Ατομική Διπλωματική Εργασία

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΚΑΙ ΙΚΑΝΟΠΟΙΗΣΗ ΠΕΡΙΟΡΙΣΜΩΝ ΣΤΗΝ ΕΠΙΧΕΙΡΗΜΑΤΟΛΟΓΙΑ

Ιωάννης Γιάγκου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2019

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Προγραμματισμός και Ικανοποίηση Περιορισμών στην Επιχειρηματολογία

Ιωάννης Γιάγκου

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 2019

Ευχαριστίες

Η ολοκλήρωση της Ατομικής Διπλωματικής Εργασίας αποτελεί έναν από τους σημαντικότερους στόχους ενός προπτυχιακού φοιτητή. Γι’ αυτό νιώθω την υποχρέωση να εκφράσω τις ένθερμες μου ευχαριστίες προς τα άτομα εκείνα τα οποία, όπως πιστεύω, διαδραμάτισαν σημαντικό ρόλο στην επίτευξη αυτού του προσωπικού μου στόχου.

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον κ. Γιάννη Δημόπουλο, ο οποίος μέσω του ρόλου του ως επιβλέποντας καθηγητής της διπλωματικής μου εργασίας, με καθοδηγούσε και με συμβούλευε σε κάθε σημείο της διαδικασίας αυτής, στηρίζοντας με σε κάθε δυσκολία.

Επίσης, θερμές ευχαριστίες εκφράζω στο σύνολο του ακαδημαϊκού προσωπικού του Πανεπιστημίου Κύπρου για όλα τα απαραίτητα εφόδια που μου παρείχαν καθ’ όλη τη διάρκεια των σπουδών μου στο τμήμα Πληροφορικής. Οι γνώσεις που απέκτησα κατέστησαν δυνατή τη διεκπεραίωση της διπλωματικής μου εργασίας, ενώ παράλληλα θα με συνοδεύουν στην μετέπειτα επαγγελματική μου πορεία.

Τέλος, επιθυμώ να πω ένα τεράστιο «ευχαριστώ» στα άτομα τα οποία, παρόλο που δεν συνδέονται άμεσα με τον κλάδο μας, η ηθική και συναισθηματική τους υποστήριξη αποτέλεσε τον σημαντικότερο παράγοντα στην ολοκλήρωση του κύκλου σπουδών μου. Αυτά τα άτομα δεν είναι άλλα από τους φίλους μου και την οικογένειά μου. Μέσω της ανιδιοτελούς βοήθειάς τους ήρθα αντιμέτωπος με μια μεγάλη αλήθεια: αρκετά από τα απαραίτητα εφόδια για να διαπρέψουμε σε μία κοινωνία, δεν είναι υλικά, μετρήσιμα, ή ορατά.

Περίληψη

Η «Επιχειρηματολογία» αποτελεί αντικείμενο μελέτης αρκετών κλάδων της Πληροφορικής όπως η Τεχνητή Νοημοσύνη, η Θεωρία Υπολογισμού, ο Λογικός Προγραμματισμός, η Βιοπληροφορική κλπ. Ορίζεται ως η διαδικασία κατά την οποία εκδηλώνεται υποστήριξη ή απαξίωση για μία συγκεκριμένη θέση που αφορά κάποιο θέμα, μέσω της παράθεσης των λόγων εκείνων που οδηγούν στην στάση αυτή, και τη δημιουργία συσχετίσεων σύγκρουσης ή υποστήριξης, μεταξύ τους. Το διαχρονικό αυτό θέμα έχει μελετηθεί σε μεγάλο βαθμό, καθώς μπορεί να αναπαραστήσει μία πληθώρα πραγματικών προβλημάτων. Καθημερινά παραδείγματα επιχειρηματολογίας αποτελούν η συνέλευση του κοινοβουλίου και του δικαστηρίου, οι πολιτικές αντιπαραθέσεις (debates) και γενικά οποιαδήποτε πράξη ανταλλαγής απόψεων και στοιχείων, που σκοπό έχουν να ενισχύσουν το κύρος μίας θέσης, έναντι άλλων.

Στα πλαίσια αυτής της εργασίας θα ασχοληθούμε με το αφαιρετικό κομμάτι της Επιχειρηματολογίας, δίνοντας έμφαση στα συστατικά που την αποτελούν, δηλαδή τα επιχειρήματα και τις μεταξύ τους συσχετίσεις. Θα μελετηθούν προβλήματα Ικανοποίησης Περιορισμών στην Επιχειρηματολογία, μέσω της μοντελοποίησης τους, τόσο στην Προτασιακή Λογική, όσο και στο Γραμμικό Προγραμματισμό.

Μεγαλύτερη σημασία θα δοθεί στο πρόβλημα της Ευσταθούς Επέκτασης, κατά το οποίο θα βρίσκουμε ευσταθείς ομάδες επιχειρημάτων που θα καταρρίπτουν αντίπαλα επιχειρήματα. Επίσης θα μελετηθεί η Μη-Ικανοποιησιμότητα στα προβλήματα Ευσταθούς Επέκτασης και διάφοροι τρόποι με τους οποίους μπορεί να αντιμετωπιστεί, ώστε να επιτευχθεί η εξεύρεση της καλύτερης δυνατής λύσης μέσω συμβιβασμών, είτε αυτό απαιτεί αφαίρεση στοιχείων από το πρόβλημα (όπως επιχειρήματα ή συσχετίσεις), είτε ρύθμιση των τιμών ανάθεσης.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο Διπλωματικής Εργασίας

Η «Επιχειρηματολογία» αποτελεί τη διαδικασία κατά την οποία εκδηλώνεται υποστήριξη ή απαξίωση για μία συγκεκριμένη θέση που αφορά κάποιο θέμα, μέσω της παράθεσης των λόγων εκείνων που οδηγούν στην στάση αυτή και τη δημιουργία συσχετίσεων σύγκρουσης ή υποστήριξης, μεταξύ τους. Οι λόγοι αυτοί ονομάζονται «επιχειρήματα» και σκοπός τους είναι να πείσουν για την ορθότητα των απόψεων του εκφραστή τους. Η αιτιολόγηση της ορθότητας της υποστηριζόμενης θέσης, καθώς και η προσπάθεια μείωσης του κύρους μίας αντίπαλης θέσης, είναι οι συχνότερες και αποτελεσματικότερες κατηγορίες επιχειρημάτων. Αμφότερες οι προσεγγίσεις (η μεν άμεσα, και η δεν έμμεσα) προσπαθούν να επιτύχουν την επικράτηση μίας θέσης, έναντι μίας πληθώρας άλλων θέσεων που την ανταγωνίζονται.

Όταν η γραμμή μεταξύ σωστού και λανθασμένου τρόπου προσέγγισης ενός προβλήματος είναι αρκετά θολή (λόγω ασυνεπών ή/και ατελών πληροφοριών) η τελική απόφαση τείνει να εναπόκειται, ή έστω να βασίζεται κατά κόρον, στην λογικότερη εκ των προσεγγίσεων. Η έκφραση επιχειρημάτων και η επικράτηση των πειστικότερων εξ' αυτών, αποτελεί τον απλούστερο τρόπο εξεύρεση μίας ικανοποιητικής και (κατά πλειοψηφία) αποδεκτής λύσης σε αυτού του είδους προβλήματα. Με λίγα λόγια, η Επιχειρηματολογία προσπαθεί να εξαλείψει αδιέξοδα που έχουν να κάνουν με προβλήματα στα οποία δεν υπάρχουν ξεκάθαρα «ορθές» ή «λανθασμένες» προσεγγίσεις.

Η ορθότητα της Επιχειρηματολογίας βασίζεται στο ότι η λύση σε προβλήματα που ορίζονται στο χώρο της, είναι πλήρως τεκμηριωμένη. Αυτό προκύπτει από το γεγονός ότι αρκετά ισχυρές, και συνεκτικές μεταξύ τους, γνώμες, ακατάρριπτες από αντίθετες απόψεις, ομαδοποιούνται και καταφέρνουν να ενισχύσουν το κύρος της κοινής θέσης που υποστηρίζουν.

Η σημαντικότητα της Επιχειρηματολογίας μαρτυρείται από την εφαρμογή της σε διάφορους κλάδους όπως η νομική, η πολιτική, η βιολογία, η πληροφορική, η λογική, η φιλοσοφία και γενικότερα οπουδήποτε υφίστανται προβλήματα, τα οποία δεν είναι καθορισμένα σε ξεκάθαρα πλαίσια και απαιτούν την παράθεση επιχειρημάτων από ειδικούς, ώστε να ληφθεί μία απόφαση.

Στην εργασία αυτή θα μελετηθούν διάφορες έννοιες, εφαρμογές, προβλήματα και προσεγγίσεις που σχετίζονται με την Επιχειρηματολογία, με κυριότερο σημείο αναφοράς να αποτελεί το πρόβλημα της «Ευσταθούς Επέκτασης».

Το πρόβλημα της «Ευσταθούς Επέκτασης» αποτελεί ένα αρκετά σημαντικό είδος Προβλημάτων Ικανοποίησης Περιορισμών, ορισμένο στο χώρο της Επιχειρηματολογίας, μέσω του οποίου εξάγουμε τα επιχειρήματα των οποίων η θέση τους αποτελεί την επικρατέστερη του συνόλου των θέσεων. Θα δούμε την χρήση σχετικών εργαλείων, όπως ο Προτασιακός Λογισμός, ο Γραμμικός Προγραμματισμός και οι Επιλυτές Προβλημάτων Ικανοποιησιμότητας, για τον ορισμό, τη μοντελοποίηση και την επίλυση του συγκεκριμένου προβλήματος, ως Πρόβλημα Ικανοποίησης Περιορισμών. Ακόμη, θα μελετηθούν και κάποια υπό-προβλήματα που σχετίζονται με την Ικανοποιησιμότητα προβλημάτων «Ευσταθούς Επέκτασης», και τους τρόπους με τους οποίους χειριζόμαστε περιπτώσεις που αποτελούν «Μη-Ικανοποιήσιμες», με σκοπό την εύρεση συμβιβαστικών λύσεων.

1.2 Σκοπός Διπλωματικής Εργασίας

Σκοπός της παρούσας διπλωματικής εργασίας είναι η εκτενής μελέτη προβλημάτων που ορίζονται στο χώρο της Επιχειρηματολογίας, με κυριότερη έμφαση στην προσπάθεια εξεύρεσης συνόλων Ευσταθούς Επέκτασης, και την μοντελοποίηση τους μέσω διαφόρων τεχνικών Προγραμματισμού Ικανοποίησης Περιορισμών.

Πιο συγκεκριμένα, παρουσιάζεται το πρόβλημα της εξεύρεσης Ευσταθούς Επέκτασης ως πρόβλημα Προτασιακής Ικανοποιησιμότητας (Propositional Satisfiability) μέσω της χρήσης του SAT επιλυτή Lingeling, ενώ παράλληλα θα μελετήσουμε το ίδιο πρόβλημα, μοντελοποιημένο ως πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού (Integer Linear Programming), μέσω της χρήσης του επιλυτή intlinprog που προσφέρει η Matlab.

Επιπλέον, γίνεται αναφορά στους τρόπους μέσω των οποίων μπορούμε να χειριστούμε περιπτώσεις Μη-Ικανοποιήσιμων προβλημάτων Ευσταθούς Επέκτασης. Σημαντικότεροι από αυτούς αποτελούν η χρήση της Μέγιστης Προτασιακής Ικανοποιησιμότητας με Βάρη (Weighted, MAXSAT problems) μέσω της χρήσης του MAXSAT επιλυτή Maxino, και ο Μεικτός-Ακέραιος Γραμμικός Προγραμματισμός (Mixed-Integer Linear Programming, MILP) μέσω της χρήσης του επιλυτή intlinprog που προσφέρει η Matlab. Τέλος, θα μελετήσουμε το πώς διάφορες βελτιστοποιήσεις των λύσεων που ζητούνται από τους επιλυτές, προσφέρουν διαφορετικά αποτελέσματα. Η παρουσίαση αποτελεσμάτων και επεξήγησή τους, προς το χρήστη, γίνεται μέσω ενός προγράμματος στη γλώσσα προγραμματισμού JAVA, το οποίο αναπτύχθηκε ειδικά για τους σκοπούς της εργασίας αυτής.

1.3 Δομή Διπλωματικής Εργασίας

Στο Κεφάλαιο 1 παρουσιάσαμε, σε γενικά πλαίσια, το θέμα της εργασίας, τους σκοπούς στους οποίους εστιάζει καθώς επίσης και τα μέσα (εργαλεία, τεχνικές, προγράμματα) που θα βοηθήσουν στην επίτευξή τους.

Στο Κεφάλαιο 2 γίνεται αναφορά στο Θεωρητικό Υπόβαθρο που απαιτείται για να κατανοήσουμε πλήρως τις σημαντικότερες έννοιες της Επιχειρηματολογίας, και τις ιδιαιτερότητες που διέπουν το χώρο της (ορισμοί, γραφική αναπαράσταση, απαραίτητες δομές δεδομένων, εργαλεία κ.λπ.). Ταυτόχρονα, ορίζεται το πρόβλημα «Ευσταθούς

Επέκτασης», το οποίο αποτελεί το κυριότερο σημείο ενδιαφέροντος της εν λόγω μελέτης.

Στο Κεφάλαιο 3 παρουσιάζεται ο τρόπος με τον οποίο ο Προτασιακός Λογισμός χρησιμοποιείται για τον ορισμό, την μελέτη και την επίλυση προβλημάτων «Ευσταθούς Επέκτασης», μοντελοποιημένων ως Προβλήματα Ικανοποίησης Περιορισμών. Ο τρόπος χειρισμού των Μη-Ικανοποιήσιμων περιπτώσεων, βάσει του Προτασιακού Λογισμού, περιγράφεται στο Κεφάλαιο 4.

Στο Κεφάλαιο 5 παρουσιάζεται ο τρόπος με τον οποίο ο Γραμμικός Προγραμματισμός χρησιμοποιείται για τον ορισμό, την μελέτη και την επίλυση προβλημάτων «Ευσταθούς Επέκτασης», μοντελοποιημένων ως Προβλήματα Ικανοποίησης Περιορισμών. Ο τρόπος χειρισμού των Μη-Ικανοποιήσιμων περιπτώσεων, βάσει του Γραμμικού Προγραμματισμού, περιγράφεται στο Κεφάλαιο 6.

Και στα 4 προαναφερθέντα κεφάλαια (3, 4, 5, 6) γίνεται εκτενής αναφορά σε όλα τα εργαλεία, τεχνικές γνώσεις και διεργασίες που απαιτούνται.

Στο Κεφάλαιο 7 παρατίθενται παραδείγματα, μέσω πειραμάτων, για πλήρη κατανόηση των όσων παρουσιάζονται στα προηγούμενα κεφάλαια. Η εργασία ολοκληρώνεται στο Κεφάλαιο 8, με την παρουσίαση συμπερασμάτων, καθώς επίσης και μερικών σκέψεων για μελλοντικές επεκτάσεις της.

Με το τέλος της εργασίας, παρατίθεται η βιβλιογραφία, με όλες τις απαραίτητες πηγές που τεκμηριώνουν τα συμφραζόμενα, όπως και παραρτήματα που περιέχουν τον κώδικα του προγράμματος που αναπτύχθηκε, όπως και άλλες απαραίτητες πληροφορίες.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο και Απαραίτητα Εργαλεία

2.1 Θεωρία

2.1.1 Χρήσιμοι Ορισμοί [9, 11]

Επιχείρημα: Στο πεδίο της Λογικής, ένα επιχείρημα (argument) είναι ένας συλλογισμός ή μία δήλωση, βασισμένη στη φυσική γλώσσα, η οποία υποστηρίζει κάποιο συμπέρασμα. Προκύπτει από την πραγματικότητα και αποτελεί οτιδήποτε θα μπορούσε να στηρίξει μία συγκεκριμένη θέση, όπως π.χ. αποδεικτικό στοιχείο (αποτελέσματα ερευνών, τεκμήρια, μαρτυρίες κ.λπ.), εδραιωμένο και αποδεκτό γεγονός (νόμοι της φύσης κ.λπ.) , ή ακόμη και προσωπική άποψη (το τελευταίο δεν φέρει τον ίδιο βαθμό πειστικότητας, όσο οι άλλες κατηγορίες).

Ουδέτερο Επιχείρημα (neutral argument): Επιχείρημα το οποίο δεν έρχεται σε αντιπαράθεση με άλλα επιχειρήματα.

Επιχειρηματολογία: Στο πεδίο της Λογικής, η επιχειρηματολογία (argumentation) είναι ο μηχανισμός μέσα στον οποίο ενεργούν τα επιχειρήματα. Εντός του χώρου της, επιχειρήματα ομαδοποιούνται ή αλληλοσυγκρούονται, βάσει συμφερόντων, με σκοπό την επικράτηση κάποιας θέσης. Αποτελεί γέννημα της τάσης μας ως άνθρωποι να προσπαθούμε να δικαιολογήσουμε την επιλογή μας, προκειμένου να αποδείξουμε ότι είναι ορθή και λογική.

Μπορούμε να φανταστούμε την επιχειρηματολογία ως ένα φυσικό χώρο στον οποίο διεξάγεται ένας διάλογος αντιπαραθέσεων (debate). Σε αυτό το χώρο ακούγονται απόψεις (επιχειρήματα) από διάφορους παράγοντες, εκ των οποίων οι δυνατότερες (συνήθως οι πειστικότερες) συμβάλλουν δραματικά στην επικράτηση της θέσης που ασπάζονται, έναντι άλλων. Ένα από τα τυπικότερα παραδείγματα είναι η συνέλευση του κοινοβουλίου, όπου φυσικές οντότητες (βουλευτές) παραθέτουν απόψεις, στοιχεία και γεγονότα, προκειμένου να ψηφιστεί η καλύτερη δυνατή αντιμετώπιση ενός προβλήματος.

2.1.2 Ευσταθής Επέκταση [10]

Ευσταθής Επέκταση: Η σημαντικότερη έννοια, όσον αφορά αυτήν την εργασία. Η «Ευσταθής Επέκταση» (Stable Extension) αποτελεί ένα υποσύνολο επιχειρημάτων, τα οποία εξυπηρετούν κάποιο κοινό στόχο. Τα μέλη του συνόλου αυτού χαρακτηρίζονται από την μεταξύ τους αλληλοϋποστήριξη και την αντεπίθεση τους, ως ομάδα, σε κάθε επιχείρημα που εναντιώνεται στο σύνολο τους.

Πιο συγκεκριμένα, το σύνολο της Ευσταθούς Επέκτασης διέπεται από τους ακόλουθους περιορισμούς:

1) Κανένα από τα επιχειρήματα του συνόλου δεν επιτίθεται σε άλλο επιχείρημα του συνόλου, ούτε στον εαυτό του: Αποδεκτά επιχειρήματα δεν φάσκουν και

αντιφάσκουν. Εάν κάποιο επιχείρημα «αναιρεί» τον ίδιο του τον εαυτό (αντίφαση) ή εναντιώνεται σε επιχειρήματα που στηρίζουν την ίδια θέση, τότε είναι αναξιόπιστο, καθώς έμμεσα αντιτίθεται στον ίδιο σκοπό που προσπαθεί να υποστηρίξει.

2) Κάθε επιχείρημα που δεν ανήκει στο σύνολο, δέχεται τουλάχιστον 1 επίθεση από κάποιο επιχείρημα που βρίσκεται εντός του συνόλου: Βάσει των προϋποθέσεων, επιχειρήματα που δεν ανήκουν στην Ευσταθή Επέκταση, αντιτίθενται στα «μέλη» της. Επομένως, για να θεωρείται έγκυρο κάποιο ευσταθές σύνολο, πρέπει να μην είναι εκτεθειμένο σε επιθέσεις που θα αναιρούσαν την ισχύ του. Έτσι, καμία επίθεση εναντίον της Ευσταθούς Επέκτασης δεν πρέπει να μένει αναπάντητη, καθώς κάθε αντίπαλο επιχείρημα πρέπει να δέχεται «αντεπίθεση» από κάποιο μέλος της, με σκοπό τη διατήρηση της ακεραιότητάς του συνόλου.

3) Δεν υπάρχει λύση που να αποτελεί υποσύνολο μίας άλλης λύσης: Κάθε επιχείρημα εκτός της Ευσταθούς Επέκτασης αποτελεί «αντίπαλο» της. Γι' αυτό δεν δύναται η συνύπαρξη δύο λύσεων, εκ των οποίων η μία να καθιστά κάποιο επιχείρημα ως μέλος, ενώ η άλλη να καθιστά το ίδιο επιχείρημα ως μη-μέλος (άρα και «αντίπαλο»).

π.χ. ΔΕΝ δύναται να συνυπάρχουν οι λύσεις s_1 και s_2, για τις οποίες να ισχύει ότι:

$$s_1= \{a_1, a_2, a_5\},\ s_2= \{a_1, a_2, a_5, a_8\} \rightarrow s_1 \subsetneq s_2$$

Αυτό θα καθιστούσε το επιχείρημα a_8 ως μη-μελός (βάσει s_1) ΚΑΙ ως μέλος (βάσει s_2) της Ευσταθούς Επέκτασης, ταυτόχρονα, κάτι που αποτελεί αντίφαση.

Το πρόβλημα της εξεύρεσης Ευσταθούς Επέκτασης κατατάσσεται στα Προβλήματα Ικανοποίησης Περιορισμών, και πιο συγκεκριμένα, στο είδος «Προβλημάτων Ικανοποιησιμότητας» (SAT problems).

Η σημαντικότητα της «Ευσταθούς Επέκτασης» έγκειται στο γεγονός ότι αποτελεί το «τεκμήριο» της ορθότητας μίας θέσης που επικρατεί έναντι άλλων. Η επικρατέστερη θέση στο χώρο, είναι αυτή την οποία στηρίζουν τα επιχειρήματα-μέλη της.

2.1.3 Προβλήματα Ικανοποίησης Περιορισμών [4]

Προβλήματα Ικανοποίησης Περιορισμών (Π.Ι.Π.): Είδος προβλημάτων με το οποίο ασχολείται ο κλάδος της Πληροφορικής, και πιο συγκεκριμένα, ο τομέας της Τεχνητής Νοημοσύνης. Τα προβλήματα αυτά ορίζονται βάσει ενός πλήθους μεταβλητών, ενός πλήθους περιορισμών (που συνήθως εκφράζονται με γραμμικές ανισότητες/ εξισώσεις ή Προτασιακή Λογική) και ενός πεδίου ορισμού (ένα για κάθε μεταβλητή). Λύση στο πρόβλημα αποτελεί η ανάθεση τιμών (οι οποίες απορρέουν από το πεδίο ορισμού) στις μεταβλητές, ικανοποιώντας ταυτόχρονα, όλο το πλήθος των περιορισμών. Μία τέτοια ανάθεση ονομάζεται «μοντέλο λύσης». Αδυναμία εύρεσης ενός μοντέλου λύσης καθιστά το πρόβλημα «Μη- Ικανοποιήσιμο». Είναι της τάξης των NP-Hard προβλημάτων, καθώς αύξηση των μεταβλητών εισόδου προκαλεί εκθετική αύξηση στον χρόνο επίλυσης του προβλήματος. Για μεγάλης κλίμακας Π.Ι.Π., η επίλυση τους από τον άνθρωπο καθίσταται χρονοβόρα, έως και αδύνατη, και αποτελούν περιπτώσεις στις οποίες αναγκαστικά χρησιμοποιούνται ειδικά προγράμματα για την εξεύρεση μίας λύσης, τα οποία ονομάζονται «επιλυτές».

Βάση Γνώσης: Το «τι χαρακτηριστικά γνωρίζουμε» για κάποιο πρόβλημα, δοσμένα από τους περιορισμούς που το διέπουν. Μία «Βάση Γνώσης» για κάποιο πρόβλημα, ουσιαστικά, είναι όλοι οι περιορισμοί που το διέπουν, συγκεντρωμένοι σε ένα αρχείο ή μία βάση δεδομένων.

Στοιχεία Ευσταθούς Επέκτασης ως Π.Ι.Π.:

1) Μεταβλητές (Problem Variables/ Literals): Τα επιχειρήματα/κόμβοι του γράφου είναι οι μεταβλητές στις οποίες θα ανατεθούν τιμές από το Πεδίο Ορισμού.

2) Πεδίο Ορισμού (Domain): {0,1}, για κάθε μεταβλητή. Χρήση Άλγεβρας Boolean, όπου:

i. $a_1 = 1 \rightarrow$ Το επιχείρημα a_1 είναι μέλος της Ευσταθούς Επέκτασης
ii. $a_1 = 0 \rightarrow$ Το επιχείρημα a_1 δεν είναι μέλος της Ευσταθούς Επέκτασης

3) Περιορισμοί (Constraints as Clauses): Εξάγονται από τους όρους που περιγράφονται στον ορισμό της Ευσταθούς Επέκτασης, αφού πρώτα μεταφραστούν σε Προτασιακή μορφή ή Γραμμικές Ανισότητες/ Εξισώσεις.

4) Ικανοποιησιμότητα (Satisfiability): Επιτυχία ανάθεσης τιμών σε όλους τους κόμβους, ικανοποιώντας παράλληλα όλους τους απαραίτητους περιορισμούς.

5) Μη-Ικανοποιησιμότητα (Unsatisfiability): Αποτυχία ανάθεσης τιμών, λόγω αδυναμίας ικανοποίησης όλων των απαραίτητων περιορισμών.

2.2 Γραφική Αναπαράσταση

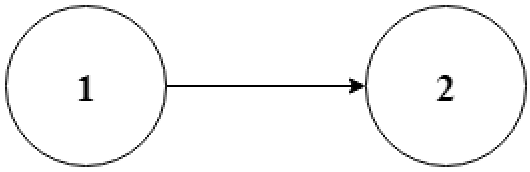
2.2.1 Δομή Γράφου [5, 6, 10]

Για να μπορέσουμε να κατανοήσουμε ευκολότερα το χώρο της επιχειρηματολογίας, χρησιμοποιούμε μεθόδους γραφικής αναπαράστασης. Πιο συγκεκριμένα, μπορούμε να φανταστούμε αφαιρετικά, ολόκληρο το σύστημα της επιχειρηματολογίας ως ένα κατευθυνόμενο γράφο για τον οποίο ισχύουν τα πάρα κάτω:

1) Κόμβος (κύκλος) = Επιχείρημα με δείκτη N
2) Ακμή (βέλος με κατεύθυνση) = Συσχέτιση Επίθεσης

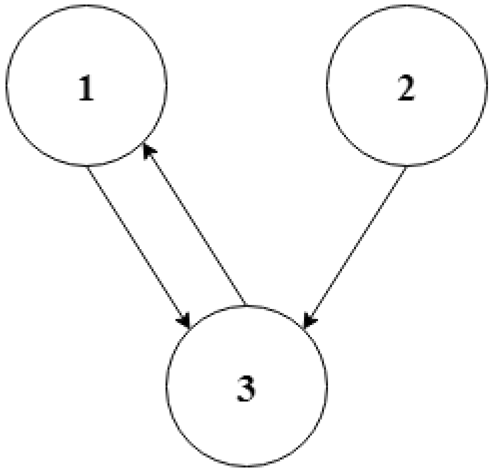
Προτιμούμε την αφαιρετική προσέγγιση στην γραφική αναπαράσταση, καθώς έτσι αποκλίνουμε από την εκτενή παρουσίαση περιττών περιγραφών και σχολίων, εστιάζοντας έτσι στις πιο σημαντικές πληροφορίες που μπορούμε να εξάγουμε με μία πρώτη ματιά από ένα σχήμα επιχειρηματολογίας (π.χ. πυκνότητα γράφου, μέγεθος του χώρου, ομαδοποιήσεις και αντιπαρατάξεις μεταξύ επιχειρημάτων κ.λπ.). Εξ’ ου και η ύπαρξη μόνο δύο πολύ απλών σχημάτων (κύκλος = κόμβος, βέλος = ακμή).

Σημασία Ακμής: Η ύπαρξη ακμής από τον κόμβο 1 προς τον κόμβο 2 ερμηνεύεται ως: «Το επιχείρημα 1 αντιτίθεται/επιτίθεται στο επιχείρημα 2» (βλ. Σχήμα 2.2.1.1).



Σχήμα 2.2.1.1 Συσχέτιση Επίθεσης

Ακολουθεί παράδειγμα εύρεσης του συνόλου Ευσταθούς Επέκτασης, δοθέντος ενός κατευθυνόμενου γράφου (βλ. Σχήμα 2.2.1.2), με σκοπό την περαιτέρω κατανόηση, τόσο του όρου αυτού, όσο και της ερμηνείας μίας γραφικής αναπαράστασης Επιχειρηματολογίας.



Σχήμα 2.2.1.2 Παράδειγμα: Εφαρμογή στην πολιτική (σε περιορισμένη κλίμακα)

Επιχειρήματα:

1: «Θα στηρίξουμε τον πολιτικό Α διότι είναι νεαρός και αντιλαμβάνεται τα προβλήματα των νέων, σε αντίθεση με τον ηλικιωμένο πολιτικό Β».

2: «Θα στηρίξουμε τον πολιτικό Α διότι δεν ενεπλάκη ποτέ σε οποιουδήποτε είδους σκάνδαλα, κάτι για το οποίο κατηγορείται ο πολιτικός Β».

3: «Θα στηρίξουμε τον πολιτικό Β, ο οποίος είναι αρκετά έμπειρος σε θέματα εκτελεστικής εξουσίας, τομέας στον οποίο υστερεί ο πολιτικός Α λόγω του νεαρού της ηλικίας τους».

Παρατηρήσεις:

Τόσο ο κόμβος 1, όσο και ο κόμβος 2, επιτίθενται στον κόμβο 3. Ταυτόχρονα, ο κόμβος 3 απαντά, με δική του επίθεση, στον κόμβο 1. Παρατηρούμε ότι δεν υπάρχει επίθεση μεταξύ κόμβων 1 και 2.

Συμπεράσματα:

Οι κόμβοι 1 και 2 είναι τα επιχειρήματα που απαρτίζουν την Ευσταθή Επέκταση του εν λόγω προβλήματος. Στηρίζουν τον ίδιο απώτερο σκοπό (την στήριξη του πολιτικού Α), δεν αντιτίθενται το ένα το άλλο, και ταυτόχρονα επιτίθενται στον κόμβο (επιχείρημα) 3, ο οποίος εναντιώνεται στη θέση που στηρίζει το επιχείρημα 1 (εφόσον στηρίζει την υποψηφιότητα διαφορετικού πολιτικού Β, με επιχείρημα που έρχεται σε άμεση αντιπαράθεση με αυτό που εκφράζεται στον κόμβο 1 (θετικά/ αρνητικά του νεαρού της ηλικίας του πολιτικού Α)).

2.2.2 Χρήσιμα Στοιχεία Δομής Γράφου [5, 6]

Πυκνότητα: Πραγματικός αριθμός-συντελεστής (με πεδίο τιμών $d \geq 0.0$). Δείχνει το πόσο πυκνός είναι ο γράφος, όσον αφορά το πλήθος ακμών που εισέρχονται/εξέρχονται, σε/από κόμβους. Η πυκνότητα, στην επιχειρηματολογία, μπορεί να ερμηνευτεί και ως το μέγεθος της σύγκρουσης μεταξύ των διαφόρων επιχειρημάτων του χώρου. Χρησιμοποιείται κυρίως για την εύρεση ακμών σε κάποιο τυχαιοποιημένο, κατευθυνόμενο γράφο, δοθέντος του αριθμού των κόμβων:

Φόρμουλα 2.2.2.1: Αρ. Ακμών = Αρ. Κόμβων * (Αρ. Κόμβων - 1) * Πυκνότητα

Σημαντική Σημείωση: Για το πρόβλημα της Ευσταθούς Επέκτασης, ο μέγιστος αριθμός ακμών, είναι αυτός που προκύπτει από την περίπτωση όπου κάθε επιχείρημα επιτίθεται σε κάθε άλλο επιχείρημα, δηλαδή:

Φόρμουλα 2.2.2.2: Μέγιστος Αριθμός Ακμών = Αρ. Κόμβων * (Αρ. Κόμβων - 1)

Η πάρα πάνω σημείωση βασίζεται στο ότι δεν είναι δυνατόν επιχειρήματα να επιτίθενται στον εαυτό τους (αντίφαση), ενώ θέλουμε ταυτόχρονα να αποφύγουμε περιττά στοιχεία στον γράφο (π.χ. 2 ακμές να εξέρχονται από κάποιο κόμβο και να εισέρχονται και οι 2, σε ίδιο κόμβο). Αυτό σημαίνει ότι ο Αριθμός Ακμών, όπως αυτός προκύπτει από τη Φόρμουλα 2.2.2.1, πρέπει να είναι μικρότερος ή ίσος από το Μέγιστο Αριθμό Ακμών, όπως αυτός προκύπτει από τη Φόρμουλα 2.2.2.2. Επομένως, προκειμένου να διασφαλιστούν τα πάρα πάνω, θα πρέπει η τιμή της πυκνότητας να έχει ένα άνω όριο, τέτοιο ώστε:

Φόρμουλα 2.2.2.3: Αριθμός Ακμών ≤ Μέγιστος Αριθμός Ακμών

Τετριμμένο Γράφημα: Γράφημα με ένα και μοναδικό κόμβο, χωρίς ακμές. Λόγω της έλλειψης άλλων επιχειρημάτων/κόμβων, δεν υπάρχουν συσχετίσεις επίθεσης, με αποτέλεσμα να επικρατεί, αναγκαστικά, η θέση την οποία στηρίζει το ένα και μοναδικό επιχείρημα του σχήματος.

Βαθμός Εισόδου Κόμβου: Το πλήθος ακμών που εισέρχονται σε κάποιο κόμβο. Στον τομέα της επιχειρηματολογίας, μεγάλος βαθμός εισόδου για κάποιο κόμβο/επιχείρημα υποδηλώνει ότι το εν λόγω επιχείρημα τείνει μεγάλης αμφισβήτησης, καθώς αυτό συνεπάγεται με αποδοχή πολλών επιθέσεων εναντίον του.

Βαθμός Εξόδου Κόμβου: Το πλήθος ακμών που εξέρχονται από κάποιο κόμβο. Στον τομέα της επιχειρηματολογίας, μικρός βαθμός εξόδου για κάποιο κόμβο/επιχείρημα υποδηλώνει ότι το εν λόγω επιχείρημα δεν επιτίθεται σε μεγάλο αριθμό άλλων επιχειρημάτων, κάτι που μπορεί να σημαίνει ότι πολύ πιθανόν να χρειάζεται την υποστήριξη άλλων επιχειρημάτων, για την επικράτηση της θέσης που υποστηρίζει.

2.2.3 Μοντελοποίηση Γραφήματος [5, 6]

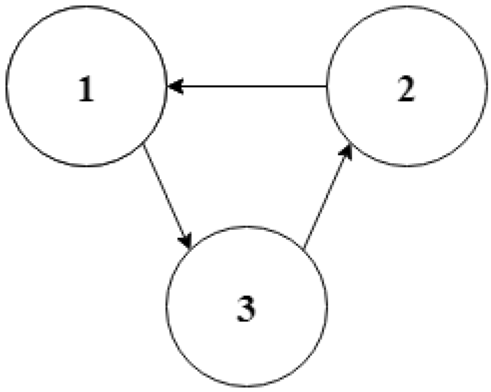
Οι γράφοι που αναπαριστούν προβλήματα Επιχειρηματολογίας μπορούν να μοντελοποιηθούν μέσω δισδιάστατων πινάκων ακεραίων αριθμών, με πεδίο τιμών {0,1}. Στη θεωρία των Γράφων, οι πίνακες αυτοί ονομάζονται «Πίνακες Γειτνίασης» (Adjacency Matrices) και παρουσιάζουν τις ακμές/συσχετίσεις μεταξύ κόμβων. Ο

Πίνακας Γειτνίασης έχει ίσο αριθμό στηλών-γραμμών, με μέγεθος NxN, όπου N είναι το πλήθος των κόμβων του γραφήματος. Συγκεκριμένα, για πίνακα γειτνίασης ΠΓ και για κάθε i, k ισχύει ότι:

1. Αν ΠΓ[i][k] = 1 → ο κόμβος i έχει ακμή που εξέρχεται από αυτόν και εισέρχεται στον κόμβο k → Για το πρόβλημα της Ευσταθούς Επέκτασης ισχύει ότι: «ο κόμβος i επιτίθεται στον κόμβο k».

2. Αν ΠΓ[i][k] = 0 → ο κόμβος i ΔΕΝ έχει ακμή που να εξέρχεται από αυτόν και να εισέρχεται στον κόμβο k → Για το πρόβλημα της Ευσταθούς Επέκτασης ισχύει ότι: «ο κόμβος i ΔΕΝ επιτίθεται στον κόμβο k».

3. ΠΓ[i][i] = 0, για το πρόβλημα της Ευσταθούς Επέκτασης, εφόσον κανένας κόμβος δεν μπορεί να επιτεθεί στον ίδιο του τον εαυτό. Αυτό θα θεωρείτο αντίφαση, εφόσον τα επιχειρήματα δεν πρέπει να αναιρούν τα ίδια τους τα συμφραζόμενα.



Παράδειγμα 2.2.3.1 – Γραφική Αναπαράσταση

001
100
010

Παράδειγμα 2.2.3.1 – Αντίστοιχος Πίνακας Γειτνίασης

2.3 Επιλυτές και Άλλα Εργαλεία

2.3.1 Εισαγωγή στους «Επιλυτές» [19]

Εφόσον καθοριστούν τα πλαίσια του προβλήματος, σειρά έχει η επίλυση του, για την οποία αρμόδια είναι τα προγράμματα «Επιλυτές» (Solvers). Δίνοντας ως είσοδο τους περιορισμούς και τις μεταβλητές του προβλήματος, ο επιλυτής μας παρέχει ένα μοντέλο λύσης, δηλαδή μία κατάλληλη ανάθεση τιμών στις μεταβλητές, που θα ικανοποιεί όλους τους περιορισμούς.

Η λύση ερμηνεύεται κατάλληλα ώστε να εξαχθούν, τόσο το σύνολο επιχειρημάτων της Ευσταθούς Επέκτασης, όσο και σημαντικά συμπεράσματα που αφορούν το πρόβλημα και τον τρόπο επίλυσης (π.χ. χρόνοι αναζήτησης και επίλυσης, κατάσταση ικανοποιησιμότητας κ.λπ.).

2.3.2 Σημαντικότητα Χρήσης Επιλυτών [19]

Η χρήση των επιλυτών αντικαθιστά τη χρονοβόρα, κουραστική και δύσκολα επαληθεύσιμη διαδικασία επίλυσης με το χέρι. Η ανάπτυξη τους αποτελεί ένα σπουδαίο επίτευγμα της Τεχνητής Νοημοσύνης καθώς η χρήση τους επιτρέπει την αποκέντρωση μας από τη διαδικασία επίλυσης, δίνοντας μας έτσι την ευκαιρία να εστιάσουμε την ενέργεια και το ενδιαφέρον μας στα ίδια τα αποτελέσματα, καθώς επίσης και στα συμπεράσματα που εξάγονται από αυτά. Χρησιμοποιούνται ευρέως, τόσο για ακαδημαϊκούς, όσο και για βιομηχανικούς σκοπούς και αποτελούν, με λίγα λόγια, τις «υπολογιστικές μηχανές» του κλάδου της Ικανοποίησης Περιορισμών.

2.3.3 Επιλυτές που χρησιμοποιήθηκαν [8, 12, 15]

1) Lingeling: SAT Solver (Johannes Kepler University, Linz, Austria). Επίλυση Προβλημάτων Ικανοποιησιμότητας (SAT) μέσω Προτασιακής Λογικής. Απέσπασε αρκετά βραβεία σε διαγωνισμούς ανάπτυξης επιλυτών SAT προβλημάτων. Διανέμεται δωρεάν και απαιτεί την χρήση συστήματος Linux/UNIX, καθώς το εκτελέσιμο είναι αρχείο αντικειμένου (object file με

επέκταση .o), για το οποίο οι εντολές που χρησιμοποιούνται για την διαχείρισή του εκτελούνται στο τερματικό των προαναφερθέντων λειτουργικών συστημάτων. Για σκοπούς συγγραφής του κειμένου αυτού χρησιμοποιήθηκε η νεότερη έκδοση «ats1», η οποία κατά την εκτύπωση των αποτελεσμάτων εκτυπώνει και γενικές πληροφορίες για τον επιλυτή και τη διαδικασία επίλυσης. Για σκοπούς πειραματικών δοκιμών χρησιμοποιήθηκε η παλαιότερη έκδοση 276-6264d55-100731, η οποία εκτυπώνει μόνο κατάσταση και μοντέλο, χωρίς επιπλέον πληροφορίες (minimal version). Παρόλα αυτά, γενικά συνίσταται η χρήση του «ats1» καθώς περιέχει αρκετές βελτιώσεις.

2) Maxino: MAXSAT Solver (Mario Alviano - Department of Mathematics and Computer Science, University of Calabria, Italy). Επίλυση Προβλημάτων Μέγιστης Ικανοποιησιμότητας (MAXSAT) μέσω Προτασιακής Λογικής. Χρήση βαρών για καθορισμό χαλαρών/ισχυρών περιορισμών. Διανέμεται δωρεάν και απαιτεί την χρήση συστήματος Linux/UNIX, καθώς το εκτελέσιμο είναι ένα αρχείο αντικειμένου (object file με επέκταση .o), για το οποίο οι εντολές που χρησιμοποιούνται για την διαχείρισή του εκτελούνται στο τερματικό των προαναφερθέντων λειτουργικών συστημάτων. Για τους σκοπούς της εργασίας αυτής χρησιμοποιείται ο επιλυτής maxino-2015-k16-static.

3) Intlinprog: Linear Programming solver (προσφέρεται μέσω της MATLAB). Επίλυση Προβλημάτων Ικανοποιησιμότητας (SAT) μοντελοποιημένων ως προβλήματα Γραμμικού Προγραμματισμού (μέσω χρήσης Γραμμικών Ανισοτήτων/ Εξισώσεων).

2.3.4 Επιπλέον Εργαλεία [7, 13, 16, 18]

1) Eclipse: Java IDE (Developed by Eclipse Foundation). Περιβάλλον ανάπτυξης προγραμμάτων, κυρίως σε γλώσσα προγραμματισμού JAVA. Χρησιμοποιήθηκε η έκδοση Neon, για την ανάπτυξη του προγράμματος της Ατομικής Διπλωματικής Εργασίας.

2) MATLAB: Numerical Computing Environment (Developed by MathWorks). Περιβάλλον ανάπτυξης προγραμμάτων αριθμητικού υπολογισμού. Χρησιμοποιήθηκε η έκδοση MATLAB R2018b ως «παροχέας» του επιλυτή «intlinprog».

3) Draw.io (https://www.draw.io/): Ιστοσελίδα που προσφέρει τη δυνατότητα δημιουργίας γραφικών αναπαραστάσεων/ διαγραμμάτων/ σχεδιαγραμμάτων.

4) Microsoft Excel: Spreadsheet (Developed by Microsoft). Λογισμικό Δημιουργίας Λογιστικών/ Στατιστικών Φύλλων. Χρησιμοποιήθηκε για την συγκέντρωση, επεξεργασία και γραφική αναπαράσταση αποτελεσμάτων χρόνων εκτέλεσης, που προέκυψαν από την εκτέλεση πειραμάτων. Για σκοπούς αυτής της εργασίας χρησιμοποιήθηκε η έκδοση «Microsoft Office Excel 2007»

Κεφάλαιο 3

Μελέτη Ευσταθούς Επέκτασης στην Προτασιακή Λογική

3.1 Αρχική Ιδέα [1, 10]

Ένας αρκετά απλός τρόπος να ορίσουμε τους περιορισμούς που διέπουν ένα πρόβλημα Ευσταθούς Επέκτασης είναι η χρήση του Προτασιακού Λογισμού. Με την αξιοποίηση προτασιακών τελεστών και μεταβλητών, δηλώνουμε πράξεις σε άλγεβρα Boole, οι οποίες αντιστοιχούν σε περιορισμούς. Η σύζευξη όλων των περιορισμών, δηλαδή η δημιουργία μίας ενιαίας πρότασης σε Συζευκτική Κανονική Μορφή (CNF), της οποίας όροι αποτελούν οι περιορισμοί, αποτελεί την πρόταση που θέλουμε να ικανοποιείται, ώστε το πρόβλημα να είναι Ικανοποιήσιμο. Μοντέλα ανάθεσης τιμών True (= 1) ή False (= 0) στις μεταβλητές, με τέτοιο τρόπο ώστε να ικανοποιούνται όλοι οι περιορισμοί, αποτελούν λύσεις του προβλήματος. Συμπληρωματική ανάλυση της τεχνικής πτυχής του Κεφαλαίου 3 υπάρχει στο Παράρτημα Α (υποκεφάλαιο Παραρτήματος Α: Α.1).

3.2 Χρήση Επιλυτών

3.2.1 Επιλυτές Προτασιακής Ικανοποιησιμότητας [12, 19]

Τα πλείστα προγράμματα επίλυσης SAT προβλημάτων (SAT solvers) χρησιμοποιούν ως βάση τους την Προτασιακή Λογική. Για την εν λόγω εργασία θα χρησιμοποιηθεί ο επιλυτής «Lingeling». Ο εν λόγω επιλυτής λαμβάνει ως είσοδο ένα πρότυπο αρχείο Συζευκτικής Κανονικής Μορφής (CNF file με επέκταση .cnf) το οποίο περιέχει τους Προτασιακούς περιορισμούς του προβλήματος, σε Συζευκτική Κανονική Μορφή. Αυτό σημαίνει ότι περιέχει τους περιορισμούς ως προτάσεις, αποτελούμενες αποκλειστικά από διαζεύξεις μεταβλητών ή/και αρνήσεων τους, οι οποίες μεταξύ τους συνδέονται με συζεύξεις.

Ο Lingeling κάνει τις απαραίτητες μειώσεις περιορισμών και πεδίων τιμών ώστε να καταλήξει σε κάποιο μοντέλο ανάθεσης τιμών που να ικανοποιεί όλους τους περιορισμούς. Σε περίπτωση επιτυχίας επιστρέφεται το μοντέλο με μία διαβεβαίωση ότι το πρόβλημα είναι «Ικανοποιήσιμο» (SATISFIABLE), αλλιώς επιστρέφεται μία διαβεβαίωση ότι το πρόβλημα είναι «Μη- Ικανοποιήσιμο» (UNSATISFIABLE).

3.2.2 Αρχείο Συζευκτικής Κανονικής Μορφής [17, 20]

Τα αρχεία με επέκταση .cnf αποτελούν την επίσημη μορφή πρότυπου αρχείου εισόδου που αποδέχονται οι επιλυτές SAT Προβλημάτων. Περιγράφουν ένα πρόβλημα περιορισμών το οποίο εκφράζεται σε Συζευκτική Κανονική Μορφή Προτασιακής Λογικής, της οποίας ο κάθε ένας από τους όρους που απαρτίζουν τις συζεύξεις της, αποτελεί περιορισμό που πρέπει να ικανοποιηθεί.

Ακολουθούν οι συντακτικοί κανόνες των αρχείων .cnf, καθώς και οι αντίστοιχες εκφράσεις Προτασιακής Λογικής, στη μορφή που απαιτούνται:

1) Μεταβλητή/ Όρος (literal): Οι μεταβλητές στα αρχεία .cnf είναι σε μορφή ακέραιων αριθμών, με τιμή μεγαλύτερη από το 0. Ο αριθμός 0 είναι δεσμευμένος χαρακτήρας, ο οποίος χρησιμοποιείται για την επισήμανση του τέλους μίας γραμμής.

2) Άρνηση ή Συμπλήρωμα Μεταβλητής (complement): Η παρουσία του χαρακτήρα '-' μπροστά από κάποια μεταβλητή δηλώνει το συμπλήρωμα/άρνηση της.

   Π.χ. Συμβολική Μορφή: $\neg 1$

   Πρότυπη μορφή .cnf αρχείου: -1

3) Λογική Διάζευξη (Logical Disjunction): Αναπαρίσταται με ένα κενό (space) μεταξύ μεταβλητών ίδιας γραμμής κειμένου.

   Π.χ. Συμβολική Μορφή: $1 \vee 2 \vee \neg 3$

   Πρότυπη μορφή .cnf αρχείου: $1\ 2\ -3\ 0$

4) Λογική Σύζευξη (Logical Conjunction): Αναπαρίσταται με την αλλαγή γραμμής, η οποία υποδηλώνει Σύζευξη μεταξύ των προτάσεων των 2 γραμμών κειμένου που διαχωρίζει.

   Π.χ. Συμβολική Μορφή: $1 \wedge 2 \wedge \neg 3$

   Πρότυπη μορφή .cnf αρχείου: $1\ 0$

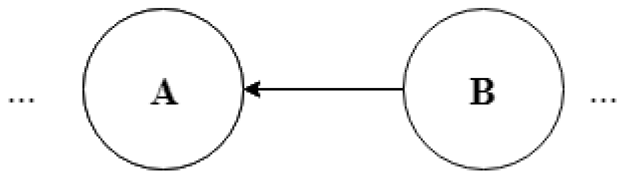
   $2\ 0$

   $-3\ 0$

3.3 Προτασιακοί Περιορισμοί [1, 10]

Μέσα από τον ορισμό της Ευσταθούς Επέκτασης πηγάζουν οι απαραίτητοι περιορισμοί, οι οποίοι είναι οι εξής:

1. **1.** Αποκλεισμός Κοινής Συμμετοχής Αντίπαλων Επιχειρημάτων: Εάν κάποιο επιχείρημα δέχεται επίθεση από κάποιο άλλο επιχείρημα, τότε δεν είναι δυνατόν να αποτελούν, και τα δύο, μέλη της Ευσταθούς Επέκτασης.



Σχήμα 3.3.1 Συσχέτιση Επίθεσης

Για κάθε συσχέτιση επίθεσης μεταξύ δύο επιχειρημάτων, όπου το επιχείρημα A δέχεται επίθεση από το επιχείρημα B, ισχύουν οι πάρα κάτω περιορισμοί:

- Περιορισμός (σε Προτασιακή Λογική):

$$(A \rightarrow \neg B) \wedge (B \rightarrow \neg A)$$

$$\equiv$$

$$\neg A \vee \neg B$$

- Περιορισμός (σε μορφή CNF Αρχείου):

–A –B 0

- Επεξήγηση:

  Εάν το A είναι μέλος της Ευσταθούς Επέκτασης, το B δεν είναι ΚΑΙ εάν το B είναι μέλος της Ευσταθούς Επέκτασης, το A δεν είναι.

  $\equiv$

  Τα επιχειρήματα A και B δεν είναι και τα δύο μαζί, ταυτόχρονα, μέλη της Ευσταθούς Επέκτασης.

2. Κριτήρια Συμμετοχής: Για κάθε επιχείρημα ισχύει ότι, είτε το ίδιο, είτε οποιοσδήποτε αριθμός από επιχειρήματα που του επιτίθενται, αποτελούν μέλη της Ευσταθούς Επέκτασης.

- Περιορισμός (σε Προτασιακή Λογική):

$$(\neg A \rightarrow attA_1 \vee \dots \vee attA_n)$$
$$\equiv$$
$$(A \vee attA_1 \vee \dots \vee attA_n)$$

*Σημείωση: Στη θέση των «…», τοποθετούνται όλα τα επιχειρήματα που επιτίθονται στο επιχείρημα A, διαχωρισμένα από σύμβολο διάζευξης ('$\vee$'). Ο δείκτης n αποτελεί το πλήθος των επιχειρημάτων που επιτίθονται στο A.

- Περιορισμός (σε μορφή CNF Αρχείου):

$$A\ attA_i \dots attA_n\ 0$$

Όπου:

A: επιχείρημα
$attA_i$: το i-οστό επιχείρημα που επιτίθεται στο επιχείρημα A

*Σημείωση: Στη θέση των «…», τοποθετούνται όλα τα επιχειρήματα που επιτίθονται στο επιχείρημα A, διαχωρισμένα από ένα κενό διάστημα. Ο δείκτης n αποτελεί το πλήθος των επιχειρημάτων που επιτίθονται στο A.

- Επεξήγηση:

Εάν το A δεν είναι μέλος της Ευσταθούς Επέκτασης, τότε σίγουρα είναι μέλος της κάποιο/α από τα επιχειρήματα που επιτίθενται στο A.

$$\equiv$$

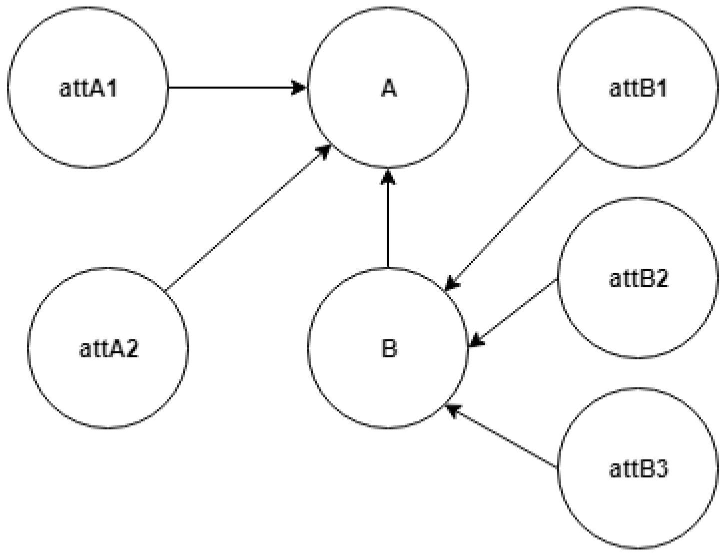
Είτε το A, είτε κάποιο/α από τα επιχειρήματα που του επιτίθενται, είναι μέλος/μέλη της Ευσταθούς Επέκτασης.

- Παράδειγμα:

  Για περαιτέρω κατανόηση του περιορισμού αυτού, ορίζουμε τα ακόλουθα επιχειρήματα, με τις μεταξύ τους συσχετίσεις:

  - A: επιχείρημα που δέχεται επίθεση από το επιχείρημα B.
  - B: επιχείρημα που επιτίθεται στο επιχείρημα A.
  - $attA_i$: Κάθε επιχείρημα που επιτίθεται στο επιχείρημα A
  - $attB_j$: Κάθε επιχείρημα που επιτίθεται στο επιχείρημα B

Ας δημιουργήσουμε ένα σχήμα που να αναπαριστά τα πάρα πάνω στοιχεία:



Σχήμα 3.3.2 Πολλαπλές Συσχετίσεις

Για A, B, $attA_1$, $attA_2$, $attB_1$, $attB_2$, $attB_3$, ισχύουν οι ακόλουθοι περιορισμοί:

- Περιορισμοί που προκύπτουν (σε Προτασιακή Λογική):

$$(\neg A \rightarrow B \lor attA_1 \lor attA_2) \land (\neg B \rightarrow attB_1 \lor attB_2 \lor attB_3)$$

$$\equiv$$

$$(A \lor B \lor attA_1 \lor attA_2) \land (B \lor attB_1 \lor attB_2 \lor attB_3)$$

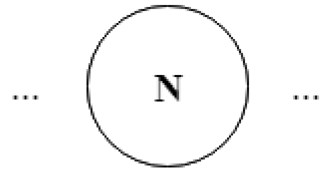
- Περιορισμοί (σε μορφή CNF Αρχείου):

$$A\ B\ attA_1\ attA_2\ 0$$

$$B\ attB_1\ attB_2\ attB_3\ 0$$

3. Τετριμμένη Συμμετοχή Ουδέτερων Επιχειρημάτων : Εάν κάποιο επιχείρημα δεν δέχεται, ούτε εξαπολύει, επίθεση από/σε οποιοδήποτε άλλο επιχείρημα, τότε τετριμμένα ανήκει στην Ευσταθή Επέκταση.

Ο περιορισμός αυτός δεν εξυπηρετεί ιδιαίτερα το σύνολο της Ευσταθούς Επέκτασης στο να αντιμετωπίσει τις απειλές που δέχεται (εφόσον ένα ουδέτερο επιχείρημα δεν επιτίθεται σε άλλα επιχειρήματα). Παρόλα αυτά, ο ορισμός του εν λόγω περιορισμού βασίζεται κυρίως στο γεγονός ότι ούτε προβλήματα δημιουργεί η συμμετοχή ενός ουδέτερου κόμβου στην Ευσταθή Επέκταση.



Σχήμα 3.3.3 Ουδέτερο Επιχείρημα

Για ΚΑΘΕ ουδέτερο επιχείρημα **N**, ισχύει ο πάρα κάτω μοναδιαίος περιορισμός:

- Περιορισμός (σε Προτασιακή Λογική):

N

- Περιορισμός (σε μορφή CNF Αρχείου):

N 0

- Επεξήγηση: Το ουδέτερο επιχείρημα N είναι μέλος της Ευσταθούς Επέκτασης.

- Σημείωση: Ο εν λόγω περιορισμός προκύπτει από τον περιορισμό υπ' αριθμό 2 (Κριτήρια Συμμετοχής), εφόσον για κάθε ουδέτερο επιχείρημα ισχύει ότι, είτε το ίδιο, είτε κάποια από τα επιχειρήματα που του επιτίθενται, είναι μέλη της Ευσταθούς Επέκτασης. Το γεγονός όμως ότι τα ουδέτερα επιχειρήματα δεν δέχονται επιθέσεις, τα κατατάσσει, αναγκαστικά, στο σύνολο της Ευσταθούς Επέκτασης.

3.4 Αποτελέσματα

3.4.1 Εμφάνιση Αποτελεσμάτων [17, 20]

Η επιτυχής κλήση του επιλυτή οδηγεί στην έξοδο των αποτελεσμάτων, εντός του τερματικού. Εκεί αναγράφονται διάφορες πληροφορίες σχετικά με τον επιλυτή (π.χ. έκδοση, πληροφορίες εκτέλεσης κ.λπ.) καθώς επίσης και στοιχεία που αφορούν το ίδιο το πρόβλημα (π.χ. πλήθος μεταβλητών και περιορισμών, Ικανοποιησιμότητα, μοντέλο επίλυσης, ποσοστά χρόνου απλοποίησης και αναζήτησης κ.λπ.). Εξ' αυτών, ξεχωρίζουμε τις σημαντικότερες πληροφορίες, οι οποίες είναι το μοντέλο ανάθεσης τιμών (λύση προβλήματος) και η κατάσταση του προβλήματος (Ικανοποιησιμότητα).

Μπορούμε να διακρίνουμε το είδος της πληροφορίας που μας παρέχει η κάθε γραμμή κειμένου, της εξόδου του επιλυτή, μέσω του πρώτου χαρακτήρα της κάθε γραμμής:

Γραμμές που ξεκινούν με τον χαρακτήρα 'c' (comment): περιέχουν σχόλια και πληροφορίες σχετικά με τον επιλυτή και το πρόβλημα.

Μία μοναδική γραμμή που ξεκινά με τον χαρακτήρα 's' (satisfiability): περιέχει την Κατάσταση Ικανοποιησιμότητας του προβλήματος.

Γραμμές που ξεκινούν με τον χαρακτήρα 'v' (variables): περιέχουν το Μοντέλο Λύσης του προβλήματος (εάν και εφόσον είναι Ικανοποιήσιμο).

3.4.2 Ερμηνεία Αποτελεσμάτων

Κατάσταση Ικανοποιησιμότητας Προβλήματος: Εάν επιστραφεί «SATISFIABLE», τότε το πρόβλημα είναι Ικανοποιήσιμο και ακολουθεί το μοντέλο ανάθεσης τιμών. Εάν επιστραφεί «UNSATISFIABLE», τότε το πρόβλημα δεν είναι Ικανοποιήσιμο και δεν επιστρέφεται κάποιο μοντέλο ανάθεσης τιμών, λόγω της αδυναμίας εύρεσης ενός.

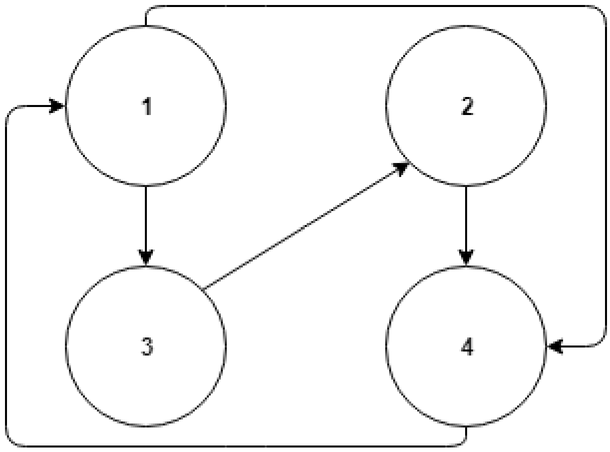
Μοντέλο Ανάθεσης Τιμών: Οι τιμές που ανατίθενται στις μεταβλητές έχουν ως εξής:

Μεταβλητή = True (1), εάν δεν υπάρχει πρόσημο ‘-‘ μπροστά από το όνομά της.
Μεταβλητή = False (0), εάν υπάρχει πρόσημο ‘-‘ μπροστά από το όνομά της.

3.5 Παραδείγματα Διαδικασίας

3.5.1 Ικανοποιήσιμο πρόβλημα

3.5.1.1 Γραφική Αναπαράσταση



Σχήμα 3.5.1.1 Ικανοποιήσιμο Πρόβλημα

3.5.1.2 Πίνακας Γειτνίασης

0011
0001
0100
1000

3.5.1.3 Περιορισμοί

Προτασιακή Μορφή:

$$(\neg 3 \lor \neg 1) \land (\neg 4 \lor \neg 1) \land (\neg 3 \lor \neg 2) \land (\neg 4 \lor \neg 2) \land (1 \lor 4) \land (2 \lor 3) \land (3 \lor 1) \land (4 \lor 1 \lor 2)$$

Μορφή Αρχείου CNF:

```
-3 -1 0
-4 -1 0
-3 -2 0
-4 -2 0
1 4 0
2 3 0
3 1 0
4 1 2 0
```

3.5.1.4 Αποτελέσματα

```
s SATISFIABLE
v 1 2 -3 -4 0
```

Σχήμα 3.5.1.2 Μοντέλο Λύσης

3.5.1.5 Ερμηνεία Αποτελεσμάτων

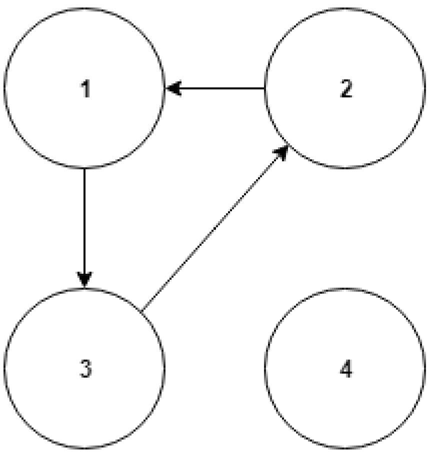
Κατάσταση Προβλήματος: ΙΚΑΝΟΠΟΙΗΣΙΜΟ (SATISFIABLE)

Μοντέλο Λύσης: $1\ 2\ -3\ -4\ 0 \rightarrow$ Ευσταθής Επέκταση $= \{1, 2\}$

Ερμηνεία: Τα επιχειρήματα 1 και 2 ομαδοποιούνται για να αντιμετωπίσουν τα επιχειρήματα 3 και 4, ενώ ταυτόχρονα δεν επιτίθενται το ένα στο άλλο.

3.5.2 Μη-Ικανοποιήσιμο πρόβλημα

3.5.2.1 Γραφική Αναπαράσταση



Σχήμα 3.5.2.1 Μη-Ικανοποιήσιμο Πρόβλημα

3.5.2.2 Πίνακας Γειτνίασης

0010

1000

0100

0000

3.5.2.3 Περιορισμοί

Προτασιακή Μορφή:

$$(\neg 2 \vee \neg 1) \wedge (\neg 3 \vee \neg 1) \wedge (\neg 3 \vee \neg 2) \wedge (1 \vee 2) \wedge (2 \vee 3) \wedge (3 \vee 1) \wedge (4)$$

Μορφή Αρχείου CNF:

```
-2 -1 0
-3 -1 0
-3 -2 0
1 2 0
2 3 0
3 1 0
4 0
```

3.5.2.4 Αποτελέσματα

```
s UNSATISFIABLE
```

Σχήμα 3.5.2.2 Αδυναμία Εύρεσης Μοντέλου Λύσης

3.5.2.5 Ερμηνεία Αποτελεσμάτων

Κατάσταση Προβλήματος: ΜΗ-ΙΚΑΝΟΠΟΙΗΣΙΜΟ (UNSATISFIABLE)

Μοντέλο Λύσης: Δεν υπάρχει λύση → Ευσταθής Επέκταση = { }

Ερμηνεία: Δεν υπάρχουν επιχειρήματα που να μπορούν να ομαδοποιηθούν μαζί, ώστε να αντιμετωπίσουν όλα τα αντίπαλα επιχειρήματα, και ταυτόχρονα, μεταξύ τους να μην υπάρχει συσχέτιση επίθεσης. Δεν υπάρχει δηλαδή, δυνατό υποσύνολο του συνόλου των επιχειρημάτων, το οποίο να ικανοποιεί τον ορισμό της Ευσταθούς Επέκτασης, για το συγκεκριμένο πρόβλημα.

Κεφάλαιο 4

Μελέτη Μη-Ικανοποιησιμότητας στην Προτασιακή Λογική

4.1 Αρχική Ιδέα [8, 14]

Παρόλο που σε αρκετές περιπτώσεις προβλημάτων «Ευσταθούς Επέκτασης» καθίσταται αδύνατον να εξευρεθεί μία λύση, υπάρχουν τρόποι χειρισμού τους οι οποίοι, μέσω της ελάχιστης δυνατής χαλάρωσης περιορισμών, προσφέρουν την καλύτερη δυνατή συμβιβαστική λύση στο πρόβλημα.

Σε αρκετά προβλήματα του πραγματικού κόσμου, αρκετά ικανοποιητικές είναι και οι λύσεις οι οποίες είναι «σχεδόν» τόσο καλές, όσο η «ιδανική» λύση. Πολλές φορές, κάποιος οργανισμός αντιμετωπίζει τον κίνδυνο να μην πετύχει όλους του τους στόχους, όπως αναμενόταν (π.χ. χαμηλό κέρδος/ μεγάλο κόστος). Στα σενάρια αυτά, αρκείται στο να πετύχει όσο το δυνατόν περισσότερους από τους στόχους του, υποκύπτοντας, όσο το δυνατόν λιγότερο, σε αναγκαστικούς, αλλά βιώσιμους συμβιβασμούς.

Η διαδικασία χειρισμού μη-ικανοποιήσιμων περιπτώσεων είναι αρκετά σημαντική, καθώς εξαλείφει την δυαδικότητα που διέπει την Ικανοποιησιμότητα (δηλαδή τις επιλογές «Ικανοποιήσιμο» = επιτυχία και «Μη- Ικανοποιήσιμο» = αποτυχία), προσφέροντας και ενδιάμεσες περιπτώσεις, κάτι που χαρακτηρίζει τον πολύπλευρο, πολύπλοκο πραγματικό κόσμο.

Π.χ. Όταν κάποιο πρόβλημα γίνεται Ικανοποιήσιμο μέσω της αφαίρεσης ελάχιστων περιορισμών, τότε έχουμε επιτυχία ελάχιστου συμβιβασμού. Όταν κάποιο πρόβλημα γίνεται Ικανοποιήσιμο μέσω της αφαίρεσης πολλών περιορισμών, τότε πλησιάζουμε στα πλαίσια της αποτυχίας, καθώς έγιναν πολλοί συμβιβασμοί.

Συμπληρωματική ανάλυση της τεχνικής πτυχής του Κεφαλαίου 4 υπάρχει στο Παράρτημα Α (υποκεφάλαιο Παραρτήματος Α: Α.2).

4.2 Τρόποι Χειρισμού μέσω Προτασιακής Λογικής [1, 14, 21]

Εφόσον τα προβλήματα Ικανοποιησιμότητας, γραφικά, αναπαρίστανται ως κατευθυνόμενοι γράφοι, μπορούν να ακολουθηθούν δύο προσεγγίσεις.

Οι γράφοι χαρακτηρίζονται από τους κόμβους («επιχειρήματα», στην περίπτωση της Ευσταθούς Επέκτασης) που τους απαρτίζουν, και τις μεταξύ τους συσχετίσεις, δηλαδή τις ακμές («επιθέσεις», στην περίπτωση της Ευσταθούς Επέκτασης). Επομένως μπορούμε να «τροποποιήσουμε» το πρόβλημα σε ένα πρόβλημα αρκετά όμοιας μορφής, αλλά με πιο χαλαρούς περιορισμούς, αφαιρώντας, είτε κάποιους από τους κόμβους, είτε κάποιες από τις ακμές που αποτελούν το γράφο.

Προκύπτουν, λοιπόν, οι δύο πάρα κάτω δυνατές προσεγγίσεις χαλάρωσης του προβλήματος:

1) Αφαίρεση των λιγότερων δυνατών ακμών (συσχετίσεων επίθεσης) από κάποιο «Μη- Ικανοποιήσιμο» πρόβλημα, των οποίων η απουσία θα το καθιστά «Ικανοποιήσιμο».

2) Αφαίρεση των λιγότερων δυνατών κόμβων (επιχειρημάτων) από κάποιο «Μη-Ικανοποιήσιμο» πρόβλημα, των οποίων η απουσία θα το καθιστά «Ικανοποιήσιμο».

4.3 Σημασία Βελτιστότητας [2, 21]

Αναφερόμαστε πάντοτε στην εύρεση και εξαγωγή του ελάχιστου δυνατού πλήθους ακμών ή κόμβων, καθώς σκοπός μας είναι η εύρεση μίας συγκαταβατικής, βέλτιστης λύσης η οποία απαιτεί το μικρότερο βαθμό συμβιβασμού.

Δύο θεωρητικές, αλλά καθόλου πρακτικές, προσεγγίσεις, τις οποίες απορρίπτουμε, θα ήταν οι ακόλουθες:

1) Αφαίρεση όλων των ακμών από το γράφο, διαγράφοντας έτσι όλους τους περιορισμούς της Βάσης Γνώσης που υποδηλώνουν την ύπαρξη επίθεσης.

2) Αφαίρεση όλων των κόμβων από το γράφο, διαγράφοντας έτσι όλους τους περιορισμούς της Βάσης Γνώσης.

Τα πάρα πάνω είναι πολύ απλές προσεγγίσεις, ωστόσο, αποτελούν προϊόντα της μεγαλύτερης δυνατής υποχώρησης που θα μπορούσε να συμβεί, κάτι το οποίο δεν είναι επιθυμητό όταν ο σκοπός μας είναι να προσεγγίσουμε την «τέλεια» λύση, δηλαδή να φτάσουμε όσο πιο κοντά γίνεται στην περίπτωση όπου όλοι οι περιορισμοί ικανοποιούνται.

4.4 Χρήση Επιλυτών

4.4.1 Επιλυτές Προτασιακής Ικανοποιησιμότητας με Βάρη [8]

Εφόσον ο επιλυτής Lingeling επιστρέφει μοντέλα επίλυσης μόνο για Ικανοποιήσιμα (SATISFIABLE) προβλήματα, και δεν υπάρχει τρόπος χρήσης του για περαιτέρω μελέτη των Μη-Ικανοποιήσιμων (UNSAT) προβλημάτων, θα χρησιμοποιηθούν επιπλέον εργαλεία, σε συνδυασμό με τον Lingeling, ο οποίος θα χρησιμοποιηθεί κυρίως για επαληθεύσεις. Για τις ανάγκες αυτού του μέρους της εργασίας θα χρησιμοποιηθεί ο «Maxino».

Ο «Maxino» είναι ένας επιλυτής που ασχολείται με MAXSAT, δηλαδή την εκδοχή προβλημάτων ικανοποιησιμότητας, των οποίων απώτερος σκοπός αποτελεί η ικανοποίηση των όσο το δυνατόν περισσότερων περιορισμών, με έμφαση στην ικανοποίηση των πιο «σημαντικών» εξ’ αυτών, ως προτεραιότητα. Η «σημαντικότητα» ενός περιορισμού (clause) παρουσιάζεται με κάποιο βάρος (weight), το οποίο όσο πιο μεγάλη τιμή έχει, τόσο πιο σημαντική αποτελεί η ικανοποίηση του. Βάσει αυτών των βαρών, οι MAXSAT solvers αναγνωρίζουν την σημαντικότητα, προσπαθώντας να ικανοποιήσουν περιορισμούς των οποίων το άθροισμα των βαρών τους είναι το μέγιστο δυνατό.

Το πλεονέκτημα του Maxino έναντι του Lingeling, όσον αφορά την εφαρμογή του στη μελέτη Μη-Ικανοποιήσιμων προβλημάτων, έγκειται στο γεγονός ότι μπορεί να επιστρέφει μοντέλα λύσεων για προβλήματα που είναι Μη-Ικανοποιήσιμα. Αυτό γίνεται πραγματοποιώντας τις απαραίτητες (ελάχιστες δυνατές) χαλαρώσεις σε περιορισμούς, σταματώντας τη διαδικασία επίλυσης στο σημείο όπου εάν προσπαθήσει να ικανοποιήσει έστω και ένα ακόμα περιορισμό, το πρόβλημα γίνεται «Μη-Ικανοποιήσιμο».

Ο εν λόγω επιλυτής είναι βασισμένος στον Προτασιακό Λογισμό, με αποδεκτή είσοδο δεδομένων να αποτελεί η μορφή αρχείων .cnf, για τα οποία έγινε αναφορά σε προηγούμενα κεφάλαια, με κάποιες όμως τροποποιήσεις, ώστε να δέχονται την ανάθεση βαρών σε περιορισμούς. Απαιτεί τη χρήση λειτουργικού συστήματος Linux/UNIX καθώς είναι διαθέσιμος ως αρχείο τύπου .o (object file), εκτελέσιμου μέσω τερματικού (terminal).

4.4.2 Αρχείο Συζευκτικής Κανονικής Μορφής με Βάρη [17]

Οι επιλυτές MAXSAT δέχονται ως είσοδο αρχεία CNF, τα οποία όμως τυγχάνουν τροποποιήσεων ώστε να υποστηρίζουν ανάθεση βαρών σε περιορισμούς. Επομένως, τα αρχεία αυτά διέπονται από τα ίδια χαρακτηριστικά που περιγράφονται στο Κεφάλαιο 3 (βλ. υποκεφάλαιο 3.2.2) για τα αρχεία CNF, με κάποια επιπλέον στοιχεία.

Μέγιστο Δυνατό Επιτρεπτό Βάρος (Maximum Possible Weight): Στην αρχή του αρχείου, στη γραμμή όπου ορίζονται τα στοιχεία του προβλήματος (γραμμή που ξεκινά με το χαρακτήρα p), καθορίζουμε και το μέγιστο βάρος που μπορεί να ανατεθεί σε κάποιο περιορισμό. Έτσι, για τα Weighted CNF αρχεία, η γραμμή αυτή έχει ως εξής:

$$\boldsymbol{p\ Είδος Αρχείου\ \#Μεταβλητών\ \#Όρων Συζεύξεων\ Μέγιστο Βάρος}$$

Στην περίπτωση μας ισχύει: $\boldsymbol{Είδος Αρχείου} = wcnf$, το οποίο υποδηλώνει ότι το τρέχον αρχείο περιέχει ένα πρόβλημα Προτασιακής Ικανοποιησιμότητας, δοσμένο σε μορφή CNF, με βάρη.

Σημαντικότητα Περιορισμού – Ισχυροί/Χαλαροί Περιορισμοί (Hard/Soft Clauses): Υπάρχουν περιορισμοί οι οποίοι είναι απαραίτητο να ικανοποιηθούν, καθώς είναι χαρακτηριστικοί του προβλήματος. Αυτοί οι περιορισμοί ονομάζονται «Ισχυροί» (Hard) και τους ανατίθεται το μέγιστο δυνατό βάρος ώστε να καθίσταται σημαντική η ικανοποίηση τους. Έτσι, οι επιλυτές δίνουν μεγαλύτερη έμφαση σε αυτούς.

Απ' την άλλη έχουμε περιορισμούς οι οποίοι δύναται να τύχουν χαλάρωσης, ώστε να επιλυθεί το πρόβλημα ακόμα και αν μερικοί από αυτούς δεν είναι δυνατόν να ικανοποιηθούν. Ονομάζονται Χαλαροί (Soft) περιορισμοί και τους ανατίθεται αρκετά χαμηλή τιμή βάρους (σε σχέση με το μέγιστο δυνατό βάρος) ώστε να εννοηθεί η μειωμένη σημαντικότητα τους έναντι των ισχυρών περιορισμών. Έτσι οι επιλυτές, εάν βρεθούν στη δύσκολη θέση του να πρέπει να χαλαρώσουν κάποιο περιορισμό ώστε το πρόβλημα να γίνει Ικανοποιήσιμο, θα το πράξουν χαλαρώνοντας κάποιο χαλαρό περιορισμό, πρώτα. Αυτή η ομάδα περιορισμών συνήθως περιγράφει επιθυμία για βελτιστοποίηση προβλήματος (π.χ. μέγιστη/ελάχιστη ανάθεση κόμβων σε κάποιο σύνολο) και όχι περιορισμούς που πηγάζουν από ορισμούς, καθώς δυνατότητα χαλάρωσης περιορισμών ορισμού θα σήμαινε παραβίαση αναγκαίων και απαραίτητων προϋποθέσεων.

Βάρος Περιορισμού (Clause Weight): Τα βάρη των περιορισμών, στα αρχεία Weighted CNF εισάγονται στην αρχή της γραμμής του περιορισμού στον οποίο ανήκουν. Διαχωρίζουμε με κενό το βάρος, από την υπόλοιπη πρόταση, ώστε να γνωρίζει ο επιλυτής ότι το πρώτος χαρακτήρας κάθε γραμμής είναι το βάρος της πρότασης που δηλώνεται στην γραμμή αυτή.

Π.χ. Για Μέγιστο Δυνατό Επιτρεπτό Βάρος = 100

- 1 −4 −1 0
  «Με σημαντικότητα 1, επέλεξε είτε τη μεταβλητή 4, είτε τη μεταβλητή 1, αλλά όχι και τις δύο μαζί»
  → Βάρος = 1 → Χαλαρός Περιορισμός

- 100 −4 −2 0
  «Με σημαντικότητα 100, επέλεξε είτε τη μεταβλητή 4, είτε τη μεταβλητή 2, αλλά όχι και τις δύο μαζί»
  → Βάρος = 100 = Μέγιστο Δυνατό Επιτρεπτό Βάρος → Ισχυρός Περιορισμός

4.5 Προσέγγιση #1: Εξαγωγή Πυρήνα Προτάσεων [14, 17, 22]

4.5.1 Κίνητρο [14, 17, 22]

Μία μέθοδος χειρισμού των Μη-Ικανοποιήσιμων προβλημάτων «Ευσταθούς Επέκτασης» είναι η εξαγωγή ενός ελάχιστου υποσυνόλου των περιορισμών τους, των οποίων η αφαίρεση τους από τη Βάση Γνώσης θα οδηγήσει το πρόβλημα σε κατάσταση Ικανοποιησιμότητας. Για χάριν συντομίας θα ονομάσουμε την τεχνική αυτή «Εξαγωγή Πυρήνα Προτάσεων» (Extraction of Core of Clauses).

Για την εξεύρεση των ελάχιστων δυνατών προτάσεων/περιορισμών (clauses) που προκαλούν την Μη-Ικανοποιησιμότητα του προβλήματος, θα πρέπει πρώτα να σκεφτούμε ποια πτυχή του προβλήματος προκαλεί περιορισμούς που είναι αρκετά περιοριστικοί, ώστε να αποτελούν τον παράγοντα που θέτει κάποια προβλήματα ως Μη- Ικανοποιήσιμα.

Στην δική μας περίπτωση, οι περιορισμοί προσπαθούν να προσδιορίσουν μία ομάδα που να εμπίπτει στον ορισμό της «Ευσταθούς Επέκτασης», επομένως είναι υπεύθυνοι για τον διαχωρισμό και την ομαδοποίηση επιχειρημάτων. Το γεγονός που άμεσα θέτει ως αντίπαλα δύο επιχειρήματα είναι η επίθεση του ενός εναντίον του άλλου, δηλαδή (πιο αφαιρετικά) η ύπαρξη ***ακμής*** μεταξύ τους.

Λαμβάνοντας υπ’ όψιν αυτή τη σκέψη, μπορούμε να φανταστούμε ότι αφαιρώντας συγκεκριμένους περιορισμούς από το πρόβλημα, οι οποίοι περιγράφουν την ύπαρξη και την ιδιότητα ακμών, το πρόβλημα θα απαλλαχθεί από αυτά τα στοιχεία που το καθιστούν «Μη- Ικανοποιήσιμο».

4.5.2 Ιδέα [14, 17, 22]

Μπορούμε, αρχικά, να προσπαθήσουμε να επιλύσουμε το πρόβλημα μέσω του Lingeling. Το αποτέλεσμα του θα μας ενημερώσει εάν το αρχικό πρόβλημα είναι Ικανοποιήσιμο ή όχι.

Το ίδιο CNF αρχείο, το οποίο περάσαμε ως είσοδο στον Lingeling, μπορούμε να το τροποποιήσουμε κατάλληλα προσθέτοντας βάρη στους περιορισμούς, ώστε να το περάσουμε ως είσοδο στον Maxino.

Πιο συγκεκριμένα, επειδή επιθυμούμε να ανιχνεύσουμε ακμές οι οποίες αποτελούν εμπόδιο στην Ικανοποιησιμότητα του προβλήματος, θα πρέπει να χαλαρώσουμε τους περιορισμούς που υποδηλώνουν την ύπαρξη ακμών (δηλαδή συσχετίσεις επίθεσης μεταξύ επιχειρημάτων), κρατώντας ισχυρούς όλους τους υπόλοιπους περιορισμούς.
Με αυτή την προσέγγιση, προκαλούμε τον Maxino να λύσει το πρόβλημα που περιγράφεται στο αρχείο, είτε αυτό είναι Ικανοποιήσιμο, είτε όχι, υποκύπτοντας στις ελάχιστες δυνατές χαλαρώσεις περιορισμών κάθε φορά, και οι οποίες θα εφαρμόζονται κατά προτεραιότητα, πάνω σε περιορισμούς ακμών, αρχικά.

Σκοπός μας είναι να βρούμε ποιους εξ’ αυτών των περιορισμών χαλάρωσε ο Maxino, ώστε να γνωρίζουμε ποιες ακμές αγνοήθηκαν. Οι περιορισμοί αυτοί θα αποτελέσουν τον ***«Πυρήνα Προτάσεων»***.

Ύπαρξη ακμής μεταξύ δύο κόμβων, A και B, υποδηλώνεται μέσω των περιορισμών της μορφής:

–A –B 0

➔ Αντίστοιχος Προτασιακός Περιορισμός: $\neg A \vee \neg B$

Αυτό που επιδιώκουμε είναι να ανακαλύψουμε ποια επιχειρήματα είναι ταυτόχρονα μέλη της Ευσταθούς Επέκτασης, σύμφωνα με τη λύση που μας παρέχει ο Maxino, ενώ παράλληλα υφίσταται ο πάρα πάνω μεταξύ τους περιορισμός στο αρχείο εισόδου. Στην περίπτωση αυτή έχουμε *αντίφαση*, και αυτό σημαίνει ότι ο εν λόγω περιορισμός «αγνοήθηκε» (χαλάρωση περιορισμών), για σκοπούς προσεγγιστικής επίλυσης του προβλήματος, από τον επιλυτή.

Η αφαίρεση του προαναφερθέντος περιορισμού είναι αντίστοιχη με την αφαίρεση της μεταξύ-των-δυο-επιχειρημάτων ακμής απ’ τον γράφο. Έτσι εξαλείφεται η μοναδική ικανή συνθήκη που είχε την δυνατότητα να θέσει το πρόβλημα ως Μη-Ικανοποιήσιμο.

Για επαλήθευση, μπορούμε να τρέξουμε το πρόβλημα μέσω του Lingeling, εφόσον πρώτα διαγράψουμε από το αρχείο εισόδου τους περιορισμούς που έτυχαν χαλάρωσης (δηλαδή αυτούς που επιστρέφονται ως «Πυρήνας Προτάσεων») καθώς επίσης και όλα τα στοιχεία που αναφέρονται σε βάρη (π.χ. βάρη περιορισμών, μέγιστο βάρος κ.λπ.).

4.5.3 Αλγόριθμος [14, 17, 22]

1) Δημιουργούμε το .cnf αρχείο που περιγράφει το πρόβλημα

2) Δίνουμε το αρχείο του βήματος 1 ως είσοδο στον Lingeling και τρέχουμε τον επιλυτή. Το αποτέλεσμα θα μας ενημερώσει εάν το αρχικό πρόβλημα είναι Ικανοποιήσιμο ή όχι

3) Παίρνουμε το αρχείο που εισήχθηκε στο Lingeling, αλλάζουμε το είδος του από «cnf» σε «wcnf», δηλώνουμε ως μέγιστο βάρος την τιμή 1000 και αναθέτουμε βάρη στους περιορισμούς, ως εξής:

i) Για κάθε περιορισμό ακμής αναθέτουμε βάρος 1 (χαλαρός περιορισμός)
Ανανεωμένη Μορφή Περιορισμών Ακμής: **1 –A –B 0**

ii) Για κάθε άλλο περιορισμό αναθέτουμε βάρος 1000 (ισχυρός περιορισμός)
Ανανεωμένη Μορφή Περιορισμών Συμμετοχής: **1000 A attA$_1$ … 0**

Εισάγουμε το ανανεωμένο αρχείο στον Maxino και τρέχουμε τον επιλυτή

4) Από το μοντέλο λύσης που προκύπτει από τον Maxino, απομονώνουμε τα επιχειρήματα/μέλη της Ευσταθούς Επέκτασης. Δημιουργούμε συμβολοσειρές για όλους, τους μεταξύ τους, πιθανούς συνδυασμούς που αντιπροσωπεύουν το δυαδικό περιορισμό ύπαρξης ακμής (1 –A –B 0), αποφεύγοντας τη δημιουργία διπλότυπων, όπως ακολούθως:

Π.χ. $\textit{Ευσταθής Επέκτασης} = \{1, 2, 5\}$

$\rightarrow \textit{Παραγόμενοι Περιορισμοί} = \{$ «$1\ -1\ -2\ 0$», «$1\ -1\ -5\ 0$», «$1\ -2\ -5\ 0$», «$1\ -2\ -1\ 0$», «$1\ -5\ -1\ 0$», «$1\ -5\ -2\ 0$» $\}$

5) Ελέγχουμε το αρχείο εισόδου ώστε να βρούμε, ποιους από αυτούς τους περιορισμούς που παράχθηκαν στο βήμα 4, περιέχει. Θέτουμε ως μέλη του Πυρήνα Προτάσεων όσους από αυτούς περιέχονται στο αρχείο

6) Επιστρέφουμε τον Πυρήνα Προτάσεων

7) Προαιρετικά, αφαιρούμε τους περιορισμούς του Πυρήνα Προτάσεων από το αρχείο εισόδου του Maxino, ρυθμίζουμε τα στοιχεία του αρχείου, αφαιρούμε όλα τα βάρη, αλλάζουμε το είδος του αρχείου από «wcnf» σε «cnf» και επιλύουμε το πρόβλημα που περιγράφει το ενημερωμένο αρχείο, μέσω Lingeling. Θα παρατηρήσουμε ότι το πρόβλημα πλέον είναι Ικανοποιήσιμο και επιστρέφεται κανονικά το μοντέλο λύσης.

4.5.4 Παράδειγμα διαδικασίας

ΒΗΜΑ 1:

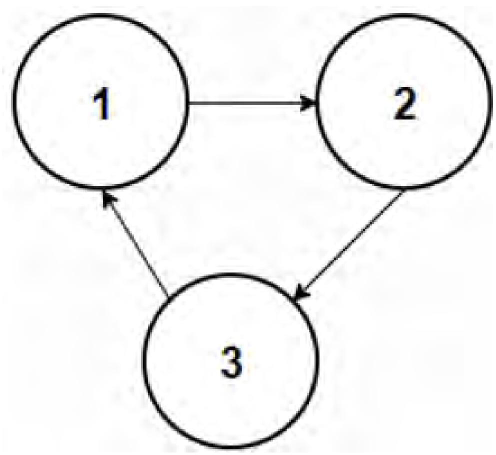
Πίνακας Γειτνίασης Γράφου

010

001

100

Γραφική Αναπαράσταση



Περιορισμοί

$$(\neg 2 \vee \neg 1) \wedge (\neg 3 \vee \neg 1) \wedge (\neg 3 \vee \neg 2) \wedge (1 \vee 3) \wedge (2 \vee 1) \wedge (3 \vee 2)$$

Αρχείο CNF

```
p cnf 3 6
-2 -1 0
-3 -1 0
-3 -2 0
1 3 0
2 1 0
3 2 0
```

Σχήμα 4.5.4.1 Μη-Ικανοποιήσιμο Πρόβλημα

ΒΗΜΑ 2:

Κλήση Lingeling για επίλυση του προβλήματος – Συλλογή Αποτελεσμάτων

```
s UNSATISFIABLE
```

ΒΗΜΑ 3:

Ενημέρωση του CNF αρχείου εισόδου για ανάθεση βαρών

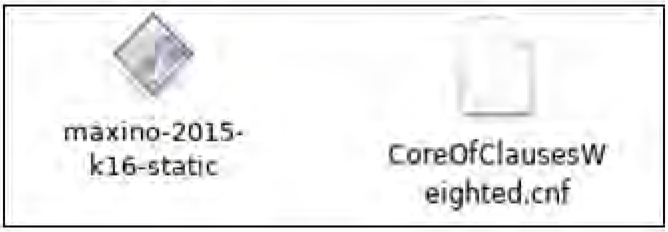
```
c LingelingInput.cnf
p cnf 3 6
-2 -1 0
-3 -1 0
-3 -2 0
1 3 0
2 1 0
3 2 0
```

```
c CoreOfClausesWeighted.cnf
p wcnf 3 6 1000
1 -2 -1 0
1 -3 -1 0
1 -3 -2 0
1000 1 3 0
1000 2 1 0
1000 3 2 0
```

Σχήμα 4.5.4.2: Κλήση Maxino και Επίλυση προβλήματος

```
./maxino-2015-k16-static CoreOfClausesWeighted.cnf
```

```
c ub 3
o 1
c ub 1
o 1
s OPTIMUM FOUND
v 1 -2 3
```



Σχήμα 4.5.4.3: Αποτελέσματα Maxino

ΒΗΜΑ 4:

Εξαγωγή Ευσταθούς Επέκτασης και δημιουργία συμβολοσειρών περιορισμών ακμής:

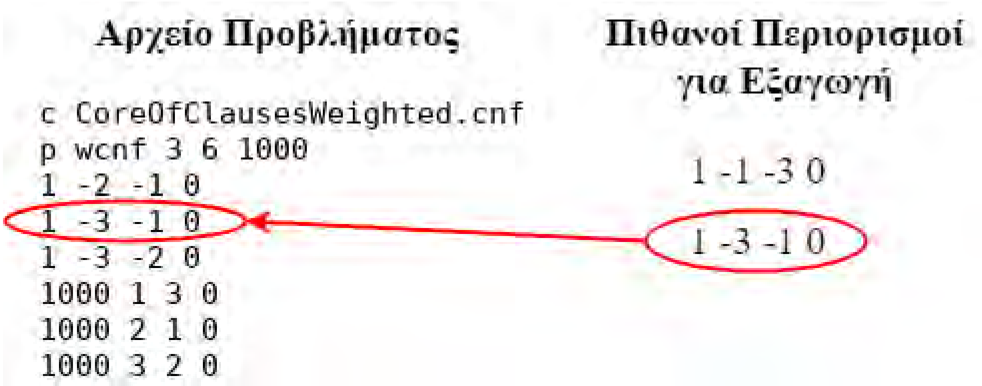
Ευσταθής Επέκταση = {1, 3}

Δυνατοί Περιορισμοί Ακμής μεταξύ επιχειρημάτων Ευσταθούς Επέκτασης = {«1 -1 -3 0», «1 -3 -1 0»}

ΒΗΜΑ 5:

Όσοι περιορισμοί υφίστανται, τόσο στο σύνολο συμβολοσειρών που δημιουργήσαμε (Βήμα 4), όσο και στο αρχείο του προβλήματος, προστίθενται στον Πυρήνα Προτάσεων.

Εάν όντως υπάρχουν τέτοιοι περιορισμοί που να συνδέουν δύο επιχειρήματα, μέλη της Ευσταθούς Επέκτασης, τότε σημαίνει ότι οι αντίστοιχες ακμές ήταν αυτές που προκαλούσαν τη Μη-Ικανοποιησιμότητα του προβλήματος, συνεπώς οι περιορισμοί αυτοί, ήταν εκείνοι που αγνοήθηκαν από τον Maxino για σκοπούς προσεγγιστικής επίλυσης.



Σχήμα 4.5.4.4: Εξαγωγή Πυρήνα Περιορισμών

ΒΗΜΑ 6:

Επιστροφή Πυρήνα Προτάσεων

`UNSAT CORE OF CLAUSES: [1 -3 -1 0]`

ΒΗΜΑ 7 (επαλήθευση):

Διαγράφουμε τους περιορισμούς του Πυρήνα Προτάσεων από το αρχείο και διορθώνουμε τον αριθμό προτάσεων (number of clauses) στα στοιχεία του αρχείου.

Ακολούθως αφαιρούμε από το αρχείο εισόδου, όλα τα στοιχεία που αναφέρονται στην ύπαρξη βαρών (μετατροπή είδους αρχείου από wcnf σε cnf, διαγραφή μέγιστου βάρους και βαρών περιορισμών).

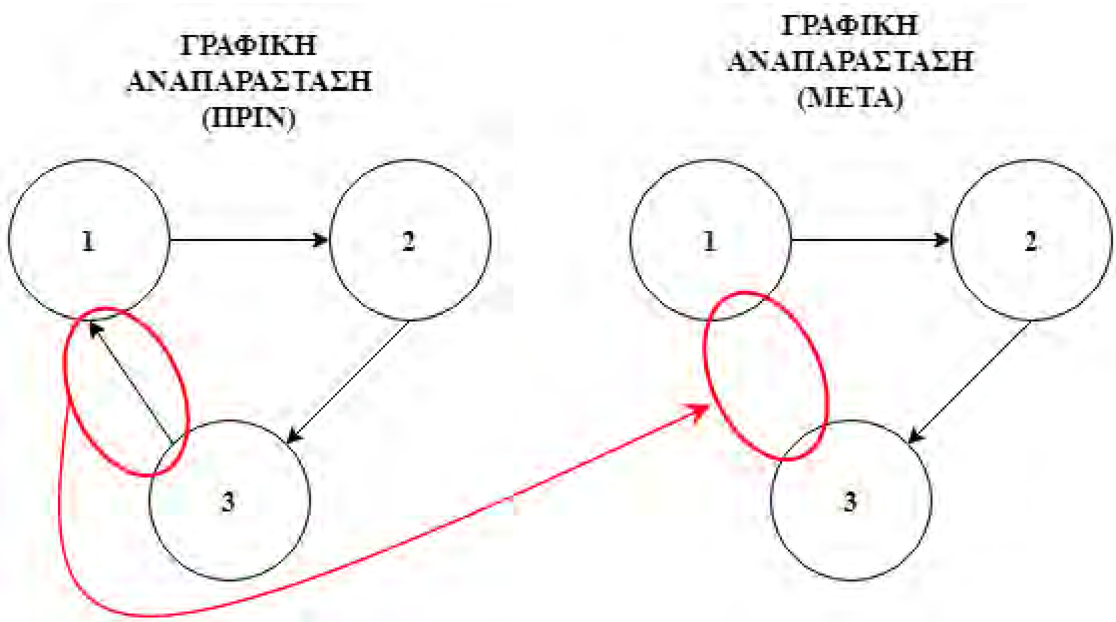
1) Αρχείο Αρχικού Προβλήματος (wcnf)

```
c CoreOfClausesWeighted.cnf
p wcnf 3 6 1000
1 -2 -1 0
1 -3 -1 0
1 -3 -2 0
1000 1 3 0
1000 2 1 0
1000 3 2 0
```

2) Αρχείο Προβλήματος (cnf με Εξαγωγή Πυρήνα Προτάσεων)

```
c CoreOfClausesWeighted.cnf
p cnf 3 5 ...
... -2 -1 0
... -3 -2 0
... 1 3 0
... 2 1 0
... 3 2 0
```

Σχήμα 4.5.4.5: Αλλαγές στο αρχείο WCNF



Σχήμα 4.5.4.6: Αλλαγές στην Γραφική Αναπαράσταση

Αποτέλεσμα Lingeling για το ενημερωμένο αρχείο εισόδου, μετά την αφαίρεση του Πυρήνα Προτάσεων από το σύνολο των περιορισμών του προβλήματος:

```
./lingeling CoreOfClausesWeighted.cnf
```

```
s SATISFIABLE
v 1 -2 3 0
```

Σχήμα 4.5.4.7: Αποτελέσματα Lingeling μετά από εξαγωγή Πυρήνα Περιορισμών

Παρατηρήσεις: Το πρόβλημα έγινε Ικανοποιήσιμο (SATISFIABLE) και επιστρέφει ένα μοντέλο επίλυσης με την ανάθεση των κατάλληλων τιμών σε μεταβλητές. Αυτό συμβαίνει διότι πλέον το επιχείρημα 3 δεν επιτίθεται στο επιχείρημα 1, και έτσι τα δύο τους μπορούν να ομαδοποιηθούν ώστε να αντιμετωπίσουν το απειλητικό επιχείρημα 2, το οποίο επιτίθεται στο επιχείρημα 3 αλλά μένει απροστάτευτο από την επίθεση του επιχειρήματος 1.

Αποτέλεσμα:
Σύνολο Ευσταθούς Επέκτασης = {1, 3}
Σύνολο Πυρήνα Προτάσεων = {attacks(3,1)}

Σχόλια: Με τους ελάχιστους δυνατούς συμβιβασμούς φτάσαμε στην καλύτερη δυνατή λύση. Στην προκειμένη περίπτωση, εφόσον δεν υπάρχει λύση που να ικανοποιεί όλους τους $\boldsymbol{N}$ αρχικούς περιορισμούς, η επόμενη καλύτερη, συμβιβαστική λύση είναι αυτή που ικανοποιεί σχεδόν όλους τους $\boldsymbol{N}$ αρχικούς περιορισμούς, εκτός από έναν. Δηλαδή ικανοποιούνται $\boldsymbol{N-1}$ περιορισμοί.

4.5.5 Σημαντικές Σημειώσεις [14]

Ο πληθικός αριθμός του «Ελάχιστου Πυρήνα Προτάσεων» ενδέχεται να διαφέρει από πρόβλημα σε πρόβλημα. Αυτό σημαίνει ότι υπάρχουν προβλήματα των οποίων η συμβιβαστική λύση απαιτεί την εξαγωγή περισσότερων από μίας πρότασης περιορισμού. Το εν λόγω γεγονός υποδηλώνει ότι πολλαπλές (και όχι μόνο μία) ακμές, πιθανόν να προκαλούν την Μη-Ικανοποιησιμότητα του προβλήματος, δημιουργώντας έτσι την ανάγκη αφαίρεσης περισσότερων στοιχείων, ώστε να μπορεί να γίνει το πρόβλημα Ικανοποιήσιμο.

Τετριμμένα, τα Ικανοποιήσιμα προβλήματα (δηλαδή αυτά των οποίων όλοι οι αρχικοί περιορισμοί ήταν δυνατόν να ικανοποιηθούν) έχουν ως Ελάχιστο Πυρήνα Προτάσεων το κενό σύνολο (Core of Clauses = { }). Το γεγονός αυτό οφείλεται στο ότι, χωρίς καν την αφαίρεση οποιουδήποτε από τους περιορισμούς, υπάρχει ήδη κατάλληλο μοντέλο ανάθεσης τιμών για τις μεταβλητές τους.

4.6 Προσέγγιση #2: Εξαγωγή Πυρήνα Επιχειρημάτων

4.6.1 Κίνητρο [14]

Μία δεύτερη προσέγγιση στο χειρισμό της Μη-Ικανοποιησιμότητας ενός προβλήματος, η οποία επίσης βασίζεται στην αφαίρεση στοιχείων από το αρχικό πρόβλημα, είναι η εξαγωγή του Πυρήνα Επιχειρημάτων (Core of Arguments).

Όσον αφορά την γραφική αναπαράσταση, η διαδικασία αυτή αντιστοιχεί στην αφαίρεση ***κόμβων*** από το γράφο που περιγράφει το πρόβλημα.

Οι συσχετίσεις επίθεσης μεταξύ επιχειρημάτων είναι αυτές που δύναται να θέσουν ένα πρόβλημα ως Μη-Ικανοποιήσιμο, καθώς περιορίζουν τη συμμετοχή επιχειρημάτων στο σύνολο της Ευσταθούς Επέκτασης.

Εντοπίζοντας τις «προβληματικές» αυτές συσχετίσεις, μπορούμε αντί να αφαιρέσουμε τις ίδιες από το πρόβλημα, να αφαιρέσουμε κάποιον από τους κόμβους/ επιχειρήματα που τις απαρτίζουν. Έτσι θα εξαλείψουμε τα επιχειρήματα, των οποίων η συμμετοχή στο πρόβλημα προκαλεί την Μη-Ικανοποιησιμότητά του.

4.6.2 Ιδέα [14]

Η έννοια της εξαγωγής επιχειρημάτων δημιουργεί νέες διαστάσεις στο μέχρι τώρα γνωστό πρόβλημα Ικανοποιησιμότητας της Ευσταθούς Επέκτασης. Πιο συγκεκριμένα, απαιτεί την ύπαρξη ενός επιπλέον συνόλου, μέσω του οποίου θα αναγνωρίζουμε ποια είναι τα επιχειρήματα τα οποία τίθενται εκτός προβλήματος, προκειμένου να επιτευχθεί η Ικανοποιησιμότητα.

Επομένως, δεν αναφερόμαστε πλέον σε απλή συμμετοχή/ μη-συμμετοχή επιχειρημάτων στο σύνολο της Ευσταθούς Επέκτασης, αλλά και σε συμμετοχή/ μη-συμμετοχή επιχειρημάτων στον ίδιο τον ορισμό του προβλήματος.

Βάσει των πάρα πάνω σκέψεων, προκύπτουν τα ακόλουθα σύνολα επιχειρημάτων με τις εξής ιδιότητες:

1) Σύνολο Α: **Συμμετοχή** στην Ευσταθή Επέκταση

2) Σύνολο Β: **Συμμετοχή** στο πρόβλημα

Μπορούμε να φανταστούμε το πρόβλημα μας ως ένα πρόβλημα Μέγιστης Ικανοποιησιμότητας (MAXSAT). Μέσω αυτού του είδους προβλημάτων, μπορούμε να ορίσουμε την έννοια της «μέγιστης συμμετοχής επιχειρημάτων στο πρόβλημα» (άρα και συμμετρικά, της «ελάχιστης εξαγωγής επιχειρημάτων από το πρόβλημα»), με τη χρήση χαλαρών περιορισμών (soft clauses).

Για την εν λόγω προσέγγιση θα χρησιμοποιηθεί ο MAXSAT επιλυτής Maxino, στον οποίο θα παρέχεται ένα αρχείο .cnf με βάρη (weighted CNF file) και θα επιστρέφει τη λύση του προβλήματος που περιγράφεται εντός του αρχείου.

4.6.3 Μεταβλητές

Για κάθε επιχείρημα θα χρειαστούν 2 διαφορετικές μεταβλητές, μέσω των οποίων μοντελοποιούμε την συμμετοχή/ μη-συμμετοχή των επιχειρημάτων, στα 2 σύνολα που προαναφέρθηκαν πάρα πάνω.

Οι MAXSAT επιλυτές αναγνωρίζουν τις μεταβλητές ως ακέραιους αριθμούς, μεγαλύτερους του 0. Θα απονέμουμε 2 διαφορετικούς αριθμούς, άνισους του 0, σε κάθε επιχείρημα ως ακολούθως:

Για πρόβλημα με $\boldsymbol{k}$ *μεταβλητές (επιχειρήματα), για κάθε επιχείρημα με δείκτη* $\boldsymbol{i}$*, ισχύει:*

1. *Μεταβλητή Συμμετοχής του i στην Ευσταθή Επέκταση:*

$$\boldsymbol{i}$$

2. *Μεταβλητή Συμμετοχής του i στο πρόβλημα:*

$$\boldsymbol{i} + \boldsymbol{k}$$

*Συμμετρικά, η **άρνηση** των μεταβλητών αυτών δηλώνει την **αντίθετη συμπεριφορά όσον αφορά τη συμμετοχή/ απουσία** στο αντίστοιχο σύνολο που περιγράφεται, για την κάθε μεταβλητή.*

Π.χ.

$\textit{Σύνολο Επιχειρημάτων στο πρόβλημα} = 4$

Για το επιχείρημα 3 ισχύουν οι πάρα κάτω μεταβλητές:

3 → Το επιχείρημα 3 είναι μέλος της Ευσταθούς Επέκτασης

-3 → Το επιχείρημα 3 ΔΕΝ είναι μέλος της Ευσταθούς Επέκτασης

7 → Το επιχείρημα 3 βρίσκεται εντός του ορισμού του προβλήματος

-7→ Το επιχείρημα 3 ΔΕΝ βρίσκεται εντός του ορισμού του προβλήματος και βρίσκεται εντός του Πυρήνα Επιχειρημάτων

4.6.4 Περιορισμοί [14, 17, 22]

Με βάση τη δήλωση της τιμής 1000, ως μέγιστο βάρος περιορισμών στο wcnf αρχείο εισόδου, έχουμε τους πάρα κάτω περιορισμούς:

4.6.4.1 Ισχυροί Περιορισμοί [17, 22]

1. **Σύνδεση των 2 Συνόλων:** Ο περιορισμός αυτός δηλώνει την υποχρέωση της συμμετοχής, του κάθε μέλους της Ευσταθούς Επέκτασης, και στον ορισμό του προβλήματος. Δεν νοείται η ύπαρξη συνόλου Ευσταθούς Επέκτασης, που να περιέχει επιχειρήματα που βρίσκονται εκτός του προβλήματος.

   Για πρόβλημα με k συνολικά επιχειρήματα, για κάθε επιχείρημα i ισχύει:

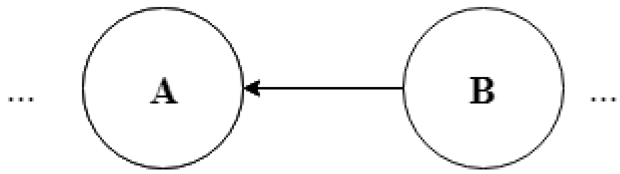
   - **Περιορισμός (σε Προτασιακή Λογική):**

$$i \rightarrow (i+k)$$
$$\equiv$$
$$-i \vee (i+k)$$

- **Αντίστοιχη WCNF Μορφή (παραλείπονται οι παρενθέσεις στο αρχείο):**

$$\mathbf{1000} - \boldsymbol{i}\ (\boldsymbol{i} + \boldsymbol{k})\ \mathbf{0}$$

2. **Περιορισμοί Συσχετίσεων Επίθεσης** Εάν κάποιο επιχείρημα δέχεται επίθεση από κάποιο άλλο επιχείρημα, τότε δεν είναι δυνατόν να αποτελούν, και τα δύο, μέλη της Ευσταθούς Επέκτασης.



Σχήμα 4.6.4.1.1: Συσχέτιση Επίθεσης

Για κάθε συσχέτιση επίθεσης μεταξύ δύο επιχειρημάτων, όπου το επιχείρημα A δέχεται επίθεση από το επιχείρημα B, ισχύουν οι πάρα κάτω περιορισμοί:

- **Περιορισμός (σε Προτασιακή Λογική):**

$$(A \rightarrow \neg B)\ \wedge\ (B \rightarrow \neg A)$$
$$\equiv$$
$$\neg A\ \vee\ \neg B$$

- **Περιορισμός (σε μορφή WCNF Αρχείου):**

$$\mathbf{1000} - \boldsymbol{A} - \boldsymbol{B}\ \mathbf{0}$$

- **Επεξήγηση:**

  Εάν το A είναι μέλος της Ευσταθούς Επέκτασης, το B δεν είναι ΚΑΙ εάν το B είναι μέλος της Ευσταθούς Επέκτασης, το A δεν είναι.

  $$\equiv$$

  Τα επιχειρήματα A και B δεν είναι και τα δύο μαζί, ταυτόχρονα, μέλη της Ευσταθούς Επέκτασης.

3. **Κριτήρια Συμμετοχής:** Τα κριτήρια συμμετοχής επιχειρημάτων στην Ευσταθή Επέκταση προσαρμόζονται στο γεγονός της ύπαρξης δύο τελικών συνόλων στην προσέγγιση αυτή. Επομένως, η δυνατότητα απουσίας ενός επιχειρήματος από το σύνολο της Ευσταθούς Επέκτασης δεν συνεπάγεται απαραίτητα μόνο με απλή απουσία από το σύνολο, αλλά μπορεί να σημαίνει ακόμη και πλήρη απουσία από τον ίδιο τον ορισμό του προβλήματος.

- ***Λεκτική Μορφή:***

1) Εάν το i δεν είναι μέλος της Ευσταθούς Επέκτασης, τότε ισχύει *τουλάχιστον* ένα από τα πάρα κάτω:

a. Τουλάχιστον ένα εκ των επιχειρημάτων που επιτίθενται στο i, αποτελεί μέλος της Ευσταθούς Επέκτασης

b. Το i τίθεται εντελώς εκτός προβλήματος (εισάγεται στον Πυρήνα Επιχειρημάτων)

- **Περιορισμός (σε Προτασιακή Λογική):**

Όπου: $atti_m$ = το m-ιοστό επιχείρημα που επιτίθεται στο επιχείρημα i

$$\neg i \rightarrow \neg(i+k) \lor atti_1 \lor atti_2 \lor \ldots \lor atti_n$$
$$\equiv$$
$$i \lor \neg(i+k) \lor atti_1 \lor atti_2 \lor \ldots \lor atti_n$$

Σημείωση: Στη θέση των «…», τοποθετούνται όλα τα επιχειρήματα που επιτίθονται στο επιχείρημα i, διαχωρισμένα από σύμβολο διάζευξης ('∨'). Ο δείκτης n αποτελεί το πλήθος των επιχειρημάτων που επιτίθονται στο i.

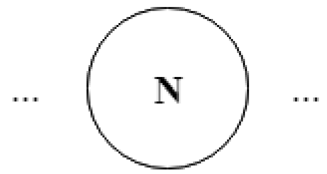
- **Περιορισμός (σε μορφή WCNF Αρχείου):**

$$1000\ i\ -(i+k)\ atti_1\ atti_2\ \ldots\ atti_n\ 0$$

Σημείωση: Στη θέση των «…», τοποθετούνται όλα τα επιχειρήματα που επιτίθονται στο επιχείρημα i, διαχωρισμένα από ένα κενό διάστημα. Ο δείκτης n αποτελεί το πλήθος των επιχειρημάτων που επιτίθονται στο i.

4. **Τετριμμένη Συμμετοχή Ουδέτερων Επιχειρημάτων :** Εάν κάποιο επιχείρημα δεν δέχεται, ούτε εξαπολύει, επίθεση από/σε οποιοδήποτε άλλο επιχείρημα, τότε τετριμμένα ανήκει στην Ευσταθή Επέκταση ή στον Πυρήνα Επιχειρημάτων

   ***ΣΗΜΕΙΩΣΗ**: Η συμμετοχή ουδέτερων επιχειρημάτων στον Πυρήνα Επιχειρημάτων αποκλείεται, μέσω της δήλωσης χαλαρών περιορισμών αργότερα, οι οποίοι ζητούν τη μέγιστη ανάθεση μη-προβληματικών επιχειρημάτων εντός του ορισμού του προβλήματος.



Σχήμα 4.6.4.1.2: Ουδέτερο Επιχείρημα

Για ΚΑΘΕ ουδέτερο επιχείρημα N, ισχύει ο πάρα κάτω μοναδιαίος περιορισμός (σε πρόβλημα με k συνολικά επιχειρήματα):

- **Περιορισμός (σε Προτασιακή Λογική):**

$$\neg N \rightarrow \neg(N+k)$$

$$\equiv$$

$$N \vee \neg(N+k)$$

- **Περιορισμός (σε μορφή WCNF Αρχείου):**

$$1000\ N\ -(N+k)\ 0$$

- **Επεξήγηση:**

  Το ουδέτερο επιχείρημα Ν είναι μέλος της Ευσταθούς Επέκτασης ή του Πυρήνα Επιχειρημάτων.

- **Σημείωση:**

  Ο εν λόγω περιορισμός προκύπτει από τον περιορισμό υπ' αριθμό 2 (Κριτήρια Συμμετοχής), εφόσον για κάθε ουδέτερο επιχείρημα ισχύει ότι, είτε το ίδιο, είτε κάποια από τα επιχειρήματα που του επιτίθενται, είναι μέλη της Ευσταθούς Επέκτασης, είτε το ίδιο ανήκει στον Πυρήνα Επιχειρημάτων. Το γεγονός όμως ότι τα ***ουδέτερα*** επιχειρήματα ***δεν δέχονται επιθέσεις***, τα κατατάσσει, αναγκαστικά, στο σύνολο της Ευσταθούς Επέκτασης ή στον Πυρήνα Επιχειρημάτων.

4.6.4.2 Χαλαροί Περιορισμοί [17, 22]

1. **Μέγιστη απουσία επιχειρημάτων από τον Πυρήνα Επιχειρημάτων:** Εφόσον επιθυμούμε την εξαγωγή των ελάχιστων δυνατών επιχειρημάτων από το πρόβλημα, αποσκοπούμε στην εύρεση του ελάχιστου δυνατού πυρήνα επιχειρημάτων. Αυτό μπορούμε να το λύσουμε συμμετρικά, ζητώντας από τον MAXSAT επιλυτή, μέσω χαλαρών περιορισμών, να αναθέσει τα περισσότερα δυνατά επιχειρήματα ΕΝΤΟΣ του προβλήματος.

   Οι χαλαροί περιορισμοί θα ικανοποιηθούν στο μέγιστο, και σε περίπτωση που θα χρειαστεί να γίνουν οποιεσδήποτε χαλαρώσεις για να επιλυθεί το πρόβλημα, ο επιλυτής θα αγνοήσει κάποιους εξ' αυτών των περιορισμών, με αποτέλεσμα να τεθούν εκτός προβλήματος, όσο το δυνατό γίνεται, λιγότερα επιχειρήματα.

   Για πρόβλημα με $\boldsymbol{k}$ συνολικά επιχειρήματα, για κάθε επιχείρημα i ισχύει:

   - **Περιορισμός (σε Προτασιακή Λογική):**

     $$i + k$$

   - **Περιορισμός (σε μορφή WCNF Αρχείου,** *παραλείπονται οι παρενθέσεις στο αρχείο***):***

     $$1\ (i + k)\ 0$$

4.6.5 Αλγόριθμος [14]

1) Δημιουργούμε το .cnf αρχείο που περιγράφει το πρόβλημα που θέλουμε να λύσουμε, χρησιμοποιώντας τους πάρα πάνω περιορισμούς με βάρη.

2) Δίνουμε το αρχείο του βήματος 1 ως είσοδο στον Maxino και τρέχουμε τον επιλυτή.

3) Από το μοντέλο λύσης που προκύπτει από τον Maxino εξάγουμε τις μεταβλητές ανά σύνολο. Για πρόβλημα με k συνολικά επιχειρήματα ισχύει ότι:

- Οι πρώτες k μεταβλητές του μοντέλου δείχνουν την κατάσταση συμμετοχής επιχειρημάτων στο σύνολο της Ευσταθούς Επέκτασης
- Οι τελευταίες k μεταβλητές του μοντέλου δείχνουν την κατάσταση παρουσίας επιχειρημάτων στον ορισμό του προβλήματος

4) Για το πρόβλημα k συνολικών επιχειρημάτων, αναθέτουμε στο κατάλληλο σύνολο ***το κάθε επιχείρημα $\boldsymbol{a}$***, ως εξής:

- **<u>Ευσταθής Επέκταση</u>**
  ➔ Εάν υπάρχει η τιμή a στο μοντέλο (και ΟΧΙ η $-a$)

- **<u>Πυρήνας Επιχειρημάτων</u>**
  ➔ Εάν υπάρχει η τιμή $-(a+k)$ στο μοντέλο (και ΟΧΙ η $(a+\kappa)$)

5) Προαιρετικά, για σκοπούς επαλήθευσης, δημιουργούμε το .cnf αρχείο του αντίστοιχου γράφου, για το απλό πρόβλημα της Ευσταθούς Επέκτασης (όπως περιγράφτηκε στο Κεφάλαιο 3). Δίνουμε το εν λόγω αρχείο ως είσοδο στον Lingeling και παίρνουμε το αποτέλεσμα. Έπειτα, αφαιρούμε τον Πυρήνα Επιχειρημάτων από τη θεωρία του πάρα πάνω προβλήματος και επιλύνουμε ξανά το πρόβλημα, βάσει της νέας, ενημερωμένης θεωρίας, μέσω Lingeling. Θα παρατηρήσουμε ότι το πρόβλημα πλέον είναι Ικανοποιήσιμο και επιστρέφεται κανονικά το μοντέλο λύσης.

***ΣΗΜΕΙΩΣΗ:** Η διαγραφή επιχειρημάτων από τον ορισμό του προβλήματος γίνεται εντός του αρχείου γράφου, μηδενίζοντας τα στοιχεία στηλών και γραμμών των οποίων οι δείκτες τους είναι δείκτες επιχειρημάτων που τέθηκαν εκτός προβλήματος.

4.6.6 Παράδειγμα διαδικασίας

ΒΗΜΑ 1:

Παίρνουμε το πρόβλημα Ευσταθούς Επέκτασης το οποίο χρησιμοποιήσαμε ως παράδειγμα και στην περίπτωση του χειρισμού μη-ικανοποιησιμότητας, μέσω εξαγωγής Πυρήνα Προτάσεων (Προσέγγιση #1, βλ. υποκεφάλαιο 4.5.4). Γνωρίζουμε ήδη ότι είναι ***Μη-Ικανοποιήσιμο.***

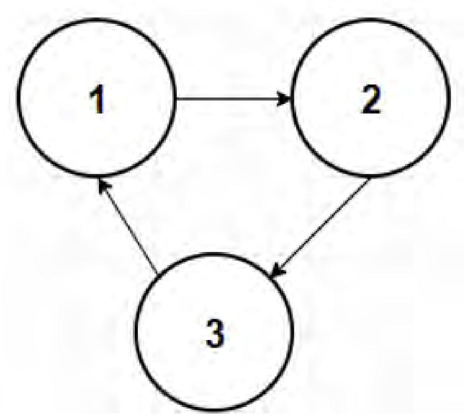
Πίνακας Γειτνίασης Γράφου

010

001

100

Γραφική Αναπαράσταση



Σχήμα 4.6.6.1 Μη-Ικανοπ. Πρόβλημα

Αντίστοιχο CNF αρχείο με βάρη

```
c UNSATCoreOfNodes.cnf
p wcnf 6 12 1000
1 4 0
1000 -1 4 0
1000 -1 -2 0
1000 -1 -3 0
1000 1 -4 3 0
1 5 0
1000 -2 5 0
1000 -2 -3 0
1000 2 -5 1 0
1 6 0
1000 -3 6 0
1000 3 -6 2 0
```

ΒΗΜΑ 2:

Κλήση Maxino για επίλυση του προβλήματος – Συλλογή Αποτελεσμάτων

```
c ub 3
o 1
c ub 1
o 1
s OPTIMUM FOUND
v -1 -2 3 4 -5 6
```

Σχήμα 4.6.6.2

ΒΗΜΑ 3:

Εξαγωγή Μοντέλου από αποτέλεσμα

Μοντέλο Λύσης = {-1, -2, 3, 4, -5, 6}

Μεταβλητές Συμμετοχής στην Ευσταθή Επέκταση = {-1, -2, 3}

Μεταβλητές Συμμετοχής στον ορισμό του προβλήματος = {4, -5, 6}

ΒΗΜΑ 4:

Ανάθεση των επιχειρημάτων στο σύνολο στο οποίο ανήκουν

```
1) STABLE EXTENSION: [3]
2) IN PROBLEM & OUT OF STABLE EXTENSION: [1]
3) UNSAT CORE of NODES/ARGUMENTS: [2]
```

Σχήμα 4.6.6.3 Αποτελέσματα Προγράμματος για Πυρήνα Επιχειρημάτων

Τα πάρα πάνω αποτελέσματα προκύπτουν από τις εξής παρατηρήσεις όσον αφορά το μοντέλο λύσης:

- **Από Μεταβλητές $\mathbf{3}$ και $\mathbf{6}$ →** το επιχείρημα 3 ανήκει στο σύνολο της Ευσταθούς Επέκτασης.
- **Από Μεταβλητές $\mathbf{4}$ και $\mathbf{-1}$ →** το επιχείρημα 1 (αφού 1 + 3 = 4) βρίσκεται εντός του προβλήματος, αλλά δεν αποτελεί μέλος της Ευσταθούς Επέκτασης, αφού στο μοντέλο δεν υπάρχει η μεταβλητή 1, ενώ υπάρχει η άρνησή της ($-\mathbf{1}$).
- **Από Μεταβλητή $\mathbf{-5}$ →** το επιχείρημα 2 (αφού 2 + 3 = 5) δεν βρίσκεται εντός του προβλήματος, άρα είναι μέλος του Πυρήνα Επιχειρημάτων.

ΒΗΜΑ 5:

Γράφος ΠΡΙΝ την αφαίρεση του Πυρήνα

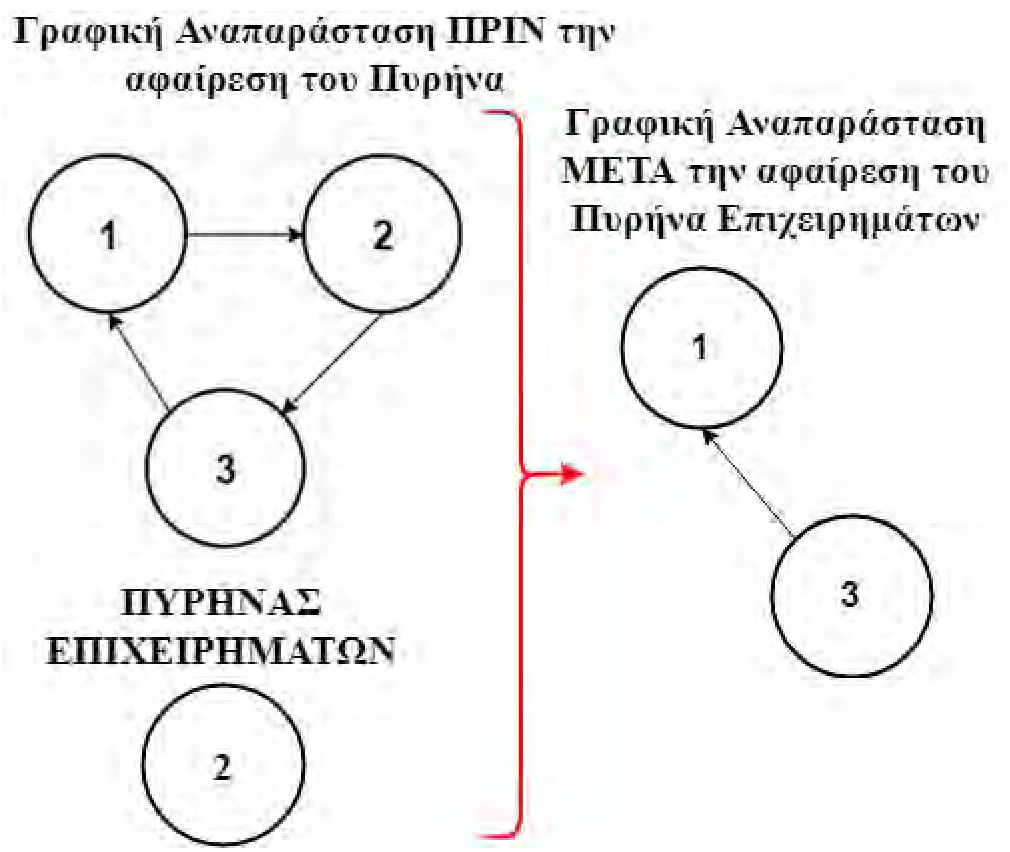
0	1	0
0	0	1
1	0	0

Γράφος ΜΕΤΑ την αφαίρεση του Πυρήνα

0	0	0
0	0	0
1	0	0

Σχήμα 4.6.6.4: Εξαγωγή Πυρήνα Επιχειρημάτων

Μηδενίσαμε τα στοιχεία της $2^{ης}$ στήλης και της $2^{ης}$ γραμμής, καθώς το επιχείρημα 2 τέθηκε εκτός προβλήματος, συνεπώς, οι συσχετίσεις στις οποίες συμμετέχει, διαγράφονται.



Σχήμα 4.6.6.5: Εξαγωγή Πυρήνα Επιχειρημάτων (Γραφικά)

Αρχείο CNF ΠΡΙΝ την αφαίρεση του Πυρήνα

```
c LingelingInput.cnf
p cnf 3 6
-2 -1 0
-3 -1 0
-3 -2 0
1 3 0
2 1 0
3 2 0
```

Αρχείο CNF ΜΕΤΑ την αφαίρεση του Πυρήνα

```
c LingelingInput.cnf
p cnf 3 4
-3 -1 0
1 3 0
2 0
3 0
```

Σχήμα 4.6.6.6: Ενημέρωση Αρχείου

Αποτέλεσμα Lingeling ΠΡΙΝ την αφαίρεση του Πυρήνα

```
s UNSATISFIABLE
```

Σχήμα 4.6.6.7 Αποτελέσματα Lingeling ΠΡΙΝ

Αποτέλεσμα Lingeling ΜΕΤΑ την αφαίρεση του Πυρήνα

```
s SATISFIABLE
v -1 2 3 0
```

Σχήμα 4.6.6.8 Αποτελέσματα Lingeling ΜΕΤΑ

***ΣΗΜΕΙΩΣΗ**: Από το μοντέλο λύσης, αγνοούμε την ύπαρξη των επιχειρημάτων που αφαιρέθηκαν από το πρόβλημα στα προηγούμενα βήματα. Εφόσον έγινε αφαίρεση όλων των δυαδικών περιορισμών στους οποίους συμμετείχαν τα επιχειρήματα του πυρήνα, δεν μας επηρεάζει η αγνόηση τους.

Παρατηρήσεις: Το πρόβλημα έγινε Ικανοποιήσιμο και επιστρέφει ένα μοντέλο επίλυσης με την ανάθεση των κατάλληλων τιμών σε μεταβλητές. Αυτό συμβαίνει διότι πλέον το επιχείρημα 2 δεν υφίσταται στο πρόβλημα. Έτσι, το επιχείρημα 3 επιτίθεται στο επιχείρημα 1, ενώ δεν μένει πλέον απροστάτευτο από την επίθεση του επιχειρήματος 2.

Αποτέλεσμα:
Σύνολο Ευσταθούς Επέκτασης = {3}
Πυρήνας Επιχειρημάτων = {2}

Σχόλια: Με τους ελάχιστους δυνατούς συμβιβασμούς (στην περίπτωση μας έγινε αφαίρεση ΜΟΝΟ ενός εκ των επιχειρημάτων), φτάσαμε στην καλύτερη δυνατή λύση.

4.6.7 Σημαντικές Σημειώσεις [14]

Ο πληθικός αριθμός του ελάχιστου «Πυρήνα Επιχειρημάτων» διαφέρει από πρόβλημα σε πρόβλημα. Αυτό σημαίνει ότι υπάρχουν προβλήματα των οποίων η συμβιβαστική λύση απαιτεί την εξαγωγή περισσότερων του ενός επιχειρήματος, από τον ορισμό τους. Αυτό υποδηλώνει ότι πολλαπλοί (και όχι μόνο ένας) κόμβοι του αντίστοιχου γράφου, πιθανόν να προκαλούν προβληματικές συσχετίσεις, άρα και την Μη-Ικανοποιησιμότητα του προβλήματος.

Τετριμμένα, τα Ικανοποιήσιμα προβλήματα Ευσταθούς Επέκτασης έχουν ελάχιστο Πυρήνα Επιχειρημάτων το κενό σύνολο (Core of Arguments = { }). Ο λόγος είναι διότι βρίσκονται ήδη σε Ικανοποιήσιμη κατάσταση, επομένως η αφαίρεση ακριβώς μηδέν επιχειρημάτων από τον ορισμό του προβλήματος θα οδηγήσει στο ίδιο αποτέλεσμα, δηλαδή την ορθή εξεύρεση ενός μοντέλου επίλυσης.

Κεφάλαιο 5

Μελέτη Ευσταθούς Επέκτασης στο Γραμμικό Προγραμματισμό

5.1 Αρχική Ιδέα [1, 10]

Ο Προτασιακός Λογισμός δεν αποτελεί το μοναδικό εργαλείο ορισμού προβλημάτων Ικανοποίησης Περιορισμών. Οι περιορισμοί που διέπουν το πρόβλημα της Ευσταθούς Επέκτασης μπορούν να οριστούν και σε μορφή γραμμικών ανισοτήτων και εξισώσεων, δίνοντας μας έτσι τη δυνατότητα να χρησιμοποιήσουμε ένα άλλο, αρκετά χρήσιμο, εργαλείο μοντελοποίησης προβλημάτων Ικανοποίησης Περιορισμών, τον Γραμμικό Προγραμματισμό.

Ο ***«Γραμμικός Προγραμματισμός»*** χαρακτηρίζεται από τη δήλωση των περιορισμών ως γραμμικές ανισότητες ή/και εξισώσεις, την αντιμετώπιση των μεταβλητών του προβλήματος ως όρους των πράξεων αυτών και την ύπαρξη μίας αντικειμενικής συνάρτησης, η οποία συσχετίζει τις μεταβλητές μεταξύ τους.

Αυτό που *επιδιώκεται* είναι η εξεύρεση ενός κατάλληλου μοντέλου ανάθεσης τιμών στις μεταβλητές ώστε, όχι μόνο να *ικανοποιούνται όλοι οι περιορισμοί*, αλλά και να *«βελτιστοποιείται» το αποτέλεσμα της αντικειμενικής συνάρτησης*. Η έννοια της βελτιστοποίησης απευθύνεται στην *ελαχιστοποίηση ή την μεγιστοποίηση* της τελικής τιμής της αντικειμενικής συνάρτησης και εξαρτάται καθαρά από τα πλαίσια του προβλήματος (π.χ. η εύρεση μέγιστου συνόλου Ευσταθούς Επέκτασης απαιτεί μεγιστοποίηση της τελικής τιμής της αντικειμενικής συνάρτησης).

Συμπληρωματική ανάλυση της τεχνικής πτυχής του Κεφαλαίου 5 υπάρχει στο Παράρτημα Α (υποκεφάλαιο Παραρτήματος Α: Α.3).

5.2 Χρήση Επιλυτών

5.2.1 Επιλυτής Ακέραιου Γραμμικού Προγραμματισμού [15]

Για τη δήλωση προβλημάτων Γραμμικού Προγραμματισμού θα χρησιμοποιηθεί το εργαλείο *Matlab*, μέσω του οποίου παρέχεται ο *επιλυτής intlinprog*. Καλείται μέσω της μεθόδου intlinprog() εντός του αρχείου που περιγράφει το πρόβλημα.

Ο συγκεκριμένος επιλυτής παρέχει τη δυνατότητα επίλυσης προβλημάτων Μικτού-Ακέραιου Γραμμικού Προγραμματισμού (Mixed-Integer Linear Programming, MILP). Τα εξής προβλήματα Γραμμικού Προγραμματισμού χαρακτηρίζονται από την επιπρόσθετη δυνατότητα ανάθεσης, τόσο ακέραιων, όσο και πραγματικών τιμών, σε μεταβλητές.

5.2.2 Λογική Επιλυτή [15]

Η *λογική* με την οποία ενεργεί ο επιλυτής ***intlinprog***, σύμφωνα με τον οδηγό τεκμηρίωσης του (https://www.mathworks.com/help/optim/ug/intlinprog.html), είναι η εξής:

$$\min_{x} \overset{1}{f^T x} \text{ subject to } \begin{cases} x(\text{intcon}) \text{ are integers} & 2 \\ A \cdot x \leq b & 3 \\ Aeq \cdot x = beq & 4 \\ lb \leq x \leq ub. & 5 \end{cases}$$

Σχήμα 5.2.2.1 Τρόπος Λειτουργίας Επιλυτή intlinprog

Επεξήγηση (βάσει πάρα πάνω υπομνήματος):

1) Προσπαθεί να *ελαχιστοποιήσει* την τιμή της αντικειμενικής συνάρτησης, για όσο το δυνατό περισσότερο το επιτρέπουν οι περιορισμοί στους οποίους υπόκειται.

2) Λαμβάνει υπ’ όψιν την επιθυμία ανάθεσης *ακέραιων* τιμών σε κάποιες από τις μεταβλητές (δύναται ακόμη και να ζητηθεί η ανάθεση ακέραιων τιμών, σε όλες τις μεταβλητές του προβλήματος).

3) Επιδιώκει να αναθέσει τιμές σε όλες τις μεταβλητές, για τις οποίες ικανοποιούνται όλοι οι γραμμικοί περιορισμοί *ανισοτήτων*, στη μορφή που παρουσιάζεται.

4) Επιδιώκει να αναθέσει τιμές σε όλες τις μεταβλητές, για τις οποίες ικανοποιούνται όλοι οι γραμμικοί περιορισμοί *εξισώσεων*, στη μορφή που παρουσιάζεται.

5) Επιδιώκει να περιορίσει τις τιμές που ανατίθενται στις μεταβλητές, ώστε να εμπίπτουν εντός κάποιων *ορίων*.

5.2.3 Αρχείο Εισόδου (Script File) [15, 23]

Τα αρχεία με κατάληξη «.m» αποτελούν την επίσημη μορφή πρότυπου αρχείου εισόδου που αποδέχεται η Matlab (Matlab Script file). Περιγράφουν ένα πρόβλημα περιορισμών το οποίο εκφράζεται σε γραμμικές ανισότητες/ εξισώσεις, μέσω της χρήσης λογισμού πινάκων (Matrices Computation). Ακολουθούν οι συνιστώσες μεταβλητές και διαδικασίες του αρχείου, των οποίων η χρήση μας επιτρέπει να μοντελοποιήσουμε κατάλληλα, προβλήματα «Ευσταθούς Επέκτασης» σε Γραμμικό Προγραμματισμό:

- ***f:*** η *αντικειμενική συνάρτηση*, για την οποία θα ζητήσουμε την βελτιστοποίησή της. Δηλώνεται ως πίνακας (matrix) και παρουσιάζει το πλήθος, αλλά και τους συντελεστές των μεταβλητών που συμμετέχουν στο πρόβλημα. Στην περίπτωση μας, αντικειμενική συνάρτηση αποτελεί το άθροισμα των μεταβλητών μας, δηλαδή των τιμών των επιχειρημάτων, ή ακόμη και η συνάρτηση f=0, αναλόγως εάν θέλουμε μέγιστο ή ελάχιστο σύνολο Ευσταθούς Επέκτασης. Οι μεταβλητές αποτελούνται από τα επιχειρήματα του προβλήματος
- ***intcon:*** διάνυσμα (vector) που υποδεικνύει ποιες εκ των μεταβλητών θα είναι *ακέραιες*. Στην περίπτωσή μας, όλες οι μεταβλητές είναι ακέραιες.
- ***A:*** πίνακας (matrix) που περιέχει τους συντελεστές των μεταβλητών που συμμετέχουν στα *αριστερά* μέρη των περιορισμών ***ανισοτήτων.***
- ***b:*** πίνακας (matrix) που περιέχει τις τιμές που βρίσκονται στα *δεξιά* μέρη των περιορισμών ***ανισοτήτων.***
- ***Aeq:*** πίνακας (matrix) που περιέχει τους συντελεστές των μεταβλητών που συμμετέχουν στα *αριστερά* μέρη των περιορισμών **εξισώσεων.**
- ***beq:*** πίνακας (matrix) που περιέχει τις τιμές που βρίσκονται στα *δεξιά* μέρη των περιορισμών **εξισώσεων.**
- ***lb:*** *κάτω φράγμα τιμών* (lower bound). Δηλώνεται ως διάνυσμα (vector). Στην περίπτωση μας, όλες οι μεταβλητές έχουν ως κάτω φράγμα την τιμή 0.
- ***ub:*** *άνω φράγμα τιμών* (upper bound). Δηλώνεται ως διάνυσμα (vector). Στην περίπτωση μας, όλες οι μεταβλητές έχουν ως κάτω φράγμα την τιμή 1.
- $[\boldsymbol{x}, \boldsymbol{fval}] = \boldsymbol{intlinprog}(\boldsymbol{f}, \boldsymbol{intcon}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{Aeq}, \boldsymbol{beq}, \boldsymbol{lb}, \boldsymbol{ub})$ ***:*** *κλήση επιλυτή intlinprog*, λαμβάνοντας υπ’ όψιν όλες τις πάρα πάνω παραμέτρους που

δηλώθηκαν, επιστρέφοντας το μοντέλο επίλυσης στη μεταβλητή x, μαζί με την τιμή της βελτιστοποιημένης αντικειμενικής συνάρτησης, στη μεταβλητή fval.

5.3 Γραμμικοί Περιορισμοί [1, 10]

Μέσα από τον ορισμό της Ευσταθούς Επέκτασης πηγάζουν οι απαραίτητοι περιορισμοί που ορίζουν αυτού του είδους προβλήματα, οι οποίοι είναι οι εξής:

4. **Ακέραιες Τιμές που Διέπονται από Δυαδικότητα:** Το πρόβλημα της Ευσταθούς Επέκτασης απαιτεί την ανάθεση επιχειρημάτων σε ένα σύνολο, επομένως χαρακτηρίζεται από την ανάθεση τιμών True ή False, ανάλογα με τη συμμετοχή/ απουσία επιχειρημάτων από το εν λόγω σύνολο. Γι’ αυτό το λόγο τα πεδία τιμών των μεταβλητών θα πρέπει να περιοριστούν ώστε να λαμβάνουν μόνο τις τιμές 1 ($= True$) και 0 ($= False$).

   Αυτό επιτυγχάνεται με τον συνδυασμό περιορισμών που θέτουν την ύπαρξη άνω και κάτω φράγματος τιμών, με άνω φράγμα να αποτελεί η τιμή 1 και κάτω φράγμα να αποτελεί η τιμή 0. Ταυτόχρονα, προκειμένου να αποφευχθεί η ανάθεση πραγματικών αριθμών, περιορίζουμε επιπλέον τα πεδία τιμών των μεταβλητών, ώστε να λαμβάνουν μόνο ακέραιες τιμές. Επομένως, προκύπτουν οι ακόλουθοι περιορισμοί:

   Για ΚΑΘΕ επιχείρημα n ισχύει:

$$\boldsymbol{n \in Z, (\text{όπου } Z\text{: } σύνολο\ ακεραίων\ αριθμών)}$$
$$\boldsymbol{n \leq 1}$$
$$\boldsymbol{n \geq 0}$$

   Οι πάρα πάνω περιορισμοί επιτυγχάνονται με τις εξής δηλώσεις (N = πλήθος μεταβλητών προβλήματος = αριθμός κόμβων γραφήματος):

   - $intcon\ = 1{:}\,N$

     → Δήλωση όλων των μεταβλητών ως ακέραιες

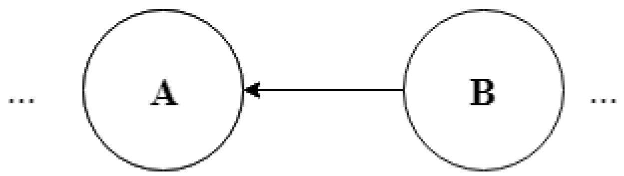
- $ub = ones(N,1)$

  → Δήλωση του αριθμού 1 ως ανώτατο όριο για τα πεδία των μεταβλητών

- $lb = zeros(N,1)$

  → Δήλωση του αριθμού 0 ως κατώτατο όριο για τα πεδία των μεταβλητών

5. **Αποκλεισμός Κοινής Συμμετοχής Αντίπαλων Επιχειρημάτων:** Εάν κάποιο επιχείρημα δέχεται επίθεση από κάποιο άλλο επιχείρημα, τότε δεν είναι δυνατόν να αποτελούν, και τα δύο, μέλη της Ευσταθούς Επέκτασης. Συνδυάζεται με τους περιορισμούς που θέτουν τα Κριτήρια Συμμετοχής (περιορισμοί υπ' αριθμόν 3).



Σχήμα 5.3.1: Συσχέτιση Επίθεσης

Για κάθε συσχέτιση επίθεσης μεταξύ δύο επιχειρημάτων, όπου το επιχείρημα A δέχεται επίθεση από το επιχείρημα B, ισχύουν οι πάρα κάτω περιορισμοί:

- **Περιορισμός (ως γραμμική ανισότητα):**

$$\boldsymbol{A} + \boldsymbol{B} \leq \mathbf{1}$$

- **Επεξήγηση:**

  Εάν το A είναι μέλος της Ευσταθούς Επέκτασης, το B δεν είναι ΚΑΙ εάν το B είναι μέλος της Ευσταθούς Επέκτασης, το A δεν είναι.

  $\equiv$

  Τα επιχειρήματα A και B δεν είναι και τα δύο μαζί, ταυτόχρονα, μέλη της Ευσταθούς Επέκτασης.

  $\equiv$

  Το άθροισμα των τιμών των επιχειρημάτων A και B δεν πρέπει να ξεπερνά την τιμή 1, ώστε να μην υπάρχει δυνατότητα ανάθεσης της τιμής 1 και στα δύο επιχειρήματα, ταυτόχρονα.

6. **Κριτήρια Συμμετοχής:** Για κάθε επιχείρημα ισχύει ότι, είτε το ίδιο, είτε οποιοσδήποτε αριθμός από επιχειρήματα που του επιτίθενται, αποτελούν μέλη της Ευσταθούς Επέκτασης.

- **Περιορισμός (ως γραμμική ανισότητα):**

$$\boldsymbol{A + \sum_{i=1}^{L} attA_i \geq 1}$$

➔ μετατροπή περιορισμών στη μορφή $A * x \leq b$ που απαιτεί ο intlinprog

$$\equiv$$

$$\boldsymbol{-A - \sum_{i=1}^{L} attA_i \leq -1}$$

Όπου:

A: επιχείρημα

attA$_i$: επιχείρημα που επιτίθεται στο επιχείρημα Α

L = πλήθος επιχειρημάτων που επιτίθενται στο επιχείρημα Α

M = πλήθος επιχειρημάτων που επιτίθενται στο επιχείρημα Β

- **Επεξήγηση:**

Εάν το Α δεν είναι μέλος της Ευσταθούς Επέκτασης, τότε σίγουρα είναι μέλος της κάποιο/α από τα επιχειρήματα που επιτίθενται στο Α.

$$\equiv$$

Είτε το Α, είτε κάποιο/α από τα επιχειρήματα που του επιτίθενται, είναι μέλος/μέλη της Ευσταθούς Επέκτασης.

$$\equiv$$

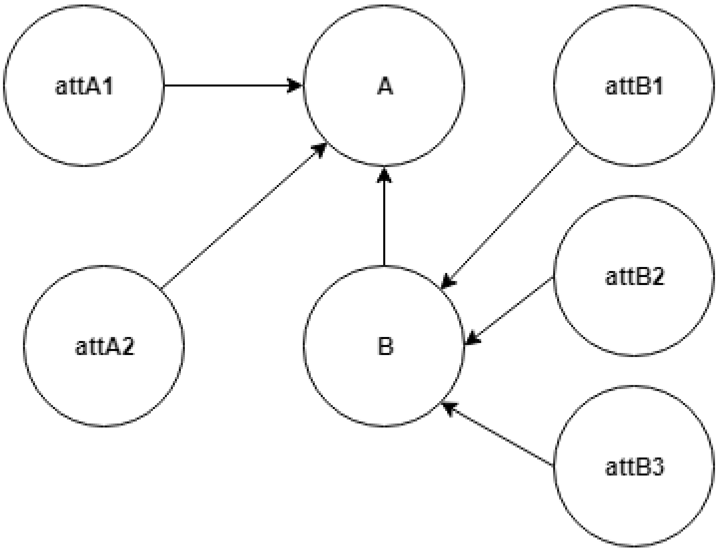
Εάν αθροίσουμε τις τιμές των επιχειρημάτων που επιτίθενται στο Α, μαζί με την τιμή του Α, θα πρέπει το άθροισμα να ξεπερνά, ή τουλάχιστον να ισούται, με την τιμή 1. Αυτό σημαίνει ότι ένα ή περισσότερα εκ των επιχειρημάτων αυτών, αποτελούν μέλη της Ευσταθούς Επέκτασης.

Για περαιτέρω κατανόηση του περιορισμού αυτού, ορίζουμε τα ακόλουθα επιχειρήματα, με τις μεταξύ τους συσχετίσεις:

- **A**: επιχείρημα που δέχεται επίθεση από το επιχείρημα Β.
- **B**: επιχείρημα που επιτίθεται στο επιχείρημα Α.

- **$attA_i$**: Κάθε επιχείρημα που επιτίθεται στο επιχείρημα Α
- **$attB_j$**: Κάθε επιχείρημα που επιτίθεται στο επιχείρημα Β

Ας δημιουργήσουμε ένα σχήμα που να αναπαριστά τα πάρα πάνω επιχειρήματα με τις μεταξύ τους συσχετίσεις:



Σχήμα 5.3.2: Συσχέτιση Πολλαπλών Επιθέσεων

Για Α, Β, $attA_1$, $attA_2$, $attB_1$, $attB_2$, $attB_3$, ισχύουν οι ακόλουθοι περιορισμοί:

$$A + B + attA_1 + attA_2 \geq 1$$

και

$$B + attB_1 + attB_2 + attB_3 \geq 1$$

➔ μετατροπή περιορισμών στη μορφή $A*x \leq b$ που απαιτεί ο intlinprog

$$\equiv$$

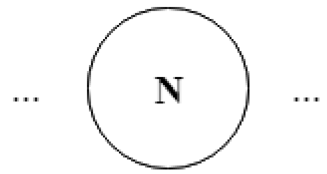
$$-A - B - attA_1 - attA_2 \leq -1$$

και

$$-B - attB_1 - attB_2 - attB_3 \leq -1$$

7. **Τετριμμένη Συμμετοχή Ουδέτερων Επιχειρημάτων :** Εάν κάποιο επιχείρημα δεν δέχεται, ούτε εξαπολύει, επίθεση από/σε οποιοδήποτε άλλο επιχείρημα, τότε τετριμμένα ανήκει στην Ευσταθή Επέκταση.

Ο περιορισμός αυτός δεν εξυπηρετεί ιδιαίτερα το σύνολο της Ευσταθούς Επέκτασης στο να αντιμετωπίσει τις απειλές που δέχεται (εφόσον ένα ουδέτερο επιχείρημα δεν επιτίθεται σε άλλα επιχειρήματα). Παρόλα αυτά, ο ορισμός του εν λόγω περιορισμού βασίζεται κυρίως στο γεγονός ότι ούτε προβλήματα δημιουργεί η συμμετοχή ενός ουδέτερου κόμβου στην Ευσταθή Επέκταση.



Σχήμα 5.3.3: Ουδέτερο Επιχείρημα

Για ΚΑΘΕ ουδέτερο επιχείρημα Ν ισχύει ο ακόλουθος μοναδιαίος περιορισμός:

- **Περιορισμός:**

$$\boldsymbol{N = 1}$$

- **Επεξήγηση:**

  Το ουδέτερο επιχείρημα Ν είναι μέλος της Ευσταθούς Επέκτασης, επομένως αναγκαστικά του ανατίθεται η τιμή 1.

- **Σημείωση:**

  Ο εν λόγω περιορισμός προκύπτει από τον περιορισμό υπ’ αριθμό 3 (Κριτήρια Συμμετοχής), εφόσον για κάθε ουδέτερο επιχείρημα ισχύει ότι, είτε το ίδιο, είτε κάποια από τα επιχειρήματα που του επιτίθενται, είναι μέλη της Ευσταθούς Επέκτασης.

  Το γεγονός όμως ότι τα ***ουδέτερα*** επιχειρήματα ***δεν δέχονται επιθέσεις***, τα κατατάσσει, αναγκαστικά, στο σύνολο της Ευσταθούς Επέκτασης.

5.4 Αποτελέσματα [15]

Τα αποτελέσματα επιστρέφονται εντός μίας μεταβλητής x. Αυτή η μεταβλητή είναι ένας Μονοδιάστατος Πίνακας (έχει μόνο μία στήλη), ο οποίος περιέχει μέσα το Μοντέλο Λύσης. Πιο συγκεκριμένα, ο δείκτης κάθε γραμμής αντιπροσωπεύει το δείκτη του επιχειρήματος. Οι τιμές που περιέχονται εντός της μοναδικής στήλης του πίνακα είναι οι τιμές που ανατίθενται στα επιχειρήματα με τον αντίστοιχο δείκτη γραμμής, ως δείκτη τους. Για παράδειγμα, το επιχείρημα 2 παίρνει την τιμή 1, εάν η στήλη της μεταβλητής x, στη γραμμή 2, περιέχει την τιμή 1.

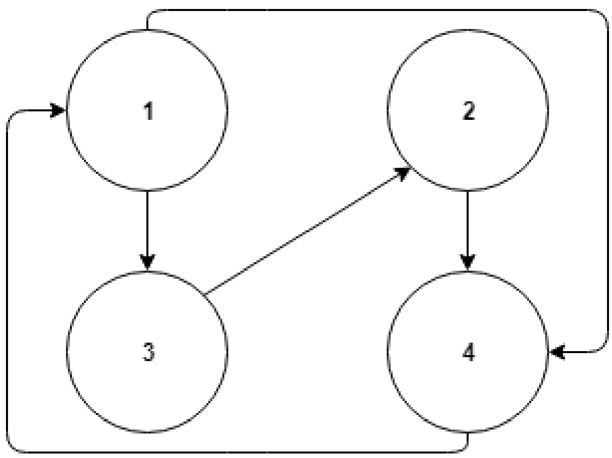
Τα αποτελέσματα ερμηνεύονται με τον ίδιο τρόπο που ερμηνεύονται και στα άλλα προβλήματα που μελετήσαμε, δηλαδή:

Αν x[k][1] = 0 → το επιχείρημα k ***δεν ανήκει*** στο σύνολο της Ευσταθούς Επέκτασης
Αν x[k][1] = 1 → το επιχείρημα k ***ανήκει*** στο σύνολο της Ευσταθούς Επέκτασης
Αν x[k][1] = Κενό → δεν υφίσταται το επιχείρημα k, στον ορισμό του προβλήματος

5.5 Παραδείγματα

5.5.1 Ικανοποιήσιμο πρόβλημα

5.5.1.1 Γραφική Αναπαράσταση



Σχήμα 5.5.1.1 Ικανοποιήσιμο Πρόβλημα

5.5.1.2 Πίνακας Γειτνίασης

0011

0001

0100

1000

5.5.1.3 Περιορισμοί

$$\{f = \min(x_1 + x_2 + x_3 + x_4),$$
$$x_1, x_2, x_3, x_4 \in Z,$$
$$0 \leq x_1, x_2, x_3, x_4 \leq 1,$$
$$x_1 + x_3 \leq 1,$$
$$x_1 + x_4 \leq 1,$$
$$x_2 + x_3 \leq 1,$$
$$x_2 + x_4 \leq 1,$$
$$-x_1 - x_4 \leq -1,$$
$$-x_2 - x_3 \leq -1,$$
$$-x_1 - x_3 \leq -1,$$
$$-x_1 - x_2 - x_4 \leq -1\}$$

5.5.1.4 Αποτελέσματα

Κατάσταση προβλήματος: ΙΚΑΝΟΠΟΙΗΣΙΜΟ (SATISFIABLE)

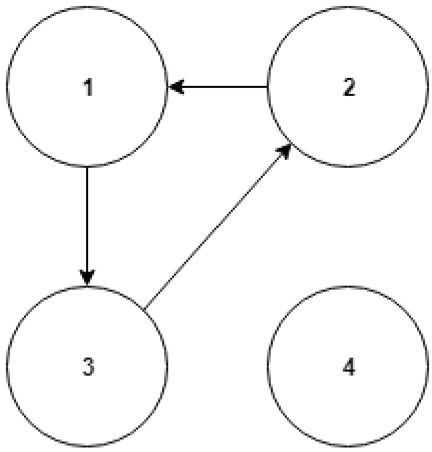
Μοντέλο λύσης: $f = 2$, $x_1 = 0$, $x_2 = 0$, $x_3 = 1$, $x_4 = 1$

Ευσταθής Επέκτασης = {3, 4}

Ερμηνεία: Τα επιχειρήματα 3 και 4 ομαδοποιούνται για να αντιμετωπίσουν τα επιχειρήματα 1 και 2, ενώ ταυτόχρονα δεν επιτίθενται το ένα στο άλλο. Η αντικειμενική συνάρτηση f παίρνει την τιμή 2, διότι δύο επιχειρήματα απαρτίζουν το σύνολο της Ευσταθούς Επέκτασης.

5.5.2 Μη-Ικανοποιήσιμο πρόβλημα

5.5.2.1 Γραφική Αναπαράσταση



Σχήμα 5.5.2.1 Μη-Ικανοποιήσιμο Πρόβλημα

5.5.2.2 Πίνακας Γειτνίασης

0010

1000

0100

0000

5.5.2.3 Περιορισμοί

$$\{f = \min(x_1 + x_2 + x_3 + x_4),$$

$$x_1, x_2, x_3, x_4 \in Z,$$

$$0 \leq x_1, x_2, x_3, x_4 \leq 1,$$

$$x_1 + x_2 \leq 1,$$

$$x_1 + x_3 \leq 1,$$

$$x_2 + x_3 \leq 1,$$

$$-x_1 - x_2 \leq -1,$$

$$-x_2 - x_3 \leq -1,$$

$$-x_1 - x_3 \leq -1,$$

$$x_4 = 1\}$$

5.5.2.4 Αποτελέσματα

Κατάσταση προβλήματος: ΜΗ-ΙΚΑΝΟΠΟΙΗΣΙΜΟ (UNSATISFIABLE)

Μοντέλο λύσης: Δεν υφίσταται

Ευσταθής Επέκταση = { }

Ερμηνεία: Δεν υπάρχουν επιχειρήματα που να μπορούν να ομαδοποιηθούν μαζί, ώστε να αντιμετωπίσουν όλα τα αντίπαλα επιχειρήματα, και ταυτόχρονα, μεταξύ τους να μην υπάρχει συσχέτιση επίθεσης. Δεν υπάρχει δηλαδή, δυνατό υποσύνολο του συνόλου των επιχειρημάτων, το οποίο να ικανοποιεί τον ορισμό της Ευσταθούς Επέκτασης, για το συγκεκριμένο πρόβλημα.

Κεφάλαιο 6

Μελέτη Μη-Ικανοποιησιμότητας στο Γραμμικό Προγραμματισμό

6.1 Αρχική Ιδέα [2, 10]

Εφόσον θέσαμε τα απαραίτητα θεμέλια για τη μοντελοποίηση, επίλυση και μελέτη προβλημάτων Ευσταθούς Επέκτασης σε Γραμμικό Προγραμματισμό, σειρά έχει να μεριμνήσουμε για τον χειρισμό των Μη- Ικανοποιήσιμων περιπτώσεων.

Ο Γραμμικός Προγραμματισμός μας προσφέρει δυνατότητες τις οποίες δεν μπορεί η εκφραστική εμβέλεια του Προτασιακού Λογισμού, η οποία περιορίζεται από τη δυαδικότητα, να ερμηνεύσει. Η χρήση γραμμικών ανισοτήτων και εξισώσεων δημιουργεί νέες διαστάσεις στις εναλλακτικές επιλογές που έχουμε καθώς μας επιτρέπει την ανάθεση τιμών σε μεταβλητές, πέραν των δυαδικών τιμών 0 και 1.

Έτσι μπορούμε να μελετήσουμε και μία άλλη πτυχή χειρισμού Μη-Ικανοποιησιμότητας η οποία δεν βασίζεται στην αγνόηση συσχετίσεων μεταξύ επιχειρημάτων (όπως στην περίπτωση του Προτασιακού Λογισμού), αλλά στην προοπτική ρύθμισης των τιμών που ανατίθενται στα επιχειρήματα ώστε, διατηρώντας τις αρχικές συσχετίσεις, να επιτευχθεί μία προσεγγιστική λύση στο πρόβλημα.

Συμπληρωματική ανάλυση της τεχνικής πτυχής του Κεφαλαίου 6 υπάρχει στο Παράρτημα Α (υποκεφάλαιο Παραρτήματος Α: Α.4).

6.2 Τρόποι Χειρισμού μέσω Γραμμικού Προγραμματισμού [1, 2]

Αφαιρώντας τους περιορισμούς που απαιτούν την ανάθεση ακέραιων τιμών στις μεταβλητές, σε συνδυασμό με κάποιους επιπλέον περιορισμούς, μπορούμε να δώσουμε τιμές στα επιχειρήματα μας, πέραν των απόλυτων τιμών συμμετοχής που είχαμε συναντήσει έως τώρα (0 = FALSE και 1 = TRUE).

Πιο συγκεκριμένα, μπορούμε να επιτρέψουμε την ανάθεση πραγματικών τιμών που εμπίπτουν στο πεδίο τιμών [0, 1], στις μεταβλητές. Έτσι μπορούμε να αναφερόμαστε σε «μερικές» λύσεις του προβλήματος, σύμφωνα με τις οποίες ισχύει ότι:

- επιχειρήματα των οποίων η τιμή **πλησιάζει περισσότερο το άνω όριο (τιμή 1)** έχουν **μεγαλύτερη προοπτική** να αποτελέσουν μέλη της Ευσταθούς Επέκτασης.
- επιχειρήματα με τιμή **ίση με 1 συμμετέχουν**, χωρίς αμφιβολία, στην Ευσταθή Επέκταση.
- επιχειρήματα των οποίων η τιμή **πλησιάζει περισσότερο το κάτω όριο (τιμή 0)** έχουν **μικρότερη προοπτική** να αποτελέσουν μέλη της Ευσταθούς Επέκτασης.

- επιχειρήματα με τιμή **ίση με 0 απουσιάζουν**, χωρίς αμφιβολία, από την Ευσταθή Επέκταση.

6.3 Εξεύρεση Ευσταθούς Επέκτασης μέσω Συμβιβασμών [10]

Για να δημιουργήσουμε ένα σύνολο Ευσταθούς Επέκτασης, υπό συμβιβασμούς, μπορούμε να αναθέσουμε, χωρίς ενδοιασμούς, όλα τα επιχειρήματα που έχουν τιμή ίση με 1.0, ως μέλη.

Συνεχίζουμε με την ανάθεση επιχειρημάτων που πλησιάζουν περισσότερο το 1.0, αντί το 0.0, με προτεραιότητα να έχουν αυτά με τη μεγαλύτερη τιμή. Σε κάθε βήμα, προσπαθούμε να μην παραβιάζουμε τον ορισμό της Ευσταθούς Επέκτασης.

Τέλος, αν παρατηρήσουμε ότι το σύνολό μας (το οποίο αποτελείτε από επιχειρήματα όπως περιγράφεται πάρα πάνω) δεν μπορεί να αντιμετωπίσει όλες τις απειλές που δέχεται, θα αναγκαστούμε να αναθέσουμε ως μέλη, και επιχειρήματα τα οποία πλησιάζουν αρκετά το κάτω όριο. Προϋπόθεση είναι ότι, ο συνδυασμός επιχειρημάτων που σίγουρα είναι μέλη του συνόλου, με αυτά που είναι πιο πιθανόν να είναι μέλη (πλησιάζουν το άνω άκρο) και τα μερικά επιχειρήματα που έχουν μικρή πιθανότητα συμμετοχής (πλησιάζουν το κάτω άκρο) και τα οποία επιλέξαμε συμβιβαστικά, θα πρέπει να επιτίθεται σε κάθε επιχείρημα που δεν αποτελεί μέλος του.

Με άλλα λόγια, καταφεύγουμε στην εσχάτη επιλογή η οποία είναι να δημιουργήσουμε Ευσταθή Επέκταση, η οποία περιέχει και επιχειρήματα που δεν στηρίζουν σε μεγάλο βαθμό την άποψη του συνόλου της. Παρόλα αυτά, η προσέγγιση αυτή βοηθά το ίδιο το σύνολο στο να αντιμετωπίσει άλλα επιχειρήματα που σίγουρα τάσσονται εναντίον του (συμμαχία με επιχειρήματα που έχουν τιμές πιο κοντά στο 0, για να αντιμετωπιστούν επιχειρήματα των οποίων οι τιμές τους ισούνται με 0 η πλησιάζουν στο μεγαλύτερο δυνατό βαθμό το κάτω άκρο). Επομένως αναφερόμαστε σε μία ελάχιστη χαλαρή συμμαχία με κάποιον όχι-τόσο φιλικό παράγοντα, για κοινή αντιμετώπιση ενός αποδεδειγμένα εχθρικού παράγοντα.

Αποφεύγουμε να αναθέσουμε επιχειρήματα, στο σύνολο της Ευσταθούς Επέκτασης, τα οποία έχουν τιμή ίση με 0.

6.4 Επιλυτής Μεικτού-Ακέραιου Γραμμικού Προγραμματισμού [15]

Στα ακόλουθα βήματα θα προσέξουμε ότι χρειάζεται ανάθεση πραγματικών τιμών στα επιχειρήματα, ενώ σε άλλες, βοηθητικές μεταβλητές ανατίθενται ακέραιες τιμές. Επομένως, η περίπτωση χειρισμού Μη- Ικανοποιήσιμων προβλημάτων Γραμμικού Προγραμματισμού που μελετάμε αποτελεί πρόβλημα Μεικτού Ακέραιου Γραμμικού Προγραμματισμού (Mixed-Integer Linear Programming, MILP).

Για προβλήματα Μεικτού Ακέραιου Γραμμικού Προγραμματισμού χρησιμοποιείται ο επιλυτής intlinprog() της MATLAB, καθώς υποστηρίζει τη δυνατότητα χειρισμού, τόσο ακέραιων, όσο και πραγματικών μεταβλητών, στα πλαίσια του ίδιου προβλήματος.

Για το αρχείο εισόδου (Matlab Script File) και τις οποιεσδήποτε ιδιαιτερότητες, απαιτήσεις και δυνατότητες της Matlab, ισχύουν τα όσα ειπώθηκαν και στο Κεφάλαιο 5 (βλ. 5.2.3, 5.2.4, 5.2.5, Παράρτημα Α: υποκεφάλαιο 4).

6.5 Περιορισμοί [1, 2, 10]

Ισχύουν τα όσα προαναφέρθηκαν στο Κεφάλαιο 5 (βλ. 5.3), με κάποιους επιπλέον περιορισμούς. Πιο συγκεκριμένα, θα προστεθούν περιορισμοί που θα είναι υπεύθυνοι για την ώθηση τιμών προς τα άκρα του πεδίου τιμών [0, 1], δηλαδή είναι υπεύθυνοι για την ανάθεση τιμών, που πλησιάζουν όσο το δυνατό πιο κοντά γίνεται στο 0 ή το 1.

Η ενέργεια αυτή αποσκοπεί στην αποφυγή ανάθεσης της τιμής 0.5 στα επιχειρήματα, καθώς αυτό οδηγεί σε αδυναμία να ξεκαθαρίσουμε, με βεβαιότητα, το πιο άκρο προσεγγίζει το επιχείρημα, λόγω του ότι η τιμή 0.5 αποτελεί τη μέση τιμή (mean value) του πεδίου τιμών και απέχει από τα δύο άκρα εξ’ ίσου.

Ακολουθούν οι ανισότητες:

$$1-a_i-k_i-\lambda_i^1 \leq 0$$

$$a_i-k_i-\lambda_i^2 \leq 0$$

$$0 \leq k_i \leq \varepsilon$$

$$\lambda_i^1 + \lambda_i^2 < 2$$

$$\lambda_i^1, \lambda_i^2 \in N$$

Όπου:

- a_i: επιχείρημα με δείκτη i
- k_i: απόσταση επιχειρήματος a_i από το άκρο στο οποίο τείνει να ανήκει
- λ_i^1: μεταβλητή που καθορίζει εάν η τιμή του επιχειρήματος a_i θα βρίσκεται πιο κοντά στο άνω άκρο, με απόσταση k_i κάτω από αυτό
- λ_i^2: μεταβλητή που καθορίζει εάν η τιμή του επιχειρήματος a_i θα βρίσκεται πιο κοντά στο κάτω άκρο, με απόσταση k_i πάνω από αυτό
- ε: μέγιστη επιτρεπτή απόσταση επιχειρήματος από τα όρια. Αποτελεί το άνω φράγμα τιμών της απόστασης k_i, του κάθε επιχειρήματος a_i

Λειτουργία Ανισοτήτων:

1. Ώθηση τιμών προς το άνω όριο (1):

$$1-a_i-k_i-\lambda_i^1 \leq 0$$

Για τιμή $\lambda_i^1 = 0$: Ο εν λόγω περιορισμός αναθέτει τιμή στο επιχείρημα a_i, η οποία βρίσκεται πιο κοντά στο άνω όριο (1), και από το οποίο απέχει ΤΟ ΠΟΛΥ κατά k_i. Π.χ.:

Για: $k_i = 0.2$ και $\lambda_i^1 = 0$ τότε: $1 - a_i - 0.2 - 0 \leq 0 \equiv a_i \geq 0.8$

➔ Πολύ πιθανόν το επιχείρημα a_i να είναι μέλος της Ευσταθούς Επέκτασης, καθώς βρίσκεται πιο κοντά στο άνω όριο 1 (παίρνει τιμές μεταξύ 0.8 και 1).

2. Ώθηση τιμών προς το κάτω όριο (0):

$$a_i - k_i - \lambda_i^2 \leq 0$$

Για τιμή $\lambda_i^2 = 0$: Ο εν λόγω περιορισμός αναθέτει τιμή στο επιχείρημα a_i, η οποία βρίσκεται πιο κοντά στο κάτω όριο (0), και από το οποίο απέχει ΤΟ ΠΟΛΥ κατά k_i. Π.χ.:

$$\text{Για: } k_i = 0.2 \text{ και } \lambda_i^2 = 0 \text{ τότε: } a_i - 0.2 \leq 0 \equiv a_i \leq 0.2$$

➔ Πολύ πιθανόν το επιχείρημα a_i να μην είναι μέλος της Ευσταθούς Επέκτασης, καθώς βρίσκεται πιο κοντά στο κάτω όριο 0 (παίρνει τιμές μεταξύ 0 και 0.2).

3. Καθορισμός απόστασης από άκρα:

$$0 \leq k_i \leq \varepsilon$$

Η απόσταση k_i, του κάθε επιχειρήματος a_i, από το άνω ή κάτω όριο, περιορίζεται από το γεγονός ότι δεν είναι επιτρεπτή η ανάθεση αρνητικής απόστασης. Ταυτόχρονα, άνω φράγμα στην τιμή του k_i αποτελεί η σταθερά ε, η οποία ορίζει τη μέγιστη επιτρεπτή απόσταση επιχειρημάτων από τα άκρα.

4. Επιλογή Κατεύθυνσης Ώθησης:

$$\lambda_i^1 + \lambda_i^2 < 2$$
$$\lambda_i^1, \lambda_i^2 \in \mathrm{N}$$

Οι αναθέσεις τιμών, είτε αυτές βρίσκονται πιο κοντά στο άνω άκρο (1), είτε βρίσκονται στο κάτω άκρο (0), καθορίζονται από τις μεταβλητές λ_i^1, λ_i^2. Ορίζουμε τις μεταβλητές αυτές ως φυσικούς, ακέραιους αριθμούς, των οποίων το άθροισμα δεν μπορεί να υπερβαίνει την τιμή 2. Αυτό σημαίνει ότι μόνο μία εκ των δύο μεταβλητών θα πάρει την τιμή 1, μοντελοποιώντας έτσι την ιδιότητα των επιχειρημάτων που τους επιτρέπει να πλησιάζουν είτε το πάνω, είτε το κάτω άκρο, όχι όμως και τα δύο άκρα μαζί, ταυτόχρονα.

6.6 Χειρισμός Μέγιστης Επιτρεπτής Απόστασης ε [2]

Η MATLAB προσπαθεί να επιλύσει το πρόβλημα, ικανοποιώντας όλους τους περιορισμούς με οποιοδήποτε τρόπο μπορεί να το πετύχει αυτό. Συχνά, οι αναθέσεις τιμών στο μοντέλο λύσης που επιστρέφεται δύναται να μην προσφέρουν κάποια σημαντική πληροφορία, δυσκολεύοντας την ερμηνεία των αποτελεσμάτων. Επιλύνοντας προβλήματα Μεικτού Ακέραιου Γραμμικού Προγραμματισμού με τη χρήση του επιλυτή intlinprog(), θα παρατηρήσουμε ότι ως προκαθορισμένη και προτιμώμενη ανάθεση τιμών, στην οποία καταφεύγει αρκετά συχνά ο επιλυτής, είναι η ανάθεση της τιμής 0.5 στα πλείστα (αν όχι και σε όλα, μερικές φορές) επιχειρήματα. Το γεγονός αυτό αυξάνει το βαθμό αβεβαιότητας, καθώς η τιμή 0.5 απέχει εξ' ίσου και από τα δύο άκρα, με αποτέλεσμα να μην είναι ξεκάθαρη η «προτίμηση» που δίνεται στο κάθε επιχείρημα.

Για να ελαχιστοποιήσουμε το βαθμό αβεβαιότητας, προσπαθούμε να ελαχιστοποιήσουμε την παράμετρο εκείνη που μπορεί να επιφέρει καλύτερες αναθέσεις, από τις έως τώρα «βολικές» αναθέσεις της τιμής 0.5 Πιο συγκεκριμένα, η μέγιστη επιτρεπτή απόσταση από τα όρια, η οποία μοντελοποιείται μέσω της μεταβλητής ε, πρέπει να ελαχιστοποιηθεί. Η αλλαγή αυτή θα επηρεάσει τις τιμές των αποστάσεων των επιχειρημάτων $\mathbf{a}_i$ από το όριο στο οποίο πλησιάζουν (τις τιμές των μεταβλητών $\mathbf{k}_i$ δηλαδή), ώστε οι αποστάσεις επιχειρήματος-άνω άκρου και επιχειρήματος-κάτω άκρου να μην είναι ίσες. Έτσι αποφεύγεται η ανάθεση της τιμής 0.5 στα επιχειρήματα. Για να γίνει εφικτό αυτό, στο σύνολο των περιορισμών μας υπολογίζουμε και τον ακόλουθο περιορισμό, ο οποίος αναφέρεται στη μεταβλητή ε, ως την αντικειμενική συνάρτηση f που θέλουμε να ελαχιστοποιήσουμε, ώστε:

Προσέγγιση #1 - Ελαχιστοποίηση: $f = \varepsilon = min(\varepsilon)$

Εναλλακτικά, η αβεβαιότητα μπορεί να αντιμετωπιστεί και με μία διαφορετική προσέγγιση. Πιο συγκεκριμένα, μπορούμε να δώσουμε την τιμή της μέγιστης επιτρεπτής απόστασης ε, ως μία σταθερά. Έτσι ο επιλυτής δεν θα χρειαστεί να ελαχιστοποιήσει την αντικειμενική συνάρτηση. Θα προσπαθήσει μόνο να αναθέσει τιμές στα επιχειρήματα, οι οποίες δεν θα απέχουν απόσταση πέραν της τιμής που δόθηκε για το ε, από το όριο που προσεγγίζουν. Αν για παράδειγμα δοθεί η τιμή ε = 0.2, κανένα επιχείρημα δεν θα πάρει τιμές κάτω από 0.8 ή πάνω από 0.2, επομένως αποκλείεται αμέσως η πιθανότητα μέγιστης αβεβαιότητας, όπου στα επιχειρήματα υπήρχε και η δυνατότητα ανάθεσης της τιμής 0.5.

Προσέγγιση #2 – Ανάθεση Τιμής: $f = \varepsilon = ΣΤΑΘΕΡΑ\_ΤΙΜΗ$

6.7 Μοντελοποίηση [15, 16, 23]

Μετά τη δήλωση και επεξήγηση των απαραίτητων περιορισμών, ακολουθεί η μοντελοποίηση του προβλήματος στην Matlab. Θα πρέπει όλα τα προαναφερθέντα στοιχεία (αντικειμενική συνάρτηση, μεταβλητές, περιορισμοί, άνω-κάτω άκρα τιμών, επίλυση κ.λπ.) να διατυπωθούν στη μορφή που ζητά η Matlab. Ισχύουν τα όσα προαναφέρθηκαν στο Κεφάλαιο 5, περί μορφής του αρχείου και ιδιαιτεροτήτων του εργαλείου Matlab, με κάποιες επιπλέον αλλαγές, που προσαρμόζουν το πρόβλημα στα νέα ζητούμενα. Πιο συγκεκριμένα, προκύπτουν οι πάρα κάτω παρατηρήσεις:

- **f:** η αντικειμενική συνάρτηση, για την οποία θα ζητήσουμε την βελτιστοποίησή της. Δηλώνεται ως πίνακας (matrix) και παρουσιάζει το πλήθος, αλλά και τους συντελεστές των μεταβλητών που συμμετέχουν στο πρόβλημα. Στην περίπτωση μας, αντικειμενική συνάρτηση αποτελεί η μέγιστη επιτρεπτή απόσταση επιχειρημάτων, από το όριο στο οποίο ανήκουν (μεταβλητή ε). Σκοπός μας αποτελεί η ελαχιστοποίηση της μεταβλητής ε.

  Οι μεταβλητές αποτελούνται από (με τη σειρά που παρουσιάζονται πάρα κάτω):

  1. την τιμή του κάθε επιχειρήματος $\mathrm{a_i}$ του προβλήματος
  2. την κάθε απόσταση $\mathrm{k_i}$, του κάθε επιχειρήματος $\mathrm{a_i}$, από κάποιο όριο
  3. την κάθε μεταβλητή λ_i^1, του κάθε επιχειρήματος $\mathrm{a_i}$
  4. την κάθε μεταβλητή λ_i^2, του κάθε επιχειρήματος $\mathrm{a_i}$
  5. την μέγιστη επιτρεπτή απόσταση ε

  Σύνολο Μεταβλητών:

$$4\ \text{μεταβλητές ανά επιχείρημα} + \text{μεταβλητή ε} = 4 * |\text{Σύνολο Επιχειρημάτων}| + 1$$

- **intcon:** διάνυσμα (vector) που υποδεικνύει ποιες εκ των μεταβλητών θα είναι ακέραιες. Στην περίπτωσή μας, εφόσον επιθυμούμε και ανάθεση πραγματικών τιμών στα επιχειρήματα και στις μεταβλητές που μοντελοποιούν αποστάσεις, δηλώνουμε ως ακέραιες **μόνο** τις μεταβλητές λ_i^1 και λ_i^2 (για κάθε επιχείρημα a_i).
- **A:** πίνακας (matrix) που περιέχει τους συντελεστές των μεταβλητών που συμμετέχουν στα αριστερά μέρη των περιορισμών **ανισοτήτων.**
- **b:** πίνακας (matrix) που περιέχει τις τιμές που βρίσκονται στα δεξιά μέρη των περιορισμών **ανισοτήτων.**
- **Aeq:** πίνακας (matrix) που περιέχει τους συντελεστές των μεταβλητών που συμμετέχουν στα αριστερά μέρη των περιορισμών **εξισώσεων.**
- **beq:** πίνακας (matrix) που περιέχει τις τιμές που βρίσκονται στα δεξιά μέρη των περιορισμών **εξισώσεων.**
- **lb:** κάτω φράγμα τιμών (lower bound). Δηλώνεται ως διάνυσμα (vector). Στην περίπτωση μας, όλες οι μεταβλητές έχουν ως κάτω φράγμα την τιμή 0.
- **ub:** άνω φράγμα τιμών (upper bound). Δηλώνεται ως διάνυσμα (vector). Στην περίπτωση μας, όλες οι μεταβλητές έχουν ως κάτω φράγμα την τιμή 1.
- $\mathbf{[x, fval] = intlinprog(f, intcon, A, b, Aeq, beq, lb, ub)}$ **:** κλήση επιλυτή intlinprog, λαμβάνοντας υπ’ όψιν όλες τις πάρα πάνω παραμέτρους που δηλώθηκαν, επιστρέφοντας το μοντέλο επίλυσης στη μεταβλητή x, μαζί με την τιμή της βελτιστοποιημένης αντικειμενικής συνάρτησης (min(ε)), στη μεταβλητή fval.

6.8 Αποτελέσματα [15]

Τα αποτελέσματα επιστρέφονται εντός μίας μεταβλητής x. Αυτή η μεταβλητή είναι ένας Μονοδιάστατος Πίνακας (έχει μόνο μία στήλη), ο οποίος περιέχει μέσα το Μοντέλο Λύσης. Πιο συγκεκριμένα, κάθε γραμμή της στήλης αντιστοιχεί σε μία μεταβλητή του προβλήματος, με τη σειρά αυτή (για πλήθος επιχειρημάτων N):

- N γραμμές που αναφέρονται στις τιμές των επιχειρημάτων $a_1 \ldots a_N$
- N γραμμές που αναφέρονται στις αποστάσεις επιχειρημάτων, $k_1 \ldots k_N$

- N γραμμές που αναφέρονται στη μεταβλητή λ^1 των επιχειρημάτων, $\lambda_1^1 \dots \lambda_N^1$
- N γραμμές που αναφέρονται στη μεταβλητή λ^2 των επιχειρημάτων, $\lambda_1^2 \dots \lambda_N^2$
- 1 γραμμή που αναφέρεται στην μέγιστη επιτρεπτή απόσταση ε

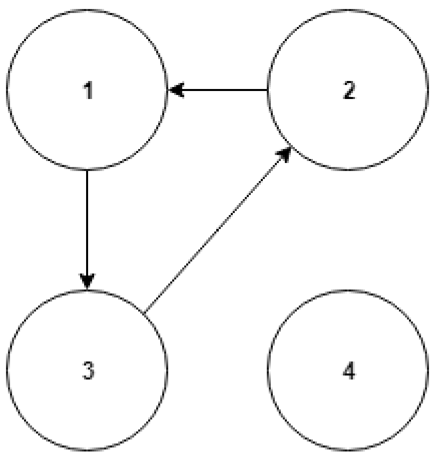
Τα αποτελέσματα ερμηνεύονται ως εξής:

- Αν x[k][1] = 0 → το επιχείρημα k ***δεν ανήκει*** στο σύνολο της Ευσταθούς Επέκτασης
- Αν x[k][1] = 1 → το επιχείρημα k ***ανήκει*** στο σύνολο της Ευσταθούς Επέκτασης
- Αν x[k][1] = Κενό → δεν υφίσταται το επιχείρημα k, στον ορισμό του προβλήματος
- Αν x[k][1] πλησιάζει το 1: το επιχείρημα k είναι πιο πιθανόν να ανήκει στο σύνολο της Ευσταθούς Επέκτασης
- Αν x[k][1] πλησιάζει το 0: το επιχείρημα k είναι πιο πιθανόν να μην ανήκει στο σύνολο της Ευσταθούς Επέκτασης

6.9 Παραδείγματα

6.9.1 Μη-Ικανοποιήσιμο πρόβλημα

6.9.1.1 Γραφική Αναπαράσταση



Σχήμα 6.9.1.1 Μη-Ικανοποιήσιμο Πρόβλημα

6.9.1.2 Πίνακας Γειτνίασης

0010

1000

0100

0000

6.9.1.3 Περιορισμοί

$$\{f = \min(\varepsilon),$$

$$0 \leq a_i,\ k_i,\ \lambda_i^1,\ \lambda_i^2,\ \varepsilon \leq 1, \text{ για κάθε i: } 1 \leq i \leq 4$$

$$x_1 + x_2 \leq 1,$$

$$x_1 + x_3 \leq 1,$$

$$x_2 + x_3 \leq 1,$$

$$-x_1 - x_2 \leq -1,$$

$$-x_2 - x_3 \leq -1,$$

$$-x_1 - x_3 \leq -1,$$

$$x_4 = 1,$$

$$1 - a_i - k_i - \lambda_i^1 \leq 0, \text{ για κάθε i: } 1 \leq i \leq 4$$

$$a_i - k_i - \lambda_i^2 \leq 0, \text{ για κάθε i: } 1 \leq i \leq 4$$

$$0 \leq k_i \leq \varepsilon, \text{ για κάθε i: } 1 \leq i \leq 4$$

$$\lambda_i^1 + \lambda_i^2 < 2, \text{ για κάθε i: } 1 \leq i \leq 4$$

$$\lambda_i^1, \lambda_i^2 \in N, \text{ για κάθε i: } 1 \leq i \leq 4\}$$

6.9.1.4 Επίλυση, Αποτελέσματα και Ερμηνεία

1	0.3333
2	0.3333
3	0.6667
4	1

Σχήμα 6.9.1.4.1 Αποτελέσματα τιμών επιχειρημάτων

17	0.3333

Σχήμα 6.9.1.4.2 Αποτέλεσμα τιμής ελαχιστοποιημένου ε

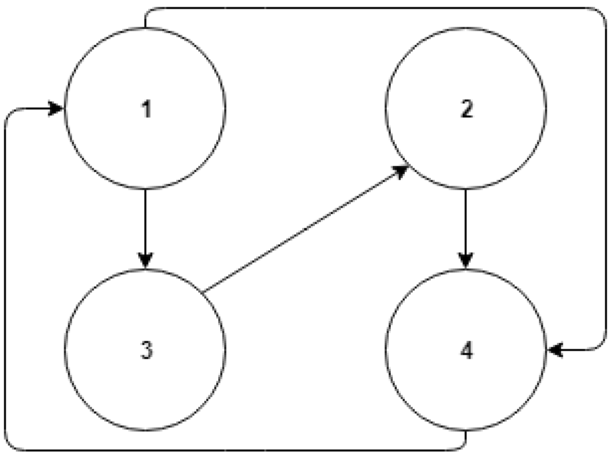
Τα επιχειρήματα 1 και 2 έχουν μικρή πιθανότητα συμμετοχής στην Ευσταθή Επέκταση ($a_1 = a_2 = 0.3333 \rightarrow$ πλησιάζει περισσότερο το κάτω όριο 0, αντί το άνω όριο 1). Το επιχείρημα 3 έχει μεγαλύτερη πιθανότητα να συμμετέχει στην Ευσταθή Επέκταση ($a_3 = 0.6667 \rightarrow$ πλησιάζει περισσότερο το άνω όριο 1, αντί το κάτω όριο 0).

Το επιχείρημα 4 ανήκει σίγουρα στο σύνολο της Ευσταθούς Επέκτασης επειδή είναι ουδέτερο επιχείρημα ($a_4 = 1$).

Επίσης παρατηρούμε ότι τα επιχειρήματα πήραν τιμές με απόσταση, όχι μεγαλύτερη του ε (= 0.3333), από τα όρια στα οποία πιο πιθανόν να ανήκουν.

6.9.2 Ικανοποιήσιμο Πρόβλημα

6.9.2.1 Γραφική Αναπαράσταση



Σχήμα 6.9.2.1 Ικανοποιήσιμο Πρόβλημα

6.9.2.2 Πίνακας Γειτνίασης

0011

0001

0100

1000

6.9.2.3 Περιορισμοί

$$\{f = \min(\varepsilon),$$

$$0 \le a_i,\ k_i,\ \lambda_i^1,\ \lambda_i^2,\ \varepsilon \le 1, \text{ για κάθε i: } 1 \le i \le 4$$

$$x_1 + x_3 \le 1,$$

$$x_1 + x_4 \le 1,$$

$$x_2 + x_3 \le 1,$$

$$x_2 + x_4 \le 1,$$

$$-x_1 - x_4 \le -1,$$

$$-x_2 - x_3 \le -1,$$

$$-x_1 - x_3 \le -1,$$

$$-x_1 - x_2 - x_4 \le -1,$$

$$1 - a_i - k_i - \lambda_i^1 \le 0, \text{ για κάθε i: } 1 \le i \le 4$$

$$a_i - k_i - \lambda_i^2 \le 0, \text{ για κάθε i: } 1 \le i \le 4$$

$$0 \le k_i \le \varepsilon, \text{ για κάθε i: } 1 \le i \le 4$$

$$\lambda_i^1 + \lambda_i^2 < 2, \text{ για κάθε i: } 1 \le i \le 4$$

$$\lambda_i^1, \lambda_i^2 \in \mathbb{N}, \text{ για κάθε i: } 1 \le i \le 4\}$$

6.9.2.4 Επίλυση, Αποτελέσματα και Ερμηνεία

1	1
2	1
3	0
4	0

Σχήμα 6.9.2.4.1 Αποτελέσματα τιμών επιχειρημάτων

17	0

Σχήμα 6.9.2.4.2 Αποτέλεσμα τιμής ελαχιστοποιημένου ε

Τα επιχειρήματα 1 και 2, με τιμές ίσες του 1, είναι σίγουρα μέλη του συνόλου Ευσταθούς Επέκτασης, ενώ τα επιχειρήματα 3 και 4, με τιμές ίσες του 0, σίγουρα απουσιάζουν από αυτό.

Παρατηρούμε την ανάθεση ακέραιων τιμών στα επιχειρήματα, και ταυτόχρονα, το μηδενισμό της τιμής της μέγιστης δυνατής απόστασης επιχειρημάτων από τα όρια τους, ε. Ο λόγος είναι ο εξής: Η προσπάθεια επίλυσης ενός προβλήματος Ακέραιου Γραμμικού Προγραμματισμού, Ευσταθούς Επέκτασης, το οποίο είναι Ικανοποιήσιμο, χρησιμοποιώντας Μεικτό Γραμμικό Προγραμματισμό, θα οδηγήσει στην επιστροφή του ίδιου μοντέλου λύσης. Αυτό οφείλεται στο γεγονός ότι, εφόσον αναφερόμαστε σε ένα Ικανοποιήσιμο πρόβλημα, γίνεται ανάθεση τιμών ΑΚΡΙΒΩΣ ίσων με 0 ή 1, κάτι που συνεπάγεται με μηδενική απόσταση από τα όρια. Επομένως, η αντικειμενική συνάρτηση που προσπαθούμε να ελαχιστοποιήσουμε (ε), η οποία αποτελεί την μέγιστη επιτρεπτή απόσταση από τα όρια, τυγχάνει ήδη τις μεγαλύτερης δυνατής ελαχιστοποίησης που μπορούσε να επιδεχθεί, μέσω του μηδενισμού της. Έτσι, στα επιχειρήματα θα ανατεθούν τιμές που απέχουν κατά 0 από τα όρια τιμών, δηλαδή γίνεται ανάθεση τιμών 0 ή 1.

Κεφάλαιο 7

Πειραματική Αξιολόγηση

7.1 Κίνητρο και Σημαντικότητα

Η πειραματική αξιολόγηση αποτελεί τη διαδικασία μέσω της οποίας το θεωρητικό υπόβαθρο μίας εικασίας, ιδέας ή μελέτης, δοκιμάζεται στην πράξη. Αποτελεί ένα σημαντικό και αναπόσπαστο κομμάτι της επιστημονικής έρευνας, καθώς μέσω αυτής της διαδικασίας δύναται να καταρριφθούν ιδέες, να βγουν στο φως νέα ευρήματα που προκύπτουν ή να επαληθευτεί η ορθότητα των όσων υποστηρίζονται στα συμφραζόμενα του θεωρητικού μέρους.

Παράλληλα, η πειραματική αξιολόγηση μπορεί να αξιοποιηθεί ως εργαλείο μέτρου συγκρίσεως. Μία από τις χρησιμότερες εξ’ αυτών, που διεξάγονται αμέσως μετά την ερμηνεία των αποτελεσμάτων ενός πειράματος, είναι η σύγκριση μεταξύ διαφόρων παραμέτρων εισόδου, στο όσον αφορά το πως επηρεάζουν τις παραμέτρους εξόδου, για κάποιο πρόβλημα. Ένα αρκετά καλό παράδειγμα αποτελεί η σύγκριση χρόνων εκτέλεσης προβλημάτων Ευσταθούς Επέκτασης, με διαφορετικό αριθμό κόμβων ή τιμή πυκνότητας κάθε φορά (με χρήση του ίδιου εργαλείου).

Επιπλέον, επιτρέπει όχι μόνο τη σύγκριση μεταξύ παραμέτρων ενός προβλήματος, αλλά και τη σύγκριση μεταξύ διαφορετικών εργαλείων επίλυσης. Ένα αρκετά καλό παράδειγμα αποτελεί η σύγκριση χρόνων επίλυσης προβλημάτων Ευσταθούς Επέκτασης, με ίδιο αριθμό κόμβων και τιμή πυκνότητας, όμως με τη χρήση διαφορετικού εργαλείου κάθε φορά (π.χ. σύγκριση επίλυσης σε Προτασιακό Λογισμό με επίλυση σε Ακέραιο Γραμμικό Προγραμματισμό).

7.2 Στόχοι

Πρώτο βήμα αποτελεί η ανάθεση ξεκάθαρων στόχων. Στα πλαίσια της εργασίας αυτής, στόχοι μας αποτελούν οι εξής:

- Πως η μεταβολή της παραμέτρου «Αριθμός Κόμβων Γραφήματος» επηρεάζει το συνολικό χρόνο επίλυσης των προβλημάτων που παρουσιάστηκαν στα προηγούμενα κεφάλαια.
- Πως η μεταβολή της παραμέτρου «Πυκνότητα Γραφήματος» επηρεάζει το συνολικό χρόνο επίλυσης των προβλημάτων που παρουσιάστηκαν στα προηγούμενα κεφάλαια.
- Πως η επίλυση των προβλημάτων που παρουσιάστηκαν στα προηγούμενα κεφάλαια, μέσω διαφορετικών εργαλείων μοντελοποίησης, επηρεάζει το συνολικό χρόνο επίλυσης, με σταθερούς τους παράγοντες «Αριθμός Κόμβων» και «Πυκνότητα».

7.3 Δημιουργία Δειγμάτων

Δεύτερο βήμα αποτελεί η δημιουργία τυχαίων δειγμάτων (random samples) για την εκτέλεση πειραμάτων. Τα δείγματα αποτελούν στιγμιότυπα των 5 προβλημάτων που μελετήθηκαν, τα οποία εκτελούνται ώστε να μελετηθεί ο χρόνος εκτέλεσης τους, βάσει των παραμέτρων τους. Η εν λόγω διαδικασία είναι κρίσιμη για την αξιοπιστία των τελικών αποτελεσμάτων, καθώς λανθασμένα ή μη-ενδεικτικά δείγματα, σε συνδυασμό με την πάντοτε υπαρκτή πιθανότητα στατιστικών λαθών, θα επιφέρουν την εξαγωγή λανθασμένων συμπερασμάτων. Προκειμένου να αποφευχθεί αυτό, λήφθηκαν τα ακόλουθα μέτρα:

- Για να μελετηθεί, με περισσότερη ακρίβεια, η διαφορά μεταξύ των χρόνων εκτέλεσης, συγκρίνουμε τιμές των παραμέτρων για τις οποίες η διαφορά αυτή θα είναι διακριτή. Πιο συγκεκριμένα, χρησιμοποιήθηκαν οι ακόλουθες τιμές για κάθε παράμετρο:

  Αριθμός Κόμβων Γραφήματος: 100, 200, 300
  Πυκνότητα Γραφήματος: 0.15, 0.20, 0.25, 0.30

  Εξήγηση: Η αύξηση των κόμβων, κατά 100 κάθε φορά, διασφαλίζει την ουσιαστική διαφορά μεταξύ των χρόνων εκτέλεσης στα αποτελέσματα. Τα προβλήματα που μελετήσαμε στα προηγούμενα κεφάλαια είναι της τάξης NP-Hard και έχουν εκθετική πολυπλοκότητα. Επομένως, εύλογα μπορούμε να υποθέσουμε ότι η σύγκριση αριθμών κόμβων με αρκετά μικρή διαφορά (π.χ. 2, 4, 6 κόμβοι) δεν θα παρουσιάσει ιδιαίτερα μεγάλη απόκλιση στα αποτελέσματα, ενώ σε αρκετές περιπτώσεις δύναται να επιφέρει ακόμη και παρόμοια αποτελέσματα (π.χ. ίδιος χρόνος εκτέλεσης τόσο για 2, όσο και για 8 κόμβους). Η δραματική αύξηση των χρόνων εκτέλεσης είναι διακριτή σε μεγάλης κλίμακας αυξήσεις του αριθμού των κόμβων, εξ’ ου και η αύξηση ανά 100, κάθε φορά. Η πυκνότητα επιδέχεται αύξηση κατά 0.5 κάθε φορά, για τους ίδιους λόγους.

- Για να μειώσουμε, όσο το δυνατόν περισσότερο, την επίδραση που επιφέρουν τυχόν στατιστικά λάθη, θα δημιουργηθούν περισσότερα από 1 δείγματα για κάθε συνδυασμό <Αριθμού Κόμβων, Πυκνότητας>. Ιδανικότερη επιλογή αποτελεί η δημιουργία 10 ή περισσότερων δειγμάτων ανά συνδυασμό.

Εφόσον επιλέξαμε 3 διαφορετικές τιμές κόμβων (100, 200, 300) και 4 διαφορετικές τιμές πυκνοτήτων (0.15, 0.20, 0.25, 0.30), έχουμε 12 διαφορετικούς συνδυασμούς προβλημάτων Ευσταθούς Επέκτασης (100 κόμβοι με 0.15 πυκνότητα, 100 κόμβοι με 0.20 πυκνότητα … 300 κόμβοι με 0.30 πυκνότητα). Άρα καταλήγουμε στα εξής συμπεράσματα:

$$Αρ.\,Συνολικών\ Δειγμάτων = N * 3 * 4 = 12 * N,$$
$$όπου:$$
$$N: πλήθος\ δειγμάτων\ ανά\ συνδιασμό,$$
$$N \geq 10$$

Επεξήγηση: Η δημιουργία 10 ή περισσότερων δειγμάτων για κάθε συνδυασμό <Αριθμού Κόμβων, Πυκνότητας> αποσκοπεί στην ρύθμιση των τελικών αποτελεσμάτων, με τέτοιο τρόπο ώστε να είναι ενδεικτικά. Γνωρίζουμε ότι η εκθετική πολυπλοκότητα των προβλημάτων Ευσταθούς Επέκτασης καθιστά την επίλυση προβλημάτων, με μεγαλύτερο αριθμό κόμβων, ως πιο χρονοβόρα. Εν τούτοις δεν είναι σπάνιο να πάρουμε χρόνο επίλυσης προβλήματος με π.χ. 200 κόμβους, μικρότερο από χρόνο επίλυσης άλλου προβλήματος με π.χ. 100 κόμβους. Πρέπει όμως να αντιληφθούμε ότι οι εν λόγω περιπτώσεις δεν είναι ενδεικτικές, αλλά αποτελούν εξαιρέσεις, και ότι ο μόνος τρόπος να μην επηρεάσουν αρνητικά τα αποτελέσματα μας είναι με την πολλαπλή δειγματοληψία. Οι ελάχιστες τέτοιες περιπτώσεις που προκύπτουν, θα τύχουν «απόσβεσης» από τις συχνότερες, ορθές περιπτώσεις, μέσω της εύρεσης του μέσου όρου των αποτελεσμάτων του κάθε δείγματος.

- Τα δείγματα είναι τυχαιοποιημένα. Για το κάθε ένα από αυτά, δίνουμε μόνο αριθμό κόμβων και πυκνότητα, και η συνδεσιμότητα μεταξύ των κόμβων δημιουργείται τυχαία, βάσει των παραμέτρων. Έτσι διασφαλίζεται η αμεροληψία στην όλη διαδικασία.

- Διασφαλίζουμε ότι όλα τα πειράματα εκτελούνται στην ίδια μηχανή, για όλους τους συνδυασμούς <Αριθμός Κόμβων, Πυκνότητα>, για κάθε είδος προβλήματος και για κάθε εργαλείο επίλυσης. Για την πειραματική αξιολόγηση που περιγράφεται στη συνέχεια, χρησιμοποιήθηκε μία μηχανή εξυπηρετητή (server) Dell Inc. PowerEdge R610

7.4 Εκτέλεση Πειραμάτων βάσει Δειγμάτων

Ακολούθως, συνεχίζουμε με την εκτέλεση των πειραμάτων. Ζητούμε από τον ανάλογο επιλυτή, να επιλύσει τα προβλήματα που περιγράφονται στα τυχαία δείγματα που δημιουργήσαμε σε προηγούμενο στάδιο, αποθηκεύοντας τον απαιτούμενο χρόνο επίλυσης του κάθε προβλήματος. Προτιμάται, ως μονάδα μέτρησης χρόνου, τα milliseconds (ms).

7.5 Επεξεργασία Αποτελεσμάτων

Για κάθε συνδυασμό <Αριθμού Κόμβων, Πυκνότητας>, θέλουμε να βρούμε τις κατάλληλες ενδεικτικές τιμές που να αντιπροσωπεύουν το συνολικό τους αποτέλεσμα. Υπάρχουν διάφορες στατιστικές μετρικές που μπορούμε να εφαρμόσουμε στα δείγματα, ώστε να προκύψουν τα ενδεικτικά αποτελέσματα που απαιτούνται. Η σημαντικότερη, στην περίπτωση μας, αποτελεί ο Μέσος Όρος (Average) ο οποίος για κάθε συνδυασμό, μας δίνει το Μέσο Χρόνο Επίλυσης. Κοινώς, το αποτέλεσμα που μας επιστρέφει ερμηνεύεται ως «ο αναμενόμενος χρόνος εκτέλεσης, εάν δώσουμε n κόμβους και d πυκνότητα ως είσοδο», καθώς δίνει μία γενική εικόνα του ποια χρονική τιμή να αναμένουμε, για τον κάθε συνδυασμό. Εκτελείται για κάθε συνδυασμό ξεχωριστά, επομένως θα καταλήξουμε με 12 διαφορετικές τιμές μέσων όρων.

7.6 Εξαγωγή Δεδομένων για Γραφική Αναπαράσταση

Προκειμένου να καταλήξουμε στη δημιουργία ευκολονόητων γραφικών παραστάσεων, θα πρέπει να μειώσουμε τις 12 τιμές Μέσων Όρων σε ακόμα πιο μικρού πλήθους δεδομένα, τα οποία θα πρέπει να εξακολουθούν να βγάζουν νόημα και να είναι ενδεικτικά.

Για να το πετύχουμε αυτό, θα πρέπει πρώτα να θυμηθούμε ξανά τους στόχους μας. Εφόσον θέλουμε να μελετήσουμε τόσο την επίδραση του αριθμού των κόμβων, όσο και την επίδραση της πυκνότητας, στο χρόνο επίλυσης, θα πρέπει τα καινούργια δεδομένα να αναφέρονται ξεκάθαρα σε αυτά τα δύο στοιχεία. Για να γίνει αυτό, θα χρησιμοποιήσουμε και πάλι το Μέσο Όρο, ο οποίος αποτελεί την καλύτερη στατιστική μετρική, μέσω της οποίας εκφράζεται η έννοια της «ενδεικτικής τιμής» για κάποιο σύνολο αριθμών. Πιο συγκεκριμένα θα βρούμε τους Μέσους Όρους των ακόλουθων στοιχείων:

- Μέσοι Όροι χρόνων πυκνοτήτων, για κάθε αριθμό κόμβων (Παράδειγμα 7.6.1)
- Μέσοι Όροι χρόνων πληθών κόμβων, για κάθε πυκνότητα (Παράδειγμα 7.6.2)

Κοινώς, βρίσκουμε τους μέσους όρους, των μέσων όρων.

Παράδειγμα 7.6.1 (Μέσος Όρος + Πυκνότητα):

Παρουσιάζονται οι πυκνότητες (με έντονο χρώμα), στον πίνακα. Για την πυκνότητα 0.15, η τιμή 79053.27 είναι ο μέσος όρος των χρόνων όλων των συνδυασμών <Αρ. Κόμβων, Πυκνότητας>, που περιέχουν την πυκνότητα αυτή. Δηλαδή:

$$\begin{aligned} Μέσος\_Όρος(Όλοι\ οι\ Κόμβοι, 0.15) \\ &= (Μέσος\_Όρος(100, 0.15) + Μέσος\_Όρος(200, 0.15) \\ &+ Μέσος\_Όρος(300, 0.15))\ /\ 3 \end{aligned}$$

Μέσοι Όροι Χρόνων (Σύγκριση Πυκνοτήτων, για όλα τα πλήθη κόμβων)

0.15	0.20	0.25	0.30
79053.27	18988.88	17727.83	48016.75

Σχήμα 7.6.1 Μέσοι Χρόνοι Πυκνοτήτων

Πράττοντας ανάλογα για τις υπόλοιπες πυκνότητες, παίρνουμε τις τιμές που χρειαζόμαστε για τη γραφική αναπαράσταση, βάσει της οποίας θα μελετηθεί το πώς επηρεάζει το χρόνο επίλυσης, ο παράγοντας «Πυκνότητα», ανεξαρτήτως του πλήθους κόμβων.

Παράδειγμα 7.6.2 (Μέσος Όρος + Αριθμός Κόμβων):

Παρουσιάζονται τα πλήθη κόμβων (με έντονο χρώμα), στον πίνακα. Για το πλήθος κόμβων 100, η τιμή 786.10 είναι ο μέσος όρος των χρόνων όλων των συνδυασμών <Αρ. Κόμβων, Πυκνότητας>, που περιέχουν το πλήθος κόμβων αυτό. Δηλαδή:

$$\begin{aligned} &\textit{Μέσος\_Όρος}(100, \textit{Όλες οι πυκνότητες}) \\ &\quad = (\textit{Μέσος\_Όρος}(100, 0.15) + \textit{Μέσος\_Όρος}(100, 0.20) \\ &\quad + \textit{Μέσος\_Όρος}(100, 0.25) + \textit{Μέσος\_Όρος}(100, 0.30)) \;/\; 4 \end{aligned}$$

Μέσοι Όροι Χρόνων (Σύγκριση Πληθών Κόμβων, για όλες τις τιμές πυκνοτήτων)

100.00	**200.00**	**300.00**
786.10	6463.65	115590.30

Σχήμα 7.6.2 Μέσοι Χρόνοι Κόμβων

Πράττοντας ανάλογα για τα υπόλοιπα πλήθη κόμβων, παίρνουμε τις τιμές που χρειαζόμαστε για τη γραφική αναπαράσταση, βάσει της οποίας θα μελετηθεί το πώς επηρεάζει το χρόνο επίλυσης, ο παράγοντας «Πλήθος Κόμβων», ανεξαρτήτως της τιμής του συντελεστή πυκνότητας.

7.7 Πειραματική Αξιολόγηση Προβλημάτων

Για τα 5 προβλήματα που μελετήσαμε σε προηγούμενα κεφάλαια, θα διεξάγουμε πειραματική αξιολόγηση για να συγκρίνουμε το πώς διαφορετικοί παράμετροι εισόδου, σε κάθε είδος προβλήματος, επηρεάζουν τις παραμέτρους εξόδου.

7.7.1 Προτασιακή Ικανοποιησιμότητα

Αποτελέσματα Πειραματικών Δοκιμών

NODES		100.00			
DENSITY		0.15	0.20	0.25	0.30
# of Sample	1	442.00	635.00	916.00	2398.00
	2	468.00	631.00	893.00	971.00
	3	477.00	660.00	887.00	1149.00
	4	487.00	656.00	861.00	1182.00
	5	462.00	655.00	845.00	975.00
	6	458.00	650.00	761.00	946.00
	7	465.00	677.00	828.00	1021.00
	8	493.00	643.00	921.00	1149.00
	9	482.00	664.00	791.00	1168.00
	10	467.00	732.00	858.00	1131.00
	11	456.00	646.00	913.00	1159.00
	12	460.00	645.00	907.00	1055.00
	13	466.00	690.00	928.00	1022.00
	14	475.00	650.00	913.00	952.00
	15	462.00	629.00	977.00	1134.00
	16	475.00	683.00	855.00	1021.00
	17	469.00	617.00	900.00	1142.00
	18	452.00	674.00	731.00	1158.00
	19	499.00	687.00	896.00	1036.00
	20	453.00	667.00	907.00	1072.00
Avg		468.40	659.55	874.40	1142.05

Σχήμα 7.7.1.1 Δείγματα 100 κόμβων

200.00			
0.15	0.20	0.25	0.30
3731.00	3016.00	7597.00	10094.00
21011.00	4505.00	5447.00	10681.00
1991.00	2888.00	5670.00	10262.00
2488.00	15637.00	7023.00	10466.00
1557.00	10595.00	8503.00	11059.00
2759.00	3444.00	5477.00	10176.00
4108.00	3042.00	5826.00	10152.00
3028.00	4528.00	6085.00	10332.00
3989.00	4148.00	5583.00	9791.00
2842.00	7556.00	5665.00	10082.00
3489.00	4330.00	6786.00	11047.00
8480.00	4832.00	6034.00	10353.00
1719.00	5760.00	5371.00	10150.00
1971.00	3418.00	5683.00	10172.00
3015.00	3444.00	6115.00	9930.00
3156.00	3168.00	6986.00	11320.00
2563.00	2797.00	5466.00	10361.00
3084.00	3487.00	6014.00	9752.00
3153.00	5127.00	6158.00	12520.00
5788.00	4124.00	6696.00	10439.00
4196.10	4992.30	6209.25	10456.95

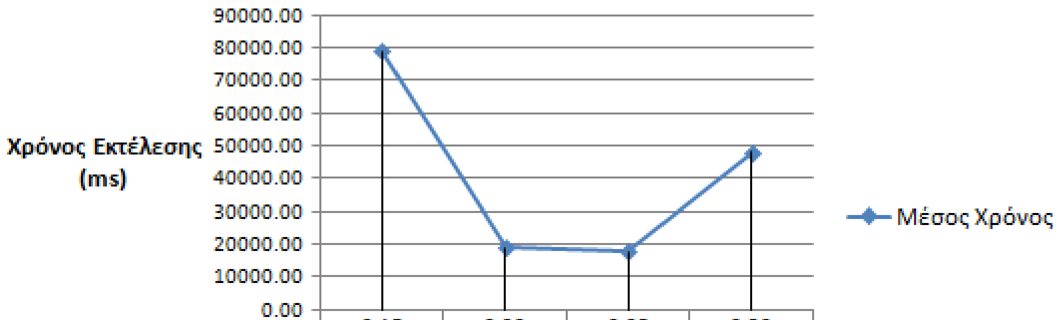
Σχήμα 7.7.1.2 Δείγματα 200 κόμβων

300.00			
0.15	0.20	0.25	0.30
50380.00	61873.00	45575.00	87890.00
83117.00	86200.00	50566.00	109964.00
6809.00	50718.00	47034.00	78195.00
70346.00	78698.00	33917.00	186610.00
195469.00	69811.00	48609.00	71020.00
663057.00	28978.00	63641.00	329903.00
266689.00	74201.00	85661.00	51172.00
70460.00	40606.00	33822.00	113630.00
1266500.00	30806.00	35540.00	61285.00
267155.00	28550.00	36734.00	533658.00
88447.00	50076.00	31843.00	192235.00
190640.00	64357.00	42664.00	139983.00
171908.00	53942.00	56348.00	64891.00
275680.00	31767.00	38509.00	68588.00
226708.00	49195.00	38045.00	21826.00
32398.00	16538.00	54165.00	47174.00
7733.00	19625.00	75499.00	63477.00
106724.00	35326.00	33809.00	260617.00
478704.00	125180.00	32704.00	67789.00
130982.00	29849.00	37312.00	99118.00
232495.30	51314.80	46099.85	132451.25

Σχήμα 7.7.1.3 Δείγματα 300 κόμβων Προτασιακής Ικανοποιησιμότητας

Για την πειραματική αξιολόγηση των προβλημάτων Ευσταθούς Επέκτασης, ως προβλήματα Προτασιακής Ικανοποιησιμότητας, χρησιμοποιήθηκαν 20 διαφορετικά δείγματα, όπως παρουσιάζονται στον πάρα πάνω πίνακα. Για τον κάθε συνδυασμό <Αρ. Κόμβων, Πυκνότητα> βρίσκουμε το Μέσο Όρο των 20 τιμών των δειγμάτων του.

Γραφική Παράσταση 7.7.1.1



	0.15	0.20	0.25	0.30
Μέσος Χρόνος Επίλυσης	79053.27	18988.88	17727.83	48016.75

Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι, αρχικά, η αύξηση του συντελεστή Πυκνότητας επιφέρει δραματική μείωση του χρόνου επίλυσης, όπως φαίνεται από την κατακόρυφη πτώση της γραμμής στη γραφική παράσταση 7.7.1.1, στα σημεία όπου έχουμε πυκνότητα από 0.15 μέχρι 0.20, με αυτή τη μείωση να «εξασθενεί» απότομα, στα σημεία όπου έχουμε πυκνότητα από 0.20 μέχρι 0.25. Η γραμμή ανακάμπτει, επιφέροντας αύξηση του χρόνου εκτέλεσης, στα σημεία όπου έχουμε πυκνότητα από 0.25 μέχρι 0.30.

Η μεταβολή της τιμής του συντελεστή Πυκνότητας επηρεάζει το χρόνο επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας, όπως μαρτυρεί η γραφική παράσταση 7.7.1.1, όμως όχι πάντα ανάλογα, ή αντιστρόφως ανάλογα. Αυτό σημαίνει ότι αύξηση του συντελεστή Πυκνότητας δεν αυξάνει πάντοτε ή μειώνει πάντοτε το χρόνο επίλυσης, όπως και ούτε μείωση του συντελεστή Πυκνότητας θα μειώσει πάντοτε ή θα αυξήσει πάντοτε, το χρόνο επίλυσης. Σε συγκεκριμένα σημεία, αύξηση της Πυκνότητας θα επιφέρει αύξηση του χρόνου εκτέλεσης (συνήθως για μικρές τιμές πυκνότητας), ενώ σε άλλα σημεία, πιθανόν να επιφέρει μείωση του (συνήθως για μεγάλες τιμές πυκνότητας).

Γραφική Παράσταση 7.7.1.2

Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι η αύξηση του αριθμού των κόμβων επιφέρει σημαντική αύξηση του χρόνου επίλυσης, όπως φαίνεται από την άνοδο της γραμμής στη γραφική παράσταση 7.7.1.2, στα σημεία με αριθμό κόμβων 100 με 200, η κλίση της οποίας αυξάνεται στα σημεία με αριθμό κόμβων από 200 μέχρι 300.

Η αύξηση του αριθμού των κόμβων επιφέρει αύξηση των χρόνων επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας. Η αυξομείωση του αριθμού των κόμβων επιφέρει πάντοτε ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης, σε εκθετική κλίμακα. Συνεπώς, μπορούμε να συμπεράνουμε ότι το Πλήθος Κόμβων του γραφήματος αποτελεί ένα ιδιαίτερα αξιόπιστο κριτήριο σύγκρισης της πολυπλοκότητας προβλημάτων Ευσταθούς Επέκτασης, στην Προτασιακή Ικανοποιησιμότητα.

7.7.2 Εξαγωγή Πυρήνα Προτάσεων/ Περιορισμών

Αποτελέσματα Πειραματικών Δοκιμών

NODES		100.00			
DENSITY		0.15	0.20	0.25	0.30
# of Sample	1.00	197.00	314.00	33.00	24.00
	2.00	20.00	198.00	57.00	41.00
	3.00	17.00	25.00	22.00	26.00
	4.00	69.00	29.00	19.00	56.00
	5.00	34.00	33.00	24.00	294.00
	6.00	173.00	29.00	37.00	20.00
	7.00	29.00	52.00	20.00	37.00
	8.00	25.00	31.00	34.00	22.00
	9.00	19.00	147.00	35.00	28.00
	10.00	49.00	18.00	24.00	31.00
	11.00	19.00	32.00	993.00	20.00
	12.00	12.00	22.00	24.00	47.00
	13.00	17.00	21.00	78.00	34.00
	14.00	148.00	21.00	61.00	30.00
	15.00	132.00	36.00	34.00	38.00
	16.00	15.00	18.00	93.00	32.00
	17.00	15.00	22.00	27.00	29.00
	18.00	28.00	614.00	24.00	763.00
	19.00	32.00	31.00	116.00	20.00
	20.00	52.00	25.00	40.00	38.00
Avg		55.10	85.90	89.75	81.50

Σχήμα 7.7.2.1 Δείγματα 100 κόμβων

200.00			
0.15	0.20	0.25	0.30
445.00	746.00	39.00	261.00
1517.00	2023.00	12093.00	176.00
5034.00	786.00	72.00	45.00
537.00	21956.00	485.00	1301.00
2872.00	1865.00	1792.00	22629.00
14655.00	311.00	199892.00	67.00
449.00	4174.00	279.00	567.00
30029.00	7992.00	67.00	446.00
345.00	41926.00	1352.00	1595.00
6234.00	2867.00	1227.00	95.00
395.00	10073.00	3017.00	393.00
316.00	3200.00	4850.00	117.00
1551.00	6628.00	850.00	596.00
1010.00	2092.00	607.00	1035.00
20618.00	2862.00	7081.00	3772.00
1128.00	1254.00	38.00	2288.00
37600.00	1703.00	1204.00	654.00
3689.00	7243.00	224.00	1492.00
1021.00	24.00	1093.00	420.00
5376.00	2637.00	625.00	473.00
6741.05	6118.10	11844.35	1921.10

Σχήμα 7.7.2.2 Δείγματα 200 κόμβων

NODES		300.00			
DENSITY		0.15	0.20	0.25	0.30
# of Sample	1.00	74262.00	155788.00	113.00	10787.00
	2.00	304734.00	145117.00	71391.00	8261.00
	3.00	1135889.00	3128.00	10618.00	71287.00
	4.00	275274.00	1624827.00	45641.00	213.00
	5.00	455458.00	362422.00	10282.00	9053.00
	6.00	397117.00	27038.00	89857.00	11185.00
	7.00	1486924.00	565378.00	110049.00	4184.00
	8.00	123910.00	8405.00	2756.00	39035.00
	9.00	663528.00	31187.00	5924.00	3769.00
	10.00	68095.00	14876.00	7787.00	104309.00
	11.00	446518.00	336703.00	8429.00	66262.00
	12.00	24028.00	6094.00	58073.00	50391.00
	13.00	838891.00	234840.00	68453.00	91.00
	14.00	246143.00	314087.00	1532.00	1681.00
	15.00	556912.00	5170222.00	12234.00	340.00
	16.00	1855285.00	22033.00	2194.00	13216.00
	17.00	3512541.00	1901772.00	193214.00	65688.00
	18.00	73467.00	189600.00	189467.00	3503.00
	19.00	2981852.00	95050.00	3993.00	146894.00
	20.00	294905.00	76899.00	2316.00	52651.00
Avg		790786.65	564273.30	44716.15	33140.00

Σχήμα 7.7.2.3 Δείγματα 300 κόμβων

Για την πειραματική αξιολόγηση των προβλημάτων εξεύρεσης Πυρήνα Προτάσεων σε γράφους, μέσω Προτασιακής Ικανοποιησιμότητας με βάρη (MAXSAT), χρησιμοποιήθηκαν 20 διαφορετικά δείγματα, όπως παρουσιάζονται στον πάρα πάνω πίνακα. Για τον κάθε συνδυασμό <Αρ. Κόμβων, Πυκνότητα> βρίσκουμε το Μέσο Όρο των 20 τιμών των δειγμάτων του.

Επεξεργασία Δεδομένων και Προετοιμασία για Γραφική Αναπαράσταση

Μέσοι Όροι Χρόνων (Σύγκριση Πυκνοτήτων, για όλα τα πλήθη κόμβων)

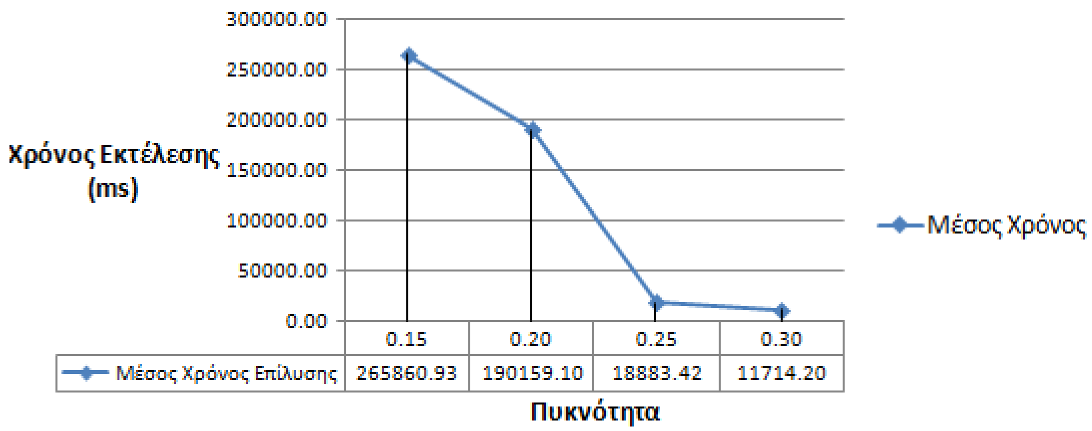
0.15	0.20	0.25	0.30
265860.93	190159.10	18883.42	11714.20

Μέσοι Όροι Χρόνων (Σύγκριση Πληθών Κόμβων, για όλες τις τιμές πυκνοτήτων)

100.00	200.00	300.00
78.06	6656.15	358229.03

Σχήμα 7.7.2.4 Μέσοι Χρόνοι

Γραφική Παράσταση 7.7.2.1



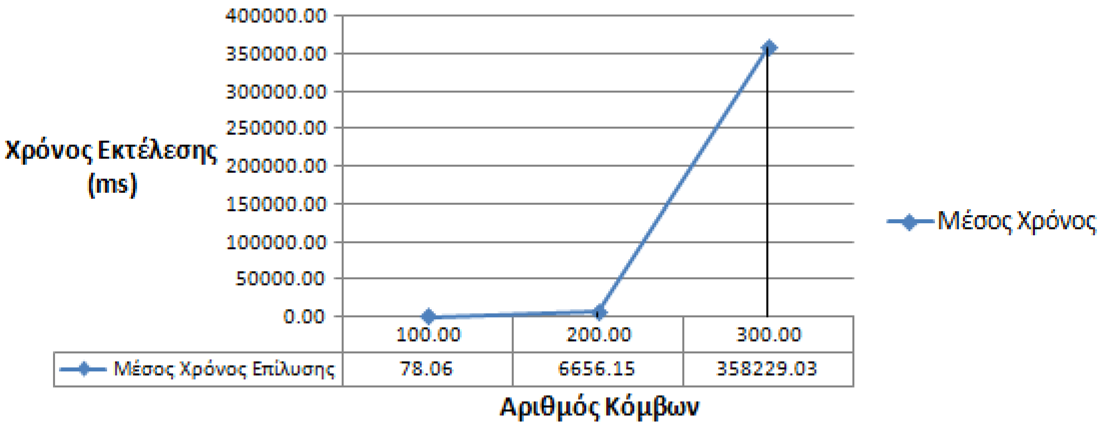
Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι η αύξηση του συντελεστή Πυκνότητας επιφέρει δραματική μείωση του χρόνου επίλυσης, όπως φαίνεται από την κατακόρυφη πτώση της γραμμής στη γραφική παράσταση 7.7.2.1, με την κλίση της γραμμής, όμως, να διαφέρει από σημείο σε σημείο.

Η μεταβολή της τιμής του συντελεστή Πυκνότητας επηρεάζει το χρόνο επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας με βάρη, όπως μαρτυρεί η γραφική παράσταση 7.7.2.1, όμως όχι πάντοτε με τον ίδιο ρυθμό (βλ. κλίση γραμμής), κάτι που δεν μας επιτρέπει να εξάγουμε σαφή συμπεράσματα, ως προς αυτό.

Παρόλο που η αυξομείωση της Πυκνότητας επιφέρει μεταβολές στις τιμές των χρόνων εκτέλεσης, όχι πάντοτε με τον ίδιο ρυθμό, εν τούτοις είναι αντιστρόφως ανάλογες σε αυτή την περίπτωση (π.χ. αύξηση πυκνότητας επιφέρει μείωση του χρόνου στην περίπτωση μας).

Γραφική Παράσταση 7.7.2.2



Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι, η αύξηση του αριθμού των κόμβων επιφέρει δραματική αύξηση του χρόνου επίλυσης, όπως φαίνεται από την ελάχιστη άνοδο της γραμμής στη γραφική παράσταση 7.7.2.2, στα σημεία με αριθμό κόμβων 100 με 200, η κλίση της οποίας αυξάνεται δραματικά, όπως φαίνεται στα σημεία με αριθμό κόμβων από 200 μέχρι 300.

Η αύξηση του αριθμού των κόμβων επιφέρει αύξηση των χρόνων επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας με βάρη. Η αυξομείωση του αριθμού των κόμβων επιφέρει πάντοτε ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης, σε εκθετική κλίμακα (αύξηση φέρει αύξηση, μείωση φέρει μείωση). Συνεπώς, μπορούμε να συμπεράνουμε ότι το Πλήθος Κόμβων του γραφήματος αποτελεί ένα ιδιαίτερα αξιόπιστο κριτήριο σύγκρισης της πολυπλοκότητας προβλημάτων Ευσταθούς Επέκτασης, στην Προτασιακή Ικανοποιησιμότητα με βάρη (MAXSAT).

7.7.3 Εξαγωγή Πυρήνα Επιχειρημάτων

Αποτελέσματα Πειραματικών Δοκιμών

NODES		100.00				200.00				300.00			
DENSITY		0.15	0.20	0.25	0.30	0.15	0.20	0.25	0.30	0.15	0.20	0.25	0.30
# of Sample	1.00	221.00	134.00	134.00	122.00	559.00	1286.00	774.00	1444.00	195629.00	10290.00	5810.00	1306.00
	2.00	142.00	123.00	122.00	371.00	1395.00	876.00	304.00	539.00	77157.00	128964.00	1452.00	6152.00
	3.00	129.00	124.00	126.00	141.00	3181.00	2661.00	3595.00	1467.00	54839.00	3796.00	8728.00	2257.00
	4.00	132.00	142.00	132.00	138.00	3600.00	1320.00	2464.00	990.00	45361.00	141873.00	4733.00	2488.00
	5.00	128.00	74.00	135.00	127.00	637.00	2922.00	1286.00	1176.00	228748.00	444214.00	2862.00	3048.00
	6.00	127.00	141.00	124.00	124.00	4993.00	4262.00	617.00	506.00	9408.00	10717.00	53136.00	781.00
	7.00	120.00	124.00	148.00	132.00	1161.00	1582.00	719.00	1050.00	152666.00	7437.00	14360.00	1757.00
	8.00	135.00	118.00	131.00	126.00	1087.00	1070.00	1134.00	1237.00	288888.00	2625.00	19697.00	4476.00
	9.00	123.00	182.00	130.00	147.00	2120.00	1561.00	1186.00	668.00	259144.00	42038.00	29357.00	4736.00
	10.00	162.00	144.00	163.00	143.00	1691.00	4361.00	1180.00	460.00	14508.00	2066.00	1368.00	3503.00
	11.00	149.00	152.00	174.00	132.00	501.00	1502.00	2061.00	826.00	234411.00	32307.00	19483.00	4537.00
	12.00	121.00	134.00	153.00	128.00	522.00	580.00	1127.00	1204.00	538925.00	43052.00	3531.00	2966.00
	13.00	162.00	130.00	264.00	126.00	1709.00	1027.00	800.00	1051.00	45314.00	5512.00	1060.00	3662.00
	14.00	149.00	171.00	136.00	157.00	2050.00	265.00	1730.00	1336.00	1093.00	12681.00	4242.00	2729.00
	15.00	126.00	162.00	175.00	128.00	1209.00	1420.00	1745.00	1035.00	36688.00	6553.00	31106.00	3690.00
	16.00	120.00	128.00	134.00	146.00	244.00	1104.00	534.00	546.00	14531.00	2262.00	4745.00	2483.00
	17.00	143.00	139.00	118.00	141.00	2445.00	2555.00	535.00	1101.00	123352.00	4706.00	25580.00	3064.00
	18.00	131.00	158.00	356.00	123.00	2317.00	79875.00	467.00	1037.00	9663.00	18147.00	3195.00	2143.00
	19.00	151.00	121.00	157.00	146.00	2980.00	7568.00	581.00	890.00	7703.00	29334.00	4179.00	4010.00
	20.00	212.00	136.00	127.00	163.00	3946.00	1191.00	901.00	1127.00	70504.00	4990.00	10476.00	2362.00
Avg		144.15	136.85	156.95	148.05	1917.35	5949.40	1187.00	984.50	120426.60	47678.20	12455.00	3107.50

Σχήμα 7.7.3.1 Αποτελέσματα Πειραματικών Δοκιμών Εξαγωγής Πυρήνα Κόμβων

Επεξεργασία Δεδομένων και Προετοιμασία για Γραφική Αναπαράσταση

Μελέτη Πυκνότητας ως κριτήριο σύγκρισης προβλημάτων Ευσταθούς Επέκτασης

Μέσοι Όροι Χρόνων (Σύγκριση Πυκνοτήτων, για όλα τα πλήθη κόμβων)

0.15	0.20	0.25	0.30
40829.37	17921.48	4599.65	1413.35

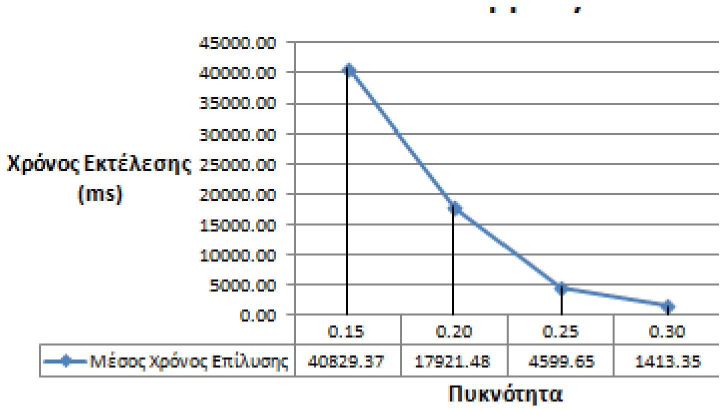
Μελέτη Αριθμού Κόμβων ως κριτήριο σύγκρισης προβλημάτων Ευσταθούς Επέκτασης

Μέσοι Όροι Χρόνων (Σύγκριση Πληθών Κόμβων, για όλες τις τιμές πυκνοτήτων)

100.00	200.00	300.00
146.50	2509.56	45916.83

Σχήμα 7.7.3.2 Μέσοι Χρόνοι

Γραφική Παράσταση 7.7.3.1

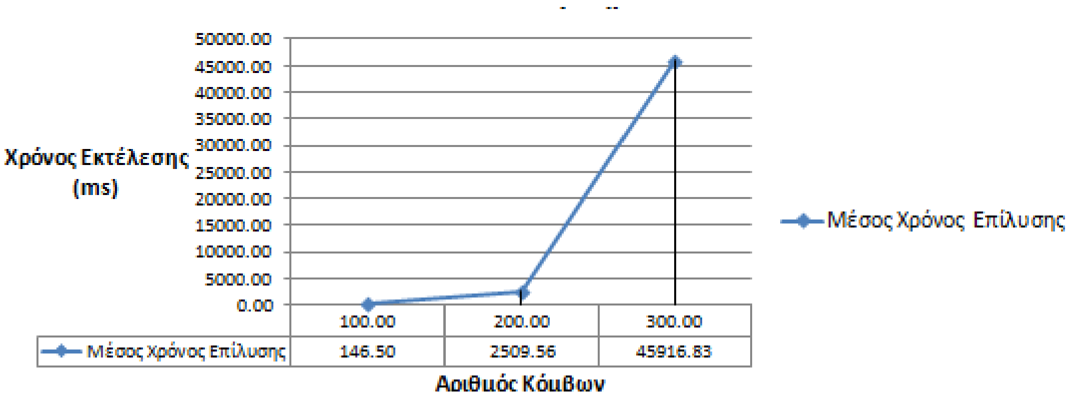


Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι η αύξηση του συντελεστή Πυκνότητας επιφέρει σημαντική μείωση του χρόνου επίλυσης, όπως φαίνεται από την κατακόρυφη πτώση της γραμμής στη γραφική παράσταση 7.7.3.1.

Η μεταβολή της τιμής του συντελεστή Πυκνότητας επηρεάζει το χρόνο επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας με βάρη, όπως μαρτυρούν οι γραφικές παραστάσεις, για το εν λόγω πρόβλημα. Η αυξομείωση της Πυκνότητας επιφέρει ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης, για το πρόβλημα Εξαγωγής Πυρήνα Επιχειρημάτων. Αυτό οφείλεται στο γεγονός ότι σε προβλήματα με αυξημένη πυκνότητα, βρίσκουμε πιο εύκολα, άρα και σχετικά γρήγορα, επιχειρήματα που προκαλούν μη-Ικανοποιησιμότητα, λόγω της πυκνής συνδεσιμότητας του γραφήματος, και των συσχετίσεων τους.

Γραφική Παράσταση 7.7.3.2



Παρατηρήσεις/ Συμπεράσματα

Παρατηρούμε ότι η αύξηση του αριθμού των κόμβων επιφέρει αύξηση του χρόνου επίλυσης, όπως φαίνεται από την άνοδο της γραμμής στη γραφική παράσταση 7.7.3.2, στα σημεία με αριθμό κόμβων 100 με 200, η κλίση της οποίας αυξάνεται, όπως φαίνεται στα σημεία με αριθμό κόμβων από 200 μέχρι 300.

Η αύξηση του αριθμού των κόμβων επιφέρει εκθετική αύξηση του χρόνου επίλυσης προβλημάτων Προτασιακής Ικανοποιησιμότητας με βάρη (MAXSAT). Η αυξομείωση του αριθμού των κόμβων επιφέρει πάντοτε ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης, σε εκθετική κλίμακα.

7.7.4 Ακέραιος Γραμμικός Προγραμματισμός

Αποτελέσματα Πειραματικών Δοκιμών

NODES		100			
DENSITY		0.15	0.2	0.25	0.3
# of sample	1	22310.409	62586.311	132360	75958.805
	2	18121.135	75628.793	78167.628	91732.804
	3	19363.859	-	-	74157.846
	4	14888.633	42723.428	-	122572.652
	5	21775.688	-	61477.065	74167.447
	6	16328.433	-	112968.21	-
	7	18565.471	78641.904	69150.384	65259.497
	8	23984.15	-	-	33195.668
	9	20235.372	-	-	-
	10	21728.672	-	73232.594	71441.071

Σχήμα 7.7.4.1 Αποτελέσματα Πειραμάτων Γραμμικού Προγραμματισμού

Για την πειραματική αξιολόγηση των προβλημάτων εξεύρεσης Ευσταθούς Επέκτασης, σε Ακέραιο Γραμμικό Προγραμματισμό, χρησιμοποιήθηκαν 10 διαφορετικά δείγματα (στιγμιότυπα αυτού του είδους προβλημάτων), όπως παρουσιάζονται στον πάρα πάνω πίνακα. Ο λόγος που μειώθηκε ο αριθμός των δειγμάτων είναι τα μεγάλα χρονικά διαστήματα που χρειάζονται για την εκτέλεση τέτοιων προβλημάτων από τον επιλυτή intlinprog της Matlab. Το πλήθος των δειγμάτων παραμένει αρκετά ικανοποιητικό, ώστε να έχουμε αντικειμενικά αποτελέσματα στο τέλος. Για κάθε συνδυασμό <Αρ. Κόμβων, Πυκνότητα> βρίσκουμε το Μέσο Όρο των 10 τιμών των δειγμάτων του.

Σημαντική Σημείωση #1: Λόγω των υπερβολικά χρονοβόρων εκτελέσεων των πλείστων προβλημάτων (ιδιαίτερα για μεγάλο αριθμό κόμβων), τα οποία πιο συχνά κατέληγαν στην «συντριβή» (crashing) του προγράμματος Matlab, χρησιμοποιήθηκε ο ακόλουθος χειρισμός. Τέθηκε ένα χρονικό όριο «ανοχής» (timeout threshold) στα 30 λεπτά (=1800000 ms), για το οποίο, οποιαδήποτε διαδικασία (process) επίλυσης δεν ολοκληρωνόταν εντός αυτού, τύχαινε ακαριαίας διακοπής, και προσδιδόταν στο πρόβλημα η κατάσταση «UNKNOWN» (άγνωστο).

Σημαντική Σημείωση #2: Το '-' ανατίθεται στα δείγματα των οποίων ο πραγματικός χρόνος ξεπερνά τον προκαθορισμένο χρόνο διακοπής διεργασιών (process timeout threshold) των 30 λεπτών (=1800000 ms), δηλαδή στα προβλήματα με κατάσταση «UNKNOWN». Αυτό γίνεται για σκοπούς δημιουργίας γραφημάτων, καθώς δεν μπορούμε να αφήσουμε κενές τιμές, ούτε όμως να αναλώσουμε τεράστιου (και αβέβαιου) μεγέθους χρόνο, στην επίλυση προβλημάτων με πολλούς κόμβους (ειδικά εφόσον από την περίπτωση των 100 κόμβων, εμφανίζονται ήδη οι πρώτες ενδείξεις χρονοβόρων εκτελέσεων). Όπου '-', υπολογίζουμε την τιμή 1800000 ms.

Επεξεργασία Δεδομένων και Προετοιμασία για Γραφική Αναπαράσταση

Μέσοι Όροι Χρόνων (Σύγκριση Πυκνοτήτων, για όλα τα πλήθη κόμβων)

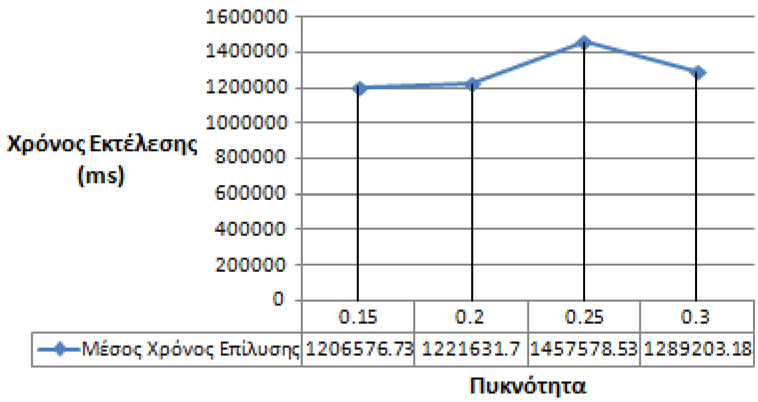
0.15	0.2	0.25	0.3
1206576.727	1221631.703	1457579	1E+06

Μέσοι Όροι Χρόνων (Σύγκριση Πληθών Κόμβων, για όλες τις τιμές πυκνοτήτων)

100		
281242.6029		

Σχήμα 7.7.4.2 Μέσοι Χρόνοι

Γραφική Παράσταση 7.7.4.1

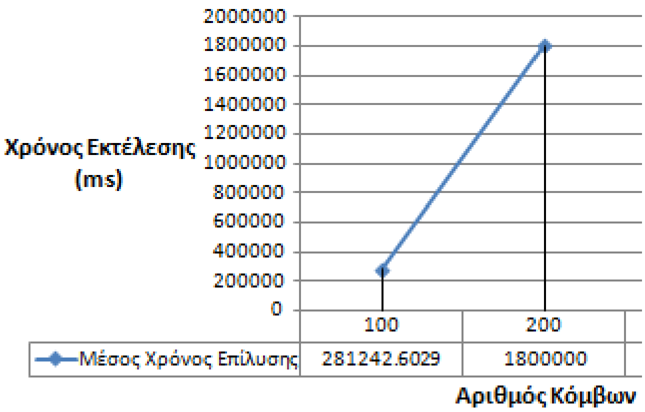


Παρατηρήσεις/ Συμπεράσματα

Στη γραφική παράσταση 7.7.4.1 μπορούμε να διακρίνουμε μία γραμμική αύξηση των χρόνων εκτέλεσης, αυξάνοντας την πυκνότητα από 0.15 σε 0.2 και από 0.2 σε 0.25, σημείο από το οποίο και μετά, κάθε διαδοχική αύξηση της τιμής της πυκνότητας, μειώνει το χρόνο εκτέλεσης του προβλήματος (γραμμική μείωση, βλ. κλίση και κατεύθυνση γραμμής στο διάστημα από 0.25 μέχρι και 0.30). Αξιοσημείωτο είναι το γεγονός ότι εξ' αρχής, οι χρόνοι εκτέλεσης είναι αρκετά μεγάλοι (βλ. κατακόρυφο άξονα Υ), ενώ οι χρόνοι εκτέλεσης δεν αυξάνονται δραματικά για την αρχική αύξηση πυκνότητας (από 0.15 σε 0.2).

Με την αλλαγή της πυκνότητας ενός προβλήματος Ευσταθούς Επέκτασης, στον Ακέραιο Γραμμικό Προγραμματισμό, γίνονται μη-ανάλογες μεταβολές στο χρόνο εκτέλεσης, οι οποίες εξαρτούνται και από το σημείο που προσπαθεί να μεταβληθεί η πυκνότητα π.χ. αύξηση πυκνότητας με ήδη μικρή τιμή επιφέρει αύξηση του χρόνου επίλυσης, ενώ αύξηση πυκνότητας με ήδη μεγάλη τιμή επιφέρει μείωση του χρόνου επίλυσης.

Γραφική Παράσταση 7.7.4.2



Παρατηρήσεις/ Συμπεράσματα

Στη γραφική παράσταση 7.7.4.2 παρατηρούμε ότι η αύξηση του πλήθους των κόμβων επιφέρει και αύξηση του χρόνου επίλυσης (βλ. διάστημα 100-200). Ήδη από τις περιπτώσεις προβλημάτων με 200 κόμβους, ο επιλυτής αδυνατούσε να επιλύσει τα δείγματα, πριν το διάστημα των 30 λεπτών. Δεν είναι αξιόπιστο το αποτέλεσμα για 200 και 300 κόμβους. Επομένως, εφόσον ήδη στα προβλήματα 200 κόμβων ανατέθηκε ως χρόνος εκτέλεσης, η μέγιστη δυνατή τιμή (1800 seconds = 1800000 milliseconds), υπάρχει αδυναμία ανάθεσης μεγαλύτερης τιμής στους χρόνους εκτέλεσης προβλημάτων με περισσότερους κόμβους. Ωστόσο, εύλογο είναι να αναφέρουμε ότι στην πραγματικότητα ο χρόνος εκτέλεσης προβλημάτων με 300 κόμβους αναμένεται να ξεπεράσει το χρόνο εκτέλεσης προβλημάτων με 200 κόμβους, σύμφωνα με την αυξητική τάση που παρουσιάζει η γραφική παράσταση 7.7.4.2 (αγνοώντας το διάστημα 200-300 κόμβων) και το γεγονός ότι μελετούμε προβλήματα εκθετικής πολυπλοκότητας.

Η αύξηση του πλήθους των κόμβων αποτελεί ένα αρκετά αξιόπιστο και συνεπές κριτήριο σύγκρισης της πολυπλοκότητας μεταξύ προβλημάτων Ευσταθούς Επέκτασης, σε Ακέραιο Γραμμικό Προγραμματισμό. Αυτό οφείλεται στο γεγονός ότι η αυξομείωση της παραμέτρου αυτής, συνεπάγεται με ανάλογη μεταβολή του χρόνου εκτέλεσης. Π.χ. αύξηση αριθμού κόμβων επιφέρει πάντοτε (εκτός στις εξαιρετικές περιπτώσεις εμφάνισης πειραματικού σφάλματος) και αύξηση του χρόνου εκτέλεσης, ενώ η μείωση του αριθμού κόμβων επιφέρει πάντοτε και μείωση του χρόνου εκτέλεσης. Σύγκριση με lingeling δείχνει, σαφώς, καλύτερες επιδόσεις για τον lingeling.

7.7.5 Μεικτός-Ακέραιος Γραμμικός Προγραμματισμός

Αποτελέσματα Πειραματικών Δοκιμών

NODES			100				200				300			
DENSITY			0.15	0.2	0.25	0.3	0.15	0.2	0.25	0.3	0.15	0.2	0.25	0.3
# of sample	ε = 0.2	1	933.11	65.70	90.96	102.06	1619.21	913.04	407.76	605.09	487.50	5308.10	869.76	785.15
	ε = 0.2	2	121.64	60.56	53.12	139.02	593.51	561.31	365.37	1030.64	488.98	4615.87	930.26	873.49
	ε = 0.2	3	99.99	108.00	84.64	83.32	1167.76	374.45	372.88	570.82	513.40	4321.11	770.23	873.50
	ε = 0.2	4	67.99	61.78	73.46	113.04	1737.68	1989.71	1295.43	721.69	775.62	5293.76	5677.98	1127.13
	ε = 0.2	5	148.09	59.46	58.93	63.42	271.60	370.30	330.69	450.65	555.98	4297.83	772.08	728.66
	min(ε)	6	13666.67	94930.44	5393.00	16787.87	-	-	-	-	-	-	-	-
	min(ε)	7	77141.64	13097.49	61654.41	44446.47	-	-	1247755.28	1791784	-	-	-	-
	min(ε)	8	11394.50	43479.49	67164.59	131834.02	-	-	-	754757.04	-	-	-	-
	min(ε)	9	42347.67	5504.14	17287.74	54426.63	-	-	1009586.89	600343.41	-	-	-	-
	min(ε)	10	1390.02	41990.80	29372.97	34239.17	334805.94	-	1871574.05	1240426.96	-	-	-	-

Σχήμα 7.7.5.1 Αποτελέσματα Πειραμάτων Μεικτού Γραμμικού Προγραμματισμού

Για την πειραματική αξιολόγηση των προβλημάτων εξεύρεσης Ευσταθούς Επέκτασης και Χειρισμού Μη-Ικανοποιησιμότητας, σε Μεικτό Γραμμικό Προγραμματισμό (MILP), ισχύουν όλα τα μέτρα που λάβαμε και για την περίπτωση του Ακέραιου Γραμμικού Προγραμματισμού (βλ. 7.7.4, Σημαντικές Σημειώσεις). Ταυτόχρονα, λόγω της ύπαρξης επιπλέον μεταβλητών, ενδιαφέρον έχει να μελετήσουμε την επίδραση που επιφέρει στο χρόνο επίλυσης, η προσπάθεια ελαχιστοποίησης της μέγιστης δυνατής απόστασης από τα άκρα (μεταβλητή ε), σε σύγκριση με την απευθείας ανάθεση της τιμής της. Πιο συγκεκριμένα ισχύουν οι πάρα κάτω επιπλέον σημειώσεις, σε συνδυασμό με αυτές του υποκεφαλαίου 7.7.4.:

Σημαντική Σημείωση #3: Για να ενισχυθεί η αμεροληψία και να παραμείνουν ενδεικτικά τα τελικά αποτελέσματα, δημιουργούμε τα μισά δείγματα (5) με σταθερή ανάθεση τιμής για τη μεταβλητή ε (όπου ε = 0.2) και τα άλλα μισά δείγματα (υπόλοιπα 5) με ανάθεση της ελάχιστης δυνατής τιμής για την εν λόγω μεταβλητή (όπου ε = min(ε)).

Σημαντική Σημείωση #4: Για σκοπούς δημιουργίας γραφικών παραστάσεων και μελέτης της επίδρασης των διαφορετικών τιμών του ε (σταθερά και ελαχιστοποιημένη), βρίσκουμε μέσους όρους για τον κάθε συνδυασμό <Αρ. Κόμβων, Πυκνότητα, ε>, όπως φαίνεται στον πίνακα πάρα πάνω (Αποτελέσματα Πειραματικών Δοκιμών).

Επεξεργασία Δεδομένων και Προετοιμασία για Γραφική Αναπαράσταση

Μέσοι Όροι Χρόνων (Σύγκριση Πυκνοτήτων, για όλα τα πλήθη κόμβων)

0.15	0.2	0.25	0.3
556344.28	607580.11	564065	515910

Μέσοι Όροι Χρόνων (Σύγκριση Πληθών Κόμβων, για όλες τις τιμές πυκνοτήτων)

100	200	300
20253.45	761669.5791	901002

Μέσοι Όροι Χρόνων (Δοθέντος ε = 0.2)

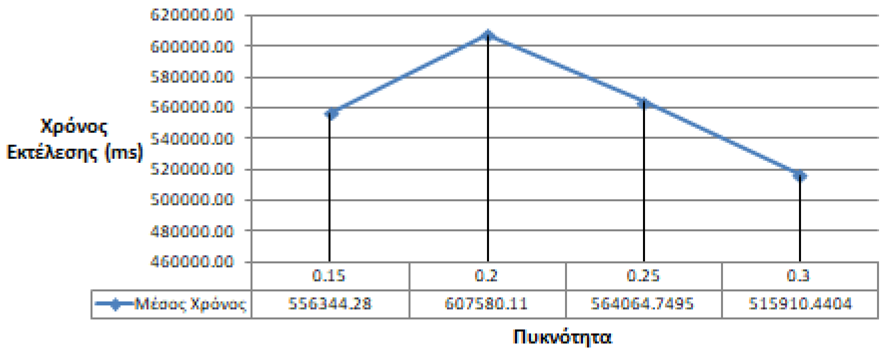
100	200	300
129.41	787.48	2003.32

Μέσοι Όροι Χρόνων (Ελαχιστοποιημένου ε)

100	200	300
40377.49	1522551.678	1800000

Σχήμα 7.7.5.2 Μέσοι Χρόνοι

Γραφική Παράσταση 7.7.5.1

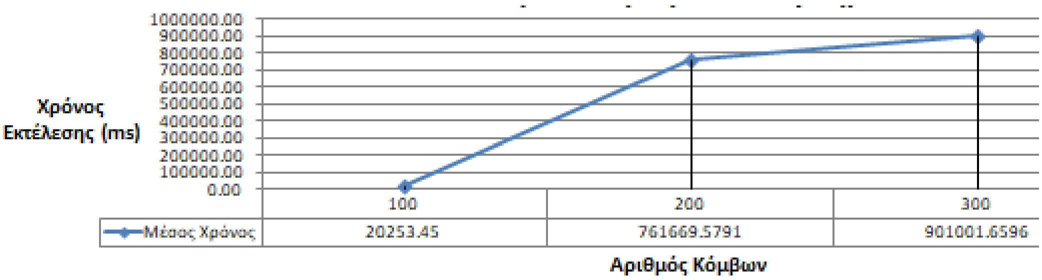


Παρατηρήσεις/ Συμπεράσματα

Η αύξηση της τιμής τους συντελεστής Πυκνότητας από 0.15 σε 0.2 οδήγησε σε αύξηση του χρόνου εκτέλεσης (βλ. κλίση γραμμής, γραφική παράσταση 7.7.5.1). Παρόλα αυτά, η κάθε μετέπειτα αύξηση επέφερε μείωση του χρόνου εκτέλεσης, όπως μαρτυρεί η καθοδική πορεία της γραμμής.

Η πυκνότητα επηρεάζει το χρόνο εκτέλεσης στα προβλήματα Μεικτού Γραμμικού Προγραμματισμού, όχι όμως πάντοτε ανάλογα ή αντιστρόφως ανάλογα, καθώς εξαρτάται και από το σημείο στο οποίο συμβαίνει η αύξηση/ μείωση της τιμής της. Αύξηση μίας ήδη μικρής τιμής πυκνότητας, επιφέρει αύξηση του χρόνου εκτέλεσης (και αντιστοίχως, με μείωση επιφέρεται μείωση), ενώ αύξηση μίας ήδη μεγάλης τιμής πυκνότητας, επιφέρει μείωση του χρόνου εκτέλεσης (και αντιστοίχως, με μείωση επιφέρεται αύξηση) (π.χ. αύξηση στο σημείο 0.15 οδήγησε σε αύξηση του χρόνου επίλυσης, ενώ αύξηση στο σημείο 0.25 οδήγησε σε μείωση).

Γραφική Παράσταση 7.7.5.2

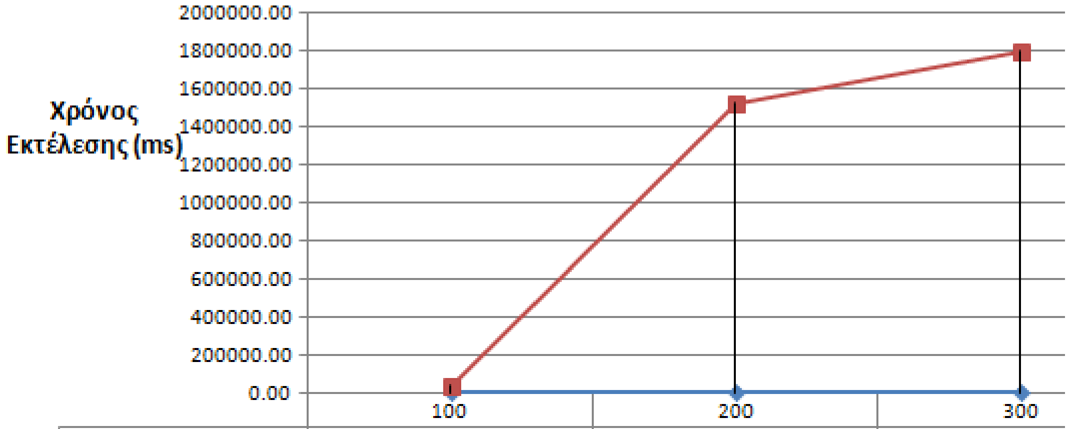


Παρατηρήσεις / Συμπεράσματα

Παρατηρούμε ότι η αύξηση του αριθμού των κόμβων επιφέρει αύξηση του χρόνου επίλυσης, όπως φαίνεται από την άνοδο της γραμμής στη γραφική παράσταση 7.7.5.2, καθ' όλη την πορεία της.

Η αύξηση του αριθμού των κόμβων επιφέρει αύξηση του χρόνου επίλυσης των προβλημάτων Μεικτού Ακέραιου Προγραμματισμού. Η αυξομείωση του αριθμού των κόμβων επιφέρει πάντοτε ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης (δηλ. αύξηση επιφέρει αύξηση, μείωση επιφέρει μείωση). Συνεπώς, μπορούμε να συμπεράνουμε ότι το Πλήθος Κόμβων του γραφήματος αποτελεί ένα ιδιαίτερα αξιόπιστο κριτήριο σύγκρισης της πολυπλοκότητας.

Γραφική Παράσταση 7.7.5.3



	100	200	300
Δοθέν ε = 0.2	129.41	787.48	2003.32
Ελαχιστοποιημένο ε	40377.49	1522551.678	1800000

Αριθμός Κόμβων

Παρατηρήσεις / Συμπεράσματα

Για την περίπτωση της ελαχιστοποιημένης τιμής της μεταβλητής μέγιστης απόστασης (ε), παρατηρούμε την ευδιάκριτη, εξαιρετικά μεγάλης κλίμακας αύξηση στο χρόνο επίλυσης, με κάθε αύξηση του αριθμού των κόμβων. Αυξητική τάση παρουσιάζει και η περίπτωση όπου δινόταν ως σταθερά, η τιμή της μεταβλητής ε, ωστόσο δεν είναι διακριτή στη γραφική παράσταση 7.7.5.3 (διακρίνονται όμως στον πίνακα, κάτωθεν του άξονα X, της γραφικής παράστασης 7.7.5.3).

Ο λόγος που επιλέχτηκε αυτή η μορφή γραφικής παράστασης είναι ότι, από τη στιγμή που ήδη γνωρίζουμε ότι το πλήθος των κόμβων επηρεάζει ανάλογα το χρόνο εκτέλεσης, μας ενδιαφέρει μόνο να δούμε τη σύγκριση καθαρά μεταξύ των τρόπων χειρισμού της μεταβλητής ε. Με την ευδιάκριτη διαφορά μεταξύ των δύο γραμμών (η μία παρουσιάζει τεράστια μεταβολή, στην άλλη δεν διακρίνεται καν η μεταβολή της) μπορούμε να συμπεράνουμε ποιος χειρισμός της μεταβλητής ε είναι πιο δαπανηρός, χρονικά.

Η προσπάθεια ελαχιστοποίησης της μέγιστης δυνατής απόστασης από τα άκρα (τιμή ε), επιφέρει μεγάλο χρονικό κόστος (αυξάνει δηλαδή το χρόνο εκτέλεσης, με κάθε αύξηση των κόμβων), σε υπερβολικά μεγαλύτερο βαθμό από την περίπτωση της ανάθεσης μίας σταθεράς τιμής χωρίς βελτιστοποιήσεις. Αυτό οφείλεται στο γεγονός ότι ο επιλυτής δαπανεί αρκετό χρόνο ώστε να βρει την βέλτιστη λύση για το πρόβλημα, δηλαδή αυτή που περιέχει την ελάχιστη δυνατή τιμή για την ε.

Αντίθετα, στην περίπτωση της σταθερής ανάθεσης τιμής, αρκείται στην απλή ανάθεση τιμών σε επιχειρήματα, οι οποίες εμπίπτουν σε πιο χαλαρά όρια (αφού η τιμή της ε μπορεί να είναι αρκετά μεγάλη, άρα να επιτρέπει μεγαλύτερο πεδίο τιμών για τα επιχειρήματα), εφόσον ήδη γνωρίζει την τιμή της μεταβλητής ε (δεν προσπαθεί να την υπολογίσει).

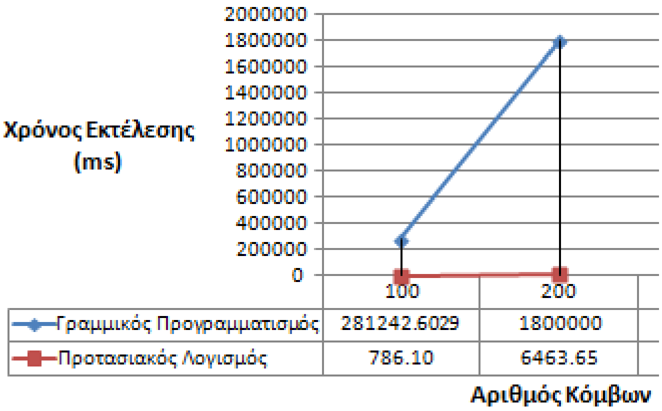
7.7.6 Σύγκριση Εργαλείων για Ευσταθή Επέκταση

Όπως αναφέρθηκε και στο σημείο 7.1, η πειραματική αξιολόγηση μπορεί να αξιοποιηθεί και για τη σύγκριση μεταξύ εργαλείων που επιλύνουν παρόμοια προβλήματα, συνήθως για να διαπιστωθεί το αποδοτικότερο εξ' αυτών (όσον αφορά το χρόνο εκτέλεσης).

Στην περίπτωση μας χρησιμοποιήθηκαν δύο διαφορετικά εργαλεία για το πρόβλημα εξεύρεσης της Ευσταθούς Επέκτασης, με παρόμοιο τρόπο, δηλαδή με απώτερο σκοπό την εξεύρεση αποδεκτής λύσης, με ανάθεση τιμών 0 ή 1 στα επιχειρήματα. Αυτά τα εργαλεία είναι ο Προτασιακός Λογισμός (βλ. 7.7.1) και ο Ακέραιος Γραμμικός Προγραμματισμός (βλ. 7.7.4). Οι δύο αυτές προσεγγίσεις θα μελετηθούν ως προς τους χρόνους εκτέλεσης, τόσο για τον παράγοντα «Αριθμός Κόμβων», όσο και για τον παράγοντα «Πυκνότητα Γραφήματος» με σκοπό τη σύγκρισή τους.

7.7.6.1 Σύγκριση για «Αριθμό Κόμβων»

Γραφική Παράσταση 7.7.6.1.1 Σύγκριση Προτασ.Λογ./ Γραμμικού.Προγρ.



Παρατηρήσεις/ Συμπεράσματα

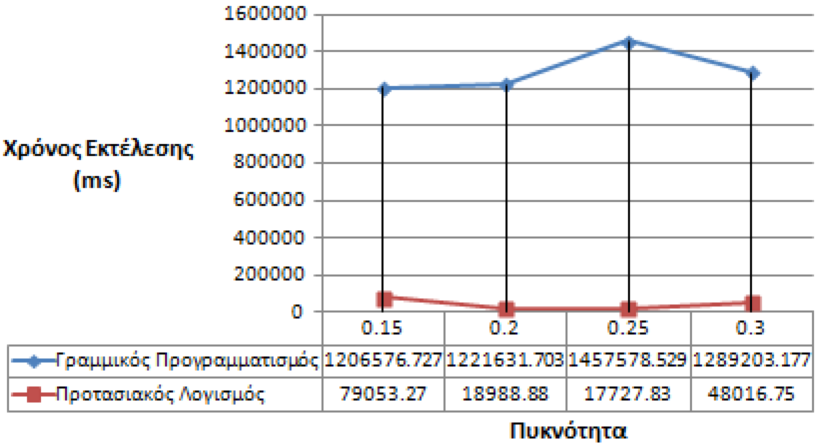
Τόσο στην περίπτωση του Προτασιακού Λογισμού, όσο και στην περίπτωση του Ακέραιου Γραμμικού Προγραμματισμού, παρουσιάζεται θετική κλίση της γραμμής στη γραφική παράσταση (βλ. 7.7.6.1.1), κάτι που υποδηλώνει αντίστοιχη μεταβολή του χρόνου εκτέλεσης, με την αλλαγή του αριθμού κόμβων (αύξηση κόμβων επιφέρει αύξηση χρόνου επίλυσης, μείωση κόμβων επιφέρει μείωση χρόνου επίλυσης). Παρόλα αυτά όμως, παρατηρούμε τις τιμές του άξονα Υ, όπου παρουσιάζονται οι χρόνοι εκτέλεσης. Πιο συγκεκριμένα, στην περίπτωση του Προτασιακού Λογισμού έχουμε πολύ μικρότερες τιμές χρόνων εκτέλεσης, σε κάθε στάδιο της γραφικής παράστασης, απ' ότι στο κάθε αντίστοιχο στάδιο, στην περίπτωση του Ακέραιου Γραμμικού Προγραμματισμού.

Αμφότερες οι περιπτώσεις παρουσιάζουν παρόμοια συμπεριφορά, δηλαδή η μεταβολή του πλήθους των κόμβων επιφέρει αντίστοιχη μεταβολή και στο χρόνο εκτέλεσης. Όμως, ο Ακέραιος Γραμμικός Προγραμματισμός δαπανεί πολύ περισσότερο χρονικό διάστημα για να επιλύσει προβλήματα, των οποίων η επίλυση θα ήταν πολύ πιο σύντομη με τη χρήση του Προτασιακού Λογισμού. Επομένως, βάσει των ευρημάτων καταλήγουμε στο συμπέρασμα ότι η μοντελοποίηση ενός προβλήματος Ευσταθούς Επέκτασης σε Προτασιακή

Λογική είναι καλύτερη επιλογή μεταξύ των δύο, εάν σκοπός είναι η ελάχιστη δυνατή χρονική δαπάνη.

7.7.6.2 Σύγκριση για «Πυκνότητα Γραφήματος»

Γραφική Παράσταση 7.7.6.2.1 Σύγκριση Προτασ.Λογ./ Γραμμικού.Προγρ.



	0.15	0.2	0.25	0.3
Γραμμικός Προγραμματισμός	1206576.727	1221631.703	1457578.529	1289203.177
Προτασιακός Λογισμός	79053.27	18988.88	17727.83	48016.75

Παρατηρήσεις / Συμπεράσματα

Η μεταβολή στις τιμές των πυκνοτήτων επιφέρει διαφορετικές αλλαγές στους χρόνους επίλυσης, όπως παρατηρούμε από τη συμπεριφορά των γραμμών (βλ. 7.7.6.2.1). Παρόλα αυτά, οι χρόνοι εκτέλεσης, καθ' όλη την πειραματική διαδικασία, έχουν συνεχώς μεγαλύτερη τιμή στην περίπτωση του Ακέραιου Γραμμικού Προγραμματισμού (βλ. κατακόρυφο άξονα Υ).

Ο Ακέραιος Γραμμικός Προγραμματισμός δαπανεί πολύ περισσότερο χρονικό διάστημα για να επιλύσει προβλήματα, των οποίων η επίλυση θα ήταν πολύ πιο σύντομη με τη χρήση του Προτασιακού Λογισμού. Επομένως, βάσει των ευρημάτων καταλήγουμε στο συμπέρασμα ότι η μοντελοποίηση ενός προβλήματος Ευσταθούς Επέκτασης σε Προτασιακή Λογική είναι καλύτερη επιλογή μεταξύ των δύο, εάν σκοπός είναι η ελάχιστη δυνατή χρονική δαπάνη.

7.8 Γενικά Συμπεράσματα και Σημειώσεις [1]

Από όλα τα πάρα πάνω πειράματα πηγάζουν κάποια γενικά συμπεράσματα που ισχύουν για κάθε είδος προβλήματος, και για κάθε εργαλείο επίλυσης.

Το πλήθος των μεταβλητών εισόδου σε ένα πρόβλημα Ικανοποίησης Περιορισμών αποτελεί τον παράγοντα που επηρεάζει το χρόνο εκτέλεσης. Πιο συγκεκριμένα, εφόσον τα προβλήματα που μελετήθηκαν έχουν εκθετική πολυπλοκότητα (εφόσον ανήκουν στα NP-Hard προβλήματα), η αύξηση του αριθμού των μεταβλητών, όπου στην δική μας περίπτωση, μεταβλητές αποτελούν οι κόμβοι του γραφήματος, φαίνεται να επιφέρει σημαντική αύξηση του χρόνου επίλυσης.

Ακόμη, όσον αφορά την πυκνότητα, η αυξομείωση της επιφέρει, όχι πάντοτε ανάλογες, ή πάντοτε αντιστρόφως ανάλογες μεταβολές στις τιμές των χρόνων εκτέλεσης, εφόσον εξαρτάται και από ποιο σημείο γίνεται η αυξομείωση της.

Η Πυκνότητα είναι ο παράγοντας που καθορίζει το ποσοστό των ακμών σε ένα γράφημα, με την αύξησή της να επιφέρει ταυτόχρονη αύξηση και του πλήθους ακμών. Κάθε ακμή προσθέτει επιπλέον περιορισμούς στον ορισμό του προβλήματος. Ύπαρξη εξαιρετικά μικρού αριθμού ακμών (άρα και πολύ μικρής τιμής Πυκνότητας) συνεπάγεται με ύπαρξη ελάχιστων περιορισμών. Έτσι δεν έχουν τη δυνατότητα να περιορίσουν το πρόβλημα σε τέτοιο βαθμό, που να μην επιτρέπουν στον επιλυτή να εντοπίσει σύντομα κάποιο αποδεκτό μοντέλο λύσης. Δηλαδή, πολύ σπάνια εμφανίζονται αναθέσεις τιμών που επιφέρουν Μη-Ικανοποιησιμότητα, σε περιπτώσεις πολύ μικρού συνόλου περιορισμών. Το φαινόμενο αυτό ορίζεται ως «Περιοριστική Ανεπάρκεια» (Under - Constrainedness).

Από την άλλη, ύπαρξη εξαιρετικά μεγάλου αριθμού ακμών (άρα και πολύ μεγάλης τιμής Πυκνότητας) συνεπάγεται με ύπαρξη υπερβολικά πολλών περιορισμών, οι οποίοι έχουν τη δυνατότητα να περιορίσουν το πρόβλημα σε τέτοιο βαθμό που να επιτρέπουν στον επιλυτή να εντοπίσει εύκολα, και σχετικά σύντομα, τα μη- αποδεκτά μοντέλα λύσης. Δηλαδή, εμφανίζονται αρκετά σύντομα οι αναθέσεις που επιφέρουν Μη-Ικανοποιησιμότητα, λόγω της ύπαρξη εξαιρετικά μεγάλου συνόλου περιορισμών. Αυτό επιτρέπει την έγκαιρη αγνόηση τους και την ενασχόληση του επιλυτή με καινούργιες αναθέσεις τιμών, μέχρι την εύρεση της κατάλληλης. Το φαινόμενο αυτό ορίζεται ως «Περιοριστική Υπέρ-Επάρκεια» (Over - Constrainedness).

Στο ενδιάμεσο των δύο καταστάσεων, υπάρχει και το ενδεχόμενο ύπαρξης μόλις αρκετών περιορισμών, ώστε να μην διευκολύνεται ιδιαίτερα ο οποιοσδήποτε επιλυτής, κάτι που έχει αντίκτυπο στους χρόνους επίλυσης. Αυτό συμβαίνει καθώς περιπτώσεις στο χώρο αναζήτησης του επιλυτή, για τις οποίες δεν ικανοποιείται το πρόβλημα, ανακαλύπτονται σε πολύ μετέπειτα στάδιο, μετά από ήδη αρκετές κατάλληλες αναθέσεις τιμών στις υπόλοιπες μεταβλητές, μέχρι να βρεθεί η πρώτη μεταβλητή για την οποία δεν υπάρχει καμία κατάλληλη τιμή. Έτσι, αυτές οι περιπτώσεις αποκόπτονται από το δέντρο αναζήτησης, ενώ έχουν ήδη απασχολήσει αρκετά τον επιλυτή, ώστε να αυξήσουν σημαντικά το χρόνο επίλυσης.

Τα σημεία όπου παρατηρούνται τα φαινόμενα «Under - Constrainedness» και «Over - Constrainedness» διαφέρουν από εργαλείο σε εργαλείο. Γενικά, η συμπεριφορά της γραφικής παράστασης της πυκνότητας συνήθως παρουσιάζεται ως εξής: αρχικά μικρός χρόνος εκτέλεσης, έπειτα σταδιακή αύξηση και ακολουθεί σταδιακή μείωση.

Κεφάλαιο 8

Συμπεράσματα

8.1 Γενικά Συμπεράσματα και Παρατηρήσεις [1, 2, 3, 9, 10, 11]

Με τη μελέτη σημαντικών προβλημάτων Επιχειρηματολογίας, τη μοντελοποίηση τους σε διάφορα εργαλεία Προγραμματισμού Ικανοποίησης Περιορισμών και τη διεξαγωγή πειραματικής αξιολόγησης στα αποτελέσματά τους, σε συνδυασμό με όλες τις χρήσιμες γνώσεις που μας παρείχε η βιβλιογραφία μας, μπορούμε να καταλήξουμε στα εξής γενικά συμπεράσματα.

Τα προβλήματα εξεύρεσης Ευσταθούς Επέκτασης σε ένα γράφο/ πλαίσιο επιχειρηματολογίας, καθώς επίσης και ο χειρισμός των Μη-Ικανοποιήσιμων περιπτώσεων τους, είναι NP-δύσκολα, καθώς εμπίπτουν στην κατηγορία προβλημάτων Ικανοποιησιμότητας. Τόσο στην περίπτωση της Προτασιακής Ικανοποιησιμότητας (είτε με βάρη, είτε χωρίς), όσο και στην περίπτωση του Γραμμικού Προγραμματισμού (είτε Ακέραιου, είτε Μεικτού), γενικός σκοπός ήταν η εξεύρεση μίας κατάλληλης ανάθεσης τιμών, πρόβλημα το οποίο απαιτεί εκθετικό χρόνο επίλυσης. Αυτό σημαίνει ότι, αύξηση των μεταβλητών εισόδου (κυρίως του αριθμού των κόμβων), στη θεωρία, επιφέρει και εκθετική αύξηση του απαραίτητου χρόνου επίλυσης, μερικές φορές ακόμη και σε μη-πρακτικά επίπεδα (π.χ. περίπτωση Γραμμικού Προγραμματισμού, όπου υπήρξαν χρόνοι επίλυσης, εκτός του πλαισίου διακοπής διεργασιών/ process timeout threshold).

Σημαντικό σημείο αναφοράς είναι η βέλτιστη προσέγγιση στην εξεύρεση λύσης, στις περιπτώσεις χειρισμού των Μη-Ικανοποιήσιμών περιπτώσεων των προβλημάτων αυτών. Θέλουμε πάντα μία προσεγγιστική λύση που να προκύπτει από τον ελάχιστο δυνατό βαθμό συμβιβασμού. Αυτό συνεπάγεται με εξεύρεση ελάχιστου Συνόλου Πυρήνα για την Προτασιακή Ικανοποιησιμότητα (τόσο για την περίπτωση των επιχειρημάτων/κόμβων, όσο και για την περίπτωση περιορισμών/ακμών) και με υπολογισμό ελάχιστης δυνατής απόστασης από τα άκρα (ε) για το Γραμμικό Προγραμματισμό. Η βελτιστοποίηση (= ελαχιστοποίηση, στην περίπτωση μας) επιφέρει και κάποιες συνέπειες, κυρίως στο χρόνο εκτέλεσης, εφόσον μία «καλύτερη δυνατή λύση» χρειάζεται περισσότερο χρόνο για να εξευρεθεί, από τη στιγμή που πρέπει να τηρεί και επιπλέον κριτήρια, πέραν των γενικών περιορισμών (περισσότερες δοκιμές τιμών = περισσότερες πιθανές αποτυχίες και επανεκκινήσεις αναζήτησης). Έτσι, η πολυπλοκότητα εξαρτάται από την είσοδο δεδομένων (μεταβλητές και περιορισμοί).

Σημειώνεται ότι η Μη-Ικανοποιησιμότητα δεν επιφέρεται από την αύξηση των δεδομένων εισόδου, αλλά κυρίως από τους ξεχωριστούς, για κάθε πρόβλημα, συνδυασμούς περιορισμών. Η Μη-Ικανοποιησιμότητα επιφέρεται κυρίως από την ασταθή φύση κάποιου γραφήματος, δηλαδή τη συσχέτιση επιχειρημάτων του με τέτοιο τρόπο ώστε να μην υπάρχει δυνατότητα ομαδοποίησης τους, που να ικανοποιεί τον ορισμό της Ευσταθούς Επέκτασης.

Επιπλέον, τα προβλήματα μελετήθηκαν σε αφαιρετικό επίπεδο για διάφορους λόγους. Πρώτον, η μελέτη της αφαιρετικής πτυχής μας επέτρεψε να δημιουργήσουμε ένα γενικό μοντέλο, το οποίο επιλύει κάθε είδους παρόμοια προβλήματα. Επιπλέον, ο χώρος της Επιχειρηματολογίας, στην πραγματικότητα, χαρακτηρίζεται από εκφράσεις, τεκμηριώσεις, απόψεις, συναισθήματα κ.λπ., δηλαδή διέπεται από το ανθρώπινο, φυσικό στοιχείο (human/ natural element). Αποφεύγοντας να αναφερθούμε σε αυτά τα χαρακτηριστικά, και βλέποντας την Επιχειρηματολογία εντελώς αφαιρετικά, ως απλά ένα σύστημα/ γράφο από κόμβους και συσχετίσεις μεταξύ τους, χωρίς να προσδίδεται κάποια ιδιότητα στα ίδια τα επιχειρήματα, μπορούμε να εστιάσουμε το ενδιαφέρον μας στις μεταξύ τους συσχετίσεις και στα αποτελέσματα που προκύπτουν από τους επιλυτές. Έτσι, τη γενική συμπεριφορά των επιχειρημάτων μπορούμε να την αντιληφθούμε μέσω των συσχετίσεων και ομαδοποιήσεων τους, και όχι μέσω των εκφράσεων τους ή άλλων ιδιοτήτων.

8.2 Μελλοντικές Βελτιώσεις

Ακολουθούν κάποιες βελτιώσεις, εισηγήσεις και επεκτάσεις που θα μπορούσαν να υλοποιηθούν, με σκοπό, όχι μόνο τη βελτίωση στην επίλυση των προβλημάτων που μελετήσαμε, αλλά και στην γενική βελτίωση της εμπειρίας χρήσης του προγράμματος που αναπτύχθηκε για τους σκοπούς αυτής της εργασίας.

Μία βελτίωση που θα μπορούσε να υλοποιηθεί είναι η έννοια της «Τοπικότητας Γραφήματος». Με απλά λόγια, να δίνεται η δυνατότητα στο χρήστη να ζητήσει την μελέτη ενός προβλήματος σε τοπικό επίπεδο, επιλέγοντας μία «περιοχή» του γραφήματος, την οποία θα απαρτίζουν τα επιχειρήματα και οι συσχετίσεις που θα επιλέξει ο ίδιος. Προγραμματιστικά, αυτό ερμηνεύεται ως πρόβλημα Ευσταθούς Επέκτασης με ανάθεση βαρών, η οποία ανάθεση όμως θα γίνεται έμμεσα με τις επιλογές του χρήστη. Τα βάρη θα δίνουν περισσότερη έμφαση στην βέλτιστη επίλυση του προβλήματος για την περιοχή που επιλέχτηκε, με την υπόλοιπη περιοχή να λύνεται μη-βέλτιστα. Σκοπός αυτής της βελτίωσης είναι να παρέχει τη δυνατότητα εστίασης του ενδιαφέροντος σε μία συγκεκριμένη περιοχή του πλαισίου της Επιχειρηματολογίας. Ένα παράδειγμα από την πραγματικότητα, το οποίο αντιστοιχεί σε αυτή την προσέγγιση της «Τοπικότητας» στον ορισμό του προβλήματος, είναι η προεκλογική εκστρατεία, κατά την οποία, περισσότερο ενδιαφέρον δίνεται στο κομμάτι όπου αλληλοσυγκρούονται οι δύο επικρατέστεροι υποψήφιοι. Δηλαδή υφίσταται ένα πρόβλημα, από το οποίο θέλουμε να μελετήσουμε ένα μέρος του, μόνο, χωρίς να μας ενδιαφέρει ιδιαίτερα το υπόλοιπο μέρος. Αυτό δύναται να επιφέρει και παράλληλα μείωση στον χρόνο εκτέλεσης, καθώς θα μπορούσε το πρόγραμμα να επιλύσει βέλτιστα την «περιοχή» ενδιαφέροντος και να επιλύσει την υπόλοιπη περιοχή του γραφήματος χωρίς κάποια βελτιστοποίηση. Η μείωση της πολυπλοκότητας με αυτό τον τρόπο, έστω και σε μικρή κλίμακα, επιφέρει τεράστιες, θετικές αλλαγές στα εκθετικά προβλήματα NP-Hard.

Μία αρκετά ενδιαφέρουσα επέκταση, η οποία όμως απαιτεί την ενασχόληση περισσότερων ατόμων με διάφορες γνώσεις στον τομέα της Πληροφορικής, είναι η παροχή δυνατότητας ανάθεσης πιο ειδικών χαρακτηριστικών στο πρόγραμμα, ώστε να ξεφύγει από το αφαιρετικό πλαίσιο. Η επέκταση αυτή θα περιλαμβάνει τα εξής:

δυνατότητα ανάθεσης βαθμού «συμπάθειας» και «αντιπάθειας» μεταξύ επιχειρημάτων, δυνατότητα ανάθεσης γνωμών, τεκμηριωμένων απόψεων και έκφρασης συναισθημάτων σε κάθε επιχείρημα. Μέσω μίας βάσης δεδομένων θα φυλάγονται τα δεδομένα αυτά και θα ανακτούνται από το πρόγραμμα, για να δημιουργείται η θεωρία των προβλημάτων, βάσει αυτών. Την έννοια της «ανθρώπινης, φυσικής συμπεριφοράς» που θα έχουν τα επιχειρήματα, θα τη διαχειρίζεται ένα γνωστικό σύστημα (cognitive system), που μέσω της συνεργασίας του με το χρήστη θα μπορεί να προσαρμοστεί στα ζητούμενα του. Έτσι ο χρήστης θα μπορεί εύκολα να μεταφράζει οποιοιδήποτε μη-αφαιρετικό στοιχείο της Επιχειρηματολογίας (εκφράσεις, απόψεις, συναισθήματα κ.λπ.), από φυσική γλώσσα σε μεταβλητές, παραμέτρους και περιορισμούς που θα πρέπει να ικανοποιηθούν. Τα στοιχεία αυτά θα μπορούν να εισάγονται μέσω συμπλήρωσης πεδίων μίας φόρμας (form filling), είτε ακόμη και με ρητή εισαγωγή (explicit input) στη φυσική γλώσσα. Σκοπός της επέκτασης αυτής είναι να αξιοποιηθεί η δυνατότητα του προγράμματος, όχι μόνο ως επιστημονικό, αφαιρετικό εργαλείο, αλλά και ως ένας πλήρες οδηγός χρήστη, ένας προσομοιωτής (simulator) πραγματικών καταστάσεων, που θα επιτρέπει και σε άτομα με περιορισμένες γνώσεις πληροφορικής να ερμηνεύσουν τα αποτελέσματα που θα επιφέρει η κατάσταση που τους ενδιαφέρει.

Τέλος, για περαιτέρω ενίσχυση της διαδραστικότητας μεταξύ προγράμματος/ χρήστη (human/program interaction) και τη βελτίωση της εμπειρίας χρήσης, θα μπορούσε το πρόγραμμα να αναπτυχθεί σε γραφικό περιβάλλον. Όλες οι είσοδοι (επιλογή προβλημάτων και βελτιστοποιήσεων, εισαγωγές δεδομένων κ.λπ.) θα μπορούσαν να γίνονται μέσω συμπλήρωσης πεδίων φόρμας. Όλες οι έξοδοι (εμφάνιση και ερμηνεία αποτελεσμάτων) θα μπορούσαν να παρουσιάζονται με γραφικό τρόπο π.χ. σχεδιασμός γραφήματος και χρωματισμός των επιχειρημάτων και συσχετίσεων που ανήκουν στο σύνολο Ευσταθούς Επέκτασης ή στον Πυρήνα που θα εξαχθεί (ανάλογα με το πρόβλημα που επιλέχθηκε από το χρήστη).

Γενικά, υπάρχει μεγάλη δυνατότητα βελτίωσης της εν λόγω εργασίας, με διάφορες ιδέες από διαφορετικά πεδία.

Βιβλιογραφία

[1] Δημόπουλος Γ., Διάλεξη «Γραμμικός Προγραμματισμός», Μάθημα ΕΠΛ 433 – Προγραμματισμός και Ικανοποίηση Περιορισμών, Πανεπιστήμιο Κύπρου

[2] Δημόπουλος Γ., Διάλεξη «Προβλήματα Βελτιστοποίησης», Μάθημα ΕΠΛ 433 – Προγραμματισμός και Ικανοποίηση Περιορισμών, Πανεπιστήμιο Κύπρου

[3] Δημόπουλος Γ., Διάλεξη «An introduction to Propositional Satisfiability (SAT) and Answer Set Programming (ASP)», Μάθημα ΕΠΛ 433 - Προγραμματισμός και Ικανοποίηση Περιορισμών, Πανεπιστήμιο Κύπρου

[4] Κουμπαράκης Μ. (2018), Διάλεξη «Προβλήματα Ικανοποίησης Περιορισμών», http://cgi.di.uoa.gr/~ys02/dialekseis2018/csp1spp.pdf

[5] Σπινέλλης Δ. (n.d.), Γράφοι. Τμήμα Διοικητικής Επιστήμης και Τεχνολογίας- Οικονομικό Πανεπιστήμιο Αθηνών.
http://dmst.aueb.gr/dds/ads/graph/indexw.htm

[6] Φιλίππου Α. (2010), Διάλεξη «Γράφοι», Μάθημα «ΕΠΛ 231: Δομές Δεδομένων και Αλγόριθμοι», Πανεπιστήμιο Κύπρου
http://www.cs.ucy.ac.cy/courses/EPL231/2010-winter/notes/notes17-18.pdf

[7] Alder G., (n.d.), draw.io's Official Website
https://about.draw.io/

[8] Alviano M. (n.d.), Maxino
https://alviano.com/software/maxino/

[9] Atkinson et al.(2017), Towards Artificial Argumentation.
http://www.cs.uu.nl/groups/IS/archive/henry/aimagDef15march17.pdf, AI Magazine, 38(3), 1.-16.

[10] Baroni P., Caminada M. Giacomin M. (2004), An introduction to argumentation semantics.
https://users.cs.cf.ac.uk/CaminadaM/publications/KER-BaroniCaminadaGiacomin

[11] Bench-Capon, Dunne(2007), Argumentation in artificial intelligence.
https://www.sciencedirect.com/science/article/pii/S0004370207000793

[12] Biere A. (n.d.), Lingeling, Plingeling and Treengeling Entering the SAT Competition 2013
http://fmv.jku.at/papers/Biere-SAT-Competition-2013-Lingeling.pdf

[13] Eclipse Foundation, (n.d.), Eclipse Neon Official Website
https://www.eclipse.org/neon/

[14] Liffiton M.H., Sakallah K.A.(n.d.). Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints.
https://sun.iwu.edu/~mliffito/publications/jar_liffiton_CAMUS.pdf

[15] MathWorks, (n.d.), intlinprog, Mixed-integer linear programming (MILP)
https://www.mathworks.com/help/optim/ug/intlinprog.html

[16] MathWorks, (n.d.), Matlab's Official Website
https://www.mathworks.com/products/matlab.html

[17] Max-SAT (2016), Eleventh Max-SAT Evaluation
http://maxsat.ia.udl.cat/requirements/

[18] Microsoft (n.d.), Microsoft Excel's Official Support Website
https://support.office.com/en-in/excel

[19] Rodriguez-Carbonell (2015). Combinatorial Problem Solving (CPS). Tutorial on SAT Solvers.
https://www.cs.upc.edu/~erodri/webpage/cps/lab/sat/tutorial-sat-solvers-slides/slides.pdf

[20] SAT Competition (2009), Benchmark Submission Guidelines
http://www.satcompetition.org/2009/format-benchmarks2009.html

[21] Silva J.M. (2012). MaxSAT and Related Optimization Problems.
https://es-static.fbk.eu/events/satsmtschool12/slides/1x04_SS12.pdf

[22] SLP Scheduler (n.d.), Hard constraints vs. soft constraints
https://help.slpscheduler.com/knowledge-bases/2/articles/231-hard-constraints-vs-soft-constraints

[23] TutorialsPoint (n.d.), MATLAB - M-Files
https://www.tutorialspoint.com/matlab/matlab_m_files.htm

Παράρτημα Α

Στο Παράρτημα Α παρουσιάζονται διάφορα, κυρίως τεχνικά, συμπληρωματικά στοιχεία των προβλημάτων που μελετήθηκαν. Έμφαση δίνεται στην πλήρη επεξήγηση της μορφής των αρχείων εισόδου για κάθε πρόβλημα, στην παρουσίαση οθόνων εκτέλεσης και αποτελεσμάτων των επιλυτών, καθώς και στην πλήρη κατανόηση των ιδιαιτεροτήτων του κάθε προβλήματος, όσον αφορά τα εργαλεία και τους επιλυτές που χρησιμοποιούνται.

Σκοπός είναι να διασαφηνιστούν, περαιτέρω, τα συμφραζόμενα των Κεφαλαίων 3, 4, 5 και 6, μέσω πραγματικών παραδειγμάτων, συνδέοντας έτσι τη θεωρητική προσέγγιση με την πρακτική εφαρμογή.

Α.1 Κεφάλαιο 3 – Προτασιακή Λογική

Στα πλαίσια του Κεφαλαίου 3 μελετήθηκε το πρόβλημα της Ευσταθούς Επέκτασης στην Προτασιακή Ικανοποιησιμότητα (Propositional SAT). Έγινε αναφορά στη δημιουργία αρχείων CNF, τα οποία περιείχαν τους Προτασιακούς περιορισμούς σε μορφή Συζευκτικής Κανονικής Μορφής, στη χρήση τους ως είσοδο στους επιλυτές και την ανάκτηση και ερμηνεία αποτελεσμάτων. Ο επιλυτής που χρησιμοποιήθηκε ήταν ο SAT επιλυτής, Lingeling.

Α.1.1 Μορφή Αρχείου Εισόδου [3, 20]

Ακολουθεί παράδειγμα που επεξηγεί πλήρως την πρότυπη μορφή που ακολουθούν τα αρχεία εισόδου των επιλυτών Προτασιακής Ικανοποιησιμότητας (SAT Solvers). Τα πρότυπα καθορίστηκαν για σκοπούς συνέπειας εισόδου και αποτελεσμάτων σε όλους τους επιλυτές, ως προϋπόθεση για συμμετοχή στους διαγωνισμούς SAT.

Παράδειγμα πρότυπου αρχείου CNF, με την ορθή μορφή που αποδέχεται ο επιλυτής:

```
1) c LingelingInput.cnf
2) p cnf 4 7
3) -4 -1 0
   -3 -2 0
   -4 -2 0
   1 0
   2 0
   3 2 0
   4 1 2 0
```

Σχήμα A.1.1.1 Αρχείο CNF

ΕΠΕΞΗΓΗΣΗ:

1) Γραμμές σχολίων: Αρχίζουν με το χαρακτήρα ‘c’.

2) Γραμμές που περιγράφουν τις ιδιότητες του αρχείου: Αρχίζουν με το χαρακτήρα ‘p’. Ιδιότητες με τη σειρά που εμφανίζονται:

p ΕίδοςΑρχείου #Μεταβλητών #ΌρωνΣυζεύξεων

Επομένως, το εν λόγω αρχείο είναι τύπου «cnf», περιέχει 4 διαφορετικές μεταβλητές (εξαιρουμένου του 0) και 7 διαφορετικές προτάσεις/μέλη/περιορισμούς που απαρτίζουν τις συζεύξεις της Συζευκτικής Κανονικής Μορφής που περιγράφεται.

3) Το αρχείο περιγράφει την ακόλουθη θεωρία Προτασιακού Λογισμού:

$$(\neg 4 \vee \neg 1) \wedge (\neg 3 \vee \neg 2) \wedge (\neg 4 \vee \neg 2) \wedge 1 \wedge 2 \wedge (3 \vee 2) \wedge (4 \vee 1 \vee 2)$$

A.1.2 Κλήση Επιλυτή [12, 19]

Ο Lingeling καλείται μέσω του τερματικού (terminal) Linux/UNIX χρησιμοποιώντας τις ακόλουθες εντολές:

```
./lingeling LingelingResults/LingelingInput.cnf
```

Σχήμα A.1.2.1 Κλήση Lingeling

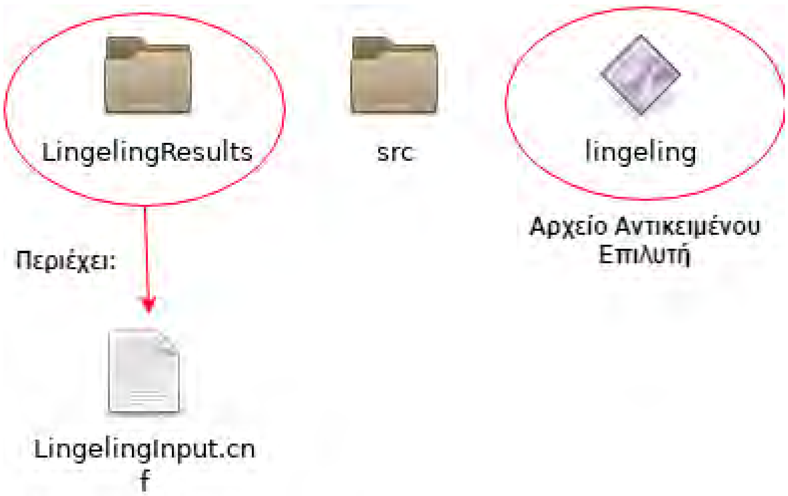
Όπου:

- ./lingeling: εντολή η οποία εκτελεί το αρχείο αντικειμένου (lingeling.o) του επιλυτή

- LingelingResults/LingelingInput.cnf: μονοπάτι αρχείου εισόδου (παράμετρος). Στη συγκεκριμένη περίπτωση δίνουμε ως είσοδο το CNF αρχείο «LingelingInput.cnf», το οποίο βρίσκεται στον φάκελο «LingelingResults», ο οποίος βρίσκεται στο ίδιο μονοπάτι με τον επιλυτή.

ΣΗΜΕΙΩΣΗ: Η παράμετρος αυτή μπορεί να αντικατασταθεί με κάποιο πλήρες μονοπάτι, έτσι μπορούμε να εισάγουμε κάποιο αρχείο σαν είσοδο, από οπουδήποτε και αν βρίσκεται στον ηλεκτρονικό υπολογιστή.

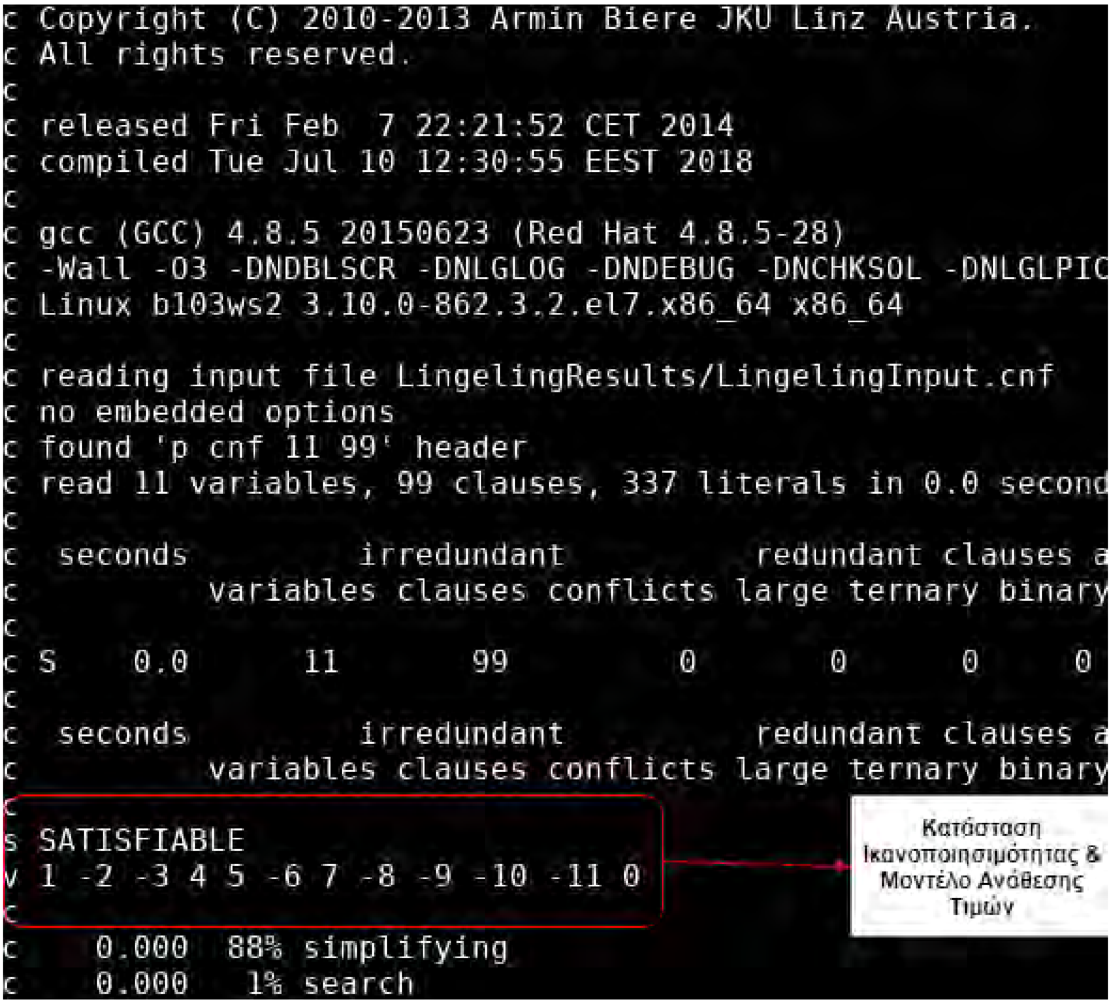
Με την πάρα πάνω εντολή εκτελείται το αρχείο αντικειμένου «lingeling.o», διαβάζεται το αρχείο εισόδου «LingelingInput.cnf», το οποίο βρίσκεται εντός του φακέλου «LingelingResults», από τον επιλυτή, και επιστρέφεται η λύση του προβλήματος που περιγράφεται. Ο φάκελος «LingelingResults» πρέπει να βρίσκεται εντός του ίδιου μονοπατιού με το εκτελέσιμο αρχείο του επιλυτή. Σε διαφορετική περίπτωση θα πρέπει να δίνεται ως παράμετρος εισόδου του επιλυτή, το ολοκληρωμένο όνομα του μονοπατιού που περιέχει το αρχείο εισόδου που επιθυμούμε.



Σχήμα Α.1.2.2 Μονοπάτι επιλυτή lingeling/ CNF αρχείου εισόδου

A.1.3 Ερμηνεία Αποτελεσμάτων Επιλυτή [12, 19]

Παράδειγμα εξόδου επιλυτή Lingeling, για τυχαίο πρόβλημα Ευσταθούς Επέκτασης:



```
c Copyright (C) 2010-2013 Armin Biere JKU Linz Austria.
c All rights reserved.
c
c released Fri Feb  7 22:21:52 CET 2014
c compiled Tue Jul 10 12:30:55 EEST 2018
c
c gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-28)
c -Wall -O3 -DNDBLSCR -DNLGLOG -DNDEBUG -DNCHKSOL -DNLGLPIC
c Linux b103ws2 3.10.0-862.3.2.el7.x86_64 x86_64
c
c reading input file LingelingResults/LingelingInput.cnf
c no embedded options
c found 'p cnf 11 99' header
c read 11 variables, 99 clauses, 337 literals in 0.0 second
c
c  seconds         irredundant          redundant clauses a
c         variables clauses conflicts large ternary binary
c
c S    0.0      11       99        0       0       0     0
c
c  seconds         irredundant          redundant clauses a
c         variables clauses conflicts large ternary binary
c
s SATISFIABLE
v 1 -2 -3 4 5 -6 7 -8 -9 -10 -11 0
c
c    0.000  88% simplifying
c    0.000   1% search
```


Σχήμα A.1.3.1 Μορφή Αποτελεσμάτων Lingeling

ΕΡΜΗΝΕΙΑ ΑΠΟΤΕΛΕΣΜΑΤΩΝ:

Κατάσταση Προβλήματος: Ικανοποιήσιμο (SATISFIABLE)

Μοντέλο = {1, -2, -3, 4, 5, -6, 7, -8, -9, -10, -11}

= {1=T, 2=F, 3=F, 4=T, 5=T, 6=F, 7=T, 8=F, 9=F, 10=F, 11=F},

όπου:

F = False → μπροστά από τη μεταβλητή υπάρχει ο χαρακτήρας '-'

T = True → μπροστά από τη μεταβλητή δεν υπάρχει ο χαρακτήρας '-'

➔ Σύνολο Ευσταθούς Επέκτασης = {1, 4, 5, 7}

A.2 Κεφάλαιο 4 – Μη-Ικανοποιησιμότητα στην Προτασιακή Λογική

Στα πλαίσια του Κεφαλαίου 4 μελετήθηκε η Μη-Ικανοποιησιμότητα στο πρόβλημα της Ευσταθούς Επέκτασης, μέσω της Μέγιστης Προτασιακής Ικανοποιησιμότητας (Propositional MAXSAT), με την εξαγωγή στοιχείων από τον ορισμό του προβλήματος (εξαγωγή επιχειρημάτων ή περιορισμών). Έγινε αναφορά στη δημιουργία αρχείων WCNF, τα οποία περιείχαν τους Προτασιακούς περιορισμούς με τα βάρη τους, σε μορφή Συζευκτικής Κανονικής Μορφής, στη χρήση τους ως είσοδο στους επιλυτές και την ανάκτηση και ερμηνεία αποτελεσμάτων. Ο επιλυτής που χρησιμοποιήθηκε ήταν ο MAXSAT επιλυτής, Maxino.

A.2.1 Μορφή Αρχείου Εισόδου [3, 17, 20, 21, 22]

Ακολουθεί παράδειγμα που επεξηγεί πλήρως την πρότυπη μορφή που ακολουθούν τα αρχεία εισόδου των επιλυτών Μέγιστης Προτασιακής Ικανοποιησιμότητας με βάρη (MAXSAT Solvers). Τα πρότυπα καθορίστηκαν για σκοπούς συνέπειας εισόδου και αποτελεσμάτων σε όλους τους επιλυτές, ως προϋπόθεση για συμμετοχή στους διαγωνισμούς MAXSAT. Παράδειγμα πρότυπου αρχείου WCNF, με την ορθή μορφή και εισαγωγή βαρών, που αποδέχεται ο επιλυτής:

```
c UNSATCoreOfNodes.cnf
p wcnf 6 12 1000
1 4 0
1000 -1 4 0
1000 -1 -2 0
1000 -1 -3 0
1000 1 -4 3 0
1 5 0
1000 -2 5 0
1000 -2 -3 0
1000 2 -5 1 0
1 6 0
1000 -3 6 0
1000 3 -6 2 0
```

ΕΠΕΞΗΓΗΣΗ:

1) Γραμμές σχολίων: Αρχίζουν με το χαρακτήρα 'c'.

2) Γραμμές που περιγράφουν τις ιδιότητες του αρχείου: Αρχίζουν με το χαρακτήρα ‘p’. Ιδιότητες με τη σειρά που εμφανίζονται:

$$p\ ΕίδοςΑρχείου\ \#Μεταβλητών\ \#ΌρωνΣυζεύξεων\ \#ΜέγιστοΔυνατόΒάρος$$

Επομένως, το πάρα πάνω αρχείο είναι τύπου wcnf (Weighted CNF), περιέχει 6 μεταβλητές, 12 προτάσεις/ περιορισμούς και το μέγιστο δυνατό βάρος, που η ανάθεση του κάνει κάποιον περιορισμό «Ισχυρό», είναι η τιμή 1000.

3) Το αρχείο περιγράφει την ακόλουθη θεωρία Προτασιακού Λογισμού:

Ισχυροί Περιορισμοί (βάρος = 1000):

$$(\neg 1 \vee 4) \wedge (\neg 1 \vee \neg 2) \wedge (\neg 1 \vee \neg 3) \wedge (1 \vee \neg 4 \vee 3) \wedge (\neg 2 \vee 5) \wedge (\neg 2 \vee \neg 3) \wedge$$
$$(2 \vee \neg 5 \vee 1) \wedge (\neg 3 \vee 6) \wedge (3 \vee \neg 6 \vee 2)$$
$$\wedge$$

Χαλαροί Περιορισμοί (βάρος = 1):

$$4 \wedge 5 \wedge 6$$

4) Όσοι εκ των περιορισμών έχουν βάρος ίσο με 1000 είναι ισχυροί, και έχουν προτεραιότητα στο να ικανοποιηθούν πρώτοι. Όσοι εκ των περιορισμών έχουν βάρος ίσο με 1 είναι χαλαροί, και σε περίπτωση που ο επιλυτής θα χρειαστεί να αγνοήσει κάποιους από τους περιορισμούς ώστε το πρόβλημα να γίνει Ικανοποιήσιμο, θα επιλέξει τους λιγότερους δυνατούς εξ’ αυτών πρώτα.

A.2.2 Κλήση Επιλυτή [8]

Ο Maxino καλείται μέσω του τερματικού (terminal) Linux/UNIX χρησιμοποιώντας τις ακόλουθες εντολές:

```
./maxino-2015-k16-static CoreOfClausesWeighted.cnf
```

Σχήμα A.2.2.1 Κλήση Επιλυτή maxino

Όπου:

- ./maxino-2015-k16-static: εντολή η οποία εκτελεί το αρχείο αντικειμένου (maxino-2015-k16-static.o) του επιλυτή

- CoreOfClausesWeighted.cnf: μονοπάτι αρχείου εισόδου (παράμετρος). Στη συγκεκριμένη περίπτωση δίνουμε ως είσοδο το CNF αρχείο «CoreOfClausesWeighted.cnf», το οποίο βρίσκεται στο ίδιο μονοπάτι με τον επιλυτή.

ΣΗΜΕΙΩΣΗ: Η παράμετρος αυτή μπορεί να αντικατασταθεί με κάποιο πλήρες μονοπάτι, έτσι μπορούμε να εισάγουμε κάποιο αρχείο σαν είσοδο, από οπουδήποτε και αν βρίσκεται στον ηλεκτρονικό υπολογιστή.

Με την πάρα πάνω εντολή εκτελείται το αρχείο αντικειμένου «maxino-2015-k16-static.o», διαβάζεται το αρχείο εισόδου «CoreOfClausesWeighted.cnf», το οποίο βρίσκεται στο ίδιο μονοπάτι με τον επιλυτή και επιστρέφεται η λύση του προβλήματος που περιγράφεται.

Σχήμα A.2.2.2 Επιλυτής maxino και αρχείο εισόδου

A.2.3 Ερμηνεία Αποτελεσμάτων Επιλυτή [2, 8, 14, 17, 21, 22]

Ακολουθεί παρουσίαση και ερμηνεία αποτελεσμάτων που επιστρέφονται για τα δύο είδη προβλημάτων που μελετήθηκαν για τη Μη-Ικανοποιησιμότητα στην Προτασιακή Λογική (εξαγωγή πυρήνα περιορισμών/ επιχειρημάτων).

A.2.3.1 Περίπτωση Εξαγωγής Πυρήνα Περιορισμών/Ακμών [2, 8, 14, 17, 21, 22]

Το πρόγραμμα που αναπτύχθηκε για τους σκοπούς της εργασίας αυτής, συλλέγει τις πληροφορίες από το αρχείο που παράγει ο επιλυτής Maxino και ελέγχει για να βρει τον

Πυρήνα Περιορισμών από το αρχείο εισόδου (με τον τρόπο που περιγράφεται στο Κεφάλαιο 4, υποκεφάλαιο 4.5.3). Επιστρέφει τη λύση για το πρόβλημα Εξαγωγής Πυρήνα Περιορισμών, με την ακόλουθη μορφή:

```
******************************************
        RESULTS
******************************************
1) UNSAT CORE OF CLAUSES: [1 -29 -13 0]
```

Σχήμα Α.2.3.1.1 Αποτελέσματα για Πυρήνα Περιορισμών

ΕΡΜΗΝΕΙΑ ΑΠΟΤΕΛΕΣΜΑΤΩΝ:

Πυρήνας Περιορισμών/Ακμών = {[1 -29 -13 0]} = {συσχέτιση(29, 13)}

Η συσχέτιση μεταξύ των επιχειρημάτων 29 και 13 οδηγούσε το πρόβλημα στη Μη-Ικανοποιησιμότητα, και απαρτίζει τον ελάχιστο Πυρήνα Περιορισμών. Η αφαίρεση της από τον ορισμό του προβλήματος θα επιφέρει την Ικανοποιησιμότητα του.

Α.2.3.2 Περίπτωση Εξαγωγής Πυρήνα Επιχειρημάτων [2, 8, 14, 17, 21, 22]

Το πρόγραμμα που αναπτύχθηκε για τους σκοπούς της εργασίας αυτής επιστρέφει τη λύση για το πρόβλημα Εξαγωγής Πυρήνα Επιχειρημάτων, με την ακόλουθη μορφή:

```
******************************************
        RESULTS
******************************************
1) STABLE EXTENSION: [3]
2) IN PROBLEM & OUT OF STABLE EXTENSION: [1, 2]
3) UNSAT CORE of NODES/ARGUMENTS: [4]
```

Σχήμα Α.2.3.2.1 Αποτελέσματα για Πυρήνα Επιχειρημάτων

ΕΡΜΗΝΕΙΑ ΑΠΟΤΕΛΕΣΜΑΤΩΝ:

Σύνολο Ευσταθούς Επέκτασης = {3}

Σύνολο Επιχειρημάτων εκτός Ευσταθούς Επέκτασης/ εντός προβλήματος = {1, 2}

Πυρήνας Επιχειρημάτων/Κόμβων = {4}

Το επιχείρημα 4 συμμετείχε σε συσχετίσεις με τέτοιο τρόπο που οδηγούσε το πρόβλημα στη Μη-Ικανοποιησιμότητα, και επομένως απαρτίζει τον ελάχιστο Πυρήνα Επιχειρημάτων. Η αφαίρεση του από τον ορισμό του προβλήματος θα επιφέρει την Ικανοποιησιμότητα του.

Α.3 Κεφάλαιο 5 – Ακέραιος Γραμμικός Προγραμματισμός

Στα πλαίσια του Κεφαλαίου 5 μελετήθηκε το πρόβλημα της Ευσταθούς Επέκτασης, μέσω του Ακέραιου Γραμμικού Προγραμματισμού (Integer Linear Programming). Έγινε αναφορά στη δημιουργία αρχείων script Matlab (.m files), τα οποία περιείχαν τους γραμμικούς περιορισμούς ως ανισότητες/ εξισώσεις. Επίσης έγινε αναφορά στη χρήση τους ως είσοδο στην κονσόλα της Matlab και την ανάκτηση και ερμηνεία αποτελεσμάτων από τους πίνακες μεταβλητών. Ο επιλυτής που χρησιμοποιήθηκε ήταν ο LP επιλυτής, intlinprog, ο οποίος παρέχεται μέσω της Matlab.

Α.3.1 Ιδιαιτερότητες, Απαιτήσεις και Δυνατότητες Matlab [15, 16, 23]

Ακολουθούν μερικές στοιχειώδεις ιδιαιτερότητες, απαιτήσεις αλλά και δυνατότητες του εργαλείου Matlab, τις οποίες θα πρέπει να γνωρίζουμε πριν προβούμε στη δημιουργία του αρχείου εισόδου:

1) Δήλωση *κενού πίνακα* γίνεται με χρήση «[ ]».

Π.χ. $Aeq = [\,];$

2) Ο επιλυτής intlinprog χρησιμοποιεί πίνακες για να λύσει προβλήματα της μορφής:

$$\boldsymbol{A} * \boldsymbol{x} \leq \boldsymbol{b}$$

Επομένως, *μετατρέπουμε* όλους τους περιορισμούς ανισοτήτων σε αυτή τη μορφή με τους εξής τρόπους:

i. Φέρνουμε όλες τις μεταβλητές στο αριστερά μέρος της ανισότητας.

Π.χ. $x_1 \geq x_2 - 4 \rightarrow x_1 - x_2 \geq -4$

ii. Εάν έχουμε «≥», πολλαπλασιάζουμε και τα δύο μέρη (αριστερά και δεξιά) με -1, ώστε να μετατραπεί ο τελεστής σύγκρισης της ανισότητας σε «≤».

Π.χ. $x_1 - x_2 \geq -4 \rightarrow -x_1 + x_2 \leq 4$

3) Ερωτηματικό (';') εντός ενός πίνακα δηλώνει την αλλαγή ανισότητας/εξίσωσης, (ανάλογα με τον πίνακα), επομένως σημάνει το τέλος της δήλωσης ενός περιορισμούς.

4) Κόμμα (',') εντός ενός πίνακα υποδηλώνει την αλλαγή μεταβλητής εντός της ίδιας ανισότητας/εξίσωσης, επομένως υποδηλώνει ότι συνεχίζεται η δήλωση του ίδιου περιορισμού.

5) Γραμμές Σχολίων (προαιρετική χρήση): ξεκινούν με το χαρακτήρα «%»

6) Διάνυσμα μηδενικών: χρήση εντολής zeros(n,1), η οποία δημιουργεί ένα διάνυσμα μεγέθους n θέσεων, οι οποίες περιέχουν όλες μέσα την τιμή 0. Στην περίπτωση της μοντελοποίησης του προβλήματος της Ευσταθούς Επέκτασης, η εν λόγω εντολή χρησιμοποιείται για τη δήλωση του κατώτατου ορίου τιμών των μεταβλητών.

7) Διάνυσμα μονάδων: χρήση εντολής ones(n,1), η οποία δημιουργεί ένα διάνυσμα μεγέθους n θέσεων, οι οποίες περιέχουν όλες μέσα την τιμή 1. Στην περίπτωση της μοντελοποίησης του προβλήματος της Ευσταθούς Επέκτασης, η εν λόγω εντολή χρησιμοποιείται για τη δήλωση του ανώτατου ορίου τιμών των μεταβλητών.

A.3.2 Αναπαράσταση Αρχείου Εισόδου Matlab [15, 16, 23]

Για σκοπούς περαιτέρω κατανόησης, ακολουθεί παράδειγμα της μορφής ενός πρότυπου αρχείου εισόδου για τη Matlab (Matlab script file: .m file), για τον Ακέραιο Γραμμικό Προγραμματισμό:

```
% Declare function
f = [1; 1; 1; ];

% Declare integer variables
intcon = 1:3;

% Declare the left parts of the inequalities
A = [1,1,0;
1,0,1;
0,1,1;
-1,0,-1;
-1,-1,0;
0,-1,-1;
];

% Declare the right parts of the inequalities
b = [1;1;1;-1;-1;-1;];

% Declare the left and right parts of the equalities
Aeq = [];

beq = [];

% Declare lower & upper bounds
lb = zeros(3,1);
ub = ones(3,1);

[x,fval] =intlinprog(f, intcon, A, b, Aeq, beq, lb, ub);
```

Σχήμα A.3.2.1 Αρχείο σε μορφή Matlab script file (επέκταση .m)

Ερμηνεία:

- Αντικειμενική Συνάρτηση: f = x1 + x2 + x3
- Στόχος Βελτιστοποίησης: Ελαχιστοποίηση (ο intlinprog είναι προκαθορισμένος να βρίσκει λύσεις βάσει της ελαχιστοποίησης της αντικειμενικής συνάρτησης)
- Ακέραιες μεταβλητές: intcon = $\{x_1, x_2, x_3\}$

- Περιορισμοί Ανισοτήτων (Μορφή: $A * x \leq b$):

$$\{x_1 + x_2 \leq 1,$$
$$x_1 + x_3 \leq 1,$$
$$x_2 + x_3 \leq 1,$$
$$-x_1 - x_3 \leq -1,$$
$$-x_1 - x_2 \leq -1,$$
$$-x_2 - x_3 \leq -1 \}$$

- Περιορισμοί Εξισώσεων (Μορφή: $Aeq * x = beq$):

  Δεν υφίστανται στο συγκεκριμένο αρχείο (κενοί οι πίνακες Aeq και beq)

- Άνω/ Κάτω Όρια Τιμών (ub/lb):

$$0 \leq x_1 \leq 1$$
$$0 \leq x_2 \leq 1$$
$$0 \leq x_3 \leq 1$$

- Κλήση Επιλυτή: Ο επιλυτής καλείται να λύσει το πρόβλημα, σύμφωνα με τα πάρα πάνω στοιχεία. Ως αποτέλεσμα, στην μεταβλητή x επιστρέφεται το μοντέλο λύσης, ενώ στη μεταβλητή fval επιστρέφεται η βελτιστοποιημένη τιμή της αντικειμενικής συνάρτησης f.

A.3.3 Επίλυση Προβλήματος και Ερμηνεία Αποτελεσμάτων

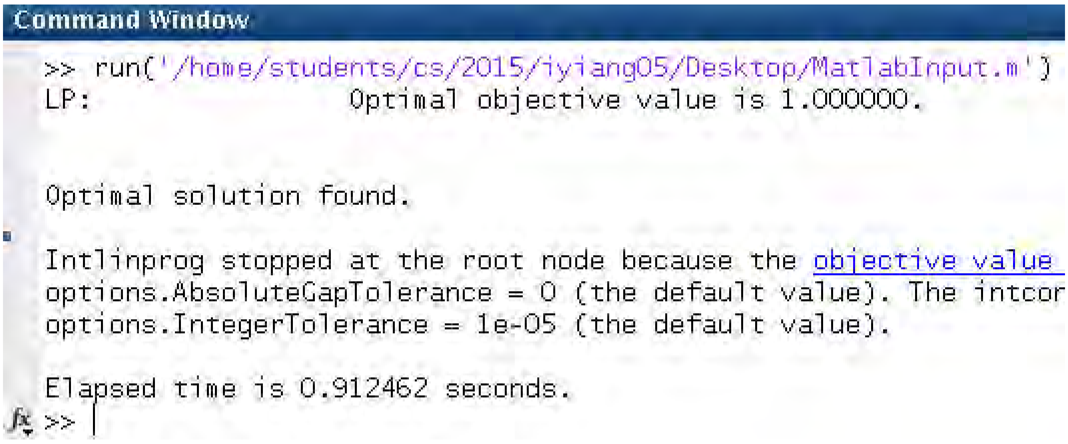
Μετά τη συγκέντρωση όλων των κατάλληλων περιορισμών για το πρόβλημα που μελετούμε, επόμενα βήματα αποτελούν η επίλυση του προβλήματος που περιγράφει το αρχείο εισόδου («.m» αρχείο) και η ανάκτηση και ερμηνεία των αποτελεσμάτων που επιστρέφει ο επιλυτής. Για τις ακόλουθες οθόνες, ως παράδειγμα χρησιμοποιείται η επίλυση και ερμηνεία των αποτελεσμάτων του προβλήματος που παρουσιάζεται στο αρχείο του υποκεφαλαίου A.3.2 του Παραρτήματος αυτού.

A.3.3.1 Διαδικασία Επίλυσης Προβλήματος [15, 16, 23]

Ανοίγουμε το λογισμικό MATLAB (Προτεινόμενες Εκδόσεις: R2018b, R2018a).

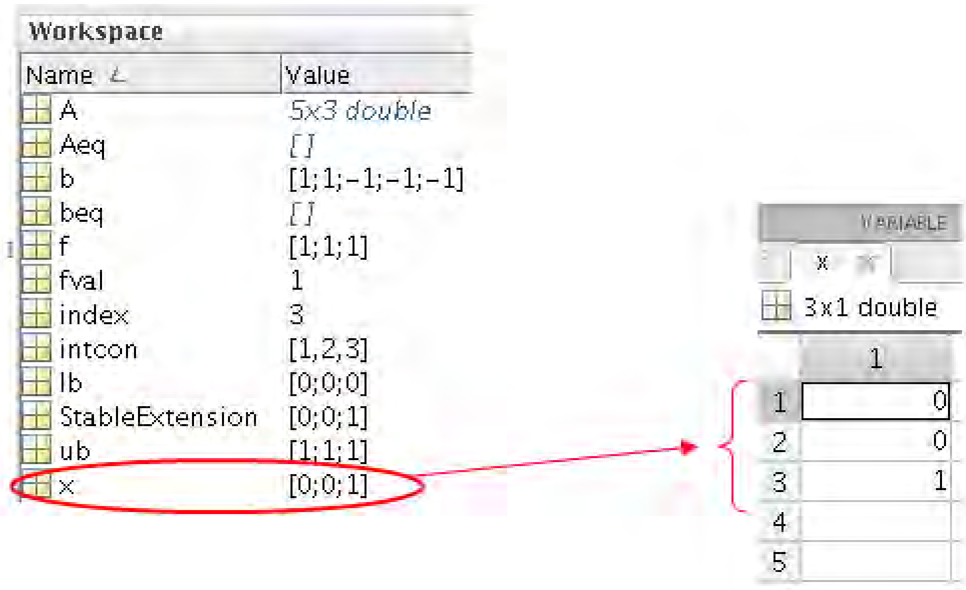
Σχήμα Α.3.3.1.1 Πρόγραμμα MATLAB/ έκδοση R2018b

1) «Σύρουμε» το .m αρχείο (Matlab Script File) στο Παράθυρο Εντολών (Command Window). Η MATLAB θα καλέσει αυτόματα τη συνάρτηση run('…'), η οποία θα εκτελέσει τον κώδικα που περιέχεται εντός του αρχείου που δώσαμε ως είσοδο και θα επιστρέψει στοιχεία που περιγράφουν τη διαδικασία εκτέλεσης, στην κονσόλα.



Σχήμα Α.3.3.1.2 Μορφή Αποτελεσμάτων Επίλυσης από κονσόλα MATLAB

2) Τα αποτελέσματα συγκεντρώνονται και παρουσιάζονται μέσω μεταβλητών, στο Χώρο Εργασίας (Workspace). Πατώντας διπλό-κλικ στη μεταβλητή που θέλουμε να επιθεωρήσουμε, θα ανοίξει ένα καινούργιο παράθυρο που παρουσιάζει το αποτέλεσμα της εν λόγω μεταβλητής, σε μορφή πίνακα.



Σχήμα Α.3.3.1.3 Αποτελέσματα σε μεταβλητές

Α.3.3.2 Αποτελέσματα [1, 15, 16, 23]

Η μεταβλητή-πίνακας **x**, η οποία απεικονίζεται πάρα πάνω, αποτελεί το τελικό αποτέλεσμα της διαδικασίας αυτής. Αποτελεί, δηλαδή, το μοντέλο ανάθεσης τιμών που επιστρέφει ο intlinprog.

Ακολουθούν σημαντικά χαρακτηριστικά για ορθή ερμηνεία των αποτελεσμάτων της:

i. Είναι Μονοδιάστατος Πίνακας ως Μοντέλο Λύσης (έχει μία και μοναδική στήλη)
ii. Αριθμός Γραμμής = Αριθμός επιχειρήματος
iii. Τιμές:
   - 0 = το επιχείρημα **δεν ανήκει** στο σύνολο της Ευσταθούς Επέκτασης
   - 1 = το επιχείρημα **ανήκει** στο σύνολο της Ευσταθούς Επέκτασης
   - Κενό = δεν υφίσταται το επιχείρημα με δείκτη τον αριθμό της τρέχουσας γραμμής, στον ορισμό του προβλήματος

Βάσει των πάρα πάνω οδηγιών, ερμηνεύουμε τα αποτελέσματα της μεταβλητής ***x*** ως εξής (για το πρόβλημα της Ευσταθούς Επέκτασης που μελετούμε πάρα πάνω):

Σύνολο Ευσταθούς Επέκτασης = Σύνολο Γραμμών της x που περιέχουν την τιμή 1={3}

➔ Το επιχείρημα 3 αποτελεί το σύνολο της Ευσταθούς Επέκτασης

A.4 Κεφάλαιο 6 – Μεικτός-Ακέραιος Γραμμικός Προγραμματισμός

Στα πλαίσια του Κεφαλαίου 6 μελετήθηκε το πρόβλημα της Ευσταθούς Επέκτασης, μέσω του Μεικτού-Ακέραιου Γραμμικού Προγραμματισμού (Mixed-Integer Linear Programming), καθώς επίσης και ο χειρισμός των Μη-Ικανοποιήσιμων περιπτώσεων του. Έγινε αναφορά στη δημιουργία αρχείων script Matlab (.m files), τα οποία περιείχαν τους γραμμικούς περιορισμούς ως ανισότητες/ εξισώσεις. Επίσης έγινε αναφορά στη χρήση τους ως είσοδο στην κονσόλα της Matlab και την ανάκτηση και ερμηνεία αποτελεσμάτων από τους πίνακες μεταβλητών. Ο επιλυτής που χρησιμοποιήθηκε ήταν ο MILP επιλυτής, intlinprog, ο οποίος παρέχεται μέσω της Matlab.

A.4.1 Ιδιαιτερότητες, Απαιτήσεις και Δυνατότητες Matlab [15, 16, 23]

Ισχύει ότι προαναφέρθηκε στο υποκεφάλαιο A.3.1

A.4.2 Αναπαράσταση Αρχείου Εισόδου Matlab [15, 16, 23]

Για σκοπούς περαιτέρω κατανόησης, ακολουθεί παράδειγμα της μορφής ενός πρότυπου αρχείου εισόδου για τη Matlab (Matlab script file: .m file), για τον Μεικτό Ακέραιο Γραμμικό Προγραμματισμό:

```
% Declare objective function
f = [0; 0; 0; 0; 0; 0; 0; 0; 1;];
% Declare integer variables
intcon = 5:8;
% Declare the left and right parts of the inequalities
A = [-1,0,-1,0,-1,0,0,0, 0;
0,-1,0,-1,0,-1,0,0, 0;
1,0,-1,0,0,0,-1,0, 0;
0,1,0,-1,0,0,0,-1, 0;
0,0,1,0,0,0,0,0, -1;
0,0,0,1,0,0,0,0, -1;
0,0,-1,0,0,0,0,0, 0;
0,0,0,-1,0,0,0,0, 0;
0,0,0,0,1,0,0,0, 0;
0,0,0,0,0,1,0,0, 0;
0,0,0,0,-1,0,0,0, 0;
0,0,0,0,0,-1,0,0, 0;
0,0,0,0,0,0,1,0, 0;
0,0,0,0,0,0,0,1, 0;
0,0,0,0,0,0,-1,0, 0;
0,0,0,0,0,0,0,-1, 0;
0,0,0,0,1,0,1,0, 0;
0,0,0,0,0,1,0,1, 0;];
b = [-1;-1;0;0;0;0;0;0;1;1;0;0;1;1;0;0;1;1;];
% Declare the left and right parts of the equalities
Aeq = [1,0,0,0,0,0,0,0, 0;
0,1,0,0,0,0,0,0, 0;];
beq = [1;1;];
lb = zeros(9,1); % Declare lower bound
ub = ones(9,1); % Declare upper bound
[x,fval] =intlinprog(f, intcon, A, b, Aeq, beq, lb, ub);
```

Σχήμα Α.4.2.1 Αρχείο MATLAB script (επέκταση .m)

Ερμηνεία:

- Αντικειμενική Συνάρτηση:

$$f = 0 * a_1 + 0 * a_2 + 0 * k_1 + 0 * k_2 + 0 * \lambda_1^1 + 0 * \lambda_2^1 + 0 * \lambda_1^2 + 0 * \lambda_2^2 + 1 * \varepsilon = \varepsilon$$

- Στόχος Βελτιστοποίησης: Ελαχιστοποίηση (ο intlinprog είναι προκαθορισμένος να βρίσκει λύσεις βάσει της ελαχιστοποίησης της αντικειμενικής συνάρτησης)

- Ακέραιες μεταβλητές:

$$\text{intcon} = \{\lambda_1^1, \quad \lambda_2^1, \quad \lambda_1^2, \quad \lambda_2^2\}$$

- Περιορισμοί Ανισοτήτων (Μορφή: $A * x \le b$):

$$
\begin{aligned}
\{- a_1 - k_1 - \lambda_1^1 &\le -1, \\
-a_2 - k_2 - \lambda_2^1 &\le -1, \\
a_1 - k_1 - \lambda_1^2 &\le 0, \\
a_2 - k_2 - \lambda_2^2 &\le 0, \\
k_1 - \varepsilon &\le 0, \\
k_2 - \varepsilon &\le 0, \\
-k_1 &\le 0, \\
-k_2 &\le 0, \\
\lambda_1^1 &\le 1, \\
\lambda_2^1 &\le 1, \\
-\lambda_1^1 &\le 0, \\
-\lambda_2^1 &\le 0, \\
\lambda_1^2 &\le 1, \\
\lambda_2^2 &\le 1, \\
-\lambda_1^2 &\le 0, \\
-\lambda_2^2 &\le 0, \\
\lambda_1^1 + \lambda_1^2 &\le 1, \\
\lambda_2^1 + \lambda_2^2 &\le 1\}
\end{aligned}
$$

- Περιορισμοί Εξισώσεων (Μορφή: $Aeq * x = beq$):

$$
\begin{aligned}
\{a_1 &= 1, \\
a_2 &= 1\}
\end{aligned}
$$

- Άνω/ Κάτω Όρια Τιμών:

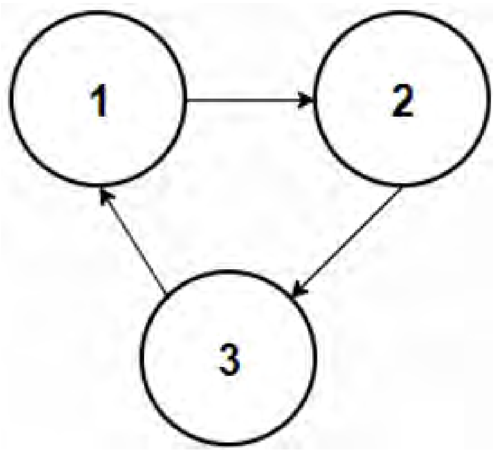
$$0 \le a_1, a_2, k_1, k_2, \lambda_1^1, \lambda_2^1, \lambda_1^2, \lambda_2^2, \varepsilon \le 1$$

- Κλήση Επιλυτή: Ο επιλυτής καλείται να λύσει το πρόβλημα, σύμφωνα με τα πάρα πάνω στοιχεία. Ως αποτέλεσμα, στην μεταβλητή x επιστρέφεται το μοντέλο λύσης, ενώ στη μεταβλητή fval επιστρέφεται η βελτιστοποιημένη τιμή της αντικειμενικής συνάρτησης f.

A.4.3 Επίλυση Προβλήματος και Ερμηνεία Αποτελεσμάτων [15, 16, 23]

Μετά τη συγκέντρωση όλων των κατάλληλων περιορισμών, επόμενα βήματα αποτελούν η επίλυση του προβλήματος και η ανάκτηση και ερμηνεία των αποτελεσμάτων του επιλυτή.

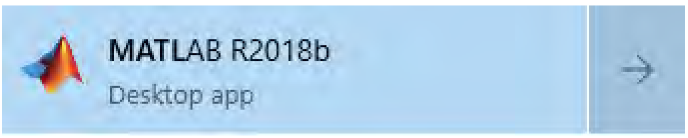
Τα πάρα κάτω αποτελέσματα δεν αναφέρονται στο παράδειγμα αρχείου που δόθηκε προηγουμένως (υποκεφάλαιο A.4.2), καθώς το πρόβλημα που περιέγραφε ήταν Ικανοποιήσιμο. Σε αυτό το υποκεφάλαιο μας ενδιαφέρει να δούμε για κάποιο Μη-Ικανοποιήσιμο πρόβλημα, πως γίνεται ο χειρισμός και η ανάθεση πραγματικών τιμών στις μεταβλητές και ποια μορφή έχει η έξοδος των αποτελεσμάτων. Δεν ασχολούμαστε με τη διαδικασία επίλυσης κάποιου συγκεκριμένου προβλήματος, μόνο παρατηρούμε, γενικά, τον τρόπο λειτουργίας της MATLAB σε τέτοια προβλήματα. Οι διαδικασίες που ακολουθούν (A.4.3.1-A.4.3.2) αφορούν το εξής Μη-Ικανοποιήσιμο πρόβλημα:



Σχήμα A.4.3.1 Μη-Ικανοποιήσιμο Πρόβλημα

A.4.3.1 Διαδικασία Επίλυσης Προβλήματος [15, 16, 23]

1) Ανοίγουμε το λογισμικό MATLAB (Προτεινόμενες Εκδόσεις: R2018b, R2018a).



Σχήμα A.4.3.1.1 Πρόγραμμα MATLAB/ έκδοση R2018b

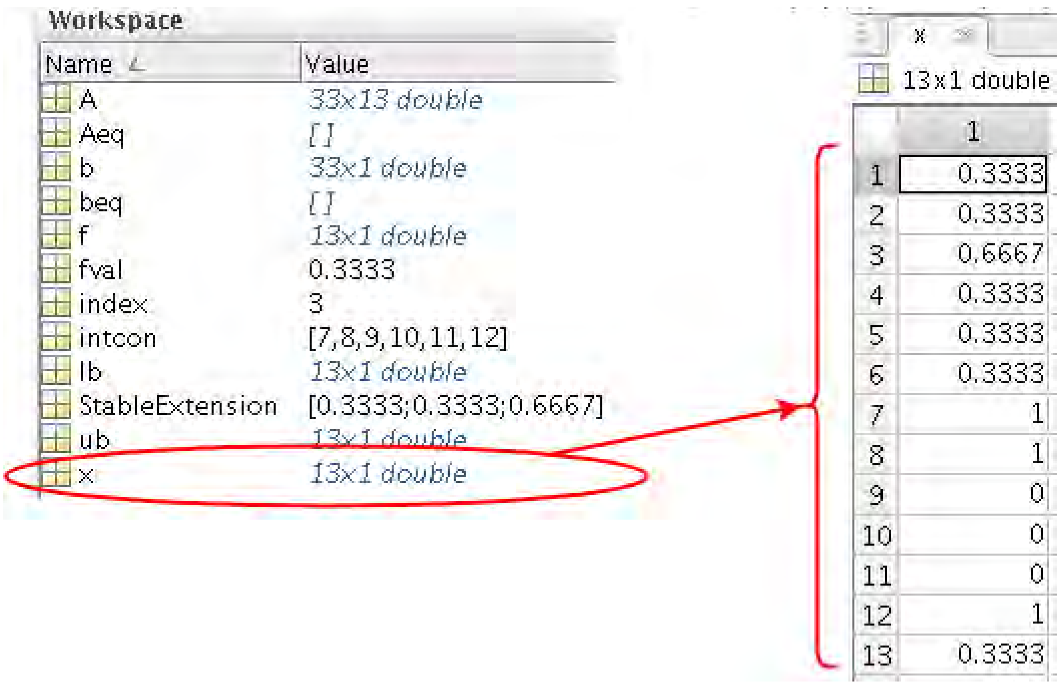
2) «Σύρουμε» το .m αρχείο (Matlab Script File) στο Παράθυρο Εντολών (Command Window). Η MATLAB θα καλέσει αυτόματα τη συνάρτηση run(‘…’), η οποία θα εκτελέσει τον κώδικα που περιέχεται εντός του αρχείου που δώσαμε ως είσοδο και θα επιστρέψει στοιχεία που περιγράφουν τη διαδικασία εκτέλεσης, στην κονσόλα.

Command Window

```
>> run('/home/students/cs/2015/iyiang05/Desktop/MatlabInput.m')
LP:                 Optimal objective value is 0.000000.

Cut Generation:     Applied 3 mir cuts, and 5 Gomory cuts.
                    Lower bound is 0.282051.

Heuristics:         Found 1 solution using rounding.
                    Upper bound is 0.333333.
                    Relative gap is 3.85%.

Cut Generation:     Applied 2 Gomory cuts.
                    Lower bound is 0.288889.
                    Relative gap is 3.33%.
```

Σχήμα Α.4.3.1.2 Αποτελέσματα Επίλυσης από κονσόλα MATLAB

3) Τα αποτελέσματα συγκεντρώνονται και παρουσιάζονται μέσω μεταβλητών, στο Χώρο Εργασίας (Workspace). Πατώντας διπλό-κλικ στη μεταβλητή που θέλουμε να επιθεωρήσουμε, θα ανοίξει ένα καινούργιο παράθυρο που παρουσιάζει το αποτέλεσμα της εν λόγω μεταβλητής, σε μορφή πίνακα.



Σχήμα Α.4.3.1.3 Αποτελέσματα στις μεταβλητές

A.4.3.2 Αποτελέσματα [1, 15, 16, 23]

Η μεταβλητή-πίνακας $\mathbf{x}$, η οποία απεικονίζεται πάρα πάνω, αποτελεί το τελικό αποτέλεσμα της διαδικασίας αυτής. Αποτελεί, δηλαδή, το μοντέλο ανάθεσης τιμών που επιστρέφει ο intlinprog. Ακολουθούν σημαντικά χαρακτηριστικά για την ορθή ερμηνεία των αποτελεσμάτων της, βάσει του πάρα πάνω παραδείγματος:

i. Είναι Μονοδιάστατος Πίνακας ως Μοντέλο Λύσης (έχει μία και μοναδική στήλη)

ii. Οι γραμμές παρουσιάζουν τα εξής στοιχεία, για πλήθος επιχειρημάτων Ν (με τη σειρά αυτή):
   - Ν γραμμές που αναφέρονται στις τιμές των επιχειρημάτων $a_1 \dots a_N$
   - Ν γραμμές που αναφέρονται στις αποστάσεις επιχειρημάτων, $k_1 \dots k_N$
   - Ν γραμμές που αναφέρονται στη μεταβλητή λ^1 των επιχειρημάτων, $\lambda_1^1 \dots \lambda_N^1$
   - Ν γραμμές που αναφέρονται στη μεταβλητή λ^2 των επιχειρημάτων, $\lambda_1^2 \dots \lambda_N^2$
   - 1 γραμμή που αναφέρεται στην μέγιστη επιτρεπτή απόσταση ε

iii. Τιμές για επιχειρήματα (δηλαδή τιμές των πρώτων Ν γραμμών, όπου Ν το πλήθος των επιχειρημάτων του προβλήματος):
   - Εάν τιμή=0: το επιχείρημα **δεν ανήκει** στο σύνολο της Ευσταθούς Επέκτασης
   - Εάν τιμή=1: το επιχείρημα **ανήκει** στο σύνολο της Ευσταθούς Επέκτασης
   - Κενό = δεν υφίσταται το επιχείρημα με δείκτη τον αριθμό της τρέχουσας γραμμής, στον ορισμό του προβλήματος
   - Εάν τιμή πλησιάζει το 1: το επιχείρημα είναι **πιο πιθανόν να ανήκει** στο σύνολο της Ευσταθούς Επέκτασης
   - Εάν τιμή πλησιάζει το 0: το επιχείρημα είναι **πιο πιθανόν να μην ανήκει** στο σύνολο της Ευσταθούς Επέκτασης

Βάσει των πάρα πάνω οδηγιών και του παραδείγματος που μελετούμε, ερμηνεύουμε τα αποτελέσματα της μεταβλητής x ως εξής:

- Τιμή Επιχειρήματος $α_1$ = 0.3333 → πιο πιθανόν δεν ανήκει στην Ευσταθή Επέκταση
- Τιμή Επιχειρήματος $α_2$ = 0.3333 → πιο πιθανόν δεν ανήκει στην Ευσταθή Επέκταση
- Τιμή Επιχειρήματος $α_3$ = 0.6667 → πιο πιθανόν ανήκει στην Ευσταθή Επέκταση
- Τιμή απόστασης $κ_1$ = 0.3333 → Το επιχείρημα $α_1$ έχει απόσταση 0.3333 από το άκρο 0
- Τιμή απόστασης $κ_2$ = 0.3333 → Το επιχείρημα $α_2$ έχει απόσταση 0.3333 από το άκρο 0
- Τιμή απόστασης $κ_3$ = 0.3333 → Το επιχείρημα $α_3$ έχει απόσταση 0.3333 από το άκρο 1
- Τιμή απόστασης $λ^1{}_1$ = 1 → Το επιχείρημα $α_1$ ωθείται προς το άκρο 0
- Τιμή απόστασης $λ^1{}_2$ = 1 → Το επιχείρημα $α_2$ ωθείται προς το άκρο 0
- Τιμή απόστασης $λ^1{}_3$ = 0 → Το επιχείρημα $α_3$ δεν ωθείται προς το άκρο 0
- Τιμή απόστασης $λ^2{}_1$ = 0 → Το επιχείρημα $α_1$ δεν ωθείται προς το άκρο 1
- Τιμή απόστασης $λ^2{}_2$ = 0 → Το επιχείρημα $α_2$ δεν ωθείται προς το άκρο 1
- Τιμή απόστασης $λ^2{}_3$ = 1 → Το επιχείρημα $α_3$ ωθείται προς το άκρο 1
- Τιμή μέγιστης δυνατής απόστασης ε = 0.3333 → Κανένα επιχείρημα δεν έχει απόσταση από το άκρο του, μεγαλύτερη του 0.3333

Παράρτημα Β

Β.1 Εισαγωγή

Για σκοπούς πειραματικής αξιολόγησης, αναγκαία ήταν η ανάπτυξη ενός προγράμματος που να υποστηρίζει τον ορισμό, τη μοντελοποίηση και την επίλυση των προβλημάτων που μελετήθηκαν σε αυτή τη διπλωματική εργασία, με απλό και σαφή τρόπο, ώστε να είναι εύχρηστο και κατανοητό. Επίσης, η ανάπτυξη του εν λόγω προγράμματος δεν βοήθησε μόνο στη διεκπεραίωση της πειραματικής αξιολόγησης, αλλά πρόσφερε και την ευκαιρία να δούμε τον τρόπο με τον οποίο, όλα τα συμφραζόμενα της θεωρητικής πτυχής της εργασίας, εφαρμόζονται στην πράξη.

Β.2 Γενικά Στοιχεία Τεχνικού Χαρακτήρα [8, 12, 13, 15, 16, 17, 20, 23]

Το εν λόγω πρόγραμμα αναπτύχθηκε στη γλώσσα προγραμματισμού JAVA, μέσω της πλατφόρμας ανάπτυξης προγραμμάτων Eclipse (έκδοση Neon). Συστήνεται η μεταγλώττιση και εκτέλεση του, αποκλειστικά σε περιβάλλον λειτουργικού συστήματος Linux/Unix, καθώς μέρος της υλοποίησης περιέχει κλήση αρχείων τύπου αντικειμένου (.o – object files), των οποίων η εκτέλεση απαιτεί τη χρήση της γραμμής εντολών (command line/ terminal) των εν λόγω λειτουργικών συστημάτων.

Β.3 Απαραίτητα Επιπλέον Εργαλεία [8, 12, 13, 15, 16, 17, 20, 23]

Απαραίτητη είναι η εγκατάσταση των επιλυτών Lingeling και Maxino (2 αρχεία αντικειμένου, «.o»), και η τοποθέτησή τους στο ίδιο μονοπάτι με το πρόγραμμα, καθώς γίνεται χρήση και των δύο, ως μέρος της υλοποίησης.

Σχήμα Β.3.1 Απαραίτητοι επιλυτές

B.4 Γενικά Στοιχεία Υλοποίησης [8, 12, 13, 15, 16, 17, 20, 23]

Το πρόγραμμα είναι πλήρως τεκμηριωμένο καθώς περιέχει τόσο σχολιασμό γραμμών κώδικα, όσο και σχολιασμό κλάσεων, μεθόδων και δομών δεδομένων, όπου αυτό θεωρήθηκε απαραίτητο.

```
/**
 * Class which contains the main method of the program
 *
 * @author iyiang05
 *
 */
public class Main {

        // declare scanners for all user's choices
        private static Scanner choice, choice2, choice3, choice4;

        /**
         * Static method "main", responsible for the execution of the program
         *
         * @param args
         *           arguments passed from command line
         * @throws IOException
         *            Exception of Input/Output nature
         */
        public static void main(String args[]) throws IOException {
                try {
                        // print welcome messages
                        System.out.print("***************************\n");
                        System.out.print("STABLE EXTENSION MODELLING\n");
                        System.out.print("***************************\n");
```

Σχήμα B.3.2 Τεκμηρίωση Γραμμών/Κλάσεων/Μεθόδων Κώδικα μέσω Σχολίων

Επιπλέον, ακολουθεί το πρότυπο του Δομημένου Προγραμματισμού, καθώς περιέχει χρήση πολλαπλών, μικρών μεθόδων και τη διάσπαση του πηγαίου κώδικα σε πολλαπλά αρχεία κλάσεων, βάση ιδιοτήτων, δηλαδή:

Main: κύρια κλάση προγράμματος, ArgumentationGraph: κλάση που δημιουργεί γράφους, Checker: κλάση που διεξάγει επικυρώσεις και ελέγχους, LogicHandler: κλάση χειρισμού διαδικασιών Προτασιακής Λογικής, MatlabHandler: κλάση χειρισμού διαδικασιών Γραμμικού Προγραμματισμού.

Έτσι διασφαλίζεται η αφαιρετικότητα και η επαναχρησιμοποίηση του προγράμματος, όπως επίσης και η καλή αναγνωσιμότητα του πηγαίου κώδικα.

Σχήμα Β.3.3 Αρχεία που περιέχουν τις κλάσεις του προγράμματος σε JAVA

Β.5 Επαφή με το Χρήστη

Β.5.1 Περιγραφή Μορφής Περιβάλλοντος Χρήστη

Το πρόγραμμα είναι βασισμένο στην επικοινωνία μέσω κειμένου (text-based use), καθώς πλησιάζει περισσότερο τα επιστημονικά πρότυπα προγραμμάτων. Επομένως, σημασία δίνεται στην ευχρηστία και τη μεθοδικότητα του προγράμματος, και στα ίδια τα αποτελέσματα, σε αντίθεση με τα πιο εμπορικής μορφής λογισμικά, τα οποία εστιάζουν στον τρόπο εμφάνισης των δεδομένων και στο γραφικό περιβάλλον.

Β.5.2 Μεταγλώττιση και Εκτέλεση Προγράμματος

Η μεταγλώττιση του προγράμματος γίνεται μέσω της εντολής:

$$javac\ *.java$$

➔ Μεταγλωττίζει όλα τα αρχεία JAVA (.java files) του τρέχοντος μονοπατιού.

Η εκτέλεση του προγράμματος γίνεται μέσω της εντολής:

$$java\ Main$$

➔ Εκτελεί την κύρια μέθοδο (main) του προγράμματος, η οποία βρίσκεται εντός της κλάσης Main.

B.5.3 Οθόνη Καλωσορίσματος/ Επιλογή Προβλήματος

Με την εκτέλεση του προγράμματος παρουσιάζεται η αρχική οθόνη καλωσορίσματος, μέσω της οποίας ο χρήστης τίθεται να επιλέξει ένα εκ των 5 διαθέσιμων ειδών προβλημάτων, προς επίλυση. Αυτό γίνεται με την καταχώρηση ενός αριθμού μεταξύ 1 και 5 (συμπεριλαμβανομένων) και πατώντας έπειτα Enter, ώστε να εκτελεστεί η αντίστοιχη εντολή που παρουσιάζεται στη λίστα επιλογής προβλήματος.

```
***************************
STABLE EXTENSION MODELLING
***************************
Select Problem:
--------------------
1) Stable Extension (Propositional Logic)
2) Stable Extension (Integer Linear Programming)
3) Stable Extension (NON-Integer Linear Programming)
4) Extraction of UNSAT Core of Clauses for Stable Exten
5) Extraction of UNSAT Core of Nodes/Arguments for Stab

Submit your choice (# of problem):
```

Σχήμα B.5.3.1: Οθόνη «Καλωσορίσματος»

B.5.4 Οθόνες Επιλογής Μεθόδου και Καταχώρησης Εισόδου

Στη συνέχεια ακολουθεί η επιλογή μεθόδου εισόδου και η καταχώρηση τιμών στις μεταβλητές εισόδου. Οι δύο επιλογές μεθόδου είναι οι εξής:

1) Δημιουργία Τυχαίου Γραφήματος → απαιτεί εισαγωγή αριθμού Κόμβων και Πυκνότητας

```
Select Input Method:
--------------------
1) Random Graph
2) Graph File

Submit your choice (# of input method): 1

Give Graph Elements:
--------------------
1) Give # of nodes (integer): 100
2) Give graph density (real): 0.2
```

Σχήμα B.5.4.1: Δημιουργία Τυχαίου

2) Άνοιγμα Αρχείου Γράφου → απαιτεί εισαγωγή μονοπατιού, έγκυρου Αρχείου Γράφου

```
Select Input Method:
--------------------
1) Random Graph
2) Graph File

Submit your choice (# of input method): 2

Give Graph Elements:
--------------------
1) Give a valid graph file: graph1.txt
```

Σχήμα Β.5.4.2: Άνοιγμα Αρχείου Γράφου

Β.5.5 Οθόνες Στοιχείων Γραφήματος

Με την καταχώρηση της εισόδου ακολουθεί η εμφάνιση πληροφοριών σχετικά με το γράφο που αναπαριστά το πρόβλημα, όπως ο αριθμός κόμβων και ακμών, ο τρόπος υπολογισμού των ακμών, καθώς και ο πίνακας γειτνίασης (adjacency matrix) του.

```
        GRAPH INFO
*******************************************
a) Nodes in graph:      5
b) Edges in graph:      8
c) Edges calculated based on the following formula:

        Edges (directed graph) = floor {d*n*(n-1)}

where:
 d: density
 n: number of nodes

d) Table of links in graph:

01100
00001
10010
00001
10010
```

Σχήμα Β.5.5.1 Πληροφορίες Γράφου Προβλήματος

B.5.6 Οθόνες Αποτελεσμάτων

Κατά τη διάρκεια της επίλυσης ενός προβλήματος, το πρόγραμμα ενημερώνει το χρήστη μέσω κατάλληλων μηνυμάτων, για τη διαδικασία. Με την ολοκλήρωση της επίλυσης, τυπώνονται στην οθόνη τα κατάλληλα αποτελέσματα, ανάλογα με το πρόβλημα που επέλεξε ο χρήστης στην αρχή. Πιο συγκεκριμένα, για τα 5 προβλήματα έχουμε τις εξής μορφές αποτελεσμάτων:

1) **Εύρεση Συνόλου Ευσταθούς Επέκτασης Επιχειρημάτων, ως πρόβλημα Προτασιακής Ικανοποιησιμότητας:** Επιστρέφεται η κατάσταση του προβλήματος (SATISFIABLE ή UNSATISFIABLE, αναλόγως), μαζί με το σύνολο Ευσταθούς Επέκτασης και την πληθικότητά του. Επίσης επιστρέφεται ο συνολικός χρόνος επίλυσης.

   Π.χ.

```
1) PROBLEM STATUS:       SATISFIABLE
2) RESULTS:
------------------------------------
       Stable Extension: 2 nodes
------------------------------------
[2, 3]

3) LINGELING EXECUTION TIME:    58ms
```

Σχήμα Β.5.6.1 Αποτελέσματα για Προτασιακή Ικανοποιησιμότητα

2) **Εύρεση Συνόλου Ευσταθούς Επέκτασης Επιχειρημάτων, ως πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού:** Επιστρέφεται η κατάσταση του προβλήματος (SATISFIABLE ή UNSATISFIABLE, αναλόγως), και οδηγίες για το πώς μπορεί να αξιοποιηθεί το αρχείο εισόδου Matlab (.m file), το οποίο δημιουργήθηκε εντός του ίδιου μονοπατιού.

   Π.χ.

```
4) APPLY THE RESULTS:

        a) MatlabInput.m file has been created su
        b) Load MatlabInput.m file on MATLAB's co
        c) Check the variables for the Stable Ext
```

Σχήμα Β.5.6.2 Αποτελέσματα για Γραμμικό Προγραμματισμό

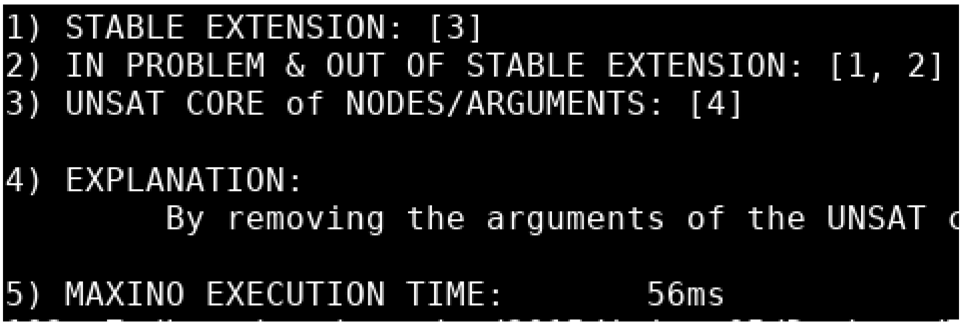
3) **Εύρεση Συνόλου Ευσταθούς Επέκτασης Επιχειρημάτων, ως πρόβλημα Μεικτού-Ακέραιου Γραμμικού Προγραμματισμού:** Πριν να επιστραφούν τα αποτελέσματα, ζητείται από το χρήστη να υποβάλει την επιθυμία του, είτε για ελαχιστοποίηση της μέγιστης δυνατής απόστασης από τα άκρα (ε), είτε για ανάθεση μίας σταθερής τιμής σε αυτή, την οποία υποβάλλει ο ίδιος. Επιστρέφονται παρόμοια αποτελέσματα με την περίπτωση 2 (Εύρεση Συνόλου Ευσταθούς Επέκτασης Επιχειρημάτων, ως πρόβλημα Ακέραιου Γραμμικού Προγραμματισμού).

4) **Εύρεση Πυρήνα Προτάσεων (Ακμών) σε προβλήματα Ευσταθούς Επέκτασης, ως πρόβλημα Προτασιακής Ικανοποιησιμότητας με βάρη:** Επιστρέφεται ο Πυρήνας Προτάσεων που πρέπει να αφαιρεθούν από τη θεωρία του προβλήματος, μία επεξήγηση για τη χρησιμότητα των αποτελεσμάτων, όπως επίσης και οι χρόνοι εκτέλεσης, τόσο του κάθε επιλυτή (Lingeling και Maxino) ξεχωριστά, όσο και ο συνολικός χρόνος εκτέλεσης.
Π.χ.

```
1) UNSAT CORE OF CLAUSES: [1 -3 -1 0]

2) EXPLANATION:
        By removing the clauses of the UNSA

3) LINGELING EXECUTION TIME:    59ms
4) MAXINO EXECUTION TIME:       10ms
5) TOTAL EXECUTION TIME:        69ms
```

Σχήμα B.5.6.3 Αποτελέσματα για Πυρήνα Περιορισμών

5) **Εύρεση Πυρήνα Επιχειρημάτων (Κόμβων), σε προβλήματα Ευσταθούς Επέκτασης, ως προβλήματα Προτασιακής Ικανοποιησιμότητας με βάρη:** Επιστρέφεται το σύνολο Ευσταθούς Επέκτασης επιχειρημάτων, το σύνολο επιχειρημάτων που ανήκουν στον ορισμό του προβλήματος αλλά όχι στην Ευσταθή Επέκταση, και το σύνολο του Πυρήνα Επιχειρημάτων που πρέπει να αφαιρεθούν για την επίτευξη της Ικανοποιησιμότητας του προβλήματος. Επίσης επιστρέφεται μία εξήγηση για το πώς μπορούν να αξιοποιηθούν τα αποτελέσματα, και ο χρόνος που χρειάστηκε ο Maxino για να επιλύσει το πρόβλημα. Π.χ.



Σχήμα Β.5.6.4 Αποτελέσματα για Πυρήνα Επιχειρημάτων

Β.6 Εξαγωγή Αποτελεσμάτων

Για περαιτέρω διευκόλυνση στην πειραματική διαδικασία, όλα τα αποτελέσματα και τα απαραίτητα αρχεία αποθηκεύονται σε ένα φάκελο, εντός του ίδιου μονοπατιού που περιέχεται και το πρόγραμμα, όπως παρουσιάζεται στο ακόλουθο σχήμα.

Σχήμα Β.6.1 Εξαγωγή Αποτελεσμάτων σε φάκελο

Β.7 Κώδικας

Ακολουθεί ο πηγαίος κώδικας του προγράμματος, διαιρεμένος σε κλάσεις, στα ανάλογα αρχεία τύπου JAVA (.java). Συνοδεύεται με πλήρη τεκμηρίωση, μέσω σχολιασμού γραμμών, κλάσεων, δομών δεδομένων και μεθόδων (στα αγγλικά).

B.7.1 Κλάση «Main» (κύρια κλάση του προγράμματος)

```java
import java.io.*;
import java.util.*;

/**
 * Class which contains the main method of the program
 *
 * @author iyiang05
 *
 */
public class Main {

	// declare scanners for all user's choices
	private static Scanner choice, choice2, choice3, choice4;

	/**
	 * Static method "main", responsible for the execution of the program
	 *
	 * @param args
	 *            arguments passed from command line
	 * @throws IOException
	 *             Exception of Input/Output nature
	 */
	public static void main(String args[]) throws IOException {
		try {
			// print welcome messages
			System.out.print("***************************\n");
			System.out.print("STABLE EXTENSION MODELLING\n");
			System.out.print("***************************\n");

			// give options, to select from
			System.out.println("Select Problem:\n--------------------");
			System.out.println("1) Stable Extension (Propositional Logic)");
```

```
                        System.out.println("2) Stable Extension (Integer Linear
Programming)");
                        System.out.println("3) Stable Extension (NON-Integer Linear
Programming)");
                        System.out.println("4) Extraction of UNSAT Core of Clauses for
Stable Extension");
                        System.out.println("5) Extraction of UNSAT Core of
Nodes/Arguments for Stable Extension");
                        String problem = null;

                        // ask user to give desired problem and keep validating choice
until
                        // a valid one is given
                        do {
                                choice2 = new Scanner(System.in);
                                System.out.print("\nSubmit your choice (# of problem):
");
                                problem = choice2.nextLine();
                        } while (!Checker.isValidChoiceNumber(problem));

                        // ask user to give desired input method and keep validating
choice
                        // until a valid one is given
                        System.out.println("\nSelect Input Method:\n--------------------");
                        System.out.println("1) Random Graph\n2) Graph File");
                        String input = null;
                        do {
                                choice3 = new Scanner(System.in);
                                System.out.print("\nSubmit your choice (# of input
method): ");
                                input = choice3.nextLine();
                        } while (!Checker.isValidChoiceNumber(input) ||
Integer.parseInt(input) > 2);
```

```
// declare useful variables (number of problem, number of input,
// graph nodes and edges, graph density and the graph itself)
int probNum = Integer.parseInt(problem), inputNum =
Integer.parseInt(input), nodes, edges;
double density;
ArgumentationGraph g = null;
System.out.println("\nGive Graph Elements:\n--------------------");

// case of Random Graph
if (inputNum == 1) {
	String n, d;
	do { // validate user's inputs until a valid one is given
		choice4 = new Scanner(System.in);
		System.out.print("1) Give # of nodes (integer): ");
		n = choice4.nextLine();
		System.out.print("2) Give graph density (real): ");
		d = choice4.nextLine();
		System.out.println("");
	} while (!Checker.isAppropriateForRandom(n, d));

	// build random graph, using collected information
	nodes = Integer.parseInt(n);
	density = Double.parseDouble(d);
	edges = (int) Checker.extractEdgesFromDensity(density,
nodes);
	g = new ArgumentationGraph(nodes, edges, true, null);
}

// case of Graph File
else if (inputNum == 2) {
	String fileName = null;
	try {
```

```
                                        Scanner fileScanner;
                                        do { // validate user's inputs until a valid one is
given
                                             choice = new Scanner(System.in);
                                             System.out.print("1) Give a valid graph
file: ");
                                             fileName = choice.nextLine();
                                             fileScanner = new Scanner(new
File(fileName));
                                        } while
(!Checker.checkGraphFileValidity(fileScanner));

                                        // build graph using information from file given by
the user
                                        fileScanner.close();
                                        fileScanner = new Scanner(new File(fileName));
                                        g = new ArgumentationGraph(0, 0, false,
fileScanner);

                                   } catch (FileNotFoundException e) { // catch case of non-

          // existent file
                                        System.out.println("\nERROR: Could not read
input file!\nPROGRAM TERMINATED!");
                                        System.exit(0);
                                   }
                              }
                              System.out.println();

                              // print graph to file
                              g.printGraphToFile();

                              /**
```

```
 * PROCEED TO PROBLEM SOLVING
 **/
// case of classic Stable Extension (Propositional Logic solution)
if (probNum == 1) {
        // solve using lingeling
        g = LogicHandler.solveSE(args, g, probNum);
        System.out.println(
                        "3) LINGELING EXECUTION TIME: \t"
+ Long.toString(LogicHandler.lingelingExecutionTime) + "ms");
}
// case of Stable Extension as Integer Linear Problem
if (probNum == 2) {
        // solve using lingeling
        g = LogicHandler.solveSE(args, g, probNum);
        // create the equivalent Matlab input file based on
lingeling's
        // input file

    MatlabHandler.createMATLABInputFile("LingelingInput.cnf", g, true, -1.0);
        // print results messages
        System.out.println(
                        "3) LINGELING EXECUTION TIME: \t"
+ Long.toString(LogicHandler.lingelingExecutionTime) + "ms");
        System.out.println("\n4) APPLY THE RESULTS:\n");
        System.out.println("\ta) MatlabInput.m file has been
created succesfully!");
        System.out
                        .println("\tb) Load MatlabInput.m file on
MATLAB's console, press ENTER and wait for results");
        System.out.println("\tc) Check the variables for the Stable
Extension");
}
// case of Stable Extension as NON-Integer Linear Problem
```

```
// (Mixed Integer Linear Problem, MILP)
if (probNum == 3) {

        // create scanners to collect information from user
        Scanner eScan = new Scanner(System.in);
        Scanner eOption = new Scanner(System.in);
        String epsilon = null, epsilonOption = null;
        double e = -1.0;

        // print messages asking for user to input an epsilon
handling
        // choice
        System.out.println(
                "\nDetermine maximum distance from
bounds:\n-----------------" + "-----------------------");
        System.out.println("1) Minimize maximum distance (min
e)");
        System.out.println("2) Manually give maximum distance
(e = user's input))\n");
        do {
            System.out.print("Submit your choice (1 or 2): ");
            // collect user's submission for choice of handling
epsilon
            // and validate it until a valid one is given
            epsilonOption = eOption.nextLine();
        } while (epsilonOption.compareTo("1") != 0 &&
epsilonOption.compareTo("2") != 0);

        // case of epsilon given, manually, by user
        if (epsilonOption.compareTo("2") == 0) {
            do {
                do {
```

```
                                        System.out.println("\nGive the
maximum distance (epsilon) from upper/lower bounds "
                                                + "(real number
between 0.0 and 1.0, inclusive): ");
                                        epsilon = eScan.nextLine();
                                    } while (!epsilon.matches("[0-1].\\d+"));

                                    e = Double.parseDouble(epsilon);
                                } while (e > 1.0 || e < 0.0);
                            }

                            // solve using lingeling
                            g = LogicHandler.solveSE(args, g, probNum);

                            // create the equivalent Matlab input file based on
lingeling's
                            // input file

        MatlabHandler.createMATLABInputFile("LingelingInput.cnf", g, false, e);

                            // print results messages
                            System.out.println(
                                    "3) LINGELING EXECUTION TIME: \t"
+ Long.toString(LogicHandler.lingelingExecutionTime) + "ms");
                            System.out.println("\n4) APPLY THE RESULTS:\n");
                            System.out.println("\ta) MatlabInput.m file has been
created succesfully!");
                            System.out
                                    .println("\tb) Load MatlabInput.m file on
MATLAB's console, press ENTER and wait for results");
                            System.out.println("\tc) Check the variables for the Stable
Extension");
                            eScan.close();
```

```
                        eOption.close();
                }
                // case of extraction of UNSAT Core of Clauses
                if (probNum == 4) {
                        // solve using lingeling
                        g = LogicHandler.solveSE(args, g, probNum);

                        // solve using MAXSAT solver
                        LogicHandler.executeMAXSATSolver(g);

                        // print results messages
                        System.out.println(
                                        "3) LINGELING EXECUTION TIME: \t"
+ Long.toString(LogicHandler.lingelingExecutionTime) + "ms");
                        System.out.println(
                                        "4) MAXINO EXECUTION TIME: \t" +
Long.toString(LogicHandler.MAXSATExecutionTime) + "ms");
                        Long totalTime = LogicHandler.lingelingExecutionTime
+ LogicHandler.MAXSATExecutionTime;
                        System.out.println("5) TOTAL EXECUTION TIME: \t"
+ Long.toString(totalTime) + "ms");
                }
                // case of extraction of UNSAT Core of Nodes
                if (probNum == 5) {
                        try {
                                // export core of nodes from given graph
                                LogicHandler.exportUNSATCoreOfNodes(g);

                                // print results messages
                                System.out.println(
                                                "5) MAXINO EXECUTION
TIME: \t" + Long.toString(LogicHandler.MAXSATExecutionTime) + "ms");
```

```
                    } catch (IOException e) { // catch exception of
Input/Output
                                                                  //
problem
                        System.out.println("ERROR: COULD NOT
EXPORT CNF OUTPUT FILE!\nPROGRAM TERMINATED!");
                        System.exit(0);
                    }
                }

                // print all important files of problem (problem graph, input file,
                // results file) to a folder
                File fd = new File("RESULTS");
                fd.mkdir(); // create destination folder (as directory)
                int files = fd.listFiles().length;

                // send all files to folder
                File newDirectory = new File("RESULTS");
                newDirectory.mkdir();
                // create unique file names, based on the number of files in the
                // destination folder
                File newDirectory2 = new File("RESULTS/" + files);
                newDirectory2.mkdir();
                Runtime run = Runtime.getRuntime();
                try {
                    // move files to created destination folder
                    Process proc = run
                            .exec("mv MatlabInput.m LingelingResults
MAXSAT_core_nodes_output.txt MAXSAT_output.txt CoreOfClausesWeighted.cnf
UNSATCoreOfNodes.cnf exported_graph.txt "
                                    + " " + newDirectory2);
                    try {
                        proc.waitFor(); // wait for process to finish, before
```

```
                                                // proceeding
                                } catch (InterruptedException e2) { // catch exception
related

                // to the interruption of a

                // process
                                        System.out.println("ERROR: Could NOT MOVE
files to folder!" + "!\nPROGRAM TERMINATED!");
                                        System.exit(0);
                                }
                        } catch (IOException e1) { // catch exception related to the
                                                                //
input/output
                                System.out.println("ERROR: Could NOT MOVE files to
folder!" + "!\nPROGRAM TERMINATED!");
                                System.exit(0);
                        }
                        File gr = new File("EXP_RES/exported_graph.txt");
                        if (gr.exists()) {
                                // rename file
                                gr.renameTo(new File("EXP_RES/exported_graph" +
files + ".txt"));
                        }

                } catch (Exception except) { // collect all possible, unhandled
                                                                // exceptions
and terminate program
                        System.out.println("ERROR FOUND!\nPROGRAM
TERMINATED!");
                        System.exit(0);
                }
        }
```

```
}// END OF FILE
```

B.7.2 Κλάση «ArgumentationGraph» (κλάση που δημιουργεί και χειρίζεται αντικείμενα «γράφους πλαισίου Επιχειρηματολογίας»)

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.*;

/**
 * Class responsible for creating ArgumentationGraph objects. An
 * ArgumentationGraph object is a model of the Argumentation mechanism, with
 * each node depicting an argument, and each edge depicting an attacking
 * relationship between the two involved arguments.
 *
 * @author iyiang05
 *
 */
public class ArgumentationGraph {

	// declare variables
	int nodes, edges, clauses, links[][];

	/**
	 * Constructor of ArgumentationGraph objects (argumentation models)
	 *
	 * @param n
	 *            number of nodes in graph
	 * @param e
	 *            number of edges in graph
	 * @param isRandom
	 *            true if graph is random, false if not
	 * @param fs
```

```
     *           scanner which reads the input graph file, in case the graph is
     *           not random
     */
    public ArgumentationGraph(int n, int e, boolean isRandom, Scanner fs) {
        /**
         * CASE OF GRAPH BEING RANDOM (features manually given by
user)
         */
        if (isRandom) {
            // initialize graph features
            this.nodes = n;
            this.edges = e;
            this.clauses = 0;
            this.links = new int[n][n];
            LogicHandler.graph_edges = e;
            LogicHandler.graph_nodes = n;
            // randomly connect nodes with edges
            this.createLinksBetweenNodes();
        }
        /**
         * CASE OF GRAPH NOT BEING RANDOM (features given via an
input file)
         */
        else {
            int rows = 0;
            ArrayList<String> temp = new ArrayList<String>();
            while (fs.hasNextLine()) {
                rows++; // find number of nodes by counting lines in file
                temp.add(fs.nextLine());
            }
            // initialize graph features with information collected by the file
            this.nodes = rows;
            this.edges = 0;
```

```
			this.links = new int[rows][rows];
			this.clauses = 0;

			// calculate number of edges in graph and save graph to array
			for (int i = 0; i < rows; i++) {
				for (int k = 0; k < rows; k++) {
					if (temp.get(i).charAt(k) == '1') {
						this.edges++;
					}
					// turn chars into ints, by subtracting 48 from
ASCII codes
					this.links[i][k] = temp.get(i).charAt(k) - 48;
				}
			}
			LogicHandler.graph_edges = edges;
			LogicHandler.graph_nodes = nodes;
			this.printTables(); // print the adjacency matrix for observation
		}
	}

	/**
	 * Method which initializes the adjacency matrix and fills it with randomly
	 * created and randomly distributed links between nodes
	 */
	public void createLinksBetweenNodes() {
		for (int i = 0; i < this.nodes; i++)
			for (int k = 0; k < this.nodes; k++)
				this.links[i][k] = 0;
		createRandomLinksBetweenNodes(); // call function to create random
links
		this.printTables(); // print the adjacency matrix for observation
	}
```

```
/**
 * Method which is responsible for generating random links between 2 nodes
 * in a graph
 */
public void createRandomLinksBetweenNodes() {
	int i = 0, x, y; // declare variables
	Random r = new Random(); // create random-numbers' generator
	while (i < this.edges) {
		x = r.nextInt(this.nodes); // randomly picked attacker node
		y = r.nextInt(this.nodes); // randomly picked attacked node
		if (this.links[x][y] == 0 && x != y) {
			this.links[x][y] = 1; // if nodes do not attack themselves,
link
											// them
			i++;
		}
	}
}

/**
 * Method responsible for printing the graph's adjacency matrix to file
 *
 * @throws IOException
 *           error in case of anything going wrong with writing to file
 */
public void printGraphToFile() throws IOException {
	FileWriter fw = new FileWriter(new File("exported_graph.txt"));
	for (int i = 0; i < this.nodes; i++) {
		for (int k = 0; k < this.nodes; k++)
			fw.append("" + this.links[i][k]); // print to file
		fw.append("\n");
	}
	fw.flush(); // flush file writer
```

```
            fw.close(); // close file writer in order to write to file
        }

        /**
         * Method which prints all the information collected for the graph
         */
        public void printTables() {
            System.out.print(

        "\n\n*****************************************\n\tGRAPH
INFO\n*****************************************\n");
            System.out.print("a) Nodes in graph: \t" + this.nodes + "\nb) Edges in
graph: \t" + this.edges);
            System.out.print("\nc) Edges calculated based on the following formula:
\n\n\tEdges (directed graph) = "
                        + "floor {d*n*(n-1)} ,\n\nwhere:\n d: density \n n:
number of nodes");
            System.out.print("\n\nd) Table of links in graph:\n\n");
            for (int i = 0; i < this.nodes; i++) {
                for (int k = 0; k < this.nodes; k++)
                    System.out.print(this.links[i][k]); // print adjacency matrix
                System.out.print("\n");
            }
        }
}// END OF FILE
```

B.7.3 Κλάση «Checker» (κλάση που διεξάγει ελέγχους και επικυρώνει δεδομένα)

```
import java.util.Scanner;

/**
 * Class responsible for offering methods used for all kinds of validations and
 * checks
 *
```

```
 * @author iyiang05
 *
 */
public class Checker {

	/**
	 * Static method which checks if the user has given a valid choice for
	 * problem selection (must be a number between 1 and 5, inclusive)
	 *
	 * @param p
	 *            user's choice
	 * @return true or false, accordingly
	 */
	public static boolean isValidChoiceNumber(String p) {
		// check if user's choice is a number, between 1 and 5, inclusive
		if (isValidUserInput(p) && Integer.parseInt(p) < 6 &&
Integer.parseInt(p) > 0)
			return true;
		return false;
	}

	/**
	 * Static method which checks if user's choice is a number, using regular
	 * expressions
	 *
	 * @param input
	 *            User's choice
	 * @return true or false, accordingly
	 */
	public static boolean isValidUserInput(String input) {
		if (!(input.matches("[0-9]+"))) // check if input is a number
			return false;
		return true;
```

```
	}

	/**
	 * Static method, responsible for validating input given by user. Checks if
	 * the number of nodes and density co-efficient are valid integer, and real
	 * numbers, respectively. Also checks if they produce the right amount of
	 * edges for the given graph (really high density could mean 2 or more edges
	 * exiting one node and both entering the same node, which is considered
	 * redundant)
	 *
	 * @param n
	 *            number of nodes
	 * @param d
	 *            density co-efficient
	 * @return true or false, accordingly
	 */
	public static boolean isAppropriateForRandom(String n, String d) {
		// check if any input is empty or invalid
		if (n.compareTo("") == 0 || d.compareTo("") == 0 || n.length() <= 0 ||
d.length() <= 0
				|| !Checker.isValidUserInput(n) ||
!Checker.isRealNumber(d))
			return false;
		if (!Checker.isRelevantEdgeNum(
				(int)
Checker.extractEdgesFromDensity(Double.parseDouble(d), Integer.parseInt(n)),
				Integer.parseInt(n))) {
			return false;
		}
		return true;
	}

	/**
```

```
 * Static method which checks if an input, passed as argument, is a real
 * number, using regular expressions. Used to validate density given by
 * user.
 *
 * @param input
 *          User's input
 * @return true or false, accordingly
 */
public static boolean isRealNumber(String input) {
        if (!(input.matches("[0-1]*\\.[0-9]+"))) // validate input
                return false;
        return (input.matches("[0-1]*\\.[0-9]+")); // return result
}

/**
 * Static method which checks if the number of edges, produced by a given
 * number of nodes, is valid. Takes into consideration that no nodes can
 * attack themselves and that no nodes attack another node twice. Those
 * cases are considered redundant.
 *
 * @param edges
 *          number of edges in graph
 * @param nodes
 *          number of nodes in graph
 * @return true or false, accordingly
 */
public static boolean isRelevantEdgeNum(int edges, int nodes) {
        // check if edges number is greater than the value: nodes * (node-1)
        if (edges > (nodes * (nodes - 1))) { // validate number of edges
                System.out.println("\n*** WRONG INPUT ***");
                System.out.println("Reason: Edges number is irrelevant to the
problem, which implies that:");
```

```
				System.out.print("\ta) at least 1 node attacks itself \t\t-->
irrelevant \n\t");
				System.out.print("b) at least 1 node attacks another node twice \t-
-> redundant\n");
				return false;
			}
			return true;
		}

		/**
		 * Static method which calculates the number of edges, based on given number
		 * of nodes and density co-efficient
		 *
		 * @param density
		 *            density of graph
		 * @param nodes
		 *            nodes in graph
		 * @return number of edges
		 */
		public static double extractEdgesFromDensity(double density, double nodes) {
			return (density * (nodes * (nodes - 1.0)));
		}

		/**
		 * Static method which calculates the density co-efficient, based on given
		 * number of nodes and number of edges in graph.
		 *
		 * @param edges
		 *            edges in graph
		 * @param nodes
		 *            nodes in graph
		 * @return calculated density co-efficient
		 */
```

```
public static double extractDensityFromEdges(double edges, double nodes) {
	return (edges / (nodes * (nodes - 1)));
}

/**
 * Static method which checks the input graph file for validity
 *
 * @param fileScanner
 *            scanner which reads the file
 * @return true or false, accordingly
 */
public static boolean checkGraphFileValidity(Scanner fileScanner) {
	int n = 0, c = 0, numOfLines = 0;
	while (fileScanner.hasNextLine()) { // keep reading the file
		String currentLine = fileScanner.nextLine(); // collect current line
		if (c == 0) {
			// get length of first line to compare with every other line
			n = currentLine.length();
			c = -1;
		}
		// 1st case of invalid file: Graph is not a square because of
		// different column number on two or more rows (NxN)
		if (currentLine.length() != n)
			return false;

		// 2nd case of invalid file: contains characters other than 1 and 0
		for (int p = 0; p < currentLine.length(); p++) {
			if (currentLine.charAt(p) != '0' && currentLine.charAt(p)
!= '1' && currentLine.charAt(p) != '\n')
				return false;
		}
		numOfLines++;
	}
```

```
			// continue 1st case of invalid file: Graph is not a square because of
			// number of rows not being equal to the number of columns
			if (numOfLines != n)
				return false;
			return true;
		}
} // END OF FILE
```

B.7.4 Κλάση «LogicHandler» (κλάση που χειρίζεται διαδικασίες που σχετίζονται με την Προτασιακή Λογική, όπως: δημιουργία αρχείων CNF, καλέσματα επιλυτών, ερμηνεία αποτελεσμάτων κ.λπ.)

```
import java.io.*;
import java.util.*;

/**
 * Class which offers methods, responsible for handling all processes related to
 * Logical computation (creating input files for solvers, translating input to
 * the correct form of clauses, calling solvers for solving problems,
 * interpreting and displaying results on screen etc.)
 *
 * @author iyiang05
 *
 */
public class LogicHandler {
		// declare static variables of class
		static char problemStatus; // U if UNSAT, S if SAT problem
		static String stableExtension;
		static ArrayList<String> UNSAT_Core_static;
		static int graph_edges, graph_nodes; // edges and nodes of graph
		static double graph_density; // density of graph
```

```
	static long lingelingExecutionTime, MAXSATExecutionTime, startTime; //
execution

										// times

	/**
	 * Static method responsible for calling other methods to solve the problem
	 * of Classic Stable Extension (Stable Extension defined as a problem of
	 * Propositional SATisfiability)
	 *
	 * @param args
	 *            arguments passed from command line
	 * @param g
	 *            the graph created from previous steps
	 *
	 * @param probNum
	 *            number of problem selected by the user
	 * @return graph
	 */
	public static ArgumentationGraph solveSE(String args[], ArgumentationGraph
g, int probNum) {
		if (g != null) { // if a correct graph is given, proceed
			try {
				// export CNF statements from graph and solve
				LogicHandler.exportCNFStatementLogic(g, probNum);
				// move occurring filers to a folder
				LogicHandler.moveFilesToFolder();
			} catch (IOException e1) { // collect input/output exception
				System.out.println("ERROR: Problem Exporting CNF
output file!\nPROGRAM TERMINATED!");
				System.exit(0); // terminated program
			}
		}
```

```
		return g; // return graph again
	}

	/**
	 * Static method responsible for calling other methods to extract the Core
	 * of Nodes from a problem described from a graph given
	 *
	 * @param g
	 *          graph depicting the problem
	 * @throws IOException
	 *           error of input/output nature
	 */
	public static void exportUNSATCoreOfNodes(ArgumentationGraph g) throws
IOException {

		// export logic in CNF (create all important clauses)
		int vars = exportCNFForUNSATCoreOfNodesNew(g);

		// execute MAXSAT solver for the CNF file created, to solve the
problem
		executeMAXSATSolverForUNSATCoreOfNodes();

		// extract core of nodes from solution file and present it
		extractUNSATCoreOfNodesFromFile(vars);
	}

	/**
	 * Static method responsible from extracting the Core of Nodes from a file
	 * of results returned by the MAXSAT solver
	 *
	 * @param vars
	 *          number of variables in problem
	 */
```

```
private static void extractUNSATCoreOfNodesFromFile(int vars) {

        // model extracted from MAXSAT solver's results file
        ArrayList<String> extractedModel = new ArrayList<String>();
        // set 1: arguments of Stable Extension
        ArrayList<String> inSE = new ArrayList<String>();
        // set 2: arguments outside of Stable Extension BUT part of the problem
        ArrayList<String> outOfSE = new ArrayList<String>();
        // set 3: arguments entirely removed from the problem (UNSAT Core of
        // Nodes)
        ArrayList<String> outOfProblem = new ArrayList<String>();

        try {
                // read file and find model
                Scanner fScan = new Scanner(new
File("MAXSAT_core_nodes_output.txt"));
                boolean foundModel = false;
                while (fScan.hasNextLine()) {
                        String currLin = fScan.nextLine();
                        if (currLin.charAt(0) == 'v')
                                foundModel = true;
                        if (foundModel) { // collect and split strings
                                String stemp[] = currLin.split(" ");
                                for (String s : stemp)
                                        extractedModel.add(s); // collect model
from splits of

        // strings
                        }
                }
                // first remove 'v' character from position 0
                extractedModel.remove(0);
```

```
                    // extract the 3 sets (inSE, outOfSE, outOfProblem)
                    // trim strings from unwanted characters ('-' or spaces etc.)
                    int cv = 1;
                    for (String ems : extractedModel) {
                        if (cv <= (vars / 2) && !ems.contains("-"))
                            inSE.add(ems + "");
                        else if (cv <= (vars / 2) && ems.contains("-"))
                            outOfSE.add(ems.replace("-", "") + "");
                        else if (cv > (vars / 2) && ems.contains("-"))

        outOfProblem.add((Integer.parseInt(ems.replace("-", "")) - (vars / 2)) + "");
                        cv++;
                    }

                    // fix sets
                    for (String o : outOfProblem)
                        outOfSE.remove(o);

                    // print all results
                    System.out.println(

        "\n******************************************\n\tRESULTS\n******
****************************************");
                    System.out.println("1) STABLE EXTENSION: " + inSE);
                    System.out.println("2) IN PROBLEM & OUT OF STABLE
EXTENSION: " + outOfSE);
                    System.out.println("3) UNSAT CORE of
NODES/ARGUMENTS: " + outOfProblem);
                    System.out.println(
                            "\n4) EXPLANATION: \n\tBy removing the
arguments of the UNSAT core from an UNSATISFIABLE problem's knowledge base,
it becomes SATISFIABLE\n");
                    fScan.close();
```

```
		} catch (FileNotFoundException e) { // collect error of not finding file
											// to
read
			System.out.println("ERROR: Problem opening input
file!\nPROGRAM TERMINATED!");
			System.exit(0); // terminated program
		}
	}

	/**
	 * Static method, from which the MAXSAT solver is called, to solve the
	 * problem of finding the UNSAT Core of Nodes (Arguments) in a graph
	 */
	private static void executeMAXSATSolverForUNSATCoreOfNodes() {
		// create a process builder
		ProcessBuilder MAXSatExecutor = new ProcessBuilder("./maxino-
2015-k16-static", "UNSATCoreOfNodes.cnf");
		// redirect results on terminal, to an output file, for later use
		MAXSatExecutor.redirectOutput(new
File("MAXSAT_core_nodes_output.txt"));
		try {
			// start counting execution time
			LogicHandler.startTime = System.currentTimeMillis();
			// start the process via the process builder
			Process MAXSatProcess = MAXSatExecutor.start();
			try {
				System.out.print(

	"\n\n*******************************************\n\tPROCEDURES\n
*******************************************\n");
				System.out.println("\n1) Running Maxino\n2) Results are
being written to file: Please wait . . .");
```

```
MAXSatProcess.waitFor(); // wait process for completion,
before
                                                                                                //
proceeding
                        } catch (InterruptedException e) { // collect error which occurs

        // with the interruption of the

        // process
                                System.err.print("\nERROR: MAXSAT Solver (maxino-
2015-k16-static) executable or ");
                                System.err.println("MAXSAT_core_nodes_output.txt file
NOT FOUND!\nPROGRAM TERMINATED!");
                                System.exit(0); // terminate program
                        }
                } catch (IOException e) { // catch error which occurs if the output file
                                                                        // is not found or
could not write to file
                        System.out.println(
                                        "ERROR: MAXSAT Solver executable or
MAXSAT_core_nodes_output.txt file NOT FOUND!\nPROGRAM TERMINATED!");
                        System.exit(0); // terminate program
                }
                // calculate total execution time of process
                LogicHandler.MAXSATExecutionTime = System.currentTimeMillis() -
LogicHandler.startTime;
                // print success message
                System.out.println("\n2) MAXSAT_core_nodes_output.txt file has been
created succesfully!");
        }

        /**
         * Static method, from which the MAXSAT solver is called, to solve the
```

```
         * problem of finding the UNSAT Core of Clauses (edges) in a graph
         *
         * @param g
         *            graph depicting the problem
         */
        public static void executeMAXSATSolver(ArgumentationGraph g) {

                // create weighted CNF version of Lingeling's input file
                LogicHandler.addWeightsToCNFFile(new
File("LingelingResults/LingelingInput.cnf"), g);

                // create process builder
                ProcessBuilder MAXSatExecutor = new ProcessBuilder("./maxino-
2015-k16-static", "CoreOfClausesWeighted.cnf");

                // redirect results of MAXSAT Solver, from terminal, to an output file
                MAXSatExecutor.redirectOutput(new File("MAXSAT_output.txt"));
                try {
                        // start counting execution time
                        LogicHandler.startTime = System.currentTimeMillis();
                        // start process
                        Process MAXSatProcess = MAXSatExecutor.start();
                        try {
                                System.out.print(

        "\n\n*******************************************\n\tPROCEDURES\n
*******************************************\n");
                                System.out.println("\n1) Running Maxino\n2) Results are
being written to file: Please wait . . .");
                                MAXSatProcess.waitFor(); // wait process for completion,
before
                                                                                                        //
proceeding
```

```
                    } catch (InterruptedException e) { // collect error of interrupted

        // process
                            System.err.print("\nERROR: MAXSAT Solver (maxino-
2015-k16-static) executable or ");
                            System.err.println("MAXSAT_output.txt file NOT
FOUND!\nPROGRAM TERMINATED!");
                            System.exit(0); // terminate program
                    }
                } catch (IOException e) { // collect error of not being able to find the
                                                        // output file, or
write to it
                    System.out.println(
                                "ERROR: MAXSAT Solver executable or
MAXSAT_output.txt file NOT FOUND!\nPROGRAM TERMINATED!");
                    System.exit(0); // terminate program
                }
                // calculate total execution time of process
                LogicHandler.MAXSATExecutionTime = System.currentTimeMillis() -
LogicHandler.startTime;
                // print success message
                System.out.println("3) MAXSAT_output.txt file has been created
succesfully!");
                LogicHandler.extractMAXSATResults(); // extract results from output
file
        }

        /**
         * Method responsible for extracting the solution model, from the results in
         * the output file, returned by the MAXSAT solver.
         */
        private static void extractMAXSATResults() {
                try {
```

```
                    // create scanner to read the results file
                    Scanner fScanner = new Scanner(new
File("MAXSAT_output.txt"));
                    // create model variables
                    ArrayList<String> maxsatModelSE = new ArrayList<String>();
                    ArrayList<String> maxsatModelNonSE = new
ArrayList<String>();
                    boolean startCollecting = false;
                    while (fScanner.hasNextLine()) {
                        String currentLine = fScanner.nextLine();
                        if (currentLine.charAt(0) == 'v')
                            startCollecting = true;

                        // collect clear models
                        if (startCollecting) {
                            String[] tempTokens = currentLine.split(" ");
                            for (String s : tempTokens) {
                                if (s.matches("[0-9]+"))
                                    maxsatModelSE.add(s);
                                else if (!s.matches("v"))

    maxsatModelNonSE.add(s.substring(1));
                            }
                        }
                    }
                    // extract UNSAT core process
                    LogicHandler.UNSAT_Core_static = null;
                    LogicHandler.UNSAT_Core_static =
LogicHandler.extractUNSATCore(maxsatModelSE);

                    // print results
                    System.out.println(
```

```
	"\n****************************************\n\tRESULTS\n******
**********************************");
			System.out.println("1) UNSAT CORE OF CLAUSES: " +
UNSAT_Core_static);
			System.out.println(
					"\n2) EXPLANATION:\n\tBy removing the
clauses of the UNSAT core from an UNSATISFIABLE problem's knowledge base, it
becomes SATISFIABLE\n");
			fScanner.close(); // close scanner
		} catch (FileNotFoundException e) { // catch error, in case the results
											// file
is not found
			System.out.println(
					"ERROR: MAXSAT Solver executable or
MAXSAT_output.txt file NOT FOUND!\nPROGRAM TERMINATED!");
			System.exit(0); // terminate program
		}
	}

	/**
	 * Static method which returns the clauses removed to make the problem
	 * SATISFIABLE. Generates all possible clauses that can be removed.
	 *
	 * @param SE
	 *            Stable Extension returned by the MAXSAT Solver
	 */
	private static ArrayList<String>
generatePossibleUNSATCoreClauses(ArrayList<String> SE) {
		ArrayList<String> possibleUNSATCoreClauses = new
ArrayList<String>();

		// to find the minimum UNSAT core of clauses that are needed to be
```

```
		// removed, to set the problem to SATISFIABLE, we must extract the
least
		// amount of clauses possible, which corresponds to the cases of an
		// argument attacking each other (those are the clauses which say: NOT
		// BOTH arguments are in Stable Extension)
		for (String s1 : SE)
			for (String s2 : SE) {
				possibleUNSATCoreClauses.add("1 -" + s1 + " -" + s2);
				possibleUNSATCoreClauses.add("1 -" + s2 + " -" + s1);
			}
		// call function to remove duplicate clauses generated
		return removeDuplicateClauses(possibleUNSATCoreClauses);
	}

	/**
	 * Private, auxiliary static method, which is used to remove duplicate
	 * clauses from a set of generated clauses.
	 *
	 * @param a
	 *            set of clauses to reduce
	 * @return set of clauses, without duplicates
	 */
	private static ArrayList<String> removeDuplicateClauses(ArrayList<String> a)
{
		for (int i = 0; i < a.size(); i++)
			for (int k = i + 1; k < a.size(); k++)
				if (a.get(i).compareTo(a.get(k)) == 0) // check for
duplicates
					a.set(k, " "); // remove duplicate
		return a; // return set of clauses, without duplicates
	}

	/**
```

```
	 * Private, auxiliary static method which finds the Core of Clauses, by
	 * comparing the problematic, UNSATisfiability-causing input file of the
	 * MAXSAT solver, and all the possible clauses created, that could cause the
	 * issue. It removes the ones presented in both.
	 *
	 * @param SE
	 *            Stable Extension, as returned by the MAXSAT Solver
	 *
	 * @return Core of Clauses (clauses that have to be removed from the
	 *         original input file)
	 */
	private static ArrayList<String> extractUNSATCore(ArrayList<String> SE) {

		// generate clauses to discover the members of the UNSAT core
		ArrayList<String> possibleUNSATCoreClauses =
LogicHandler.generatePossibleUNSATCoreClauses(SE),
				UNSAT_Core = new ArrayList<String>();
		try {
			// create scanner to read file
			Scanner fScanner = new Scanner(new
File("CoreOfClausesWeighted.cnf"));

			while (fScanner.hasNextLine()) {
				String curLin = fScanner.nextLine(); // collect current line
				for (String pc : possibleUNSATCoreClauses)
					if (curLin.compareTo(pc + " 0") == 0)
						UNSAT_Core.add(curLin); // add to
UNSAT Core of Clauses,

	// all clauses present in both

	// the possible problematic
```

```
	// clauses' set AND the input

	// file
				}
				fScanner.close(); // close scanner
			} catch (FileNotFoundException e) { // catch error if input file not
													//
found
				System.out.println("ERROR: LingelingInput.cnf file NOT
FOUND!\nPROGRAM TERMINATED!");
				System.exit(0); // terminate program
			}
			return UNSAT_Core; // return core of clauses
	}

	/**
	 * Private, auxiliary, static method which is responsible for creating the
	 * input file for the MAXSAT solver, in order to solve the problem of
	 * finding the UNSAT Core of Nodes (Arguments) in a graph.
	 *
	 * @param g
	 *          graph depicting the problem
	 *
	 * @return number of variables needed for the problem
	 *
	 * @throws IOException
	 *           exception which occurs if there is a problem while writing to
	 *           the input file
	 */
	private static int exportCNFForUNSATCoreOfNodesNew(ArgumentationGraph
g) throws IOException {
			// create file write to write to file
```

```
		FileWriter fw = new FileWriter(new File("UNSATCoreOfNodes.cnf"));

		// calculate number of clauses
		g.clauses = g.nodes * 3;
		for (int i = 0; i < g.nodes; i++)
			for (int x = i + 1; x < g.nodes; x++)
				if (g.links[i][x] == 1 || g.links[x][i] == 1)
					g.clauses = g.clauses + 1;
		// append comment and information line to the problem
		fw.append("c UNSATCoreOfNodes.cnf\np wcnf " + (g.nodes * 2) + " "
+ g.clauses + " 1000" + "\n");

		// create constraints
		for (int i = 0; i < g.nodes; i++) {
			// soft clause: try to add node to problem
			fw.append("1 " + (i + 1 + g.nodes) + " 0\n");
			// if an argument is in Stable Extension, it remains in the problem
			fw.append("1000 -" + (i + 1) + " " + (i + 1 + g.nodes) + " 0\n");
			// attack constraints (no two linked nodes can co-exist in Stable
			// Extension)
			for (int x = i + 1; x < g.nodes; x++)
				if (g.links[i][x] == 1 || g.links[x][i] == 1)
					fw.append("1000 -" + (i + 1) + " -" + (x + 1) + "
0\n");
			// all arguments constraint (presence)
			int indexAttacksA[] = new int[g.nodes];

			// call function to find all nodes attacking a specific node
			findThreatNodes(i, g, indexAttacksA);
			// a node is either in Stable Extension OR is entirely out of
			// problem OR any number of nodes attacking it are members of
the
			// Stable Extension
```

```
                fw.append("1000 " + (i + 1) + " -" + (i + 1 + g.nodes));
                for (int m = 0; m < g.nodes; m++)
                    if (indexAttacksA[m] == 1)
                        fw.append(" " + (m + 1));
                fw.append(" 0\n");
            }
            fw.flush(); // flush file writer
            fw.close(); // close file writer
            return (g.nodes * 2); // return number of variables used for the problem
                                  // (2 for each argument)
        }

        /**
         * Static method used to create the CNF logic of the classic Stable
         * Extension problem, based on a graph given. The resulted theory is written
         * to a file, which consists the input file for lingeling.
         *
         * @param g
         *          graph depicting the problem
         *
         * @param probNum
         *          number of problem selected by the user
         *
         * @throws IOException
         *           exception which occurs if there is a problem while writing to
         *           the input file
         */
        public static void exportCNFStatementLogic(ArgumentationGraph g, int
probNum) throws IOException {

            // create file writer to write theory to input file
            FileWriter fw = new FileWriter(new File("LingelingInput.cnf"));
```

```
// collect number of clauses and variables needed to describe the
// problem
g.clauses = g.nodes;
for (int i = 0; i < g.nodes; i++)
        for (int x = i + 1; x < g.nodes; x++)
                if (g.links[i][x] == 1 || g.links[x][i] == 1)
                        g.clauses = g.clauses + 1; // (NOT a) OR (NOT b)
// append comments and information line to file
fw.append("c LingelingInput.cnf\np cnf " + g.nodes + " " + g.clauses +
"\n");

// create CNF rules for attacking/attacked nodes
for (int i = 0; i < g.nodes; i++)
        for (int x = i + 1; x < g.nodes; x++)
                if (g.links[i][x] == 1 || g.links[x][i] == 1)
                        // no two connected nodes can co-exist in Stable
Extension
                        fw.append("-" + (x + 1) + " " + "-" + (i + 1) + "
0\n");

// collect all nodes that attack to each node and create necessary
// clauses
for (int x = 0; x < g.nodes; x++) {
        int indexAttacksA[] = new int[g.nodes];
        // collect all nodes attacking to a specific node
        findThreatNodes(x, g, indexAttacksA);
        fw.append("" + (x + 1));
        for (int m = 0; m < g.nodes; m++)
                if (indexAttacksA[m] == 1)
                        fw.append(" " + (m + 1));
        // a node is either in Stable Extension OR any number of nodes
which
        // attack it are members of Stable Extension
```

```
                fw.append(" 0\n");
            }
            fw.flush(); // flush file writer
            fw.close(); // close file writer
            // execute lingeling solver to solve the problem described in the file
            // created by this method
            if (probNum == 2 || probNum == 3)
                return; // DONT SOLVE FOR PROBLEMS 2 AND 3 (where
MATLAB is needed)
            LogicHandler.executeLingeling();
        }

        /**
         * Static method which finds all nodes that attack a specific node
         *
         * @param nodeAttacked
         *            the node for which the method finds all nodes that attack it
         * @param g
         *            graph depicting the problem
         * @param threatNodes
         *            all nodes attacking nodeAttacked
         * @return 0
         */
        public static int findThreatNodes(int nodeAttacked, ArgumentationGraph g, int
threatNodes[]) {
            for (int i = 0; i < g.nodes; i++) {
                threatNodes[i] = 0;
                if (g.links[i][nodeAttacked] == 1)
                    threatNodes[i] = 1;
            }
            return 0;
        }
```

```java
	/**
	 * Static method which executes lingeling to solve the Classic problem of
	 * Stable Extension (Stable Extension as a problem of Propositional
	 * Satisfiability)
	 */
	public static void executeLingeling() {
		// create process builder for running lingeling with the specified input
		// file
		ProcessBuilder lingelingExecutor = new ProcessBuilder("./lingeling",
"LingelingInput.cnf");
		// redirect results to an output file, for later use
		lingelingExecutor.redirectOutput(new File("LingelingOutput.cnf"));
		try {
			// start counting for execution time calculations
			LogicHandler.startTime = System.currentTimeMillis();
			// start process
			Process lingelingProcess = lingelingExecutor.start();
			try {
				System.out.print(

	"\n\n****************************************\n\tPROCEDURES\n
****************************************\n");
				System.out.println("\n1) Results are being written to file:
Please wait . . .");
				lingelingProcess.waitFor(); // wait until process
completion,
																//
before proceeding
			} catch (InterruptedException e) { // case of interruption of

	// process
				System.out.println(
```

```
					"ERROR: Lingeling executable or
LingelingInput.cnf file NOT FOUND OR process INTERRUPTED unexpectedly"
							+ "!\nPROGRAM
TERMINATED!");
				System.exit(0); // terminate program
			}
		} catch (IOException e) { // case of not being able to write to file
			System.out.println(
					"ERROR: Lingeling executable or
LingelingInput.cnf file NOT FOUND OR process INTERRUPTED unexpectedly"
							+ "!\nPROGRAM
TERMINATED!");
			System.exit(0); // terminate program
		}
		// calculate total execution time for solution process
		LogicHandler.lingelingExecutionTime = System.currentTimeMillis() -
LogicHandler.startTime;
		// print success message
		System.out.println("\n2) LingelingOutput.cnf file has been created
succesfully!");
		LogicHandler.extractSATResults(); // extract results from returned
model
	}

	/**
	 * Method which extracts the results (solution model and SATISFIABILITY
	 * status) from the output of lingeling
	 */
	private static void extractSATResults() {
		// read lingeling's results which are redirected to an output file
		File f = new File("LingelingOutput.cnf");
		ArrayList<String> extractedModel = new ArrayList<String>();
		try {
```

```java
			System.out.println(

	"\n******************************************\n\tRESULTS\n******
************************************");
			Scanner fr = new Scanner(f);
			while (fr.hasNextLine()) {
				String buffer = fr.nextLine();
				// collected problems status (SAT or UNSAT)
				if (buffer.matches("s SATISFIABLE")) {
					System.out.println("\n1) PROBLEM STATUS:\t
SATISFIABLE");
					problemStatus = 'S';
				} else if (buffer.matches("s UNSATISFIABLE")) {
					System.out.println(
							"\n1) PROBLEM STATUS:\t
UNSATISFIABLE\n******************************************");
					extractedModel = null;
					problemStatus = 'U';
				}
				// collect model from output file
				if (buffer.charAt(0) == 'v') {
					do {
						extractedModel.add(buffer);
						if (fr.hasNextLine())
							buffer = fr.nextLine();
					} while (buffer.charAt(0) != 'c' &&
fr.hasNextLine());
				}
			}
			fr.close(); // close file scanner

			// split model into tokens
			if (extractedModel != null) {
```

```
                                        String[] tokens = null;
                                        ArrayList<String> stableExtension = new
ArrayList<String>();
                                        ArrayList<String> NONstableExtension = new
ArrayList<String>();
                                        int count = 0;
                                        do {
                                              tokens = extractedModel.get(count++).split(" ");
                                              for (String token : tokens)
                                                    if (token.matches("\\-[0-9]+"))

      NONstableExtension.add(token.replace("-", ""));
                                                    else if (token.matches("[0-9]+"))
                                                          stableExtension.add(token);
                                        } while (count < extractedModel.size());

                                        // remove terminal 0 from model
                                        if (stableExtension.contains("0"))

      stableExtension.remove(stableExtension.indexOf("0"));

                                        // print final results
                                        System.out.println("2) RESULTS:\n-------------------------
-----------------\n\tStable Extension: "
                                                    + stableExtension.size() + " nodes\n--------
----------------------------------\n"
                                                    + stableExtension + "\n");
                                  }
                            } catch (FileNotFoundException e) { // case of output file not found
                                  System.out.println("ERROR: LingelingOutput.cnf file NOT
FOUND!" + "!\nPROGRAM TERMINATED!");
                                  System.exit(0); // terminate program
                            }
```

```
}

/**
 * Static method responsible for adding weights to a simple CNF file,
 * turning it into a WCNF file, where each clause is assigned a weight.
 * Creates soft and hard constraints from previously same priority
 * constraints
 *
 * @param f
 *          CNF file to assign weights and turn to WCNF
 * @param g
 *          graph describing the problem
 */
private static void addWeightsToCNFFile(File f, ArgumentationGraph g) {

        Scanner fs = null;
        try {
                fs = new Scanner(f);// read file using a scanner

                FileWriter fw = null;
                int counter = 0;
                try {
                        // write a new file out of the old one, with the addition of
                        // weights
                        fw = new FileWriter("CoreOfClausesWeighted.cnf");
                        while (fs.hasNextLine()) {
                                counter++;
                                String currLin = fs.nextLine();
                                if (counter == 1)
                                        // append comment line
                                        fw.append("c
CoreOfClausesWeighted.cnf\n");
                                else if (counter == 2) {
```

```
									// append problem information line
									String[] splits = currLin.split(" ");
									fw.append("p wcnf " + splits[2] + " " +
splits[3] + " 1000\n");
								} else {
									// append weighted clauses
									if (currLin.contains("-"))
										// append soft clause
										fw.append("1 " + currLin + "\n");
									else
										// append hard clause
										fw.append("1000 " + currLin +
"\n");
								}
							}
							fw.flush(); // flush file writer to write to file
							fw.close(); // close file writer
						} catch (IOException e) { // case of not being able to write or
															// create file
							System.out.println("ERROR: Could NOT create file
CoreOfClausesWeighted.cnf!\nPROGRAM TERMINATED!");
						}
				} catch (FileNotFoundException e1) { // case of not being able to read

		// the input file
						System.out.println("ERROR: Could NOT read file
LingelingInput.cnf!\nPROGRAM TERMINATED!");
				}
		}

		/**
		 * Static method responsible for moving files created, to another folder
		 */
```

```
	public static void moveFilesToFolder() {
		// send all Lingeling files to folder
		File newDirectory = new File("LingelingResults");
		newDirectory.mkdir(); // create directory
		Runtime run = Runtime.getRuntime(); // get run time
		try {
			// run terminal process through JAVA
			Process proc = run.exec("mv LingelingInput.cnf
LingelingOutput.cnf LingelingResults");
			try {
				proc.waitFor(); // wait for process completion before
proceeding
			} catch (InterruptedException e2) { // case of interrupting process
				System.out.println("ERROR: Could NOT MOVE files to
folder!" + "!\nPROGRAM TERMINATED!");
				System.exit(0); // terminate program
			}
		} catch (IOException e1) { // case of not being able to create new
										// directory
			System.out.println("ERROR: Could NOT MOVE files to folder!"
+ "!\nPROGRAM TERMINATED!");
			System.exit(0); // terminate program
		}
	}
} // END OF FILE
```

B.7.5 Κλάση «MatlabHandler» (κλάση που χειρίζεται διαδικασίες που σχετίζονται με τον Γραμμικό Προγραμματισμό, όπως: «μετάφραση» εισόδου σε γραμμικούς περιορισμούς, δημιουργία αρχείων Matlab κ.λπ.)

```
import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;
```

```
/**
 * Class which offers methods, responsible for handling all processes related to
 * Integer/Mixed-Integer Linear Programming computation (creating script files
 * for MATLAB, translating input to the correct form of clauses using
 * mathematical equalities/inequalities etc.)
 *
 * @author iyiang05
 *
 */
public class MatlabHandler {

	/**
	 * Method responsible for creating script files for MATLAB, for the problem
	 * of Stable Extension, both as a problem of Integer Linear Programming
	 * (ILP), as well as a problem of Mixed-Integer Linear Programming (MILP)
	 *
	 * @param equivalentLingelingInput
	 *            lingeling's input file to translate to MATLAB's accepted
	 *            format
	 * @param g
	 *            graph describing the problem
	 * @param intDesired
	 *            true if creating file for Integer Linear Programming, false
	 *            for Mixed-Integer Linear Programming
	 * @param epsilon
	 *            maximum allowed distance of arguments' values from the bounds
	 */
	public static void createMATLABInputFile(String equivalentLingelingInput,
ArgumentationGraph g, boolean intDesired,
			double epsilon) {
		// declare variables
		boolean integerDesired = intDesired, equalityExists = false;
```

```
		int rightPartsInequalities[] = new int[g.clauses], v = 0, clauseCount = -1;
		double nonIntRightParts[] = new double[g.nodes * 9 + 1];
		ArrayList<Integer> equalityIndexes = new ArrayList<Integer>();
		FileWriter fw;

		try {
			// create MATLAB input file and write comment lines first
			fw = new FileWriter(new File("MatlabInput.m"));
			fw.append("% *********************************\n");
			fw.append("% STABLE EXTENSION PROBLEM\n%
*********************************\n");
			fw.append("% File\t: \tMatlabInput.m\n% Author: \tIoannis
Yiangou\n");
			fw.append("% *********************************\n\n");

			// re-open lingeling file to support MATLAB file creation
			Scanner reopenedScanner = new Scanner(new
File("LingelingResults/" + equivalentLingelingInput));
			reopenedScanner.nextLine(); // skip the two (comment) lines of
file
			reopenedScanner.nextLine();

			/**
			 * Case of creating script file for the Stable Extension as a
			 * problem of Integer Linear Programming (ILP = problem with
only
			 * integer-valued variables)
			 */
			if (integerDesired) {
				// append declaration of objective function to file
				fw.append("% Declare function\nf = [");
				for (v = 0; v < g.nodes; v++)
					fw.append("1; ");
```

```
                                        // append declaration of integer variables (all of them) to
file
                                        fw.append("];\n\n% Declare integer variables\nintcon =
1:" + g.nodes + ";\n\n");

                                        // declare matrix A, which depicts the left parts of all
                                        // inequality constraints
                                        fw.append("% Declare the left parts of the inequalities\nA
= [");

                                        // read re-opened lingeling's input file
                                        while (reopenedScanner.hasNextLine()) {
                                                String buffer = reopenedScanner.nextLine();
                                                // get expression type for each clauses
                                                int exprType =
MatlabHandler.getExpressionType(buffer);
                                                clauseCount++;

                                                // case of type 2 expression ((1-X1) + (1-X2) > =
1)

                                                if (exprType == 2) {
                                                        buffer = buffer.replace("-", "");
                                                        int index1 = Integer.parseInt(buffer.split("
")[0]);
                                                        int index2 = Integer.parseInt(buffer.split("
")[1]);
                                                        for (int p = 0; p < g.nodes; p++) {
                                                                if ((p == index1 - 1) || (p == index2
- 1))

                                                                        fw.append("1");
                                                                else
                                                                        fw.append("0");
                                                                if (p + 1 == g.nodes)
```

```
                                                fw.append(";\n");
                                        else
                                                fw.append(",");
                                }
                                // add result to an array, so that later we
can declare
                                // the right parts of all inequality
constraints (matrix
                                // b)
                                rightPartsInequalities[clauseCount] = 1;
                        }

                        // case of type 3 expression (X1 + X2 + X3 + ... >
= 1)
                        else if (exprType == 3) {
                                int indexes[] = new int[g.nodes + 1], p = 0;
                                for (p = 0; p < g.nodes + 1; p++)
                                        indexes[p] = 0;

                                String token[] = buffer.split(" ");
                                for (p = 0; p < token.length; p++)
                                        indexes[Integer.parseInt(token[p])]
= -1;
                                for (p = 1; p < g.nodes + 1; p++) {
                                        fw.append("" + indexes[p]);
                                        if (p + 1 == g.nodes + 1)
                                                fw.append(";\n");
                                        else
                                                fw.append(",");
                                }
                                // add result to an array, so that later we
can declare
```

```
                                        // the right parts of all inequality
constraints (matrix
                                        // b)
                                        rightPartsInequalities[clauseCount] = -1;
                                    }

                                    // case of type 1 expression (X1 = 1)
                                    else if (exprType == 1) {
                                        // found an equality constraints
                                        // handle it differently than the inequalities
(they are
                                        // declared in separate matrices than the
inequalities)

        equalityIndexes.add(Integer.parseInt(buffer.split(" ")[0]));
                                        equalityExists = true;
                                    }
                                }
                                fw.append("];\n\n");

                                // append declaration of right parts of all inequality
                                // constraints (matrix b) to script file
                                fw.append("% Declare the right parts of the
inequalities\nb = [");
                                for (clauseCount = 0; clauseCount <
rightPartsInequalities.length; clauseCount++)
                                    if (rightPartsInequalities[clauseCount] != 0)
                                        // declare matrix "b" using pre-collected
data

        fw.append(rightPartsInequalities[clauseCount] + ";");
```

```
fw.append("];\n\n% Declare the left and right parts of the
equalities\nAeq = [");

// if an equality constraint exists, we have to also declare
// matrices Aeq (left parts of equality constraints) and beq
// (right parts of the equality constraints)
if (equalityExists) {
	int t = g.nodes;
	for (int l = 0; l < equalityIndexes.size(); l++) {
		int temp = equalityIndexes.get(l) - 1;
		for (int pp = 0; pp < t; pp++) {
			if (temp == pp)
				fw.append("1");
			else
				fw.append("0");
			if ((pp + 1) != t)
				fw.append(",");
			else {
				fw.append(";");
			}
		}
		fw.append("\n");
	}
}
fw.append("];\n\nbeq = [");
if (equalityExists)
	for (int l = 0; l < equalityIndexes.size(); l++)
		fw.append("1;");
fw.append("];\n\n");

// append declaration of lower and upper bounds for all
// variables, to script file
```

```
                    fw.append("% Declare lower & upper bounds\nlb =
zeros(" + g.nodes + ",1);\nub = ones(" + g.nodes
                            + ",1);\n\n");
                    fw.append("tic;"); // begin execution time counting
                    // call solver to solve problem
                    fw.append("\n[x,fval] =intlinprog(f, intcon, A, b, Aeq,
beq, lb, ub);");
                    fw.append("\ntoc;"); // end execution time counting
                    fw.append("\n\nif(~isempty(x))\n\n\t% Extract Stable
Extension from results\n\t\tStableExtension=[");
                    // create spare variable to collect the Stable Extension
from
                    // the model
                    for (int d = 0; d < g.nodes; d++)
                        fw.append("1;");
                    fw.append("];\n\n\tfor index = 1:" + g.nodes +
"\n\tStableExtension(index)=x(index);\n\tend\nend");
                }

                /**
                 * Case of creating script file for the Stable Extension as a
                 * problem of Mixed-Integer Linear Programming (MILP =
problem with
                 * both integer-valued and real-valued variables)
                 */
                else {
                    // append declaration of objective function to file
                    fw.append("% Declare function\nf = [");
                    for (v = 0; v < g.nodes * 4; v++)
                        fw.append("0; ");
                    // append declaration of integer-valued variables to file
                    fw.append("1;];\n\n% Declare integer variables\nintcon =
" + (g.nodes * 2 + 1) + ":" + g.nodes * 4
```

```
                                        + ";\n");

                                // append declaration of matrix A (left parts of all
inequality
                                // constraints) to script file
                                fw.append("% Declare the left parts of the inequalities\nA
= [");

                                // read re-opened lingeling's input file
                                while (reopenedScanner.hasNextLine()) {
                                        String buffer = reopenedScanner.nextLine();
                                        // get expression type for each clause
                                        int exprType =
MatlabHandler.getExpressionType(buffer);
                                        clauseCount++;

                                        // case of type 2 expression ((1-X1) + (1-X2) > =
1)
                                        if (exprType == 2) {
                                                buffer = buffer.replace("-", "");
                                                int index1 = Integer.parseInt(buffer.split("
")[0]);
                                                int index2 = Integer.parseInt(buffer.split("
")[1]);
                                                for (int p = 0; p < g.nodes * 4; p++) {
                                                        if ((p == index1 - 1) || (p == index2
- 1))
                                                                fw.append("1");
                                                        else
                                                                fw.append("0");
                                                        if (p + 1 == g.nodes * 4)
                                                                fw.append(", 0;\n");
                                                        else
```

```
                                        fw.append(",");
                                }
                                rightPartsInequalities[clauseCount] = 1;
                        }
                        // case of type 3 expression (X1 + X2 + X3 + ... >
= 1)
                        else if (exprType == 3) {
                                int indexes[] = new int[g.nodes * 4 + 1], p
= 0;
                                for (p = 0; p < g.nodes * 4 + 1; p++)
                                        indexes[p] = 0;

                                String token[] = buffer.split(" ");
                                for (p = 0; p < token.length; p++)
                                        indexes[Integer.parseInt(token[p])]
= -1;
                                for (p = 1; p < g.nodes * 4 + 1; p++) {
                                        fw.append("" + indexes[p]);
                                        if (p + 1 == g.nodes * 4 + 1)
                                                // fw.append(", 0;\n");
                                                fw.append(", -1;\n");
                                        else
                                                fw.append(",");
                                }
                                rightPartsInequalities[clauseCount] = -1;
                        }
                        // case of type 1 expression (X1 = 1)
                        else if (exprType == 1) {

        equalityIndexes.add(Integer.parseInt(buffer.split(" ")[0]));
                                equalityExists = true;
                        }
                }
```

```
		// CREATE CODE FOR arguments, k, lambda1, lambda2
& epsilon
		/**
		 * ki : distance of argument ai from its specified bound on
		 * which it belongs (e.g. if ai = 0.8, ki = 0.2, as upper
bound
		 * is 1 and 1-0.2=0.8
		 *
		 * lambda1i: flag which determines if argument ai is near
		 * (belongs to) the lower bound
		 *
		 * lambda2i: flag which determines if argument ai is near
		 * (belongs to) the upper bound
		 *
		 * epsilon: maximum possible distance value, that the
variable
		 * ki can be assigned with. Valid for all arguments ai.
		 *
		 */
		// declare variables
		int cs = g.nodes * 4 + 2, nIntRPC = 0, p = 0, n = g.nodes;
		int lC1[] = new int[cs], lC2[] = new int[cs], lC3[] = new
int[cs];
		int lC4[] = new int[cs], lC5[] = new int[cs], lC6[] = new
int[cs];
		int lC7[] = new int[cs], lC8[] = new int[cs], lC9[] = new
int[cs];
		for (p = 0; p < g.nodes * 4 + 2; p++) { // initialize
			lC1[p] = lC2[p] = lC3[p] = lC4[p] = lC5[p] =
lC6[p] = lC7[p] = lC8[p] = lC9[p] = 0;
			nonIntRightParts[p] = 0.0;
		}
```

```
// CLAUSE 1 (for every i):
// 1-ai-ki-lambda1i <= 0 -> -ai-ki-lambda1i <= -1
for (p = 0; p < n; p++) {
        lC1[p] = lC1[p + n] = lC1[p + n * 2] = -1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC1[w]);
                if (w + 1 == n * 4)
                        fw.append(", 0;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC1[p] = lC1[p + n] = lC1[p + n * 2] = 0;
        nonIntRightParts[nIntRPC++] = -1.0;
}

// CLAUSE 2 (for every i):
// ai-ki-lambda2i <= 0 -> ai-ki-lambda2i <= 0
for (p = 0; p < n; p++) {
        lC2[p] = 1;
        lC2[p + n] = lC2[p + n * 3] = -1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC2[w]);
                if (w + 1 == n * 4)
                        fw.append(", 0;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC2[p] = lC2[p + n] = lC2[p + n * 3] = 0;
        nonIntRightParts[nIntRPC++] = 0.0;
}
```

```
// CLAUSE 3 (for every i):
// 0 <= ki <= e -> ki-e <= 0
for (p = 0; p < n; p++) {
        lC3[p + n] = 1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC3[w]);
                if (w + 1 == n * 4)
                        fw.append(", -1;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC3[p + n] = 0;
        nonIntRightParts[nIntRPC++] = 0;
}

// CLAUSE 4 (for every i):
// -ki <= 0
for (p = 0; p < n; p++) {
        lC4[p + n] = -1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC4[w]);
                if (w + 1 == n * 4)
                        fw.append(", 0;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC4[p + n] = 0;
        nonIntRightParts[nIntRPC++] = 0.0;
}
```

```
// CLAUSE 5 (for every i):
// lambda1i <=1
for (p = 0; p < n; p++) {
        lC5[p + n * 2] = 1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC5[w]);
                if (w + 1 == n * 4)
                        fw.append(", 0;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC5[p + n * 2] = 0;
        nonIntRightParts[nIntRPC++] = 1;
}

// CLAUSE 6 (for every i):
// -lambda1i <=0
for (p = 0; p < n; p++) {
        lC6[p + n * 2] = -1;
        for (int w = 0; w < n * 4; w++) {
                fw.append("" + lC6[w]);
                if (w + 1 == n * 4)
                        fw.append(", 0;\n");
                else
                        fw.append(",");
        }
        // re-initialize for next argument
        lC6[p + n * 2] = 0;
        nonIntRightParts[nIntRPC++] = 0;
}

// CLAUSE 7 (for every i):
```

```
// lambda2i <=1
for (p = 0; p < n; p++) {
	lC7[p + n * 3] = 1;
	for (int w = 0; w < n * 4; w++) {
		fw.append("" + lC7[w]);
		if (w + 1 == n * 4)
			fw.append(", 0;\n");
		else
			fw.append(",");
	}
	// re-initialize for next argument
	lC7[p + n * 3] = 0;
	nonIntRightParts[nIntRPC++] = 1;
}

// CLAUSE 8(for every i):
// -lambda2i <=0
for (p = 0; p < n; p++) {
	lC8[p + n * 3] = -1;
	for (int w = 0; w < n * 4; w++) {
		fw.append("" + lC8[w]);
		if (w + 1 == n * 4)
			fw.append(", 0;\n");
		else
			fw.append(",");
	}
	// re-initialize for next argument
	lC8[p + n * 3] = 0;
	nonIntRightParts[nIntRPC++] = 0;
}

// CLAUSE 9 (for every i):
// lambda1i + lamdba2i <= 1
```

```
for (p = 0; p < n; p++) {
	lC9[p + n * 2] = 1;
	lC9[p + n * 3] = 1;
	for (int w = 0; w < n * 4; w++) {
		fw.append("" + lC9[w]);
		if (w + 1 == n * 4)
			fw.append(", 0;\n");
		else
			fw.append(",");
	}
	// re-initialize for next argument
	lC9[p + n * 2] = lC9[p + n * 3] = 0;
	nonIntRightParts[nIntRPC++] = 1;
}
fw.append("];\n\n");

// append declaration of right parts of all inequality
// constraints (in matrix "b"), to script file
fw.append("% Declare the right parts of the
inequalities\nb = [");
for (clauseCount = 0; clauseCount <
rightPartsInequalities.length; clauseCount++)
	if (rightPartsInequalities[clauseCount] != 0)

fw.append(rightPartsInequalities[clauseCount] + ";");

for (clauseCount = 0; clauseCount <
nonIntRightParts.length - 1; clauseCount++)
	fw.append((int) (nonIntRightParts[clauseCount]) +
";");

fw.append("];\n\n% Declare the left and right parts of the
equalities\nAeq = [");
```

```
// if any equality constraints exists, matrices Aeq and beq
// (which depict the left and right parts of those
constraints,
// respectively) must be declared in the script file
if (equalityExists) {
	int t = n * 4;
	for (int l = 0; l < equalityIndexes.size(); l++) {
		int temp = equalityIndexes.get(l) - 1;
		for (int pp = 0; pp < t; pp++) {
			if (temp == pp)
				fw.append("1");
			else
				fw.append("0");
			if ((pp + 1) != t)
				fw.append(",");
			else {
				fw.append(", 0;");
			}
		}
		fw.append("\n");
	}
}
if (epsilon > -1.0) {
	int t = n * 4;
	for (int pp = 0; pp < t; pp++) {
		fw.append("0");
		if ((pp + 1) != t)
			fw.append(",");
		else {
			fw.append(", 1;");
			break;
		}
```

```
                }
                fw.append("\n");
            }
            // declare beq matrix
            fw.append("];\n\nbeq = [");
            if (equalityExists)
                for (int l = 0; l < equalityIndexes.size(); l++)
                    fw.append("1;");
            if (epsilon > -1.0) {
                fw.append(epsilon + ";");
            }
            fw.append("];\n\n");

            // declare upper and lower bounds for all variables
            fw.append("% Declare lower & upper bounds\nlb =
zeros(" + (n * 4 + 1) + ",1);\nub = ones(" + (n * 4 + 1)
                    + ",1);\n\n");
            fw.append("tic;"); // start execution time counting
            // call solver to solve problem
            fw.append("\n[x,fval] =intlinprog(f, intcon, A, b, Aeq,
beq, lb, ub);");
            fw.append("\ntoc;"); // end execution time counting
            // extract results for Stable Extension
            fw.append("\n%Extract Stable Extension from
results\nStableExtension=[");
            for (int d = 0; d < n; d++)
                fw.append("1;");
            fw.append("];\n\n");
            fw.append("if(exist('x', 'var') ==1 && ~isempty(x))\n");
            fw.append("\tfor index = 1:" + n +
"\n\tStableExtension(index)" + " = " + "x(index);\n\tend\nend\n");
        }
        fw.flush(); // flush file writer to write to file
```

```
                        fw.close(); // close file writer

                } catch (IOException e) { // case of not being able to create the Matlab
                                                                            // script file
                        System.out.println("ERROR: COULD NOT CREATE
MatlabInput.m FILE\nPROGRAM TERMINATED!");
                        System.exit(0);
                }
        }

        /**
         * Private, auxiliary, static method which returns the type of expression
         * given as string, using regular expressions.
         *
         * Used in the translation of lingeling's CNF clauses to Matlab's
         * mathematical equality/inequality clauses.
         *
         * @param expression
         *            string given to find its type
         * @return type of expression
         */
        private static int getExpressionType(String expression) {
                if (expression.matches("[0-9]+ 0"))
                        return 1; // case of 1st type of expression (X1 0)
                else if (expression.matches("\\-[0-9]+ \\-[0-9]+ 0"))
                        return 2; // case of 2nd type of expression (-X1 -X2 0)
                else
                        return 3; // case of 3rd type of expression (X1 attX1i attX1j ... 0)
        }
} // END OF FILE
```