. Μπορούμε εδώ να επισημάνουμε ότι το (not $X_{ij} \vee$ not X_{il}) είναι ισοδύναμο με το (not $X_{il} \vee$ not X_{ij}), άρα ένας όρος (clause) θα δημιουργηθεί για αυτό τον συνδυασμό.

Με το πιο πάνω όμως αντιμετωπίζουμε το πρόβλημα του ότι καμιά από τις μεταβλητές αυτές δεν θα γίνει αληθής στην λύση που θα μπορούσε να δώσει ο επιλυτής. Έτσι λοιπόν πρέπει να δημιουργηθούν νέοι περιορισμοί για την εξασφάλιση του ότι μια από αυτές τις n μεταβλητές, που αντιστοιχούν στις n τιμές του πεδίου τιμών κάποιας από τις μεταβλητές στο ΠΙΠ, θα πρέπει να γίνει αληθής. Αυτό σε προτασιακό τύπο μεταφράζεται σε :

$\vee_{j=1..n}$ Xij όπου Xi μια μεταβλητή από το ορισμό του προβλήματος σε ΠΙΠ

= > Xi1∨Xi2∨…..∨Xin

Και σε μορφή την οποία δέχεται ο επιλυτής θα γραφτεί ένας όρος (clause) της μορφής :

w_var_1_var_2_....._var_n_0

Όπου w είναι το βάρος του όρου (clause) και τα var_1 var_2… var_n οι μεταβλητές στο πρόβλημα Προτασιακής Ικανοποιησιμότητας που αντιστοιχούν σε n τιμές μιας μεταβλητής στο ΠΙΠ.

5.4 Δημιουργία του Δικτύου Προτιμήσεων

5.4.1 Εισαγωγή

Έχοντας ένα σύνολο από μεταβλητές, ο χρήστης επιλέγει ένα υποσύνολο των μεταβλητών πάνω στις οποίες θα εκφράζει κάποιες προτιμήσεις εξάρτησης. Αυτή η εξάρτηση μεταφράζεται σε Δίκτυο Προτιμήσεων. Δηλαδή φτιάχνεται ένα Δίκτυο Προτιμήσεων (CP-Net) το οποίο σχολιάζεται με τις προτιμήσεις, που έχει δώσει ο χρήστης. Οι σχολιασμοί αφορούν την σειρά προτίμησης της ανάθεσης κάποιας τιμής της μεταβλητής με βάση την ανάθεση που έχει γίνει στον άμεσο πρόγονο του στο δίκτυο. Με την κατάλληλη σάρωση του δικτύου αυτού οι προτιμήσεις μεταφράζονται σε όρους (clauses) στο CNF και αναθέτουμε το κατάλληλο βάρος με βάση την βαρύτητα της μεταβλητής στις σχέσεις προτίμησης.

5.4.2 Μορφή του Δικτύου Προτιμήσεων

Έχουμε σαν δεδομένο ένα CP-Net το οποίο είναι της μορφής δυαδικού δέντρου. Δηλαδή σε κάθε επίπεδο του δέντρου ο κάθε κόμβος έχει 2 απογόνους. Ξεκινώντας από την ρίζα η οποία έχει δύο παιδιά τότε τα παιδιά της έχουν από δύο παιδιά. Επομένως το κάθε επίπεδο έχει 2^i (όπου το i το επίπεδο) κόμβους. Το δυαδικό δέντρο δεν είναι απαραίτητο να είναι τέλειο αλλά να είναι πλήρες. Τέλειο δέντρο λέγεται το δέντρο όπου όλα τα φύλλα του είναι στο ίδιο επίπεδο. Πλήρες δέντρο είναι το δέντρο όπου σε κάθε επίπεδο, εκτός από το τελευταίο, είναι γεμάτο, δηλαδή το επίπεδο έχει όλους τους κόμβους του, και όλοι οι κόμβοι του τελευταίου επίπεδου είναι εντελώς αριστερά

Η μορφή του δυαδικού δέντρου εξαρτάται κυρίως από το σύνολο των μεταβλητών που λαμβάνουν μέρος στους περιορισμούς. Μπορεί να μην λαμβάνουν μέρος αρκετές μεταβλητές στις προτιμήσεις έτσι ώστε να δημιουργηθεί ένα τέλειο δέντρο, αλλά ένα πλήρες δένδρο.

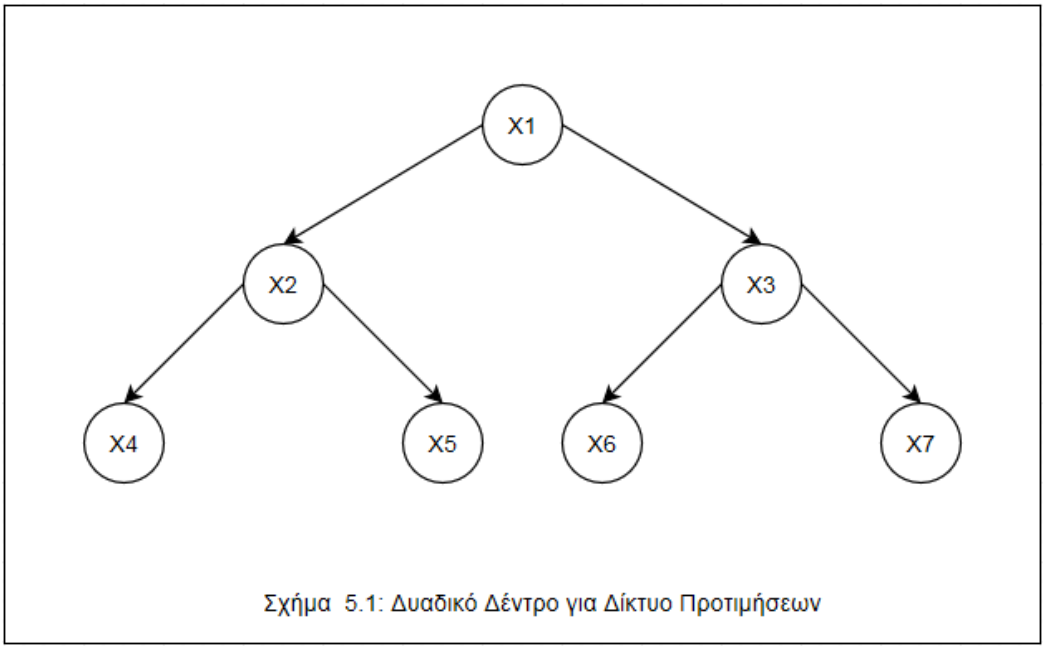
Το δέντρο, που χρησιμοποιείται ως Δίκτυο Προτιμήσεων, δημιουργείται τυχαία για λόγους πειραματικής αξιολόγησης. Δηλαδή με βάση τον αριθμό των μεταβλητών που ο χρήστης επιθυμεί, επιλέγεται ένα υποσύνολο των μεταβλητών με πλήθος ο αριθμός που δόθηκε. Το υποσύνολο είναι τυχαίο κάθε φορά. Για πολλές εκτελέσεις όμως του ίδιου προβλήματος πιθανότατα να επιλεχτεί το ίδιο υποσύνολο. Από αυτό το σύνολο δημιουργείται, και πάλι τυχαία, το δυαδικό δέντρο, δηλαδή ο καθορισμός των εξαρτήσεων μεταξύ των μεταβλητών είναι με τυχαία επιλογή. Επίσης για πολλές εκτελέσεις του ίδιου προβλήματος μπορεί να προκύψει το ίδιο δέντρο. Τέλος για κάθε κόμβο-μεταβλητή δημιουργούμε τις προτιμήσεις στο πεδίο τιμών της για κάθε διαφορετική τιμή του πεδίου τιμών του πατέρα της. Και πάλι αυτό γίνεται τυχαία. Έτσι τέλος θα παραχθεί ένα τυχαίο δέντρο προτιμήσεων το οποίο θα χρησιμοποιηθεί για την μοντελοποίηση του προβλήματος.

5.4.3 Υπολογισμός του Βάρους για τους όρους των Προτιμήσεων

Έχοντας πλέον το δυαδικό μας δέντρο, το δίκτυο προτιμήσεων, καλούμαστε να υπολογίσουμε τα βάρη όρων (clauses) που θα γραφτούν για τις προτιμήσεις. Κάθε επίπεδο θα έχει το δικό του βάρος και οι κόμβοι (μεταβλητές) στο κάθε επίπεδο θα έχουν αυτό το βάρος. Ξεκινούμε τον υπολογισμό από τα κάτω προς τα πάνω, από τα φύλλα δηλαδή του δέντρου και καθώς ανεβαίνουμε το βάρος θα αυξάνεται. Με βάση το τελικό βάρος που υπολογίστηκε στην ρίζα, οι περιορισμοί που περιγράφηκαν (οι αναγκαίοι περιορισμοί) πιο πάνω θα έχουν βάρος μεγαλύτερο από αυτό αφού όπως εξηγήθηκε είναι απαραίτητη η ικανοποίηση τους. Δηλαδή το βάρος θα είναι ίσο με το άθροισμα όλων των βαρών των κόμβων του δέντρου αυτού +2.(Ή δύο φορές το βάρος της ρίζας +1)

Ξεκινούμε από τα φύλλα του δέντρου τα οποία είναι τα τελευταία στην εξάρτηση. Για κάθε φύλλο κοιτάζουμε τον άμεσο πρόγονο του, και για κάθε τιμή του πεδίου τιμών του προγόνου βλέπουμε την κατάταξη προτίμησης του φύλλου για την τιμή αυτή. Αυτό το μεταφράζουμε σε CNF μορφή, δηλαδή για κάθε συνδυασμό της τιμής του προγόνου με όλες τις τιμές του πεδίου τιμών του φύλλου που κοιτάζουμε, δημιουργούμε και ένα όρο, το βάρος του οποίου, αφού βρισκόμαστε στα φύλλα του δέντρου ακόμη ,έχουν όλα βάρος 2 για την προτιμότερη και όλες οι υπόλοιπες 1.

Παράδειγμα:



Σχήμα 5.1: Δυαδικό Δέντρο για Δίκτυο Προτιμήσεων

Έχουμε το πιο πάνω παράδειγμα όπου στο Δίκτυο Προτιμήσεων λαμβάνουν μέρος 7 μεταβλητές. Μεγαλύτερη προτεραιότητα ικανοποίησης έχει η Χ1, εφόσον οι άλλες εξαρτώνται αποκλειστικά από την τιμή που θα ανατεθεί σε αυτή. Επομένως και το βάρος στους όρους που την αφορούν θα είναι πιο μεγάλο από τα βάρη των όρων (clauses) των μεταβλητών στα πιο κάτω επίπεδα.

Η ιδέα του καθορισμού της τιμής του βάρους για κάθε όρο (clause) είναι η εξής:
Αρχίζοντας όπως αναφέραμε πιο πάνω από τα φύλλα του δένδρου (Χ4,Χ5,Χ6,Χ7 του παραδείγματος) οι όροι τους θα έχουν βάρος 2, ανεβαίνοντας επίπεδο το βάρος για τα Χ2,Χ3 είναι το άθροισμα των βαρών των κόμβων του πιο κάτω επιπέδου συν 1, στο παράδειγμα μας θα έχουν βάρος 9, εφόσον έχουμε συνολικά 4 φύλλα στο τελευταίο επίπεδο με βάρος δύο το κάθε φύλλο. Συνεχίζοντας την διάσχιση στο πιο πάνω επίπεδο το βάρος είναι το άθροισμα των βαρών όλων των πιο κάτω επιπέδων +1, δηλαδή 8+9+9+1=27(όπου 8 είναι το άθροισμα των Χ4,Χ5,Χ6,Χ7 και όπου 9 του Χ2,Χ3 αντίστοιχα). Με το ίδιο σκεπτικό προχωρούμε στα πιο πάνω επίπεδα υπολογίζοντας το βάρος των κόμβων του επιπέδου αθροίζοντας τα βάρη των κόμβων των πιο κάτω επιπέδων. Το συνολικό βάρος του δέντρου στο παράδειγμα είναι 54.

Πιο συγκεκριμένα τα βάρη αυτά που υπολογίζονται με το πιο πάνω τρόπο θα έχουν οι όροι οι οποίοι θα περιέχουν την προτιμότερη τιμή του πεδίου τιμών του κάθε κόμβου. Δηλαδή για κάθε τιμή του πατέρα, η προτιμότερη τιμή που έχει οριστεί, ο όρος που της αντιστοιχεί θα έχει το βάρος του αντίστοιχου κόμβου. Για τις υπόλοιπες τιμές στην σειρά προτίμησης οι όροι θα έχουν βάρος 1, το οποίο είναι το ελάχιστο βάρος που αποδέχεται ο επιλυτής και το οποίο δείχνει ότι αν ικανοποιηθεί δεν θα μας προσφέρει μεγάλη διαφορά ικανοποίησης.

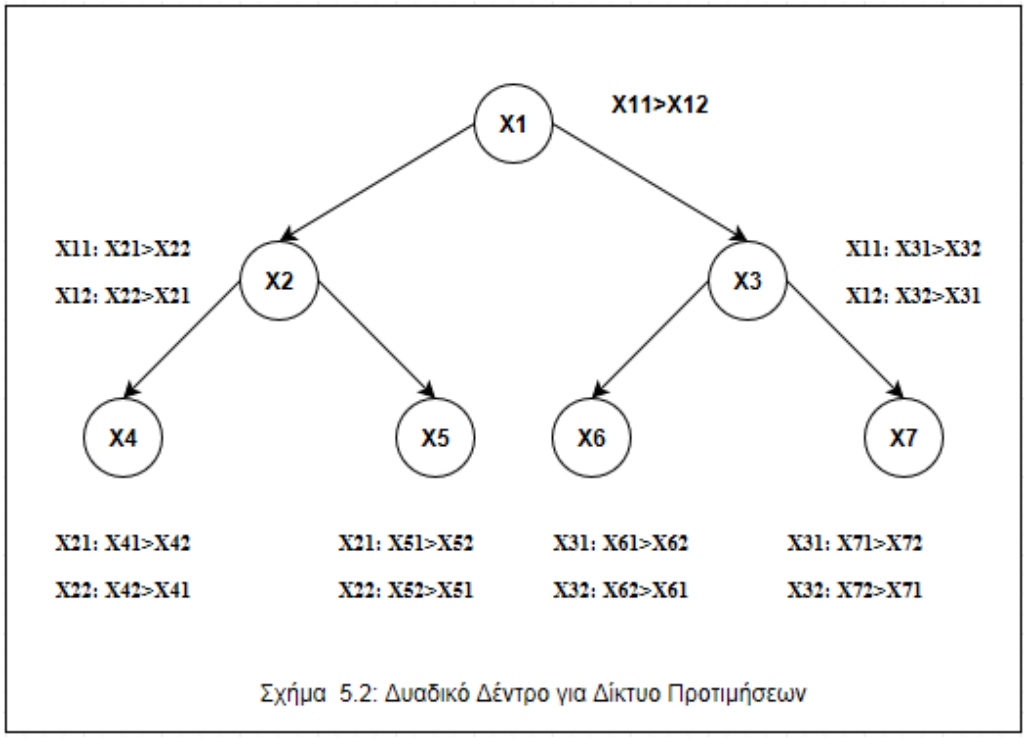
Βάζοντας πιο μεγάλο βάρος τότε είναι σαν να ορίζουμε ότι το να ικανοποιήσουμε αυτόν τον όρο είναι πιο σημαντικό από το να ικανοποιήσουμε πολλούς όρους με πιο μικρό βάρος. Επίσης έχουμε ορίσει και τον περιορισμό που μας καθορίζει ότι για κάθε μεταβλητή, μόνο μια τιμή από το πεδίο τιμών της, ανατίθεται σε αυτή. Επομένως ξέρουμε ότι για κάθε κόμβο μόνο ένας όρος θα ικανοποιηθεί. Άρα για κάθε επίπεδο του

δυαδικού δέντρου γνωρίζουμε ότι συνολικά θα ικανοποιηθούν ίσος αριθμός όρων όσος και ο αριθμός των κόμβων του επίπεδου.

5.4.4 Δημιουργία των όρων και ερμηνεία τους

Όπως προαναφέραμε για κάθε μεταβλητή στο Δίκτυο Προτιμήσεων δημιουργούμε τους κατάλληλους όρους. Οι όροι αυτοί έχουν το βάρος που τους δόθηκε όπως υπολογίσαμε πιο πάνω και συνεχίζουν με τον συνδυασμό της τιμής του πεδίου τιμών του πατέρα και των τιμών από το πεδίο τιμών της μεταβλητής που μελετάμε.

Ας υποθέσουμε ότι στο παράδειγμα μας οι μεταβλητές έχουν πεδίο τιμών {1,2} και δόθηκε το Δίκτυο Προτιμήσεων σχολιασμένο με τις προτιμήσεις για την κάθε μεταβλητή. Για την αναπαράσταση των πεδίων τιμών λέμε ότι η Xij αντιστοιχεί στην j τιμή του πεδίου τιμών της μεταβλητής Xi. Παραδείγματος χάριν το X_{11} δηλώνει ότι η μεταβλητή X_1 έχει την τιμή 1 από το πεδίο τιμών της. Αυτό προκύπτει από την μετατροπή ενός Προβλήματος Ικανοποίησης Περιορισμών σε πρόβλημα Προτασιακής Ικανοποιησιμότητας, όπου για κάθε τιμή από το πεδίο τιμών μια μεταβλητής δημιουργούμε μία νέα μεταβλητή στο Πρόβλημα Προτασιακής Ικανοποιησιμότητας (όπως εξηγήθηκε στο Κεφάλαιο 2).



Σχήμα 5.2: Δυαδικό Δέντρο για Δίκτυο Προτιμήσεων

Η μορφή του όρου που θα δημιουργήσουμε είναι η πιο κάτω :

w_-Xfi_Xsj_ 0

Όπου:

- το w είναι το βάρος (weight).
- το Xfi είναι η i τιμή από το πεδίο τιμών του πατέρα (Xf).
- το Xsj είναι η j τιμή από το πεδίο τιμών της μεταβλητής που αναφερόμαστε στον όρο (Xs).
- το '_' δεν το γράφουμε, εδώ υποδηλώνει το κενό το οποίο αφήνουμε μεταξύ των δεδομένων.
- το '-' είναι το not στην μορφή που δέχεται ο επιλυτής.
- το 0 το γράφουμε λόγω του ότι τα δεδομένα που δέχεται ο επιλυτής πρέπει να περιέχουν 0 στο τέλος του κάθε όρου. Αυτό δηλώνει ότι είναι ένας όρος και το τέλος της γραμμής .

Αυτή η μορφή του όρου προκύπτει από την υπόθεση ότι αν ο πρόγονος πάρει την τιμή i από το πεδίο τιμών του τότε η μεταβλητή θα πάρει την τιμή j από το πεδίο τιμών της. Με εφαρμογή κανόνων παίρνει την μορφή αυτή δηλαδή όχι η τιμή του προγόνου ή η τιμή της μεταβλητής.

Xfi-> Xsj => NOT Xfi OR Xsj όπου τα Xf είναι η μεταβλητή πρόγονος και Xs η μεταβλητή που μελετάμε, και $0 \leq i,j \leq n$ τιμές από το πεδίο τιμών των μεταβλητών αυτών.

Το βάρος του όρου ανάλογα με το επίπεδο στο οποίο βρισκόμαστε στον γράφο έχει υπολογιστεί ανάλογα και θα δείξει την βαρύτητα ικανοποίησης των όρων της μεταβλητής αυτής. Στην συνέχεια αρχίζουμε από την πρώτη τιμή του πεδίου τιμών της μεταβλητής του άμεσου προγόνου και για όλες τις τιμές του πεδίου τιμών της εκάστοτε μεταβλητής δημιουργούμε τους αντίστοιχους όρους. Για την προτιμότερη βάζουμε το βάρος που αντιστοιχεί στην μεταβλητή, και για τις υπόλοιπες βάζουμε βάρος 1. Έπειτα συνεχίζουμε με το ίδιο τρόπο για όλες τις τιμές που μπορούν να ανατεθούν στον πρόγονο, να δημιουργούμε όρους με τους συνδυασμούς αυτούς.

Για παράδειγμα η ανάθεση τιμής της μεταβλητής X4 (Σχήμα 5.2) εξαρτάται από την ανάθεση που θα γίνει στον άμεσο πρόγονο της (X2). Επομένως θα δημιουργηθούν οι

όροι που μας ορίζουν ότι αν ανατεθεί για παράδειγμα η X21 τότε θέλουμε να γίνει ανάθεση της X41. Αυτό σε προτασιακό τύπο έχει την μορφή
X21→ X41 (Αν X21 τότε X41)
Με απαλοιφή της συνεπαγωγής προκύπτει το:
=>not X21 \/ X41
Όπως και η μορφή του όρου στο CNF.

Οι όροι της ρίζας διαφέρουν στην μορφή τους εφόσον δεν έχει εξάρτηση με κάποιον άλλο και έχουν την μορφή:
w_dom-val _0

Γράφουμε όρους τέτοιας μορφής για όλες τις τιμές του πεδίου τιμών της μεταβλητής που βρίσκεται στην ρίζα. Η προτιμότερη τιμή της ρίζας έχει το βάρος που υπολογίστηκε για αυτή την μεταβλητή, ενώ οι υπόλοιπες βάρος 1.

5.5 Επίλυση του Προβλήματος

Μετατρέποντας τους πιο πάνω περιορισμούς και τις προτιμήσεις σε Προτασιακή Ικανοποιησιμότητα και σε CNF δημιουργούμε ένα αρχείο το οποίο θα τα περιέχει όλα αυτά.

Το αρχείο αυτό θα δοθεί σαν είσοδος στον επιλυτή. Ο επιλυτής θα βρει την κατάλληλη ανάθεση που θα ικανοποιήσει το πρόβλημα και θα επιστρέψει το αποτέλεσμά του. Με κατάλληλη επεξεργασία αυτού του αποτελέσματος, δηλαδή μετατροπή από Προτασιακή Ικανοποιησιμότητα σε Πρόβλημα Ικανοποίησης Περιορισμών παρουσιάζουμε το τελικό αποτέλεσμα, το ποιά τιμή έχει τελικά ανατεθεί στις αρχικές μεταβλητές του προβλήματος.

Η αντίστροφη διαδικασία δεν διαφέρει πολύ από πριν δηλαδή αν γνωρίζουμε ότι ο επιλυτής έθεσε την X_{ij} μεταβλητή ως αληθή, τότε ξέρουμε ότι στην μεταβλητή Xi δόθηκε η τιμή j.

5.6 Μοντελοποίηση σε Προγραμματισμό Συνόλου Απαντήσεων

Εκτός από την μοντελοποίηση του Προβλήματος Ικανοποίησης Περιορισμών σε Προτασιακή Ικανοποιησιμότητα, προσπαθήσαμε να μοντελοποιήσουμε το πρόβλημα και σε Προγραμματισμό Συνόλου Απαντήσεων.

Ο σκοπός ήταν, να μπορέσουμε έχοντας δυο διαφορετικές μοντελοποιήσεις οι οποίες θα δίνονταν στους επιλυτές, να πάρουμε τα αποτελέσματα και να τα συγκρίνουμε. Για αυτό με τον ίδιο τρόπο όπως και πριν παίρνουμε το πρόβλημα μας με τις μεταβλητές και το πεδίο τιμών τους, όπως και τους περιορισμούς και τις προτιμήσεις. Έπειτα με κατάλληλες μετατροπές μοντελοποιούμε το πρόβλημα και το δίνουμε στον κατάλληλο επιλυτή. Η λύση που θα μας δοθεί είναι η ανάθεση των τιμών στις μεταβλητές και η λύση του αρχικού προβλήματος.

Για την αναπαράσταση των μεταβλητών και του πεδίου τιμών έχουμε τα εξής κατηγορήματα (predicates) var/1 και va/1. Το var είναι για να ορίσουμε τις μεταβλητές και θα γραφτεί ως εξής var (0..(n-1))**.** όπου n είναι ο αριθμός των μεταβλητών. Με αυτό τον τρόπο δηλώνουμε το πεδίο που μπορεί να πάρει η μεταβλητή του κατηγορήματος (predicate) και δηλώνει ότι η μεταβλητή αυτή έχει την ιδιότητα να είναι var (δηλαδή μεταβλητή για το αρχικό πρόγραμμα). Το va είναι για να ορίσουμε τις τιμές του πεδίου τιμών. Θα γραφτεί va(0..(n-1))**.** όπου n το μέγεθος του πεδίου τιμών του προβλήματος.

Επομένως αρχίζουμε την μοντελοποίηση ορίζοντας τα πιο πάνω κατηγορήματα. Έπειτα συνεχίζουμε με τους κανόνες που αφορούν το πεδίο τιμών του προβλήματος. Όπως έχει εξηγηθεί οι περιορισμοί αυτοί προκύπτουν από τον ορισμό του προβλήματος, δηλαδή την ανάθεση μίας ακριβώς τιμής σε κάθε μεταβλητή. Εδώ η μορφή τους αλλάζει σε σύγκριση με την Προτασιακή Ικανοποιησιμότητα. Ορίζουμε λοιπόν ένα νέο κατηγόρημα, το val(X,Y) το οποίο μας λέει ότι το Y είναι τιμή που μπορεί να πάρει η μεταβλητή X του προβλήματος.

Αν δοθεί μια τιμή από το πεδίο τιμών της μεταβλητής X τότε δεν πρέπει να δοθεί κάποια άλλη. Έτσι για κάθε τιμή του πεδίου τιμών της κάθε μεταβλητής δημιουργούμε

ένα διαφορετικό τύπο για κάθε διαφορετικό συνδυασμό των τιμών του πεδίου τιμών της.
Ο τύπος έχει την μορφή :
val(X, ai) :-not val(X,aj),...,not val(X,aj). όπου ai τιμή του πεδίου τιμών με $0 \leq i,j \leq n$ και $i \neq j$, X είναι κάποια μεταβλητή.

Αυτό όμως δεν μας εγγυάται το ότι η μεταβλητή μας θα πάρει ακριβώς μια τιμή. Από το πιο πάνω μπορεί η μεταβλητή μας να πάρει από καθόλου μέχρι και πολλές διαφορετικές τιμές από πεδίο τιμών της. Για τον λόγο αυτό συνεχίζουμε ορίζοντας περιορισμούς ακεραιότητας (Integrity constraints). Για κάθε μεταβλητή θα γραφτεί ένας περιορισμός ο οποίος θα περιέχει όμως όλες τις τιμές του πεδίου τιμών της.
Θα είναι της μορφής:
:- val(Xi,a_0),val(Xi,a2),…,val(Xi,a_{n-1}). όπου Xi μεταβλητή και ai τιμή με $0 \leq i \leq n$.

5.6.1 Μετατροπή των Βασικών Περιορισμών σε Λογικό Προγραμματισμό

Οι περιορισμοί μεταξύ των μεταβλητών για τις αναθέσεις που απαγορεύονται να συνυπάρχουν σε μία πιθανή λύση δίνονται όπως και πριν σε ένα αρχείο. Το αρχείο δεν διαφέρει σε τίποτα και χειριζόμαστε τους περιορισμούς με τον ίδιο τρόπο.
Ο περιορισμός θα έχει την μορφή:
:- val(X, ai), val(Y,bj). με X και Y δύο διαφορετικές μεταβλητές και ai και bj τιμές από τα πεδία τους που δόθηκαν στον περιορισμό.
Για να ισχύει και να γίνει αληθής πρέπει το δεξιά μέρος να είναι ψευδής. Άρα ψευδής συνεπάγεται ψευδής τότε όλο γίνεται αληθές.

Για να γίνει η ανάθεση τιμών πρέπει να γραφτεί ένας κανόνας ο οποίος θα περιέχει μεταβλητές και ο επιλυτής με βάση την γνώση που του δόθηκε με τα πιο πάνω θα κάνει αναθέσεις σε αυτές μεταβλητές. Κάποιες από αυτές θα ικανοποιήσουν το πρόβλημα και θα τις επιστρέψει ως λύση του προβλήματος .

Ο κανόνας για αυτό κάνει χρήση ενός άλλου κατηγορήματος, του assign(X,Y) όπου δηλώνει ότι στην X έχει ανατεθεί η τιμή Y.

Ορίζεται ως εξής:

Assign(X,Y):- var(X), va(Y), val(X,Y), not assign(X,Y1), va(Y1), Y!=Y1**.**
Με λίγα λόγια αυτό μας λέει ότι για να γίνει ανάθεση του Υ στο Χ πρέπει το Χ να είναι μία μεταβλητή, το Υ να είναι μία τιμή, το Χ να μπορεί να πάρει το Υ, και να μην έχει γίνει στην Χ κάποια άλλη ανάθεση Υ1, το οποίο Υ1 να είναι και αυτό τιμή άλλα διαφορετική από την Υ.

Ο επιλυτής θα μας δώσει την λύση σε στατικό μοντέλο και με κατάλληλες επεξεργασίες θα μπορέσουμε να πάρουμε την τελική λύση, όπου θα γνωρίζουμε τί ανάθεση έγινε σε κάθε μεταβλητή.

Κεφάλαιο 6

Πειραματική Αξιολόγηση και Συμπεράσματα

6.1 Εισαγωγή

Σε αυτό το κεφάλαιο κάνουμε πειραματική αξιολόγηση των μεθόδων που παρουσιάσαμε. Μέσα από αυτή την πειραματική αξιολόγηση προσπαθήσαμε να δούμε το μέγεθος των προβλημάτων που δέχονταν τα συστήματα. Όπως επίσης και τους παράγοντες οι οποίοι επηρέαζαν τα προβλήματα και τα καθιστούσαν πιο περίπλοκα με αποτέλεσμα η εύρεση λύσης τους να ήταν πιο δύσκολη. Έπειτα κάναμε σύγκριση των μοντελοποιήσεων.

Λόγω της τυχαιότητας που υπάρχει, γίνεται περίπλοκη η εύρεση κάποιων γενικών συμπερασμάτων. Οι προτιμήσεις μπορεί να διαφέρουν σε κάθε εκτέλεση του προγράμματος και το Δίκτυο Προτιμήσεων να είναι διαφορετικό, επομένως το μέτρο σύγκρισης δεν υπάρχει.

Μπορούμε όμως να συμπεράνουμε αν για κάποιο συγκεκριμένο πρόβλημα και ένα Δίκτυο Προτιμήσεων η ανάθεση που έγινε έχει ικανοποιήσει τους απαραίτητους περιορισμούς και η λύση είναι η προτιμότερη για το αντίστοιχο προβλήματος.

6.2 Πειραματική Αξιολόγηση και Συμπεράσματα

6.2.1 Μοντελοποίηση σε Προτασιακή Ικανοποιησιμότητα

Για την Πειραματική Αξιολόγηση της μοντελοποίησης σε Προτασιακή Ικανοποιησιμότητα τρέξαμε κάποια διαφορετικά σενάρια ενός προβλήματος και με βάση τα αποτελέσματα βγάλαμε τα κατάλληλα συμπεράσματα.

Σενάριο 1: `5 2 3 2`

```
1   2: (0 1) (1 0)
1   3: (1 1) (0 1)
0   3: (0 1) (1 0)
```

Έγινε χρήση του πιο πάνω σεναρίου όπου έχουμε 5 μεταβλητές με μέγεθος πεδίου τιμών 2. Με την χρήση του προγράμματος Random Uniform CSP Generator πήραμε τους περιορισμούς αυτούς για το πρόβλημά μας.

Το σενάριο αρχικά δεν έχει Δίκτυο Προτιμήσεων. Θεωρούμε τις μεταβλητές ανεξάρτητες μεταξύ τους και χωρίς προτιμήσεις στα πεδία τιμών τους.

```
p wcnf 10 16 1
1 -3 -6 0
1 -4 -5 0
1 -4 -8 0
1 -3 -8 0
1 -1 -8 0
1 -2 -7 0
1 -1 -2 0
1 -3 -4 0
1 -5 -6 0
1 -7 -8 0
1 -9 -10 0
1 1 2 0
1 3 4 0
1 5 6 0
1 7 8 0
1 9 10 0
```

Μοντελοποιήθηκε σε CNF και παράχθηκε το εξής αρχείο εισόδου για τον επιλυτή: Οι μεταβλητές του ΠΠΠ έχουν κωδικοποιηθεί σε μεταβλητές Προτασιακής Ικανοποιησιμότητας με την χρήση του τύπου που αναφέραμε σε προηγούμενο κεφάλαιο.

(αριθμητική τιμή μεταβλητής * μέγεθος πεδίου τιμών της)+ τιμή +1.

Επομένως η πρώτη γραμμή (-3 -6) αναφέρεται στον περιορισμό μεταξύ των μεταβλητών 1 και 2 για το ζεύγος (0 1) στους περιορισμούς ((1*2)+0+1=3 και (2*2)+1+1=6)

Ο επιλυτής έδωσε την πιο κάτω ανάθεση τιμών για λύση του προβλήματος:

```
c ub 0
o 0
s OPTIMUM FOUND
v 1 -2 3 -4 5 -6 7 -8 -9 10
```

Η οποία λύση μας λέει ότι στις μεταβλητές έχει γίνει η πιο κάτω ανάθεση:

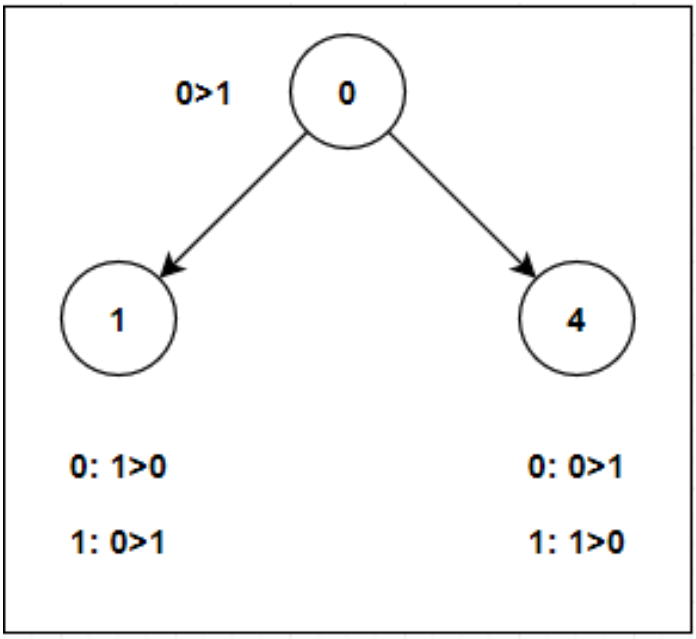
```
The var 0 has the value 0
The var 1 has the value 0
The var 2 has the value 0
The var 3 has the value 0
The var 4 has the value 1
```

Μπορούμε να συμπεράνουμε ότι η λύση ικανοποιεί όλους τους περιορισμούς που έχουν δοθεί για το πρόβλημα και είναι μια αποδεκτή λύση του προβλήματος.

Σενάριο 2 :

Στα ίδια δεδομένα του Σεναρίου 1 προσθέτουμε προτιμήσεις. Επιλέγουμε προτιμήσεις για 3 μεταβλητές, θα επιλεγούν τυχαία και θα παραχθεί το τυχαίο Δυαδικό Δέντρο Προτιμήσεων.

Το Δίκτυο Προτιμήσεων έχει την μορφή:



Σχήμα 7.1: Δίκτυο Προτιμήσεων με 3 μεταβλητές

Και με βάση αυτό έχει παραχθεί το πιο κάτω CNF αρχείο το οποίο δόθηκε στον Maxino επιλυτή.

```
p wcnf 10 26 11
11 -3 -6 0
11 -4 -5 0
11 -4 -8 0
11 -3 -8 0
11 -1 -8 0
11 -2 -7 0
11 -1 -2 0
11 -3 -4 0
11 -5 -6 0
11 -7 -8 0
11 -9 -10 0
11 1 2 0
11 3 4 0
11 5 6 0
11 7 8 0
11 9 10 0
2 -1 9 0
1 -1 10 0
2 -2 10 0
1 -2 9 0
2 -1 4 0
1 -1 3 0
2 -2 3 0
1 -2 4 0
5 1 0
1 2 0
```

Η λύση που δόθηκε από τον επιλυτή είναι η εξής και η ερμηνεία της φαίνεται πιο κάτω:

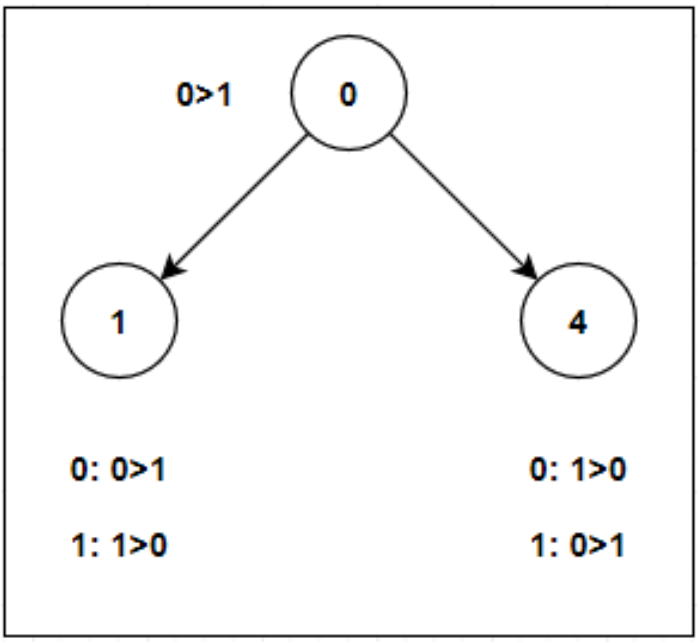
```
c ub 13
o 1
c ub 5
o 2
o 3
c ub 3
o 3
s OPTIMUM FOUND
v 1 -2 -3 4 -5 6 7 -8 9 -10
```

```
The var 0 has the value 0
The var 1 has the value 1
The var 2 has the value 1
The var 3 has the value 0
The var 4 has the value 0
```

Η ανάθεση αυτή ικανοποιεί τους περιορισμούς και με βάση την τιμή που δόθηκε στην μεταβλητή 0, η οποία είναι η προτιμότερη της τιμή, δόθηκαν και οι τιμές στις μεταβλητές 1 και 4. Οι δύο μεταβλητές πήραν τις προτιμότερες τους τιμές. Επομένως η λύση για το συγκεκριμένο σενάριο είναι ικανοποιητική και βέλτιστη.

Σενάριο3:

Με τον ίδιο ακριβώς αριθμό μεταβλητών για προτιμήσεις, τυχαία παράχθηκε ένα δυαδικό δέντρο της ίδιας μορφής με πριν αλλά με διαφορετικές προτιμήσεις. Δηλαδή οι ίδια δεντρική μορφή με τις ίδιες μεταβλητές, αλλά οι προτιμήσεις για την κάθε μεταβλητή έχουν αλλαχθεί.



Σχήμα 7.2: Δίκτυο Προτιμήσεων με 3 μεταβλητές

Το παραγόμενο CNF αρχείο για το πιο πάνω Δίκτυο Προτιμήσεων είναι το εξής:

```
p wcnf 10 26 11
11 -3 -6 0
11 -4 -5 0
11 -4 -8 0
11 -3 -8 0
11 -1 -8 0
11 -2 -7 0
11 -1 -2 0
11 -3 -4 0
11 -5 -6 0
11 -7 -8 0
11 -9 -10 0
11 1 2 0
11 3 4 0
11 5 6 0
11 7 8 0
11 9 10 0
2 -1 10 0
1 -1 9 0
2 -2 9 0
1 -2 10 0
2 -1 3 0
1 -1 4 0
2 -2 4 0
1 -2 3 0
5 1 0
1 2 0
```

Η λύση του προβλήματος με βάση τον επιλυτή και η ερμηνεία της είναι η εξής:

```
c ub 13
o 1
c ub 5
o 2
o 3
c ub 3
o 3
s OPTIMUM FOUND
v 1 -2 3 -4 5 -6 7 -8 -9 10
```

```
The var 0 has the value 0
The var 1 has the value 0
The var 2 has the value 0
The var 3 has the value 0
The var 4 has the value 1
```

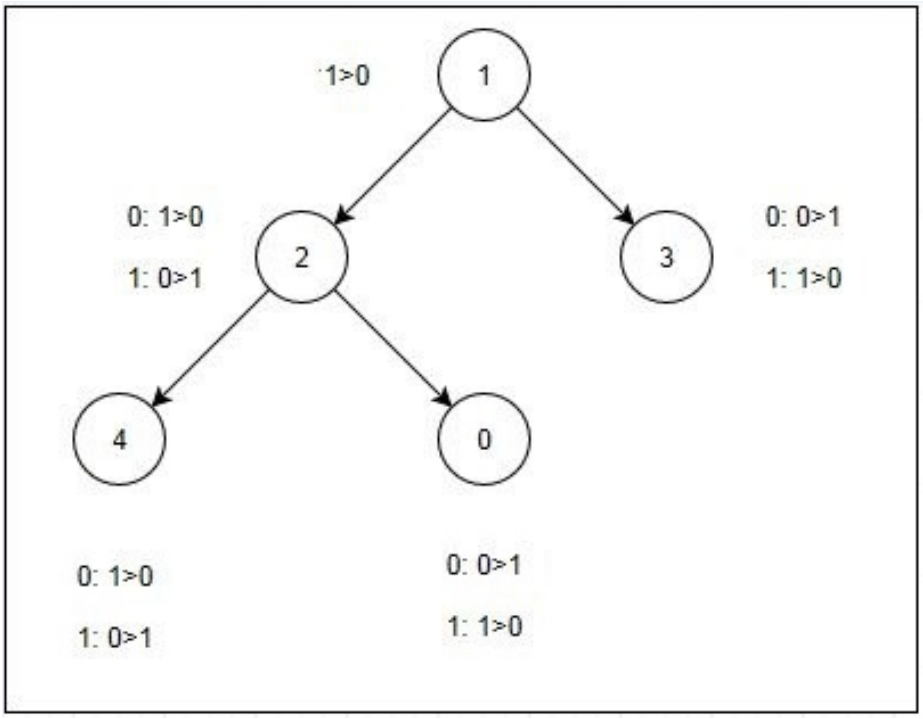
Όπως βλέπουμε το αποτέλεσμα ικανοποιεί τους περιορισμούς και διαφέρει από την προηγούμενη λύση. Επιπρόσθετα είναι και εδώ η καλύτερη δυνατή λύση για το Δίκτυο Προτιμήσεων. Βλέπουμε ότι η 0 μεταβλητή παίρνει την προτιμότερη της τιμή και στα δύο σενάρια. Ακόμα βλέπουμε πως και η τιμή της 3 παραμένει σταθερή αφού δεν επηρεάζεται, ούτε από την τιμή που θα δοθεί στην 0, ούτε από αυτή που θα δοθεί στην 1, που συμμετέχουν μαζί της στους περιορισμούς. Αντιθέτως βλέπουμε ότι η 1 επηρεάζει την 2 και έχουμε αλλαγή στις τιμές τους. Επίσης η 4 αλλάζει αφού έχουν αλλάξει και οι προτιμήσεις της στο Δίκτυο Προτιμήσεων.

Σενάριο 4:

Για να κάνουμε το πρόβλημα πιο περιοριστικό επιλέγουμε όλες τις μεταβλητές του προβλήματος να συμμετέχουν στο Δίκτυο Προτιμήσεων. Επομένως η ανάθεση θα εξαρτηθεί και από τις άλλες αναθέσεις, εκτός από τους περιορισμούς. Δηλαδή οι αναθέσεις που θα γίνονται στις μεταβλητές θα επηρεάζονται από τις αναθέσεις των μεταβλητών που είναι σε πιο πάνω επίπεδο στον Δίκτυο. Όχι από άμεση σχέση, δηλαδή να υπάρχει εξάρτηση μεταξύ τους, αλλά και από μεταβλητές που είναι ανεξάρτητες στο Δίκτυο. Αυτό μπορεί να συμβεί λόγω του ότι οι μεταβλητές μπορεί να σχετίζονται με περιορισμούς στο πρόβλημα, οι οποίοι περιορισμοί επηρεάζουν και το Δίκτυο

Αυτό μπορεί να φανεί στο συγκεκριμένο σενάριο λόγω του ότι οι μεταβλητές επηρεάζουν η μια την άλλη παρόλο που δεν συσχετίζονται στο δίκτυο άμεσα.

Το Δίκτυο Προτιμήσεων που φτιάχτηκε έχει αυτή την μορφή:



Σχήμα 7.3:Δίκτυο Προτιμήσεων για 5 μεταβλητές

Και το παραγόμενο CNF είναι το πιο κάτω:

```
p wcnf 10 34 31
31 -3 -6 0
31 -4 -5 0
31 -4 -8 0
31 -3 -8 0
31 -1 -8 0
31 -2 -7 0
31 -1 -2 0
31 -3 -4 0
31 -5 -6 0
31 -7 -8 0
31 -9 -10 0
31 1 2 0
31 3 4 0
31 5 6 0
31 7 8 0
31 9 10 0
2 -5 1 0
1 -5 2 0
2 -6 2 0
1 -6 1 0
2 -5 10 0
1 -5 9 0
2 -6 9 0
1 -6 10 0
5 -3 7 0
1 -3 8 0
5 -4 8 0
1 -4 7 0
5 -3 6 0
1 -3 5 0
5 -4 5 0
1 -4 6 0
15 4 0
1 3 0
```

Το αποτέλεσμα που δόθηκε από τον επιλυτή για το πρόβλημα είναι το εξής:

```
c ub 51
c ub 37
o 5
o 10
c ub 27
o 12
c ub 23
c ub 21
o 13
o 14
c ub 17
c ub 14
o 14
s OPTIMUM FOUND
v 1 -2 -3 4 -5 6 7 -8 9 -10
```

```
The var 0 has the value 0
The var 1 has the value 1
The var 2 has the value 1
The var 3 has the value 0
The var 4 has the value 0
```

Το αποτέλεσμα που δίνεται είναι η βέλτιστη λύση που θα μπορούσε να δοθεί στο πρόβλημα. Αφού προτιμάμε η μεταβλητή 1 να πάρει την τιμή 1 τότε το ικανοποιούμε. Στις προτιμήσεις όμως της μεταβλητής 2 για την τιμή αυτή της 1 προτιμείται η τιμή 0, αυτό όμως απαγορεύεται από τους περιορισμούς για αυτό και τελικά της δίνει την τιμή 1. Το ίδιο συμβαίνει και για τη μεταβλητή 3 η οποία προτιμά την 1, αλλά ο συνδυασμός αυτός δεν είναι εφικτός, για αυτό της δίνεται η τιμή 0. Αφού δώσουμε την τιμή 1 στην μεταβλητή 2, η 4 μπορεί να πάρει την προτιμότερη της τιμή εφόσον δεν έχει κάποιο περιορισμό που να το εμποδίζει. Όμως η μεταβλητή 0 δεν μπορεί να πάρει την προτιμότερη της τιμή για την τιμή που δώσαμε στην 2. Αυτό συμβαίνει εξαιτίας της μεταβλητής 3 και του περιορισμού που έχουν μεταξύ τους, Έτσι λοιπόν η μεταβλητή 0 παίρνει την λιγότερο προτιμητέα της τιμή, την τιμή 0.

6.2.2 Προβλήματα και Χρόνοι Επίλυσης

Έπειτα τρέξαμε προβλήματα διαφορετικού αριθμού μεταβλητών, όπως επίσης και διαφορετικών περιορισμών για το κάθε ένα. Ανάλογα με το μέγεθος των προβλημάτων επιλέξαμε να φτιάξουμε Δίκτυα Προτιμήσεων το οποία συσχετίζουν το 20% των μεταβλητών του κάθε προβλήματος.

Τα προβλήματα αυτά ήταν για 30, 60, 100 και 200 μεταβλητές, 10 προβλήματα για τη κάθε κατηγορία. Επίσης δώσαμε περιορισμούς, όπου και αυτοί ήταν ανάλογοι με το 20% του πλήθους των μεταβλητών, δηλαδή για το 30 είχαμε 6 ζευγάρια μεταβλητών σε περιορισμούς, στο 60 είχαμε 12, στο 100, 20 και για 200 είχαμε 40. Για κάθε ζευγάρι

μεταβλητών δώσαμε μόνο 2 ζευγάρια με απαγορευμένες τιμές. Όπως είπαμε και πριν για κάθε τέτοιο πρόβλημα δημιουργήσαμε και τα Δίκτυα Προτιμήσεων όπου για 30 μεταβλητές δημιουργήσαμε Δίκτυο με 6, για 60 με 12, για 100 με 20 και τέλος για 200 με 40 μεταβλητές. Για κάθε πρόβλημα πήραμε τον χρόνο που χρειάστηκε για να επιλυθεί. Για κάθε αριθμό μεταβλητών πήραμε το άθροισμα αυτών των χρόνων και βγάλαμε το μέσο όρο τους. Όπως βλέπουμε πιο κάτω:

Μεταβλητές	30(6 CPNET)	60(12 CPNET)	100(20 CPNET)	200(40 CPNET)
Πρόβλημα				
1	0.002	0.001	0.001	0.002
2	0.002	0.002	0.002	0.003
3	0.001	0.002	0.002	0.003
4	0.001	0.002	0.002	0.003
5	0.001	0.001	0.001	0.002
6	0.001	0	0.003	0.003
7	0	0.001	0.002	0.002
8	0.001	0.001	0	0.003
9	0	0	0.002	0.004
10	0.001	0	0.002	0
SUM	0.01	0.01	0.017	0.025
AVG	0.001	0.001	0.0017	0.0025

Πίνακας 7.1 :Χρόνοι Εκτέλεσης

Οι χρόνοι που υπολογίστηκαν είναι σε δευτερόλεπτα. Όπως μπορούμε να παρατηρήσουμε οι χρόνοι είναι σχετικά μικροί, αλλά ανάλογα με το μέγεθος του προβλήματος ο χρόνος αυξάνεται. Επίσης ο χρόνος επηρεάζεται και από την πολυπλοκότητα του προβλήματος, δηλαδή τους περιορισμούς που ίσως έχουν δοθεί και το Δίκτυο Προτιμήσεων του.

6.2.3 Συμπεράσματα

Από την πειραματική αξιολόγηση που έγινε συμπεράναμε ότι ανάλογα με την πολυπλοκότητα του προβλήματος, δηλαδή πρόβλημα στο οποίο οι περιορισμοί ήταν πάρα πολλοί και αρκετές από τις μεταβλητές λάμβαναν μέρος σε συσχετισμούς, καθιστούσε το πρόβλημα δύσκολο στην επίλυση. Αυτό λόγω του ότι κάποιοι

περιορισμοί με κοινές μεταβλητές αλληλοεπηρεάζονταν και έτσι η ανάθεση τιμής σε αυτή την μεταβλητή ήταν πιο δύσκολη.

Επιπρόσθετα το Δίκτυο Προτιμήσεων σε συνδυασμό με τους περιορισμούς έκαναν ακόμα πιο δύσκολο το πρόβλημα άρα και την ικανοποίηση του. Για αυτό υπήρξαν περιπτώσεις όπου διαπιστώσαμε ότι τα προβλήματα που δόθηκαν δεν είχαν λύση.

Αν όμως δεν είναι αρκετά περιοριστικό το πρόβλημα έτσι ώστε να υπάρχουν συγκρούσεις, τότε το πρόβλημα έχει σίγουρα λύση. Αυτή η λύση, δηλαδή η ανάθεση των τιμών που θα δοθεί είναι η βέλτιστη για το Δίκτυο Προτιμήσεων που παράγεται, λαμβάνοντας υπόψη και τους αντίστοιχους περιορισμούς του προβλήματος.

Για οποιαδήποτε ανάθεση τιμής σε πιο πάνω επίπεδο στο Δίκτυο Προτιμήσεων επηρεάζει την τιμή κάποιας άλλης όταν αυτές συμμετέχουν σε ένα ισχυρό περιορισμό. Με αποτέλεσμα να γίνει ανάθεση μιας λιγότερο προτιμότερης τιμής για την δεύτερη που είναι πιο αδύναμη. Με το αδύναμη εννοούμε ότι εφόσον βρίσκεται σε πιο χαμηλό επίπεδο στον Δίκτυο δεν έχει την δύναμη να επηρεάσει τις μεταβλητές που βρίσκονται πιο πάνω της αλλά μόνο τις πιο κάτω.

6.2.4 Μοντελοποίηση σε Προγραμματισμό Συνόλου Απαντήσεων

Για την μοντελοποίηση του προβλήματος σε Προγραμματισμό Συνόλου Απαντήσεων τρέξαμε διάφορα προβλήματα περιορισμών και είδαμε τα αποτελέσματα τα οποία δόθηκαν. Δηλαδή τις αναθέσεις στις μεταβλητές για κάθε τέτοιο πρόβλημα.

Πρόβλημα 1:

Είσοδος προβλήματος :

```
4 2 2 2

  0   3: (0 1) (1 0)
  1   2: (1 1) (0 1)
```

Έχουμε 4 μεταβλητές με μέγεθος πεδίου τιμών 2. Και μεταξύ των μεταβλητών έχουμε τους συγκεκριμένους περιορισμούς που φαίνονται πιο πάνω. Η μοντελοποίηση του :

```
%
va(0..1).
var(0..3).


val(0,0):-not val(0,1).
val(0,1):-not val(0,0).
val(1,0):-not val(1,1).
val(1,1):-not val(1,0).
val(2,0):-not val(2,1).
val(2,1):-not val(2,0).
val(3,0):-not val(3,1).
val(3,1):-not val(3,0).


:-val(0,0),val(0,1).
:-val(1,0),val(1,1).
:-val(2,0),val(2,1).
:-val(3,0),val(3,1).
:-val(0,0),val(3,1).
:-val(0,1),val(3,0).
:-val(1,1),val(2,1).
:-val(1,0),val(2,1).


assign(X,Y):-var(X),va(Y),val(X,Y),not assign(X,Y1),va(Y1),Y!=Y1.

#show assign/2.
```

Και η ανάθεση που δόθηκε είναι η πιο κάτω:

```
clingo version 4.5.4
Reading from ASP.lp
Solving...
Answer: 1
assign(0,0) assign(1,0) assign(2,0) assign(3,0)
Answer: 2
assign(0,0) assign(1,1) assign(2,0) assign(3,0)
Answer: 3
assign(0,1) assign(1,0) assign(2,0) assign(3,1)
Answer: 4
assign(0,1) assign(1,1) assign(2,0) assign(3,1)
SATISFIABLE

Models       : 4
Calls        : 1
Time         : 0.004s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s
```

Για το πρόβλημα που δόθηκε ο επιλυτής βρίσκει 4 πιθανές λύσεις. Σε όλες τις λύσεις οι περιορισμοί έχουν ικανοποιηθεί για αυτό και είναι αποδεκτές.

Παράδειγμα 2:

```
10 5 2 2

  0   1: (2 3) (0 3)
  1   7: (4 0) (0 3)
```

Έχουμε το πρόβλημα με 10 μεταβλητές και η κάθε μεταβλητή έχει μέγεθος πεδίου τιμών 4. Οι πιο πάνω περιορισμοί θα περιορίσουν τις αναθέσεις τιμών στις μεταβλητές αυτές.

Παίρνουμε το πιο κάτω αποτέλεσμα από τον επιλυτή clingo.

```
clingo version 4.5.4
Reading from ASP.lp
Solving...
Answer: 1
assign(2,0) assign(3,0) assign(7,0) assign(8,0) assign(1,3) assign(0,4) assign(4,4) assign(5,4) assign(6,4) assign(9,4)
Answer: 2
assign(2,0) assign(3,0) assign(7,0) assign(8,0) assign(1,3) assign(5,3) assign(0,4) assign(4,4) assign(6,4) assign(9,4)
Answer: 3
assign(2,0) assign(3,0) assign(7,0) assign(8,0) assign(5,2) assign(1,3) assign(0,4) assign(4,4) assign(6,4) assign(9,4)
Answer: 4
assign(2,0) assign(3,0) assign(5,1) assign(7,0) assign(8,0) assign(1,3) assign(0,4) assign(4,4) assign(6,4) assign(9,4)
Answer: 5
assign(2,0) assign(3,0) assign(5,0) assign(7,0) assign(8,0) assign(1,3) assign(0,4) assign(4,4) assign(6,4) assign(9,4)
Answer: 6
assign(3,0) assign(5,0) assign(7,0) assign(8,0) assign(1,3) assign(0,4) assign(2,4) assign(4,4) assign(6,4) assign(9,4)
Answer: 7
assign(2,1) assign(3,0) assign(5,0) assign(7,0) assign(8,0) assign(1,3) assign(0,4) assign(4,4) assign(6,4) assign(9,4)
SATISFIABLE

Models       : 7+
Calls        : 1
Time         : 0.014s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.010s
```

Τα αποτελέσματα πιο πάνω είναι μερικά από αυτά που μπορεί να βρεί ο επιλυτής για το πιο πάνω πρόβλημα. Εδώ φαίνονται τα 7, ωστόσο υπάρχουν πάρα πολλοί συνδυασμοί αναθέσεων οι οποίοι είναι αποδεκτοί από τους περιορισμούς. Αυτό οφείλεται στο ότι οι περιορισμοί που δόθηκαν δεν είναι πολύ περιοριστικοί και επιτρέπουν την εύρεση πολλών εναλλακτικών λύσεων

Παράδειγμα 3:

```
20 6 5 4

 10  12: (3 5) (0 4) (2 3) (1 0)
  3   5: (1 5) (1 4) (1 3) (3 5)
 12  14: (5 0) (4 0) (0 4) (3 0)
  2  17: (3 3) (4 2) (0 3) (3 5)
 12  18: (2 1) (0 0) (0 3) (1 1)
```

Στο παράδειγμα αυτό έχουμε 20 μεταβλητές με πεδίο τιμών μεγέθους 6, για κάθε ζεύγος μεταβλητών έχουμε 4 διαφορετικούς περιορισμούς τιμών.

```
clingo version 4.5.4
Reading from ASP.lp
Solving...
Answer: 1
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 2
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3) assign(0,5)
assign(1,5) assign(4,5) assign(9,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 3
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3) assign(9,4)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 4
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(9,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 5
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(9,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 6
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,1) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 7
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,1) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(19,4) assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5)
SATISFIABLE

Models       : 7+
Calls        : 1
Time         : 0.024s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.020s
```

Και στο παράδειγμα αυτό μπορούμε να διακρίνουμε τις διάφορες λύσεις οι οποίες είναι εφικτές για το παράδειγμα το οποίο δώσαμε πιο πάνω. Οι περιορισμοί επιτρέπουν την εύρεση πολλών λύσεων χωρίς να μειώσουν αισθητά το πρόβλημα.

6.2.5 Σύγκριση των δύο Μοντελοποιήσεων

Και για τις δύο μοντελοποιήσεις τρέξαμε το πιο κάτω παράδειγμα 20 μεταβλητών με μέγεθος πεδίου τιμών 6.

```
20 6 5 4

 10  12: (3 5) (0 4) (2 3) (1 0)
  3   5: (1 5) (1 4) (1 3) (3 5)
 12  14: (5 0) (4 0) (0 4) (3 0)
  2  17: (3 3) (4 2) (0 3) (3 5)
 12  18: (2 1) (0 0) (0 3) (1 1)
```

Πήραμε τα πιο κάτω αποτελέσματα από τους δύο επιλυτές. Πρώτα από τον clingo και μετά από τον Maxino

Η Clingo:

```
clingo version 4.5.4
Reading from ASP.lp
Solving...
Answer: 1
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 2
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3) assign(0,5)
assign(1,5) assign(4,5) assign(9,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 3
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3) assign(9,4)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 4
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(9,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 5
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(9,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 6
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,1) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5) assign(19,5)
Answer: 7
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(9,1) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3)
assign(19,4) assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5)
Answer: 8
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(9,2) assign(17,2) assign(2,3) assign(10,3)
assign(19,4) assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5)
Answer: 9
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(9,3) assign(10,3)
assign(19,4) assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5)
Answer: 10
assign(3,1) assign(6,0) assign(7,0) assign(8,0) assign(12,0) assign(14,0) assign(18,1) assign(5,2) assign(17,2) assign(2,3) assign(10,3) assign(9,4)
assign(19,4) assign(0,5) assign(1,5) assign(4,5) assign(11,5) assign(13,5) assign(15,5) assign(16,5)
SATISFIABLE

Models       : 10+
Calls        : 1
Time         : 0.020s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.010s
```

Maxino:

```
c ub 0
o 0
s OPTIMUM FOUND
v -1 -2 3 -4 -5 -6 -7 -8 -9 -10 11 -12 13 -14 -15 -16 -17 -18 -19 -20 -21 -22 23 -24 -25 -26 -27 -28 -29 30 -31 -32 33 -34 -35 -36 -37
-38 -39 -40 -41 42 43 -44 -45 -46 -47 -48 -49 -50 -51 -52 -53 54 -55 -56 57 -58 -59 -60 -61 -62 -63 -64 65 -66 -67 -68 -69 70 -71 -72
-73 -74 75 -76 -77 -78 -79 -80 -81 -82 -83 84 -85 -86 -87 88 -89 -90 -91 -92 -93 -94 95 -96 97 -98 -99 -100 -101 -102 -103 -104 -105
-106 107 -108 -109 -110 -111 -112 113 -114 -115 -116 -117 -118 -119 120

The var 0 has the value 2
The var 1 has the value 4
The var 2 has the value 0
The var 3 has the value 4
The var 4 has the value 5
The var 5 has the value 2
The var 6 has the value 5
The var 7 has the value 0
The var 8 has the value 5
The var 9 has the value 2
The var 10 has the value 4
The var 11 has the value 3
The var 12 has the value 2
The var 13 has the value 5
The var 14 has the value 3
The var 15 has the value 4
The var 16 has the value 0
The var 17 has the value 4
The var 18 has the value 4
The var 19 has the value 5
```

Τα δύο αποτελέσματα διαφέρουν αρκετά. Αυτό συμβαίνει λόγω του ότι ο κάθε επιλυτής χρησιμοποιεί διαφορετική μεθοδολογία για να επιλύσει το πρόβλημα που του δίνεται. Ο Maxino προσπαθεί με την ικανοποίηση περισσότερων όρων (clauses) να μεγιστοποιήσει το τελικό βάρος. Εδώ οι όροι έχουν όλοι το μέγιστο βάρος και για αυτό ικανοποιούνται δίνοντας το αποτέλεσμα που ταιριάζει. Η λογική του clingo διαφέρει, αφού βγάζει λύσεις με βάση την γνώση που προκύπτει από τους τύπους που του δίνονται. Μας δίνει όλες τις πιθανές αναθέσεις που μπορούν να δοθούν για λύση του

προβλήματος. Αυτές μπορεί να είναι πάρα πολλές. Κάποια από αυτές μπορεί να είναι και αυτή που έδωσε ο Maxino. Στο δείγμα αυτό δεν φαίνεται όμως να υπάρχει κοινή λύση.

Κεφάλαιο 7

Μελλοντικές Βελτιώσεις

7.1 Εισαγωγή

Στα προηγούμενα κεφάλαια αναλύσαμε τις διάφορες έννοιες που θα έπρεπε να είναι γνωστές για την κατανόηση των δύο μοντελοποιήσεων. Έπειτα αναλύσαμε τον τρόπο με τον οποίο μοντελοποιήθηκαν τα προβλήματα σε Προτασιακή Λογική και σε Προγραμματισμό Απαντήσεων Συνόλου.

Επεξηγήθηκε η διαδικασία για την κάθε μοντελοποίηση όπως αυτή δημιουργείται από τα συστήματα τα οποία αναπτύχθηκαν για τον σκοπό της διπλωματικής εργασίας. Όμως αυτά τα συστήματα αποδέχονται βελτιστοποιήσεις και επεκτάσεις για πιο περίπλοκα προβλήματα. Σε αυτό το κεφάλαιο λοιπόν θα εισηγηθούμε κάποιες βελτιώσεις κι επεκτάσεις και για τις δύο αυτές μοντελοποιήσεις.

7.2 Εισηγήσεις για Μελλοντικές Βελτιώσεις

Η μοντελοποίηση των Προβλημάτων Ικανοποίησης Περιορισμών σε Προτασιακή Ικανοποιησιμότητα έγινε για Δυαδικά Δέντρα ως Δίκτυα Προτιμήσεων. Αυτό μας περιορίζει όσο αφορά τις εξαρτήσεις που μπορεί να έχει μια μεταβλητή. Δηλαδή κάθε μεταβλητή μπορεί να έχει μόνο ένα πατέρα, περιορισμός που προκύπτει από την δεντρική αυτή δομή. Όπως επίσης κάθε πατέρας μπορεί να έχει μόνο δύο μεταβλητές ως εξαρτώμενες από αυτόν.

Μια επέκταση λοιπόν θα ήταν η δημιουργία γράφου όπου ένας κόμβος θα μπορεί να έχει περισσότερους από ένα πατέρα, άρα και ο ίδιος θα μπορεί να έχει πολλά παιδιά. Θα πρέπει όμως να λάβουμε υπόψη ότι απαγορεύονται οι κύκλοι στο γράφημα αυτό, λόγω του ότι θέλουμε καθαρές εξαρτήσεις μεταξύ των μεταβλητών δημιουργώντας με λίγα λόγια επίπεδα εξάρτησης. Σε κάθε επίπεδο να μπορεί να ενωθεί μια μεταβλητή μόνο με μεταβλητή του πιο πάνω επιπέδου της και όχι με προηγούμενα επίπεδα. Επίσης να μην μπορεί κάποια μεταβλητή χαμηλού επιπέδου να είναι πατέρας κάποιας σε πιο ψηλό επίπεδο. Επιπρόσθετα ένας άλλος περιορισμός είναι η ανεξαρτησία μεταξύ των μεταβλητών σε κάθε επίπεδο, όχι συσχέτιση μεταξύ τους.

Για την δημιουργία αυτού του γράφου θα πρέπει να γίνει και αλλαγή όσο αφορά το βάρος που θα δοθούν στους κόμβους. Επομένως η μεθοδολογία υπολογισμού που έχουμε εξηγήσει θα πρέπει να αλλαχθεί αναλόγως. Πιθανότατα η αλλαγή αυτή να μην είναι πολύ μεγάλη ή περίπλοκη.

Με κάποιες πολύ μικρές διαφοροποιήσεις στον υπολογισμό μπορεί να υλοποιηθεί και να γίνει πιο αποδοτικό έτσι θα δίνει και την πιο βέλτιστη λύση στις αναθέσεις.

Όσον αφορά την δεύτερη μοντελοποίηση των Προβλημάτων Ικανοποίησης Περιορισμών, σε Προγραμματισμό Συνόλου Απαντήσεων, θα πρέπει να συνεχιστεί το σύστημα που έχει υλοποιηθεί.

Η συνέχεια του συστήματος αυτού θα έχει σκοπό να μπορεί να μοντελοποιεί και προτιμήσεις από Δίκτυο Προτιμήσεων όπως και το σύστημα για μοντελοποίηση σε Προτασιακή Ικανοποιησιμότητα. Η έως τώρα λειτουργία του είναι μόνο για ανάθεση τιμών σε μεταβλητές ικανοποιώντας μόνο περιορισμούς που αφορούν απαγορεύσεις συνύπαρξης ανάθεσης συγκεκριμένων τιμών σε μεταβλητές σε μια λύση.

Βιβλιογραφία

[1] Μ. Κουμπαράκης, Διάλεξη «Προβλήματα Ικανοποίησης Περιορισμών» ,Μάθημα Τεχνητής Νοημοσύνης, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

[2] Άννα Φιλίππου, Διάλεξη «Ο Κανόνας Συμπερασμού της Ανάλυσης», Μάθημα Λογική στην Πληροφορική, Πανεπιστήμιο Κύπρου

[3] Mario Alviano,Carmine Dodaro, and Francesco Ricca.«A MaxSAT Algorithm Using Cardinality Constraints of Bounded Size»,Proceedings of the 24th International Joint Conference on Artificial Intelligence
http://www.ijcai.org/Proceedings/15/Papers/379.pdf

[4] Holger H. Hoos & Thomas Stützle, «Stochastic Local Search Foundations and Applications», « Max-Sat and Max-CSP» Chapter 7
http://www.cs.ubc.ca/~hoos/SLS-Internal/ch7.pdf

[5] Toby Walsh , « SAT v CSP» University of York, York England
http://www.cs.toronto.edu/~fbacchus/csc2512/Lectures/2012Readings/Walsh_SATvCSP.pdf

[6] Craig Boutilier,Ronen I. Brafman,Carmel Domshlak,Holger H. Hoos, David Poole «CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements, Journal of Artificial Intelligence Research 21 (2004) 135–191

[7] http://potassco.sourceforge.net/

[8] Torsten Schaub, «Answer Set Solving in Practice» University of Potsdam
http://www.cs.uni-potsdam.de/~torsten/Potassco/Slides/asp.pdf

[9] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, Sven Thiele «Using gringo, clingo and iclingo» 2008

[10] Random Uniform CSP Generators , Οκτώβρη 1996
http://www.lirmm.fr/~bessiere/generator.html

[11] http://alviano.net/software/maxino/

[12] Gerhand Brewka, James Delgrande, Javier Romero Torsten Shaub, «asprin:Customizing Answer Set Preferences without Headache»

Παράρτημα Α

Bintree.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <math.h>
#include <assert.h>
#include <sys/time.h>


typedef struct dep{
  int parent;
  int child;
  int level;
  int **prefs;
}DEP;

int clean_in();
void  malfun(int ***depps,int vals1);
void shuffle(int *array, size_t n);

int clean_in()
{
  while (getchar()!='\n');
  return 1;
}

void  malfun(int ***deps,int vals1){
  int l;
  (*deps)=NULL;
  (*deps)=(int**)malloc(sizeof(int*)*(vals1));
```

```
  for(l=0;l<vals1;l++){
    (*deps)[l]=NULL;
    (*deps)[l]=(int*) malloc(sizeof(int)*vals1);
    if((*deps)[l]==NULL){
      printf("There is not enough memory to allocate \n");
      exit(-1);
    }
  }
}

int ran(){
  struct timeval tim;
  gettimeofday(&tim, NULL);
  return (((unsigned long long) tim.tv_sec + (tim.tv_usec)) % RAND_MAX ) +0;
}

void shuffle(int *array, size_t n){
  srand(time(NULL));
  if (n > 1)
    {
      size_t i;
      for (i = 0; i < n - 1; i++)
        {
          size_t j = i +  rand() / (RAND_MAX / (n - i) + 1);
          int t = array[j];
          array[j] = array[i];
          array[i] = t;
        }
    }
}

int main(int argc,char *argv[]){
```

```
  char  vars[10],vals[10],constraints[10], nogoods[10];
  int  vars1,vals1,constraints1,nogoods1;
  FILE *fp=NULL;
  FILE *fp1=NULL;
  FILE *fp2=NULL;
  FILE *fp3=NULL;
  FILE *fp4=NULL;

  FILE *fp5=NULL;

  int i,j,k;
  int c=0;

  char temp;
  int var1,var2;
  int val1,val2;
  int count=0;

  int count1=0;
  int count2=0;
  int flag;
  int prefvar;
  int prefval;
  char ch;

  int num=0;
  int divi=0;
  int modu=0;

if(argc!=7){
   printf("Something is given wrong in the command line\nGive num constraints.txt temp.txt cnf.txt maxsat.txt solution.txt\n");
   exit(-1);
```

```
  }
  fp=fopen(argv[2]/*"constraints.txt"*/,"r");
  if(fp==NULL){
    printf("Error in opening file\n");
    exit(-1);
  }

  //diavazei tin proti grammi toy constraints.txt gia na paroyme ta dedomena mas
  //poses metablites poso pedio timon kai ta constrains apo poses diades apotelountai
  // kai posous periorismous exei kathe diada
  fscanf(fp,"%s",&vars);
  fscanf(fp,"%s",&vals);
  fscanf(fp,"%s",&constraints);
  fscanf(fp,"%s\n",&nogoods);

  //metatropi se integer
  vars1=atoi(vars);
  vals1=atoi(vals);
  constraints1=atoi(constraints);
  nogoods1=atoi(nogoods);

  //ypologismos grammon gia tin eksasfalisi oti mia timi mporei na exei i kathe
metavliti, voithitikos ipologismos
  for(i=0;i<vars1;i++){
    for(k=0;k<vals1;k++){
      for(j=1; j<vals1; j++){
          if(k!=j && k<j){
            c++;
          }
      }
    }
  }
```

```
  prefvar=atoi(argv[1]);

  while(prefvar>vars1 || prefvar<0){
    printf("You must give again the nummber of varibale,something is wrong:\n");
    printf("Please give the number of variables that must have preferences on their
domain values: (between 0-%d)",vars1);
    flag=scanf("%d",&prefvar);
    while(flag!=1 && clean_in()){
      printf("You must give again something is wrong:\n");
      flag=scanf("%d",&prefvar);
    }
  }

  int vartable[prefvar];
  int bintree[prefvar];
  int suf[vals1];
  int varss[vars1];

  for(i=0;i<vars1;i++){
    varss[i]=i;
  }

  for(i=0;i<vals1;i++){
    suf[i]=i;
  }

  shuffle(varss,vars1);
  //Epilogi metablhton gia tin dimioyrgia toy graphou
  for(i=0;i<prefvar;i++){
    vartable[i]=varss[i];
  }

  //Dimioyrgia toy diadikou dentrou
```

```
shuffle(vartable,prefvar);

for(i=0;i<prefvar;i++){
  bintree[i]=vartable[i];
}

DEP *deps=NULL;
deps=malloc(prefvar* sizeof(DEP));
if(deps==NULL){
  printf("There is not enough memory.\n");
  return 0;
}
int l;
int lev;
int p;
if(prefvar>0){
  //gia tin riza tou dentrou poy den eksartate apo kanena
  deps[0].parent=-1;
  deps[0].child=bintree[0];
  deps[0].level=0;
  deps[0].prefs=NULL;

  deps[0].prefs=(int**)malloc(sizeof(int*)*vals1);
  if(deps[0].prefs==NULL){
    printf("There is not enough memory to allocate \n");
    exit(-1);;
  }

  for(l=0;l<vals1;l++){
    deps[0].prefs[l]=NULL;
    deps[0].prefs[l]=(int*)malloc(sizeof(int));
    if(deps[0].prefs[l]==NULL){
        printf("There is not enough memory to allocate \n");
```

```
        exit(-1);
  }
}


for(i=0;i<vals1;i++){
  deps[0].prefs[i][0]=-1;
}

//Dimioyrgia protimiseon tixaia gia tin riza toy dentroy
shuffle(suf,vals1);
for(i=0;i<vals1;i++){
  deps[0].prefs[i][0]=suf[i];
}

//gia to dentro
for(j=1;j<prefvar;j++){
  if(j%2==0){
        deps[j].parent=bintree[j/2-1];
             }
  else{
        deps[j].parent=bintree[j/2];
  }
  deps[j].child=bintree[j];
  lev=log(j+1)/log(2);
  deps[j].level=lev;
  deps[j].prefs=NULL;
  malfun(&deps[j].prefs,vals1);

  int n=0;
  int v=0;
  for(n=0; n <vals1 ;n++){
        for(v=0;v< vals1;v++){
```

```
            deps[j].prefs[n][v]=-1;
          }
      }

      //protimiseis gia ton komvo gia kathe diaforetiki timi toy patera
      int randtemp;
      int x=-1;
      struct timeval tim;

      for(n=0; n <vals1; n++){
          count1=0;
          while(count1<vals1){
            x=-1;
            gettimeofday(&tim, NULL);
            randtemp=(((unsigned long long) tim.tv_sec + (tim.tv_usec)) % vals1);
            for(v=0; v<vals1;v++){
              if (randtemp==deps[j].prefs[n][v]){
                x=0;
                break;
              }
            }
            if(x==-1){
              deps[j].prefs[n][count1]=randtemp;
              count1++;
            }
          }
      }
    }

}
//dimioyrgia ton clauses toy dentrou
fp5=fopen(argv[3]/*"temp.txt"*/,"w");
if(fp5==NULL){
```

```
   printf("Error in opening file\n");
   exit(-1);
  }

  long long int weight=2;
  int pos=0;
  int thesi=prefvar-1;
  int nodes=prefvar;
  long long int fatherw=0;
  int grammes=0;
  lev=log(nodes)/log(2);

  //dimioyrgia ton clauses mazi me ton ipologismo tou baroys gia kathe komvo kai epipedo
  while(pos<prefvar-1){
   while(deps[thesi].level==lev){
    for(j=0;j<vals1;j++){
         for(l=0;l<vals1;l++){
           if(l==0){
           fprintf(fp5,"%lli -%d %d 0\n",weight,(deps[thesi].parent*vals1)+(j+1),((deps[thesi].child*vals1)+(deps[thesi].prefs[j][l]))+1);
           }
           else{
 fprintf(fp5,"%lli -%d %d 0\n",1,(deps[thesi].parent*vals1)+(j+1),((deps[thesi].child*vals1)+(deps[thesi].prefs[j][l]))+1);
           }
           grammes++;
         }
    }
    fatherw+=weight;
    thesi--;
```

```
      pos++;
    }
    nodes=prefvar-pos;
    lev=log(nodes)/log(2);
    // printf("varos %lli gia ton patera %lli\n",weight,fatherw);
    weight=fatherw+1;
    // printf("%lli\n",weight);
  }

  if(prefvar>0){
    for(i=0;i<vals1;i++){
      if(i==0){
          fprintf(fp5,"%lli %d 0\n",weight,(deps[0].child*vals1)+(deps[0].prefs[i][0])+1);
}
      else{
          fprintf(fp5,"%lli %d 0\n",1,(deps[0].child*vals1)+(deps[0].prefs[i][0])+1);
}
      grammes++;

    }
  }
  else{
    weight=0;
    fprintf(fp5,"\n");
  }
  fclose(fp5);

  fp1=fopen(argv[4]/*"cnf.txt"*/,"w");
  if(fp1==NULL){
    printf("Error in opening file\n");
    exit(-1);
  }
  //Dimioyrgia CNF arxeiou gia eisodo ston solver
```

```
 fprintf(fp1,"p wcnf %d %d %lli\n",vars1*vals1,(((constraints1*nogoods1)+c+vars1)+grammes),(2*weight)+1);

 //Periorismoi pou dinontai apo tin arxi
 while(!feof(fp)){
  fscanf(fp,"%3d %3d: ",&var1,&var2);

  while(count<nogoods1){
   fscanf(fp,"(%d %d) ",&val1,&val2);
   fprintf(fp1,"%lli -%d -%d 0\n",(2*weight)+1,((var1*vals1)+val1)+1,((var2*vals1)+val2)+1);
   count++;
  }
  fscanf(fp,"\n",&temp);
  count=0;
 }
 //periorismoi logo pediou timon anagkasmos min siniparksis dio timon tis metavlitis
 for(i=0;i<vars1;i++){
  for(k=0;k<vals1;k++){
   for(j=1; j<vals1; j++){
       if(k!=j && k<j){
         fprintf(fp1,"%lli -%d -%d 0\n",(2*weight)+1,((i*vals1)+k)+1,((i*vals1)+j)+1);
       }
   }
  }
 }

 //periorismoi logo pediou timon anagkazoyn na dothei mia timi mono
 for(i=0;i<vars1;i++){
  fprintf(fp1,"%lli ",(2*weight)+1);
  for(j=0; j<vals1; j++){
   fprintf(fp1,"%d ",((i*vals1)+j)+1);
  }
```

```
  fprintf(fp1,"0\n");
}

fp5=fopen(argv[3]/*"temp.txt"*/,"r");
if(fp5==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

char apotmp;
while(!feof(fp5)){
  fscanf(fp5,"%c",&apotmp);
  fprintf(fp1,"%c",apotmp);
}

fclose(fp);
fclose(fp1);
fclose(fp5);

//Epilitis maxino  kalesma me to CNF arxeio
char name[200];
sprintf(name,"time ./maxino-2015-k16-static  <%s >%s",argv[4],argv[5]);

clock_t begin, end;
double time_spent;
begin=clock();
printf("begin time: %f\n",begin);
system(name);
end=clock();
printf("end time: %f\n",end);
time_spent=((double)(end-begin))/CLOCKS_PER_SEC;
printf("Time is:%f\n",time_spent);
```

```
//eksagogi apotelesmaton apo to arxeio poy epistrefei o maxino
fp3=fopen(argv[5]/*"maxsat.txt"*/,"r");
if(fp3==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

fp2=fopen(argv[5]/*"maxsat.txt"*/,"r");
if(fp2==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

fp4=fopen(argv[6]/*"solution.txt"*/,"w");
if(fp4==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

while(!feof(fp3)&& !feof(fp2)){
  fscanf(fp3,"%c",&ch);
  while(ch=='c'|| ch=='s'|| ch=='o'){
    while(ch!='\n'){
         fscanf(fp3,"%c",&ch);
    }
  }
  fp2=fp3;
  while(ch=='v'){
    fscanf(fp3,"%c",&ch);
    while(ch!='\n'){
         fscanf(fp2,"%d",&num);
         if(num>0){
```

```
            divi=num/vals1;
            modu=num%vals1;
            if(modu==0){
              divi--;
              modu=vals1;
            }
            fprintf(fp4,"The var %d has the value %d\n",divi,modu-1);
          }
          fscanf(fp3,"%c",&ch);
      }
    }
  }
  fclose(fp4);
  fclose(fp2);


  return 0;

}
```

Παράρτημα Β

APS.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <math.h>
#include <assert.h>
#include <sys/time.h>

int clean_in()
{
  while (getchar()!='\n');
  return 1;
}

int main(int argc,char *argv[]){

  char  vars[10],vals[10],constraints[10], nogoods[10];
  int  vars1,vals1,constraints1,nogoods1;
  FILE *fp=NULL;
  FILE *fp1=NULL;
  FILE *fp2=NULL;
  FILE *fp3=NULL;
  FILE *fp4=NULL;

  FILE *fp5=NULL;

  int i,j,k;

  char temp;
```

```
int var1,var2;
int val1,val2;
int count=0;

int count1=0;
int count2=0;
int flag;
int prefvar;
int prefval;
char ch[100];

int num=0;
int divi=0;
int modu=0;

fp=fopen("constraints.txt","r");
if(fp==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

//diavazei tin proti grammi toy constraints.txt gia na paroyme ta dedomena mas
//poses metablites poso pedio timon kai ta constrains apo poses diades apotelountai
// kai posous periorismous exei kathe diada
fscanf(fp,"%s",&vars);
fscanf(fp,"%s",&vals);
fscanf(fp,"%s",&constraints);
fscanf(fp,"%s\n",&nogoods);

//metatropi se integer
vars1=atoi(vars);
vals1=atoi(vals);
constraints1=atoi(constraints);
```

```
nogoods1=atoi(nogoods);

fp1=fopen("ASP.lp","w");
if(fp1==NULL){
  printf("Error in opening file\n");
  exit(-1);
}

//Diloseis var(0..n) kai va(0..n)
fprintf(fp1,"%\n");
fprintf(fp1,"va(0..%d).\n",vals1-1);
fprintf(fp1,"var(0..%d).\n",vars1-1);
fprintf(fp1,"\n\n");
int fl=0;
int me=0;

//gia toys periorismou tou pediou timon
 for(i=0;i<vars1;i++){
  for(k=0;k<vals1;k++){
    me=0;
    fprintf(fp1,"val(%d,%d):-",i,k);
    for(j=0;j<vals1; j++){
          fl=0;

          if(k!=j){
            fprintf(fp1,"not val(%d,%d)",i,j);
            fl=1;
            me++;
          }
          if(fl==1){
            if(me<vals1-1){
              fprintf(fp1,",");
              fl==0;
```

```
            }
           }
      }
      fprintf(fp1,".\n");
    }
   }

   fprintf(fp1,"\n\n");
for(i=0;i<vars1;i++){
    fprintf(fp1,":-");
    for(j=0; j<vals1; j++){
      if(j!=vals1-1){
      fprintf(fp1,"val(%d,%d),",i,j);
      }
      else{
      fprintf(fp1,"val(%d,%d)",i,j);
      }
    }
    fprintf(fp1,".\n");
  }

//Periorismous toy problimatos dothikan sto arxeio
 while(!feof(fp)){
    fscanf(fp,"%3d %3d: ",&var1,&var2);

    while(count<nogoods1){
      fscanf(fp,"(%d %d) ",&val1,&val2);
      fprintf(fp1,":-val(%d,%d),val(%d,%d).\n",var1,val1,var2,val2);
      count++;
    }
    fscanf(fp,"\n",&temp);
    count=0;
  }
```

```
 fprintf(fp1,"\n\n");

 //typos gia tin anathesi timis
  fprintf(fp1,"assign(X,Y):-var(X),va(Y),val(X,Y),not assign(X,Y1),va(Y1),Y!=Y1.");
 fprintf(fp1,"\n\n");

 fprintf(fp1,"#show assign/2.");

 fclose(fp);
 fclose(fp1);

 clock_t begin, end;
 double time_spent;

 begin=clock();
system("./clingo ASP.lp >>clingosol.txt 10");
 end=clock();

 time_spent=(double)(end-begin);
 printf("Time is %d\n",time_spent);

 return 0;

}
```