ΤΕΙ ΜΕΣΟΛΟΓΓΙΟΥ, ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ & ΟΙΚΟΝΟΜΙΑΣ,
ΤΜΗΜΑ ΕΦΑΡΜΟΓΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΣΤΗ ΔΙΟΙΚΗΣΗ & ΣΤΗΝ ΟΙΚΟΝΟΜΙΑ

Πτυχιακή Εργασία

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΙΚΑΝΟΠΟΙΗΣΗΣ ΠΕΡΙΟΡΙΣΜΩΝ

ΝΤΟΥΣΚΑ ΑΛΕΞΑΝΔΡΑ
Α.Μ:9471

Μάιος 2007

Επιβλέπων καθηγητής: **Δρ. Δημήτριος Σωτηρόπουλος**

Περιεχόμενα

1

Εισαγωγή

1.1 Βασικά χαρακτηριστικά του προγραμματισμού περιορισμών

Αυτή η μελέτη αναφέρεται στον ***προγραμματισμό περιορισμών,*** μια εναλλακτική προσέγγιση στον προγραμματισμό που στηρίζεται σε έναν συνδυασμό τεχνικών που εξετάζουν ***το συλλογισμό*** και ***τον υπολογισμό.*** Έχει εφαρμοστεί επιτυχώς σε διάφορους τομείς συμπεριλαμβανομένης της μοριακής βιολογίας, της ηλεκτρικής εφαρμοσμένης μηχανικής, των διαδικασιών έρευνας και της αριθμητικής ανάλυσης. Η κεντρική έννοια είναι αυτή ενός περιορισμού. Ανεπίσημα, ένας ***περιορισμός*** σε μια ακολουθία μεταβλητών είναι ένας συσχετισμός στις περιοχές τους. Μπορεί να αντιμετωπισθεί ως απαίτηση που διασαφηνίζει ποιοι συνδυασμοί τιμών αναγνωρίζονται από τις περιοχές των μεταβλητών. Στη συνέχεια, ***ένα πρόβλημα ικανοποίησης περιορισμών*** αποτελείται από ένα πεπερασμένο σύνολο περιορισμών, κάθε ένα σε μία υπο-ακολουθία μιας δεδομένης ακολουθίας μεταβλητών.

Για να λύσουμε ένα δεδομένο πρόβλημα με τη βοήθεια του προγραμματισμού περιορισμών το διατυπώνουμε αρχικά ως πρόβλημα ικανοποίησης περιορισμών. Για αυτόν τον σκοπό

- εισάγουμε μερικές μεταβλητές που κυμαίνονται πέρα από τις συγκεκριμένες περιοχές και περιορισμούς οι οποίοι κυμαίνονται πέρα από αυτές τις μεταβλητές.
- επιλέγουμε κάποια γλώσσα στην οποία οι περιορισμοί εκφράζονται (συνήθως με ένα μικρό υποσύνολο της first-order λογικής).

Αυτό το μέρος της επίλυσης προβλήματος καλείται ***διαμόρφωση.*** Γενικά, υπάρχουν περισσότερες από μια αντιπροσωπεύσεις ενός προβλήματος ως πρόβλημα ικανοποίησης περιορισμών. Κατόπιν για να λύσουμε την επιλεγμένη αντιπροσώπευση χρησιμοποιούμε

- μεθόδους συγκεκριμένων περιοχών,

ή

- γενικές μεθόδους,

ή έναν συνδυασμό και των δύο.

Οι ***μέθοδοι συγκεκριμένων περιοχών*** παρέχονται συνήθως υπό μορφή εφαρμογών ειδικού σκοπού αλγόριθμων. Χαρακτηριστικά παραδείγματα είναι τα εξής:

- ένα πρόγραμμα που λύνει τα συστήματα των γραμμικών εξισώσεων,
- ένα πλήρες πρόγραμμα για το γραμμικό προγραμματισμό,
- μια εφαρμογή του αλγόριθμου ενοποίησης, ένας ακρογωνιαίος λίθος της αυτοματοποιημένης παρουσίασης αποδείξεων του θεωρήματος.

Στη συνέχεια, οι ***γενικές μέθοδοι*** αφορούν τους τρόπους μείωσης του διαστήματος αναζήτησης και τις συγκεκριμένες ***μεθόδους αναζήτησης.*** Οι αλγόριθμοι που εξετάζουν τη μείωση του διαστήματος αναζήτησης καλούνται συνήθως ***αλγόριθμοι διάδοσης περιορισμών***, αν και πολλές φορές έχουν χρησιμοποιηθεί διάφορα άλλα ονόματα. Αυτοί οι αλγόριθμοι διατηρούν την ισοδυναμία απλοποιώντας το εξεταζόμενο πρόβλημα. Επιτυγχάνουν τις διάφορες μορφές ***τοπικής σταθερότητας*** που προσπαθούν να προσεγγίσουν την έννοια (της γενικής) σταθερότητας. Οι (top down) μέθοδοι αναζήτησης συνδυάζουν τις διάφορες μορφές διάδοσης περιορισμών με τη συνηθισμένη αντίστροφη διαδρομή καθώς επίσης με την διακλάδωση και τη συνδεδεμένη αναζήτηση.

Ο ορισμός του προγραμματισμού περιορισμών είναι τόσο γενικός που ενσωματώνει τόσο διαφορετικές περιοχές όπως η Γραμμική Άλγεβρα, η Γενική Βελτιστοποίηση, ο Γραμμικός Προγραμματισμός καθώς και ο Προγραμματισμός Ακέραιων αριθμών, κ.λ.π. Επομένως πρέπει να τονίσουμε ένα ουσιαστικό σημείο. Εάν οι μέθοδοι συγκεκριμένων περιοχών είναι διαθέσιμες πρέπει να εφαρμοστούν στη θέση των γενικών μεθόδων. Παραδείγματος χάριν, κατά εξέταση συστημάτων γραμμικών εξισώσεων, οι γνωστοί αλγόριθμοι γραμμικής άλγεβρας είναι εύκολα διαθέσιμοι και δεν έχει νόημα να εφαρμόσουμε τις γενικές μεθόδους σε αυτές τις εξισώσεις.

Στην πραγματικότητα, ένας από τους στόχους του προγραμματισμού περιορισμών είναι να αναζητηθούν αποδοτικές μέθοδοι συγκεκριμένων περιοχών που να μπορούν να χρησιμοποιηθούν αντί των γενικών μεθόδων και να ενσωματωθούν με έναν ανεξάρτητο τρόπο σε ένα γενικό πλαίσιο. Ένα τέτοιο πλαίσιο υποστηρίζει συνήθως

- μεθόδους συγκεκριμένων περιοχών με τη βοήθεια εξειδικευμένων πλήρη προγραμμάτων, οι οποίοι αποκαλούνται πολλές φορές ***επιλυτές περιορισμών,***
- γενικές μεθόδους με τη βοήθεια διάφορων ενσωματωμένων μεθόδων οι οποίες συγκεκριμένα εξασφαλίζουν ή διευκολύνουν τη χρήση κατάλληλων αλγόριθμων διάδοσης περιορισμών και υποστηρίζουν διάφορες μεθόδους αναζήτησης.

Μόλις παρουσιάσουμε ένα πρόβλημα ως πρόβλημα ικανοποίησης περιορισμών πρέπει να το λύσουμε. Στην πράξη ενδιαφερόμαστε:

- να ορίσουμε εάν η επιλεγμένη παρουσίαση έχει μια λύση (είναι σταθερή),
- για την εύρεση μιας λύσης, αντίστοιχα, όλων των λύσεων,
- για εύρεση μιας βέλτιστης λύσης, αντίστοιχα, όλων των βέλτιστων λύσεων w.r.t. κάποιο ποιοτικό μέτρο.

Μετά από αυτήν την σύντομη παρουσίαση μπορούμε να διατυπώσουμε τα ακόλουθα βασικά χαρακτηριστικά του προγραμματισμού περιορισμών:

Προσέγγιση δύο φάσεων: Η διαδικασία προγραμματισμού αποτελείται από δύο φάσεις: την δημιουργία της παρουσίασης ενός προβλήματος με τη βοήθεια των περιορισμών και την λύση του. Στην πράξη, και οι δύο φάσεις αποτελούνται από διάφορες μικρότερες βαθμίδες στις οποίες μπορούμε να παρεμβάλουμε λευκές σελίδες.

Ευελιξία: Η παρουσίαση ενός προβλήματος με τη βοήθεια περιορισμών είναι πολύ ευέλικτη επειδή οι περιορισμοί μπορούν να προστεθούν, να αφαιρεθούν ή να τροποποιηθούν. Αυτή η ευελιξία κληρονομείται με τον προγραμματισμό περιορισμών.

Παρουσία των Built-ins(ενσωματωμένων μεθόδων): Για να υποστηρίξουμε αυτήν την προσέγγιση στον προγραμματισμό είναι διαθέσιμες διάφορες ενσωματωμένες μέθοδοι. Εξετάζουν συγκεκριμένους επιλυτές περιορισμών, αλγόριθμους διάδοσης περιορισμών και μεθόδους αναζήτησης.

Μια πρόσθετη πτυχή που παρουσιάζεται μέσα από τον προγραμματισμό περιορισμών είναι ότι η διαμόρφωση με τη βοήθεια περιορισμών οδηγεί στην παρουσίαση ενός προβλήματος με τη βοήθεια συσχετισμών. Αυτό φέρει κάποια ομοιότητα στα συστήματα βάσεων δεδομένων, παραδείγματος χάριν στις σχεσιακές βάσεις δεδομένων. Στην πραγματικότητα, οι περιορισμοί μελετώνται επίσης στα πλαίσια των συστημάτων βάσεων δεδομένων. Είναι χρήσιμοι σε καταστάσεις όπου κάποιες πληροφορίες, για παράδειγμα ο ορισμός μιας περιοχής ενός χάρτη, πρέπει να δοθούν άρρητα, με τη βοήθεια περιορισμών στους πραγματικούς αριθμούς .
Η διαφορά είναι ότι στα πλαίσια των συστημάτων βάσεων δεδομένων ο στόχος αποτελείται από την άντληση πληροφοριών από τις εξεταζόμενες σχέσεις αποτελεσματικά, ανεξάρτητα εάν ορίζονται ρητά (παραδείγματος χάριν με τη βοήθεια πινάκων) ή με ασάφεια (παραδείγματος χάριν με τη βοήθεια επιστροφών ή ανισοτήτων). Αντίθετα, στον προγραμματισμό περιορισμών οι εξεταζόμενες σχέσεις ορίζονται συνήθως άρρητα και ο στόχος αποτελείται από την επίλυση τους ή τον ορισμό ότι δεν υπάρχει καμία λύση. Αυτό οδηγεί σε διαφορετικές μεθόδους και τεχνικές.

1.2 Εφαρμογές του προγραμματισμού περιορισμών

Τα προβλήματα που μπορούν να λυθούν καλύτερα με τη βοήθεια του προγραμματισμού περιορισμών είναι συνήθως εκείνα που μπορούν να διατυπωθούν απλά από την άποψη των απαιτήσεων, των γενικών ιδιοτήτων, ή νόμων, και για τα οποία οι μέθοδοι συγκεκριμένων περιοχών οδηγούν σε υπερβολικά σύνθετες διαμορφώσεις. Ο προγραμματισμός περιορισμών έχει ήδη εφαρμοστεί επιτυχώς σε πολυάριθμες περιοχές συμπεριλαμβάνοντας:

- διαλογικά γραφικά συστήματα (για να εκφράσει τη γεωμετρική συνοχή στην περίπτωση της ανάλυσης γεγονότων),
- προβλήματα διαδικασιών έρευνας (διάφορα προβλήματα βελτιστοποίησης, ειδικότερα προβλήματα που προσδιορίζονται σε ορισμένο χρόνο),
- μοριακή βιολογία (αλληλουχία DNA, κατάρτιση των τρισδιάστατων προτύπων των πρωτεϊνών),
- επαγγελματικές εφαρμογές (επιλογή εμπορικής συναλλαγής),
- ηλεκτρική εφαρμοσμένη μηχανική (θέση των σφαλμάτων στα κυκλώματα, υπολογίζοντας τη διάταξη κυκλωμάτων, δοκιμάζοντας και επαληθεύοντας τη διάταξη),
- αριθμητικό υπολογισμό (που λύνει τους πολυωνυμικούς περιορισμούς με εγγυημένη ακρίβεια),
- επεξεργασία φυσικής γλώσσας (κατασκευή αποδοτικών κατατμητών),
- άλγεβρα υπολογιστών (επίλυση ή/και απλούστευση των εξισώσεων πέρα από τις διάφορες αλγεβρικές δομές).

Οι πιο πρόσφατες εφαρμογές περιορισμών περιλαμβάνουν την δημιουργία συνδεδεμένων μουσικών ράδιο-προγραμμάτων, εφαρμογές μηχανολογίας λογισμικού (αποκατάσταση σχεδίου και βελτιστοποίηση κώδικα), και επιλογή και σχεδιασμό παρατηρήσεων που εκτελούνται από τους δορυφόρους. Επίσης, ο προγραμματισμός περιορισμών επαλήθευσε μια εφαρμόσιμη προσέγγιση για να αντιμετωπίσει ορισμένα δυσεπίλυτα υπολογιστικά προβλήματα.

Η αναπτυσσομένη σπουδαιότητα αυτής της περιοχής μπορεί να επιβεβαιωθεί από το γεγονός ότι τώρα υπάρχουν ετήσια συνέδρια και εργαστήρια πάνω στον προγραμματισμό περιορισμών και οι εφαρμογές του προσελκύουν σταθερά περισσότερους από εκατό (περιστασιακά διακόσιους) συμμετέχοντες. Περαιτέρω, το 1996 προωθήθηκε ένα (δυστυχώς ακριβό) περιοδικό το οποίο ονομάζεται "Περιορισμοί". Επίσης, έχουν εμφανιστεί διάφορες ειδικές εκδόσεις περιοδικών πληροφορικής που αφιερώνονται στο θέμα των περιορισμών. Αλλά αυτός ο τομέας είναι ακόμα νέος και μέχρι στιγμής έχουν εμφανιστεί μόνο μερικά βιβλία πάνω σε αυτό το θέμα. Αυτό μας οδήγησε στο γράψιμο αυτού του βιβλίου.

1.3 Μια πολύ σύντομη περιγραφή του θέματος

Προτού να ασχοληθούμε με τον προγραμματισμό περιορισμών στην παρουσίασή μας, ας συνοψίσουμε εν συντομία την ιστορία αυτού του θέματος. Θα μας επιτρέψει να καταλάβουμε καλύτερα την κατεύθυνση που ακολουθεί ο συγκεκριμένος τομέας.

Η έννοια του περιορισμού χρησιμοποιήθηκε ήδη το 1963 σε μια πρόωρη εργασία της I.Sutherland για ένα διαλογικό σύστημα SKETCHPAD σχεδίων. Στη δεκαετία του '70 προτάθηκαν διάφορες πειραματικές γλώσσες οι οποίες χρησιμοποίησαν την έννοια των περιορισμών και στηρίχθηκαν στην έννοια της επίλυσης περιορισμών.

Η έννοια ενός προβλήματος ικανοποίησης περιορισμών διατυπώθηκε επίσης στη δεκαετία του '70 από ερευνητές στην τεχνητή νοημοσύνη (AI). Προσδιόρισαν επίσης τις κύριες έννοιες της τοπικής σταθερότητας και των αλγόριθμων που μας επιτρέπουν να τους επιτύχουμε. Ανεξάρτητα, ορίστηκαν διάφορες μέθοδοι αναζήτησης. Μερικές από αυτές, όπως η αντίστροφη-διαδρομή μπορούν να εξακριβωθούν πίσω στο δέκατο ένατο αιώνα, ενώ άλλες, όπως η διακλάδωση και η συνδεδεμένη αναζήτηση, προσδιορίστηκαν στα πλαίσια της συνδυαστικής βελτιστοποίησης. Η συμβολή του προγραμματισμού περιορισμών ήταν να προσδιοριστούν διάφορες νέες μορφές αναζήτησης που συνδυάζουν τις γνωστές τεχνικές με τους διάφορους αλγόριθμους διάδοσης περιορισμών. Μελετήθηκαν ήδη μερικοί συγκεκριμένοι συνδυασμοί στον τομέα της συνδυαστικής βελτιστοποίησης.

Στη δεκαετία του '80 προτάθηκαν και εφαρμόστηκαν οι πρώτες γλώσσες προγραμματισμού περιορισμών οι οποίες ήταν μεγάλης σπουδαιότητας. Οι σημαντικότερες ήταν οι γλώσσες που βασίζονταν στο παράδειγμα λογικού προγραμματισμού. Αυτό οδήγησε στην ανάπτυξη ***του προγραμματισμού λογικών περιορισμών,*** μια επέκταση του λογικού προγραμματισμού υπό την έννοια των περιορισμών. Ο προγραμματισμός οδήγησε έκτακτα στον προσδιορισμό του ***περιορισμού αποθέματος*** ως κεντρική έννοια. Η διάδοση περιορισμών και οι διάφορες μορφές αναζήτησης είναι συνήθως διαθέσιμες σε αυτές τις γλώσσες υπό μορφή ενσωματωμένων μεθόδων(built-ins).

Προς το τέλος της δεκαετίας του '80 και του '90 πραγματοποιήθηκε μια μορφή σύνθεσης μεταξύ αυτών των δύο εξελίξεων. Οι ερευνητές βρήκαν διάφορες νέες εφαρμογές του προγραμματισμού περιορισμών, ειδικότερα στους τομείς των διαδικασιών έρευνας και της αριθμητικής ανάλυσης. Η πρόοδος πολλές φορές επιτεύχθηκε προσδιορίζοντας νέους σημαντικούς τύπους περιορισμών και νέους αλγόριθμους διάδοσης περιορισμών. Κάποιος επίσης συνειδητοποίησε ότι η περαιτέρω πρόοδος μπορεί να εξαρτηθεί από έναν συνδυασμό τεχνικών από το AI, τις διαδικασίες έρευνας, την άλγεβρα υπολογιστών και τη μαθηματική λογική. Αυτό μετέτρεψε τον προγραμματισμό περιορισμών σε μία ενδιαφέρουσα μη φυσιολογική περιοχή, στην οποία η θεωρητική εργασία οδηγείται συχνά από τις εφαρμογές και στη συνέχεια οι εφαρμογές οδηγούν σε νέες προκλήσεις σχετικά με τις εφαρμογές του προγραμματισμού περιορισμών.

1.4 Η προσέγγισή μας

Κατά την παρουσίασή των βασικών εννοιών και των τεχνικών του προγραμματισμού περιορισμών προσπαθούμε για μια βελτιωμένη παρουσίαση στην οποία διευκρινίζουμε τη φύση αυτών των τεχνικών και την αλληλεξάρτησή τους. Για αυτόν τον σκοπό οργανώσαμε την παρουσίαση γύρω από διάφορες απλές αρχές.

Αρχή 1: Ο προγραμματισμός περιορισμών σχετίζεται με τη διατύπωση του προβλήματος ως πρόβλημα ικανοποίησης περιορισμών και με την επίλυση του με τη βοήθεια συγκεκριμένων περιοχών ή γενικών μεθόδων.

Αυτό εξηγεί την εστίασή μας στα προβλήματα ικανοποίησης περιορισμών και στους επιλυτές περιορισμών.

Αρχή 2: Πολλοί επιλυτές περιορισμών μπορούν να εξηγηθούν απλά χρησιμοποιώντας ένα πλαίσιο βασισμένο σε κανόνες. Ο επιλυτής περιορισμών αποτελείται έπειτα από ένα σύνολο κανόνων που ορίζουν τη συμπεριφορά του κι έναν χρονοπρογραμματιστή. Αυτή η άποψη τονίζει τις συνδέσεις μεταξύ του προγραμματισμού που βασίζεται σε κανόνες και του προγραμματισμού περιορισμών.

Αυτό εξηγεί την απόφασή μας να ορίσουμε τους επιλυτές περιορισμών με τη βοήθεια των κανόνων απόδειξης που μετασχηματίζουν τα προβλήματα ικανοποίησης περιορισμών.

Αρχή 3: Οι αλγόριθμοι διάδοσης περιορισμών μπορούν να εξηγηθούν απλά ως περιπτώσεις απλών γενικών αλγόριθμων επανάληψης.

Αυτή η άποψη μας επιτρέπει να διευκρινίσουμε τη φύση των αλγόριθμων διάδοσης περιορισμών. Επίσης, μας παρέχει μια φυσική μέθοδο εφαρμογής των επιλυτών περιορισμών που αναφέραμε, δεδομένου ότι ένας κανόνας χρονοπρογραμματισμού είναι απλώς μια άλλη περίπτωση ενός γενικού αλγόριθμου επανάληψης.

Αρχή 4: Οι (Top down) τεχνικές αναζήτησης μπορούν να αντιμετωπισθούν εννοιολογικά ως αλγόριθμοι διαχωρισμού των δέντρων αναζήτησης.

Αυτό εξηγεί γιατί οργανώσαμε το κεφάλαιο σχετικά με την αναζήτηση με βάση το σλόγκαν: Αλγόριθμος Αναζήτησης = Δέντρο Αναζήτησης + Αλγόριθμος Διαχωρισμού,

και γιατί ερμηνεύσαμε τους αλγόριθμους που προέκυψαν υπό μορφή διαδοχικών αναδιατυπώσεων.

1.5 Οργάνωση του βιβλίου

Οι παραπάνω αρχές που εξηγήσαμε οδήγησαν σε μια απλή οργάνωση του υλικού. Κάνουμε μια σύντομη περιγραφή των υπόλοιπων κεφαλαίων. Στο **Κεφάλαιο 2** αναφέρουμε διάφορα παραδείγματα των προβλημάτων ικανοποίησης περιορισμών. Τονίζουμε ότι σε πολλά προβλήματα είναι πιθανές διάφορες φυσικές αντιπροσωπεύσεις. Στο **Κεφάλαιο 3** εισάγουμε ένα γενικό πλαίσιο που μας επιτρέπει να εξηγήσουμε τα βασικά του προγραμματισμού περιορισμών. Προσδιορίζουμε τα απλά συστατικά αυτού του πλαισίου. Αυτό κάνει ευκολότερο να καταλάβουμε το θέμα των επόμενων κεφαλαίων.

Κατόπιν, στο **Κεφάλαιο** 4, παρέχουμε τρία γνωστά παραδείγματα πλήρη επιλυτών περιορισμού. Ασχολούνται, αντίστοιχα, με την επίλυση των εξισώσεων πέρα από τους όρους, τις γραμμικές εξισώσεις πέρα από τους πραγματικούς αριθμούς και τις γραμμικές ανισότητες πέρα από τους πραγματικούς. Τέλος, στο **Κεφάλαιο 5,** αναφέρουμε τους διάφορους (top down) αλγόριθμους αναζήτησης. Τους παρουσιάζουμε κατά τέτοιο τρόπο ώστε να μπορεί κάποιος να δει πώς αυτοί οι αλγόριθμοι συνδέονται ο ένας με τον άλλον.

2

Προβλήματα ικανοποίησης περιορισμών: παραδείγματα

Ο στόχος αυτού του κεφαλαίου είναι να συζητηθούν τα διάφορα παραδείγματα των προβλημάτων ικανοποίησης περιορισμών (CSPs [2] εν ολίγης). Η έννοια ενός CSP είναι πολύ γενική, έτσι δεν μας εκπλήσσει το γεγονός ότι αυτά τα παραδείγματα καλύπτουν ένα ευρύ φάσμα των θεμάτων. Περιοριζόμαστε εδώ στα παραδείγματα των CSPs που είναι απλά να εξηγηθούν και που επεξηγούν τη χρήση των γενικών μεθόδων προγραμματισμού περιορισμών. Ειδικότερα, περιελάβαμε εδώ μερικούς αιώνιους γρίφους, αφού, όπως έχει αναγνωριστεί για κάποιο χρονικό διάστημα, διαμορφώνανε ένα άριστο μέσο για να εξηγήσουν ορισμένες αρχές του προγραμματισμού περιορισμών.

Όπως αναφέρεται ήδη στο Κεφάλαιο 2.1 η αντιπροσώπευση ενός προβλήματος ως CSP καλείται συνήθως **διαμόρφωση**. Τα επιλεγμένα παραδείγματα διευκρινίζουν διάφορες πτυχές της διαμόρφωσης. Κατ' αρχήν, όπως θα δούμε, μερικά από τα προβλήματα μπορούν να τυποποιηθούν ως CSP με έναν απλό τρόπο. Για άλλα προβλήματα η κατάλληλη αντιπροσώπευση ως CSP δεν είναι με κανένα τρόπο απλή και στηρίζεται σε μια μη μηδενική θεωρία "υποβάθρου" που εξασφαλίζει την ακρίβεια της υιοθετημένης αντιπροσώπευσης. Επίσης για διάφορα προβλήματα, υπάρχουν περισσότερες από μια φυσικές αντιπροσωπεύσεις.

Κατά την παρουσίαση των CSPs είναι χρήσιμο να τα ταξινομήσουμε σύμφωνα με κάποιο κριτήριο. Γενικά, οι τεχνικές που χρησιμοποιούνται για να λύσουμε τα CSPs εξαρτώνται από τις περιοχές πέρα από τις οποίες ορίζονται και από τη σύνταξη των χρησιμοποιούμενων περιορισμών. Στα περισσότερα παραδείγματα χρησιμοποιούμε κάποια απλή γλώσσα για να ορίσουμε τους περιορισμούς. Αργότερα, στο Κεφάλαιο 4, θα γίνουμε πιο ακριβείς και θα συζητήσουμε λεπτομερώς τις συγκεκριμένες γλώσσες στις οποίες θα οριστούν οι περιορισμοί. Αλλά τώρα είναι πάρα πολύ νωρίς για να εκτιμηθεί ο ρόλος της σύνταξης. Έτσι κατά κάποιο τρόπο ταξινομούμε τα CSPs σύμφωνα με τις περιοχές πέρα από τις οποίες ορίζονται. Αυτό εξηγεί τη δομή αυτού του κεφαλαίου.

Κατ' αρχήν, τυποποιούμε στην παράγραφο 2.1 την έννοια ενός περιορισμού και ενός CSP. Κατόπιν, στην παράγραφο 2.2 εισάγουμε μερικά γνωστά προβλήματα και γρίφους που μπορούν να τυποποιηθούν φυσικά ως CSPs με τις περιοχές ακέραιων αριθμών. Στην παράγραφο 2.3 εξετάζουμε τα παραδείγματα των προβλημάτων η διαμόρφωση των οποίων οδηγεί σε CSPs με μεταβλητές που κυμαίνονται πέρα από τους πραγματικούς. Στη συνέχεια, στην παράγραφο 2.4 εξετάζουμε **τα CSPs ως προς τις Λογικές τιμές.** Αυτά είναι τα CSPs στα οποία οι μεταβλητές κυμαίνονται πέρα από την περιοχή των ακέραιων αριθμών [0...1] ή, ισοδύναμα, **{ψευδές, αληθές}** και στα οποία οι περιορισμοί εκφράζονται με τη βοήθεια των λογικών εκφράσεων.

Μια σημαντική κατηγορία CSPs είναι αυτή στην οποία οι μεταβλητές κυμαίνονται πέρα από τις μη αριθμητικές περιοχές. Τα ονομάζουμε **συμβολικά CSPs.** Εξετάζονται στην παράγραφο 2.5. Σε περίπτωση που ενδιαφερόμαστε για την εύρεση μιας βέλτιστης λύσης σε ένα CSP συνδέουμε με κάθε λύση μια αντικειμενική λειτουργία που θέλουμε να ελαχιστοποιήσουμε ή να μεγιστοποιήσουμε. Αυτό οδηγεί σε μια τροποποίηση ενός CSP το οποίο ονομάζουμε **περιορισμένο πρόβλημα βελτιστοποίησης.** Εξετάζονται στην παράγραφο 2.6.

2.1 Βασικές έννοιες

Τα προβλήματα ικανοποίησης περιορισμών, ή CSPs, είναι μια θεμελιώδης έννοια στον προγραμματισμό περιορισμών. Για να προχωρήσουμε πρέπει να τους ορίσουμε τυπικά. Ο ακριβής ορισμός είναι απολύτως απλός. Πρώτα εισάγουμε την έννοια ενός περιορισμού.

Εξετάστε μια πεπερασμένη ακολουθία μεταβλητών Y := $y_1,\ldots,y_k$ όπου k > 0, με τις αντίστοιχες περιοχές $D_1,\ldots,D_k$ συνδεδεμένες μεταξύ τους. Έτσι κάθε μεταβλητή y_i κυμαίνεται πέρα από την περιοχή D_i. Από έναν **περιορισμό** C προς Y εννοούμε ένα υποσύνολο D_1x... x D_2. Όταν k = 1 λέμε ότι ο περιορισμός είναι **μοναδιαίος** και όταν k = 2, ο περιορισμός είναι **δυαδικός.** Από ένα **πρόβλημα ικανοποίησης περιορισμών**, ή ένα **CSP,** υπολογίζουμε μια πεπερασμένη ακολουθία μεταβλητών X := $x_1,\ldots,x_n$ με τις αντίστοιχες περιοχές $D_1,\ldots, D_n$, μαζί με ένα πεπερασμένο σύνολο περιορισμών C, καθένα σε μια υπο-ακολουθία του X. Γράφουμε ένα τέτοιο CSP ως [C;DE], όπου DE := x_1 E D_1,...x_n E D_n και καλούμε κάθε κατασκεύασμα της μορφής x E D **έκφραση περιοχής**. Για να απλοποιήσουμε τη σημείωση παραλείπουμε τις "{ }" αγκύλες κατά την παρουσίαση των συγκεκριμένων συνόλων περιορισμών C.

Ορίζουμε τώρα την κρίσιμη έννοια μιας λύσης σε ένα CSP. Διαισθητικά, μια λύση σε ένα CSP είναι μια ακολουθία νόμιμων τιμών για όλες τις μεταβλητές του έτσι ώστε όλοι οι περιορισμοί του να είναι ικανοποιημένοι. Ακριβέστερα, εξετάστε ένα CSP ⟨ C;DE ⟩με DE := x_1 E $D_1,\ldots,x_n$ E D_n. Λέμε ότι ένα n-tuple (d_1,...,dn) E D_1 x...x D_n ικανοποιεί έναν περιορισμό C E C στις μεταβλητές $x_{i1},\ldots, x_{im}$ εάν

$$(d_{i1},\ldots,d_{im}) \text{ E } C.$$

Κατόπιν λέμε ότι ένα n-tuple (d_1,...,dn) E D_1 x...x D_n είναι μια **λύση** [C;DE] εάν ικανοποιεί κάθε περιορισμό C E C. Εάν ένα CSP έχει μια λύση, λέμε ότι είναι **σταθερό** , ειδάλλως λέμε ότι είναι **μεταβλητό**.

Σημειώστε ότι στον ορισμό ενός περιορισμού και ενός CSP καμία σύνταξη δεν θεωρήθηκε δεδομένη. Στην πράξη, φυσικά, κάποιος πρέπει να ορίσει τους περιορισμούς και τις εκφράσεις περιοχών. Στη συνέχεια υποθέτουμε ότι ορίζονται σε κάποια συγκεκριμένη, περαιτέρω απροσδιόριστη, γλώσσα. Σε αυτήν την αντιπροσώπευση είναι αυτονόητο ότι κάθε περιορισμός είναι ένα υποσύνολο του καρτεσιανού προϊόντος των σχετικών περιοχών των μεταβλητών. Παραδείγματος χάριν, εάν εξετάζουμε το CSP $\langle x<y; xE[0..10], yE[5..10]\rangle$, κατόπιν θεωρούμε τον περιορισμό x < y ως σύνολο όλων των ζευγαριών (a,b) με ένα a E [0..10] και το b E [5..10] έτσι ώστε a < b.

Ας επεξηγήσουμε αυτές τις έννοιες με ένα απλό παράδειγμα. Θεωρείστε την ακολουθία τεσσάρων μεταβλητών x, y, z, u που κυμαίνεται πέρα από τους φυσικούς αριθμούς και τους ακόλουθους τρεις περιορισμούς τους: $x^3 + y^3 + z^3 + u^3 = 100$, $x < u$, $x + y = z$. Σύμφωνα με την παραπάνω σημείωση γράφουμε αυτό το CSP ως εξης:⟨

$$\langle\ x^3 + y^3 + z^3 + u^3 = 100, x < u,\ x + y = z\ ; x \in N,\ \ \rangle y \in N,\ z \in N, u \in N\ \rangle,$$

όπου N δείχνει το σύνολο των φυσικών αριθμών.

Κατόπιν η ακολουθία (1, 2, 3, 4) είναι μια λύση σε αυτό το CSP δεδομένου ότι αυτή η ακολουθία ικανοποιεί όλους τους περιορισμούς. Πράγματι, έχουμε το $1^3 + 2^3 + 3^3 + 4^3 = 100$, $1 < 4$ και $1 + 2 = 3$.

Τέλος, ας διευκρινίσουμε ένα απλό θέμα. Κατά τον ορισμό των περιορισμών και των CSPs αναφερόμαστε στις ακολουθίες (υπό- ακολουθίες αντίστοιχα) μεταβλητών και όχι στα σύνολα (αντίστοιχα υποσύνολα) μεταβλητών. Δηλαδή, λαμβάνοντας υπόψη ένα CSP κάθε ένας από τους περιορισμούς του ορίζεται σε μια υπό- ακολουθία και όχι σε ένα υποσύνολο των μεταβλητών του. Ειδικότερα, ο παραπάνω περιορισμός $x < y$ ορίζεται με την υπό – ακολουθία x, y της ακολουθίας x, y, z, u.

Επίσης, η ακολουθία z,y δεν είναι μια υπό –ακολουθία x, y, z, u, έτσι εάν προσθέσουμε στο παραπάνω CSP τον περιορισμό $z = y + 2$ δεν μπορούμε να το θεωρήσουμε ως περιορισμό στο z, y. Αλλά μπορούμε να το δούμε φυσικά ως περιορισμό στο y,z και, εάν επιθυμούμε, μπορούμε να τον ξαναγράψουμε ως $y + 2 = z$. Η εμπιστοσύνη στις ακολουθίες και τις υπό – ακολουθίες των μεταβλητών αντί των συνόλων και των υποσύνολων θα μας επιτρέψει να αναλύσουμε με έναν απλό τρόπο τις orderings μεταβλητές όταν ερευνούμε για λύσεις σε ένα δεδομένο CSP. Έτσι αυτή η παρουσίαση δεν εισάγει οποιουσδήποτε περιορισμούς και απλοποιεί μερικές εκτιμήσεις.

2.2 Προβλήματα ικανοποίησης περιορισμών στους ακέραιους αριθμούς

Το υπόλοιπο αυτού του κεφαλαίου αφιερώνεται σε μια παρουσίαση των διάφορων παραδειγμάτων των προβλημάτων ικανοποίησης περιορισμών. Αρχίζουμε με τα παραδείγματα CSPs με τις περιοχές ακέραιων αριθμών.

Παράδειγμα 2.1 SEND + MORE = MONEY.

Αυτό είναι ένα κλασικό παράδειγμα ενός αποκαλούμενου **κρυπτοαριθμητικού προβλήματος.** Αυτοί είναι μαθηματικοί γρίφοι στους οποίους τα ψηφία αντικαθίστανται με γράμματα της αλφαβήτου ή άλλα σύμβολα. Τα προβλήματα που εξετάζουν τα έγκυρα ποσά καλούνται **αλφαριθμητικά προβλήματα**. Στο υπό εξέταση πρόβλημα καλούμαστε να αντικαταστήσουμε κάθε γράμμα από ένα διαφορετικό ψηφίο έτσι ώστε το παραπάνω άθροισμα, το οποίο είναι

$$\begin{array}{r} SEND \\ +\ \underline{MORE} \\ MONEY \end{array}$$

να είναι σωστό. Εδώ οι μεταβλητές είναι οι S, E, N, D, M, O, R, Y. Επειδή το S και το M είναι τα κύρια ψηφία, η περιοχή για κάθε ένα από αυτά αποτελείται από το διάστημα ακέραιων αριθμών [1..9]. Η περιοχή κάθε μιας από τις υπόλοιπες μεταβλητές αποτελείται από το διάστημα ακέραιων αριθμών [0..9]. Αυτό το πρόβλημα μπορεί να διατυπωθεί ως περιορισμός ισότητας

$$\begin{aligned} & 1000.S + 100.E + 10.N + D \\ & + 1000.M + 100.O + 10.R + E \\ = {} & 10000 \bullet M + 1000.O + 100.N + 10.E + Y \end{aligned}$$

συνδυασμένος με 28 περιορισμούς ανισότητας $x \neq y$ για x, y να κυμαίνονται πέρα από το σύνολο {S, E, N, D, M, O, R, Y} με το x να προηγείται του y, για παράδειγμα, στη σειρά που παρουσιάζεται.

Ισχύει μια δευτερεύουσα παραλλαγή στην παραπάνω αντιπροσώπευση αυτού του προβλήματος ως ένα CSP θεωρώντας ότι οι περιοχές όλων των μεταβλητών είναι οι ίδιες, δηλαδή το διάστημα [0..9], και προσθέτοντας στους παραπάνω περιορισμούς δύο περιορισμούς ανισότητας:

$$S \neq 0, \ M \neq 0.$$

Ακόμα μια δυνατότητα αποτελείται από την επιπλέον εισαγωγή ανά στήλη της μεταβλητής "carry" που εκτείνεται πάνω από [0..1] και από τη χρησιμοποίηση αντί του παραπάνω ενιαίου περιορισμού ισότητας των ακόλουθων πέντε, έναν για κάθε στήλη:

$$D + E = 10 \bullet C1 + Y,$$

$$C1 + N + R = 10 . C_2 + E,$$

$$C_2 + E + O = 10 . C_3 + N,$$

$$C_3 + S + M = 10.C_4 + O,$$

$$C_4 = M.$$

Εδώ, το C1,..., C4 είναι οι 'carry' μεταβλητές.

Στη συνέχεια, οι περιορισμοί μπορούν να αντικατασταθούν από έναν ενιαίο περιορισμό που ορίζει ότι οι μεταβλητές είναι pairwise διαφορετικές. Λαμβάνοντας υπόψη μια ακολουθία μεταβλητών $x_1,\ldots,x_n$ με τις αντίστοιχες περιοχές $D_1,\ldots,D_n$ ορίζουμε

$$\text{all\_different}(x_1,\ldots,x_n) := \{(d_1,\ldots, d_n) \mid di \neq dj \text{ για } i \neq j\}.$$

Κατόπιν μπορούμε να αντικαταστήσουμε τους παραπάνω 28 περιορισμούς ανισότητας από έναν ενιαίο περιορισμό all_different(S, E, N, D, M, O, R, Y).

Ακόμα ένας τρόπος για να εξετάσουμε αυτούς τους περιορισμούς ανισότητας είναι να ακολουθήσουμε μια τυποποιημένη μέθοδο που χρησιμοποιείται στον τομέα του Γραμμικού Προγραμματισμού Ακέραιων αριθμών, στόχος της οποίας είναι να βρεθούν οι βέλτιστες λύσεις ακέραιων αριθμών στους γραμμικούς περιορισμούς. Για κάθε ζευγάρι x,y διαφορετικών μεταβλητών από το (S, E, N, D, M, O, R, Y) εισάγουμε μια μεταβλητή $z_{x>y}$

κυμαινόμενη από [0..1] και μετασχηματίζουμε την ανισότητα $x \neq y$ στους δύο ακόλουθους περιορισμούς:

- $x-y \leq 10- 11\, z_{x,y}$
- $y - x \leq 11 z_{x,y} - 1$.

Σημειώστε ότι η $x \neq y$ είναι ισοδύναμη με την ανισότητα $x< y$ ή $y< x$ που είναι ισοδύναμη με το $x-y \leq -1$ ή $y -x \leq -1$, το οποίο είναι στη συνέχεια ισοδύναμο με την ανισότητα

$$(x - y \leq 10 - 11 z_{x,y} \text{ και } z_{x,y} = 1) \text{ ή } (y - x \leq 11 z_{x,y} - 1 \text{ και } z_{x,y} = 0).$$

Έτσι για να ικανοποιήσουμε τον περιορισμό $x \neq y$ αρκεί να ικανοποιήσουμε έναν από τους δύο παραπάνω περιορισμούς x,y και $z_{x,y}$. Το μειονέκτημα αυτής της προσέγγισης είναι ότι πρέπει να εισάγουμε 28 νέες μεταβλητές.

Το παραπάνω πρόβλημα έχει μια μοναδική λύση η οποία αναπαριστάται με το ακόλουθο άθροισμα:

$$\begin{array}{r} 9567 \\ +\ 1085 \\ \hline 10652 \end{array}$$

Κατά συνέπεια, κάθε μια από τις παραπάνω αντιπροσωπεύσεις του έχει μία μοναδική λύση, εξίσου καλά όπως ένα CSP .

Παράδειγμα 2.2 Το Πρόβλημα των n Βασιλισσών.

Αυτό είναι πιθανώς το πιό γνωστό CSP. Κάποιος καλείται να τοποθετήσει τις βασίλισσες n στον πίνακα σκακιού διαστάσεων n x n, όπου $n \geq 3$, έτσι ώστε να μην επιτίθενται η μια στην άλλη. Το Σχήμα 2.1 παρουσιάζει μια λύση στο πρόβλημα για n = 8.

Ακολουθεί μία πιθανή απεικόνιση αυτού του προβλήματος καθώς ένα CSP χρησιμοποιεί n μεταβλητές, $x_1,\dots ,x_n$, κάθε μια με την περιοχή [1..n]. Η ιδέα είναι ότι η xi δείχνει τη θέση της βασίλισσας που τοποθετείται στη στήλη ith του πίνακα σκακιού. Παραδείγματος χάριν, η λύση που παρουσιάζεται στο Σχήμα 2.1 αντιστοιχεί στην ακολουθία τιμών (6,4,7,1,8,2,5,3), δεδομένου ότι η πρώτη βασίλισσα από τα αριστερά τοποθετείται στον 6^{η} σειρά υπολογίζοντας από το κατώτατο σημείο, και ομοίως με τις άλλες βασίλισσες.

Οι κατάλληλοι περιορισμοί μπορούν να διατυπωθούν ως ακόλουθες ανισότητες για το i Ε [1..n — 1] και j Ε [i + 1..n]:

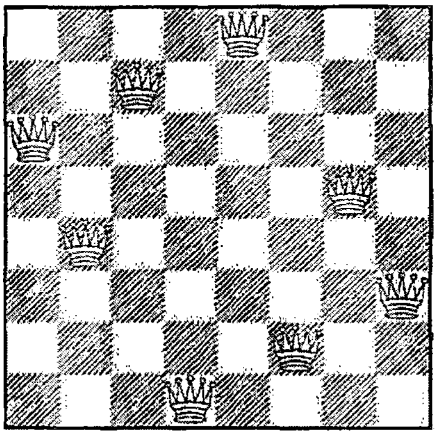
- $x_i \neq x_j$ (καμία από τις δύο βασίλισσες στην ίδια σειρά),
- $x_i - x_j \neq i - j$ (καμία από τις δύο βασίλισσες σε κάθε Νοτιοδυτική - Βορειοανατολική διαγώνιο),
- $x_i - x_j \neq j - i$ (καμία από τις δύο βασίλισσες σε κάθε Βορειοδυτική - Νοτιοανατολική διαγώνιο).

Χρησιμοποιώντας τον all_different περιορισμό που εισάγεται στο προηγούμενο παράδειγμα μπορούμε να αντικαταστήσουμε το πρώτο σύνολο ανισοτήτων {n.(n-1)}/2 από έναν ενιαίο περιορισμό all_different ($x_1,\dots ,x_n$). Στην Παράγραφο 2.4 θα εξετάσουμε μία άλλη φυσική αντιπροσώπευση αυτού του προβλήματος.

Παράδειγμα 2.3 Ο γρίφος Ζέμπρα.

Ένα άλλο παράδειγμα εξετάζει τον ακόλουθο διάσημο γρίφο του Lewis Carroll. Μια μικρή οδός έχει πέντε διαφορετικά χρωματισμένα σπίτια. Πέντε άτομα διαφορετικών υπηκοοτήτων ζουν σε αυτά τα πέντε σπίτια. Κάθε άτομο έχει ένα διαφορετικό επάγγελμα, κάθε άτομο προτιμά ένα διαφορετικό ποτό, και κάθε ένα έχει ένα διαφορετικό κατοικίδιο ζώο. Έχουμε τις ακόλουθες πληροφορίες:

Ο Άγγλος ζει στο κόκκινο σπίτι.
Ο Ισπανός έχει ένα σκυλί.
Ο Ιάπωνας είναι ζωγράφος.
Ο Ιταλός πίνει το τσάι.



Σχήμα 2.1. Μια από τις 92 λύσεις στο πρόβλημα των 8 βασιλισσών

Ο Νορβηγός ζει στο πρώτο σπίτι στα αριστερά.

Ο ιδιοκτήτης του θερμοκηπίου πίνει καφέ.

Το θερμοκήπιο είναι στα δεξιά του άσπρου σπιτιού.

Ο γλύπτης δημιουργεί τα σαλιγκάρια.

Ο διπλωμάτης ζει στο κίτρινο σπίτι.

Πίνουν γάλα στο μεσαίο σπίτι.

Ο Νορβηγός ζει στην επόμενη πόρτα στο μπλε σπίτι.

Ο βιολιστής πίνει χυμό φρούτων.

Η αλεπού είναι δίπλα στο σπίτι του γιατρού.

Το άλογο είναι δίπλα στο σπίτι του διπλωμάτη.

Η ερώτηση είναι ποιος έχει την ζέμπρα και ποιος πίνει νερό;

Μια ενδιαφέρουσα πτυχή αυτού του γρίφου είναι ότι στα στοιχεία του, ούτε η ζέμπρα ούτε το νερό δεν αναφέρονται και όμως έχει μια μοναδική λύση. Για να το διατυπώσουμε ως CSP προσπαθούμε αρχικά να ορίσουμε τις μεταβλητές και τις περιοχές τους. Σημειώστε ότι αυτός ο γρίφος περιλαμβάνει:

- πέντε σπίτια, τα οποία αριθμούμε από τα αριστερά προς τα δεξιά: 1, 2 ,3 ,4 ,5,
- πέντε χρώματα, δηλαδή κόκκινο, πράσινο, άσπρο, κίτρινο, μπλε,
- πέντε υπηκοότητες, δηλαδή αγγλικά, ισπανικά, ιαπωνικά, ιταλικά, νορβηγικά,

- πέντε κατοικίδια ζώα, δηλαδή σκυλί, σαλιγκάρια, αλεπού, άλογο, και (ανεπιφύλακτα) ζέμπρα,
- πέντε επαγγέλματα, δηλαδή ζωγράφος, γλύπτης, διπλωμάτης, βιολιστής, γιατρός,
- πέντε ποτά, δηλαδή τσάι, καφές, γάλα, χυμός, και (ανεπιφύλακτα) νερό.

Για να λύσει κανείς αυτόν τον γρίφο αρκεί να αποφασίσει πέντε χαρακτηριστικά γνωρίσματα για κάθε σπίτι:

- χρώμα,
- υπηκοότητα του ιδιοκτήτη,
- κατοικίδιο ζώο του ιδιοκτήτη,
- επάγγελμα του ιδιοκτήτη,
- αγαπημένο ποτό του ιδιοκτήτη.

Έτσι εισάγουμε 25 μεταβλητές, πέντε για κάθε ένα από τα πέντε παραπάνω χαρακτηριστικά. Για αυτές τις μεταβλητές χρησιμοποιούμε τα ακόλουθα μνημονικά ονόματα:

- μεταβλητές "χρώματος": κόκκινο, πράσινο, άσπρο, κίτρινο, μπλε,
- μεταβλητές "υπηκοότητας": αγγλικά, ισπανικά, ιαπωνικά, ιταλικά, νορβηγικά,
- μεταβλητές "κατοικίδιων ζώων": σκυλί, σαλιγκάρια, αλεπού, άλογο, ζέμπρα,
- μεταβλητές "επαγγέλματος": ζωγράφος, γλύπτης, διπλωμάτης, βιολιστής, γιατρός,
- μεταβλητές "ποτών": τσάι, καφές, γάλα, χυμός, νερό.

Υποθέτουμε ότι κάθε μια από αυτές τις μεταβλητές κυμαίνεται στο διάστημα [1... 5]. Εάν, παραδείγματος χάριν, βιολιστής = 3 τότε αυτό το ερμηνεύουμε ως δήλωση ότι ο βιολιστής ζει στο σπίτι με αριθ. 3. Μετά από αυτές τις προετοιμασίες μπορούμε τώρα να τυποποιήσουμε τις δεδομένες πληροφορίες ως τους ακόλουθους περιορισμούς:

- Ο Άγγλος ζει στο κόκκινο σπίτι: αγγλικά = κόκκινο,
- Ο Ισπανός έχει ένα σκυλί: Ισπανός = σκυλί,
- Ο Ιάπωνας είναι ζωγράφος: ιαπωνικά = ζωγράφος,
- Ο Ιταλός πίνει το τσάι: ιταλικά = τσάι,
- Ο Νορβηγός ζεί στο πρώτο σπίτι στα αριστερά: νορβηγικά = 1,
- Ο ιδιοκτήτης του θερμοκηπίου πίνει καφέ: πράσινο = καφές,
- Το θερμοκήπιο είναι στα δεξιά του άσπρου σπιτιού: πράσινο = λευκό + 1,
- Ο γλύπτης δημιουργεί σαλιγκάρια: γλύπτης = σαλιγκάρια,
- Ο διπλωμάτης ζει στο κίτρινο σπίτι: διπλωμάτης = κίτρινο,
- Πίνουν γάλα στο μεσαίο σπίτι: γάλα = 3,
- Ο Νορβηγός ζει στην επόμενη πόρτα στο μπλε σπίτι: |νορβηγικά — μπλέ| = 1,
- Ο βιολιστής πίνει το χυμό φρούτων: βιολιστής = χυμός,
- Η αλεπού είναι δίπλα στο σπίτι του γιατρού: | αλεπού — γιατρός | = 1,
- Το άλογο είναι δίπλα στο σπίτι του διπλωμάτη: | άλογο —διπλωμάτης | = 1.

Επιπλέον, πρέπει να ορίσουμε ως προϋπόθεση ότι για κάθε ένα από τα χαρακτηριστικά οι αντίστοιχες μεταβλητές είναι διαφορετικές. Αυτό σημαίνει ότι εισάγουμε πενήντα περιορισμούς ανισότητας, δέκα για κάθε χαρακτηριστικό. Παραδείγματος χάριν το κόκκινο≠λευκό είναι μία από αυτές τις ανισότητες. Εναλλακτικά, αντί να θεωρήσουμε δεδομένους αυτούς τους πενήντα περιορισμούς ανισότητας μπορούμε να χρησιμοποιήσουμε τον all_different περιορισμό που εισάγεται στο Παράδειγμα 2.1 και στη θέση των παραπάνω να θεωρήσουμε δεδομένους τους εξής:

all_different(κόκκινο, πράσινο, άσπρο, κίτρινο, μπλε),

all_different(αγγλικά, ισπανικά, ιαπωνικά, ιταλικά, νορβηγικά),

all_different(σκυλί, σαλιγκάρια, αλεπού, άλογο, ζέμπρα),

all_different(ζωγράφος, γλύπτης, διπλωμάτης, βιολιστής, γιατρός),

all_different(τσάι, καφές, γάλα, χυμός, νερό).

Τώρα, αποδεικνύει ότι υπάρχει ακριβώς μια ανάθεση τιμών και στις 25 μεταβλητές για τις οποίες όλοι οι περιορισμοί είναι ικανοποιημένοι. Λόγω αυτού πράγματι ο γρίφος έχει μια μοναδική λύση, ο γρίφος λύνεται μόλις βρούμε αυτήν την μοναδική ανάθεση για την οποία οι μεταβλητές "επαγγέλματος"

x, y Ε {ζωγράφος, γλύπτης, διπλωμάτης, βιολιστής, γιατρός}

έχουμε

x = ζέμπρα και y= νερό.

Η απάντηση είναι x=ιαπωνικά και y=νορβηγικά, το οποίο σημαίνει πως οι Ιάπωνες έχουν την ζέμπρα και οι Νορβηγοί πίνουν νερό.

2.3 Προβλήματα ικανοποίησης περιορισμών στους πραγματικούς

Ας προχωρήσουμε τώρα στην περίπτωση των CSPs οι μεταβλητές των οποίων κυμαίνονται πέρα από τους πραγματικούς.

Παράδειγμα 2.4. Προγράμματα λογιστικού φύλλου (spreadsheets)

Τα συστήματα προγραμμάτων λογιστικού φύλλου (spreadsheet), όπως το Excel, είναι πολύ δημοφιλή για τις διάφορες εφαρμογές του office. Γενικά αυτά τα συστήματα έχουν διάφορα πολύ προηγμένα χαρακτηριστικά γνωρίσματα αλλά η ουσία τους στηρίζεται στους περιορισμούς.

Πιό συγκεκριμένα θεωρήστε τον Πίνακα 2.1. Αντιπροσωπεύει έναν υπολογισμό με πρόγραμμα λογιστικού φύλλου (spreadsheet), στο οποίο οι τιμές για τα κελιά D4, D5, E7 και E8 υπολογίζονται με τη βοήθεια των τύπων που υπάρχουν σε αυτά τα κελιά.

Ισοδύναμα, μπορούμε να αναπαραστήσουμε το πρόγραμμα λογιστικού φύλλου του Πίνακα 2.1 με τη βοήθεια ενός CSP που αποτελείται από εννέα μεταβλητές: B1, B4, B5, C4, C5, D4, D5, E7 και E8, που κάθε μια κυμαίνεται πέρα από τους πραγματικούς αριθμούς, και τους ακόλουθους εννέα περιορισμούς:

B1 = 0.17,
B4 = 3.5,

	A	B	Γ	Δ	E
1	**Φόρος**	0,17			
2					
3	**Προϊόν**	**Τιμή**	**Ποσότητα**	**Σύνολο**	
4	ντομάτες	3.5	**1.5**	= B4 * C4	
5	πατάτες	1.7	4.5	= B5 * C5	
6					
7				**Μεγάλο**	= D4 + D5
8				**Τελικό ποσό**	= E7 * (1 + B1)

Πίνακας 2.1. Ένα πρόγραμμα λογιστικού φύλλου

	A	B	Γ	Δ	E
1	**Φόρος**	0,17			
2					
3	**Προϊόν**	**Τιμή**	**Ποσότητα**	**Σύνολο**	
4	ντομάτες	3.5	1.5	5.25	
5	πατάτες	1.7	4.5	7.65	
6					
7				**Μεγάλο**	12.9
8				**Τελικό ποσό**	15.093

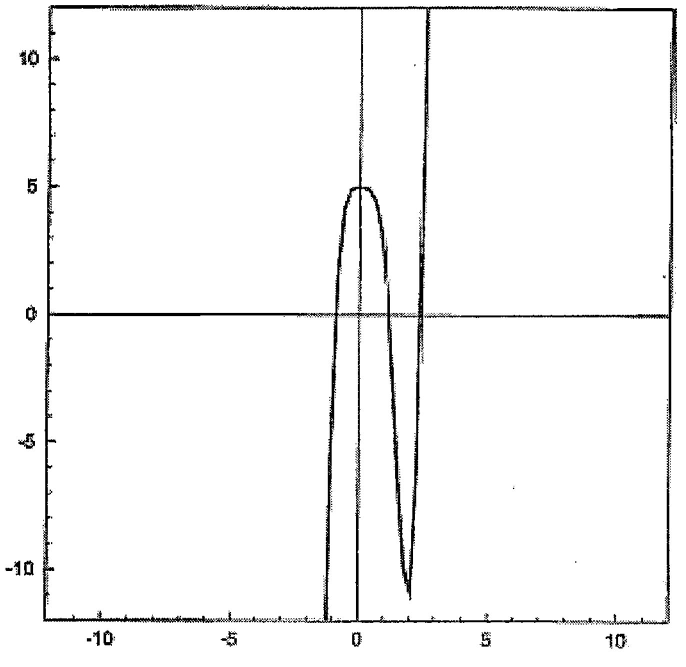
Πίνακας 2.2. Λύση στο πρόγραμμα λογιστικού φύλλου του Πίνακα 2.1

B5=1,7,
C4=1,5,
C5=4,5,
D4=B4*C4,
D5=B5*C5,
E7=D4+D5,
E8=E7*(1+B1).

Ένα σύστημα προγράμματος λογιστικού φύλλου λύνει αυτούς τους περιορισμούς και ενημερώνει τη λύση κάθε φορά που τροποποιείται μια παράμετρος. Το τελευταίο αντιστοιχεί στην τροποποίηση μιας αριθμητικής τιμής u σε έναν περιορισμό της μορφής x = u, όπου a είναι μια μεταβλητή. Παραδείγματος χάριν, μια τροποποίηση του κελιού C5 αντιστοιχεί σε μια τροποποίηση της δεξιάς μεριάς του περιορισμού C5 = 4.5.

Στην περίπτωση του προγράμματος λογιστικού φύλλου του Πίνακα 2.1 η λύση απεικονίζεται από το πρόγραμμα λογιστικού φύλλου του Πίνακα 2.2.

Παράδειγμα 2.5 Βρίσκει τα μηδενικά των πολυωνύμων υψηλότερου βαθμού.



Σχήμα 2.2. Το διάγραμμα των πολυωνύμου $2 * x^5 — 5 * x^4 + 5$

Εξετάστε το πολυώνυμο $2*x^5 — 5*x^4 + 5$ που αντιπροσωπεύεται στο Σχήμα 2.2. Υποθέστε ότι επιθυμούμε να βρούμε τα μηδενικά του. Σύμφωνα με το διάσημο θεώρημα του Galois, η δυσκολία βρίσκεται στο γεγονός ότι γενικά, τα μηδενικά των πολυωνύμων βαθμού υψηλότερου από τέσσερα δεν μπορούν να εκφραστούν ως ριζικά, δηλαδή με τη βοήθεια των τεσσάρων αριθμητικών διαδικασιών και της εξαγωγής της ρίζας. Στην πραγματικότητα, το πολυώνυμο $2 * x^5 — 5 * x^4 + 5$ είναι ένα από αυτά τα πολυώνυμα, που ανακαλύφθηκαν το δέκατο ένατο αιώνα από το Νορβηγό μαθηματικό Niels Henrik Abel. Μια άλλη δυσκολία είναι ότι οι πραγματικοί αριθμοί δεν μπορούν να απεικονιστούν πιστά στον υπολογιστή.

Το τελευταίο πρόβλημα ασχολείται με τη χρησιμοποίηση των πραγματικών αριθμών που αντιπροσωπεύονται μέσω υπολογιστή, δηλ., των αριθμών κινητής υποδιαστολής, και με την παραγωγή λύσεων υπό μορφή διαστημάτων ακολουθιών με όρια κινητής υποδιαστολής.

Οι τεχνικές του προγραμματισμού περιορισμών αποδεικνύεται πως είναι χρήσιμες για την αντιμετώπιση τέτοιων προβλημάτων. Γενικά, λαμβάνοντας υπόψη ένα πολυώνυμο f(x), θεωρούμε ένα CSP με έναν ενιαίο περιορισμό f(x)=0 όπου η μεταβλητή x κυμαίνεται πέρα από τους πραγματικούς αριθμούς. Περαιτέρω, υποθέτουμε ένα σταθερό πεπερασμένο σύνολο αριθμών κινητής υποδιαστολής που αυξάνονται με -∞ και ∞, και που δηλώνονται από το F. Από ένα **διάστημα CSP** υπολογίζουμε εδώ ένα CSP με τον ενιαίο περιορισμό f(x) =0 και μια έκφραση περιοχών της μορφής, x e [l,r], όπου l,r E F το οποίο είναι ένα CSP της μορφής ⟨ (f (x) = 0 ; x E[l, r] ⟩.

Έπειτα το αρχικό CSP μετασχηματίζεται σε μία αποσύνδεση των διαστημάτων CSPs τα διαστήματα των οποίων είναι μικρότερα από κάποια σταθερά εκ των προτέρων ε. Αυτή η αποσύνδεση είναι ισοδύναμη με το αρχικό CSP, υπό την έννοια ότι το σύνολο των λύσεων στο αρχικό CSP είναι ίσο με την ένωση των συνόλων των λύσεων σε σύγκριση με το διαχωρισμένο διάστημα των CSPs.

Στην περίπτωση του παραπάνω πολυωνύμου, που επιλέγει την ακρίβεια 16 ψηφίων μετά από την δεκαδική υποδιαστολή, παίρνουμε την αποσύνδεση τριών διαστημάτων CSPs βασισμένο στα ακόλουθα διαστήματα:

X€ [-.9243580149260359080,

-.9243580149260359031],

X€ [1.171162068483181786,

1.171162068483181791],

X€ [2.428072792707314923,

2.428072792707314924].

Απαιτείται ένα πρόσθετο επιχείρημα για να αποδειχθεί ότι κάθε διάστημα περιέχει το μηδέν. Αυτές οι εκτιμήσεις γενικεύονται στα πολυώνυμα σε έναν αυθαίρετο αριθμό από τις μεταβλητές και στα περιορισμένα προβλήματα βελτιστοποίησης σε πραγματικούς αριθμούς βάση των οποίων καλούμαστε να βελτιστοποιήσουμε κάποια πραγματική-εκτιμώμενη λειτουργία εξαρτημένη από τους πολυωνυμικούς περιορισμούς πέρα από τους πραγματικούς.

2.4 Λογικά προβλήματα ικανοποίησης περιορισμών

Μόλις προσδιορίσουμε το **ψευδές** με 0 και το **αληθές** με 1, τα λογικά CSPs διαμορφώνουν μια ειδική περίπτωση αριθμητικών CSPs στα οποία οι μεταβλητές κυμαίνονται πέρα από τη δυαδική περιοχή [0..1] και οι περιορισμοί εκφράζονται με τη βοήθεια λογικών εκφράσεων που δημιουργούνται χρησιμοποιώντας κάποιο βασικό συνδετικό σύνολο. Από αυτήν την άποψη το πρώτο προσδιορισμένο NP-ολοκληρωμένο πρόβλημα, το πρόβλημα satisfiability, σύμφωνα με το οποίο ρωτάμε εάν μια δεδομένη λογική έκφραση μπορεί να ικανοποιηθεί, είναι ισοδύναμο με την ερώτηση εάν το αντίστοιχο CSP που λαμβάνει λογικές τιμές 0 η΄ 1 είναι σταθερό. Γενικά, όπως θα δούμε στο Κεφάλαιο 6, η χρήση των τεχνικών προγραμματισμού περιορισμών δεν φέρνει οποιεσδήποτε νέες ιδέες σε αυτήν την περιοχή. Εντούτοις, αυτές οι τεχνικές μας επιτρέπουν να διαμορφώσουμε ορισμένες μορφές συλλογισμού με απλό τρόπο, επίσης ως προγράμματα.

Παράδειγμα2.6 Το πλήρες κύκλωμα αθροιστών.

Αυτό το παράδειγμα εξετάζει τα ψηφιακά κυκλώματα κατασκευασμένα από τις πύλες AND, OR και XOR. Αυτές οι πύλες παράγουν μια τιμή κυκλώματος δεδομένων δύο εισαγόμενων τιμών.

Οι πιθανές τιμές προέρχονται από [0..1], έτσι εδώ ασχολούμαστε με τα δυαδικά ψηφία ή, εναλλακτικά, με τις λογικές μεταβλητές. Η συμπεριφορά αυτών των πυλών ορίζεται από τους ακόλουθους πίνακες:

AND	0	1
0	0	0
1	0	1

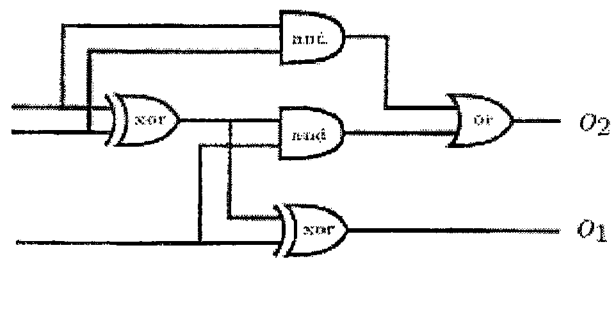
OR	0	1
0	0	1
1	1	1

XOR	0	1
0	0	1
1	1	0

Εναλλακτικά, μπορούμε να δούμε αυτές τις πύλες ως συνδετικά: η πύλη AND αντιστοιχεί στην σύνδεση, η πύλη OR στην αποσύνδεση και η πύλη XOR στην αποκλειστική αποσύνδεση. Επομένως τα κυκλώματα που κατασκευάζονται από αυτές τις πύλες μπορούν να αντιπροσωπευθούν φυσικά από τις εξισώσεις μεταξύ των λογικών εκφράσεων που περιλαμβάνουν την σύνδεση, που γράφεται ως $\wedge$, την αποσύνδεση, που γράφεται ως $\vee$, και την αποκλειστική αποσύνδεση, που γράφεται ως $\oplus$. Ειδικότερα, το κύκλωμα που απεικονίζεται στο Σχήμα 2.3 μπορεί να αντιπροσωπευθεί από τις δύο ακόλουθες εξισώσεις:

$$(i_1 \oplus i_2) \oplus i_3 = o_1, \tag{2.1}$$

$$(i_1 \wedge i_2) \vee (i_3 \wedge (i_1 \oplus i_2)) = o_2. \tag{2.2}$$



Σχήμα 2.3. Πλήρες κύκλωμα αθροιστών

Αυτό το κύκλωμα καλείται **πλήρης αθροιστής** δεδομένου ότι υπολογίζει το δυαδικό άθροισμα $i_1+ i_2 + i_3$ στη δυαδική λέξη $0_2 0_1$. Παραδείγματος χάριν το δυαδικό άθροισμα 1+1+0 παράγει 10. Για να ελέγξουμε την ακρίβεια αυτού του κυκλώματος αρκεί να χρησιμοποιήσουμε τους τρεις παραπάνω πίνακες και να υπολογίσουμε χρησιμοποιώντας τις εξισώσεις (2.1) και (2.2) τα αποτελέσματα $0_1, 0_2$ για όλους τους συνδυασμούς των εισαγόμενων i_1, i_2, i_3.

Το γεγονός ότι αντιπροσωπεύουμε αυτό το κύκλωμα ως ένα λογικό CSP, χρησιμοποιώντας τις σχέσεις μεταξύ των λογικών μεταβλητών i_1, i_2, i_3 και 0_2 που ορίζονται από τις παραπάνω εξισώσεις, αντί ως λειτουργία από (i_1, i_2, i_3) σε (0_1, 0_2) θα μας επιτρέψουν να εξάγουμε με έναν συστηματικό τρόπο πιό σύνθετα συμπεράσματα όπως ότι $i_3 = 0$ και $0_2= 1$ που υπονοούν ότι $i_1, i_2= 1$ και $0_1 = 0$.

Παράδειγμα 2.7 Το πρόβλημα των n Βασιλισσών, ξανά.

Κατά την εξέταση του Προβλήματος των n Βασιλισσών (Παράδειγμα 2.2) επιλέξαμε μια απεικόνιση που περιλαμβάνει τις n μεταβλητές , κάθε μια συνδεδεμένη με μια σειρά. Μια διαφορετική απλή απεικόνιση περιλαμβάνει n^2 λογικές μεταβλητές x_{ij}, όπου i E [1..n] και j E [1..n], καθεμιά από τις οποίες απεικονίζουν έναν τομέα του πίνακα σκακιού. Οι κατάλληλοι περιορισμοί μπορούν έπειτα να γραφούν ως λογικοί περιορισμοί που αντιπροσωπεύονται από τις λογικές εκφράσεις οι οποίοι δημιουργούνται χρησιμοποιώντας τα συνδετικά ένωσης και αναίρεσης (γραμμένα ως$\neg$) .

Για αυτόν τον σκοπό εισάγουμε την ακόλουθη σύντμηση. Λαμβάνοντας υπόψη k λογικές εκφράσεις $s_1,...,s_k$ δηλώνουμε από ένα σύνολο ($s_1,...,s_k$) την λογική έκφραση που διατυπώνει ότι ακριβώς μια από τις εκφράσεις $s_1,...,s_k$ είναι αληθινή. Έτσι ένα σύνολο ($s_1,...,s_k$) είναι μια αποσύνδεση k εκφράσεων , κάθε μια από τις οποίες είναι μια K- ary σύνδεση της μορφής $\neg s_1 \wedge ... \neg s_{i-1} \wedge s_i \wedge \neg s_{i+1} ... \wedge \neg s_k$, όπου i E [1..k].

Οι ακόλουθοι περιορισμοί τυποποιούν έπειτα το πρόβλημα:

- Ένα σύνολο ($x_1...$, $x_{i,n}$) για κάθε i E [1..n] (ακριβώς μια βασίλισσα ανά σειρά),
- Ένα σύνολο($x_{i,i}...$, $x_{n,i}$) για κάθε i E [1..n] (ακριβώς μια βασίλισσα ανά στήλη),
- $\neg(x_{i,j} \wedge x_{k,l}$) για κάθε i, j, k,l, E [1..n] έτσι ώστε $i \neq k$ και and |i — k| = |j — l| (το πολύ-πολύ μια βασίλισσα ανά διαγώνιο).

Ο όρος $i \neq k$ εξασφαλίζει ότι στο τελευταίο στοιχείο οι τομείς που αντιπροσωπεύονται από τα $x_{i,j}$ και $x_{k,l}$ είναι διαφορετικοί.

2.5 Συμβολικά προβλήματα ικανοποίησης περιορισμών

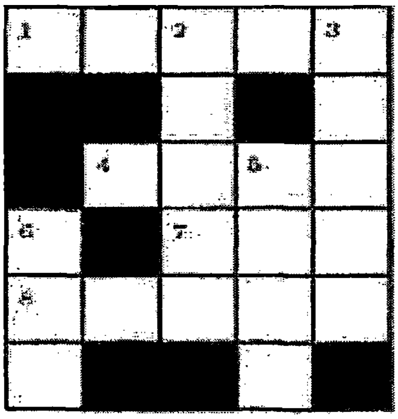
Από ένα συμβολικό πρόβλημα ικανοποίησης περιορισμών υπολογίζουμε ένα CSP οι μεταβλητές του οποίου εκτείνονται πέρα από τις μη αριθμητικές περιοχές.

Παράδειγμα 2.8 Ο Γρίφος Σταυρόλεξων.

Εξετάστε το πλέγμα του σταυρόλεξου του Σχήματος 2.4 και υποθέστε ότι πρόκειται να το γεμίσουμε με τις λέξεις από τον ακόλουθο κατάλογο:

- HOSES, LASER, SAILS, SHEET, STEER,
- HEEL, HIKE, KEEL, KNOT, LINE,
- AFT, ALE, EEL, LEE, TIE.

Αυτό το πρόβλημα μπορεί να διατυπωθεί ως CSP ως εξής. Κατ' αρχήν, συνδυάστε με κάθε θέση i E [1 ..8] σε αυτό το πλέγμα μια μεταβλητή. Κατόπιν συνδυάστε με κάθε μεταβλητή την περιοχή που αποτελείται από το σύνολο λέξεων η οποία μπορεί να χρησιμοποιηθεί για να γεμίσει αυτήν την θέση. Παραδείγματος χάριν, η θέση 6 πρέπει να γεμιστεί με



Σχήμα 2.4. Ένα πλέγμα σταυρόλεξου

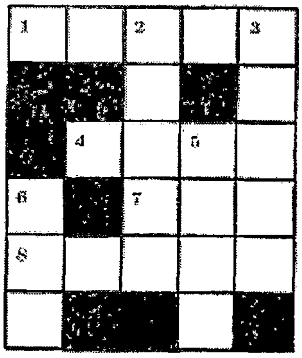
τη λέξη τριών γραμμάτων, έτσι ώστε η περιοχή της μεταβλητής που συνδέεται με τη θέση 6 να αποτελείται από το παραπάνω σύνολο πέντε λέξεων 3 γραμμάτων.

Τέλος, ορίζουμε τους περιορισμούς. Χειρίζονται τους περιορισμούς που προκύπτουν από το γεγονός ότι οι λέξεις που παραλείπονται παραχωρούν ένα γράμμα στις κατάλληλες θέσεις. Παραδείγματος χάριν, η διασταύρωση των θέσεων 1 και 2 δίνουν τον ακόλουθο περιορισμό:

$C_{1,2}$: = {(HOSES, SAILS), (HOSES, SHEET), (HOSES, STEER),

(LASER, SAILS), (LASER, SHEET), (LASER, STEER)}.

Αυτός ο περιορισμός τυποποιεί το γεγονός ότι το τρίτο γράμμα της θέσης 1 πρέπει να είναι το ίδιο με το πρώτο γράμμα της θέσης 2. Στο σύνολο υπάρχουν δώδεκα περιορισμοί. Η μοναδική λύση σε αυτό το CSP απεικονίζεται στο Σχήμα 2.5.



Σχήμα 2.5. Η λύση στο γρίφο σταυρόλεξων

Τα δύο επόμενα παραδείγματα ασχολούνται με **τον ποιοτικό συλλογισμό**. Με αυτήν την μορφή συλλογισμού κάποιο αφαιρείται από τις αριθμητικές ποσότητες, όπως ο ακριβής χρόνος ενός γεγονότος, ή η θέση ενός αντικειμένου στο διάστημα, και των λόγων αντ' αυτού στο επίπεδο αφαιρέσεών τους. Συνήθως, αυτές οι αφαιρέσεις διαμορφώνουν ένα πεπερασμένο σύνολο από εναλλακτικές λύσεις, σε αντιδιαστολή με το άπειρο σύνολο δυνατοτήτων που είναι διαθέσιμες στο αριθμητικό επίπεδο. Αυτό μας επιτρέπει να εφαρμόσουμε συμπεράσματα σε θεωρητικό επίπεδο που στο αριθμητικό επίπεδο θα ήταν δύσκολο να επιτευχθεί κάτι τέτοιο. Εξετάζουμε τώρα δύο παραδείγματα του ποιοτικού συλλογισμού.

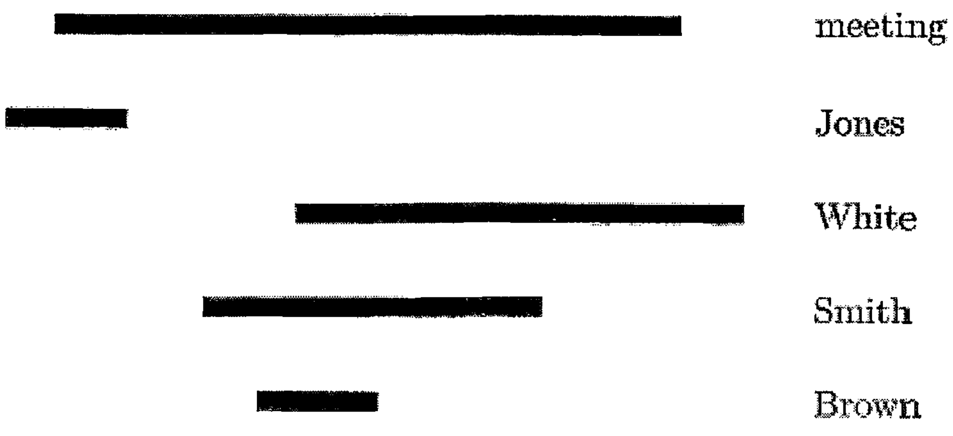
Παράδειγμα 2.9. Ποιοτικός χρονικός συλλογισμός.

Εξετάστε το ακόλουθο πρόβλημα.

Η συνεδρίαση πραγματοποιήθηκε ασταμάτητα ολόκληρη την ημέρα. Κάθε πρόσωπο έμεινε στη συνεδρίαση για μια συνεχή χρονική περίοδο. Η συνεδρίαση άρχισε ενώ ο κ. Jones ήταν παρών και τελείωσε ενώ η κα White ήταν παρούσα. Η κα White έφθασε αφότου η συνεδρίαση είχε ξεκινήσει. Στη συνέχεια, ο διευθυντής Smith ήταν επίσης παρών αλλά έφθασε αφότου είχε φύγει ο Jones. Ο κ. Brown μίλησε στην κα White ενώ ήταν παρών ο Smith. Θα μπορούσε ενδεχομένως ο Jones και η White να έχουν μιλήσει κατά τη διάρκεια αυτής της συνεδρίασης;

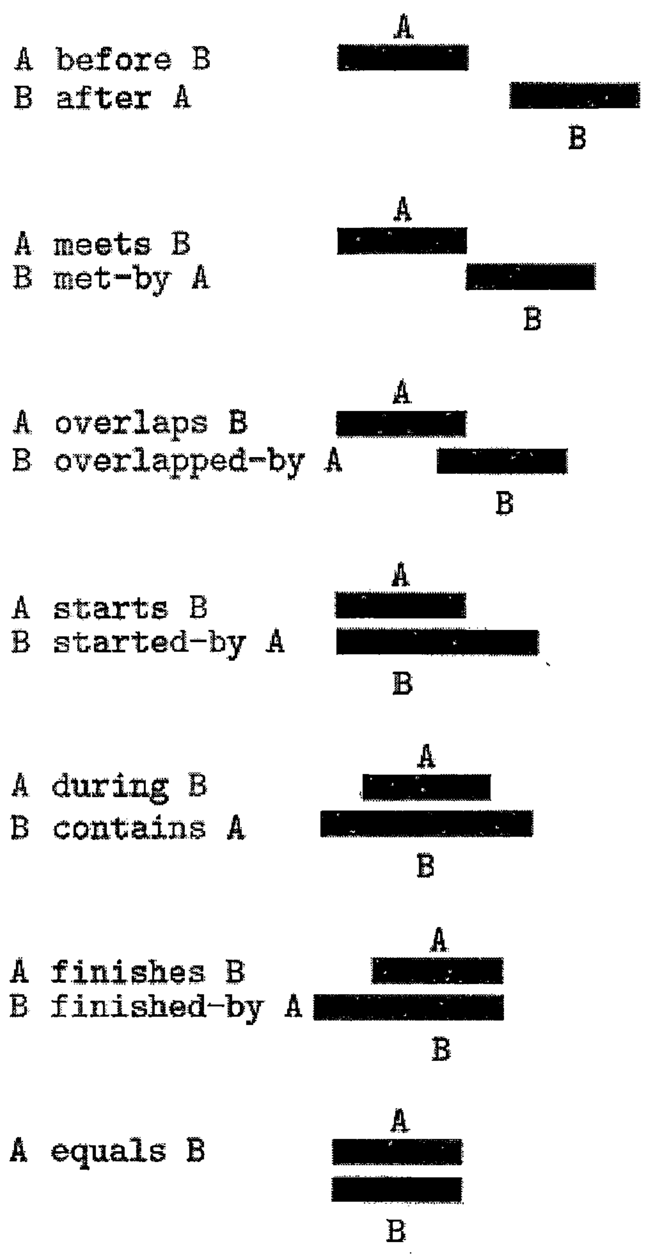
Για να αναλύσουμε κατάλληλα τέτοια προβλήματα οδηγούμαστε αυθόρμητα σε μια αφηρημένη ανάλυση δραστηριοτήτων που καταναλώνουν χρόνο, όπως το να είσαι παρών κατά τη διάρκεια της συνεδρίασης, να μιλάς σε κάποιον, να πηγαίνεις στη δουλειά, να κάνεις ένα μεσημεριανό διάλειμμα, να συμπληρώσεις μια φόρμα, να λάβεις ένα τηλεφώνημα, κ.λ.π.... Στη συνέχεια καλούμε τέτοιες δραστηριότητες **χρονικά γεγονότα,** ή απλά **γεγονότα.** Εάν λάβουμε υπόψη μας μόνο το γεγονός ότι τέτοια γεγονότα κρατάνε μια συνεχή αλλά περιορισμένη χρονική περίοδο, τότε μπορούμε να τα προσδιορίσουμε με κλειστά μη-άδεια διαστήματα της πραγματικής γραμμής.

Στην περίπτωση του παραπάνω προβλήματος ένα πιθανό σενάριο που ταιριάζει με την περιγραφή του παρέχεται στο Σχήμα 2.6, όπου κάθε γεγονός (όπως κατά τη διάρκεια της συνεδρίασης ή της περιόδου κατά τη διάρκεια της οποίας ο Jones ήταν παρών) αντιπροσωπεύεται από ένα κλειστό μη-άδειο πραγματικό διάστημα.



Σχήμα 2.6. Ένα πιθανό σενάριο για το "πρόβλημα συνεδρίασης"

Γενικά, κατά τον προσδιορισμό των γεγονότων με κλειστά μη-άδεια πραγματικά διαστήματα καταλήγουμε σε δεκατρείς πιθανές **χρονικές σχέσεις** μεταξύ ενός ζευγαριού γεγονότων. Παρουσιάζονται στο Σχήμα 2.7. Διαισθητικά, αυτές οι σχέσεις προκύπτουν όταν εξετάσουμε δύο διαστήματα, Α και Β, και συνεχίζουν στο διάστημα Α από τα αριστερά προς τα δεξιά και καταγράφουν όλες τις πιθανές σχετικές θέσεις του w.r.t. B, λαμβάνοντας υπόψη τα ενδεχομένως διαφορετικά σχετικά μεγέθη τους.



Σχήμα 2.7. Δεκατρείς χρονικές σχέσεις

Αφήστε τα TEMP να δείξουν το σύνολο που διαμορφώνεται από τις δεκατρείς δυνατότητες του Σχήματος 2.7. Στη συνέχεια παρέχουμε δύο αντιπροσωπεύσεις των CSPs που εξετάζουν το χρονικό συλλογισμό σε αυτήν την ρύθμιση. Ο πρώτος είναι εννοιολογικά απλούστερος αλλά περιλαμβάνει άπειρες περιοχές. Ο δεύτερος είναι αρχικά πιο σύνθετος αλλά έχει το πλεονέκτημα ότι οι περιοχές είναι πεπερασμένες και αντιστοιχούν άμεσα σε ορισμένα TEMP.

Πρώτη αντιπροσώπευση

Σε αυτήν την αντιπροσώπευση οι μεταβλητές αναπαριστάνουν τα γεγονότα. Οι περιοχές τους αντικατοπτρίζουν την εικόνα ότι τα γεγονότα προσδιορίζονται με κλειστά μη- άδεια διαστήματα της πραγματικής γραμμής.

Έτσι κάθε περιοχή αποτελείται από το σύνολο τέτοιων διαστημάτων, που είναι το σύνολο

$$D := \{(a, b) \mid a, b \in R, a < b\},$$

Όπου R είναι το σύνολο όλων των πραγματικών και κάθε ζευγάρι (α, b) αντιπροσωπεύει το κλειστό διάστημα[α, b] των πραγματικών αριθμών.

Έπειτα, κάθε μια από τις εισαχθείσες δεκατρείς χρονικές σχέσεις αντιπροσωπεύεται ως δυαδικός περιορισμός με τέτοιο τρόπο που να απεικονίζει την επιδιωκόμενη έννοια. Παραδείγματος χάριν, η σχέση **overlaps** αντιπροσωπεύεται όπως το ακόλουθο υποσύνολο D x D:

$$[[\text{overlaps}]] := \{((a_{begin}, a_{,end}), (b_{begin}, b_{,end})) \mid a_{begin} < b_{begin},\ b_{begin} < a_{,end},\ a_{,end} < b_{,end}\}$$

Τέλος, οι αυθαίρετοι περιορισμοί είναι ένα σύνολο θεωρητικών ενώσεων των στοιχειωδών δυαδικών περιορισμών που αντιπροσωπεύουν τις δεκατρείς χρονικές σχέσεις.

Ας επεξηγήσουμε τώρα αυτήν την αντιπροσώπευση τυποποιώντας ως CSP το πρόβλημα από το οποίο ξεκινήσαμε. Κατ' αρχήν, προσδιορίζουμε τα σχετικά γεγονότα και αντιστοιχίζουμε σε καθένα από αυτά μια μεταβλητή. Συνολικά, εξετάζουμε τα ακόλουθα πέντε γεγονότα και τις αντίστοιχες μεταβλητές:

event	variable
the duration of the meeting	M
the period Jones was present	J
the period Brown was present	B
the period Smith was present	S
the period White was present	W

Έπειτα, ορίζουμε τους κατάλληλους περιορισμούς. Οι τρεις πρώτοι τυποποιούν τις πληροφορίες σχετικά με την παρουσία κάθε προσώπου κατά τη διάρκεια της συνεδρίασης. Για λόγους συντομίας εδώ δείχνουμε την ένωση όλων των στοιχειωδών περιορισμών αποκλείοντας τους [[before]], [[after]], [[meets]] και [[met-by]] ως [[REAL-OVERLAP]]. Σημειώστε ότι

$$[[\text{REAL-OVERLAP}]] = \{((a_{begin}, a_{end}), (b_{begin}, b_{end})) \mid a_{begin} < b_{end},\ b_{begin} < a_{end}\}.$$

Αυτοί οι τρεις περιορισμοί είναι:

([[overlaps]] U [[contains]] U [[finished-by]])(J,M),

[[overlaps]] (M, W),

[[REAL-OVERLAP]](M, S).

Σημειώστε ότι ο πρώτος περιορισμός τυποποιεί το γεγονός ότι

- το J άρχισε αυστηρά νωρίτερα απ' ότι άρχισε το M ,
- το M ξεκίνησε αυστηρά πριν τελειώσει το J .

Στη συνέχεια, ο δεύτερος περιορισμός τυποποιεί το γεγονός ότι

- το W ξεκίνησε αυστηρά πριν τελειώσει το M ,
- το M τελείωσε αυστηρά νωρίτερα απ΄ ότι τελείωσε το W,
- το M ξεκίνησε αυστηρά πριν ξεκινήσει W.

Τέλος, ο περιορισμός [[REAL-OVERLAP]](M, S) τυποποιεί το γεγονός ότι τα M και S "πράγματι" συμπίπτουν εγκαίρως, δηλαδή μοιράζουν κάποιο χρονικό διάστημα του θετικού μήκους. Σημειώστε ότι δεν θεωρούμε δεδομένο οποιονδήποτε περιορισμό στα M και B, ειδικότερα όχι τον [REAL-OVERLAP]](M, B), επειδή από τη διατύπωση του προβλήματος δεν είναι σαφές εάν ο Brown ήταν πραγματικά παρών κατά τη διάρκεια της συνεδρίασης.

Επιπλέον, οι ακόλουθοι περιορισμοί τυποποιούν τις πληροφορίες σχετικά με τη σχετική παρουσία των εν λόγω προσώπων:

[[before]] (J, S),

[[REAL-OVERLAP]](B,S),

[[REAL-OVERLAP]] (B,W),

[[REAL-OVERLAP]] (S, W).

Η ερώτηση "Θα μπορούσε ενδεχομένως ο Jones και η White να έχουν μιλήσει κατά τη διάρκεια αυτής της συνεδρίασης;" μπορεί τώρα να τυποποιηθεί ως πρόβλημα, εάν ισχύει για κάποια λύση σε αυτό το CSP [[REAL-OVERLAP]](J, W). Με άλλα λόγια, εάν αληθεύει ότι το ανωτέρω CSP που αυξάνεται από τον περιορισμό [[REAL-OVERLAP]](J, W) είναι σταθερό.

Δεύτερη αντιπροσώπευση

Στην πρώτη αντιπροσώπευση οι περιοχές, κατά συνεπεία και οι εκφράσεις περιοχών, ορίστηκαν μεμονωμένα και το μόνο που είχαμε να κάνουμε ήταν να ορίσουμε τους περιορισμούς που ήταν δυαδικοί. Στη δεύτερη αντιπροσώπευση ορίζουμε αρχικά τις εκφράσεις περιοχών που ορίζουν μεμονωμένα τους περιορισμούς. Σε αυτήν την αντιπροσώπευση μια μεταβλητή συνδέεται με κάθε διαταγμένο ζευγάρι των γεγονότων. Κάθε περιοχή μιας τέτοιας μεταβλητής είναι ένα υποσύνολο από ορισμένα TEMP. Όλοι οι περιορισμοί είναι τριαδικοί.

Πιό συγκεκριμένα, θεωρήστε τα τρία γεγονότα, A, B και C και υποθέστε ότι ξέρουμε τις χρονικές σχέσεις μεταξύ των ζευγαριών A και B, καθώς και των B και C. Η ερώτηση είναι ποια είναι η χρονική σχέση μεταξύ του A και του C. Παραδείγματος χάριν εάν το A επικαλύπτει το B και το B είναι πριν από το C, κατόπιν το A είναι πριν από το C. Για να απαντήσουμε σε αυτήν την ερώτηση πρέπει να εξετάσουμε 169 δυνατότητες. Αντιπροσωπεύονται στον πίνακα αποκαλούμενο **πίνακα σύνθεσης**, που δίνεται στα Σχήματα 2.8 και 2.9. Η είσοδος R-OVERLAP (μια σύντμηση για την REAL-OVERLAP) που εμφανίζεται εκεί τρεις φορές, είναι μια στενογραφία για το σύνολο των χρονικών σχέσεων που εκφράζουν το γεγονός ότι δύο γεγονότα "αληθή" επικαλύπτονται εγκαίρως, κι έτσι το σύνολο αποκτήθηκε από το TEMP με τον αποκλεισμό των σχέσεων before, after, meets and met-by:

REAL-OVERLAP := TEMP- {before, after, meets, met-by}.

Θεωρήστε τώρα όλα τα νόμιμα πιθανά τριπλάσια (t_1, t_2, t_3) των χρονικών σχέσεων έτσι ώστε το t_1 να είναι η χρονική σχέση μεταξύ του A και του B, το t_2 η χρονική σχέση μεταξύ του B και C και το t_3 η χρονική σχέση μεταξύ του A και του C. Το σύνολο αυτών των τριπλάσιων διαμορφώνει ένα υποσύνολο του TEMP ^{3}. Δηλώστε το με T3. Ο πίνακας που μόλις αναφέραμε μας επιτρέπει να υπολογίσουμε το T3. Παραδείγματος χάριν, ήδη παρατηρήσαμε ότι (overlaps, before, before) E T3, δεδομένου ότι το A επικαλύπτει το B και το B είναι πριν το C υπονοήσει ότι το A είναι πριν από το C. Το ορισμένοT3 έχει 409 στοιχεία.

Από μια **διαζευκτική χρονική σχέση** υπολογίζουμε τώρα μια αποσύνδεση των χρονικών σχέσεων. Παραδείγματος χάριν η before V meets είναι μια διαζευκτική χρονική χρονική σχέση. Οι διαζευκτικές χρονικές σχέσεις μας επιτρέπουν να διαμορφώσουμε τις καταστάσεις στις οποίες οι χρονικές εξαρτήσεις μεταξύ των γεγονότων είναι εν_μέρει γνωστές.

Σημειώστε ότι κάθε χρονική σχέση μεταξύ των γεγονότων A και B ορίζει μεμονωμένα τη διαζευκτική χρονική σχέση μεταξύ του γεγονότος B και A. Παραδείγματος χάριν εάν το προηγούμενο είναι before V meets, κατόπιν το επόμενο είναι after V met — by . Έτσι χωρίς απώλεια πληροφοριών μπορούμε να περιορίσουμε την προσοχή μας στις διαζευκτικές χρονικές σχέσεις που συνδέονται με κάθε διαταγμένο ζευγάρι γεγονότων.

Μπορούμε τώρα να ορίσουμε τα CSPs. Υποθέτουμε n γεγονότα $e_1,\dots,e_n$ με $n > 2$. Εξετάζουμε ((n-1).n)/2 εκφράσεις περιοχών, κάθε μια της μορφής x_{ij} E D_{ij} όπου ι, j E [1..n] με ι < j και $D_{ij} \subseteq$TEMP. Η πρόθεση είναι ότι η μεταβλητή x_{ij} περιγράφει τη διαζευκτική χρονική σχέση που συνδέεται με τα γεγονότα e_i και e_j.

Τέλος, ορίζουμε τους περιορισμούς. Κάθε περιορισμός συνδέει μια διαδοχική τριάδα από γεγονότα. Έτσι συνολικά έχουμε ((n-2).(n-1).n)/6 περιορισμούς. Για κάθε i,j,k E [1..n] έτσι ώστε $i < j < k$ ορίζουμε τον περιορισμό $C_{i,j,k}$ θέτοντας

$$C_{i,j,k} := T_3 \cap (D_{i,j} \times D_{j,k}, \times D_{i,k}).$$

Έτσι το $C_{i,j,k}$ περιγράφει τις πιθανές καταχωρήσεις στο T3 ορισμένες από τις περιοχές των μεταβλητών $x_{i,j}$, $x_{j,k}$ και $x_{i,k}$.

	before	after	meets	met-by	overlaps	overl.-by
before	before	*TEMP*	before	before meets overlaps starts during	before	before meets overlaps starts during
after	*TEMP*	after	during finishes after met-by overl.-by	after	during finishes after met-by overl.-by	after
meets	before	after met-by overl.-by started-by contains	before	finishes finished-by equals	before	overlaps starts during
met-by	before overlaps meets contains finished-by	after	starts started-by equals	after	during finishes overl.-by	after
overlaps	before	after met-by overl.-by started-by contains	before	overl.-by started-by contains	before meets overlaps	*R-OVERLAP*
overl.-by	before meets overlaps contains finished-by	after	overlaps contains finished-by	after	*R-OVERLAP*	after met-by overl.-by
starts	before	after	before	met-by	before meets overlaps	during finishes overl.-by
started-by	before meets overlaps contains finished-by	after	overlaps contains finished-by	met-by	overlaps contains finished-by	overl.-by
during	before	after	before	after	before meets overlaps starts during	during finishes after met-by overl.-by
contains	before meets overlaps contains finished-by	after met-by overl.-by contains started-by	overlaps contains finished-by	overl.-by started-by contains	overlaps contains finished-by	overl.-by started-by contains
finishes	before	after	meets	after	overlaps starts during	after met-by overl.-by
finished-by	before	after met-by overl.-by started-by contains	meets	overl.-by started-by contains	overlaps	overl.-by started-by contains
equals	before	after	meets	met-by	overlaps	overl.-by

Σχήμα 2.8. Ο πίνακας σύνθεσης για τις δεκατρείς χρονικές σχέσεις, μέρος 1

Αυτός συμπληρώνει την περιγραφή των CSPs. Επειδή κάθε περιορισμός ορίζεται από το αντίστοιχο τριπλάσιο των περιοχών των μεταβλητών, κάθε τέτοιο CSP ορίζεται μεμονωμένα από τις εκφράσεις των περιοχών του. Επομένως κατά το ορισμό τέτοιων CPSs είναι ικανοποιητικό να οριστούν οι κατάλληλες εκφράσεις περιοχών.

Ας επιστρέψουμε τώρα στο πρόβλημα από το οποίο ξεκινήσαμε. Μπορεί να διατυπωθεί ως CSP, ως εξής. Έχουμε πέντε γεγονότα, M, J, B, S, W. Εδώ το

M αντιπροσωπεύει "τη διάρκεια της συνεδρίασης", το J αντιπροσωπεύει "την περίοδο που ο Jones ήταν παρών",

	starts	started-by	during	contains	finishes	finished-by	equals
before	before	before	before meets overlaps starts during	before	before meets overlaps starts during	before	before
after	during finishes after met-by overl.-by	after	during finishes after met-by overl.-by	after	after	after	after
meets	meets	meets	overlaps starts during	before	overlaps starts during	before	meets
met-by	during finishes overl.-by	after	during finishes overl.-by	after	met-by	met-by	met-by
overlaps	overlaps	overlaps contains finished-by	overlaps starts during	before meets overlaps contains finished-by	overlaps starts during	before meets overlaps	overlaps
overl.-by	during finishes overl.-by	after met-by overl.-by	during finishes overl.-by	after meets overl.-by started-by contains	overl.-by	overl.-by started-by contains	overl.-by
starts	starts	starts started-by equals	during	before meets overlaps contains finished-by	during	before meets overlaps	starts
started-by	starts started-by equals	started-by	during finishes overl.-by	contains	overl.-by	contains	started-by
during	during	during finishes after met-by overl.-by	during	*TEMP*	during	before meets overlaps starts during	during
contains	overlaps contains finished-by	contains	*R-OVERLAP*	contains	overl.-by contains started-by	contains	contains
finishes	during	after met-by overl.-by	during	after met-by overl.-by started-by contains	finishes	finishes finished-by equals	finishes
finished-by	overlaps	contains	overlaps starts during	contains	finishes finished-by equals	finished-by	finished-by
equals	starts	started-by	during	contains	finishes	finished-by	equals

Σχήμα 2.9. Ο πίνακας σύνθεσης για τις δεκατρείς χρονικές σχέσεις, μέρος 2

και ομοίως με τα γεγονότα S (για Smith), B (για Brown) και W (για White). Έπειτα, διατάσσουμε τα γεγονότα με κάποιο αυθαίρετο τρόπο, για παράδειγμα

J, M, B, S, W.

Έχουμε συνολικά δέκα μεταβλητές, όπου κάθε μια συνδέεται με ένα διατεταγμένο ζευγάρι γεγονότων. Ανάλογα, όπως στην πρώτη αντιπροσώπευση χρησιμοποιούμε τη σχέση σύμφωνα με την οποία δύο γεγονότα "αληθή" αλληλεπικαλύπτονται εγκαίρως. Έτσι πλέον αντιπροσωπεύονται από το σύνολο REAL-OVERLAP των εννέα χρονικών σχέσεων που έχει ήδη εισαχθεί.

Κατόπιν οι ακόλουθες εκφράσεις περιοχών εξετάζουν την παρουσία κάθε προσώπου κατά τη διάρκεια της συνεδρίασης:

$x_{J,M}$ E {overlaps, contains, finished-by},

$x_{M,W}$ E **{overlaps},**

$x_{M,S}$ E REAL-OVERLAP,

και οι ακόλουθες εκφράσεις περιοχών τυποποιούν τις πληροφορίες σχετικά με τη σχετική παρουσία των εν λόγω προσώπων:

$x_{J,S}$ E **{before},**

$x_{B,S}$ E REAL-OVERLAP,

$x_{B,W}$ E REAL-OVERLAP,

$x_{s,W}$ E REAL-OVERLAP.

Οι περιοχές των υπόλοιπων τριών μεταβλητών $x_{J,B}$, $x_{J,W}$ και $x_{M,B}$, ισοδυναμούν με TEMP. Η τελική ερώτηση μπορεί εν συνεχεία να διατυπωθεί ως πρόβλημα εάν για κάποια λύση σε αυτό το CSP έχουμε $x_{J,W}$ E REAL-OVERLAP. Με άλλα λόγια, ισχύει ότι, το παραπάνω CSP με τις εκφράσεις περιοχών $x_{J,W}$ E TEMP που αντικαθίστανται από

$X_{J,W}$ E REAL-OVERLAP

είναι σταθερό.

Σημειώστε ότι και στις δύο αντιπροσωπεύσεις υποθέσαμε ότι τα γεγονότα "ήταν παρών κατά τη διάρκεια της συνεδρίασης" και "να μιλάς "είναι θετικής διάρκειας.

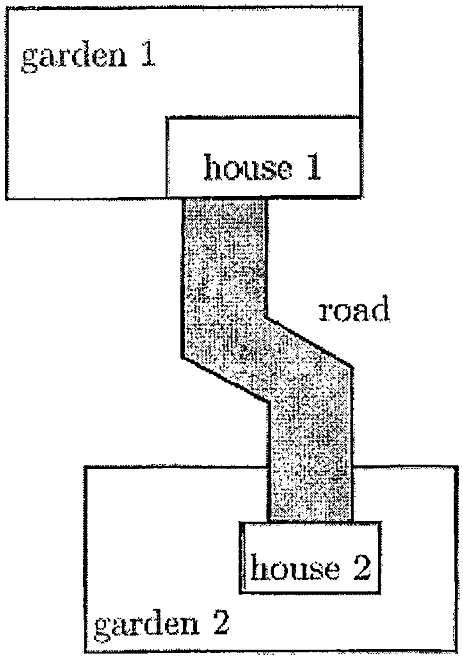
Παράδειγμα 2.10 Ποιοτικός χωρικός συλλογισμός.

Εξετάστε τώρα το ακόλουθο πρόβλημα που διευκρινίζεται στο Σχήμα 2.10.

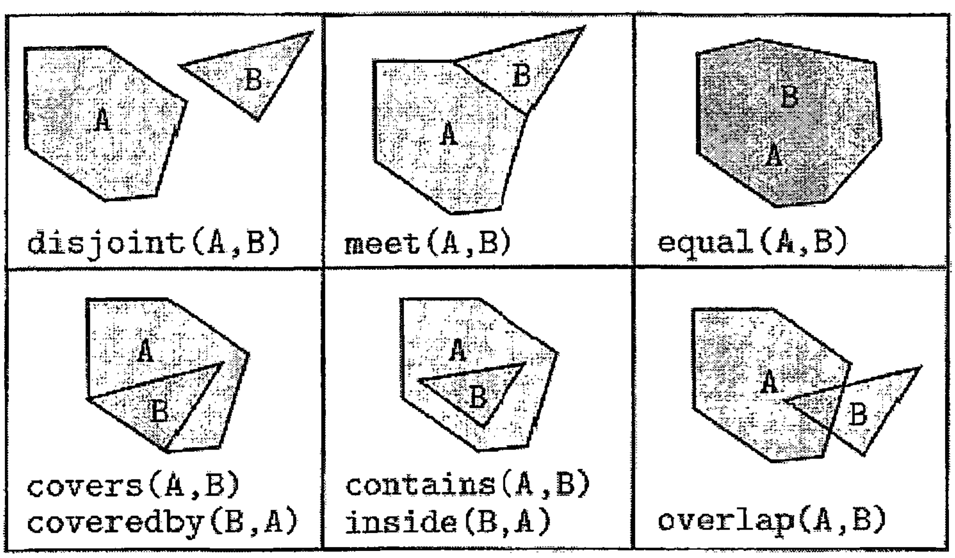
Δύο free-standing σπίτια συνδέονται με έναν δρόμο. Το πρώτο σπίτι περιβάλλεται από τον κήπο του ή είναι δίπλα στο όριό του ενώ το δεύτερο σπίτι περιβάλλεται από τον κήπο του. Η ερώτηση είναι τι μπορούμε εμείς να συμπεράνουμε για τη σχέση μεταξύ του δεύτερου κήπου και του δρόμου.

Για να αναλύσουμε τέτοια προβλήματα πρέπει να σκεφτούμε τις σχετικές θέσεις μεταξύ παραπλήσιων αντικειμένων. (Υποθέτουμε εδώ για να το απλουστεύσουμε ότι κάθε κήπος περιλαμβάνει επίσης την περιοχή κάτω από ένα σπίτι.) Αυτό μας φέρνει στις οκτώ δυνατότητες που συνοψίζονται στο Σχήμα 2.11.

Αυτές οι οκτώ σχέσεις καλούνται συνήθως **χωρικές σχέσεις** και το σύνολό τους,



Σχήμα 2.10. Δύο σπίτια με έναν κήπο και έναν δρόμο



Σχήμα 2.11. Οκτώ χωρικές σχέσεις

{disjoint, meet, equal, covers, coveredby, contains, inside, overlap},

φαίνεται από το RCC8 (για την Ανάλυση της Περιοχής Σύνδεσης με 8 σχέσεις).

Θεωρήστε τώρα τρία παραπλήσια αντικείμενα, A, B και C και υποθέστε ότι γνωρίζουμε τις χωρικές σχέσεις μεταξύ των ζευγαριών A και B, καθώς και των B και C. Η ερώτηση είναι ποια είναι η χωρική σχέση μεταξύ των A και C. Παραδείγματος χάριν εάν το A βρίσκεται μέσα στο B και το B συναντά το C, έπειτα το A διαχωρίζεται από το C. Για να απαντήσουμε σε αυτήν την ερώτηση πρέπει να εξετάσουμε 64 δυνατότητες. Σε μερικές από αυτές υπάρχουν πολλαπλές απαντήσεις. Παραδείγματος χάριν, εάν το A είναι **καλυμμένοαπό** το B και το B είναι **καλυμμένοαπό** το C, κατόπιν το A είναι

είτε στο εσωτερικό του C είτε είναι **καλυμμένοαπό** το C. Ολόκληρος ο πίνακας εξάρτησης, αποκαλούμενος **πίνακα σύνθεσης,** παρουσιάζεται στο Σχήμα 2.12. Η είσοδος RCC8, που εμφανίζεται τρεις φορές, είναι μια στενογραφία για το σύνολο και των οκτώ σχέσεων.

	disjoint	meet	equal	inside	coveredby	contains	covers	overlap
disjoint	RCC8	disjoint meet inside coveredby overlap	disjoint	disjoint meet inside coveredby overlap	disjoint meet inside coveredby overlap	disjoint	disjoint	disjoint meet inside coveredby overlap
meet	disjoint meet contains covers overlap	disjoint meet equal coveredby covers overlap	meet	inside coveredby overlap	meet inside	disjoint	disjoint meet	disjoint meet inside coveredby overlap
equal	disjoint	meet	equal	inside	coveredby	contains	covers	overlap
inside	disjoint	disjoint	inside	inside	inside	RCC8	disjoint meet inside coveredby overlap	disjoint meet inside coveredby overlap
coveredby	disjoint	disjoint meet	coveredby	inside	inside coveredby	disjoint meet contains covers overlap	disjoint meet equal coveredby covers overlap	disjoint meet overlap coveredby overlap
contains	disjoint meet contains covers overlap	contains covers overlap	contains	equal inside coveredby contains covers overlap	contains covers overlap	contains	contains	contains covers overlap
covers	disjoint meet contains covers overlap	meet contains covers overlap	covers	inside coveredby overlap	equal coveredby covers overlap	contains	contains covers	contains covers overlap
overlap	disjoint meet contains covers overlap	disjoint meet contains covers overlap	overlap	inside coveredby overlap	inside coveredby overlap	disjoint meet contains covers overlap	disjoint meet contains covers overlap	RCC8

Σχήμα 2.12. Ο πίνακας σύνθεσης για τις οκτώ χωρικές σχέσεις

Ο πίνακας σύνθεσης του Σχήματος 2.12 ορίζει μια τριαδική σχέση που είναι ένα υποσύνολο του $(RCC8)^3$ και που τη δηλώνουμε με S3. Αποτελείται από όλες τις πιθανές τριάδες (r_1, r_2, r_3) των χωρικών σχέσεων έτσι ώστε το r1 να είναι η χωρική σχέση μεταξύ των A και B, το r2 η χωρική σχέση μεταξύ των B και C, και το r3 η χωρική σχέση μεταξύ των A και C. Συνολικά το S3 περιέχει 193 τριάδες.

Χρησιμοποιώντας το ορισμένο S3 μπορούμε τώρα να τυποποιήσουμε το πρόβλημά μας επιλέγοντας μια αντιπροσώπευση ανάλογη με τη δεύτερη αντιπροσώπευση του χρονικού συλλογισμού του Παραδείγματος 2.9. Έτσι σε αυτήν την αντιπροσώπευση μια μεταβλητή συνδέεται με κάθε διατεταγμένο ζευγάρι αντικειμένων. Οι περιοχές αυτών των μεταβλητών είναι υποσύνολα του ορισμένου RCC8. Όλοι οι περιορισμοί είναι τριαδικοί και ορίζονται μεμονωμένα από τις περιοχές. Πιο συγκεκριμένα, κάθε περιορισμός συνδέεται με μία διατεταγμένη τριάδα αντικειμένων και είναι ίσος με τη ορισμένη-θεωρητική διατομή του S3 με το Καρτεσιανό προϊόν των αντίστοιχων περιοχών των μεταβλητών.

Στο παράδειγμά μας έχουμε πέντε χωρικά αντικείμενα, H1 (για το σπίτι 1), G1 (για τον κήπο 1), H2 (για το σπίτι 2), G2 (για τον κήπο 2), and R (για τον δρόμο). Διατάσσουμε αυτά τα αντικείμενα αυθαίρετα, παραδείγματος χάριν με τον προαναφερθέντα τρόπο. Αυτό μας οδηγεί σε μια εισαγωγή δέκα μεταβλητών, κάθε μια συνδεδεμένη με ένα διατεταγμένο ζευγάρι χωρικών αντικειμένων. Λαμβάνοντας υπόψη δύο τέτοια αντικείμενα, για παράδειγμα A και B, δηλώνουμε την αντίστοιχη μεταβλητή με $x_{A,B}$. Κατόπιν οι ακόλουθες εκφράσεις περιοχών τυποποιούν την περιγραφή του προβλήματος:

- $x_{H1,G1}$ E {inside, coveredby},
- $x_{H2.G2}$ E {inside},
- $x_{H1,H2}$ E {disjoint},
- $x_{H1,R}$ E{meet},
- $x_{H2,R}$ E {meet},
- $x_{G1,G2}$ E {disjoint, meet},
- $x_{H1,G2}$ E {disjoint, meet},
- $x_{G1,H2}$ E {disjoint, meet},
- $x_{G1,R}$ E RCC8,
- $x_{G2,R}$ E RCC8.

Επιπλέον, για κάθε διατεταγμένη τριάδα A, B, C των εξεταζόμενων αντικειμένων έχουμε έναν περιορισμό στις μεταβλητές $x_{A,B}$, $x_{B,C}$, $x_{A,C}$ με τις αντίστοιχες περιοχές $D_{A,B}$, $D_{B,C}$, $D_{A,C}$ που ορίζεται από

$$C_{A,B,C} := S_3 \cap (D_{A,B} \times D_{B,C} \times D_{A,C}).$$

Δεδομένου ότι έχουμε πέντε αντικείμενα, έχουμε συνολικά δέκα περιορισμούς.

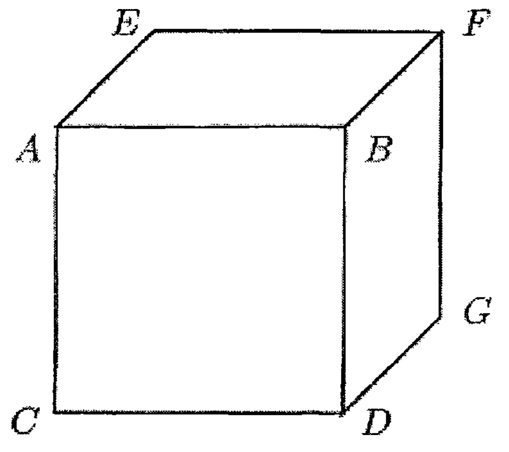
Αυτό συμπληρώνει την περιγραφή του αρχικού μας προβλήματος ως ένα CSP. Σημειώστε ότι σε αυτήν την διαμόρφωση 'αιχμαλωτίσαμε' επίσης μερικές υποθέσεις κοινής αίσθησης οι οποίες υπονοούνται. Παραδείγματος χάριν, υποθέσαμε ότι και τα δύο σπίτια είναι διαχωρισμένα ακόμη κι αν και αυτές οι πληροφορίες δεν δίνονται με σαφήνεια. Ομοίως, υποθέσαμε ότι και οι δύο κήποι είτε **διαχωρίζονται** είτε **ενώνονται**. Αντίθετα, δεν υποθέσαμε τίποτα για τη σχέση μεταξύ των κήπων και του δρόμου. Για να λύσουμε αυτό το πρόβλημα αρκεί να ορίσουμε το σύνολο τιμών που η μεταβλητή x_{G2R}, μπορεί να πάρει σε όλες τις λύσεις σε αυτό το CSP.

Παράδειγμα 2.11 Ανάλυση των Πολύεδρων Εικόνων

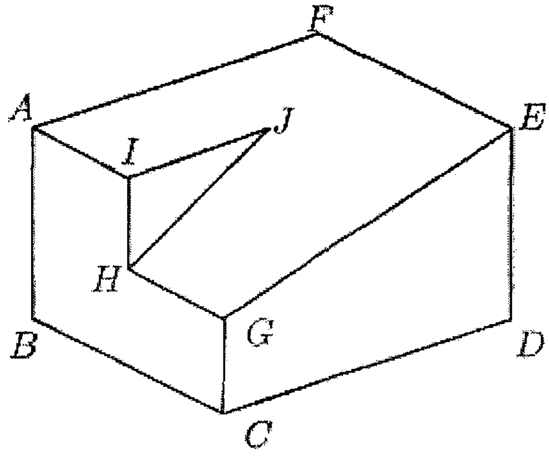
Θεωρήστε τώρα δύο σχήματα (δηλαδή δύο διαστατικά σχέδια) που αντιπροσωπεύουν τα αντικείμενα που διαμορφώνονται από ευθείες πλευρές, τα οποία δίνονται στα Σχήματα 2.13 και 2.14.

Είναι σαφές ότι η εικόνα που δίνεται στο Σχήμα 2.13 πράγματι αντιπροσωπεύει έναν κύβο ενώ αυτή που δίνεται στο Σχήμα 2.14 δεν αντιστοιχεί σε οποιοδήποτε πιθανό τρισδιάστατο αντικείμενο. Ενδιαφερόμαστε εδώ για την ανάλυση των εικόνων που διαμορφώνονται από ευθείες πλευρές και θα επιθυμούσαμε να ξέρουμε ποιες εικόνες δέχονται τρισδιάστατη ερμηνεία

,η οποία αντιστοιχεί σε μια συλλογή πιθανών τρισδιάστατων αντικειμένων. Κάθε τέτοιο αντικείμενο είναι ένα τρισδιάστατο **πολύεδρο**.



Σχήμα 2.13. Ένας κύβος

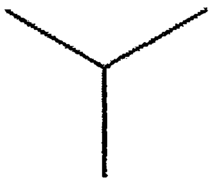


Σχήμα 2.14. Ένα ακατόρθωτο αντικείμενο

Έρευνα πάνω στον οραματισμό των υπολογιστών που πραγματοποιήθηκε στη δεκαετία του '60 και του '70, επικεντρώθηκε στην εύρεση των κανόνων που θα μπορούσαν να μας επιτρέψουν να ορίσουμε ποιες σκηνές δέχονται τρισδιάστατη ερμηνεία. Αυτή η εργασία οδήγησε σε έναν προσδιορισμό τεσσάρων τύπων συνδέσεων που μπορούν να δημιουργηθούν στις εικόνες που αντιστοιχούν στα πιθανά αντικείμενα, στα οποία όλες οι συνδέσεις έχουν είτε δύο είτε τρεις πλευρές:

- η σύνδεση L, που απεικονίζεται από:

- η σύνδεση fork, που απεικονίζεται από:



- η σύνδεση T, που απεικονίζεται από:

- και η σύνδεση arrow, που απεικονίζεται από:

Στη συνέχεια περιορίζουμε την προσοχή μας στις εικόνες που αποτελούνται από τέτοιες συνδέσεις. Για να διευκρινίσουμε τη θέση των πλευρών που διαμορφώνουν μια εικόνα, ορίζουμε σε κάθε ένα από αυτά μια ετικέτα. Κάθε πλευρά προκύπτει από τη διασταύρωση δύο επιπέδων και η ετικέτα παρέχει τις πληροφορίες για τη σχετική θέση αυτών των δύο επιπέδων. Χρησιμοποιούμε τις ακόλουθες τέσσερις ετικέτες:

- +, για να χαρακτηρίσουμε τις **κυρτές** πλευρές,
- —, για να χαρακτηρίσουμε τις **κοίλες** πλευρές,
- τα βέλη, → και ←, για να χαρακτηρίσουμε τις **άκρες** ορίου.

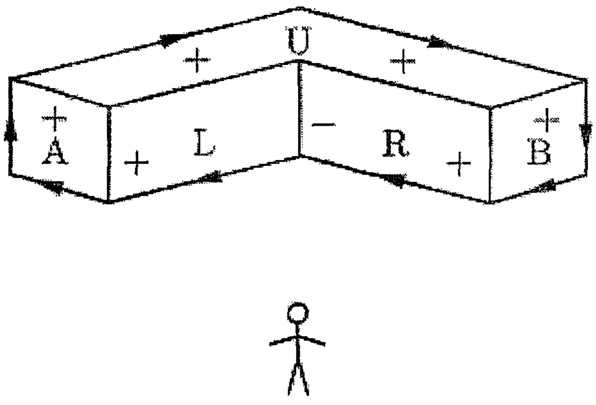
Καλούμε εδώ μια πλευρά

- **κυρτή** εάν χρειάζονται 270 μοίρες για να περιστραφεί ένα αεροπλάνο επάνω σε άλλο μέσω του μέρους του διαστήματος που περιέχει τον θεατή,
- **κοίλη** εάν χρειάζονται 90 μοίρες για να περιστραφεί ένα αεροπλάνο επάνω σε άλλο μέσω του μέρους του διαστήματος που περιέχει το θεατή,
- **όριο** εάν διαμορφώνεται με δύο αεροπλάνα ένα από τα οποία είναι κρυμμένο.

Για να διευκρινίσουμε αυτήν την ορολογία θεωρήστε την εικόνα που απεικονίζεται στο Σχήμα 2.15. Από την σκοπιά του απεικονισμένου ατόμου η πλευρά κατά τη διασταύρωση των περιοχών A και L σημειώνεται με +, δεδομένου ότι είναι μια κυρτή πλευρά. Πράγματι, χρειάζονται 270 μοίρες για να περιστραφεί η μια περιοχή επάνω στην άλλη μέσω του μέρους του διαστήματος που περιέχει το απεικονισμένο άτομο. Για τον ίδιο λόγο οι πλευρές κατά τη διασταύρωση των περιοχών R και B, L και U και R και U σημειώνονται με +. Αντίθετα, η πλευρά κατά τη διασταύρωση των περιοχών L και R από σημειώνεται με —, δεδομένου ότι είναι μια κοίλη πλευρά. Πράγματι, χρειάζονται 90 μοίρες για να περιστραφεί η μια περιοχή επάνω στην άλλη μέσω του μέρους του διαστήματος που περιέχει το απεικονισμένο άτομο.

Τέλος, οι εξωτερικές πλευρές σημειώνονται με βέλη, δεδομένου ότι είναι πλευρές ορίου. Εδώ και αλλού η κατεύθυνση των βελών επιλέγεται κατά τέτοιο τρόπο ώστε όταν τα ακολουθούμε η εικόνα να βρίσκεται στη δεξιά πλευρά.

Γενικά, οι πλευρές δεν είναι απαραίτητο να έχουν ένα μοναδικό μαρκάρισμα ορισμένο σ' αυτή.



Σχήμα 2.15. Τρεις τύποι ακρών

. Παραδείγματος χάριν, στην εικόνα που δίνεται στο Σχήμα 2.13 οι πλευρές AC και AE μπορούν να είναι είτε κυρτές είτε κοίλες. Η πρώτη δυνατότητα αντιστοιχεί στην κατάσταση στην οποία ο κύβος "επιπλέει" στον αέρα.

Η δεύτερη δυνατότητα αντιστοιχεί στην κατάσταση στην οποία ο κύβος είναι "κολλημένος" σε έναν τοίχο στην αριστερή πλευρά. Σε αυτήν την περίπτωση η πλευρά AC βρίσκεται στη διασταύρωση του αεροπλάνου που ορίζεται από το τετράγωνο ABDC και του τοίχου και η πλευρά AE βρίσκεται στη διασταύρωση του επιπέδου που ορίζεται από το τετράγωνο AEFB και του τοίχου. Έτσι σύμφωνα με την παραπάνω ορολογία αυτές οι δύο πλευρες είναι κοίλες.

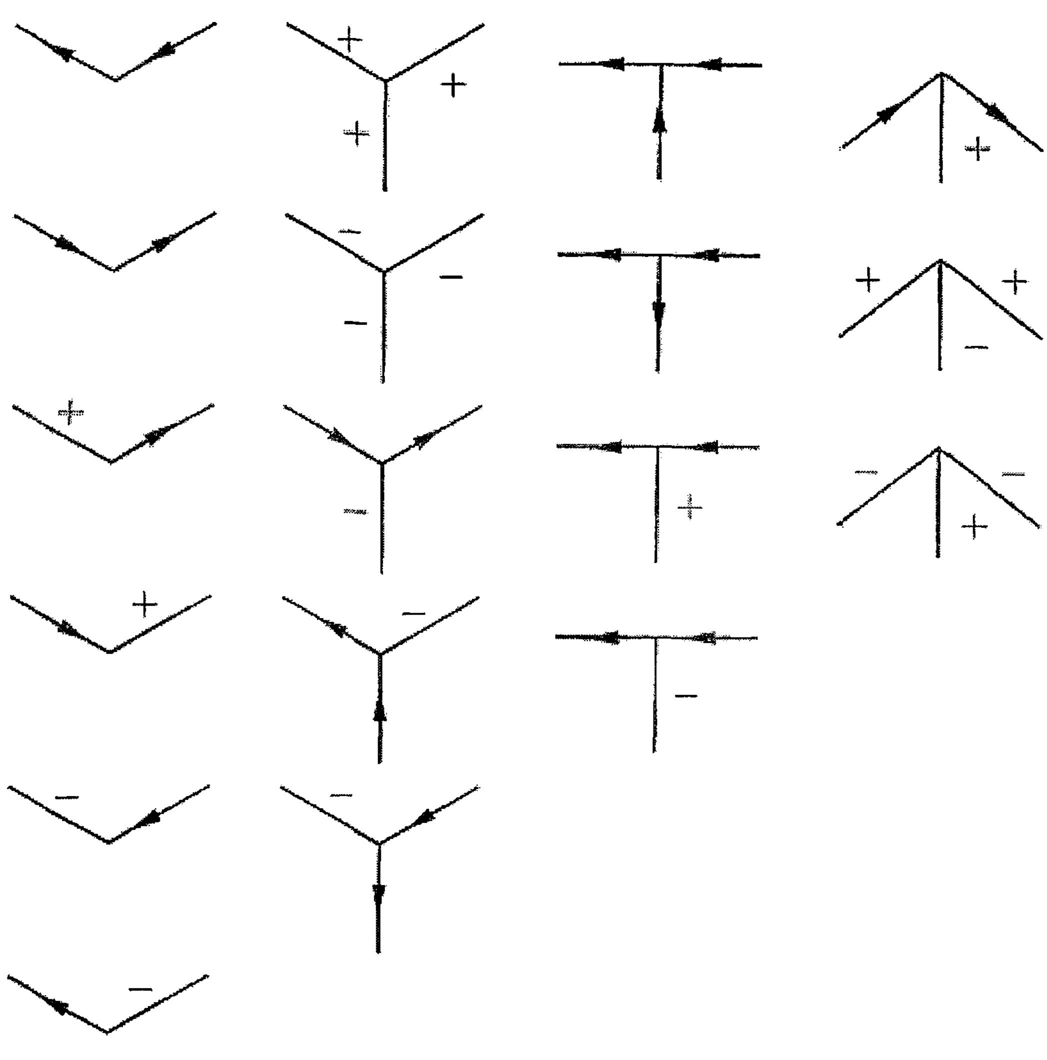
Περαιτέρω, η πλευρά AB μπορεί να είναι σε γενικές γραμμές είτε κυρτή είτε κοίλη. Δηλαδή, εάν ερμηνεύσουμε αυτήν την εικόνα ως κύβο έπειτα η πλευρά AB είναι απαραιτήτως κυρτή. Αφ' ετέρου, εάν ερμηνεύσουμε αυτή την εικόνα ως μέρος ενός κιβωτίου με το κατώτατο σημείο ABDC και τις πλευρές να συνδέονται με τις χαμένες πλευρές AE και DG, τις οποίες εξετάζουμε παραπάνω, τότε η πλευρά AB είναι κοίλη. (Σημειώστε ότι η δεύτερη ερμηνεία δεν αντιστοιχεί σε οποιοδήποτε πιθανό αντικείμενο. Παραδείγματος χάριν σε αυτήν την περίπτωση λείπει μια fork σύνδεση που συνδέει το C, το E και το G.)

Οι Clowes [1971] και Huffman [1971] ανεξάρτητα διαπίστωσαν ότι μια εικόνα που διαμορφώνεται από τους τέσσερις τύπους συνδέσεων που ήδη αναφέραμε αντιπροσωπεύει ένα πιθανό αντικείμενο ακριβώς όταν οι πλευρές του μπορούν να σημαδευτούν κατά τέτοιο τρόπο ώστε να χρησιμοποιηθούν μόνο οι μαρκαρισμένες συνδέσεις που απαριθμούνται στο Σχήμα 2.16. Παραδείγματος χάριν, η εικόνα που δίνεται στο Σχήμα 2.13 αντιπροσωπεύει ένα πιθανό αντικείμενο, δηλαδή έναν κύβο, και τέσσερα σωστά μαρκαρίσματα των πλευρών του παρουσιάζονται στο Σχήμα 2.17.

Στη συνέχεια, είναι πιθανόν να επαληθευτεί ότι δεν υπάρχει κανένα τέτοιο μαρκάρισμα των πλευρών για την εικόνα που δίνεται στο Σχήμα 2.14. Στη συνέχεια παρέχουμε δύο διαμορφώσεις του εξεταζόμενου προβλήματος από την σκοπιά των CSPs.

Πρώτη απεικόνιση

Σε αυτήν την απεικόνιση οι μεταβλητές συμβολίζουν τις συνδέσεις. Έτσι έχουμε τέσσερις τύπους μεταβλητών, ανάλογα με τον τύπο της σύνδεσης: τη μεταβλητή L ,

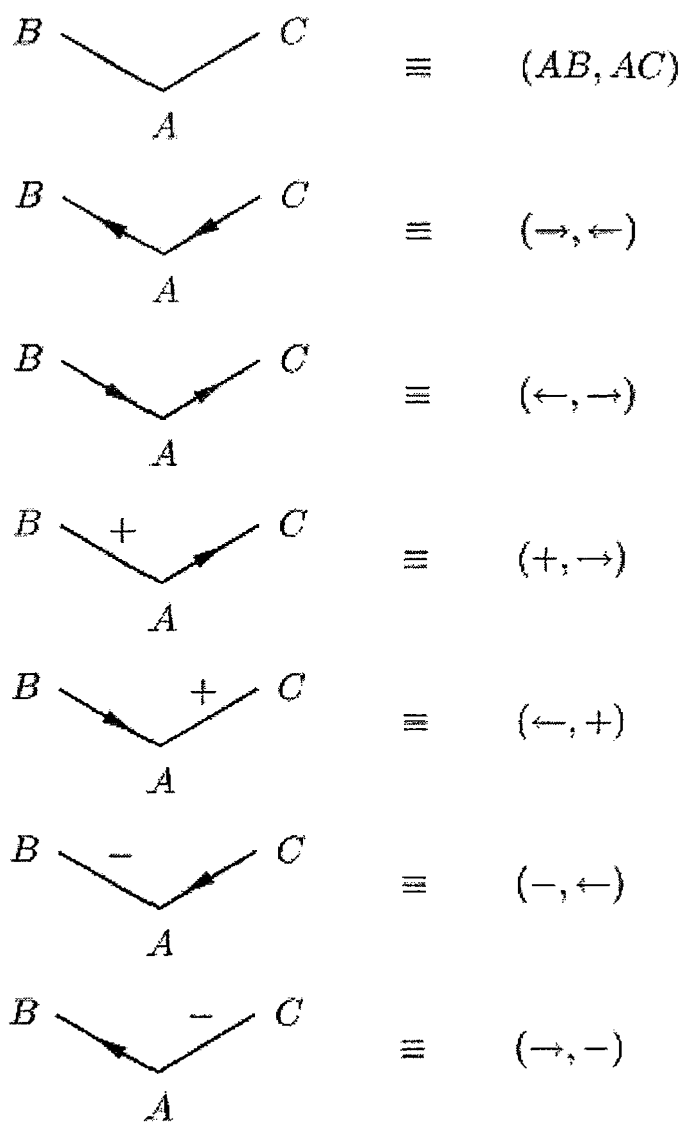


Σχήμα 2.16. Επιτρεπόμενες συνδέσεις

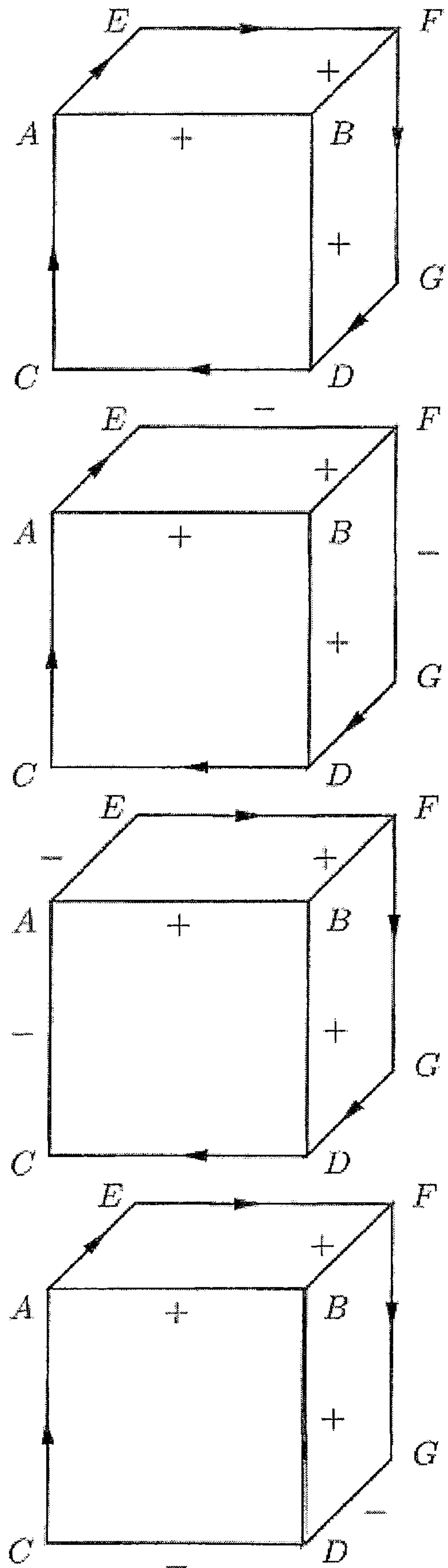
τη μεταβλητή fork, τη μεταβλητή T και τη μεταβλητή arrow. Η περιοχή της μεταβλητής L αποτελείται από τα έξι μαρκαρίσματα της σύνδεσης L που δίνεται στο Σχήμα 2.16, και ομοίως για τις άλλες τρεις μεταβλητές. Για να παρουσιάσουμε αυτά τα μαρκαρίσματα σε μορφή κειμένου εισάγουμε τέσσερα στοιχεία, +, —, $\rightarrow$, $\leftarrow$, και χρησιμοποιούμε τους πίνακες μετάφρασης από τις σημαδεμένες συνδέσεις σε μία μορφή κειμένου που παρουσιάζεται στα Σχήματα 2.18, 2.19, 2.20, και 2,21.

Έτσι η περιοχή της μεταβλητής L είναι ίση με

$$\{(\rightarrow, \leftarrow), (\leftarrow, \rightarrow), (+, \rightarrow), (\leftarrow, +), (-, \leftarrow), (\rightarrow, -)\}.$$



Σχήμα 2.17 Μετάφραση σύνδεσης L



Σχήμα 2.17. Τέσσερα σωστά μαρκαρίσματα ενός κύβου

Ομοίως, η περιοχή της μεταβλητής fork είναι ίση με

$$\{(+,+,+),(-,-,-),(\rightarrow,\leftarrow,-),(-,\rightarrow,\leftarrow),(\leftarrow,-,\rightarrow)\},$$

η περιοχή της μεταβλητής T είναι ίση μ

$$\{(\rightarrow,\leftarrow,\leftarrow),(\rightarrow,\leftarrow,\rightarrow),(\rightarrow,\leftarrow,+),(\rightarrow,\leftarrow,-)\},$$

και η περιοχή της μεταβλητής arrow είναι ίση με

$$\{(\leftarrow,\rightarrow,+),(+,+,-),(-,-,+)\}.$$

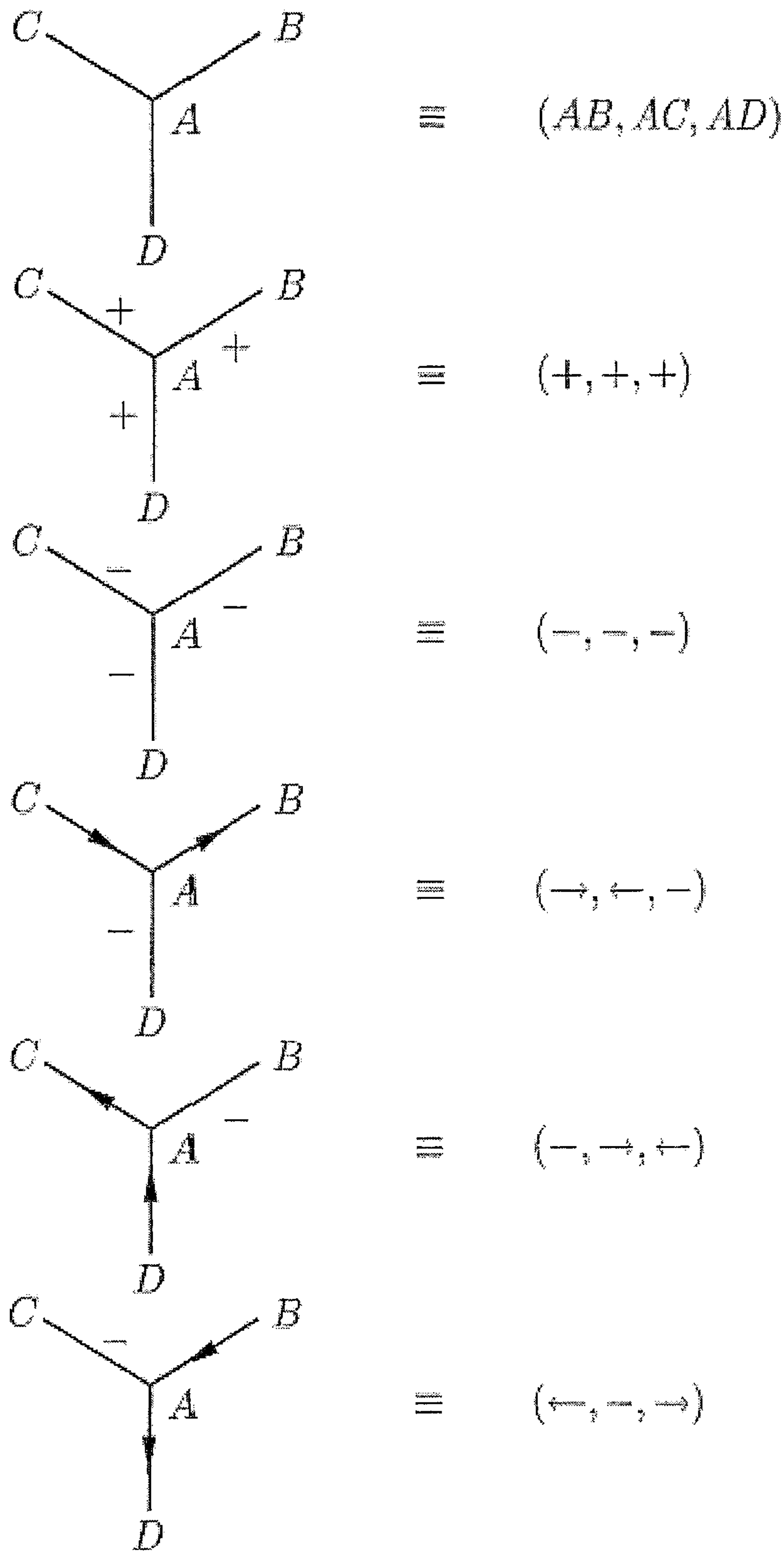
Οι περιορισμοί 'αιχμαλωτίζουν' την πληροφορία ότι οι συνδέσεις μοιράζονται τις πλευρές. Παραδείγματος χάριν, στη εικόνα που δίνεται στο Σχήμα 2.13 οι συνδέσεις A και B μοιράζονται την πλευρά AB. Αυτό περιορίζει τις πιθανές τιμές που χρησιμοποιούνται για τις συνδέσεις A και B με παρόμοιο τρόπο όπως η διασταύρωση δύο λέξεων περιόρισε τις πιθανές τιμές στο γρίφο σταυρόλεξων του Παραδείγματος 2.8. Η διαφορά εδώ είναι ότι μια διασταύρωση, που απεικονίζεται από μια πλευρά, είναι προσανατολισμένη.

Παραδείγματος χάριν, για τη σύνδεση A η πλευρά AE χρησιμοποιείται (ως δεύτερο επιχείρημα), ενώ για τη σύνδεση E η πλευρά EA χρησιμοποιείται (επίσης ως δεύτερο επιχείρημα). Έτσι εάν η πλευρά AE της σύνδεσης A χαρακτηρίζεται από το +, κατόπιν έτσι είναι και η πλευρά EA της σύνδεσης E , και αντίστοιχα για το —. Εντούτοις, εάν η πλευρά AE χαρακτηρίζεται από το→, τότε η πλευρά EA χαρακτηρίζεται από το ←, και ανάλογα για το ←. Αφού λάβουμε υπόψη μας αυτό το γεγονός μπορεί εύκολα κανείς να ελέγξει ότι ο αντίστοιχος περιορισμός C_{AE} στις συνδέσεις A και E αποτελείται από το ακόλουθο σύνολο τεσσάρων ζευγαριών:

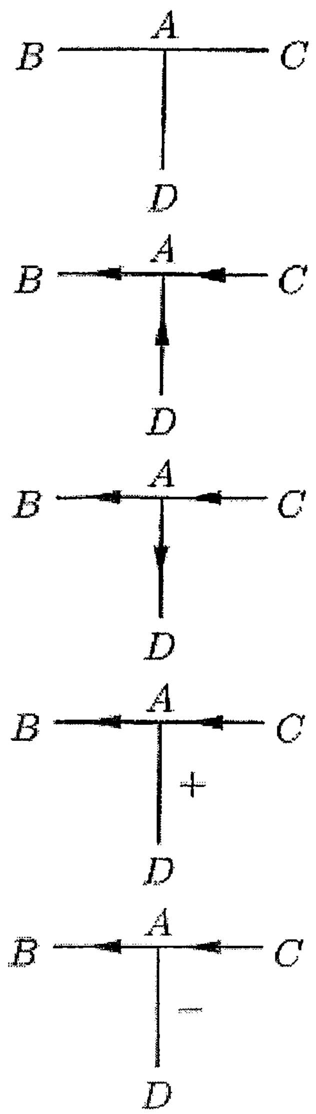
$$\begin{aligned} C_{AE} := \{&((\leftarrow,\rightarrow,+),(\rightarrow,\leftarrow)),((\leftarrow,\rightarrow,+),(-,\leftarrow)),\\ &((+,+,-),(\leftarrow,+)),((-,-,+),(\rightarrow,-))\}. \end{aligned}$$

Κάθε ζευγάρι, παραδείγματος χάριν ((←,→, +), (—,←)), αντιπροσωπεύει ένα νόμιμο μαρκάρισμα των πλευρών των συνδέσεων A και E. Γενικά, κάθε πλευρά συμβάλλει σε έναν, δυαδικό, περιορισμό. Έτσι η εικόνα που δίνεται στο Σχήμα 2.13 τυποποιείται από ένα CSP που αποτελείται από εννέα περιορισμούς, ενώ η εικόνα που δίνεται στο Σχήμα 2.14 τυποποιείται από ένα CSP που αποτελείται από δεκατρείς περιορισμούς.

Κάθε λύση σε ένα τέτοιο CSP ορίζει τις ετικέτες που ορίζονται σε κάθε πλευρά. Η επιλογή των μεταβλητών περιοχών και των περιορισμών εξασφαλίζει ότι όλες οι συνδέσεις χαρακτηρίζονται από έναν νόμιμο τρόπο, δηλαδή όλες προέρχονται από το Σχήμα 2.16.



Σχήμα 2.19. Μετάφραση για την fork σύνδεση



$$\equiv \quad (AB, AC, AD)$$

$$\equiv \quad (\rightarrow, \leftarrow, \leftarrow)$$

$$\equiv \quad (\rightarrow, \leftarrow, \rightarrow)$$

$$\equiv \quad (\rightarrow, \leftarrow, +)$$

$$\equiv \quad (\rightarrow, \leftarrow, -)$$

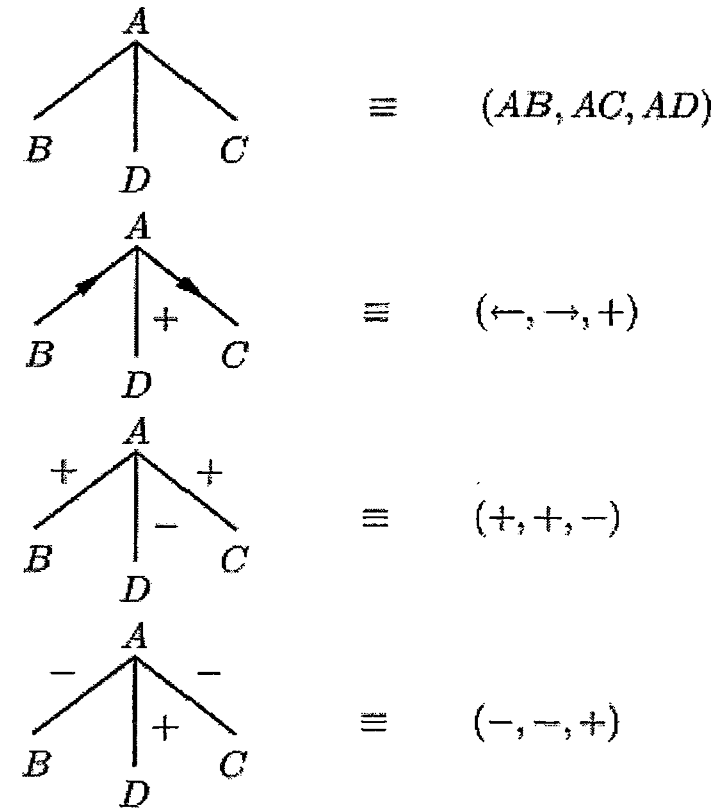
Σχήμα 2.20. Μετάφραση για την T σύνδεση

Δεύτερη απεικόνιση

Σε αυτήν την απεικόνιση οι μεταβλητές απεικονίζουν τις πλευρές. Η περιοχή κάθε μεταβλητής είναι ίση με (+, —,→, ←) το σύνολο των χρησιμοποιούμενων ετικετών. Στη συνέχεια, οι περιορισμοί απεικονίζουν τις συνδέσεις. Έτσι έχουμε τέσσερεις τύπους περιορισμών: L, fork, T και arrow, καθένας ορισμένος από την αντίστοιχη λίστα στο Σχήμα 2.16.

Παραδείγματος χάριν έχουμε

$$L := \{(\rightarrow, \leftarrow), (\leftarrow, \rightarrow), (+, \rightarrow), (\leftarrow, +), (-, \leftarrow), (\rightarrow, -)\}.$$



Σχήμα 2.21. Μετάφραση για τη σύνδεση arrow

Κάθε εικόνα τυποποιείται με τη βοήθεια των περιορισμών που απεικονίζουν τις συνδέσεις της. Παραδείγματος χάριν, η εικόνα που δίνεται στο Σχήμα 2.13 τυποποιείται από τους ακόλουθους επτά περιορισμούς, όπου για κάθε περιορισμό γράφουμε συγκεκριμένα σε ποιες άκρες και ποια σειρά χρησιμοποιείται:

arrow (AC , AE, AB),
fork (BA, BF, BD),
L (CA, CD),
arrow (DG, DC, DB),
L (EF, EA),
arrow (FE, FG, FB),
L (GD, GF).

Έτσι το arrow (AC, AE, AB) δείχνει το γεγονός ότι βλέπουμε τη σύνδεση A ως περιορισμό arrow στις μεταβλητές AC, AE και AB και αντίστοιχα με τις άλλες έξι συνδέσεις. Για να ολοκληρώσουμε τη διαμόρφωση, όπως στην περίπτωση της πρώτης απεικόνισης, πρέπει να συσχετίζουμε κάθε πλευρά με την αντίστροφη της, παραδείγματος χάριν στην περίπτωση του Σχήματος 2.13 την πλευρά AE με την πλευρά EA.

Αυτό απαιτείται επειδή εάν μια πλευρά , για παράδειγμα η AE του Σχήματος 2.13,χαρακτηρίζεται από το —, τότε εισάγει τον περιορισμό arrow (AC, AE, AB) ως $\rightarrow$, και τον περιορισμό L (EF, EA) ως $\leftarrow$ (στο EA), και ομοίως για το μαρκάρισμα με $\leftarrow$. Για αυτόν τον σκοπό χρησιμοποιούμε έναν δυαδικό περιορισμό, τον edge, ο οποίος έχει τη συμπληρωματική μορφή των τιμών $\rightarrow$ και $\leftarrow$. Ορίζεται με

$$edge := \{(+,+),(-,-),(\rightarrow,\leftarrow),(\leftarrow,\rightarrow)\}.$$

Αυξάνουμε τώρα την απεικόνιση κάθε εικόνας σύμφωνα με τους περιορισμούς edge για όλα τα ζεύγη πλευρών που είναι αντίστροφες η μία με την άλλη. Στην περίπτωση της εικόνας του Σχήματος 2.13 αυξάνουμε τους ήδη επτά ταξινομημένους περιορισμούς στους ακόλουθους εννέα περιορισμούς:

edge (AB, BA),
edge (AC, CA),
edge (CD, DC),
edge (BD, DB),
edge (AE, EA),
edge (EF, FE),
edge (BF, FB),
edge (FG, GF),
edge (DG,GD).

Έτσι σε αυτήν την απεικόνιση χρησιμοποιούνται και οι δυαδικοί και οι τριαδικοί περιορισμοί. Οι δυαδικοί περιορισμοί είναι οι edge και οι L περιορισμοί, ενώ οι τριαδικοι περιορισμοί είναι οι fork και οι arrow περιορισμοί. Όπως στην πρώτη απεικόνιση, κάθε λύση στο CSP η οποία τυποποιεί μια σεβαστή εικόνα παρέχει ένα σταθερό μαρκάρισμα των πλευρών της εικόνας, υπό την έννοια ότι όλες οι επονομαζόμενες συνδέσεις προέρχονται από το Σχήμα 2.16.

2.6 Προβλήματα περιορισμένης βελτιστοποίησης

Ο γενικός στόχος της περιορισμένης βελτιστοποίησης είναι να βρεθούν οι βέλτιστες λύσεις σε ένα σύνολο περιορισμών υπαγόμενες σε κάποια αντικειμενική λειτουργία obj. Συγκεκριμένα, θεωρήστε ένα CSP P := $\langle$ C; x_1 E D_1,..., xn E D_n $\rangle$ μαζί με μια λειτουργία

$$\text{obj: Sol} \rightarrow \text{R}$$

από το σύνολο όλων των λύσεων Sol αναφορικά με το P στο σύνολο των πραγματικών αριθμών R. (Συνήθως η λειτουργία obj ορίζεται σε όλες τις ακολουθίες d E D_1x...xD_n αλλά αυτή η υπόθεση δεν απαιτείται εδώ.) Ενδιαφερόμαστε για την εύρεση μιας λύσης d αναφορικά με το P για το οποίο η τιμή obj(d) είναι βέλτιστη. Γενικά υποθέτουμε ότι "βέλτιστο" σημαίνει "μέγιστο". Καλούμε έπειτα το ζευγάρι (P, obj) **πρόβλημα περιορισμένης βελτιστοποίησης.** Ο τομέας της περιορισμένης βελτιστοποίησης είναι τεράστιος καθώς και ο αριθμός των βιβλίων που γράφτηκαν σε αυτό το θέμα. Εδώ περιοριζόμαστε σε μια παρουσίαση τριών παραδειγμάτων, τα πρώτα δύο από τα οποία είναι κλασικά παραδείγματα των NP-ολοκληρωμένων προβλημάτων.

Παράδειγμα 2.12 Το πρόβλημα των Σακιδίων.

Μας δίνονται n αντικείμενα , καθένα με έναν όγκο και μια τιμή, και ένα σακίδιο σταθερού όγκου. Το πρόβλημα είναι να βρεθεί η συλλογή των αντικειμένων με τη μέγιστη συνολική τιμή που ταιριάζει στο σακίδιο. Συγκεκριμένα, έχουμε n αντικείμενα με όγκους $a_1,\dots,a_n$ και τιμές $b_1,\dots,b_n$ και το σακίδιο με όγκο u. Περαιτέρω, έχουμε n μεταβλητές $x_1,..,x_n$, κάθε μια με την περιοχή {0,1}. Η συμπερίληψη του αντικειμένου i σε μια συλλογή διαμορφώνεται θέτοντας τη τιμή xi ως 1. Η αξίωση ότι η συλλογή ταιριάζει στο σακίδιο μεταφράζεται στον περιορισμό

$$\sum_{i=1}^{n} a_i \cdot x_i \leq v$$

με τις μεταβλητές $x_1,\dots,x_n$. Επιδιώκουμε μια λύση σε αυτόν τον περιορισμό για τον οποίο το άθροισμα

$$\sum_{i=1}^{n} b_i \cdot x_i$$

είναι μέγιστο. Δηλαδή, το άθροισμα $\sum_{i=1}^{n} bixi$ είναι η αντικειμενική λειτουργία.

Παράδειγμα 2.13 Ένα πρόβλημα Νομισμάτων.

Εξετάστε το ακόλουθο πρόβλημα.

Ποιος είναι ο ελάχιστος αριθμός νομισμάτων που επιτρέπει σε κάποιον να πληρώσει οποιοδήποτε ποσό είναι μικρότερο από ένα ευρώ ακριβώς; Θυμηθείτε ότι υπάρχουν έξι διαφορετικά νομίσματα σε λεπτά(υποδιαίρεση το Ε), αξίας 1, 2, 5, 10, 20, 50.

Το διατυπώνουμε ως περιορισμένο πρόβλημα βελτιστοποίησης. Για αυτόν τον σκοπό ορίζουμε αρχικά τις μεταβλητές και τις περιοχές τους. Δεδομένου ότι η ερώτηση αναφέρεται στον αριθμό των νομισμάτων, είναι φυσικό να εφαρμοστούν οι μεταβλητές που αντιπροσωπεύουν τα επιλεγμένα ποσά κάθε νομίσματος. Αυτό μας φέρνει σε έξι μεταβλητές που τις δηλώνουμε με x_1, x_2, x_5, x_{10}, x_{20}, x_{50}. Η ιδέα είναι ότι κάθε μεταβλητή x δείχνει το επιλεγμένο ποσό νομισμάτων τιμής i. Δεδομένου ότι πρέπει να είμαστε σε θέση να καταβάλουμε ακριβώς οποιοδήποτε ποσό μέχρι και_των 99 λεπτών, καταλήγουμε με τις ακόλουθες εκφράσεις περιοχών:

x_1 Ε [0..99], x_2 Ε [0..49], x_5 Ε [0..19], x_{10} [0..9], x_{20} Ε [0..4], x_{50} Ε[0..1].

Έπειτα, δηλώνουμε τους κατάλληλους περιορισμούς. Λαμβάνοντας υπόψη ότι i Ε [1..99] διατυπώνουμε έναν περιορισμό δηλώνοντας ότι το ποσό i σε λεπτά μπορεί να πληρωθεί ακριβώς χρησιμοποιώντας τα επιλεγμένα νομίσματα. Για αυτόν τον σκοπό χρησιμοποιούμε τις existentially ποσοτικοποιημένες μεταβλητές που κυμαίνονται πέρα από τα διαστήματα ακεραίων:

$$\exists i_1 \in [0..x_1]\ \exists i_2 \in [0..x_2]\ \exists i_5 \in [0..x_5]\ \exists i_{10} \in [0..x_{10}]$$

$$\exists i_{20} \in [0..x_{20}]\, \exists i_{50} \in [0..x_{50}]\; i_1 + 2i_2 + 5i_5 + 10i_{10} + 20i_{20} + 50i_{50} = i.$$

Έτσι χρησιμοποιούμε συνολικά 99 περιορισμούς. Σαφώς, για κάθε λύση στο διατυπωμένο CSP το ποσό

$$x_1 + x_2 + x_5 + x_{10} + x_{20} + x_{50}$$

αντιπροσωπεύει το συνολικό ποσό νομισμάτων που θα επιθυμούσαμε να ελαχιστοποιήσουμε. Δεδομένου ότι στη διαδικασία που ακολουθούμε επιδιώκουμε λύσεις με μια μέγιστη τιμή της αντικειμενικής λειτουργίας, χρησιμοποιούμε το ποσό

$$-(x_1 + x_2 + x_5 + x_{10} + x_{20} + x_{50})$$

ως αντικειμενική λειτουργία.

Ένα μειονέκτημα της προτεινόμενης παρουσίασης του προβλήματος, είναι ότι οι περιορισμοί περιλαμβάνουν τις existentially ποσοτικοποιημένες μεταβλητές. Τέτοιοι περιορισμοί είναι δύσκολοι ως προς τον χειρισμό. Έτσι τώρα παρέχουμε μια διαφορετική παρουσίαση που περιλαμβάνει απλούστερους περιορισμούς.

Εκτός από τις ήδη εξεταζόμενες μεταβλητές x_1, x_2, x_5, x_{10}, x_{20}, x_{50} εισάγουμε για κάθε i Ε [1..99] έξι μεταβλητές x_1^i, x_2^i, x_5^i, x_{10}^i, x_{20}^i, x_{50}^i αντιστοίχως με τις ίδιες περιοχές x_1, x_2, x_5, x_{10}, x_{20} και x_{50}. Έτσι τώρα έχουμε 600 μεταβλητές αντί για έξι. Εντούτοις οι περιορισμοί γίνονται αρκετά απλούστεροι. Δηλαδή για κάθε i Ε [1..99] χρησιμοποιούμε τον ακόλουθο περιορισμό:

$$x_1^i + 2x_2^i + 5x_5^i + 10x_{10}^i + 20x_{20}^i + 50x_{50}^i = i$$

που δηλώνει ότι το ποσό i Ε [1..99] μπορεί να εξοφληθεί χρησιμοποιώντας τις παραπάνω μεταβλητές και συγκεκριμένα την x_1^i για νομίσματα του 1 λεπτού, την x_2^i για νομίσματα των 2 λεπτών, την x_5^i για νομίσματα των 5 λεπτών, την x_{10}^i για νομίσματα των 10 λεπτών, την x_{20}^i για νομίσματα των 20 λεπτών και την x_{50}^i για νομίσματα των 50 λεπτών.

Επιπλέον προσθέτουμε τους περιορισμούς δηλώνοντας ότι για κάθε i Ε [1...99] τα ποσά x_2^i είναι μικρότερα από τα αντίστοιχα x_j. Ακριβέστερα για κάθε i Ε [1..99] χρησιμοποιούμε τον ακόλουθο περιορισμό:

$$x_j^i \leq x_j.$$

Αυτοί οι 594 περιορισμοί ανισότητας εξασφαλίζουν ότι κάθε ποσό i λεπτών μπορεί επίσης να εξοφληθεί χρησιμοποιώντας τη συλλογή που αντιπροσωπεύεται από τις μεταβλητές x_j. Τέλος, χρησιμοποιούμε την ιδία αντικειμενική λειτουργία όπως στην προηγουμένη παρουσίαση.

Για τον ενδιαφερόμενο αναγνώστη η απάντηση είναι οκτώ νομίσματα και τα αντίστοιχα ποσά νομισμάτων αξίας 1, 2, 5, 10, 20, 50 σε έναν τέτοιο συνδυασμό είναι 1, 2, 1, 1, 2, 1.

Παράδειγμα 2.14. Κανόνας Golomb

Ένας **κανόνας Golomb με m** σημεία είναι μια διατεταγμένη ακολουθία φυσικών αριθμών m έτσι ώστε η απόσταση μεταξύ δύο οποιωνδήποτε στοιχείων σε αυτήν την ακολουθία να είναι μοναδική. Το μεγαλύτερο στοιχείο ενός κανόνα Golomb καλείται **length. Ένας βέλτιστος κανόνας Golomb με m** σημεία είναι ένας κανόνας Golomb με τα σημάδια μ έχοντας ένα ελάχιστο μήκος. Παραδείγματος χάριν οι αριθμοί

0,1,4,9,11

είναι ένας κανόνας Golomb με 5 σημεία. Πράγματι, οι αποστάσεις μεταξύ των στοιχείων σε αυτές τις ακολουθίες είναι:

- για τα σημεία one part: 1,3,5,2,
- για τα σημεία two part: 4,8,7,
- για τα σημεία three part: 9,10,
- για τα σημεία four part: 11,

και όλες αυτές οι τιμές είναι διαφορετικές. Κάποιος μπορεί να δείξει ότι οι αριθμοί 0,1,4,9,11 είναι στην πραγματικότητα ένας βέλτιστος κανόνας Golomb με 5 σημεία που υπάρχουν με κάποιους άλλους αριθμούς 0,3,4,9,11.

Για να παρουσιάσουμε το πρόβλημα εύρεσης ενός βέλτιστου κανόνα Golomb με m σημεία ως περιορισμένο πρόβλημα βελτιστοποίησης χρησιμοποιούμε την ακόλουθη ορολογία. Καλούμε δύο αριθμούς ι, j ζεύγος έτσι ώστε $1 \leq i \leq j \leq m$. Λέμε ότι τα ζεύγη i, j και k, l είναι διαφορετικά εάν $i \neq k$ ή $j \neq i$ και διαχωρισμένα εάν $i \neq k$ και $j \neq l$.

Χρησιμοποιούμε έπειτα τις μεταβλητές m, $x_1,\ldots,x_m$, κάθε μια με την περιοχή N, και εφαρμόζουμε τους ακόλουθους περιορισμούς:

- $x_i < x_{i+1}$ για i E [1..m —1],
- $x_j - x_i \neq x_l - x_k$ για όλα τα διαφορετικά ζεύγη i, j και k,l.

Ως αντικειμενική λειτουργία επιλέγουμε απλά την $-x_m$. Κατόπιν ο στόχος μεγιστοποίησης της x_m είναι ισοδύναμος με το στόχο ελαχιστοποίησης της x_m.

Ισχύει μια καλύτερη παρουσίαση σημειώνοντας ότι για i, k < j εάν $x_i \neq x_k$ τότε $x_j - x_i \neq x_j - x_k$, και για i <j , l εάν $x_j \neq x_l$, τότε $x_j - x_i \neq x_l - x_i$. Αυτή η παρατήρηση μας επιτρέπει να αντικαταστήσουμε το δεύτερο σύνολο περιορισμών από ένα μικρότερο:

- $x_j - x_i \neq x_l - x_k$ για όλα τα διαχωρισμένα ζεύγη i j και k, l.

Ισχύει μια ακόμη παρουσίαση εισάγοντας τις βοηθητικές μεταβλητές που αντιπροσωπεύουν τις διαφορές μεταξύ των στοιχείων στην ακολουθία $x_1,\ldots,$ x_m και χρησιμοποιώντας τους περιορισμούς ανισότητας μεταξύ αυτών των βοηθητικών μεταβλητών. Έτσι για κάθε ζεύγος i,j εισάγουμε μία μεταβλητή $z_{i,j}$ που κυμαίνεται πέρα από τους θετικούς φυσικούς αριθμούς και εφαρμόζουμε τους ακόλουθους περιορισμούς:

$z_{i,j} = x_j - x_i$ για κάθε ζεύγος i , j ,

$z_{i,j} \neq z_{k,l}$ για όλα τα διαφορετικά ζεύγη i, j και k,l.

Επιπλέον μπορούμε εκ του ασφαλούς να αντικαταστήσουμε εδώ το "διαφορετικό" από το "διαχωρισμένο". Επίσης μπορούμε να αντικαταστήσουμε τους περιορισμούς ανισότητας από έναν ενιαίο περιορισμό all_different στις μεταβλητές $z_{i,j}$.

Οι κανόνες Golomb διαδραματίζουν έναν σημαντικό ρόλο στη ραδιοεπικοινωνία, στις ακτίνες-x, στην κρυσταλλογραφία, στη θεωρία κωδικοποίησης, στην ραδιοαστρονομία και στην οπτική επικοινωνία. Ο μεγαλύτερος γνωστός βέλτιστος κανόνας Golomb έχει 21 σημεία και μήκους 333.

2.7 Περίληψη

Στόχος αυτού του κεφαλαίου ήταν να εξεταστούν τα διάφορα παραδείγματα των προβλημάτων ικανοποίησης περιορισμών (CSPs). Πρώτα ορίσαμε ακριβώς τις έννοιες ενός περιορισμού και ενός CSP. Κατόπιν παρουσιάσαμε παραδείγματα των CSPs που ταξινομήθηκαν σύμφωνα με τις περιοχές που χρησιμοποιήθηκαν. Συζητήσαμε στη συνέχεια για τα προβλήματα που μπορούν να τυποποιηθούν ως CSPs με τις μεταβλητές να ταξινομούνται σε

- ακέραιους αριθμούς,
- πραγματικούς
- το διάστημα [0..1] που αντιπροσωπεύει τις λογικές τιμές **ψευδές, αληθές**,
- συμβολικές περιοχές, δηλ., μη αριθμητικές περιοχές.

Μελετήσαμε επίσης τα περιορισμένα προβλήματα βελτιστοποίησης για τα οποία ενδιαφερθήκαμε να βρούμε μια βέλτιστη λύση σε ένα CSP σύμφωνα με κάποιο κριτήριο.

Διάφορα παραδείγματα που εξετάστηκαν εδώ είναι τα πιο χρήσιμα στην εξήγηση των τεχνικών του προγραμματισμού περιορισμών. Πιο συγκεκριμένα, χρησιμοποιώντας τον γρίφο SEND + MORE= MONEY του Παραδείγματος 2.1 μπορούμε να διευκρινίσουμε με έναν απλό τρόπο την επίδραση της διάδοσης περιορισμών. Θα επιστρέψουμε σ' αυτό στο Κεφάλαιο 6. Στη συνέχεια, το Πρόβλημα με τις n Βασίλισσες του Παραδείγματος 2.2 είναι ένας τρόπος για να εξηγήσουμε τις διάφορες μορφές αναζήτησης. Θα επιστρέψουμε σ' αυτό στο Κεφάλαιο 5.

Έπειτα, το πρόβλημα διαπίστωσης των μηδενικών των πολυωνύμων που εξετάζονται στο Παράδειγμα 2.5 είναι ένα χαρακτηριστικό πρόβλημα που αχολείται με τις τεχνικές που εξετάζονται στο τέλος του Κεφαλαίου 6. Ο γρίφος σταυρόλεξων του παραδείγματος 2.8 θα μας επιτρέψει να επεξηγήσουμε την έννοια της τοξοειδούς σταθερότητας. Τέλος, οι κανόνες Golomb που εξετάζονται στο Παράδειγμα 2.14 δείχνουν πώς ένα απλό πρόβλημα μπορεί να αναγνωρίσει διάφορες φυσικές διατυπώσεις ως CSP.

Κατά την παρουσίαση αυτών των παραδειγμάτων τονίσαμε το γεγονός ότι κάθε πρόβλημα μπορεί να τυποποιηθεί ως CSP με πολλούς διαφορετικούς τρόπους. Γενικά, είναι δύσκολο να βρούμε ποια αναπαράσταση είναι καλύτερη και οι καλές ιδέες στις αρχές του προγραμματισμού περιορισμών βοηθούν. Κατά την εξέταση των διάφορων εναλλακτικών αναπαραστάσεων ενός προβλήματος ως CSP εστιάσαμε σε απλά παραδείγματα.

3

Προγραμματισμός περιορισμών εν συντομία

Σε αυτό το στάδιο είναι χρήσιμο να αποκτήσουμε μια καλύτερη αίσθηση του προγραμματισμού περιορισμών για τον οποίο κάνουμε λόγο. Θυμηθείτε ότι δηλώσαμε στο Κεφάλαιο 1 πως στον προγραμματισμό περιορισμών η διαδικασία προγραμματισμού περιορίζεται σε μια παραγωγή των περιορισμών και μια λύση αυτών των περιορισμών με τη βοήθεια ειδικών πεδίων ή γενικών μεθόδων. Στόχος αυτού του κεφαλαίου είναι να προσφέρει μια διαισθητική εισαγωγή σε αυτές τις μεθόδους. Στα επόμενα κεφάλαια θα τους συζητήσουμε με έναν πιο λεπτομερή και ακριβή τρόπο.

Για να συζητήσουμε αυτές τις τεχνικές είναι χρήσιμο να τυποποιηθούν οι κατάλληλες έννοιες της ισοδυναμίας μεταξύ των CSPs. Αυτό το κάνουμε στην Παράγραφο 3.1. Κατόπιν στην Παράγραφο 3.2 περιγράφουμε ένα γενικό πλαίσιο που μας επιτρέπει να εξηγήσουμε τα βασικά του προγραμματισμού περιορισμών. Περιλαμβάνει διάφορες διαδικασίες που 'αιχμαλωτίζουν' τις συγκεκριμένες πλευρές του προγραμματισμού περιορισμών. Στη συνέχεια επεξηγούμε αυτές τις διαδικασίες με τη βοήθεια δύο εκτεταμένων παραδειγμάτων. Στην Παράγραφο 3.3 εξετάζουμε τα CSPs Λογικών Τιμών και στην Παράγραφο 3.4 τα προβλήματα περιορισμένης βελτιστοποίησης που περιλαμβάνουν τους πολυωνυμικούς περιορισμούς στα διαστήματα ακέραιων αριθμών.

3.1 Ισοδυναμία των CSPs

Αναφέραμε ήδη στο Κεφάλαιο 1 ότι βλέπουμε τον προγραμματισμό περιορισμών ως διαδικασία μετασχηματισμού των CSPs. Για να περιγράψουμε αυτήν την διαδικασία ακριβέστερα είναι χρήσιμο να τυποποιηθούν διάφορες έννοιες σχετικά με τα CSPs.

Κατ' αρχήν, θυμηθείτε ότι ένας περιορισμός C σε μια ακολουθία μεταβλητών $y_1,\dots,y_k$ με τις αντίστοιχες περιοχές $D_1,\dots,D_k$ που συνδέονται με αυτές είναι ένα υποσύνολο του $D_1,\dots,D_k$. Εάν το C είναι ίσο με το $D_1 \times \dots \times D_k$ και είναι μη-άδειο, κατόπιν λέμε ότι το C είναι **λυμένο.** Υποθέσαμε σε αυτούς τους ορισμούς ότι $k > 0$. Στην περίπτωση ορίου όταν $k = 0$ αναγνωρίσαμε δύο περιορισμούς, που δηλώθηκαν από το $\top$ και $\bot$, που αντιπροσωπεύουν αντίστοιχα τον **αληθή περιορισμό** (παραδείγματος χάριν $0 = 0$) και τον **ψευδή περιορισμό** (παραδείγματος χάριν $0 = 1$).

Καλούμε ένα CSP **λυμένο** εάν όλοι οι περιορισμοί του λύνονται και καμία περιοχή του δεν είναι κενή, και **αποτυχημένο** εάν περιέχει τον ψεύτικο περιορισμό $\bot$ ή μερικές από τις περιοχές του ή τους περιορισμούς του είναι κενοί. Σαφώς, κάθε λυμένο CSP είναι σταθερό και κάθε αποτυχημένο CSP είναι μεταβλητό. Είναι χρήσιμο να σημειωθεί ότι σε ένα CSP

$$\langle C \; ; \; x_1 \in D_1, \ldots, x_n \in D_n \rangle$$

κάθε λύση $(d_1, \ldots, d_n)$αντιστοιχεί σε ένα μοναδικό λυμένο CSP. Αυτό το λυμένο CSP είναι της μορφής

$$\langle C' \; ; \; x_1 \in \{d_1\}, \ldots, x_n \in \{d_n\} \rangle,$$

όπου C΄= {C΄Ι C Ε C}και πού για κάθε περιορισμό C στο C σε μια υπο-ακολουθία $x_{i1},\ldots,x_{im}$ του $x_1,\ldots,x_n$ ο περιορισμός C ορίζεται από το C΄ := $\{(d_{i1},\ldots,d_{im})\}$. Έτσι κάθε περιορισμός στο C είναι ένα ορισμένο μονοσύνολο.

Για να λύσουμε τα CSPs συχνά τα μετασχηματίζουμε με έναν συγκεκριμένο τρόπο μέχρι να βρεθεί μια λύση, αντίστοιχα όλες οι λύσεις, ή είναι σαφές ότι καμία λύση δεν υπάρχει. Τα "τελικά" CSPs που παράγονται από αυτήν την διαδικασία έπειτα είναι λυμένα ή αποτυχημένα. Εάν ένα τελικό CSP είναι λυμένο, παράγει μια λύση σε περίπτωση που όλες οι περιοχές είναι ορισμένα μονοσύνολα, ή περισσότερες λύσεις σε περίπτωση που κάποια περιοχή έχει περισσότερα από ένα στοιχεία. Εάν ένα τελικό CSP είναι αποτυχημένο, δεν παράγει καμία λύση. Οι μετασχηματισμοί των CSPs πρέπει να είναι τέτοιοι ώστε να διατηρείται η ισοδυναμία τους υπό μια κατάλληλη έννοια. Αυτό μας φέρνει στην έννοια της ισοδυναμίας των CSPs. Την ορίζουμε αρχικά για μια ειδική περίπτωση.

Θεωρήστε δύο CSPs P και P με την ίδια ακολουθία μεταβλητών. Λέμε ότι τα P και P είναι **ισοδύναμα** εάν έχουν το ίδιο σύνολο λύσεων. Παραδείγματος χάριν, τα CSPs

$$\langle 3x - 5y = 4 \; ; \; x \in [0..9], y \in [1..8] \rangle$$

και

$$\langle 3x - 5y = 4 \; ; \; x \in [3..8], y \in [1..4] \rangle$$

είναι ισοδύναμα, δεδομένου ότι και τα δύο έχουν x = 3, y = 1 και x = 8, y= 4 ως τις μοναδικές λύσεις.

Γενικά, δεδομένου ότι χρησιμοποιούμε κάποια γλώσσα για να περιγράψουμε τους περιορισμούς, μπορεί να είναι δύσκολο να ορίσουμε ότι τα δύο CSPs είναι ισοδύναμα. Παραδείγματος χάριν, δεν είναι εύκολο να δούμε ότι το CSP

$$\langle x^4 - 10x^3 + 35x^2 - 50x + 24 = 0 \; ; \; x \in \mathcal{R} \rangle$$

είναι ισοδύναμο με το CSP

$$\langle \; ; \; x \in \{1, 2, 3, 4\} \rangle$$

στο οποίο κανένας περιορισμός δεν υπάρχει και η περιοχή του x αποτελείται από τέσσερις τιμές ακριβώς. Ευτυχώς, θα χρησιμοποιήσουμε την παραπάνω έννοια σε πολύ περισσότερο στοιχειώδεις καταστάσεις στις οποίες θα είναι απλό να επαληθεύσουμε ότι τα εξεταζόμενα CSPs είναι ισοδύναμα.

Ο παραπάνω ορισμός είναι μάλλον περιορισμένος καθώς δεν μπορεί να χρησιμοποιηθεί για να συγκρίνει CSPs με διαφορετικές ακολουθίες μεταβλητών. Τέτοιες καταστάσεις προκύπτουν συχνά, παραδείγματος χάριν όταν εισάγονται οι νέες μεταβλητές ή όταν μερικές μεταβλητές διαγράφονται. Εξετάστε παραδείγματος χάριν το πρόβλημα λύσης της εξίσωσης

$$2x^5 - 5x^4 + 5 = 0 \tag{3.1}$$

πέρα από τους πραγματικούς αριθμούς. Ένας τρόπος να προχωρήσουμε είναι με την εισαγωγή μιας νέας μεταβλητής y και το μετασχηματισμό του (3.1) στις ακόλουθες δύο εξισώσεις:

$$2x^5 - y + 5 = 0, \tag{3.2}$$

$$y = 5x^4, \tag{3.3}$$

έτσι ώστε σε κάθε εξίσωση κάθε μεταβλητή να εμφανίζεται το πολύ-πολύ μιά φορά.

Τώρα, το CSP που διαμορφώνεται από τις εξισώσεις (3.2) και (3.3) έχει μια μεταβλητή περισσότερη από αυτό που διαμορφώνεται από την (3.1), έτσι δεν μπορούμε να υποστηρίξουμε ότι αυτά τα δύο CSPs είναι ισοδύναμα από την άποψη του ορισμού που μόλις εισήχθη. Για να εξετάσουμε τέτοιες καταστάσεις εισάγουμε μια γενικότερη έννοια αυτή της ισοδυναμίας.

Ορισμός 3.1 Θεωρήστε δύο CSPs P_1 και P_2 και μια ακολουθία των κοινών μεταβλητών τους X (δηλαδή το X είναι μια υπο-ακολουθία των X_1 και X_2, όπου X_1 και X_2 είναι αντίστοιχα οι ακολουθίες των μεταβλητών του P_1 και P_2).Λέμε ότι τα P_1 και P_2 είναι **ισοδύναμα w.r.t.** X εάν

- για κάθε λύση d στο P_1 υπάρχει μια λύση στο P_2 που συμπίπτει με το d στις μεταβλητές στο X,
- για κάθε λύση e στο P_2 υπάρχει μια λύση στο P_1 που συμπίπτει με το e στις μεταβλητές στο X.

Για μια περαιτέρω συζήτηση είναι χρήσιμο να εισαχθεί η ακόλουθη απλή έννοια.

Ορισμός 3.2 Θεωρήστε μια ακολουθία μεταβλητών X := $x_1,...,x_n$ με την αντίστοιχη ακολουθία περιοχών $D_1,...,D_n$. Πάρτε ένα στοιχείο d:= $(d_1,...,d_n)$ του D_1 X...X D_n και μια υπό-ακολουθία Y := $x_{i1},...x_{il}$ του X. Κατόπιν δηλώνουμε την ακολουθία $(d_{i1},...,d_{il})$ με d[Y] και την καλούμε **προβολή** του d στο Y. Ειδικότερα για to d E D_1 X...X D_n d[x_i] δείχνει το ith στοιχείο του d.

Χρησιμοποιώντας την έννοια της προβολής μπορούμε να ορίσουμε την έννοια της λύσης σε ένα CSP με έναν σύντομο τρόπο. Δηλαδή, λαμβάνοντας υπόψη ένα CSP P :=⟨ C ; x E D1,...,x E D_n) ένα n-tuple (d_1,...,d_n) E D_1 X...X D_n είναι μια λύση στο P εάν για κάθε περιορισμό C του P έχουμε d[Y] E C σε μια ακολουθία μεταβλητών Y .

Χρησιμοποιώντας την έννοια της προβολής μπορούμε επίσης να ορίσουμε την έννοια της ισοδυναμίας με έναν συντομότερο τρόπο. Δηλαδή, τα CSPs P_1 και P_2 είναι ισοδύναμα w.r.t. X εαν

$$\{d[X] \mid d \text{ is a solution to } \mathcal{P}_1\} = \{d[X] \mid d \text{ is a solution to } \mathcal{P}_2\}.$$

Σαφώς, δύο CSPs με την ίδια ακολουθία μεταβλητών X είναι ισοδύναμα εάν είναι ισοδύναμα w.r.t. X, έτσι η έννοια της ισοδυναμίας που παρουσιάσαμε πρόσφατα είναι μια γενίκευση της προηγούμενης. Κατά το μετασχηματισμό των CSPs συχνά η αρχική ακολουθία μεταβλητών προσπαθεί να διατηρήσει την ισοδυναμία w.r.t.. Σημειώστε παραδείγματος χάριν ότι το CSP που διαμορφώνεται από την εξίσωση (3.1) και αυτό που διαμορφώνεται από τις εξισώσεις (3.2) και (3.3) είναι ισοδύναμα w.r.t. x.

Γενικά, μια αναγωγή ενός CSP σε ένα άλλο που είναι ισοδύναμο με αυτό δεν αρκεί για να λυθεί. Επομένως επιτρέπουμε τη διάσπαση ενός δεδομένου CSP σε δύο ή περισσότερα CSPs. Γι' αυτό το λόγο πρέπει να τυποποιήσουμε μια άλλη αρχή ισοδυναμίας για μια τέτοια διάσπαση.

Ορισμός 3.3 Θεωρήστε τα CSPs P_0,...,P_m, όπου m ≥1, και μια ακολουθία X των κοινών μεταβλητών τους (δηλαδή το X είναι μια υπο-ακολουθία σε κάθε μία από τις ακολουθίας των μεταβλητών P_0,...P_m).Λέμε ότι η **ένωση του** P_1...,P_m **είναι ισοδύναμη w.r.t.** X **στο** P_0 εάν

- για κάθε λύση d σε κάποιο P_i, όπου i≥1, υπάρχει μια λύση στο P_0 που συμπίπτει με το d στις μεταβλητές στο X,
- για κάθε λύση d σε κάποιο Pi όπου i≥1 υπάρχει μια λύση στο P_0, που συμπίπτει με το e στις μεταβλητές στο X.

Πάλι, χρησιμοποιώντας την έννοια της προβολής μπορούμε να ορίσουμε αυτήν την έννοια πιό περιληπτικά ως εξής. Η ένωση P_1,...,P_m είναι ισοδύναμη w.r.t. X στο P_0 εάν

$$\{d[X] \mid d \text{ is a solution to } \mathcal{P}_0\} = \bigcup_{i=1}^{\cdots} \{d[X] \mid d \text{ is a solution to } \mathcal{P}_i\}.$$

Εάν m = 1, αυτός ο ορισμός συμπίπτει με την έννοια της ισοδυναμίας δύο CSPs που παρουσιάσαμε προηγουμένως, P_1 και P_0 εδώ, w.r.t. μια ακολουθία μεταβλητών X.

Παρακάτω αντιπροσωπεύουμε τους μετασχηματισμούς των CSPs με έναν άτυπο τρόπο, με τη βοήθεια των κανόνων απόδειξης. Αυτή η προσέγγιση θα γίνει ακριβέστερη στην Παράγραφο 4.1. Εδώ αρκεί να θυμηθούμε ότι ένας κανόνας της μορφής

$$\frac{\phi}{\psi}$$

αντιπροσωπεύει έναν μετασχηματισμό του CSP φ στο CSP ψ και ένας κανόνας της μορφής

$$\frac{\phi}{\psi_1 \mid \ldots \mid \psi_n}$$

αντιπροσωπεύει έναν μετασχηματισμό του CSP φ στο CSP $\psi_1,\ldots,\psi_n$.

Υποθέστε ότι το X είναι η ακολουθία των μεταβλητών που υπάρχουν στο φ. Οι κανόνες που εξετάστηκαν διατηρούν την ισοδυναμία υπό την έννοια ότι, στον πρώτο κανόνα φ και ψ είναι ισοδύναμα w.r.t. X, και στο δεύτερο κανόνα η ένωση των $\psi_1,\ldots,\psi_n$ είναι ισοδύναμη στο φ w.r.t. X. Όταν ένα μέρος του CSP είναι άσχετο, το παραλείπουμε και αναφέρουμε στον κανόνα μόνο το μέρος που αλλάζει.

3.2 Βασικό πλαίσιο για τον προγραμματισμό περιορισμών

Μπορούμε τώρα να διατυπώσουμε ένα βασικό πλαίσιο για τον προγραμματισμό περιορισμών που θα χρησιμοποιήσουμε σε αυτό το βιβλίο για να εξηγήσουμε τις συγκεκριμένες πτυχές του. Κατ' αρχήν, διατυπώνουμε το αρχικό πρόβλημά μας ως CSP. Όπως είδαμε στο Κεφάλαιο 2 αυτό μπορεί από μόνο του να είναι ένα μη- μηδενικό πρόβλημα. Ειδικότερα, σε αυτή τη φάση πρέπει να πάρουμε αποφάσεις σχετικά με την επιλογή των μεταβλητών, των περιοχών και των περιορισμών. Όπως αναφέρεται ήδη στο Κεφάλαιο 1 αυτή η φάση προγραμματισμού περιορισμών καλείται **διαμόρφωση** και σε αντίθεση με τον προγραμματισμό σε άλλες μορφές προγραμματισμού υπάρχει περισσότερος χρόνος κατανάλωσης και πιο σύνθετος. Η διαμόρφωση είναι περισσότερο τέχνη παρά επιστήμη και ένας αριθμός από εμπειρικούς κανόνες και διάφορες ευρετικές λύσεις είναι χρήσιμα σε αυτή τη φάση.

Στη συνέχεια, εφαρμόζουμε στο διατυπωμένο CSP τη γενική διαδικασία SOLVE ορισμένη στο Σχήμα 3.1. Παραμετροποιείται από τις δευτερεύουσες διαδικασίες PREPROCESS, CONSTRAINT PROPAGATION, HAPPY, ATOMIC, SPLIT, και PROCEED BY CASES. Η PROCEED BY CASES οδηγεί σε μια επαναλαμβανόμενη κλήση της SOLVE για κάθε πρόσφατα διαμορφωμένο CSP. Η Λογική μεταβλητή CONTINUE είναι τοπική στην SOLVE κι έτσι δηλώνεται εκ νέου κάθε φορά ότι η μεταβλητή SOLVE καλείται κατ' επανάληψη.

Η SOLVE διαδικασία αντιπροσωπεύει το βασικό **βρόγχο** του προγραμματισμού περιορισμών. Στη συνέχεια εξηγούμε, με έναν άτυπο τρόπο, την έννοια όλων των διαδικασιών που χρησιμοποιούνται στην SOLVE. Δεδομένου ότι η έννοια της διάδοσης περιορισμών είναι κεντρική στον προγραμματισμό περιορισμών αναβάλλουμε τη συζήτησή της στο τέλος του κεφαλαίου. Σε αυτή τη φάση αρκεί να ξέρουμε ότι η διάδοση περιορισμών μετασχηματίζει ένα δεδομένο CSP σε ένα άλλο που είναι ισοδύναμο με αυτό, ενδεχομένως w.r.t. μια ακολουθία των αρχικών μεταβλητών.

```
SOLVE:

VAR CONTINUE: BOOLEAN;
CONTINUE:= TRUE;
WHILE CONTINUE AND NOT HAPPY DO
  PREPROCESS;
  CONSTRAINT PROPAGATION;
  IF NOT HAPPY
  THEN
    IF ATOMIC
    THEN
      CONTINUE:= FALSE
    ELSE
      SPLIT;
      PROCEED BY CASES
    END
  END
END
```

Σχήμα 3.1. Γενική διαδικασία SOLVE

3.2.1 PREPROCESS

Στόχος αυτής της διαδικασίας είναι να παρουσιαστεί το εξεταζόμενο CSP σε μια επιθυμητή συντακτική μορφή. Το τελικό CSP πρέπει να είναι ισοδύναμο με το αρχικό w.r.t. την αρχική ακολουθία μεταβλητών. Για να το επεξηγήσουμε εξετάζουμε δύο απλά παραδείγματα.

Κατ' αρχήν, θεωρήστε τους περιορισμούς Λογικών Τιμών όπως αναφερθήκαν στην Παράγραφο 2.4. Οι περισσότερες από τις διαδικασίες που τους εξετάζουν υποθέτουν ότι αυτοί οι περιορισμοί είναι σε μια συγκεκριμένη συντακτική μορφή. Ένα γνωστό παράδειγμα είναι η **συνδετική κανονική μορφή** σύμφωνα με την οποία ο περιορισμός Λογικών Τιμών είναι μια σύνδεση προτάσεων. Μια **πρόταση** είναι μια αποσύνδεση των literals, όπου στη συνέχεια ένας **literal** είναι μια μεταβλητή η οποία λαμβάνει λογικές τιμές 0 ή 1, ή η άρνησή της. Παραδείγματος χάριν, ο περιορισμός

$$(x \vee y) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg z)$$

είναι με συνδετική κανονική μορφή, δεδομένου ότι κάθε σύνδεσμος του είναι μια πρόταση. Σε αυτήν την περίπτωση η προεπεξεργασία αποτελείται από τους κανόνες που μετασχηματίζουν μια αυθαίρετη έκφραση Λογικής Τιμής σε μια ισοδύναμη η οποία είναι σε συνδετική κανονική μορφή.

Ως δεύτερο παράδειγμα θεωρήστε τους περιορισμούς στις Πραγματικές Τιμές όπως αναφέρθηκε στην Παράγραφο 2.3. Συχνά, οι συγκεκριμένες διαδικασίες που εξετάζουν τέτοιους περιορισμούς υποθέτουν ότι σε κάθε περιορισμό κάθε μεταβλητή εμφανίζεται το πολύ-πολύ μία φορά. Σε αυτήν την περίπτωση η προεπεξεργασία αποτελείται από το μετασχηματισμό κάθε περιορισμού σε μια τέτοια μορφή με την εισαγωγή των βοηθητικών μεταβλητών. Παραδείγματος χάριν, λαμβάνοντας υπόψη μια εξίσωση

$$ax^7 + bx^5y + cy^{10} = 0$$

υιοθετούμε μια βοηθητική μεταβλητή z και την αντικαθιστάμε από δύο εξισώσεις,

$$ax^7 + z + cy^{10} = 0$$

και

$$bx^5y = z.$$

Συνήθως, η διαδικασία PREPROCESS εφαρμόζεται μόνο μία φορά, στο ανώτερο επίπεδο της διαδικασίας SOLVE. Για να εξετάσουμε τις αυθαίρετες καταστάσεις την βάζουμε μέσα στη διαδικασία SOLVE, έτσι ώστε να καλείται επιπλέον κάθε φορά που υλοποιείται η διαδικασία SPLIT. Μπορεί παραδείγματος χάριν η SPLIT να διασπάσει τους περιορισμούς που πρέπει να προεπεξεργαστούν πριν περάσουν στη διαδικασία CONSTRAINT PROPAGATION.

3.2.2 HAPPY

Ανεπίσημα, HAPPY σημαίνει πως ο τελικός στόχος που διαμορφώνει το αρχικό CSP έχει επιτευχθεί. Ποιος είναι αυτός ο τελικός στόχος εξαρτάται φυσικά από τις εφαρμογές. Οι ακόλουθες πιθανότητες είναι οι πιό κοινές:

- έχει βρεθεί μια λύση,
- έχουν βρεθεί όλες οι λύσεις,
- έχει επιτευχθεί μια "λυμένη μορφή" από την οποία είναι απλό να δημιουργηθούν όλες οι λύσεις:

Αυτή η πιθανότητα είναι χρήσιμη για να ασχοληθούμε με προβλήματα όταν υπάρχουν αρκετές, ενδεχομένως απείρως πολλές, λύσεις,

- εντοπίστηκε μία αστάθεια ,
- βρέθηκε μια βέλτιστη λύση w.r.t. κάποιας αντικειμενικής λειτουργίας,
- βρέθηκαν όλες οι βέλτιστες λύσεις w.r.t. κάποιας αντικειμενικής λειτουργίας.
- (στην περίπτωση των περιορισμών στις Πραγματικές Τιμές) όλες οι περιοχές διαστήματος μειώνονται σε μικρότερα μεγέθη από κάποιο σταθερό εκ των προτέρων ε.

Γενικά, η διαδικασία HAPPY μπορεί να αντιμετωπισθεί ως δοκιμή που εφαρμόζεται στο τρέχον CSP στο οποίο μπορούν να ληφθούν υπόψη μερικές πρόσθετες παράμετροι σε περίπτωση που ψάχνουμε μια βέλτιστη, αντίστοιχα όλες τις βέλτιστες, λύσεις w.r.t. κάποιας αντικειμενικής λειτουργίας.

3.2.3 ATOMIC

Προτού χωρίσουμε ένα CSP πρέπει να ελέγξουμε εάν υπάγεται στο διαχωρισμό. Αυτό γίνεται χρησιμοποιώντας ως δοκιμή την διαδικασία ATOMIC. Συνήθως, ορίζουμε ότι ένα CSP ικανοποιεί αυτήν την δοκιμή εάν οι περιοχές του είναι ορισμένα μονοσύνολα ή κενές. Αλλά ένα CSP P μπορεί επίσης να θεωρηθεί ως ατομικό εάν δεν απαιτείται άλλο περαιτέρω αναζήτηση "under". Αυτή η περίπτωση μπορεί να είναι για παράδειγμα, όταν το P είναι λυμένο ή, σε περίπτωση που αναζητάμε μια βέλτιστη λύση, μπορεί να υπολογιστεί άμεσα μια βέλτιστη λύση για το P.

3.2.4 SPLIT

Εάν μετά από τη λήξη της διάδοσης περιορισμών δεν επιτύχουμε τον αρχικό στόχο, αυτό συμβαίνει γιατί η δοκιμή HAPPY αποτυγχάνει, και το τρέχον CS P δεν είναι ατομικό, το οποίο NOT ATOMIC εξακολουθεί να ισχύει, το P είναι χωρισμένο σε δύο ή περισσότερα CSPs, η ένωση των οποίων είναι ισοδύναμη με το P. Μια τέτοια διάσπαση ισχύει είτε με το διαχωρισμό μιας περιοχής είτε με το διαχωρισμό ενός περιορισμού. Οδηγεί σε μια αντικατάσταση του τρέχοντος CSP από δύο ή περισσότερα CSPs που διαφέρουν από το τρέχων, δεδομένου ότι η διασπασμένη περιοχή, αντίστοιχα ο διασπασμένος περιορισμός, αντικαθίσταται με ένα από τα συστατικά. Επιπλέον, σε περίπτωση που μια περιοχή είναι χωρισμένη, κάθε περιορισμός περιορίζεται στις νέες περιοχές.

Στα ακόλουθα τρία παραδείγματα η διάσπαση μιας περιοχής αντιπροσωπεύεται από έναν κανόνα που μετασχηματίζει μια έκφραση περιοχών σε δύο ή περισσότερες εκφράσεις περιοχών που χωρίζονται με τη βοήθεια του συμβόλου '|'.

- **Απαρίθμηση.**

Υποθέστε ότι η περιοχή D είναι πεπερασμένη και περιλαμβάνει τουλάχιστον δύο στοιχεία. Ο ακόλουθος κανόνας μπορεί έπειτα να χρησιμοποιηθεί:

$$\frac{x \in D}{x \in \{a\} \mid x \in D - \{a\}}$$

όπου a E D.

Αυτός ο κανόνας αντιστοιχεί σε έναν συλλογισμό με περιπτώσεις. Στην πρώτη περίπτωση υποθέτουμε ότι το x εμμένει (ή αντικαθίσταται από) την τιμή α. Στη δεύτερη περίπτωση εξετάζουμε μια τροποποιημένη περιοχή από την οποία αφαιρείται το στοιχειό α. Έτσι αυτός ο κανόνας οδηγεί στην αντικατάσταση ενός CSP

$$\langle \mathcal{C} \ ; \ \mathcal{DE}, x \in D \rangle$$

από δύο CSP,

$$\langle \mathcal{C}_1 \ ; \ \mathcal{DE}, x \in \{a\} \rangle$$

και

$$\langle C_2 \,;\, \mathcal{DE}, x \in D - \{a\}\rangle,$$

όπου το C_1 αποτελείται από τους περιορισμούς των περιορισμών στο C στις νέες περιοχές (με την περιοχή του x που είναι τώρα ένα ορισμένο μονοσύνολο {α}), και ομοίως με το C_2. Έτσι μια επαναλαμβανόμενη χρήση αυτού του κανόνα οδηγεί σε μια εκτίμηση των δυαδικών δέντρων.

•Μαρκάρισμα

Εναλλακτικά, μπορούμε να χρησιμοποιήσουμε τον ακόλουθο κανόνα ο οποίος αντιστοιχεί σε έναν συλλογισμό με k περιπτώσεις:

$$\frac{x \in \{a_1, \ldots, a_k\}}{x \in \{a_1\} \mid \ldots \mid x \in \{a_k\}}$$

Κάθε εφαρμογή αυτού του κανόνα οδηγεί σε μια αντικατάσταση ενός CSP από τα k CSPs, όπου το k είναι ίσο με το τρέχον μέγεθος της περιοχής της επιλεγμένης μεταβλητής. Έτσι αυτός ο κανόνας παραμετροποιείται από το k και οι επανειλημμένες χρήσεις του οδηγούν σε μια εκτίμηση των αυθαίρετων μετρήσιμων διακλαδισμένων δέντρων.

• Διχοτόμηση

Υποθέστε ότι η περιοχή είναι ένα μη-άδειο πραγματικό διάστημα, όπως το [a,b]. Μπορούμε έπειτα να εφαρμόσουμε τον ακόλουθο κανόνα:

$$\frac{x \in [a, b]}{x \in [a, \frac{a+b}{2}] \mid x \in [\frac{a+b}{2}, b]}$$

ο οποίος διατηρεί την ιδιότητα ότι οι περιοχές που είναι μη- άδειες στο κόστος δημιουργίας των διασπασμένων περιοχών συμπίπτουν. Σαφώς, ο ίδιος κανόνας μπορεί να εφαρμοστεί στα μη- άδεια διαστήματα ακεραίων αριθμών.

Στη συνέχεια, τα ακόλουθα δύο παραδείγματα, γραμμένα επίσης ως κανόνες, επεξηγούν μια διάσπαση ενός περιορισμού.

• Διαζευκτικοί περιορισμοί

Υποθέστε ότι ο περιορισμός είναι μια διαζευκτική πρόταση Λογικής Τιμής. Τα συστατικά αυτής της διαζευκτικής πρότασης μπορεί να είναι αυθαίρετοι περιορισμοί. Τέτοιοι περιορισμοί καλούνται **διαζευκτικοί περιορισμοί**. Ένα παράδειγμα είναι ο ακόλουθος περιορισμός

$$\begin{aligned}&\mathtt{Start}[\mathtt{task}_1] + \mathtt{Duration}[\mathtt{task}_1] \le \mathtt{Start}[\mathtt{task}_2] \vee \\ &\mathtt{Start}[\mathtt{task}_2] + \mathtt{Duration}[\mathtt{task}_2] \le \mathtt{Start}[\mathtt{task}_1]\end{aligned}$$

ο οποίος εμφανίζεται τυπικά στο σχεδιασμό προβλημάτων. Δηλώνει ότι το task1 σχεδιάζεται πριν από το task2 ή αντίστροφα. Για να τον επεξεργαστούμε μπορούμε να εφαρμόσουμε τον ακόλουθο κανόνα:

$$\frac{C_1 \vee C_2}{C_1 \mid C_2}$$

Αυτός αντιστοιχεί στο συλλογισμό στον οποίο αντιμετωπίζουμε κάθε έναν διαζευκτικό περιορισμό χωριστά, παρουσία άλλων αμετάβλητων περιορισμών.

• **Περιορισμοί σε "σύνθετη" μορφή**

Η ιδέα είναι ότι τέτοιοι περιορισμοί είναι διαχωρισμένοι στους συντακτικά απλούστερους περιορισμούς οι οποίοι μπορούν να αντιμετωπιστούν άμεσα. Θεωρήστε παραδείγματος χάριν τον περιορισμό $|p(\bar{x})| = a$, όπου το $p(\bar{x})$ είναι ένα πολυώνυμο που ξεπερνά τους πραγματικούς και το α είναι ένας μη-αρνητικός πραγματικός αριθμός.

Μπορεί να αντιμετωπισθεί χρησιμοποιώντας τον ακόλουθο κανόνα:

$$\frac{|p(\bar{x})| = a}{p(\bar{x}) = a \mid p(\bar{x}) = -a}$$

Εφαρμόζοντας τον ισοδυναμεί με τον αρχικό περιορισμό ξαναγράφοντας τον ως διαζευκτικό περιορισμό

$$p(\bar{x}) = a \vee p(\bar{x}) = -a$$

και χρησιμοποιώντας τον παραπάνω κανόνα για διαζευκτικούς περιορισμούς.

Είναι χρήσιμο να αναφέρουμε ότι οι περιορισμοί που μπορούν να ξαναγραφτούν ως διαζεύξεις δεν πρέπει να υποβληθούν σε επεξεργασία ως διαζευκτικοί περιορισμοί. Παραδείγματος χάριν, ένας περιορισμός της μορφής |x–y|=a θα μπορούσε επίσης να υποβληθεί σε επεξεργασία μετασχηματίζοντας τον σε έναν γραμμικό περιορισμό x-y=z, όπου το z είναι μια νέα μεταβλητή με την περιοχή {—a,a}. Γενικά, η χρήση των διαζευκτικών περιορισμών με τον αντίστοιχο κανόνα οδηγεί σε μια συνδυαστική έκρηξη που, εάν είναι δυνατόν, πρέπει να αποφευχθεί.

Γενικά, η διαδικασία SPLIT ορίζει επίσης ποια διασπασμένη λειτουργία πρόκειται να εφαρμοστεί στη συνέχεια. Γενικά, προκύπτουν διάφορες εναλλακτικές λύσεις. Παραδείγματος χάριν, ακόμα κι αν περιορίσουμε την προσοχή μας στη διαδικασία απαρίθμησης, εν τούτοις αντιμετωπίζουμε την επιλογή στην οποία η μεταβλητή x πρόκειται να επιλεχτεί και στην οποία η τιμή α πρόκειται να χρησιμοποιηθεί από την περιοχή της. Με άλλα λόγια, ο κανόνας ο οποίος ασχολείται με την απαρίθμηση παραμετροποιείται από τη μεταβλητή x και την τιμή α και η διαδικασία SPLIT ορίζει ποια περίπτωση αυτού του κανόνα πρόκειται να χρησιμοποιηθεί αμέσως μετά.

Αυτό μας οδηγεί στο ζήτημα των ευρετικών λύσεων που χρησιμοποιήθηκαν κατά την επίλυση των CSPs. Ανεπίσημα, ένας **heuristic** είναι ένας κανόνας που μας λέει πώς να επιλέξουμε ανάμεσα σε πολλές εναλλακτικές λύσεις. Εδώ πρέπει να ορίσουμε

- ποια μεταβλητή να επιλέξουμε,
- ποια τιμή να επιλέξουμε, ή
- ποιο περιορισμό να διασπάσουμε.

Υπάρχουν διάφορες ευρετικές λύσεις. Ας αναφέρουμε δύο παραδείγματα και να τα επεξηγήσουμε δίνοντας μια πρώτη εικόνα:

• επιλέξτε μια μεταβλητή που να εμφανίζεται στο μεγαλύτερο αριθμό περιορισμών; μια τέτοια μεταβλητή καλείται **εντελώς περιορισμένη μεταβλητή**.

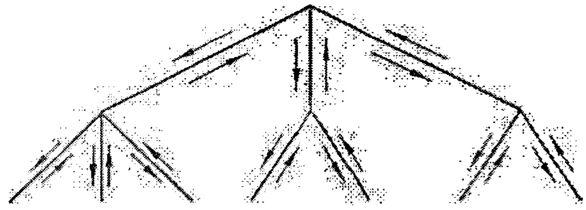
• για μια περιοχή που είναι διάστημα ακέραιων αριθμών: επιλέξτε τη μεσαία τιμή.

3.2.5 PROCEED BY CASES

Η διαδικασία SPLIT παράγει δύο ή περισσότερα νέα CSPs. Έπειτα αντιμετωπίζονται μέσω της διαδικασία PROCEED BY CASES.

Η σειρά με την οποία αυτά τα νέα CSPs μελετώνται εξαρτάται από την εφαρμοσμένη **τεχνική αναζήτησης**. Γενικά, λόγω της επανειλημμένης χρήσης της διαδικασίας SPLIT δημιουργείται το δέντρο των CSPs. Σκοπός της διαδικασίας PROCEED BY CASES είναι να εξετάσει αυτό το δέντρο αναζήτησης με μια συγκεκριμένη σειρά και, εάν είναι απαραίτητο, να ενημερώσει τις σχετικές μεταβλητές με κάποιες πρόσφατα συγκεντρωμένες πληροφορίες (στην περίπτωση της αναζήτησης της βέλτιστης λύσης). Δύο πολύ γνωστές τεχνικές αναζήτησης είναι η backtracking(αντίστροφη-διαδρομή) και, όταν αναζητάμε τη βέλτιστη λύση, η branch(διακλάδωση) και η bound(συνδεδεμένη αναζήτηση).

Άτυπα, λαμβάνοντας υπόψη ένα πεπερασμένο δέντρο, η **backtracking search** αρχίζει στη ρίζα του δέντρου και συνεχίζει κατεβαίνοντας στον πρώτο απόγονό της. Αυτή η διαδικασία συνεχίζεται εφ' όσον ο κόμβος δεν είναι ένα φύλλο. Εάν είναι φύλλο η αναζήτηση συνεχίζει κινουμένη αντίστροφα προς τον κόμβο ρίζας του φύλλου. Κατόπιν ο επόμενος απόγονος, ενδεχομένως, επιλέγεται από αυτόν τον κόμβο ρίζας. Αυτή η διαδικασία συνεχίζεται έως ότου ο έλεγχος επιστρέψει στον κόμβο ρίζας και όλοι οι απόγονοι του τον έχουν επισκεφτεί. Στο Σχήμα 3.2 απεικονίζουμε την αναζήτηση αντίστροφης-διαδρομής σε ένα απλό δυαδικό δέντρο, όπου τα βέλη δείχνουν τη σειρά με την οποία επισκέπτονται τους κόμβους.



Σχήμα 3.2. Αναζήτηση αντίστροφης-διαδρομής

Μια σημαντική πτυχή εδώ της εξεταζόμενης αναζήτησης είναι ότι το δέντρο αναζήτησης που διερευνάται δεν δίνεται εκ των προτέρων: παράγεται "on the fly". Περαιτέρω, στην περίπτωσή μας οι κόμβοι του δέντρου αναζήτησης είναι CSPs και τα φύλλα είναι επίσης CSPs τα οποία είναι λυμένα ή αποτυχημένα.

Συνήθως, όλες οι περιοχές ενός φύλλου που είναι ένα λυμένο CSP θα είναι ορισμένα μονοσύνολα που σημαίνει ότι ένα τέτοιο λυμένο CSP παράγει μια λύση. Στα φύλλα που είναι λυμένα CSPs με μη-ορισμένα μονοσύνολα η περιοχή χαρακτηριστικά προκύπτει όταν μειώσουμε ένα δεδομένο CSP σε μια "λυμένη μορφή", και το ενδεχόμενο αυτό αναφέρθηκε κατά τη διάρκεια της διαδικασίας HAPPY στην Υποενότητα 3.2.2.

Εάν ενδιαφερόμαστε για την εύρεση μιας μόνο λύσης, η αναζήτηση αντίστροφης-διαδρομής σταματά μόλις παραχθεί ένα φύλλο που είναι ένα λυμένο CSP. Εάν επιθυμούμε να βρούμε όλες τις λύσεις, η αναζήτηση συνεχίζεται έως ότου έχουν παραχθεί όλα τα φύλλα. Εάν ενδιαφερόμαστε για την ανίχνευση της ανακολουθίας, η αναζήτηση σταματά μόλις παραχθεί ένα φύλλο που είναι ένα λυμένο CSP ειδάλλως συνεχίζεται έως ότου έχουν παραχθεί όλα τα φύλλα.

Η **branch and bound search** είναι μια τροποποίηση της αναζήτησης αντίστροφης-διαδρομής η οποία λαμβάνει υπόψη την τιμή της αντικειμενικής λειτουργίας. Άτυπα, μπορεί να εξηγηθεί ως εξής. Υποθέστε ότι ενδιαφερόμαστε για την εύρεση μιας λύσης με τη μέγιστη τιμή της αντικειμενικής λειτουργίας. Κατά τη διάρκεια της αναζήτησης μια διατηρεί την **επί του παρόντος καλύτερη τιμή** αντικειμενικής λειτουργίας σε μια συνδεδεμένη μεταβλητή. Αυτή η μεταβλητή μονογράφεται σε $-\infty$ και ενημερώνεται κάθε φορά που παράγεται μια λύση με μια μεγαλύτερη τιμή. Αυτή η λύση έπειτα καταγράφεται.

Η αντικειμενική λειτουργία χρησιμοποιείται συνήθως σε συνδυασμό με μια **ευρετική λειτουργία** η οποία ορίζει μια πραγματική τιμή σε κάθε εξεταζόμενο CSP.Για να αντιμετωπίσουμε τις καταστάσεις όταν δεν παρέχεται καμία χρήσιμη πληροφορία από την ευρετική λειτουργία μία μπορεί να επεκτείνει το σύνολο τιμών που μπορεί να πάρει από το $-\infty$. Σημειώσαμε ήδη στην Παράγραφο 3.1 ότι κάθε λύση σε ένα δεδομένο CSP αντιστοιχεί σε ένα μοναδικό λυμένο CSP με τις περιοχές των ορισμένων μονοσυνόλων. Έτσι μπορούμε να υποθέσουμε ότι σε κάθε τέτοιο λυμένο CSP P η αντικειμενική λειτουργία obj ορίζει μια πραγματική τιμή. Γενικότερα, μπορούμε να υποθέσουμε ότι η αντικειμενική λειτουργία obj ορίζει μια πραγματική τιμή obj (P) σε κάθε ατομικό CSP P, δηλαδή, σε κάθε CSP που ικανοποιεί την δοκιμή ATOMIC. Η σωστή χρήση της ευρετικής λειτουργίας h αξιώνει ότι ικανοποιεί τις δύο ακόλουθες ιδιότητες:

- εάν το CSP ψ είναι άμεσος απόγονος του CSP φ στο δέντρο αναζήτησης, τότε $h(\psi) \leq h(\varphi)$.
- εάν το ψ είναι ένα ατομικό CSP, τότε $obj(\psi) \leq h(\psi)$.

Ο πρώτος όρος δηλώνει ότι η ευρετική λειτουργία γίνεται "ακριβέστερη" μόλις ένας μεταβιβαστεί κάτω στο δέντρο αναζήτησης. Ο δεύτερος όρος δηλώνει ότι η ευρετική λειτουργία υπερεκτιμά την αντικειμενική λειτουργία.

Η ευρετική λειτουργία χρησιμοποιείται έπειτα μαζί με την επί του παρόντος bound καλύτερη τιμή για να προσδιορίσει τους κόμβους κάτω από τους οποίους δεν μπορεί να βρεθεί καμία λύση με μια μέγιστη τιμή της αντικειμενικής λειτουργίας. Αυτοί είναι οι κόμβοι φ για τους οποίους κατά τη διάρκεια της διαδικασίας αναζήτησης έχουμε $h(\varphi) < bound$. Πράγματι, από τις δύο παραπάνω υποθέσεις για όλα τα λυμένα ψ που βρίσκονται κεκλιμένα κάτω από την φ έχουμε $obj(\psi) < bound$. Έτσι η ευρετική λειτουργία μας επιτρέπει να αγνοήσουμε μερικά μέρη του δέντρου κατά τη διάρκεια της διαδικασίας αναζήτησης.

Στο Σχήμα 3.3 απεικονίζουμε μια πιθανή διακλάδωση και μια συνδεδεμένη αναζήτηση στο δυαδικό δέντρο που εξετάστηκε ήδη στο Σχήμα 3.2. Τα μέρη του δέντρου που αγνοήθηκαν κατά τη διάρκεια της αναζήτησης απεικονίζονται χρησιμοποιώντας τις διαστιγμένες γραμμές.

Σχήμα 3.3. Διακλάδωση και συνδεδεμένη αναζήτηση

Η αναζήτηση αντίστροφης-διαδρομής ,η αναζήτηση διακλάδωσης καθώς και συνδεδεμένη αναζήτηση, όταν συνδυάζονται με τις κατάλληλες περιπτώσεις της διαδικασίας CONSTRAINT PROPAGATION οδηγούν σε πιό σύνθετες μορφές μεθόδων αναζήτησης που είναι ειδικές για τον προγραμματισμό περιορισμών. Ας αναφέρουμε εδώ ακόμη ότι κάποιοι μελετημένοι τρόποι γρηγορότερης αντίστροφης-διαδρομής, μας επιτρέπουν σε περίπτωση που δημιουργηθεί ένα αποτυχημένο CSP να πηδήξουμε πιο πίσω στο δέντρο παρά ακριβώς στον κόμβο ρίζας. Αυτές οι τεχνικές αποκαλούνται συνήθως **intelligent backtracking.**

3.2.6 CONSTRAINT PROPAGATION

Σε αυτή τη φάση ας επιστρέψουμε στη διαδικασία CONSTRAINT PROPAGATION. Γενικά, αυτή η διαδικασία αντικαθιστά ένα δεδομένο CSP από ένα «απλούστερο», εν τούτοις ισοδύναμο. Η ιδέα είναι ότι μια τέτοια αντικατάσταση, εάν είναι αποδοτική, είναι κερδοφόρα, αφού η αναζήτηση υπό-ακολουθίας που προκύπτει από τις επαναλαμβανόμενες κλήσεις των διαδικασιών SPLIT και PROCEED BY CASES εκτελείται έπειτα σε ένα (συνήθως ουσιαστικά) μικρότερο διάστημα αναζήτησης. Η σημασία της λέξης "απλούστερο" εξαρτάται από τις εφαρμογές. Χαρακτηριστικά, αναφέρεται το γεγονός ότι οι περιοχές ή/και οι περιορισμοί γίνονται μικρότεροι. Η διάδοση περιορισμών εκτελείται μειώνοντας επανειλημμένα τις περιοχές ή/και μειώνοντας τους περιορισμούς ενώ διατηρείται η ισοδυναμία.

Ας εξετάσουμε μερικά παραδείγματα. Σε κάθε ένα από αυτά τονίζουμε ότι το όφελος αυξήθηκε από την επαναλαμβανόμενη μείωση. Εδώ χρησιμοποιούμε πάλι κανόνες απόδειξης αλλά, τώρα αντιπροσωπεύουν μια αντικατάσταση ενός (τεμάχιο ενός) CSP από ένα άλλο. Τα πρώτα δύο παραδείγματα εξετάζουν τη μείωση περιοχών.

- Αυθαίρετα CSPs.

Θεωρήστε έναν περιορισμό C. Επιλέξτε μια μεταβλητή x από αυτόν και εκτελέστε την ακόλουθη λειτουργία στην περιοχή του D:

αφαιρέστε από το D όλες τις τιμές για το x που δεν συμμετέχουν στη λύση για το C.

Η ιδέα είναι ότι οι αφαιρούμενες τιμές δεν μπορούν να υπάρχουν σε οποιαδήποτε λύση στο εξεταζόμενο CSP. Καλούμε αυτήν την λειτουργία **προβολή του C στο x**.

Σαν παράδειγμα, θεωρήστε τον γρίφο σταυρόλεξων που αναφέρεται στο Παράδειγμα 2.8. Εάν εφαρμόσουμε τον παραπάνω κανόνα στον περιορισμό $C_{1,2}$ στις μεταβλητές x_1 και x_2 και στην περιοχή της x_1, κατόπιν μειώνουμε την περιοχή της x_1 στο σύνολο {HOSES,LASER}. Πράγματι, μόνο αυτές οι δύο λέξεις πέντε γραμμάτων μπορούν να χρησιμοποιηθούν για τη θέση 1 σε συνδυασμό με μια λέξη πέντε γραμμάτων για τη θέση 2; και οι δύο μπορούν να συνδυαστούν με τη λέξη SAILS για τη θέση 2.

Αυτή η διαδικασία μείωσης μπορεί να συνεχιστεί. Παραδείγματος χάριν, τώρα μπορούμε να εφαρμόσουμε τον παραπάνω κανόνα στον περιορισμό $C_{1,3}$ στις μεταβλητές x_1 και x_3, καθώς και στη μεταβλητή x_3. Με αυτόν τον τρόπο μειώνουμε την περιοχή της x_3 στο σύνολο {SAILS, SHEET, STEER}.

• Γραμμικές ανισότητες στους ακέραιους αριθμούς.

Υποθέστε ότι οι περιοχές είναι μη-άδεια διαστήματα ακέραιων αριθμών, γραμμένα ως [a..b], και οι περιορισμοί είναι γραμμικές ανισότητες της μορφής x < y. Κατόπιν μπορούμε να εφαρμόσουμε τον ακόλουθο κανόνα:

$$\frac{\langle x < y \ ; \ x \in [l_x..h_x], y \in [l_y..h_y]\rangle}{\langle x < y \ ; \ x \in [l_x..min(h_x, h_y - 1)], y \in [max(l_y, l_x + 1)..h_y]\rangle}$$

που μας επιτρέπει να μετασχηματίσουμε και τα δύο εξεταζόμενα διαστήματα σε μικρότερα.

Για να δούμε ότι αυτός ο κανόνας διατηρεί την ισοδυναμία, αρκεί να δείξουμε ότι οποιαδήποτε λύση που παρουσιάζεται εισαγωγικά στο CSP αποτελεί επίσης μια λύση στο CSP σαν συμπέρασμα. Έτσι λοιπόν πάρτε ένα ζευγάρι (α, b) το οποίο είναι μια λύση στο CSP που παρουσιάζεται εισαγωγικά, δηλ., έτσι ώστε a Ε $[l_X..h_X]$, b Ε $[l_Y..h_y]$ και a < b. Κατόπιν δεδομένου ότι $b \leq h_y$ παίρνουμε $α \leq h_y - 1$ και από $l_x \leq a$ παίρνουμε $l_x + 1 \leq b$. Έτσι $α \leq \min(h_x, h_y - 1)$ και $\max(l_y, l_x + 1) < b$, το οποίο σημαίνει ότι το ζευγάρι (το α, b) αποτελεί επίσης μια λύση στο CSP σαν συμπέρασμα.

Αυτός ο κανόνας μπορεί να εφαρμοστεί αρκετές φορές στη σειρά. Εξετάστε παραδείγματος χάριν το CSP

$$\langle x < y, y < z \ ; \ x \in [50..200], y \in [0..100], z \in [0..100]\rangle.$$

Εφαρμόζοντας αυτόν τον κανόνα στον περιορισμό x < y μετασχηματίζουμε αυτό το CSP σε

$$\langle x < y, y < z \ ; \ x \in [50..99], y \in [51..100], z \in [0..100]\rangle.$$

Μια άλλη εφαρμογή αυτού του κανόνα, αυτή τη φορά στον περιορισμό y < z, παράγει

$$\langle x < y, y < z \ ; \ x \in [50..99], y \in [51..99], z \in [52..100]\rangle.$$

Εφαρμόζοντας αυτόν τον τρίτο χρόνο κανόνα, πάλι στον περιορισμό x < y, παράγει

$$\langle x < y, y < z \ ; \ x \in [50..98], y \in [51..99], z \in [52..100]\rangle.$$

Έτσι θα μπορούσαμε να μειώσουμε και τις τρεις περιοχές μεταβλητών. Είναι εύκολο να δούμε ότι οι περαιτέρω εφαρμογές αυτού του κανόνα δεν παράγουν καμία αλλαγή.

Έπειτα, εξετάστε τη μείωση περιορισμών. Την επεξηγούμε με τη βοήθεια δύο παραδειγμάτων. Σε κάθε ένα από αυτά εισάγεται ένας νέος περιορισμός. Ένας τέτοιος νέος περιορισμός καλείται σε αυτό το πλαίσιο **implied constraint** ή **derived constraint.** (Μερικές φορές χρησιμοποιείται μια μάλλον παραπλανητική ορολογία των περιττών περιορισμών.)

Μια τέτοια προσθήκη ενός νέου περιορισμού μπορεί να αντιμετωπισθεί ως μείωση ενός περιορισμού. Δηλαδή, μια εισαγωγή ενός νέου περιορισμού, για παράδειγμα στις μεταβλητές X, αυξάνει το σύνολο των περιορισμών που χρησιμοποιήθηκαν στο X. Δεδομένου ότι κάθε περιορισμός είναι ένα σύνολο, μπορούμε να προσδιορίσουμε αυτό το σύνολο περιορισμών στο X με τη διασταύρωση τους, χωρίς να επηρεάζεται το σύνολο των λύσεων. Κάτω από αυτόν τον προσδιορισμό το νέο σύνολο περιορισμών στο X, που αντιμετωπίζεται ως ενιαίος περιορισμός, είναι ένα υποσύνολο του παλαιού.

- Γραμμικές ανισότητες.

Εξετάστε τους ακόλουθους κανόνες που συμπεριλαμβάνουν τη μεταβατικότητα και την αντισυμμετρία της σχέσης < σε κάποια αριθμητική περιοχή:

$$\frac{\langle x < y, y < z \ ; \ \mathcal{DE}\rangle}{\langle x < y, y < z, x < z \ ; \ \mathcal{DE}\rangle}$$

$$\frac{\langle x < y, y < x \ ; \ \mathcal{DE}\rangle}{\langle x < y, y < x, \perp \ ; \ \mathcal{DE}\rangle}$$

Ο πρώτος κανόνας εισάγει έναν νέο περιορισμό x < z και ο δεύτερος κανόνας εισάγει τον λανθασμένο περιορισμό $\perp$. Ως ένα χρήσιμο παράδειγμα θεωρήστε το προφανώς μεταβλητό CSP

$$\langle x < y, y < z, z < x \ ; \ \mathcal{DE}\rangle.$$

Χρησιμοποιώντας το πρώτο κανόνα εν συνεχεία έχουμε

$$\langle x < y, y < z, x < z, z < x \ ; \ \mathcal{DE}\rangle.$$

Εφαρμόζοντας τώρα το δεύτερο κανόνα στους περιορισμούς x < z, z < x παίρνουμε το αποτυχημένο CSP

$$\langle x < y, y < z, x < z, z < x, \perp \ ; \ \mathcal{DE}\rangle.$$

• Μέθοδος ανάλυσης.

Αυτός ο κανόνας ασχολείται με τις προτάσεις, δηλαδή, τις αποσυνδέσεις των literals. (Θυμηθείτε από την Υποενότητα 3.2.1 ότι ένα literal είναι μια μεταβλητή η οποία λαμβάνει λογικές τιμές 0 ή 1, ή η άρνησή της.) Αφήστε τις C1 και C2 να είναι προτάσεις, το L literal και το $\bar{L}$ το αντίθετο literal του L, το οποίο είναι $\overline{\neg x} = x$ και $\bar{x} = \neg x$.. Για να συνοψίσουμε τη σειρά των αποσυνδέσεων σε μια πρόταση και για να εξαιρέσουμε αυτόματα τους επαναλαμβανόμενους διαχωρισμούς γράφουμε κάθε πρόταση ως σύνολο αποσυνδέσεων. Κατόπιν ο ακόλουθος κανόνας, αποκαλούμενος **διαχωριστικός κανόνας**:

$$\frac{\langle C_1 \cup \{L\}, C_2 \cup \{\bar{L}\} \; ; \; \mathcal{DE}\rangle}{\langle C_1 \cup \{L\}, C_2 \cup \{\bar{L}\}, C_1 \cup C_2 \; ; \; \mathcal{DE}\rangle}$$

εισάγει έναν νέο περιορισμό, την πρόταση C1 ∪C2. Αυτός ο κανόνας είναι ο ακρογωνιαίος λίθος της απόδειξης του αυτοματοποιημένου θεωρήματος.

Για να επεξηγήσουμε τη χρήση του θεωρήστε το Λογικό CSP

$$\langle x \vee y, \neg x \vee y \vee z, \neg x \vee \neg z \; ; \; \mathcal{DE}\rangle$$

το οποίο περιλαμβάνει τον Λογικό περιορισμό που αναφέρθηκε νωρίτερα στην Υποενότητα 3.2.1. Εφαρμόζοντας τον διαχωριστικό κανόνα στις δυο πρώτες προτάσεις, με L ≡ x, έχουμε,

$$\langle x \vee y, \neg x \vee y \vee z, \neg x \vee \neg z, y \vee z \; ; \; \mathcal{DE}\rangle.$$

Τώρα εφαρμόζοντας αυτόν τον κανόνα στις προτάσεις $\neg x \vee \neg z$ και $y \vee z$ με L ≡ x, έχουμε,

$$\langle x \vee y, \neg x \vee y \vee z, \neg x \vee \neg z, y \vee z, \neg x \vee y \; ; \; \mathcal{DE}\rangle.$$

Μια άλλη εφαρμογή του διαχωριστικού κανόνα, αυτή τη φορά στις προτάσεις x V y και ¬x V y πάλι με L ≡ x, οδηγεί στην προσθήκη του περιορισμού y στο τελευταίο CSP.

Έτσι χρησιμοποιώντας τον διαχωριστικό κανόνα θα μπορούσαμε να ανάγουμε από το αρχικό Λογικό CSP ότι το y είναι αληθινό.

Πρέπει να τονιστεί εδώ ότι δεν είναι όλοι οι περιορισμοί που προκύπτουν χρήσιμοι. Παραδείγματος χάριν, επιστρέφοντας στο παραπάνω παράδειγμα των γραμμικών ανισοτήτων, ο κανόνας

$$\frac{\langle x < y \; ; \; \mathcal{DE}\rangle}{\langle x < y, x < y + 1 \; ; \; \mathcal{DE}\rangle}$$

εισάγει έναν συνεπαγόμενο περιορισμό που είναι ασήμαντος. Συνήθως, η πρόκληση είναι να βρεθούν οι χρήσιμοι συνεπαγόμενοι περιορισμοί η χρήση των οποίων οδηγεί σε μια αποτελεσματική πρόοδο στην επίλυση του εξεταζόμενου CSP.

Ως επί το πλείστον, οι διάφορες γενικές τεχνικές που προέρχονται από τους τομείς της γραμμικής άλγεβρας, του γραμμικού προγραμματισμού, του προγραμματισμού ακεραίων, και της απόδειξης του αυτοματοποιημένου θεωρήματος μπορούν να εξηγηθούν ως μείωση περιορισμών: ο διαχωριστικός κανόνας είναι μόνο ένα παράδειγμα. Περισσότερα παραδείγματα θα δοθούν στο Κεφάλαιο 4.

3.2.7 Αλγόριθμοι διάδοσης περιορισμών

Τα παραπάνω παραδείγματα ασχολήθηκαν με τα ατομικά βήματα μείωσης στα οποία μειώθηκε μια περιοχή ή ένας περιορισμός. Οι **αλγόριθμοι διάδοσης περιορισμών** ασχολούνται με το σχεδιασμό τέτοιων ατομικών βημάτων μείωσης. Προσπαθούν να αποφύγουν τις άχρηστες εφαρμογές των ατομικών βημάτων μείωσης. Επάνω στη λήξη αυτοί οι αλγόριθμοι επιτυγχάνουν μια ιδιότητα αποκαλούμενη **έννοια τοπικής σταθερότητας.**

Η αρχή της τοπικής σταθερότητας είναι κρίσιμη για τη θεωρία του προγραμματισμού περιορισμών. Στη βιβλιογραφία έχει εισαχθεί ένας μεγάλος αριθμός τέτοιων εννοιών. Για να τους επεξηγήσουμε ας διευκρινίσουμε ποια έννοια τοπικής σταθερότητας αντιστοιχεί στον πρώτο κανόνα μείωσης περιοχών που παρουσιάζεται παραπάνω. Ορίζεται ως εξής:

για κάθε περιορισμό C και κάθε μεταβλητή του x, κάθε τιμή στην περιοχή της x συμμετέχει σε ένα στοιχείο του C.

Αυτή η ιδιότητα καλείται **υπέρ-τοξοειδή σταθερότητα**. Στην περίπτωση των δυαδικών περιορισμών καλείται **τοξοειδή σταθερότητα** και είναι ίσως η δημοφιλέστερη έννοια τοπικής σταθερότητας.

Ένα παράδειγμα για ένα σταθερό τοξοειδές CSP είναι το ακόλουθο:

$$\langle x < y, y < z, x < z \ ; \ x \in [0..5], y \in [1..7], z \in [3..8]\rangle.$$

Είναι απλό να ελέγξουμε ότι είναι τοξοειδές σταθερό: για κάθε περιορισμό και κάθε μεταβλητή του κάθε τιμή στην περιοχή αυτής της μεταβλητής ανήκει σε ένα ζευγάρι που ικανοποιεί τον επιλεγμένο περιορισμό. Παραδείγματος χάριν, εάν επιλέξουμε τον περιορισμό x < y, η μεταβλητή y και η τιμή 1 στην περιοχή της y, ανήκει στο ζευγάρι (0,1) που ικανοποιεί τον περιορισμό: x < y στις υποτιθέμενες περιοχές.

Αντίθετα, το CSP

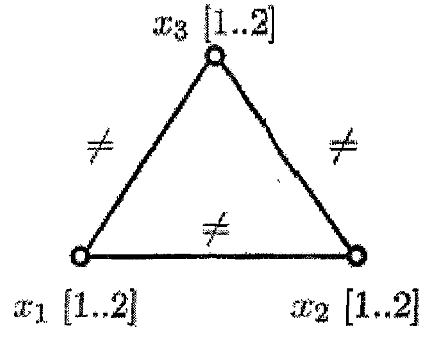
$$\langle x < y, y < z, x < z \ ; \ x \in [0..5], y \in [0..7], z \in [3..8]\rangle$$

δεν είναι τοξοειδές σταθερό, δεδομένου ότι για την τιμή 0 του y δεν υπάρχει καμία τιμή για το x μέσα στο διάστημα [0..5] έτσι ώστε x < y.

Γενικά είναι σημαντικό να συνειδητοποιήσουμε ότι ένα τοπικά σταθερό CSP δεν πρέπει να είναι σταθερό. Παραδείγματος χάριν, το CSP

$$\langle x \neq y, y \neq z, z \neq x \ ; \ x \in [1..2], y \in [1..2], z \in [1..2]\rangle,$$

που απεικονίζεται στον Σχήμα 3.4, φαίνεται αναμφίβολα ότι είναι τοξοειδές σταθερό αλλά δεν σταθερό.



Σχήμα 3.4. Ένα τοξοειδές σταθερό αλλά μεταβλητό CSP

Για να συνοψίσουμε, κάθε αλγόριθμος διάδοσης περιορισμών μειώνει ένα δεδομένο CSP σε ένα ισοδύναμο που ικανοποιεί κάποια έννοια τοπικής σταθερότητας. Ποια έννοια τοπικής σταθερότητας επιτυγχάνεται εξαρτάται από τον τύπο των εξεταζόμενων CSPs και από τη μορφή των χρησιμοποιούμενων κανόνων.

3.3 Παράδειγμα: Λογικοί περιορισμοί

Ας επεξηγήσουμε τώρα την παραπάνω αναφορά με τη βοήθεια δύο παραδειγμάτων. Θα διευκρινίσουν ποιες επιλογές πρέπει να κάνουμε κατά την επίλυση των συγκεκριμένων CSPs. Αυτές οι επιλογές θα απεικονιστούν στις συγκεκριμένες αποφάσεις σχετικά με τις δευτερεύουσες διαδικασίες της γενικής διαδικασίας SOLVE του Σχήματος 3.1. Σαν πρώτο παράδειγμα επιλέγουμε τους Λογικούς περιορισμούς. Μία πιό λεπτομερής περιγραφή της διάδοσης περιορισμών για αυτούς τους περιορισμούς θα δοθεί στην Παράγραφο 6.3.

Υποθέτουμε ότι επιθυμούμε να βρούμε όλες τις λύσεις σε ένα δεδομένο Λογικό πρόβλημα ικανοποίησης περιορισμών. Στη συνέχεια θα μπορούσαμε να εξετάσουμε την ακόλουθη επιλογή διαδικασιών που συζητήθηκαν παραπάνω.

PREPROCESS

Επιθυμούμε να φέρουμε κάθε Λογικό περιορισμό σε μια από τις ακόλουθες μορφές:

- $x = y$,
- $\neg x = y$,
- $x \wedge y = z$,
- $x \vee y = z$,

όπου τα x, y, z είναι διαφορετικές μεταβλητές.

Συνεπώς, για την διαδικασία PREPROCESS επιλέγουμε τους κανόνες μετασχηματισμού που μετασχηματίζουν κάθε Λογικό περιορισμό σε ένα σύνολο περιορισμών στην παραπάνω μορφή. Ένα παράδειγμα ενός τέτοιου κανόνα μετασχηματισμού είναι

$$\frac{x \wedge s = z}{x \wedge y = z, s = y}$$

όπου τα x, y, z είναι Λογικές μεταβλητές και το s είναι μια μη-μεταβλητή Λογική έκφραση που δεν περιέχει το y.

HAPPY

Επιλέγουμε τη δοκιμή:

όλες οι λύσεις έχουν βρεθεί.

ATOMIC

Θεωρούμε δεδομένο ότι ένα CSP είναι ατομικό εάν καμία περιοχή του δεν περιέχει περισσότερα από ένα στοιχειά.

SPLIT

Χρησιμοποιούμε τον κανόνα μαρκαρίσματος που αναφέρθηκε στην Υποενότητα 3.2.4. Στην περίπτωσή μας συνοψίζεται στον ακόλουθο κανόνα:

$$\frac{x \in \{0, 1\}}{x \in \{0\} \mid x \in \{1\}}$$

Όπως σημειώθηκε ήδη αυτός ο κανόνας παραμετροποιείται από τη μεταβλητή, η επιλογή της οποίας είναι σχετική εδώ, και από την τιμή, η επιλογή της οποίας δεν είναι καθόλου σχετική εδώ. Χρησιμοποιούμε αυτόν τον κανόνα σε συνδυασμό με την ευρετική λύση σύμφωνα με την οποία επιλέγεται πρώτα η πιο περιορισμένη μεταβλητή, δηλ., η μεταβλητή που εμφανίζεται στο μεγαλύτερο αριθμό περιορισμών.

CONSTRAINT PROPAGATION

Έπειτα, ορίζουμε τις ενέργειες του αλγόριθμου διάδοσης περιορισμών. Αυτό το κάνουμε επιλέγοντας συγκεκριμένα βήματα μείωσης περιοχών. Βασίζονται στην απλή παρατήρηση για τους Λογικούς περιορισμούς σε μία από τις μορφές που εξετάστηκαν στον ορισμό της διαδικασίας PREPROCESS, σύμφωνα με την οποία, εάν ορίζονται οι τιμές μερικών μεταβλητών, τότε μπορεί να ορίζονται και οι τιμές μερικών άλλων μεταβλητών. Παραδείγματος χάριν, για τον περιορισμό x ∧ y = z, εάν γνωρίζουμε ότι το z είναι 1 (που προσδιορίζεται με **true**), κατόπιν μπορούμε να καταλήξουμε στο συμπέρασμα ότι το x και το y είναι 1. Αυτό εκφράζεται με τη βοήθεια του ακόλουθου κανόνα:

$$\frac{\langle x \wedge y = z \ ; \ x \in D_x, y \in D_y, z \in \{1\}\rangle}{\langle \ ; \ x \in D_x \cap \{1\}, y \in D_y \cap \{1\}, z \in \{1\}\rangle}$$

όπου η απουσία του περιορισμού x ∧ y = z, στο συμπέρασμα δείχνει ότι αυτός ο περιορισμός λύνεται. Μπορούμε να τον συντομεύσουμε σε μια πιό ενδεικτική μορφή, δηλαδή

$$x \wedge y = z, z = 1 \rightarrow x = 1, y = 1.$$

Είναι χρήσιμο να καταλάβουμε γιατί δεν εφαρμόζουμε τον ακόλουθο κανόνα αντί αυτού:

$$\frac{\langle x \wedge y = z \ ; \ x \in D_x, y \in D_y, z \in \{1\}\rangle}{\langle \ ; \ x \in \{1\}, y \in \{1\}, z \in \{1\}\rangle}$$

Εάν το είχαμε κάνει, τότε με D(x) = {0} θα παίρναμε διαφορετικά αποτελέσματα. Πράγματι, χρησιμοποιώντας τον εφαρμοσμένο κανόνα θα δημιουργούσαμε κενή περιοχή για το x. Αυτό επιβεβαιώνεται στη παρατήρηση ότι το x ∧ y = z που συνδυάζεται με το z και το ¬x παράγει αστάθεια. Ανάλογη παρατήρηση ισχύει και για την περιοχή D_y. Αντιθέτως, με τον κανόνα τον οποίο δεν εφαρμόσαμε δεν θα ανακαλύπταμε την αστάθεια και αντ' αυτού λανθασμένα συμπεράναμε ότι το (1, 1, 1) είναι μία λύση στο CSP που παρουσιάζεται εισαγωγικά στον κανόνα.

Συνολικά έχουμε τους ακόλουθους έξι κανόνες για τον περιορισμό x ∧ y = z:

CONJUNCTION

$$x \wedge y = z, x = 1, y = 1 \rightarrow z = 1,$$

$$x \wedge y = z, x = 1, z = 0 \rightarrow y = 0,$$

$$x \wedge y = z, y = 1, z = 0 \rightarrow x = 0,$$

$$x \wedge y = z, x = 0 \rightarrow z = 0,$$

$$x \wedge y = z, y = 0 \rightarrow z = 0,$$

$$x \wedge y = z, z = 1 \rightarrow x = 1, y = 1.$$

Εφαρμόζουμε παρόμοιους κανόνες για τους άλλους τρεις περιορισμούς που εξετάζονται στον ορισμό της διαδικασίας PREPROCESS.

PROCEED BY CASES

Δεδομένου ότι θέλουμε να βρούμε όλες τις λύσεις επιλέγουμε εδώ την αναζήτηση αντίστροφης-διαδρομής.

Αυτό συμπληρώνει την περιγραφή μιας αντιπροσωπευτικής διαδικασίας την οποία χρησιμοποιώντας την μπορούμε να βρούμε όλες τις λύσεις σε ένα πρόβλημα ικανοποίησης Λογικών περιορισμών. Μπορεί να εφαρμοστεί εύκολα και να παράγει μια λογική προσέγγιση στην επίλυση τέτοιων περιορισμών.

3.4 Παράδειγμα: πολυωνυμικοί περιορισμοί με διαστήματα ακεραίων

Αναφορικά με το δεύτερο άμεσο παράδειγμα ενός στιγμιαίου στο πλαίσιο της Παραγράφου 3.2, θεωρήστε το πρόβλημα εύρεσης βέλτιστων λύσεων στους πολυωνυμικούς περιορισμούς με διαστήματα ακέραιων αριθμών υπαγόμενο σε μια πολυωνυμική αντικειμενική λειτουργία. Έτσι εδώ ασχολούμαστε με τα περιορισμένα προβλήματα βελτιστοποίησης. Μία πιο λεπτομερής περιγραφή της διάδοσης περιορισμών για τέτοιους περιορισμούς δίνεται στην Παράγραφο 6.5. Ως παράδειγμα ενός προβλήματος για το οποίο ενδιαφερόμαστε θεωρήστε τον στόχο εύρεσης μιας λύσης στον περιορισμό

$$x^3 + y^2 - z^3 = 0$$

στο διάστημα ακεραίων [1..1000] για το οποίο η τιμή του

$$2x \cdot y - z$$

είναι μέγιστη. (Για τον ενδιαφερόμενο αναγνώστη: η απάντηση είναι x =112, y = 832 και z= 128, για τα οποία η τιμή του 2x . y – z είναι 186240.)

Ας είμαστε ακριβέστεροι για τους περιορισμούς και τις αντικειμενικές λειτουργίες για τα οποία ενδιαφερόμαστε. Από έναν **πολυωνυμικό περιορισμό** υπολογίζουμε εδώ μια έκφραση της μορφής

$$s = 0,$$

όπου το s είναι ένα πολυώνυμο (σε ενδεχομένως διάφορες μεταβλητές) με συντελεστές ακέραιων αριθμών. Παραδείγματος χάριν το

$$2x^5 \cdot y^2 \cdot z^4 + 3x \cdot y^3 \cdot z^5 - 4x^4 \cdot y^6 \cdot z^2 + 10 = 0$$

είναι ένας πολυωνυμικός περιορισμός.

Υποθέτουμε επίσης ότι η αντικειμενική λειτουργία είναι ένα πολυώνυμο με συντελεστές ακεραίων.

PREPROCESS

Ασχολούμαστε με αυτούς τους περιορισμούς ανάγοντας τους πρώτα σε μια από τις ακόλουθες συντακτικές μορφές:

- $\Sigma_{i=1,\dots,n} a_i x_i = b$, όπου $n>0$, $a_1,\dots,a_n$ είναι μη-μηδενικοί ακέραιοι αριθμοί, $x_1,\dots,x_n$ είναι διαφορετικές μεταβλητές, και b είναι ένας ακέραιος αριθμός,
- $x \,.\, y = z$, όπου x, y, z είναι διαφορετικές μεταβλητές.

Έτσι για τη διαδικασία PREPROCESS εφαρμόζουμε τους κατάλληλους κανόνες μετασχηματισμού που μετασχηματίζουν κάθε πολυωνυμικό περιορισμό σε ένα σύνολο περιορισμών σε μια από τις δυο παραπάνω μορφές. Ένα παράδειγμα ενός τέτοιου κανόνα μετασχηματισμού είναι

$$\frac{\langle \Sigma_{i=1}^{n} m_i = 0 \ ; \ \mathcal{DE} \rangle}{\langle \Sigma_{i=1}^{n} v_i = 0, \ m_1 = v_1, \dots, \ m_n = v_n \ ; \ \mathcal{DE}, v_1 \in \mathcal{Z}, \dots, v_n \in \mathcal{Z} \rangle}$$

όπου κάποια m_i δεν είναι της μορφής ax_i και όπου οι $u_1,\dots,u_n$ είναι μεταβλητές οι οποίες δεν εμφανίζονται στο DE. Απλοποιεί ένα σύνθετο άθροισμα εισάγοντας έναν περιορισμό στην πρώτη συντακτική μορφή που απαριθμείται παραπάνω και μια ακολουθία περιορισμών κατώτερης συντακτικής πολυπλοκότητας.

Ένας άλλος κανόνας μετασχηματισμού είναι ο ακόλουθος:

$$\frac{\langle s \cdot t = v \ ; \ \mathcal{DE} \rangle}{\langle x \cdot y = z, \ s = x, \ t = y, \ v = z \ ; \ \mathcal{DE}, x \in \mathcal{Z}, y \in \mathcal{Z}, z \in \mathcal{Z} \rangle}$$

όπου τα x, y, z δεν εμφανίζονται στο DE.

Δεδομένου ότι δεν θέλουμε να εφαρμόσουμε αυτόν τον κανόνα στους περιορισμούς της μορφής x.y=z, a.y=z η΄ x.a=z, όπου τα x,y,z είναι διαφορετικές μεταβλητές και το a είναι ένας ακέραιος αριθμός, ο οποίος είναι ήδη σε μια επιθυμητή συντακτική μορφή, ορίζουμε ότι σε καμία από τις ακόλουθες πιθανότητες δεν ισχύει:

- τα s, t, u είναι διαφορετικές μεταβλητές,
- το s είναι ένας ακέραιος αριθμός και τα t και u μεταβλητές,
- το t είναι ακέραιος αριθμός και τα s και u μεταβλητές.

Και οι δύο κανόνες εισάγουν τις μεταβλητές που κυμαίνονται πέρα από το σύνολο όλων των ακέραιων αριθμών $\mathcal{Z}$. Αλλά επειδή θα εξετάσουμε στο τέλος αυτού του τμήματος, εάν οι περιοχές των αρχικών μεταβλητών είναι διαστήματα ακέραιων αριθμών, συνεπώς μπορούμε να εξασφαλίσουμε ότι και οι περιοχές των βοηθητικών μεταβλητών είναι διαστήματα ακεραίων.

HAPPY

Ενδιαφερόμαστε για την εύρεση μιας βέλτιστης λύσης, επομένως επιλέγουμε τη δοκιμή:

μια βέλτιστη λύση w.r.t. που η αντικειμενική λειτουργία βρέθηκε.

ATOMIC

Όπως στο προηγούμενο παράδειγμα θεωρούμε δεδομένο ότι ένα CSP είναι ατομικό εάν καμία περιοχή του δεν περιλαμβάνει περισσότερα από ένα στοιχεία.

CONSTRAINT PROPAGATION

Έπειτα, εξετάζουμε το κρίσιμο ζήτημα της διάδοσης περιορισμών. Φροντίζουμε αυτό τη ζήτημα προτείνοντας κανόνες για τους περιορισμούς $\Sigma_{i=1,\ldots,n} a_i x_i=b$ και x.y=z. Αυτό επιτυγχάνεται εισάγοντας πρώτα ένα **αριθμητικό διάστημα** στους ακέραιους.

Περιλαμβάνει μια γενίκευση των αριθμητικών διαδικασιών στα σύνολα των ακέραιων αριθμών. Για τα σύνολα ακεραίων X, Y ορίζουμε τις ακόλουθες διαδικασίες:

- πρόσθεση:

$$X + Y := \{x + y \mid x \in X, y \in Y\},$$

- αφαίρεση:

$$X - Y := \{x - y \mid x \in X, y \in Y\},$$

- πολλαπλασιασμός:

$$X \cdot Y := \{x \cdot y \mid x \in X, y \in Y\},$$

- διαίρεση:

$$X/Y := \{u \in \mathbb{Z} \mid \exists x \in X \exists y \in Y\, u \cdot y = x\}.$$

Για έναν ακέραιο αριθμό a και ένα op E{+, - , . , /} προσδιορίζουμε το op α X με {α} op X και το X op α με X op { α}.

Χωρίς να εισέρθουμε σε λεπτομέρειες ας αναφέρουμε ότι εκεί υπάρχουν μερικές λεπτολογίες που αφορούν τον παραπάνω ορισμό του τμήματος. Σημειώστε ότι αυτή η λειτουργία ορίζεται για όλα τα σύνολα ακεραίων, συμπεριλαμβανόμενων των Y = $\emptyset$ και Y = {0}.

Εάν περιορίζουμε την προσοχή μας στα διαστήματα των ακέραιων αριθμών τότε είναι σημαντική η ακόλουθη σημείωση.

Σημείωση Για τα διαστήματα ακέραιων αριθμών X, Y και α ένας ακέραιος ισχύουν τα εξής:

- $X \cap Y$, X+Y, X-Y είναι διαστήματα ακέραιων αριθμών.
- X/{a} είναι ένα διάστημα ακέραιου αριθμού.
- X.Y δεν είναι απαραίτητο να είναι διάστημα ακεραίων ακόμα κι αν X = {α} ή Y = {α}.
- X/Y δεν είναι απαραίτητο να είναι διάστημα ακεραίων.

Παραδείγματος χάριν έχουμε

$$[3..7] - [1..8] = [-5..6],$$

$$[3..3] \cdot [1..2] = \{3, 6\},$$

$$[3..5]/[-1..2] = \{-5, -4, -3, 2, 3, 4, 5\},$$

και

$$[-3..5]/[-1..2] = \mathcal{Z}.$$

Για να αντιμετωπίσουμε το πρόβλημα στο οποίο οι χωρίς-διάστημα περιοχές μπορούν να παραχθούν από τον πολλαπλασιασμό και την διαίρεση διαστήματος, εισάγουμε την ακόλουθη λειτουργία στα υποσύνολα του συνόλου ακεραίων $\mathcal{Z}$:

int(X) := {το μικρότερο διάστημα ακεραίων περιέχει το X, εάν το X είναι πεπερασμένο, ειδάλλως το $\mathcal{Z}$}.

Παραδείγματος χάριν int([3..3] . [1..2]) = [3..6], int ([3..5]/[- 1..2]) = [- 5..5] και int ([-3..5]/[- 1..2]) =$\mathcal{Z}$.

Τώρα είμαστε έτοιμοι να εισάγουμε τους κατάλληλους κανόνες που εξετάζουν τους περιορισμούς $\Sigma_{i=1,..n}$ a_i x_i και x.y=z. Για την πρώτη σημείωση περιορισμού

$$\sum_{i=1}^{n} a_i x_i = b$$

η οποία υπονοεί ότι για το j Ε [1..n] έχουμε

$$x_j = \frac{b - \sum_{i \in [1..n] - \{j\}} a_i x_i}{a_j}$$

Ο ακόλουθος κανόνας προτείνει το εξής:

ΓΡΑΜΜΙΚΗ ΙΣΟΤΗΤΑ

$$\frac{\langle \sum_{i=1}^{n} a_i x_i = b \; ; \; x_1 \in D_1, \ldots, x_n \in D_n \rangle}{\langle \sum_{i=1}^{n} a_i x_i = b \; ; \; x_1 \in D'_1, \ldots, x_n \in D'_n \rangle}$$

όπου

- για i≠j

$$D'_i := D_i,$$

- $$D_j' := D_j \cap \frac{b - \sum_{i\in[1..n]-\{j\}} int(a_i \cdot D_i)}{a_j}.$$

Έτσι αυτός ο κανόνας παραμετροποιείται από το j και μας επιτρέπει να μειώσουμε την περιοχή της μεταβλητής x_j. Σημειώστε τη χρήση της λειτουργίας int για να μετατρέψετε το $a_i.D_i$ σε ένα διάστημα ακέραιων αριθμών. Χάρη στην παραπάνω σημείωση η νέα περιοχή x_j παραμένει διάστημα ακέραιων. Σημειώστε ότι μετά από την εφαρμογή αυτού του κανόνα ισχύει ο συνυπολογισμός

$$D_j \subseteq \frac{b - \sum_{i\in[1..n]-\{j\}} int(a_i \cdot D_i)}{a_j}$$

Για να εξετάσουμε τον περιορισμό x.y=z εισάγουμε τρεις κανόνες που στοχεύουν σε παρόμοιους συνυπολογισμούς μεταξύ των περιοχών των μεταβλητών. Για να διατηρήσουμε την ιδιότητα ότι οι περιοχές παραμένουν διαστήματα χρησιμοποιούμε σε κάθε έναν από αυτούς τη συνάρτηση int:

ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ 1

$$\frac{\langle x \cdot y = z \; ; \; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z \; ; \; x \in D_x, y \in D_y, z \in D_z \cap int(D_x \cdot D_y) \rangle}$$

ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ 2

$$\frac{\langle x \cdot y = z \; ; \; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z \; ; \; x \in D_x \cap int(D_z/D_y), y \in D_y, z \in D_z \rangle}$$

ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΣ 3

$$\frac{\langle x \cdot y = z \; ; \; x \in D_x, y \in D_y, z \in D_z \rangle}{\langle x \cdot y = z \; ; \; x \in D_x, y \in D_y \cap int(D_z/D_x), z \in D_z \rangle}$$

Κάθε ένας από τους κανόνες ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΥ μας επιτρέπει να μειώσουμε μια περιοχή μεταβλητής. Σημειώστε ότι αφού εφαρμοστεί ο πρώτος κανόνας ισχύει ο συνυπολογισμός $D_Z \subseteq int(D_X \, . \, D_Y)$, αφού εφαρμοστεί ο δεύτερος κανόνας, ισχύει ο συνυπολογισμός $D_X \subseteq int(Dz/Dy)$, και τέλος αφού εφαρμοστεί και ο τρίτος κανόνας $D_Y \subseteq int(Dz/Dx)$, ισχύει ο συνυπολογισμός.

Αυτοί οι τρεις κανόνες, όταν συνδυάζονται μαζί, μας επιτρέπουν να πραγματοποιήσουμε σημαντικές μειώσεις περιοχών. Εξετάστε παραδείγματος χάριν το CSP

$$\langle x \cdot y = z \; ; \; x \in [1..20], y \in [9..11], z \in [155..161] \rangle.$$

Κάποιος μπορεί να ελέγξει ότι χρησιμοποιώντας και τους τρεις κανόνες ΠΟΛΛΑΠΛΑΣΙΑΣΜΟΥ μπορούμε να το μετασχηματίσουμε σε

$$\langle x \cdot y = z \ ; \ x \in [16..16], y \in [10..10], z \in [160..160]\rangle.$$

Έτσι χρησιμοποιώντας αυτούς τους κανόνες μπορούμε να βρούμε μια μοναδική λύση εδώ, x = 16, y = 10 και z =160, στο αρχικό CSP.

SPLIT

Εφαρμόζουμε τον ακόλουθο κανόνα διχοτόμησης:

$$\frac{x \in [a..b]}{x \in [a..\lfloor\frac{a+b}{2}\rfloor] \mid x \in [\lfloor\frac{a+b}{2}\rfloor + 1..b]}$$

όπου a, b είναι ακέραιοι έτσι ώστε a < b. Τον συνδυάζουμε με την ακόλουθη ευρετική λύση:

επιλέξτε τη μεταβλητή με τη μικρότερη περιοχή διαστήματος.

PROCEED BY CASES

Ενδιαφερόμαστε για την εύρεση μιας λύσης με την μέγιστη τιμή της αντικειμενικής λειτουργίας, επομένως εδώ επιλέγουμε την μέθοδο της διακλάδωσης και της συνδεδεμένης αναζήτησης. Για να είμαστε σε θέση να 'κλαδέψουμε' το δέντρο αναζήτησης χρειαζόμαστε μια κατάλληλη ευρετική λειτουργία. Για αυτόν τον σκοπό υιοθετούμε το αριθμητικό διάστημα στους ακέραιους. Χρησιμοποιώντας το μπορούμε να επεκτείνουμε την αντικειμενική λειτουργία σε μια λειτουργία από διαστήματα ακέραιων αριθμών στα πεπερασμένα σύνολα ακέραιων. Η ευρετική λειτουργία θα είναι έπειτα το μέγιστο που αυτή η επέκταση μπορεί να επιτύχει στα τρέχοντα διαστήματα.

Συγκεκριμένα, λαμβάνοντας υπόψη μια αντικειμενική λειτουργία obj που είναι ένα πολυώνυμο με τις μεταβλητές $x_1,..., x_n$, δηλώνουμε από την obj(a_1 ,...,a_n) το αποτέλεσμα της αξιολόγησης obj όταν αντικαθίσταται κάθε μεταβλητή x_i από το a_i. Παραδείγματος χάριν, για

$$\text{obj}(x, y) := x^2 . y—3x . y^2 + 5$$

έχουμε obj(2, 3) = $2^2 \cdot 3 - 3 \cdot 2 \cdot 3^2 + 5$, δηλ., obj (2, 3) = -37.

Περαιτέρω, ορίζουμε την επέκταση obj + του obj σε μια λειτουργία από τα σύνολα ακέραιων στα σύνολα ακέραιων αριθμών ερμηνεύοντας την εκθετοποίηση ως επαναλαμβανόμενο πολλαπλασιασμό και χρησιμοποιώντας του παραπάνω ορισμού της αριθμητικής διαστήματος που παρέχεται. Παραδείγματος χάριν, για obj (x, y) ορισμένο όπως παραπάνω, για τα σύνολα X, Y των ακέραιων έχουμε

$$\text{obj}^+(X, Y) = X \cdot X \cdot Y - 3 \cdot X \cdot Y \cdot Y + 5,$$

όπου ερμηνεύσαμε το x^2 ως x.x και το y^2 ως y.y.

Γενικά, εφόσον η λειτουργία πολλαπλασιασμού δεν αντιστοιχεί τα διαστήματα ακέραιων στα διαστήματα ακέραιων, η επέκταση obj + του obj, όταν εφαρμοστεί στα διαστήματα ακέραιων, δεν πρέπει να παράγει διάστημα ακέραιων. Παρ΄ όλα αυτά ισχύει το ακόλουθο λήμμα.

Λήμμα Θεωρήστε μια λειτουργία obj που είναι ένα πολυώνυμο με τις μεταβλητές $x_1,\ldots,x_n$. Αφήστε τις $X_1,\ldots,X_n$ να είναι διαστήματα ακεραίων. Έπειτα

(i) $obj^+(X_1,\ldots,X_n)$ είναι ένα πεπερασμένο σύνολο ακέραιων αριθμών.

(ii) Για οποιαδήποτε ακολουθία ακέραιων αριθμών $a_1,\ldots,a_n$ έτσι ώστε a_i Ε X_i για i Ε[1..n]

$$obj(a_1,\ldots,a_n) \in obj^+(X_1,\ldots,X_n).$$

(iii) Για οποιαδήποτε ακολουθία διαστημάτων ακέραιων αριθμών $Y_1,\ldots,Y_n$ έτσι ώστε $Y_i \subseteq X_i$ για i Ε [1...n]

$$obj^+(Y_1,\ldots,Y_n) \subseteq obj^+(X_1,\ldots,X_n).$$

Ο πρώτος ισχυρισμός ισχύει ότι αφού οι πράξεις πρόσθεσης, αφαίρεσης και πολλαπλασιασμού αντιστοιχούν τα πεπερασμένα σύνολα ακέραιων στα πεπερασμένα σύνολα ακέραιων. Ο δεύτερος και τρίτος ισχυρισμός μπορούν να αποδειχθούν εισάγοντας τους στη δομή του obj. Ο τρίτος ισχυρισμός δηλώνει ότι η λειτουργία obj^+ είναι μονοτονική w.r.t. ο ορισμένος συνυπολογισμός.

Αυτό μας φέρνει στον ορισμό της ευρετικής λειτουργίας h. Πάρτε ένα CSP P := <C ; x_1 Ε $D_1,\ldots,x_n$ Ε D_n > με $D_1,\ldots,D_n$ διαστήματα ακεραίων και μια λειτουργία obj που είναι ένα πολυώνυμο με τις μεταβλητές $x_1,\ldots,x_n$. Ορίζουμε

$$h(\mathcal{P}) := max(obj^+(D_1,\ldots,D_n)).$$

Έπειτα σύμφωνα με το παραπάνω Λήμμα και δεδομένου ότι χρησιμοποιούμε τον κανόνα διχοτόμησης καθώς και την SPLIT διαδικασία, η λειτουργία h ικανοποιεί τις δύο ιδιότητες που ορίσαμε στην Υποενότητα 3.2.5 για την ευρετική λειτουργία.

Ας επιστρέψουμε τώρα στο πρόβλημα που εισαγάγαμε στην PREPROCESS διαδικασία τις μεταβλητές με τις περιοχές που δεν είναι διαστήματα. Σημειώστε ότι κάθε φορά που εισάγουμε μία τέτοια μεταβλητή x, εξισώνεται με ένα πολυώνυμο, για παράδειγμα το s. Εξαιτίας του πρώτου στοιχείου του παραπάνω Λήμματος η λειτουργία s +, όταν εφαρμοστεί στις περιοχές των μεταβλητών που υπάρχουν στο s, παράγει ένα πεπερασμένο σύνολο ακέραιων αριθμών D. Μπορούμε έπειτα αντί του Z να χρησιμοποιήσουμε το int(D) ως περιοχή της μεταβλητής x.

Αυτό συμπληρώνει την παρουσίαση μιας συγκεκριμένης προσέγγισης για την εξέταση των περιορισμένων προβλημάτων βελτιστοποίησης σχετικά με τους πολυωνυμικούς περιορισμούς στα διαστήματα ακέραιων αριθμών.

3.5 Περίληψη

Στόχος αυτού του κεφαλαίου ήταν να εισαχθούν χωρίς διατυπώσεις οι κύριες έννοιες του προγραμματισμού περιορισμών. Για αυτόν τον σκοπό ορίσαμε ένα απλό γενικό πλαίσιο και συζητήσαμε τα κύρια συστατικά του. Ειδικότερα, διευκρινίσαμε τις ακόλουθες πτυχές του προγραμματισμού περιορισμών:

- την προεπεξεργασία,
- την αναζήτηση, και
- την διάδοση περιορισμών.

Επίσης παρουσιάσαμε δύο τυποποιημένες μεθόδους αναζήτησης:

- την αναζήτηση αντίστροφης-διαδρομής, και
- την διακλάδωση και συνδεδεμένη αναζήτηση,

και αναφέραμε το ρόλο των

- των ευρετικών λύσεων

στη διαδικασία αναζήτησης. Τέλος, επεξηγήσαμε αυτές τις δύο έννοιες σε παραδείγματα:

- τους Λογικούς περιορισμούς, και
- τους πολυωνυμικούς περιορισμούς στα διαστήματα ακέραιων αριθμών.

Πολλά στοιχεία του προγραμματισμού περιορισμών υπήρξαν από την ανάγκη που παραλείπεται σε αυτήν την άτυπη παρουσίαση. Παραδείγματος χάριν, στους αλγόριθμους διάδοσης περιορισμών αφιερώσαμε μόνο μια παράγραφο, που ανέφερε μόνο μια τοπική έννοια σταθερότητας, και δεν ανέφερε οποιουσδήποτε αλγορίθμους για την μέθοδο της αντίστροφης-διαδρομής της διακλάδωσης και της συνδεδεμένης αναζήτησης.

Ακόμα, αυτή η συνοπτική εισαγωγή στις λεπτομέρειες του προγραμματισμού περιορισμών ρίχνει κάποιο φως στο θέμα και παρέχει μερικές ιδέες στις επιλογές που πρέπει να γίνουν κατά τον προσπάθεια λύσης ενός προβλήματος με τη βοήθεια των τεχνικών προγραμματισμού περιορισμών.

4

Μερικοί πλήρη επιλυτές περιορισμών

Με τον όρο ***επιλυτής περιορισμών*** εννοούμε οποιαδήποτε διαδικασία που μετασχηματίζει ένα CSP σε ένα ισοδύναμο. Στην πράξη ενδιαφερόμαστε για αποδοτικούς επιλυτές περιορισμών. Σε αυτό το βιβλίο κάνουμε διάκριση ανάμεσα στους πλήρεις και τους ελλιπείς επιλυτές περιορισμών.

Διαισθητικά, ένας ***πλήρης επιλυτής περιορισμών*** μετασχηματίζει το αρχικό CSP *P* σε μία μορφή από την οποία είναι απλό να παράγει όλες τις λύσεις *στο P* ή να ορίσει ότι καμία λύση δεν υπάρχει. Κατά γενική ομολογία, αυτή η δήλωση είναι ανακριβής. Στην περίπτωση των επιλυτών που αναφέρονται σε αυτό το κεφάλαιο θα τον καταστήσουμε ακριβή χρησιμοποιώντας την έννοια μιας λυμένης μορφής. Ο ορισμός του εξαρτάται από τον τύπο των περιορισμών που χρησιμοποιούνται.

Καλούμε επιλυτή περιορισμών εκείνον που δεν είναι πλήρως ένας ***ελλιπής επιλυτής περιορισμών***. Διαισθητικά, ένας τέτοιος επιλυτής περιορισμών μετασχηματίζει το αρχικό CSP σε ένα απλούστερο, στο οποίο όλες οι λύσεις μπορούν να βρεθούν τελικά από το α, ενδεχομένως επαναλαμβανόμενη, περιπτωσιολογική ανάλυση που διαμορφώνεται με το διαχωρισμό. Υπό αυτήν τη μορφή τα αποτελέσματα μιας επαναλαμβανόμενης περιπτωσιολογικής ανάλυσης γενικά σε έναν συνεχή εκθετικό χρόνο, δεν επιτρέπουμε να είναι μέρος των επιλυτών περιορισμών. Για τους ελλιπής επιλυτές προκύπτει μια φυσική ερώτηση για το τι πραγματικά επιτυγχάνουν. Για να διευκρινίσουμε αυτό το ζήτημα εισάγουμε διάφορες έννοιες τοπικής σταθερότητας. Αυτό εξηγεί και τη σειρά των κεφαλαίων που ακολουθούν. Σε αυτό το κεφάλαιο εστιάζουμε σε μερικούς πλήρεις επιλυτές περιορισμών και στο επόμενο στις διάφορες έννοιες τοπικής σταθερότητας.

Αναπτύχθηκαν πλήρεις επιλυτές περιορισμών για διάφορες περιοχές και περιορισμούς. Στόχος αυτού του κεφαλαίου είναι να διευκρινιστούν τέτοιοι επιλυτές με τη βοήθεια τριών γνωστών παραδειγμάτων. Περιγράφονται με έναν ενιαίο τρόπο χρησιμοποιώντας ένα απλό θεωρητικό πλαίσιο απόδειξης που περιλαμβάνει τους κανόνες οι οποίοι μετασχηματίζουν τα CSPs διατηρώντας την ισοδυναμία. Οι εκτελέσεις ενός επιλυτή περιορισμών αντιστοιχούν έπειτα σε συγκεκριμένες παραγωγές στο εξεταζόμενο σύστημα απόδειξης. Οι πλήρεις επιλυτές περιορισμών που παρουσιάζονται σε μεγάλο βαθμό σ' αυτό το κεφάλαιο μοιάζουν με τις αρχικές διατυπώσεις των εξεταζόμενων αλγόριθμων. Αυτό το πλαίσιο θα χρησιμοποιηθεί επίσης στα δύο επόμενα κεφάλαια, για να χαρακτηρίσει τις έννοιες της τοπικής σταθερότητας και για να ορίσει διάφορους ελλιπείς επιλυτές περιορισμών.

Αυτό το κεφάλαιο οργανώνεται ως εξής. Κατ' αρχήν, στην Παράγραφο 4.1, εισάγουμε το θεωρητικό πλαίσιο απόδειξης που ήδη αναφέραμε. Κατόπιν, στην Παράγραφο 4.2 εξετάζουμε τα πεπερασμένα σύνολα εξισώσεων όρου. Το πρόβλημα έυρεσης λύσης είναι γνωστό ως πρόβλημα ενοποίησης. Για να βρούμε μία λύση εισάγουμε τον αλγόριθμο MARTELLY- MONTANARI.

Έπειτα, στην Παράγραφο 4.3 μελετάμε το πρόβλημα εύρεσης λύσης για τα πεπερασμένα σύνολα γραμμικών εξισώσεων πέρα από τα πραγματικούς. Παρουσιάζουμε δύο κλασικούς αλγόριθμους που μας επιτρέπουν να βρούμε μία λύση: τον αλγόριθμο GAUSS-JORDAN ELIMINATION και τον αλγόριθμο GAUSSIAN ELIMINATION. Τέλος, στην Παράγραφο 4.4 μελετάμε ένα σχετικό πρόβλημα εύρεσης λύσης για τα πεπερασμένα σύνολα γραμμικών ανισοτήτων πέρα από τους πραγματικούς. Για αυτόν τον σκοπό αναφέρουμε τον αλγόριθμο FOURIER- MOTZKIN ELIMINATION.

4.1 Ένα θεωρητικό πλαίσιο απόδειξης

Αρχίζουμε εισάγοντας ένα απλό τυπικό πλαίσιο το οποίο θα χρησιμοποιήσουμε για διάφορους λόγους, δηλαδή σε αυτό το κεφάλαιο για να ορίσουμε τρεις πλήρεις επιλυτές περιορισμών.

Σε αυτό το πλαίσιο εισάγουμε τους κανόνες απόδειξης και τις παραγωγές.

4.1.1 Κανόνες Απόδειξης

Οι κανόνες απόδειξης χρησιμοποιούνται για να εκφράσουν τους μετασχηματισμούς των CSPs. Έτσι είναι της μορφής

$$\frac{\phi}{\psi}$$

όπου *τα* φ και ψ είναι CSPs. Σε αυτούς τους κανόνες, ακριβώς όπως στον ορισμό των CSPs, η σειρά των περιορισμών όταν παρουσιάζονται στην αρχή και στο τέλος είναι άσχετη.

Η πρόθεσή μας είναι να χρησιμοποιήσουμε τους κανόνες απόδειξης για να μειωθεί ένα CSP σε ένα άλλο CSP κατά τέτοιο τρόπο ώστε να διατηρείται η ισοδυναμία (συνήθως w.r.t. στην αρχική ακολουθία των μεταβλητών). Αυτό οδηγεί στον ακόλουθο ορισμό.

Ορισμός 4.1 Ένας αποδεικτικός κανόνας

$$\frac{\phi}{\psi}$$

καλείται κανόνας ***διατήρησης ισοδυναμίας*** (αντίστοιχα, κανόνας ***διατήρησης ισοδυναμίας*** w.r.t. μια ακολουθία μεταβλητών *X)* εάν τα φ και ψ είναι ισοδύναμα (αντίστοιχα, ισοδύναμο w.r.t. *X)*.

Εάν το X είναι η κενή ακολουθία, τότε ο κανόνας απόδειξης ονομάζεται κανόνας ***διατήρησης σταθερότητας***.

Όπως στην περίπτωση της έννοιας της ισοδυναμίας που εισάγεται στην Παράγραφο 3.1, δεν είναι γενικά εύκολο να ορίσουμε εάν ένας κανόνας είναι κανόνας διατήρησης ισοδυναμίας. Ευτυχώς, οι κανόνες που θα εξετάσουμε θα είναι πολύ απλοί και από τον τρόπο που θα τους εισάγουμε θα είμαστε σαφείς ότι είναι όλοι κανόνες διατήρησης ισοδυναμίας. Εάν τα σύνολα μεταβλητών που παρουσιάζονται στην αρχή ενός CSP και στο τέλος του, διαφέρουν, τότε ο υπό εξέταση κανόνας θα είναι κανόνας διατήρησης ισοδυναμίας w.r.t. η ακολουθία των κοινών μεταβλητών.

Οι κανόνες διατήρησης σταθερότητας θα έχουν ενδιαφέρον μόνο στη Παράγραφο 4.4.

Σημειώστε ότι ένας κανόνας απόδειξης

$$\frac{\phi}{\psi}$$

είναι κανόνας διατήρησης σταθερότητας εάν στην ακόλουθη ισοδυναμία ισχύει:

το φ είναι σταθερό εάν το ψ είναι σταθερό.

Έτσι η διατήρηση σταθερότητας είναι μια πιο ασθενής ιδιότητα από την διατήρηση ισοδυναμίας.

Γενικά, οι επιλυτές περιορισμών στοχεύουν στη μείωση των περιοχών των εξεταζόμενων μεταβλητών ή στη μείωση των εξεταζόμενων περιορισμών, διατηρώντας την ισοδυναμία. Αυτό μεταφράζεται σε δύο τύπους κανόνων που αφορούν την διατήρηση ισοδυναμίας. Θεωρήστε τους εξής:

$$\phi := \langle \mathcal{C} \ ; \ \mathcal{DE} \rangle$$

και

$$\psi := \langle \mathcal{C}' \ ; \ \mathcal{DE}' \rangle.$$

Κανόνες μείωσης περιοχών. Αυτοί είναι κανόνες στους οποίους οι νέες περιοχές είναι αντίστοιχα υποσύνολα των παλαιών περιοχών και οι νέοι περιορισμοί είναι αντίστοιχοι περιορισμοί των παλαιών περιορισμών στις νέες περιοχές. Έτσι εδώ

- $\mathcal{DE} := x_1 \in D_1, \ldots, x_n \in D_n$,
- $\mathcal{DE}' := x_1 \in D'_1, \ldots, x_n \in D'_n$,
- *για i* $\in$*[1...n] έχουμε* $D'_i \subseteq D_i$,
- *C΄ είναι το αποτέλεσμα του περιορισμού κάθε περιορισμού στο C στην αντίστοιχη υπο-ακολουθία των περιοχών D΄*$_1$*,..., D'*$_n$

Εδώ μια αποτυχία επιτυγχάνεται μόνο όταν μια περιοχή μίας ή περισσότερων μεταβλητών μειώνεται σε κενό σύνολο.

Ένα χαρακτηριστικό παράδειγμα ενός κανόνα μείωσης περιοχών είναι ο ακόλουθος κανόνας της Υποενότητα 3.2.6, ο οποίος εξετάζει τις γραμμικές ανισότητες πέρα από τα διαστήματα ακέραιων αριθμών:

$$\frac{\langle x < y \ ; \ x \in [l_x..h_x], y \in [l_y..h_y] \rangle}{\langle x < y \ ; \ x \in [l_x..min(h_x, h_y - 1)], y \in [max(l_y, l_x + 1)..h_y] \rangle}$$

Δύο άλλα παραδείγματα είναι οι ακόλουθοι κανόνες που εξετάζουν, αντίστοιχα, τους περιορισμούς ισότητας και ανισότητας. Εδώ και γενικότερα, διαγράφουμε από το συμπέρασμα όλους τους λυμένους περιορισμούς (αλλά κρατάμε όλες τις εκφράσεις περιοχών). Επίσης, συντέμνουμε την έκφραση περιοχών *x E {α}* σε *x = α*.

ΙΣΟΤΗΤΑ

$$\frac{\langle x = y \; ; \; x \in D_x, y \in D_y \rangle}{\langle x = y \; ; \; x \in D_x \cap D_y, y \in D_x \cap D_y \rangle}$$

Σημειώστε ότι αυτός ο κανόνας παράγει μια αποτυχία όταν $D_X \cap D_Y = \emptyset$. Σε περίπτωση που το $D_X \cap D_Y$ είναι ένα μονοσύνολο ο περιορισμός $x = y$ πρώτα λύνεται και έπειτα διαγράφεται.

ΑΝΙΣΟΤΗΤΑ

$$\frac{\langle x \neq y \; ; \; x \in D, y = a \rangle}{\langle \; ; \; x \in D - \{a\}, y = a \rangle}$$

όπου *a E D*, και ομοίως με το $x \neq y$ που αντικαθίστανται από το $y \neq x$.

Ακολουθώντας τον κανόνα που μόλις παρουσιάσαμε παραλείψαμε τον περιορισμό από το συμπέρασμα αυτού του κανόνα. Αυτό εξηγεί τη διάταξη του. Σημειώστε ότι αυτός ο κανόνας παράγει μια αποτυχία όταν $D\text{-}\{a\}=\emptyset$.

Κανόνες μετασχηματισμού Αυτοί οι κανόνες δεν είναι κανόνες μείωσης περιοχών αλλά είναι αυτοί που επεκτείνουν τα $C \neq 0$ και *DE΄* σε *DE* από τις εκφράσεις περιοχών που αναφέρονται στις μη-άδειες περιοχές.

Το γεγονός ότι το *DE΄* επεκτείνεται σε *DE* σημαίνει ότι οι περιοχές των κοινών μεταβλητών είναι ίδιες και ότι οι ενδεχομένως νέες εκφράσεις περιοχών έχουν προστεθεί στο *DE*. Τέτοιες νέες εκφράσεις περιοχών εξετάζουν τις νέες μεταβλητές στις οποίες έχουν εισαχθεί μερικοί περιορισμοί. Αυτές οι μεταβλητές κυμαίνονται πέρα από τις μη-άδειες περιοχές. Εδώ μια αποτυχία επιτυγχάνεται μόνο όταν παράγεται ο λανθασμένος περιορισμός $\perp$.

Ένα παράδειγμα ενός κανόνα μετασχηματισμού είναι ο ακόλουθος κανόνας που απλοποιεί τον περιορισμό ανισότητας για τις μεταβλητές ακέραιων αριθμών:

ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ ΑΝΙΣΟΤΗΤΑΣ

$$\frac{\langle s \neq t \; ; \; \mathcal{DE} \rangle}{\langle x \neq t, \; x = s \; ; \; \mathcal{DE}, x \in \mathcal{Z} \rangle}$$

όπου

- το *s* δεν είναι μία μεταβλητή.
- το *DE* είναι μια ακολουθία εκφράσεων περιοχών που περιλαμβάνει τις μεταβλητές που υπάρχουν στα *s* και *t*,
- η *x* είναι μια μεταβλητή που δεν εμφανίζεται στο *DE*.

Ένα άλλο παράδειγμα ακολουθεί τον κανόνα που αντικαθιστά μια μεταβλητή με μια τιμή, σε περίπτωση που η περιοχή αυτής της μεταβλητής είναι ένα ορισμένο μονοσύνολο:

ΜΕΤΑΒΛΗΤΗ ΑΠΑΛΟΙΦΗΣ

$$\frac{\langle \mathcal{C} \; ; \; \mathcal{DE}, x = a \rangle}{\langle \mathcal{C}\{x/\overline{a}\} \; ; \; \mathcal{DE}, x = a \rangle}$$

όπου *το x* εμφανίζεται στο C.

Εδώ υποθέτουμε ότι οι περιορισμοί στο C γράφονται σε κάποια περαιτέρω απροσδιόριστη γλώσσα. Εδώ το $\overline{\alpha}$ ισχύει για τη σταθερότητα, το οποίο σε αυτήν την γλώσσα δηλώνει την τιμή *α* και το $C\{x/\overline{\alpha}\}$ δηλώνει το σύνολο των περιορισμών που λαμβάνονται από το *C* αντικαθιστώντας σε κάθε έναν από αυτούς κάθε εμφάνιση του *x* από το $\overline{\alpha}$. Έτσι το *x* δεν εμφανίζεται στο $C\{x/\overline{\alpha}\}$.

Για να διευκρινίσουμε περαιτέρω αυτόν τον κανόνα θεωρήστε την ακόλουθη περίπτωση του:

$$\frac{\langle 3xy^2 + 5xy - 5yz \leq 6 \; ; \; x \in [0..100], y = 2, z \in [0..100] \rangle}{\langle 22x - 10z \leq 6 \; ; \; x \in [0..100], y = 2, z \in [0..100] \rangle}$$

Μετασχηματίζει έναν μη γραμμικό περιορισμό σε έναν γραμμικό.

Χαρακτηριστικά παραδείγματα των κανόνων μετασχηματισμού είναι οι:

Κανόνες εισαγωγής. Αυτοί οι κανόνες είναι της μορφής

$$\frac{\langle \mathcal{C} \; ; \; \mathcal{DE} \rangle}{\langle \mathcal{C}, C \; ; \; \mathcal{DE} \rangle}$$

στους οποίους ένας νέος περιορισμός, ο *C*, εισήχθη στο συμπέρασμα.

Εάν ένας τέτοιος κανόνας δεν εξαρτάται από το DE, κατόπιν τον συντέμνουμε σε

$$\frac{\mathcal{C}}{\mathcal{C}, C}$$

και ομοίως ενεργούμε με άλλους κανόνες μετασχηματισμού.

Ως παράδειγμα ενός κανόνα εισαγωγής θυμηθείτε τον κανόνα ανάλυσης που ήδη αναφέραμε στην Υποενότητα 3.2.6:

ΑΝΑΛΥΣΗ

$$\frac{C_1 \vee L, C_2 \vee \bar{L}}{C_1 \vee L, C_2 \vee \bar{L}, C_1 \vee C_2}$$

Εδώ υποθέτουμε ότι τα C_1 και C_2 είναι προτάσεις, το L είναι ένα literal, το $\overline{L}$ είναι το αντίθετο literal του L και ότι όλες οι μεταβλητές κυμαίνονται πέρα από τη περιοχή $\{0,1\}$.

4.1.2 Παραγωγές

Τώρα που έχουμε ορίσει τους κανόνες απόδειξης, ορίζουμε το αποτέλεσμα της εφαρμογής ενός κανόνα απόδειξης σε ένα CSP. Διαισθητικά, μια εφαρμογή ενός κανόνα στο δεδομένο CSP οδηγεί σε μια αντικατάσταση του μέρους που συμπίπτει με το εισαγωγικό μέρος από το συμπέρασμα και στον περιορισμό των "παλαιών" περιορισμών στις νέες περιοχές σε περίπτωση που αυτός είναι ένας κανόνας μείωσης περιοχών. Λόγω των πιθανών συγκρούσεων των μεταβλητών πρέπει να είμαστε ακριβέστεροι. Έτσι θεωρήστε δεδομένο ένα CSP της μορφής $\langle C \cup C_1 ; DE, DE_1 \rangle$, όπου τα C και C_1 είναι διαχωρισμένα, και εξετάστε έναν κανόνα της μορφής

$$\frac{\langle \mathcal{C}_1 \ ; \ \mathcal{DE}_1 \rangle}{\langle \mathcal{C}_2 \ ; \ \mathcal{DE}_2 \rangle} \tag{4.1}$$

Ονομάστε ***introduced variable*** μια μεταβλητή που δεν παρουσιάζεται στην αρχή αλλά στο συμπέρασμα του κανόνα. Η εφαρμογή του κανόνα (4.1) στο CSP $(C \cup C_1 ; DE, DE_1)$ εκτελείται πραγματοποιώντας τα ακόλουθα βήματα:

- **μετονομάζουμε** τις εισαχθείσες μεταβλητές του (4.1) έτσι ώστε να μην εμφανίζονται στο$(C; DE)$,
- **αντικαθιστούμε** το τμήμα $(C_1 ; DE_1)$ από το $(C_2 ; DE_2)$,
- **περιορίζουμε** τους περιορισμούς του C στις περιοχές των DE, DE_2. Δηλώνουμε το σύνολο περιορισμών που προκύπτει από το C'.

Έπειτα λέμε ότι ο κανόνας (4.1) ***μπορεί να εφαρμοστεί*** στο $(C \cup C_1 ; DE, DE_1)$ και ονομάζουμε

$$\langle \mathcal{C}' \cup \mathcal{C}_2 \ ; \ \mathcal{DE}, \mathcal{DE}_2 \rangle$$

το ***αποτέλεσμα της εφαρμογής του κανόνα*** (4.1) ***στο*** *$(C \cup C_1 ; DE, DE_1)$*.
Το ακόλουθο λήμμα εξηγεί γιατί είναι σημαντικοί οι κανόνες διατήρησης ισοδυναμίας.

Λήμμα 4.2 (Ισοδυναμία) *Υποθέστε ότι το CSP ψ είναι το αποτέλεσμα της εφαρμογής ενός κανόνα διατήρησης ισοδυναμίας στο CSP φ. Κατόπιν τα φ και ψ είναι ισοδύναμα.*

Απόδειξη Η απόδειξη παρουσιάζεται ως Άσκηση 4.1.

Για να συζητήσουμε για την επίδραση της εφαρμογής ενός κανόνα απόδειξης σε ένα CSP εισάγουμε τις ακόλουθες έννοιες.

Ορισμός 4.3 Θεωρήστε δύο CSPs φ και ψ και έναν κανόνα R.

- Υποθέστε ότι το ψ είναι το αποτέλεσμα της εφαρμογής του κανόνα R στο CSP φ. Εάν το ψ διαφέρει από το φ, τότε το ονομάζουμε ***σχετική εφαρμογή*** του R στο φ.
- Υποθέστε ότι ο κανόνας R δεν μπορεί να εφαρμοστεί στο φ ή καμία εφαρμογή του δεν είναι σχετική με το φ. Τότε λέμε ότι το φ είναι ***περιορισμένο κάτω από τις εφαρμογές του*** R.

Όταν οι περιορισμοί αντιπροσωπεύονται με κάποια γλώσσα, δεν είναι γενικά εύκολο να οριστεί εάν μια εφαρμογή κανόνα είναι σχετική, ή εάν ένα CSP είναι περιορισμένο κάτω από τις εφαρμογές των εξεταζόμενων κανόνων. Ευτυχώς, θα χρησιμοποιήσουμε και τις δύο έννοιες μόνο στις συγκεκριμένες καταστάσεις τις οποίες θα είναι εύκολο να ελέγξουμε.

Κατά την εισαγωγή των κανόνων απόδειξης η πρόθεσή μας είναι να λάβουμε CSPs τα οποία είναι περιορισμένα κάτω από τις εφαρμογές αυτών των κανόνων. Έτσι η τελευταία έννοια είναι κρίσιμη για τις εκτιμήσεις μας. Για να το καταλάβουμε αυτό καλύτερα θεωρήστε τα ακόλουθα παραδείγματα.

Παράδειγμα 4.4

(i) Θεωρήστε τον ακόλουθο κανόνα μείωσης περιοχών που ήδη συζητήθηκε στην Υποενότητα 3.2.6:

$$\frac{\langle x < y \; ; \; x \in [l_x..h_x], y \in [l_y..h_y]\rangle}{\langle x < y \; ; \; x \in [l_x..min(h_x, h_y - 1)], y \in [max(l_y, l_x + 1)..h_y]\rangle}$$

Είναι εύκολο να ελέγξουμε ότι το CSP όπως θεωρήθηκε εκεί

$$\langle x < y, y < z \; ; \; x \in [50..98], y \in [51..99], z \in [52..100]\rangle$$

είναι περιορισμένο κάτω από τις εφαρμογές αυτού του κανόνα.

(ii) Υποθέστε για μια στιγμή την αναμενόμενη ερμηνεία των προτασιακών τύπων. Δεδομένου ότι ένα πιο περίπλοκο παράδειγμα εξετάζει τώρα το CSP

$$\phi := \langle x \wedge y = z \; ; \; x \in \{0, 1\}, y = 0, z = 0\rangle.$$

Εδώ το y=0 είναι μια σύντμηση για την έκφραση περιοχών y Ε {0,1} και ομοίως για το z= 0.

Αυτό το CSP είναι περιορισμένο κάτω από τις εφαρμογές του κανόνα μετασχηματισμού δεδομένου ότι αυτός ο κανόνας δεν μπορεί να εφαρμοστεί στο φ.

$$\frac{\langle x \wedge y = z \; ; \; x = 1, y \in D_y, z \in D_z\rangle}{\langle y = z \; ; \; x = 1, y \in D_y, z \in D_z\rangle}$$

(iii) Αντίθετα, το CSP ⟨φ := x Λ y = z ; x = 1, y Ε {0,1}, z Ε {0,1}⟩ δεν είναι περιορισμένο κάτω από τις εφαρμογές του παραπάνω κανόνα επειδή αυτός ο κανόνας μπορεί να εφαρμοστεί στο φ και το αποτέλεσμα, ⟨y = z ; x = 1, y Ε {0,1}, z Ε {0,1}⟩ , διαφέρει από το φ.

Με την επανάληψη των εφαρμογών του κανόνα λαμβάνουμε τις παραγωγές. Διαισθητικά, οι παραγωγές αντιστοιχούν στη διαδικασία υπολογισμού η οποία ξεκινάει με το αρχικό CSP. Στην πράξη, όλοι οι αποδεικτικοί κανόνες που χρησιμοποιήθηκαν θα είναι κανόνες διατήρησης ισοδυναμίας. Έτσι, ιδανικά, χρησιμοποιώντας παραγωγές θα επιθυμούσαμε να πάρουμε ένα λυμένο ή ένα αποτυχημένο CSP, επειδή με αυτόν τον τρόπο θα μπορούσαμε να παραγάγουμε όλες τις λύσεις στο αρχικό CSP ή να δείξουμε ότι καμία λύση δεν υπάρχει. Δυστυχώς, γενικά αυτό είναι αδύνατον. Επομένως προσπαθούμε να επιτύχουμε έναν λιγότερο φιλόδοξο στόχο, δηλαδή μειώνοντας το αρχικό CSP σε ένα που είναι περιορισμένο κάτω από τις εφαρμογές των εξεταζόμενων κανόνων. Αυτό μας φέρνει στις ακόλουθες έννοιες οι οποίες κινούν το ενδιαφέρον μας.

Ορισμός 4.5 Υποθέστε ένα πεπερασμένο σύνολο κανόνων απόδειξης.

- Με μια **παραγωγή** υπολογίζουμε μια ακολουθία CSPs έτσι ώστε κάθε ένα από αυτά να λαμβάνεται από το προηγούμενο μέσω της εφαρμογής ενός αποδεικτικού κανόνα.
- Μια πεπερασμένη παραγωγή καλείται
  - **successful** εάν το τελευταίο στοιχείο της είναι ένα πρώτο πετυχημένο CSP σε αυτήν την παραγωγή,
  - **failed** εάν το τελευταίο στοιχείο της είναι ένα πρώτο αποτυχημένο CSP σε αυτήν την παραγωγή,
  - **stabilising** εάν το τελευταίο στοιχείο της είναι πρώτο CSP σε αυτήν την παραγωγή, το οποίο είναι περιορισμένο κάτω από τις εφαρμογές των εξεταζόμενων κανόνων απόδειξης.

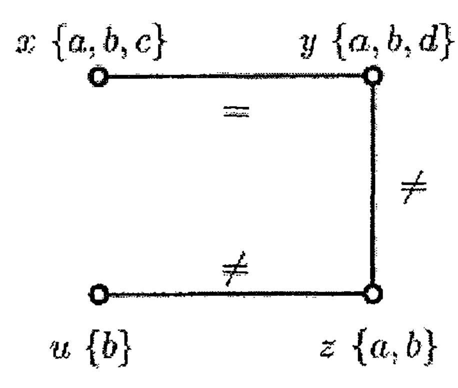
Τα ακόλουθα παραδείγματα επεξηγούν αυτές τις έννοιες.

Παράδειγμα 4.6 Θεωρήστε τους κανόνες ΙΣΟΤΗΤΑΣ και ΑΝΙΣΟΤΗΤΑΣ που εισήχθησαν νωρίτερα σε αυτό το τμήμα.

(i) Αρχικά εξετάστε το CSP

$$\langle x = y, y \neq z, u \neq z \; ; \; x \in \{a, b, c\}, y \in \{a, b, d\}, z \in \{a, b\}, u = b\rangle$$

που απεικονίζεται στο Σχήμα 4.1.



Σχήμα 4.1. Ένα CSP που αποτελείται από τους περιορισμούς ισότητας και ανισότητας

Εφαρμόζοντας τον κανόνα ΙΣΟΤΗΤΑΣ στον περιορισμό x=y λαμβάνουμε το CSP

$$\langle x = y, y \neq z, u \neq z \; ; \; x \in \{a, b\}, y \in \{a, b\}, z \in \{a, b\}, u = b\rangle.$$

Κατόπιν εφαρμόζοντας τον κανόνα ΑΝΙΣΟΤΗΤΑΣ στον περιορισμό $u \neq z$ λαμβάνουμε

$$\langle x = y, y \neq z \ ; \ x \in \{a, b\}, y \in \{a, b\}, z = a, u = b\rangle.$$

Μια άλλη εφαρμογή του κανόνα ΑΝΙΣΟΤΗΤΑΣ , αυτή τη φορά στον περιορισμό $y \neq z$, αποδίδει

$$\langle x = y \ ; \ x \in \{a, b\}, y = b, z = a, u = b\rangle.$$

Τέλος, με τη δεύτερη εφαρμογή του κανόνα ΙΣΟΤΗΤΑΣ στον περιορισμό x=y λαμβάνουμε το λυμένο CSP

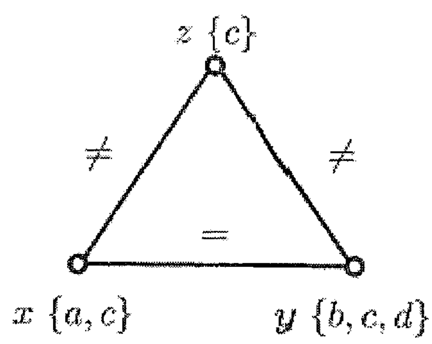
$$\langle \ ; \ x = b, y = b, z = a, u = b\rangle.$$

Έτσι εκθέσαμε μια επιτυχή παραγωγή. Αυτό δείχνει ότι χρησιμοποιώντας τους κανόνες ΙΣΟΤΗΤΑΣ και ΑΝΙΣΟΤΗΤΑΣ θα μπορούσαμε να βρούμε μια μοναδική λύση στο αρχικό CSP, δηλαδή (b, b, a, b).

(ii) Έπειτα, θεωρήστε το CSP

$$\langle \ ; \ x = b, y = b, z = a, u = b\rangle.$$

που απεικονίζεται στο Σχήμα 4.2.



Σχήμα 4.2. Ένα μεταβλητό CSP που αποτελείται από τους περιορισμούς ισότητας και ανισότητας.

Εφαρμόζοντας τον κανόνα ΙΣΟΤΗΤΑΣ λαμβάνουμε εδώ

$$\langle y \neq z, x \neq z \ ; \ x = c, y = c, z = c\rangle.$$

Τώρα εφαρμόζοντας τον κανόνα ΑΝΙΣΟΤΗΤΑΣ στον περιορισμό $y \neq z$ παίρνουμε το αποτυχημένο CSP

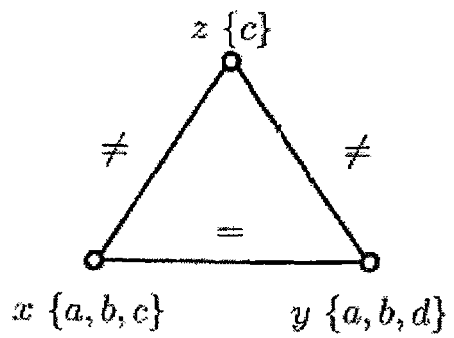
$$\langle x \neq z \; ; \; x = c, y \in \emptyset, z = c \rangle.$$

Έτσι παρουσιάσαμε μια αποτυχημένη παραγωγή. Σημειώστε ότι το τελευταίο CSP δεν είναι περιορισμένο κάτω από τις εφαρμογές του κανόνα ΑΝΙΣΟΤΗΤΑΣ, έτσι αυτή η παραγωγή δεν είναι σταθεροποιήσιμη.

(iii) Τέλος, θεωρήστε το CSP

$$\langle x = y, y \neq z, x \neq z \; ; \; x \in \{a, b, c\}, y \in \{a, b, d\}, z = c \rangle$$

που απεικονίζεται στο Σχήμα 4.3.



Σχήμα 4.3. Ένα σταθερό CSP που αποτελείται από τους περιορισμούς ισότητας και ανισότητας

Εφαρμόζοντας τον κανόνα ΙΣΟΤΗΤΑΣ λαμβάνουμε τώρα

$$\langle x = y, y \neq z, x \neq z \; ; \; x \in \{a, b\}, y \in \{a, b\}, z = c \rangle.$$

Η ΙΣΟΤΗΤΑ μπορεί να εφαρμοστεί πάλι σε αυτό το CSP αλλά η εφαρμογή της δεν είναι σχετική. Περαιτέρω, ο κανόνας ΑΝΙΣΟΤΗΤΑΣ εδώ δεν μπορεί να εφαρμοστεί. Έτσι αυτό το CSP είναι περιορισμένο κάτω από τις εφαρμογές αυτών των δύο κανόνων. Συνεπώς, αυτή η παραγωγή σταθεροποιείται. Σημειώστε επίσης ότι αυτό το CSP δεν είναι ούτε λυμένο ούτε αποτυχημένο. Έτσι αυτή η παραγωγή δεν είναι ούτε πετυχημένη ούτε αποτυχημένη.

Είναι εύκολο να δούμε ότι κάθε επιτυχής παραγωγή σταθεροποιείται. Περαιτέρω, όπως μόλις είδαμε, αυτή δεν είναι η περίπτωση που κάθε αποτυχημένη παραγωγή σταθεροποιείται.

Συνήθως, οι σταθεροποιήσιμες παραγωγές δεν είναι ούτε πετυχημένες ούτε αποτυχημένες. Στην πραγματικότητα, πολλοί επιλυτές περιορισμών παράγουν

CSPs που ούτε λύνονται ούτε αποτυγχάνουν: στόχος τους είναι να φέρουν το αρχικό CSP σε κάποια συγκεκριμένη, απλούστερη μορφή που συνήθως ικανοποιεί κάποια συγκεκριμένη έννοια τοπικής σταθερότητας.

Εξ' ορισμού οι παραγωγές μπορούν να είναι πεπερασμένες ή άπειρες. Φυσικά, στην πράξη θα επιθυμούσαμε να αποφύγουμε τις άπειρες παραγωγές. Μια δυνατότητα είναι να εξασφαλιστεί ότι κάθε άπειρη παραγωγή έχει ένα πρόθεμα που είναι μια σταθεροποιήσιμη (που εντάσσει επιτυχία) ή μια αποτυχημένη παραγωγή. Κατόπιν μπορούμε να αποτρέψουμε μια παραγωγή άπειρων παραγωγών μόλις μπορέσουμε να εξετάσουμε ότι ένα CSP είναι αποτυγχάνει ή περιορίζεται κάτω από τις εφαρμογές των εξεταζόμενων κανόνων απόδειξης.

Έπειτα προκύπτει μια απλή ερώτηση, που αφορά το πώς θα δημιουργηθούν αποτυχημένες ή σταθεροποιήσιμες παραγωγές. Φυσικά, γενικά, τέτοιες παραγωγές μπορεί να μην υπάρχουν. Εντούτοις, όταν όλοι οι κανόνες είναι κανόνες μείωσης περιοχών και το αρχικό CSP έχει πεπερασμένες περιοχές, κατόπιν είναι εύκολο να παραχθούν τέτοιες παραγωγές. Πράγματι, αρκεί τότε να περιοριστεί η προσοχή κάποιου στις σχετικές εφαρμογές κανόνα, δηλ., να εξετάζει μόνο τις παραγωγές στις οποίες κάθε εφαρμογή κανόνα είναι σχετική. Εναλλακτικά, κάποιος μπορεί να σχεδιάσει τους κανόνες με έναν κατάλληλο τρόπο. Με αυτόν τον τρόπο μπορούμε να φθάσουμε σε ένα CSP που είναι περιορισμένο κάτω από τις εφαρμογές των εξεταζόμενων κανόνων. Αυτά τα ζητήματα διαμορφώνονται περαιτέρω στην Άσκηση 4.2.

Μια άλλη απλή ερώτηση είναι υπό ποιους όρους το τελικό CSP είναι μοναδικό. Επίσης, για να αποφευχθούν οι εφαρμογές κανόνα που δεν είναι σχετικές, οι κανόνες σε οποιαδήποτε εφαρμογή πρέπει να επιλεχτούν και να σχεδιαστούν με έναν κατάλληλο τρόπο.

4.2 Εξισώσεις όρου

Ο πρώτος πλήρης επιλυτής που θα συζητήσουμε εξετάζει την επίλυση των πεπερασμένων συνόλων εξισώσεων όρου. Είναι γνωστό ως πρόβλημα ενοποίησης και είναι ύψιστης σημασίας για την αυτοματοποιημένη παρουσίαση αποδείξεων θεωρήματος και για τον λογικό προγραμματισμό.
Για να το ορίσουμε κατάλληλα εισάγουμε στις επόμενες υποενότητες μια γλώσσα όρων, καθώς επίσης την έννοια της αντικατάστασης και ενός γενικότερου συνενωτή. Κατόπιν επιδεικνύουμε πώς το πρόβλημα ενοποίησης μπορεί να διατυπωθεί ως στόχος λύσης ορισμένων τύπων CSPs. Για να λύσουμε το πρόβλημα ενοποίησης εισάγουμε ένα σύνολο κανόνων βάση του θεωρητικού πλαισίου απόδειξης του προηγούμενου μέρους και διατυπώνουμε τον κατάλληλο αλγόριθμο χρησιμοποιώντας αυτούς τους κανόνες.

4.2.1 Όροι

Αρχίζουμε με την εισαγωγή της έννοιας ενός αλφάβητου. Ένα ***αλφάβητο*** αποτελείται από τις εξής ξεχωριστές κατηγορίες συμβόλων:

- ***μεταβλητές,***
- ***σύμβολα συνάρτησης,***
- ***παρενθέσεις,*** οι οποίες είναι: "(" και ")",
- ***κόμμα,*** το οποίο είναι: ", ".

Υποθέτουμε ότι το σύνολο μεταβλητών είναι άπειρο και σταθερό. Δηλώνουμε τις μεταβλητές με x, y, z, u. Αντίθετα, το σύνολο συμβόλων λειτουργίας μπορεί να ποικίλει και ειδικότερα μπορεί να είναι κενό. Κάθε σύμβολο λειτουργίας έχει ένα σταθερό ***αριθμητικό,*** το οποίο είναι ο αριθμός επιχειρημάτων που συνδέονται με αυτό. Τα σύμβολα λειτουργίας 0-ary καλούνται

- ***σταθερές,*** και δηλώνονται με *a, b, c, d.*

Δηλώνουμε τα σύμβολα λειτουργίας του θετικού αριθμητικού με *f, g, h, k.*

Οι όροι ορίζονται επαγωγικά ως εξής:

- μια μεταβλητή είναι ένας όρος,
- εάν to f είναι ένα n- ary σύμβολο συνάρτησης και τα $t_1,\dots,t_n$ είναι όροι, έπειτα το $f(t_1,\dots,t_n)$ είναι ένας όρος.

Ειδικότερα κάθε σταθερά είναι ένας όρος. Οι όροι δηλώνονται με *s, t, u.*

Παράδειγμα 4.7 Θεωρήστε ένα αλφάβητο με δύο σταθερές, *a* και *b,* ένα μοναδιαίο σύμβολο συνάρτησης, f , και ένα δυαδικό σύμβολο συνάρτησης, *g.* Κατόπιν τα ακόλουθα είναι παραδείγματα όρων σε αυτό το αλφάβητο (που χωρίζονται από *";").*

$$f(a);\ g(x,a);\ g(f(a),f(b));\ f(g(a,y)).$$

Αντίθετα, καμία από αυτές τις σειρές συμβόλων δεν είναι όρος:

$$f(x,a);\ g(b);\ f(g(x));\ g(a,f(a,b)).$$

Έτσι το σύνολο των όρων ορίζεται από το σύνολο των συμβόλων συνάρτησης του εξεταζόμενου αλφάβητου. Στη συνέχεια δηλώνουμε με *Var(s)* το σύνολο των μεταβλητών που εμφανίζονται στο s. Λαμβάνοντας υπόψη δύο όρους το *s* και το *t* και γράφουμε $s \equiv t$ για να δηλώσουμε ότι το *s* και το *t* είναι ίδια. Το συχνά χρησιμοποιημένο για αυτόν το λόγο σύμβολο ισότητας "=" είναι προορισμένο για να γραφτούν οι εξισώσεις όρου. Επίσης γράφουμε $s \not\equiv t$ που δείχνει ότι το *s* και το *t* διαφέρουν.

4.2.2 Αντικαταστάσεις

Εξετάστε τώρα ένα σταθερό αλφάβητο και συνεπώς ένα σταθερό σύνολο όρων. Μια **αντικατάσταση** είναι μια πεπερασμένη σχεδίαση από τις μεταβλητές στους όρους που ορίζει σε κάθε περιοχή της μεταβλητής x έναν όρο t διαφορετικό από την x. Την γράφουμε ως εξής:

$$\{x_1/t_1,\dots,x_n/t_n\}$$

όπου

- $x_1,\dots,x_n$ είναι διαφορετικές μεταβλητές,
- $t_1,\dots,t_n$ είναι όροι,
- για i Ε [1,n], $x_i \not\equiv t_i$.

Άτυπα, ως αποτέλεσμα της ταυτόχρονης αντικατάστασης πρόκειται να διαβαστεί: οι μεταβλητές $x_1,\dots,x_n$ αντικαθίσταται ταυτόχρονα από τις $t_1,\dots,t_n$ αντίστοιχα. Ένα ζευγάρι x_i / t_i καλείται ***σύνδεση***. Όταν n =0, η σχεδίαση μετατρέπεται σε ***κενή σχεδίαση***. Η αντικατάσταση που προκύπτει καλείται έπειτα κενή αντικατάσταση και δηλώνεται με ε.

Περαιτέρω, δεδομένης μιας αντικατάστασης $\theta := \{ x_1/t_1,\dots,x_n/t_n\}$ δηλώνουμε

- με Dom (θ), το σύνολο μεταβλητών $\{x_1,\dots,x_n\}$, και
- με Range(θ), το σύνολο όρων $\{t_1,\dots,t_n\}$.

Τώρα ορίζουμε το αποτέλεσμα της ***εφαρμογής μιας αντικατάστασης θ σε έναν όρο*** *s*, γραμμένο σαν *sθ*, ως αποτέλεσμα της ταυτόχρονης αντικατάστασης κάθε εμφάνισης στο *s* μιας μεταβλητής από το *Dom(θ)* με τον αντίστοιχο όρο στη *Range(θ)*. Ονομάζουμε έπειτα *sθ* μία **περίπτωση του** *s*. Περαιτέρω, για δύο αντικαταστάσεις *θ* και *η* γράφουμε *θ* = *η* εάν για όλες τις μεταβλητές *x* έχουμε $x\theta \equiv x\eta$.

Παράδειγμα 4.8 Θεωρήστε μία γλώσσα η οποία μας επιτρέπει να αυξήσουμε τις αριθμητικές εκφράσεις με μορφή προθέματος. Περιέχει δύο δυαδικά σύμβολα λειτουργίας το "+" και το "•" και απείρως πολλές σταθερές: 0, 1,.... Κατόπιν το $s := +(\cdot(x,7),\cdot(4,y))$ είναι ένας όρος και για την αντικατάσταση $\theta := \{ x/0, y/+(z,2)\}$ έχουμε

$$s\theta \equiv +(\cdot(0,7),\cdot(4,+(z,2))).$$

Έπειτα, ορίζουμε την σύνθεση των δύο αντικαταστάσεων.

Ορισμός 4.9 Αφήστε το θ και το γ να είναι αντικαταστάσεις. Η σύνθεσή τους ορίζεται ως εξής. Ορίζουμε πρώτα μια σχεδίαση *η* από τις μεταβλητές στους όρους τοποθετώντας σε μια μεταβλητή *x*

$$\eta(x) := (x\theta)\gamma.$$

Με άλλα λόγια, η *η* ορίζει σε μια μεταβλητή *x* τον όρο που λαμβάνεται με την εφαρμογή της αντικατάστασης *γ* στον όρο *xθ*. Σαφώς, για τη $x \notin Dom(\theta) \cup Dom(\gamma)$ έχουμε $\eta(x) \equiv x$, έτσι η *η* δεν είναι μια ταυτότητα μόνο σε πολλές μετρήσιμες μεταβλητές. Ως εκ τούτου η *η* προσδιορίζει μεμονωμένα μια αντικατάσταση που δηλώνουμε με *θγ* και την καλούμε ***σύνθεση*** του *θ* και *γ*.

Παραδείγματος χάριν, για $\theta = \{u/z,\ x/a,\ y/f(x,b)\}$ και $\gamma = \{x/c,\ z/u\}$ μπορούμε να ελέγξουμε εκείνο το $\theta\gamma = \{x/a,\ y/f(c,b),\ z/u\}$.

Έπειτα, εισάγουμε την ακόλουθη έννοια.

Καθορισμός 4.10 Αφήνουμε το *θ* και το *η* να είναι αντικαταστάσεις. Λέμε ότι η θ είναι **γενικότερη από** τη η εάν για κάποια αντικατάσταση γ έχουμε $\eta = \theta\gamma$.

Παραδείγματος χάριν, η {x/y} είναι γενικότερη από τη {x/a, y/a} αφού {x/y}{y/a} = {x/a, y/a} . Η έννοια "γενικότερη από" είναι αρκετά λεπτή. Σημειώστε παραδείγματος χάριν ότι η αντικατάσταση {x/y} δεν είναι γενικότερη από την {x/a}. Πράγματι, εάν είχαμε {x/y}.γ = {x/a} για κάποια αντικατάσταση γ, έπειτα η δέσμευση y/a θα ανήκει στη γ και συνεπώς στην {x/y}.γ επίσης. Ένα ανάλογο επιχείρημα δείχνει ότι η {x/y} δεν είναι γενικότερη από την κενή αντικατάσταση ούτε της ε.

Κατά συνέπεια η θ είναι γενικότερο από την η εάν η η μπορεί να ληφθεί από τη θ με την εφαρμογή κάποιας αντικατάστασης γ σε αυτό. Δεδομένου ότι η γ μπορεί να επιλεχτεί για να υπάρχει η κενή αντικατάσταση ε, καταλήγουμε στο συμπέρασμα ότι κάθε αντικατάσταση είναι γενικότερη από αυτή καθαυτή.

4.2.3 Συνενωτές και mgus

Όπως δηλώνεται στην αρχή αυτού του μέρους ενδιαφερόμαστε για την επίλυση των πεπερασμένων συνόλων εξισώσεων όρου. Για να ορίσουμε ακριβώς το εξεταζόμενο πρόβλημα πρέπει να εισάγουμε μερικές έννοιες.

Ορισμός 4.11 Θεωρήστε ένα πεπερασμένο σύνολο εξισώσεων όρου (ή, απλά, εξισώσεων) $\{s_1,\ldots,t_1 = t_n\}$.

- Ορίζουμε το αποτέλεσμα της ***εφαρμογής μιας αντικατάστασης*** θ σε ένα τέτοιο σύνολο εξισώσεων με

$$\{s_1 = t_1, \ldots, s_n = t_n\}\theta := \{s_1\theta = t_1\theta, \ldots, s_n\theta = t_n\theta\}.$$

- το θ καλείται ***συνενωτής*** του $\{s_1 = t_1,\ldots, s_n = t_n\}$ εάν

$$s_1\theta \equiv t_1\theta, \ldots, s_n\theta \equiv t_n\theta.$$

  Εάν υπάρχει ένας συνενωτής θ ενός μονοσυνόλου $\{s = t\}$, κατόπιν λέμε ότι το θ είναι ένας ***συνενωτής*** $s = t$. Επίσης λέμε ότι το s και το t ***μπορούν να ενωθούν*** και καλούμε ένα τέτοιο θ ***συνενωτή*** *του s και του t.*

- Ένας συνενωτής θ ενός πεπερασμένου συνόλου εξισώσεων E ονομάζεται ***γενικότερος συνενωτής****(εν συντομία* ***mgu)*** του E εάν είναι γενικότερος από όλους τους συνενωτές του E.

Διαισθητικά, ένα mgu είναι μια αντικατάσταση που σε κάθε εξίσωση κάνει και τους δύο όρους ίδιους αλλά με έναν "γενικότερο τρόπο", χωρίς περιττές συνδέσεις. Σημειώστε ότι εάν ένα πεπερασμένο σύνολο εξισώσεων όρου έχει έναν συνενωτή, κατόπιν έχει έναν άπειρο αριθμό συνενωτών. Πράγματι, εάν το θ είναι ένας συνενωτής ενός τέτοιου συνόλου Ε, κατόπιν το θγ είναι για οποιαδήποτε αντικατάσταση γ. Εντούτοις, αυτό το άπειρο σύνολο αναγνωρίζει μια πεπερασμένη αντιπροσώπευση. Αυτή είναι η ουσία ορισμού του γενικότερου συνενωτή και του τελικού θεωρήματος αυτού του τμήματος.

Ας εξετάσουμε ένα παράδειγμα.

Παράδειγμα 4.12

(i) Θεωρήστε το E := {f(g(x,α),z) = f(y,b)}. Κατόπιν το {x/c, y/g (c, α), z/b} είναι ένας από τους συνενωτές του E και επομένως κι ο {y/g (x, α), z/b} ο οποίος είναι γενικότερος από τον {x/c, y/g(c, α),z/b} μιας και

{x/c, y/g (c, α), z/b}={y/g (x, α), z/b}{x/c}.

Στην πραγματικότητα, κάποιος μπορεί να δείξει ότι το {y/g(x, a), z/b} είναι ένας mgu του E.

(ii) Έπειτα, θεωρήστε την ισότητα {f(g(a,y),z) = f(h(x,b),b)}. Δεν έχει συνενωτή όπως για καμία αντικατάσταση θ δεν έχουμε g(a,y)θ $\equiv$ h (x, b) θ.

(iii) Τέλος, θεωρήστε την ισότητα {g(x, α) = g(f(x),α)}. Δεν έχει συνενωτή ούτε ο ένας ούτε ο άλλος επειδή f(x)θ = φ(xθ) και ως εκ τούτου για οποιαδήποτε αντικατάσταση θ ο όρος xθ είναι ένα κατάλληλο substring του f(x)θ και έτσι διαφέρει από το f(x)θ.

Το πρόβλημα λήψης απόφασης εάν ένα πεπερασμένο σύνολο εξισώσεων όρου έχει έναν συνενωτή καλείται ***πρόβλημα ενοποίησης***. Λύνεται σε περίπτωση που ένας αλγόριθμος τερματίσει με αποτυχία εάν το σύνολο δεν έχει συνενωτή ειδάλλως παράγει έναν πιο γενικό συνενωτή από αυτόν.

Γενικά, ένας συνενωτής μπορεί να μην υπάρχει για δύο λόγους. Ο πρώτος εξηγείται στην παραπάνω κατηγορία (ii) που δείχνει ότι δύο όροι ξεκινώντας από ένα διαφορετικό σύμβολο λειτουργίας δεν μπορούν να ενοποιηθούν. Ο δεύτερος εξηγείται στην παραπάνω κατηγορία (iii) που δείχνει ότι το x και το f(x) (ή γενικότερα, το x και ένας όρος διαφορετικός από το x αλλά στον οποίο το x εμφανίζεται) δεν μπορούν να ενοποιηθούν. Όπως είδαμε, κάθε δυνατότητα μπορεί να εμφανιστεί στο "εσωτερικό επίπεδο" δύο εξεταζόμενων όρων. Αυτές οι δύο δυνατότητες μπορούν να βρεθούν στον αλγόριθμο ενοποίησης που αναφέραμε πριν. Στην απόδειξη ακρίβειας αυτού του αλγόριθμου θα στηριχθούμε πάνω στην ακόλουθη έννοια.

Ορισμός 4.13 Δεδομένου ενός συνόλου εξισώσεων E λέμε ότι μια εξίσωση E είναι σε ***λυμένη μορφή*** εάν είναι της μορφής x = t, όπου x $\notin$ Var(t) και το x δεν εμφανίζεται αλλού στο E. Εάν κάθε εξίσωση στο E είναι σε λυμένη μορφή, λέμε ότι αυτό το E είναι σε **λυμένη μορφή.**

Έτσι ένα πεπερασμένο σύνολο εξισώσεων είναι σε λυμένη μορφή εάν είναι της μορφής

$$\{x_1 = t_1, \ldots, x_n = t_n\},$$

όπου x_is είναι ευδιάκριτες μεταβλητές και καμία από αυτές δεν εμφανίζεται σε έναν όρο tj.

Παράδειγμα 4.14

(i) Το σύνολο εξισώσεων {z = h(g(a)), x = g (α), y = b είναι σε λυμένη μορφή, επειδή οι μεταβλητές z, x και y εμφανίζονται σ' αυτό μόνο μία φορά.

(ii) Αντίθετα, το σύνολο εξισώσεων {z=h(g(z)), x = g(a), y = b} δεν είναι σε λυμένη μορφή, επειδή το z εμφανίζεται στο h(g(z)).

Το ενδιαφέρον για τα πεπερασμένα σύνολα εξισώσεων σε λυμένη μορφή αποκαλύπτεται από το ακόλουθο λήμμα. Η αναφορά σε ένα ισχυρό mgu θα είναι χρήσιμη στην απόδειξη ακρίβειας του αλγόριθμου ενοποίησης.

Λήμμα 4.15 (Λυμένη Μορφή) *Καλούμε ένα mgu θ ενός συνόλου εξισώσεων E* ***ισχυρό*** *εάν για κάθε συνενωτή η του E έχουμε η = θη. Εάν το E := { x_1 = t_1,..., x_n = t_n} είναι σε λυμένη μορφή, έπειτα η αντικατάσταση θ := { { x_1 / t_1,..., xn / tn} είναι ένα ισχυρό mgu του E.*

Απόδειξη Πρώτα σημειώστε ότι το θ είναι ένας συνενωτής του E. Πράγματι, για i E [1,n] έχουμε $x_i\theta \equiv t_i$ και επιπλέον $t_i\theta \equiv t_i$, εφόσον από την υπόθεση κανένα x_j δεν εμφανίζεται στο t_i.

Έπειτα, υποθέστε ότι το η είναι ένας συνενωτής του E. Κατόπιν για i E [1,n] έχουμε $x_i\eta \equiv t_i\eta \equiv x_i\theta\eta$ επειδή $t_i \equiv x_i\theta$ και για $x \notin \{x_1,\ldots,x_n\}$ έχουμε $x\eta \equiv x\theta\eta$ επειδή $x \equiv x\theta$. Έτσι δείξαμε ότι για όλες τις μεταβλητές x έχουμε $x\eta \equiv x\theta\eta$. Κατά συνέπεια ο ορισμός η = ηθ, δηλαδή, ο θ είναι ένας ισχυρός mgu.

Λέμε έπειτα ότι το E **ορίζει** το θ. Παραδείγματος χάριν, το σύνολο εξισώσεων που ήδη αναφέραμε

$$\{z = h(g(a)), x = g(a), y = b\}$$

ορίζει την αντικατάσταση

$$\{z/h(g(a)), x/g(a), y/b\}.$$ ε

4.2.4 Πρόβλημα ενοποίησης και επίλυση των CSPs

Πιστοί στο πνεύμα αυτού του βιβλίου εξετάζουμε το πρόβλημα ενοποίησης διατυπώνοντας το ως στόχο λύσης ενός CSP. Έτσι αρχικά το αναπαριστάνουμε ως CSP και έπειτα το λύνουμε μετασχηματίζοντας το κατ' επανάληψη έως ότου φθάσουμε σε ένα αποτυχημένο CSP ή ένα CSP από το οποίο είναι απλό να παράγουμε όλες τις λύσεις.

Εξηγούμε τώρα πώς βλέπουμε τα πεπερασμένα σύνολα εξισώσεων όρου ως CSP. Ορίστε ένα αλφάβητο. Υποθέτουμε ότι οι μεταβλητές κυμαίνονται πέρα από την ίδια περιοχή, το σύνολο όλων των όρων, δηλώνεται με T.

Έπειτα, ερμηνεύουμε τις εξισώσεις όρου ως περιορισμούς. Θεωρήστε μια εξίσωση s = t με τις μεταβλητές $x_1,\ldots,x_n$ που εμφανίζονται σε αυτό. Ερμηνεύουμε το s = t ως ακόλουθο υποσύνολο του Καρτεσιανού προϊόντος T^n:

$$\{(x_1\eta,\ldots,x_n\eta) \mid \text{το η είναι ένας συνενωτής του s και του t}\}.$$

Παράδειγμα 4.16

(i) θεωρήστε την εξίσωση x = f(y).Την ερμηνεύουμε ως το ακόλουθο υποσύνολο του T^2:

$$\{(x\eta, y\eta) \mid \text{το } \eta \text{ είναι ένας συνενωτής του x και του f(y)}\}.$$

Κάποιος μπορεί να δείξει ότι το η είναι ένας συνενωτής του x και του f(y) εάν είναι της μορφής {x/f(y)}γ, όπου το γ είναι μια αυθαίρετη αντικατάσταση. Έτσι το παραπάνω σύνολο είναι ίσο με

$$\{(f(y)\gamma, y\gamma) \mid \text{το } \gamma \text{ είναι μια αντικατάσταση}\}.$$

Εναλλακτικά, μπορούμε να γράψουμε αυτό το σύνολο ως

$$\{(f(t), t) \mid \text{το t είναι ένας όρος} .$$

Αυτό οδηγεί σε μια μοναδική ερμηνεία ενός πεπερασμένου συνόλου εξισώσεων όρου Ε ως CSP. Η ακόλουθη απλή παρατήρηση στη συνέχεια συνδέει τις έννοιες μιας λύσης και ενός συνενωτή σε ένα CSP. Για ένα CSP P γράφουμε εδώ Sol(P) για να δείξουμε το σύνολο όλων των λύσεων στο P.

Σημείωση 4.17 (Ενοποίηση) *Θεωρήστε ένα πεπερασμένο σύνολο εξισώσεων όρου Ε με τις μεταβλητές $x_1,\ldots, x_n$. Κατόπιν*

$$Sol(\langle E\ ;\ x_1 \in \mathcal{T}, \ldots, x_n \in \mathcal{T}\rangle) = \{(x_1\eta, \ldots, x_n\eta) \mid \text{το } \eta \text{ είναι ένας συνενωτής του } E.$$

Απόδειξη Είναι μια άμεση συνέπεια του τρόπου που ερμηνεύουμε τις εξισώσεις ως περιορισμούς.

Αυτή η παρατήρηση μας επιτρέπει να περιορίσουμε το πρόβλημα εύρεσης ενός συνενωτή στο πρόβλημα λύσης ενός CSP.

4.2.5 Το σύστημα απόδειξης *UNIF*

Για να λύσουμε το πρόβλημα ενοποίησης εισάγουμε τώρα έξι κανόνες μετασχηματισμού. Αυτοί οι κανόνες δεν εξαρτώνται από τις εκφράσεις περιοχών, γι' αυτό αναφέρονται μόνο στους περιορισμούς, δηλ., στις εξισώσεις. Αυτοί οι κανόνες ανέρχονται σε ορισμένους συντακτικούς χειρισμούς των περιορισμών, συνεπώς στην πράξη εφαρμόζονται στις συντακτικές αντιπροσωπεύσεις των εξεταζόμενων περιορισμών.

ΑΝΑΛΥΣΗ

$$\frac{f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)}{s_1 = t_1, \ldots, s_n = t_n}$$

ΑΠΟΤΥΧΙΑ 1

$$\frac{f(s_1,\ldots,s_n)=g(t_1,\ldots,t_m)}{\perp}$$

όπου f $\not\equiv$g

ΔΙΑΓΡΑΦΗ

$$\frac{x=x}{}$$

ΜΕΤΑΚΙΝΗΣΗ

$$\frac{t=x}{x=t}$$

όπου το t δεν είναι μία μεταβλητή,

ΑΝΤΙΚΑΤΑΣΤΑΣΗ

$$\frac{x=t,\ E}{x=t,\ E\{x/t\}}$$

όπου x $\notin$ Var(t) και x $\in$ var(E),

ΑΠΟΤΥΧΙΑ 2

$$\frac{x=t}{\perp}$$

όπου x $\in$ Var(t) και x$\neq$y.

Θυμηθείτε ότι στους κανόνες η σειρά των περιορισμών είναι άσχετη, έτσι στον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ η σειρά των εξεταζόμενων εξισώσεων είναι ασήμαντη. Δηλώστε το σύνολο των παραπάνω έξι κανόνων σύμφωνα με το UNIF. Για να επεξηγήσουμε τη χρήση αυτών των κανόνων θεωρήστε ένα παράδειγμα. Δεδομένου ότι οι περιοχές των μεταβλητών είναι πάντα ίσες με το σύνολο όλων των όρων, T, αφήνουμε τις αναφορές στις εκφράσεις περιοχών και εξετάζουμε μόνο τα σύνολα εξισώσεων. Οι επιλεγμένες εξισώσεις υπογραμμίζονται.

Παράδειγμα 4.18

(i) Θεωρήστε το σύνολο εξισώσεων

$$\{k(z, f(x, b, z)) = k(h(x), f(g(a), y, z))\}.$$

Χρησιμοποιώντας τον κανόνα ΑΝΑΛΥΣΗΣ παίρνουμε

$$\{z = h(x), \underline{f(x, b, z) = f(g(a), y, z)}\}.$$

Χρησιμοποιώντας τον κανόνα ΑΝΑΛΥΣΗΣ πάλι παίρνουμε

$$\{z = h(x), x = g(a), b = y, \underline{z = z}\}.$$

Χρησιμοποιώντας τον κανόνα ΔΙΑΓΡΑΦΗΣ μπορούμε να απαλλαχτούμε από την τελευταία εξίσωση. Αυτό παράγει

$$\{z = h(x), x = g(a), \underline{b = y}\}.$$

Από τον κανόνα ΜΕΤΑΚΙΝΗΣΗΣ παίρνουμε

$$\{\underline{z = h(x), x = g(a)}, y = b\}.$$

Χρησιμοποιώντας τον κανόνα *ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ* με x = g(α) ως απομονωμένη εξίσωση παίρνουμε

$$\{z = h(g(a)), x = g(a), y = b\}.$$

Σε αυτή τη φάση κανένας κανόνας δεν ισχύει κι έτσι φθάσαμε σε ένα CSP που είναι περιορισμένο κάτω από τις εφαρμογές των κανόνων UNIF.

(ii) Θεωρήστε τώρα το σύνολο εξισώσεων

$$\{k(z, f(x, b, z)) = k(h(x), f(g(z), y, z))\}.$$

Η μόνη διαφορά μεταξύ αυτού του συνόλου και εκείνου που εξετάζεται στο παράδειγμα (i) είναι ότι αντί του όρου g(α) εδώ εμφανίζεται ο όρος g(z). Ας προσπαθήσουμε να επαναλάβουμε τις επιλογές που γίνονται στο παράδειγμα (i). Από τον κανόνα ΑΝΑΛΥΣΗΣ παίρνουμε

$$\{z = h(x), \underline{f(x, b, z) = f(g(z), y, z)}\}.$$

Από μια άλλη εφαρμογή του κανόνα ΑΝΑΛΥΣΗΣ παίρνουμε

$$\{z = h(x), x = g(z), b = y, \underline{z = z}\}.$$

Χρησιμοποιώντας τον κανόνα ΔΙΑΓΡΑΦΗΣ μπορούμε να απαλλαχτούμε από την τελευταία εξίσωση. Αυτό παράγει

$$\{z = h(x), x = g(z), \underline{b = y}\}.$$

Από τον κανόνα ΜΕΤΑΚΙΝΗΣΗΣ παίρνουμε τώρα

$$\{\underline{z = h(x), x = g(z)}, y = b\}.$$

Έπειτα, από τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με x = g(z) ως απομονωμένη εξίσωση παίρνουμε

$$\{\underline{z = h(g(z))}, x = g(z), y = b\}.$$

Ωστόσο, τώρα, ισχύει ο κανόνας ΑΠΟΤΥΧΙΑ 2 και παίρνουμε το σύνολο

$$\{\perp, x = g(z), y = b\}$$

το οποίο περιέχει τον λανθασμένο περιορισμό $\perp$. Εξ' ορισμού, κανένας κανόνας δεν μπορεί να εφαρμοστεί σε ένα αποτυχημένο CSP, επομένως το τελευταίο σύνολο είναι περιορισμένο κάτω από τις εφαρμογές των κανόνων UNIF.

Δυστυχώς, εάν χρησιμοποιήσουμε ως αλγόριθμο τους κανόνες του UNIF εφαρμοσμένους με έναν αυθαίρετο τρόπο, αυτός ο αλγόριθμος μπορεί να αποκλίσει, όπως παρουσιάζει το ακόλουθο παράδειγμα. Όπως πριν οι επιλεγμένες εξισώσεις υπογραμμίζονται.

Παράδειγμα 4.19 Θεωρήστε το σύνολο εξισώσεων

$$\{\underline{x = f(y)}, y = g(x), \underline{x = a}\}.$$

Χρησιμοποιώντας τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με x = f(y) ως απομονωμένη εξίσωση παίρνουμε

$$\{x = f(y), \underline{y = g(x), f(y) = a}\}.$$

Χρησιμοποιώντας πάλι τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ, αυτή τη φορά με y = g(x) ως απομονωμένη εξίσωση, παίρνουμε

$$\{\underline{x = f(y)}, y = g(x), \underline{f(g(x)) = a}\}.$$

Επαναλαμβάνοντας αυτήν την διαδικασία παίρνουμε μια άπειρη παραγωγή. Σημειώστε επίσης ότι η τρίτη εξίσωση αλλάζει συχνά γι΄ αυτό κάθε εφαρμογή κανόνα είναι σχετική και συνεπώς κανένα πρόθεμα αυτής της άπειρης παραγωγής δεν είναι σταθεροποιήσιμη παραγωγή.

Έτσι ο κανόνας ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ ως έχει προκαλεί προβλήματα. Προτού να εξετάσουμε μια πιθανή διόρθωση ας αποδείξουμε ότι όταν δεν υπάρχει απόκλιση το σύστημα απόδειξης UNIF μας επιτρέπει να λύσουμε το αρχικό σύνολο εξισώσεων όρου. Χρειαζόμαστε πρώτα το ακόλουθο λήμμα.

Λήμμα 4.20 (UNIF) *Κάθε κανόνας UNIF είναι κανόνας διατήρησης ισοδυναμίας w.r.t. η ακολουθία των μεταβλητών που βρίσκονται στην προϋπόθεση κανόνα.*

Απόδειξη Εξαιτίας της Σημείωσης Ενοποίησης 4.17 αρκεί να αποδείξουμε ότι τα σύνολα εξισώσεων που εξετάζονται στην αρχή και στο τέλος κάθε κανόνα έχουν το ίδιο σύνολο συνενωτών.

Αυτή η αξίωση ισχύει για τον κανόνα ΑΝΑΛΥΣΗΣ επειδή για όλα τα θ έχουμε $f(s_1,\dots,sn)\theta \equiv f(t_1,\dots,t_n)\theta$ εάν για i E[1, n] ισχύει ότι $s_i\theta = t_i\theta$. Για τους κανόνες ΔΙΑΓΡΑΦΗΣ και ΜΕΤΑΚΙΝΗΣΗΣ η αξίωση είναι προφανής.

Για τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ θεωρήστε δύο σύνολα εξισώσεων {x=t} U E και {x=t} U E{x/t} και θεωρήστε δεδομένο ότι x $\notin$ Var (t).

Έπειτα

	θ είναι ένας συνενωτής της {x=t} U E
Εάν	θ είναι ένας συνενωτής της x=t και θ είναι ένας συνενωτής του E.
Εάν	{Λήμμα 4.15 (Λυμένη Μορφή)}
	θ είναι ένας συνενωτής της x=t και {x/t}θ είναι ένας συνενωτής του E.
Εάν	θ είναι ένας συνενωτής της x=t και θ είναι ενας συνενωτής της E{x/t}.
Εάν	θ είναι ένας συνενωτής της {x=t} U E{x/t}.

Τέλος, για τους κανόνες ΑΠΟΤΥΧΙΑ 1 και ΑΠΟΤΥΧΙΑ 2 εξακολουθεί να ισχύει η αξίωση αφού σε κάθε περίπτωση δεν υπάρχει συνενωτής για την εξίσωση στην προϋπόθεση κανόνα.

Μπορούμε τώρα να αποδείξουμε το σχετικό αποτέλεσμα.

Θεώρημα 4.21 (UNIF) *Θεωρήστε μια αποτυχημένη ή μια σταθεροποιήσιμη παραγωγή στο σύστημα απόδειξης UNIF αρχίζοντας από ένα πεπερασμένο σύνολο εξισώσεων E και τελειώνοντας με ένα σύνολο περιορισμών F. Εάν το E έχει έναν συνενωτή, έπειτα το F είναι σε λυμένη μορφή η οποία ορίζει ένα mgu του E ειδάλλως το F περιέχει τον λανθασμένο περιορισμό* $\perp$.

Απόδειξη Εάν το F δεν περιέχει τον λανθασμένο περιορισμό $\perp$, το τελικό σύνολο περιορισμών F είναι περιορισμένο κάτω από τις εφαρμογές όλων των κανόνων UNIF. Κατόπιν στην περιγραφή των κανόνων ΑΝΑΛΥΣΗΣ, ΑΠΟΤΥΧΙΑΣ 1 και ΜΕΤΑΘΕΣΗΣ η αριστερή πλευρά κάθε εξίσωσης στο F είναι μια μεταβλητή. Περαιτέρω, στην περιγραφή των κανόνων ΔΙΑΓΡΑΦΗΣ, ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ και ΑΠΟΤΥΧΙΑΣ 2 αυτές οι μεταβλητές είναι ευδιάκριτες και καμία από αυτές δεν εμφανίζεται στη δεξιά πλευρά μιας εξίσωσης στο F. Αυτό υπονοεί ότι το F είναι σε λυμένη μορφή. Από την Σημείωση 4.17 (Ενοποίηση) και το Λήμμα 4.20 (UNIF) το F έχει το ίδιο σύνολο συνενωτών με το αρχικό ορισμένο E. Έτσι, στην περιγραφή του Λήμματος 4.15 (Λυμένης Μορφής), το F ορίζει ένα mgu του E.

Εάν το τελικό σύνολο περιορισμών F περιέχει τον λανθασμένο περιορισμό $\perp$, κατόπιν η τελευταία επιλεγμένη εξίσωση είναι καθένα από τα $f(s_1,\ldots, s_n) = g(t_1,\ldots,t_n)$ με $f \not\equiv g$, ή $x=t$ με $x \in Var(t)$ και $x \not\equiv t$. Και στις δύο περιπτώσεις η επιλεγμένη εξίσωση δεν έχει συνενωτή. Σύμφωνα με την Σημείωση 4.17 (Ενοποίηση) και το Λήμμα 4.20 (UNIF) δεν έχουν συνενωτή.

Αυτό υπονοεί την αξίωση.

4.2.6 Ο αλγόριθμος MARTELLI-MONTANARI

Για να εξασφαλίσουμε τον τερματισμό εισάγουμε την ακόλουθη έννοια, όπου αναφερόμαστε στους αυθαίρετους κανόνες όπως εξετάζονται στην Παράγραφο 4.1.

Ορισμός 4.22 Θεωρήστε ένα CSP P := (C; DE) κι έναν κανόνα

$$\frac{\langle \mathcal{C} \,;\, \mathcal{DE} \rangle}{\langle \mathcal{C}' \,;\, \mathcal{DE}' \rangle}$$

Κατόπιν το (C΄; DE΄) είναι το αποτέλεσμα της εφαρμογής αυτού του κανόνα στο P. Καλούμε αυτόν τον κανόνα ***γενική*** εφαρμογή.

Έτσι στη κανόνα γενικής εφαρμογής όλοι οι περιορισμοί και οι εκφράσεις περιοχών που υπάρχουν στο εξεταζόμενο CSP χρησιμοποιούνται στο εισαγωγικό μέρος(πρόταση) του κανόνα.

Παράδειγμα 4.23 Επανεξετάστε το σύνολο εξισώσεων

$$\{x = f(y), y = g(x), x = a\}$$

του Παραδείγματος 4.19. Αρχίζουμε με

$$\underline{\{x = f(y), y = g(x), x = a\}},$$

όπου το x = f(y) είναι η απομονωμένη εξίσωση. Χρησιμοποιώντας τον κανόνα

ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ παίρνουμε

$$\{x = f(y), \underline{y = g(f(y))}, f(y) = a\}.$$

Τώρα, χρησιμοποιώντας τον κανόνα ΑΠΟΤΥΧΙΑ 2 παίρνουμε

$$\{x = f(y), \perp, f(y) = a\}$$

και η παραγωγή ολοκληρώνεται με ένα αποτυχημένο CSP.

Σημειώστε ότι η εξεταζόμενη εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ ήταν γενική. Αντίθετα, οι εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ που εξετάστηκαν στο Παράδειγμα 4.19 δεν ήταν γενικές: κάθε φορά μια εξίσωση "παραλείπονταν". Σημειώστε επίσης ότι και οι εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ που εξετάστηκαν στο Παράδειγμα 4.18 δεν ήταν γενικές. Εντούτοις, ανήλθαν σε γενικές εφαρμογές μόνο αφού που εμφανιζόταν κάθε φορά η μεταβλητή η οποία αντικαταστάθηκε, σε μια άλλη εξίσωση. Αντίθετα, οι εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ που εξετάζεται στο Παράδειγμα 4.19 δεν ισοδυναμούν με τις γενικές εφαρμογές αφού εμφανίζονταν κάθε φορά οι μεταβλητές οι οποίες αντικαταστάθηκαν, σε δύο άλλες εξισώσεις.

Έτσι η χρήση των γενικών εφαρμογών του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μπορεί να κάνει τη διαφορά. Στην πραγματικότητα, εάν περιοριζόμαστε σε τέτοιες εφαρμογές αυτού του κανόνα, όλες οι παραγωγές είναι πεπερασμένες. Πιο συγκεκριμένα, ισχύει το ακόλουθο λήμμα.

Λήμμα 4.24 (Τερματισμός) *Κάθε παραγωγή στο UNIF στο οποίο όλες οι εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ είναι γενικές είναι πεπερασμένη.*

Απόδειξη Η απόδειξη είναι μάλλον περίπλοκη επειδή οι κανόνες ΑΝΑΛΥΣΗΣ και ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ φαίνεται να έρχονται "σε σύγκρουση" μεταξύ τους. Ο πρώτος καθιστά τις δομικά απλούστερες εξισώσεις ενώ ο δεύτερος κανόνας μπορεί να καταστήσει πιό σύνθετες τις μεμονωμένες εξισώσεις.

Μια λύση είναι να διατάξουμε με έναν κατάλληλο τρόπο τις σχετικές ποσότητες τις οποίες υποτίθεται ότι θα μειώσουμε. Για αυτόν τον σκοπό χρησιμοποιούμε την ακόλουθη σχέση $<_3$ η οποία ορίζεται στα τριπλάσια των φυσικών αριθμών:

$$(a_1, a_2, a_3) \prec_3 (b_1, b_2, b_3)$$

εάν

$$a_1 < b_1$$

ή $a_1 = b_1$ και $a_2 < b_2$

ή $a_1 = b_1$ και $a_2 = b_2$ και $a_3 < b_3$.

Παραδείγματος χάριν, έχουμε (1,15,10000) $<_3$ (2,1,1), (1,15,10000)$<_3$(1,16,1) και (1,16,1) $<_3$ (1,16,10000).

Αυτή η σχέση, αποκαλούμενη ***λεξικογραφική σειρά*** (στα τριπλάσια εδώ των φυσικών αριθμών), είναι ***καλά θεμελιωμένη*** που σημαίνει ότι δεν υπάρχει καμία άπειρη $<_3$-κάτω ακολουθία των τριπλάσιων των φυσικών αριθμών.

Λαμβάνοντας υπόψη ένα σύνολο εξισώσεων *E*, καλούμε μια μεταβλητή *x* ***λυμένη στο*** *E* εάν για κάποιο όρο *t* έχουμε $x = t \in E$ και αυτή είναι η μόνη εμφάνιση *του x στο E,* δηλαδή εάν η εξίσωση $x = t$ είναι σε λυμένη μορφή. Καλούμε μια μεταβλητή ***άλυτη στο*** *E* εάν δεν λύνεται.

Παραδείγματος χάριν για *το* $E := \{x = f(y), y = g(b), a = z$ η μεταβλητή *x* λύνεται *στο E* ενώ οι μεταβλητές *y* και *z* παραμένουν άλυτες.

Με κάθε σύνολο εξισώσεων *E* συνδέουμε τώρα τις τρεις ακόλουθες λειτουργίες:

uns (E) - ο αριθμός των μεταβλητών που είναι άλυτες *στο E,*

lfun (E) - ο συνολικός αριθμός εμφάνισης των συμβόλων λειτουργίας στην αριστερή πλευρά μιας εξίσωσης *στο E,*

card (E) - ο αριθμός εξισώσεων στο *E.*

Υποστηρίζουμε ότι κάθε εφαρμογή κανόνα σε ένα σύνολο εξισώσεων *E* μειώνει το τριπλάσιο των φυσικών αριθμών

$$(uns(E), lfun (E), card(E))$$

στη λεξικογραφική σειρά $<_3$. Στην περίπτωση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ υποθέτουμε ότι εξετάζουμε μια γενική εφαρμογή.

Πράγματι, καμία εφαρμογή κανόνα δεν μετατρέπει μια λυμένη μεταβλητή σε μια άλυτη, έτσι το uns(E) δεν αυξάνεται ποτέ. Περαιτέρω, κάθε εφαρμογή των κανόνων ΑΝΑΛΥΣΗΣ και ΜΕΤΑΘΕΣΗΣ μειώνει το lfun(E) τουλάχιστον μέχρι 1. Στη συνέχεια, καμία εφαρμογή των κανόνων ΑΠΟΤΥΧΙΑΣ 1, ΔΙΑΓΡΑΦΗΣ και ΑΠΟΤΥΧΙΑΣ 2 δεν αλλάζει ή μειώνει το lfun (E) και ελαττώνει το card(E) μέχρι 1.

Τέλος, κάθε γενική εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μειώνει το uns(E) μέχρι 1. Πράγματι, "γενική" σημαίνει ότι τη στιγμή της εφαρμογής του κανόνα το $\{x = t\} \cup E$ είναι το σύνολο όλων των εξισώσεων. Έτσι αυτή η εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μετατρέπει το x από μια άλυτη μεταβλητή σε μια λυμένη. (Σημειώστε ότι μια αυθαίρετη εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ δεν πρέπει να μειώνει το uns (E); δείτε παραδείγματος χάριν τις εφαρμογές αυτού του κανόνα στο Παράδειγμα 4.19.)

Το τελικό αποτέλεσμα τώρα είναι το συμπέρασμα του well- foundedness $<_3$.

Μπορούμε τώρα να παρουσιάσουμε τον επιθυμητό αλγόριθμο. Εξετάστε τους έξι κανόνες του συστήματος UNIF αλλά περιορίστε τη χρήση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ στις γενικές εφαρμογές. Όπως εξηγείται στην Παράγραφο 4.1 αυτοί οι κανόνες ορίζουν έναν επιλυτή περιορισμών, σε αυτήν την περίπτωση έναν αλγόριθμο ενοποίησης. Έτσι ο αλγόριθμος ενοποίησης αποτελείται από τους έξι πιο πάνω κανόνες μαζί με έναν χρονοπρογραμματιστή που τους εφαρμόζει κατ' επανάληψη μέχρι να επιτευχθεί ένα αποτυχημένο CSP ή ένα CSP περιορισμένο κάτω από τις εφαρμογές αυτών των κανόνων. Στην περίπτωση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ εκτελούνται μόνο οι γενικές εφαρμογές.

Αυτός ο αλγόριθμος καλείται αλγόριθμος MARTELLI-MONTANARI. Μόλις δούμε τις εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ ως γενικές εφαρμογές, οι παραγωγές που παρουσιάζονται στο Παράδειγμα 4.18 είναι παραδείγματα εκτελέσεων αυτού του αλγόριθμου. Το ακόλουθο αποτέλεσμα καθιερώνει την ακρίβειά του.

Θεώρημα 4.25 (**Αλγόριθμος** MARTELLI-MONTANARI) *Ο αλγόριθμος MARTELLI-MONTANARI ολοκληρώνεται πάντα. Εάν το αρχικό πεπερασμένο σύνολο εξισώσεων E έχει έναν συνενωτή, κατόπιν κάθε εκτέλεση του αλγόριθμου ολοκληρώνεται με ένα σύνολο εξισώσεων σε λυμένη μορφή που ορίζει ένα mgu του E ειδάλλως κάθε εκτέλεση ολοκληρώνεται με ένα σύνολο που περιέχει τον λανθασμένο περιορισμό* $\perp$.

Απόδειξη Είναι μια άμεση συνέπεια του Λήμματος 4.24 *ΤΕΡΜΑΤΙΣΜΟΣ* και το Θεωρήματος 4.21 *UNIF*.

Ας διευκρινίσουμε τώρα γιατί ο αλγόριθμος MARTELLI-MONTANARI μπορεί να αντιμετωπισθεί ως πλήρης επιλυτής περιορισμών. Στην Υποενότητα 4.2.4 εξηγήσαμε πώς το κάθε σύνολο εξισώσεων όρου E με τις μεταβλητές $x_1,\ldots, x_n$ αντιστοιχεί σε ένα μοναδικό CSP (E; x_1 E T,..., x_n E T) και συνόψισαμε αυτήν την σχέση με τη Σημείωση Ενοποίησης 4.17.

Συγκεκριμένα, για κάθε πεπερασμένο σύνολο εξισώσεων E := $\{x_1 = t_1,\ldots,x_n = t_n\}$ σε λυμένη μορφή έχουμε

$$Sol(\langle E\ ;\ x_1 \in T,\ldots,x_n \in T\rangle) = \{(t_1\eta,\ldots,t_n\eta) \mid \eta \text{ is a substitution}\},$$

δεδομένου ότι από το Λήμμα 4.15 (Λυμένη Μορφή) καθένας συνενωτής του E είναι της μορφής $\{x_1/t_1,\ldots x_n/t_n\}\eta$ για μια αυθαίρετη αντικατάσταση η. Έτσι είναι απλό να παράγουμε όλες τις λύσεις (E; x_1 E T,..., x_n E T) εάν το E είναι σε λυμένη μορφή.

Τώρα, λαμβάνοντας υπόψη το παραπάνω θεώρημα, ο αλγόριθμος MARTELLI-MONTANARI μετασχηματίζει ένα δεδομένο πεπερασμένο σύνολο εξισώσεων όρου σε ένα ισοδύναμο σύνολο εξισώσεων που είναι σε λυμένη μορφή ή περιέχει τον λανθασμένο περιορισμό $\perp$. Στην ορολογία που υιοθετείται στην αρχή αυτού του κεφαλαίου εννοείται ότι ο αλγόριθμος αυτός είναι πλήρης συνενωτής περιορισμών.

Είναι χρήσιμο να επισημανθεί ότι ο αλγόριθμος MARTELLI-MONTANARI είναι ανεπαρκής, δεδομένου ότι για μερικές εισαγωγές μπορεί να πάρει έναν εκθετικό χρόνο για να υπολογιστεί ένα mgu. Ένα τυποποιημένο παράδειγμα είναι το ακόλουθο ζευγάρι δύο όρων, όπου n > 0:

$$f(x_1,\ldots,x_n) \text{ and } f(g(x_0,x_0),\ldots,g(x_{n-1},x_{n-1})).$$

Ορίστε τώρα επαγωγικά μια ακολουθία όρων $t_1,\ldots,t_n$ ως εξής:

$$t_1 := g(x_0,x_0),$$

$$t_{i+1} := g(t_i,t_i).$$

Είναι εύκολο να ελεγχθεί ότι {x1/t1,...xn/tn} είναι έπειτα ένα mgu των όρων f(x1,...,xn) και f(g(x0,x0),...,g(xn-1,xn-1)).

Παραδείγματος χάριν, για n = 2 το ζευγάρι των όρων που αναφέραμε είναι

$$f(x_1,x_2) \text{ and } f(g(x_0,x_0),g(x_1,x_1))$$

και το mgu που αναφέραμε είναι

$$\{x_1/g(x_0,x_0), x_2/g(g(x_0,x_0), g(x_0,x_0))\}.$$

Τώρα, μια απλή απόδειξη με επαγωγή δείχνει ότι το κάθε t_i έχει περισσότερο από 2^i σύμβολα. Αυτό δείχνει ότι ο συνολικός αριθμός συμβόλων σε οποιοδήποτε mgu των δύο παραπάνω όρων είναι εκθετικός στο μέγεθός του. Έτσι εφ' όσον στον αλγόριθμο MARTELLI-MONTANARI αντιπροσωπεύονται οι όροι ως σειρές αυτός ο αλγόριθμος τρέχει στον εκθετικό χρόνο.

Τέλος, σημειώστε ότι ο mgu των δύο παραπάνω όρων μπορεί να υπολογιστεί χρησιμοποιώντας n εφαρμογές κανόνα του αλγόριθμου MARTELLI-MONTANARI. Αυτό δείχνει ότι ο αριθμός εφαρμογών του κανόνα που χρησιμοποιούνται σε μια εκτέλεση του αλγόριθμου MARTELLI-MONTANARI δεν είναι το σωστό μέτρο χρονικής πολυπλοκότητας αυτού του αλγόριθμου. Οι αποδοτικότεροι αλγόριθμοι ενοποίησης αποφεύγουν τις ρητές παρουσιάσεις των γενικότερων συνενωτών και στηρίζονται σε διαφορετική εσωτερική αντιπροσώπευση των όρων από τις σειρές.

4.3 Γραμμικές εξισώσεις πέρα από τους πραγματικούς

Σε αυτό το τμήμα παρουσιάζουμε δύο πλήρεις συνενωτές περιορισμών που εξετάζουν τις γραμμικές εξισώσεις πέρα από τα τους πραγματικούς. Για να τους αναφέρουμε ορίζουμε στην επόμενη υποενότητα τη σύνταξη τους και αναφέρουμε τις γραμμικές εξισώσεις στα CSPs πέρα από τους πραγματικούς. Κατόπιν εισάγουμε ένα σύνολο κανόνων βάση της Παραγράφου 4.1 και αναφέρουμε δύο αλγόριθμους για τα πεπερασμένα σύνολα γραμμικών εξισώσεων ονομάζοντας αυτούς τους κανόνες: αλγόριθμο GAUSS_JORDAN ELIMINATION και αλγόριθμο GAUSSIAN ELIMINATION.

4.3.1 Γραμμικές εκφράσεις και γραμμικές εξισώσεις

Θεωρήστε ένα αλφάβητο έτσι ώστε

- κάθε πραγματικός αριθμός να είναι μια σταθερά.
- για κάθε πραγματικό αριθμό r να υπάρχει ένα ενιαίο μοναδιαίο σύμβολο λειτουργίας 'r.' (που αντιπροσωπεύει τον πολλαπλασιασμό με r),
- να υπάρχει ένα ενιαίο δυαδικό σύμβολο λειτουργίας '+', γραμμένο στη σημείωση επενθημάτων.

Από μια **γραμμική έκφραση πέρα από τους πραγματικούς** υπολογίζουμε έναν όρο που διαμορφώνεται σε αυτό το αλφάβητο και από μια **γραμμική εξίσωση πέρα από τους πραγματικούς** μια εξίσωση της μορφής

$$s = t,$$

όπου το s και το t είναι γραμμικές εκφράσεις. Από εδώ και στο εξής παραλείπουμε τον τίτλο "πέρα από τους πραγματικούς ".

Για να διευκολύνουμε την ανάγνωση των γραμμικών εκφράσεων και εξισώσεων εισάγουμε διάφορους συντακτικούς κανόνες. Συντέμνουμε το —1 · (s) σε —(s), μειώνουμε τις αναφορές σε 1· , και συντέμνουμε το *r · (x)* σε *rx,* όπου το r είναι διαφορετικό από 1 και —1 και το *x* είναι μια μεταβλητή. Γράφουμε επίσης τα *s* + *(t* + *u)* και *(s* + *t)* + *u* ως *s* + *t* + *u* και συντέμνουμε το *s* + *(—t)* σε *s —t.* Παρουσία αυτών των δύο κανόνων τα *4x* + 3.5y και *3x*— 1.2 ·(2 +*2.5y)*— *2x* + 5 είναι γραμμικές εκφράσεις και η

$$4x + 3.5y = 3x - 1.2 \cdot (2 + 2.5y) - 2x + 5$$

είναι μια γραμμική εξίσωση. Σημειώστε ότι οι όροι της μορφής *s . (t),* που συντέμνονται τώρα σε *s t,* δίνονται μόνο εάν *το s* είναι μια σταθερά. Συνεπώς, δεν δίνονται οι όροι της μορφής *x y.*

Ενδιαφερόμαστε για τις γραμμικές εκφράσεις και εξισώσεις σε συγκεκριμένες μορφές.

Ορισμός 4.26 Υποθέστε μια προορισμένη σειρά < στις μεταβλητές.

- Λέμε ότι μια γραμμική έκφραση είναι σε ***κανονική μορφή*** εάν είναι της μορφής

$$\Sigma_{i=1}^{n} a_i x_i + r, \tag{4.2}$$

όπου $n \geq 0$, κάθε a_i είναι ένας μη- μηδενικός πραγματικός αριθμός, *το r* είναι ένας πραγματικός αριθμός, και οι μεταβλητές $x_1,\dots,x_n$ είναι διατεταγμένα w.r.t. <.

- Λέμε ότι μια γραμμική εξίσωση είναι σε ***κανονική μορφή*** εάν είναι της μορφής

$$\Sigma_{i=1}^{n} a_i x_i = r, \tag{4.3}$$

όπου $n \geq 0$, κάθε a_i είναι ένας μη- μηδενικός πραγματικός αριθμός, *το r* είναι ένας πραγματικός αριθμός, και οι μεταβλητές $x_1,\dots,x_n$ είναι διαταγμένα w.r.t. <.

- Λέμε ότι μια γραμμική εξίσωση

$$x = t$$

είναι σε ***μορφή άξονα*** εάν *το x* $\notin$ *Var (t)* και το *t* είναι σε κανονική μορφή.

Χρησιμοποιώντας τον κατάλληλο κανόνα μετασχηματισμού κάθε γραμμική έκφρασης *s* μπορεί να ξαναγραφεί σε μια μοναδική γραμμική έκφραση (4.2) σε κανονική μορφή. Καλούμε έπειτα την (4.2) ***κανονική μορφή του*** *s* και την δηλώνουμε με *norm(s).* Ομοίως, χρησιμοποιώντας τον κατάλληλο κανόνα μετασχηματισμού κάθε γραμμική εξίσωση *s* = *t* μπορεί να ξαναγραφεί σε μια μοναδική ισοδύναμη γραμμική εξίσωση (4.3) σε κανονική μορφή. (Θα εξηγήσουμε την έννοια των "ισοδύναμων" όρων στην Υποενότητα 4.3.3.) Καλούμε έπειτα την (4.3) ***κανονική μορφή του*** *s* = *t* και λέμε ότι το *s* = *t* ***ομαλοποιείται στο*** (4.3).

Σε όλα τα επόμενα παραδείγματα υποθέτουμε ότι *το x* προηγείται *του y.* Έτσι παραδείγματος χάριν, η γραμμική έκφραση 3x— 1.2 ·(2 +2.5y) — *2x* + 5 που ήδη αναφέραμε μπορεί να ξαναγραφεί στη γραμμική έκφραση *x—3y* + 2.6 σε κανονική μορφή. Επιπλέον, η ήδη εξεταζόμενη γραμμική εξίσωση 4x +

$3.5y = 3x— 1.2 \cdot (2 + 2.5y)— 2x + 5$ ομαλοποιείται στη γραμμική εξίσωση $3x + 6.5y = 2.6$.

Ανάλογα με την τιμή *του n* και του *r* στο (4.3) διακρίνουμε τρεις τύπους κανονικών μορφών:

- $0 = 0$, όταν $n = 0$ και $r = 0$,
- $0 = r$, όταν $n=0$ και r είναι ένας μη- μηδενικός πραγματικός,
- $\Sigma_{i=1,,n} a_i x_i$ όταν $n>0$.

Οι γραμμικές εκφράσεις *s* και *norm(s)* είναι ισοδύναμες υπό την έννοια ότι η κανονική μορφή της γραμμικής εξίσωσης $s = norm(s)$ είναι $0 = 0$.

Με τον τελευταίο τύπο της κανονικής μορφής συνδέουμε τις γραμμικές εξισώσεις *n* σε μορφή άξονα:

$$x_j = \Sigma_{i\in[1..j-1]\cup[j+1..n]} - \frac{a_i}{a_j}x_i + \frac{r}{a_j},$$

όπου $j \in [1...n]$.

Λαμβάνοντας υπόψη μια γραμμική εξίσωση e σε μορφή άξονα, ορίζουμε ότι το e είναι η μόνη μορφή άξονα που έχει. Περαιτέρω, δεδομένης μιας γραμμικής εξίσωσης e όχι σε μορφή άξονα, αλλά με την κανονική μορφή (4.3), όπου $n > 0$, ονομάζουμε κάθε εξίσωση της μορφής (4.4) ***μορφή άξονα*** του *e*. Τέλος, ορίζουμε ότι μια εξίσωση με την κανονική μορφή $0 = r$, όπου *r* είναι ένας πραγματικός αριθμός δεν έχει μορφή άξονα.

Παραδείγματος χάριν, η γραμμική εξίσωση

$$z = 2x + 3y + 4$$

είναι η μόνη μορφή άξονα που έχει. Περαιτέρω, η ήδη εξεταζόμενη γραμμική εξίσωση

$$4x + 3.5y = 3x - 1.2 \cdot (2 + 2.5y) - 2x + 5$$

δεν είναι σε μορφή άξονα και ομαλοποιείται στο $3x + 6.5y = 2.6$, έτσι έχει δύο μορφές άξονα:

$$x = -\tfrac{6.5}{3}y + \tfrac{2.6}{3} \quad \text{και} \quad y = -\tfrac{3}{6.5}x + \tfrac{2.6}{3}$$

Τέλος, η εξίσωση $x + 1 + y = x + y + 1$ δεν είναι σε μορφή άξονα και ομαλοποιείται στο $0 = 0$, συνεπώς δεν έχει καμία μορφή άξονα.

4.3.2 Αντικαταστάσεις, συνενωτές και mgus

Στο υπόλοιπο της έκθεσης πρέπει να ρυθμίσουμε λίγο τις έννοιες που εισάγονται στην Παράγραφο 4.2. Ο ακόλουθος ορισμός συνοψίζει τις διαφορές.

Ορισμός 4.27

- Από μια ***αντικατάσταση*** υπολογίζουμε μια πεπερασμένη σχεδίαση από μεταβλητές σε γραμμικές εκφράσεις σε κανονική μορφή η οποία ορίζει στην περιοχή κάθε μεταβλητής x μια γραμμική έκφραση t διαφορετική από την x.Γράφουμε τις αντικαταστάσεις με τον ίδιο τρόπο όπως στο προηγούμενο τμήμα και ορίζουμε το αποτέλεσμα της εφαρμογής της αντικατάστασης σε έναν όρο (τώρα μια γραμμική έκφραση) ή σε ένα σύνολο γραμμικών εξισώσεων όπως προηγουμένως.

- Αφήστε *το θ και το γ* να είναι αντικαταστάσεις. **Η σύνθεσή** τους, γραμμένη ως *θγ*, είναι η αντικατάσταση που προσδιορίζεται μεμονωμένα από την ακόλουθη σχεδίαση *η* από τις μεταβλητές στις γραμμικές εκφράσεις σε κανονική μορφή

$$\eta(x) := norm((x\theta)\gamma).$$

- θ καλείται ένας συνενωτής ενός συνόλου γραμμικών εξισώσεων E εάν κάθε εξίσωση Eθ ομαλοποιείται στην εξίσωση 0 = 0. Εάν E = {e} , λέμε ότι το θ είναι ένας **συνενωτής** του e.

Παραδείγματος χάριν, εάν $\theta := \{x/2y+1\}$ και $\eta := \{y/2z + 1\}$, έπειτα η σύνθεσή τους $\theta\eta$ είναι ίση με $\{x/4z + 3, y/2z + 1\}$. Επίσης, η αντικατάσταση $\theta := \{x_1/x + 1, x_2/y, x_3/x + y + 1\}$ είναι ένας συνενωτής του $x_1 + x_2 = x_3$ επειδή $(x_1 + x_2)\theta \equiv x + 1 + y$, $x_3\theta \equiv x + y + 1$, και η εξίσωση $x+1+y = x+y+1$ ομαλοποιείται στο 0 = 0.

Επίσης, η ακόλουθη έννοια είναι αντίστοιχη της έννοιας ενός συνόλου εξισώσεων σε λυμένη μορφή που εισάγεται στον Ορισμό 4.13.

Ορισμός 4.28 Δεδομένου ενός συνόλου γραμμικών εξισώσεων E λέμε ότι μια εξίσωση E είναι σε **λυμένη μορφή** εάν είναι στη μορφή άξονα $x = t$ και το x δεν εμφανίζεται αλλού στο E. Εάν κάθε εξίσωση στο E είναι σε λυμένη μορφή, λέμε ότι το E είναι **σε λυμένη μορφή.**

Έτσι ένα πεπερασμένο σύνολο γραμμικών εξισώσεων είναι σε λυμένη μορφή εάν είναι της μορφής $\{x_1 = t_1,\dots,x_n = t_n\}$, όπου τα x_is είναι ευδιάκριτες μεταβλητές, καμία από τις οποίες δεν εμφανίζεται σε επίπεδο tj, και τα t_is είναι γραμμικές εκφράσεις σε κανονική μορφή. Έπειτα έχουμε το ακόλουθο συμπληρωματικό του Λήμματος 4.15 (Λυμένη Μορφή).

Λήμμα 4.29 (Λυμένη μορφή) *Ονομάζουμε ένα mgu θ ενός συνόλου γραμμικών εξισώσεων E **ισχυρό** εάν για κάθε συνενωτή η του E έχουμε η = θη. Εάν το σύνολο των γραμμικών εξισώσεων* $E := \{x_1 = t_1,\dots,x_n = t_n\}$ *είναι σε λυμένη μορφή, έπειτα η αντικατάσταση* $\theta := \{x_1/t_1,\dots,x_n/t_n\}$ *είναι ένα ισχυρό mgu του E.*

Απόδειξη Πρώτα σημειώστε ότι το θ είναι ένας συνενωτής του E. Πράγματι, i E [1,n] έχουμε $x_i\theta \equiv$ ti και επιπλέον $t_i\theta \equiv t_i$, αφού από την υπόθεση κανένα x_j δεν εμφανίζεται στο t_i. Έτσι για i E[1..n] έχουμε $x_i\theta \equiv t_i\theta$. Έτσι η εξίσωση xiθ = tiθ είναι ίδια μετ την ti = ti και συνεπώς ομαλοποιείται στο 0 = 0.

Έπειτα, υποθέστε ότι το η είναι ένας συνενωτής του E. Αυτό σημαίνει ότι για i E[1,n] η εξίσωση $x_i\eta = t_i\eta$ ομαλοποιείται στο 0 = 0. Αλλά ti $\equiv x_i\theta$, έτσι αυτό σημαίνει ότι και η $x_i\eta = x_i\theta\eta$ ομαλοποιείται στο 0 = 0. Περαιτέρω, για x E $\{x_1,\dots x_n\}$ έχουμε $x\eta \equiv x\theta\eta$ επειδή $x \equiv x\theta$. Επιπλέον ισχυρά, η $x\eta \equiv x\theta\eta$

ομαλοποιείται στο 0 = 0.

Έτσι δείξαμε ότι για όλες τις μεταβλητές x η εξίσωση $x\eta \equiv x\theta\eta$ ομαλοποιείται στο 0 = 0. Αλλά οι $x\eta$ και $x\eta\theta$ είναι και οι δύο γραμμικές εκφράσεις σε κανονική μορφή, επομένως $x\eta \equiv x\theta\eta$. Με άλλα λόγια, η $\eta = \theta\eta$, δηλαδή, το θ είναι ένα ισχυρό mgu.

Όπως στο προηγούμενο μέρος λέμε έπειτα ότι το *E* **ορίζει** το θ.

4.3.3 Γραμμικές εξισώσεις και CSPs

Ας συσχετίσουμε τώρα τις γραμμικές εξισώσεις με τους περιορισμούς. Μια απλή επιλογή για τις περιοχές των μεταβλητών είναι το σύνολο όλων των πραγματικών αριθμών. Εδώ ακολουθούμε μια γενικότερη δυνατότητα σύμφωνα με την οποία κάθε περιοχή αποτελείται από το σύνολο των γραμμικών εκφράσεων σε κανονική μορφή, την οποία δηλώνουμε με NF.

Κατόπιν ερμηνεύουμε κάθε γραμμική εξίσωση σαν περιορισμό ως εξής. Συνδέουμε το ακόλουθο υποσύνολο του Καρτεσιανού γινομένου NF με μια γραμμική εξίσωση e με τις μεταβλητές $x_1,\ldots,x_n$:

$$\{(t_1,\ldots t_n) \mid \text{η εξίσωση } e\{x_1/t_1,\ldots,x_n/t_n\} \text{ ομαλοποιείται στο } 0 = 0\ .$$

Παραδείγματος χάριν, η ερμηνεία της εξίσωσης $x_1 + x_2 = x_3$ ως περιορισμό περιέχει ως χαρακτηριστικά στοιχεία τα τριπλάσια (1,1,2), (x, y, x+y), (x, y+1, x+y+1) και (x + 1, y, x + y + 1). Ένας άλλος τρόπος να γραφτεί το παραπάνω σύνολο είναι

$$\{(t_1,\ldots t_n) \mid \{x_1/t_1,\ldots,x_n/t_n\} \text{ είναι ένας συνενωτής του } e\}.$$

Η ερμηνεία ενός συνόλου γραμμικών εξισώσεων ως CSP που προκύπτει μας επιτρέπει να μιλήσουμε για τις λύσεις σε τέτοια σύνολα και για την ισοδυναμία μεταξύ των γραμμικών εξισώσεων. Λέμε ότι δύο γραμμικές εξισώσεις e_1 και e_2 με την ίδια ακολουθία μεταβλητών $x_1,\ldots,x_n$ είναι **ισοδύναμες** εάν η αντιστοιχία των CSPs $\langle e_1 ; x_1 \text{ Ε NF},\ldots,x_n \text{ Ε NF}\rangle$ και $\langle e_2 ; x_1 \text{ Ε NF},\ldots,x_n \text{ Ε NF}\rangle$ είναι ισοδύναμη, δηλαδή εάν έχουν το ίδιο σύνολο λύσεων.

Περαιτέρω, λέμε ότι δύο γραμμικές εξισώσεις e_1 και e_2 με μια ακολουθία κοινών μεταβλητών X είναι **ισοδύναμες w.r.t.** X εάν η αντιστοιχία των CSPs που ορίζεται από αυτές τις δύο εξισώσεις είναι ισοδύναμες w.r.t. X. Κατόπιν σύμφωνα με αυτόν τον ορισμό κάθε γραμμική εξίσωση e και η κανονική μορφή της είναι ισοδύναμες w.r.t. η ακολουθία των μεταβλητών του e.

Το ακόλουθο συμπληρωματικό της Σημείωσης Ενοποίησης 4.17 εξακολουθεί να ισχύει. Όπως στο προηγούμενο μέρος εδώ δηλώνουμε με Sol(P) το σύνολο όλων των λύσεων σε ένα CSP P.

Σημείωση 4.30 (Λύση) *Εξετάστε ένα πεπερασμένο σύνολο γραμμικών εξισώσεων E με τις μεταβλητές $x_1,\ldots,x_n$. Κατόπιν*

$Sol(\langle E \ ;\ x_1 \in \mathcal{NF},\ldots,x_n \in \mathcal{NF}\rangle) = \{(x_1\eta,\ldots,x_n\eta) \mid$ είναι ένας συνενωτής του E}.

Αυτή η παρατήρηση μας επιτρέπει να συσχετίσουμε λύσεις και συνενωτές στα πεπερασμένα σύνολα γραμμικών εξισώσεων.

4.3.4 Το σύστημα απόδειξης *LIN*

Σαν τελικό βήμα προετοιμασιών για να παρουσιάσουμε τους κανόνες απόδειξης εισάγουμε την ακόλουθη σημείωση. Δεδομένης μιας γραμμικής εξίσωσης s = t δηλώνουμε με stand(s = t) την εξίσωση norm(s) = norm(t). Καλούμε έπειτα stand (s = t) την **τυποποιημένη μορφή** της s = t και λέμε ότι η s = t **τυποποιείται στην** norm(s) = norm(t). Επεκτείνουμε τη λειτουργία *stand* στα σύνολα γραμμικών εξισώσεων.

Παραδείγματος χάριν, η τυποποιημένη μορφή της εξίσωσης x—2y + y = 4 είναι η x —y = 4, ενώ η τυποποιημένη μορφή της εξίσωσης x = 2y —y + 4 είναι η x = y + 4. Έτσι μια τυποποιημένη μορφή δεν πρέπει να είναι σε κανονική μορφή.

Στους αλγόριθμους που αναφέρονται παρακάτω μετασχηματίζουμε κατ' επανάληψη τις γραμμικές εξισώσεις σε μια βασική μορφή, εκτελούμε μια κατάλληλη αντικατάσταση σε μερικές άλλες εξισώσεις και εφαρμόζουμε σε αυτές τις εξισώσεις τη λειτουργία *stand*.

Οι τρεις πιθανές μορφές της κανονικής μορφής φέρουν βεβαίως τους τρεις ακόλουθους κανόνες απόδειξης.

ΔΙΑΓΡΑΦΗ

$$\frac{s = v}{\quad}$$

Εάν η s = u ομαλοποιείται στο 0 = 0,

ΑΠΟΤΥΧΙΑ

$$\frac{s = v}{\bot}$$

εάν η s = u ομαλοποιείται στο 0 = r, όπου το r είναι μη- μηδενικός πραγματικός,

ΑΝΤΙΚΑΤΑΣΤΑΣΗ

$$\frac{s = v,\ E}{x = t,\ stand(E\{x/t\})}$$

όπου η x = t είναι μια βασική μορφή της s = u.

Δηλώστε το σύνολο αυτών των τριών κανόνων σύμφωνα με το σύστημα απόδειξης LIN. Ο κανόνας ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μπορεί να εφαρμοστεί εάν η επιλεγμένη γραμμική εξίσωση s = u έχει μια βασική μορφή. Εάν η s = u δεν έχει βασική μορφή, τότε ομαλοποιείται στο 0 =r, όπου το r είναι ένας πραγματικός αριθμός. Σ' αυτή την περίπτωση μπορεί να εφαρμοστεί ο κανόνας ΔΙΑΓΡΑΦΗΣ ή ο κανόνας ΑΠΟΤΥΧΙΑΣ. Συνεπώς, σε κάθε γραμμική εξίσωση μπορεί να εφαρμοστεί ακριβώς ένας κανόνας .

Εάν στον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ η επιλεγμένη εξίσωση s = u είναι σε βασική μορφή, τότε δεν τροποποιείται, δεδομένου ότι η x = t είναι ίδια με την s = u. Διαφορετικά η s = u μετασχηματίζεται σε μια από τις βασικές της μορφές x = t. Σημειώστε ότι δεν διευκρινίζουμε ποια βασική μορφή επιλέγουμε και δεν επιμένουμε ότι το x Ε Var (E). Η αντικατάσταση {x/t} εκτελείται στις εξισώσεις στο Ε και ακολουθείται από την εφαρμογή της λειτουργίας stand. Το ορισμένο Ε μπορεί να είναι κενό.

Χρησιμοποιώντας τους κανόνες LIN μπορούμε να λύσουμε τα πεπερασμένα σύνολα γραμμικών εξισώσεων. Αυτοί οι κανόνες μπορούν να θεωρηθούν ως κανόνες μετασχηματισμού βάση της Παραγράφου 4.1, όπου οι ακολουθίες των εκφράσεων περιοχών παραλείπονται. Ας εξετάσουμε ένα παράδειγμα. Οι επιλεγμένες εξισώσεις υπογραμμίζονται.

Παράδειγμα 4.31 Εξετάστε το ακόλουθο σύνολο γραμμικών εξισώσεων σε κανονική μορφή:

$$\{\underline{x - y = 1}, -x + y = 1, \underline{x = 0}\}.$$

Εφαρμόζουμε σ' αυτό, τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με x = 0 ως η απομονωμένη εξίσωση. Η βασική μορφή της x = 0 είναι η x = 0, έτσι μετά από την εκτέλεση της αντικατάστασης και της τυποποίησης παίρνουμε

$$\{\underline{-y = 1, -x + y = 1}, x = 0\}.$$

Χρησιμοποιώντας πάλι τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ, αυτή τη φορά με - x + y = 1 ως η απομονωμένη εξίσωση και με τη βασική της μορφή y = x + 1, παίρνουμε

$$\{\underline{x = -2}, y = x + 1, \underline{x = 0}\}.$$

Χρησιμοποιώντας ξανά τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με x = 0 ως η απομονωμένη εξίσωση μετά από την εκτέλεση της αντικατάστασης και της τυποποίησης παίρνουμε

$$\{\underline{0 = -2}, y = x + 1, x = 0\}.$$

Εφαρμόζοντας τώρα τον κανόνα ΑΠΟΤΥΧΙΑΣ φθάνουμε στο σύνολο

$$\{\bot, y = x + 1, x = 0\}.$$

Έτσι αυτή η παραγωγή καταλήγει σε αποτυχία. Σημειώστε ότι μετά από δύο εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μετασχηματίσαμε όλες τις εξισώσεις σε μια βασική μορφή.

Δυστυχώς, μια αυθαίρετη χρήση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ προκαλεί περιπλοκές, ακόμα κι αν επιμένουμε ότι το x εμφανίζεται στο σύνολο εξισώσεων E. Δηλαδή, αντίστοιχα στην περίπτωση του συστήματος UNIF μπορούμε να παράγουμε άπειρες παραγωγές. Ένα παράδειγμα δίνεται στην Άσκηση 4.7. Προτού αναφέρουμε τις πιθανές διορθώσεις ας αποδείξουμε ότι όταν δεν υπάρχει η απόκλιση το σύστημα LIN μας επιτρέπει να λύσουμε το αρχικό σύνολο γραμμικών εξισώσεων. Πρώτα χρειαζόμαστε το ακόλουθο λήμμα.

Λήμμα 4.32 (LIN) *Κάθε κανόνας LIN είναι κανόνας διατήρησης ισοδυναμίας w.r.t. την ακολουθία των μεταβλητών που υπάρχουν στην πρόταση κανόνα.*

Απόδειξη Στην περιγραφή της Σημείωσης Λύσης 4.30 αρκεί να αποδείξουμε ότι τα σύνολα εξισώσεων που εξετάζονται στην αρχή και στο τέλος κάθε κανόνα έχουν το ίδιο σύνολο συνενωτών.
Για τον κανόνα ΔΙΑΓΡΑΦΗΣ και τον κανόνα ΑΠΟΤΥΧΙΑΣ η αξίωση είναι μια άμεση συνέπεια του γεγονότος ότι κάθε γραμμική εξίσωση είναι ισοδύναμη με την κανονική της μορφή.
Για την κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ προχωράμε όπως στο Λήμμα 4.20 (UNIF). Εξετάστε δύο σύνολα εξισώσεων {s = u} U E και {x = t} U stand(E{x/t}) όπου η x = t είναι μια βασική μορφή της s = u. Έπειτα

	θ είναι ένας συνενωτής της {s = u} U E
Εάν	{κάθε εξίσωση και η βασική της μορφή έχουν τους ίδιους συνενωτές}
	θ είναι ένας συνενωτής της {x = t} U E
Εάν	θ είναι ένας συνενωτής της x = t και θ είναι ένας συνενωτής του E
Εάν	{Λυμένη Μορφή Λήμμα 4.15 }
	θ είναι ένας συνενωτής της x = t και το {x/t}θ είναι ένας συνενωτής του E
Εάν	θ είναι ένας συνενωτής της x = t και θ είναι ένας συνενωτής της E{x/t}
Εάν	θ είναι ένας συνενωτής της {x = t} U E{x/t}
Εάν	{E{x/t} και stand(E{x/t}) έχουν τους ίδιους συνενωτές}
	θ είναι ένας συνενωτής της {x = t} U stand(E{x/t}).

Τώρα μπορούμε να καθιερώσουμε το επιθυμητό θεώρημα. Η διατύπωση και η απόδειξή του είναι παρόμοιες με αυτές του Θεωρήματος 4.21 (UNIF).

Θεώρημα 4.33 (LIN) *Εξετάστε μια αποτυχημένη ή μια σταθεροποιήσιμη παραγωγή με το σύστημα απόδειξης LIN ξεκινώντας από ένα πεπερασμένο σύνολο γραμμικών εξισώσεων E και τερματίζοντας με ένα σύνολο περιορισμών F. Εάν το E έχει μια λύση, εν συνεχεία το F είναι σε λυμένη μορφή το οποίο ορίζει ένα mgu του E ειδάλλως το F περιέχει τον λανθασμένο περιορισμό ⊥.*

Απόδειξη Εάν το F δεν περιέχει τον λανθασμένο περιορισμό ⊥, το τελικό σύνολο περιορισμών F είναι περιορισμένο κάτω από τις εφαρμογές όλων των κανόνων του LIN. Κατόπιν στην περιγραφή των εξεταζόμενων κανόνων κάθε εξίσωση στο F είναι στη βασική μορφή x = t όπου το x δεν εμφανίζεται αλλού.

Με άλλα λόγια, το F είναι σε λυμένη μορφή. Από τη Σημείωση 4.30(Λύση) και το Λήμμα 4.32 (LIN) το F έχει το ίδιο σύνολο συνενωτών με το αρχικό ορισμένο Ε. Έτσι, στην περιγραφή του Λήμματος 4.29 (Λυμένη Μορφή), το F ορίζει ένα mgu του Ε.

Εάν το τελικό σύνολο περιορισμών F περιέχει τον λανθασμένο περιορισμό $\perp$, τότε η τελευταία επιλεγμένη εξίσωση έχει την κανονική μορφή 0 = r, όπου το r είναι ένας μη- μηδενικός πραγματικός αριθμός και ως εκ τούτου δεν έχει λύση. Σύμφωνα με τη Σημείωση 4.30(Λύση) και το Λήμμα 4.32(LIN) δεν έχει λύση. Αυτό υπονοεί την αξίωση.

Για να εξασφαλιστεί ο τερματισμός υπάρχουν διάφορες δυνατότητες. Η προφανέστερη είναι να περιοριστεί η προσοχή, όπως στο προηγούμενο τμήμα, στις γενικές εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ και να προστεθεί ο όρος x E Var(E). Επίσης υπάρχει μια λιγότερο δραστική εναλλακτική λύση. Ας αναφέρουμε αυτές τις επιλογές στη συνέχεια. Και στους δύο αλγόριθμους υποθέτουμε λόγω απλοποίησης ότι οι αρχικές γραμμικές εξισώσεις είναι όλες σε κανονική μορφή.

4.3.5 **Ο αλγόριθμος** GAUSS-JORDAN ELIMINATION

Ο πρώτος αλγόριθμος αποτελείται από το σύστημα LIN μαζί με έναν χρονοπρογραμματιστή που τους εφαρμόζει κατ΄ επανάληψη έως ότου επιτευχθεί ένα αποτυχημένο CSP ή ένα CSP περιορισμένο κάτω από τις εφαρμογές αυτών των κανόνων. Στην περίπτωση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ εκτελούνται μόνο οι γενικές εφαρμογές και υποτίθεται ότι ο όρος x E Var(E) εξακολουθεί να ισχύει. Αυτός ο αλγόριθμος ονομάζεται GAUSS-JORDAN ELIMINATION, αν και τον παρουσιάζουμε σε μια ελαφρώς διαφορετική μορφή. Το ακόλουθο παράδειγμα επεξηγεί τη λειτουργία του.

Παράδειγμα 4.34 Θεωρήστε το ακόλουθο σύνολο γραμμικών εξισώσεων:

$$\begin{array}{rcrcrcrcrcr}
-x_1 & + & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 & = & 4 \\
x_1 & + & x_2 & - & x_3 & - & 4\,x_4 & + & 3\,x_5 & = & 0 \\
x_1 & - & x_2 & + & x_3 & & & - & 3\,x_5 & = & 2
\end{array}$$

Επιλέγουμε την πρώτη εξίσωση και την ξαναγράφουμε στην ακόλουθη βασική μορφή:

$$x_1 = x_2 + x_3 + 2x_4 - x_5 - 4.$$

Χρησιμοποιώντας τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με τις άλλες δύο εξισώσεις ως Ε μετασχηματίζουμε το παραπάνω σύνολο, μετά την εκτέλεση της αντικατάστασης και της τυποποίησης, στο ακόλουθο σύνολο

$$\begin{array}{rcrcrcrcrcr}
x_1 & = & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 & - & 4 \\
 & & 2x_2 & & & - & 2\,x_4 & + & 2\,x_5 & = & 4 \\
 & & & & 2\,x_3 & + & 2\,x_4 & - & 4\,x_5 & = & 6
\end{array}$$

Η δεύτερη εξίσωση μπορεί να ξαναγραφεί στην ακόλουθη βασική μορφή:

$$x_2 = x_4 - x_5 + 2.$$

Χρησιμοποιώντας τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με τις άλλες δύο εξισώσεις ως Ε μετασχηματίζουμε το τελευταίο σύνολο, μετά την εκτέλεση της αντικατάστασης και της τυποποίησης, σε

$$\begin{array}{lllllllll} x_1 = & x_3 & + & 3\,x_4 & - & 2\,x_5 & - & 2 \\ x_2 = & & & x_4 & - & x_5 & + & 2 \\ & 2\,x_3 & + & 2\,x_4 & - & 4\,x_5 & = & 6 \end{array}$$

Επιλέγοντας τώρα την τελευταία εξίσωση μπορούμε να την ξαναγράψουμε στην ακόλουθη βασική μορφή:

$$x_3 = -x_4 + 2x_5 + 3.$$

Χρησιμοποιώντας τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ για τρίτη φορά, με τις δύο πρώτες εξισώσεις ως Ε, μετασχηματίζουμε το τελευταίο σύνολο σε

$$\begin{array}{lllllll} x_1 = & 2\,x_4 & & & + & 1 \\ x_2 = & x_4 & - & x_5 & + & 2 \\ x_3 = & -\,x_4 & + & 2\,x_5 & + & 3 \end{array}$$

Αυτήν τη στιγμή κανένας κανόνας του LIN δεν ισχύει, έτσι η παραγωγή που έχει εμφανιστεί είναι σταθεροποιήσιμη.

Το ακόλουθο αποτέλεσμα αποδεικνύει την ακρίβεια του αλγορίθμου GAUSS-JORDAN ELIMINATION. Η διατύπωσή και η απόδειξη του είναι παρόμοιες με αυτές του Θεωρήματος 4.25 Αλγόριθμος MARTELLI-MONTANARI.

Θεώρημα 4.35 *(GAUSS-JORDAN ELIMINATION) Ο αλγόριθμος GAUSS-JORDAN ELIMINATION πάντα ολοκληρώνεται. Εάν το αρχικό πεπερασμένο σύνολο γραμμικών εξισώσεων Ε έχει μια λύση, κατόπιν κάθε εκτέλεση του αλγόριθμου ολοκληρώνεται με ένα σύνολο γραμμικών εξισώσεων σε μια λυμένη μορφή που ορίζει ένα mgu του Ε ειδάλλως κάθε εκτέλεση ολοκληρώνεται με ένα σύνολο που περιέχει τον λανθασμένο περιορισμό* $\perp$.

Απόδειξη Δεδομένου ενός συνόλου εξισώσεων Ε, ονομάζουμε μια μεταβλητή x ***λυμένη στο*** Ε εάν για κάποιο όρο t σε κανονική μορφή ισχύει $x = t \in E$ και αυτή είναι η μόνη εμφάνιση του x στο Ε, δηλαδή, εάν η εξίσωση $x = t$ είναι σε λυμένη μορφή. Καλούμε μια μεταβλητή **άλυτη στο** Ε εάν δεν λύνεται.
Για να αποδείξουμε τον τερματισμό συνδέουμε τις ακόλουθες δύο

λειτουργίες με κάθε σύνολο γραμμικών εξισώσεων E:

uns (E) - ο αριθμός των μεταβλητών που είναι άλυτες στο E,

card (E) - ο αριθμός των εξισώσεων στο E.

Τώρα δείχνουμε ότι κάθε εφαρμογή κανόνα σε ένα σύνολο εξισώσεων E μειώνει το φυσικό αριθμό

uns (E) + card (E).

Στην περίπτωση του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ υποθέτουμε ότι εξετάζουμε μια γενική εφαρμογή και ότι x E Var(E).

Για τον κανόνα ΔΙΑΓΡΑΦΗΣ και τον κανόνα ΑΠΟΤΥΧΙΑΣ σημειώστε ότι κάθε εφαρμογή κανόνα μειώνει το card (E) κατά 1 και δεν αυξάνει το uns (E). Διαδοχικά, κάθε γενική εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με x E Var (E) μειώνει το uns (E) κατά 1 και δεν αλλάζει το card (E). Πράγματι, εξαιτίας της γενικότητας κάθε τέτοια εφαρμογή του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ αλλάζει το x από μια άλυτη μεταβλητή σε μία λυμένη.

Αυτό υπονοεί τον τερματισμό. Η άλλη αξίωση προκύπτει από το Θεώρημα 4.33 LIN, αφού κάθε εκτέλεση του αλγόριθμου GAUSS-JORDAN ELIMINATION αντιστοιχεί σε μια αποτυχημένη ή σταθεροποιήσιμη παραγωγή.

Ακριβώς όπως και στην περίπτωση του αλγόριθμου MARTELLI-MONTANARI μπορούμε να υποστηρίξουμε ότι ο αλγόριθμος GAUSS-JORDAN ELIMINATION μπορεί να αντιμετωπισθεί ως πλήρης επιλυτής περιορισμών. Για να το δούμε αυτό ας επιστρέψουμε στη Σημείωση 4.30 (Λύση) που συνόψισε τη σχέση μεταξύ των συνόλων των γραμμικών εξισώσεων και των CSPs. Από κοινού με το Λήμμα 4.29(Λυμένη Μορφή) υπονοεί ότι για ένα σύνολο γραμμικών εξισώσεων E := $\{x_1 = t_1,\dots,x_n = t_n\}$ σε λυμένη μορφή έχουμε

$Sol(\langle E\ ;\ x_1 \in \mathcal{NF},\dots,x_n \in \mathcal{NF}\rangle) =$
{norm(t1n),…,norm(tnn))) I το η είναι μια αντικατάσταση}.

Έτσι για ένα πεπερασμένο σύνολο γραμμικών εξισώσεων σε λυμένη μορφή είναι απλό να παράγουμε όλες τις λύσεις στο αντίστοιχο CSP ⟨E ; x_1 E NF,…,x_n E NF⟩. Τώρα, λαμβάνοντας υπόψη το παραπάνω θεώρημα, ο αλγόριθμος GAUSS-JORDAN ELIMINATION μετασχηματίζει ένα δεδομένο πεπερασμένο σύνολο γραμμικών εξισώσεων σε ένα ισοδύναμο σύνολο εξισώσεων που είναι σε λυμένη μορφή ή περιέχει τον λανθασμένο περιορισμό $\perp$. Έτσι στην ορολογία αυτού του κεφαλαίου ο αλγόριθμος αυτός μπορεί πράγματι να αντιμετωπισθεί ως πλήρης επιλυτής περιορισμών.

4.3.6 Ο αλγόριθμος GAUSSIAN ELIMINATION

Στο δεύτερο αλγόριθμο χρησιμοποιούμε συγκεκριμένες εφαρμογές του κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ που δεν είναι απαραίτητως γενικές. Ο αλγόριθμος έχει δύο φάσεις. Υποθέστε ότι οι εξισώσεις διατάσσονται με έναν αυθαίρετο τρόπο.

Στην πρώτη φάση, αποκαλούμενη **εμπρόσθια φάση αντικατάστασης**, παίρνουμε κατ΄ επανάληψη την πρώτη όχι ακόμα εξεταζόμενη εξίσωση από τα αριστερά. Εάν ισχύει ο κανόνας ΔΙΑΓΡΑΦΗΣ, η εξίσωση διαγράφεται και εξετάζεται η επόμενη εξίσωση. Εάν ισχύει ο κανόνας ΑΠΟΤΥΧΙΑΣ, επιτυγχάνεται μια αποτυχία και η εκτέλεση ολοκληρώνεται. Εάν ισχύει ο κανόνας ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ, τον εφαρμόζουμε λαμβάνοντας ως Ε το σύνολο εξισώσεων που βρίσκεται στα δεξιά της επιλεγμένης εξίσωσης.

Στη δεύτερη φάση, αποκαλούμενη **οπίσθια φάση αντικατάστασης**, παίρνουμε κατ΄ επανάληψη την πρώτη όχι ακόμα εξεταζόμενη εξίσωση από τα δεξιά και εφαρμόζουμε τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ λαμβάνοντας ως Ε το συνόλο εξισώσεων που βρίσκεται στα αριστερά της επιλεγμένης εξίσωσης.

Αυτός ο αλγόριθμος καλείται αλγόριθμος GAUSSIAN ELIMINATION.Σε αντίθεση με τους άλλους εξεταζόμενους αλγόριθμους εδώ, είναι ντετερμινιστικός. Το ακόλουθο παράδειγμα επεξηγεί τη λειτουργία του.

Παράδειγμα 4.36 Θεωρήστε πάλι το σύνολο των γραμμικών εξισώσεων που εξετάζονται στο Παράδειγμα 4.34, δηλ.,

$$\begin{array}{rcrcrcrcrcr} -x_1 & + & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 & = & 4 \\ x_1 & + & x_2 & - & x_3 & - & 4\,x_4 & + & 3\,x_5 & = & 0 \\ x_1 & - & x_2 & + & x_3 & & & - & 3\,x_5 & = & 2 \end{array}$$

Αρχίζουμε με την εμπρόσθια φάση αντικατάστασης. Όπως προηγουμένως επιλέγουμε την πρώτη εξίσωση, την ξαναγράφουμε στην ακόλουθη βασική μορφή:

$$x_1 = x_2 + x_3 + 2x_4 - x_5 - 4$$

και χρησιμοποιούμε τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με τις άλλες δύο εξισώσεις ως Ε. Αυτό παράγει το ακόλουθο σύνολο:

$$\begin{array}{rcrcrcrcrcr} x_1 & = & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 & - & 4 \\ & & 2x_2 & & & - & 2\,x_4 & + & 2\,x_5 & = & 4 \\ & & & & 2\,x_3 & + & 2\,x_4 & - & 4\,x_5 & = & 6 \end{array}$$

Επιλέγοντας τώρα τη δεύτερη εξίσωση μπορούμε να την ξαναγράψουμε στην ακόλουθη βασική μορφή:

$$x_2 = x_4 - x_5 + 2.$$

Τώρα, εντούτοις, σε αντίθεση με το Παράδειγμα 4.34, χρησιμοποιούμε τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ μόνο με την τρίτη εξίσωση ως Ε. Η μεταβλητή x_3 δεν εμφανίζεται σε αυτήν την εξίσωση, επομένως παίρνουμε το ακόλουθο σύνολο εξισώσεων:

$$\begin{array}{rcrcrcrcrcr} x_1 & = & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 & - & 4 \\ & & x_2 & = & & & x_4 & - & x_5 & + & 2 \\ & & & & 2\,x_3 & + & 2\,x_4 & - & 4\,x_5 & = & 6 \end{array}$$

Επιλέγοντας τώρα την τελευταία εξίσωση μπορούμε να την ξαναγράψουμε στην ακόλουθη βασική μορφή:

$$x_3 = -x_4 + 2x_5 + 3.$$

και εφαρμόζουμε τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με το κενό σύνολο εξισώσεων ως Ε. Αυτό παράγει το ακόλουθο σύνολο:

$$\begin{array}{rcrcrcrcl}
x_1 & = & x_2 & + & x_3 & + & 2\,x_4 & - & x_5 - 4 \\
 & & x_2 & = & & & x_4 & - & x_5 + 2 \\
 & & & & x_3 & = & -\,x_4 & + & 2\,x_5 + 3
\end{array}$$

Τώρα είμαστε έτοιμοι με την εμπρόσθια φάση αντικατάστασης του αλγόριθμου και συνεχίζουμε με την οπίσθια φάση αντικατάστασης.

Χρησιμοποιώντας τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με την τελευταία εξίσωση ως επιλεγμένη και τις άλλες δύο εξισώσεις ως Ε μετασχηματίζουμε το παραπάνω σύνολο σε

$$\begin{array}{rcrcrcrcl}
x_1 & = & x_2 & & & + & x_4 & + & x_5 - 1 \\
 & & x_2 & = & & & x_4 & - & x_5 + 2 \\
 & & & & x_3 & = & -\,x_4 & + & 2\,x_5 + 3
\end{array}$$

Χρησιμοποιώντας τώρα τον κανόνα ΑΝΤΙΚΑΤΑΣΤΑΣΗΣ με τη δεύτερη εξίσωση ως επιλεγμένη και την πρώτη ως Ε μετασχηματίζουμε το παραπάνω σύνολο σε

$$\begin{array}{rcrcrcrcl}
x_1 & = & & & & & 2\,x_4 & & + 1 \\
 & & x_2 & = & & & x_4 & - & x_5 + 2 \\
 & & & & x_3 & = & -\,x_4 & + & 2\,x_5 + 3
\end{array}$$

Αυτήν τη στιγμή καμία εφαρμογή ενός κανόνα του LIN δεν είναι σχετική και η παραγωγή ολοκληρώνεται με το τελευταίο σύνολο εξισώσεων. Έτσι φθάσαμε στο ίδιο αποτέλεσμα με εκείνο του Παραδείγματος 4.34.

Γενικά, ο αλγόριθμος GAUSSIAN ELIMINATION είναι αποδοτικότερος από τον αλγόριθμο GAUSS-JORDAN ELIMINATION δεδομένου ότι οδηγεί σε έναν μικρότερο αριθμό αντικαταστάσεων. Το ακόλουθο αποτέλεσμα καθιερώνει την ακρίβεια του αλγόριθμου GAUSSIAN ELIMINATION .

Θεώρημα 4.37 (*GAUSSIAN ELIMINATION*) *Ο αλγόριθμος GAUSSIAN ELIMINATION ολοκληρώνεται πάντα. Εάν η αρχική πεπερασμένη ακολουθία γραμμικών εξισώσεων Ε έχει μια λύση, κατόπιν ο αλγόριθμος ολοκληρώνεται με ένα σύνολο γραμμικών εξισώσεων σε λυμένη μορφή που ορίζει ένα mgu του Ε ειδάλλως ολοκληρώνεται με ένα σύνολο που περιέχει τον λανθασμένο περιορισμό* $\perp$.

Απόδειξη Ο τερματισμός είναι μια άμεση συνέπεια του τρόπου που ορίζεται ο αλγόριθμος. Στην πραγματικότητα, ο αλγόριθμος ολοκληρώνεται μετά από το πολύ 2n—1 εφαρμογές κανόνα, όπου n είναι ο αριθμός των εξισώσεων.

Εντούτοις, σε αντίθεση με τον αλγόριθμο GAUSS-JORDAN ELIMINATION τώρα δεν είναι σαφές ότι κάθε εκτέλεση του αλγόριθμου είναι μια αποτυχημένη ή σταθεροποιήσιμη παραγωγή. Για να το αποδείξουμε αυτό πρέπει να διακρίνουμε δύο περιπτώσεις. Εάν το τελικό σύνολο περιορισμών περιέχει το $\perp$, η αντίστοιχη εκτέλεση είναι σαφώς μια αποτυχημένη παραγωγή.

Διαφορετικά το τελικό σύνολο περιορισμών δεν περιέχει το $\perp$, που σημαίνει ότι ο αλγόριθμος ολοκληρώνεται μετά από μια επιτυχή εκτέλεση της εμπρόσθιας αντικατάστασης και της οπίσθιας φάσης αντικατάστασης. Επάνω στον τερματισμό της εμπρόσθιας φάσης αντικατάστασης φθάνουμε στο σύνολο εξισώσεων

$$\{x_1 = t_1, \ldots, x_n = t_n\},$$

όπου κάθε εξίσωση $x_i = t_i$ είναι σε βασική μορφή και για i Ε [1..n] έχουμε

$$Var(t_i) \cap \{x_1, \ldots, x_i\} = \emptyset. \tag{4.5}$$

Έτσι επάνω στη λήξη της οπίσθιας φάσης αντικατάστασης παίρνουμε το σύνολο εξισώσεων

$$\{x_1 = s_1, \ldots, x_n = s_n\}, \tag{4.6}$$

όπου

$$s_n := t_n$$

και, προχωρώντας προς τα πίσω,

$$s_i := (t_i[x_n/s_n]) \ldots [x_{i+1}/s_{i+1}],$$

όπου i Ε [1..n] και s[x/t] $\equiv$ norm (s{x/t}). Δηλαδή s[x/t] είναι η έκβαση της ομαλοποίησης της γραμμικής έκφρασης s{x/t} . Έτσι το s_i είναι η έκβαση μιας κατάλληλης εναλλαγής της κανονικοποίησης και της αντικατάστασης αρχίζοντας από τη γραμμική έκφραση t_i.

Στην περιγραφή (4.5) έχουμε

$$Var(s_n) \cap \{x_1, \ldots, x_n\} = \emptyset.$$

Προχωρώντας προς τα πίσω, από το n στο 1, υποθέτουμε από την εισαγωγή ότι για j Ε [i + 1..n]

$$Var(s_j) \cap \{x_1, \ldots, x_n\} = \emptyset.$$

Κατόπιν, πάλι στην περιγραφή (4.5), έχουμε

$$Var(t_i\{x_n/s_n\} \ldots \{x_{i+1}/s_{i+1}\}) \cap \{x_1, \ldots, x_n\} = \emptyset,$$

η οποία υπονοεί ότι Var (si) $\cap \{x_1, \ldots, x_n\} = \emptyset$

Εξ' ορισμού κάθε s_i είναι σε κανονική μορφή. Έτσι το σύνολο εξισώσεων (4.6) είναι σε λυμένη μορφή και συνεπώς είναι περιορισμένο κάτω από τις εφαρμογές όλων των κανόνων LIN. Έτσι η εξεταζόμενη εκτέλεση του αλγόριθμου είναι μια σταθεροποιήσιμη παραγωγή.

Η δεύτερη αξίωση τώρα προκύπτει από το θεώρημα 4.33(LIN).

Αυτό ολοκληρώνει τη συζήτηση μας για τους πλήρεις επιλυτές που αφορούν τις γραμμικές εξισώσεις πέρα από τους πραγματικούς αριθμούς. Θα επιστρέψουμε στους κανόνες του συστήματος απόδειξης LIN κατά τη συζήτηση ενός ελλιπή επιλυτή περιορισμών για τις μη γραμμικές εξισώσεις πέρα από τους πραγματικούς.

4.4 Γραμμικές ανισότητες πέρα από τους πραγματικούς

4.4.1 Σύνταξη

Αφού έχουμε εξετάσει τις γραμμικές εξισώσεις πέρα από τους πραγματικούς τώρα αναφέρουμε το πρόβλημα λύσης των πεπερασμένων συνόλων των γραμμικών ανισοτήτων πέρα από τους πραγματικούς. Όπως στο προηγούμενο τμήμα, εξετάζουμε τις γραμμικές εκφράσεις πέρα από τους πραγματικούς και θεωρούμε τους ίδιους συντακτικούς κανόνες. Από μια **γραμμική ανισότητα πέρα από τους πραγματικούς** υπολογίζουμε έναν περιορισμό της μορφής

$$s \leq t,$$

όπου τα s και t είναι γραμμικές εκφράσεις. Όπως προηγουμένως παραλείπουμε τον προσδιορισμό "πέρα από τους πραγματικούς". Παραδείγματος χάριν η,

$$4x - 3.5y - 1.2z \leq 3x - 1.2 \cdot (2 + 2.5y + z) - 2x - 5$$

είναι μια γραμμική ανισότητα.

Κατά την επίλυση των πεπερασμένων συνόλων γραμμικών ανισοτήτων θα αποκλείσουμε κατ΄ επανάληψη τις μεταβλητές που υπάρχουν σε αυτά. Αυτό θα γίνει μετασχηματίζοντας πρώτα κάθε γραμμική ανισότητα σε μια κανονική μορφή στην οποία θα απομονωθεί μια επιλεγμένη μεταβλητή. Αυτό μας φέρνει στις κανονικές μορφές για τις οποίες θα ενδιαφερθούμε.

Ορισμός 4.38 Υποθέστε μια προορισμένη σειρά < στις μεταβλητές. Ορίστε μία μεταβλητή x.

Λέμε ότι μια γραμμική ανισότητα είναι σε **x-κανονική μορφή** εάν είναι σε μια από τις ακόλουθες μορφές, όπου το X $\notin$ Var(t) και το t είναι μια γραμμική έκφραση σε κανονική μορφή:

- η **≤x-κανονική μορφή**: t≤x,
- η **$\underline{x}^{\leq}$-κανονική μορφή**: x≤ t,
- η **x-κανονική μορφή**: t≤0.

Έτσι η μεταβλητή x δεν εμφανίζεται στη $\bar{x}$-κανονική μορφή. Χρησιμοποιώντας τους κατάλληλους κανόνες μετασχηματισμού, δεδομένης μίας μεταβλητής x, κάθε γραμμική ανισότητα s≤t μπορεί να ξαναγραφεί σε μια μοναδική ισοδύναμη γραμμική ανισότητα σε x-κανονική μορφή. Λέμε ότι η s≤t **κανονικοποιείται στην** κανονική της μορφή. Εδώ έχουμε ένα παράδειγμα. Υποθέτουμε εδώ και οπουδήποτε αλλού μια σειρά < έτσι ώστε x < y < z.

Παράδειγμα 4.39 Επανεξετάστε τη γραμμική ανισότητα

$$4x - 3.5y - 1.2z \leq 3x - 1.2 \cdot (2 + 2.5y + z) - 2x + 5.$$

Κανονικοποιείται στην $x^{\leq}$-κανονική μορφή x≤ $\frac{0.5}{3}y + \frac{2.6}{3}$ και στην ≤y-κανονική μορφή 6x— 5.2≤y. Αντίθετα, η z-κανονική της μορφή είναι 6x —y— 5.2≤0, η οποία είναι μια z-κανονική μορφή, χωρίς μία εμφάνιση του z.

4.4.2 Γραμμικές ανισότητες και CSPs

Ερμηνεύουμε τις γραμμικές ανισότητες ως περιορισμούς αναλογικά όπως με τις γραμμικές εξισώσεις. Έτσι, όπως στο προηγούμενο τμήμα, υποθέτουμε ότι κάθε περιοχή μιας μεταβλητής αποτελείται από το σύνολο των γραμμικών εκφράσεων σε κανονική μορφή, την οποία δηλώσαμε με NF. Έπειτα συνδέουμε το ακόλουθο υποσύνολο του Καρτεσιανού γινομένου NF^n με ένα γραμμικό ανισότητας li με τις μεταβλητές $x_1,\ldots,x_n$:

{($t_1,\ldots,t_n$) | η ανισότητα li {$x_1/t_1,\ldots,x_n/t_n$} > κανονικοποιείται στο r≤0,

όπου το r είναι ένας αυστηρά αρνητικός πραγματικός αριθμός ή 0 .

Παραδείγματος χάριν, η ερμηνεία της ανισότητας $x_1 + x_2 \leq x_3$ ως περιορισμό περιέχει ως χαρακτηριστικά στοιχεία τα τριπλάσια (1, 1, 3), (x + 1, 1, x + 3), (x, y, x + y), (x, y, x + y +2), (x, y + 1, x + y + 2) και (x + 1, y, x + y +1). Παραδείγματος χάριν, η x + y ≤ x + y + 2 κανονικοποιείται στο — 2 ≤ 0 και ομοίως για τα άλλα τριπλάσια.

Η ερμηνεία ενός συνόλου γραμμικών ανισοτήτων ως CSP που προκύπτει μας επιτρέπει να μιλήσουμε για τις λύσεις σε τέτοια σύνολα και για την ισοδυναμία μεταξύ των γραμμικών ανισοτήτων. Λέμε ότι δύο γραμμικά ανισοτήτων l_{i1} και l_{i2} με την ίδια ακολουθία μεταβλητών $x_1,\ldots,x_n$ είναι **ισοδύναμα** εάν τα CSPs $\langle l_{i1}$; x_1 E NF,...,x_n E NF$\rangle$ και $\langle l_{i2}$; x_1 E NF,...,x_n E NF$\rangle$ είναι ισοδύναμα.

Περαιτέρω, λέμε ότι δύο γραμμικά ανισοτήτων l_{i1} και l_{i2} με μια ακολουθία κοινών μεταβλητών X είναι **ισοδύναμα w.r.t.** X εάν τα αντίστοιχα CSPs που ορίζονται από αυτές τις δύο ανισότητες είναι ισοδύναμα w.r.t. X. Κατόπιν σύμφωνα με αυτόν τον ορισμό κάθε γραμμικό ανισότητας l_i και η κανονική του μορφή είναι ισοδύναμα w.r.t. η ακολουθία των μεταβλητών του li. Ειδικότερα, η ανισότητα που ήδη αναφέραμε

$$4x - 3.5y - 1.2z \leq 3x - 1.2 \cdot (2 + 2.5y + z) - 2x + 5$$

και κάθε μια από τις κανονικές της μορφές, επομένως οι x ≤ $\frac{0.5}{3}y + \frac{2.6}{3}$ και 6x — y— 5.2 ≤ 0 είναι ισοδύναμες w.r.t. x, y.

4.4.3 Το σύστημα απόδειξης *INEQ*

Λόγω των πεπερασμένων συνόλων των γραμμικών ανισοτήτων εισάγουμε τώρα τρεις κανόνες απόδειξης. Για να τους παρουσιάσουμε χρειαζόμαστε την ακόλουθη σημείωση.

Ορισμός 4.40 Δεδομένης μιας μεταβλητής x και ενός συνόλου γραμμικών ανισοτήτων li εισάγουμε τα ακόλουθα τρία σύνολα γραμμικών ανισοτήτων που προέρχονται από το:

- ≤x(LI) := { li I li είναι η ≤x-κανονική μορφή μιας ανισότητας από το LI},
- x≤(LI) := { li I li είναι η x≤-κανονική μορφή μιας ανισότητας από το LI} ,
- $\bar{x}$ (LI) := { li I li είναι η $\bar{x}$-κανονική μορφή μιας ανισότητας από το LI} .

Εισάγουμε επίσης την ακόλουθη πράξη στα σύνολα των γραμμικών ανισοτήτων:

$$E \cdot F := \{s \leq v \mid s \leq t \in E,\ t \leq v \in F\}.$$

Έτσι εάν το E ή το F είναι κενό, τότε έτσι είναι το E . F.

Η ιδέα που κρύβεται πίσω από τον αλγόριθμο την οποία πρόκειται και να αναφέρουμε είναι πολύ απλή. Επιλέγουμε κατ΄ επανάληψη μια μεταβλητή, για παράδειγμα την x, και εκτελούμε τα ακόλουθα δύο βήματα:

- κανονικοποιούμε όλες τις ανισότητες του ισχύοντος συνόλου LI σε x-κανονική μορφή,
- διαγράφουμε όλες τις εμφανίσεις της x αντικαθιστώντας τα σύνολα ≤ x (LI) και x ≤ (LI) από "τη σύνθεσή τους" ≤ x (LI). x ≤ (LI).

  Σημειώστε ότι η περίπτωση κατά την οποία ένα από αυτά τα σύνολα είναι κενά, ισοδυναμεί με διαγραφή όλων των ανισοτήτων στο άλλο σύνολο.

Αυτή η διαδικασία μπορεί να αναπαρασταθεί χρησιμοποιώντας τον ακόλουθο κανόνα απόδειξης:

Χ-ΔΙΑΓΡΑΦΗ

$$\frac{LI}{{}^{\leq}x(LI) \cdot x^{\leq}(LI),\ \bar{x}(LI)}$$

Επιπλέον, εισάγουμε τους ακόλουθους δύο κανόνες που εξετάζουν τις συγκεκριμένες - x -κανονικές μορφές:

ΔΙΑΓΡΑΦΗ

$$\frac{s \leq t}{}$$

εάν s ≤ t κανονικοποιείται στο r ≤ 0, όπου το r είναι ένας αυστηρά αρνητικός πραγματικός αριθμός ή 0,

ΑΠΟΤΥΧΙΑ

$$\frac{s \leq t}{\perp}$$

εάν s ≤ t κανονικοποιείται στο r ≤ 0, όπου το r είναι αυστηρά θετικός πραγματικός.

Δείξτε το σύνολο αυτών των τριών κανόνων με το σύστημα απόδειξης INEQ.

4.4.4 **Ο αλγόριθμος** FOURIER-MOTZKIN ELIMINATION

Μπορούμε τώρα να εισάγουμε τον επιθυμητό αλγόριθμο. Αποτελείται από τους τρεις παραπάνω κανόνες μαζί με έναν χρονοπρογραμματιστή που τους εφαρμόζει κατ΄ επανάληψη μέχρι να επιτευχθεί ένα αποτυχημένο CSP ή ένα CSP περιορισμένο κάτω από τις εφαρμογές αυτών των κανόνων. Στην περίπτωση του κανόνα x-Elimination κάθε μια από τις εφαρμογές του είναι γενική και αναφέρεται σε διαφορετική μεταβλητή. Αυτός ο αλγόριθμος καλείται αλγόριθμος FOURIER-MOTZKIN ELIMINATION. Το ακόλουθο παράδειγμα επεξηγεί τη λειτουργία του.

Παράδειγμα 4.41

(i) Εξετάστε το ακόλουθο σύνολο γραμμικών ανισοτήτων:

$$\begin{array}{rcl} 0 & \leq & x \\ -x-y & \leq & 2 \\ -x+y & \leq & 3 \\ x+2y & \leq & 6 \\ 0 & \leq & y \\ -x-y+2 & \leq & z \end{array} \tag{4.7}$$

Ο μετασχηματισμός κάθε ενός από αυτούς στη x-κανονική μορφή παράγει το ακόλουθο σύνολο:

$$
\begin{aligned}
0 &\leq x \\
-y-2 &\leq x \\
y-3 &\leq x \\
x &\leq -2y+6 \\
-y &\leq 0 \\
-y-z+2 &\leq x
\end{aligned}
$$

Ως εκ τούτου χρησιμοποιώντας τον κανόνα x-ΔΙΑΓΡΑΦΗΣ λαμβάνουμε το ακόλουθο σύνολο ανισοτήτων στο οποίο το x δεν εμφανίζεται:

$$
\begin{aligned}
0 &\leq 0 \\
0 &\leq -2y+6 \\
-y-2 &\leq -2y+6 \\
y-3 &\leq -2y+6 \\
-y &\leq 0 \\
-y-z+2 &\leq -2y+6
\end{aligned}
$$

Η πρώτη ανισότητα μπορεί να διαγραφεί χρησιμοποιώντας τον κανόνα ΔΙΑΓΡΑΦΗΣ. Μετασχηματίζοντας κάθε μιας από τις υπόλοιπες πέντε ανισότητες στην y-κανονική μορφή λαμβάνουμε τώρα το ακόλουθο σύνολο:

$$
\begin{aligned}
y &\leq 3 \\
y &\leq 8 \\
y &\leq 3 \\
0 &\leq y \\
y &\leq z+4
\end{aligned}
\tag{4.8}
$$

Διαγράφοντας τώρα το y που χρησιμοποιεί τον κανόνα y-ΔΙΑΓΡΑΦΗΣ λαμβάνουμε το ακόλουθο σύνολο τεσσάρων ανισοτήτων:

$$
\begin{aligned}
0 &\leq 3 \\
0 &\leq 8 \\
0 &\leq 9 \\
0 &\leq z+4
\end{aligned}
$$

Μπορούμε τώρα να διαγράψουμε τις τρεις πρώτες ανισότητες χρησιμοποιώντας τον κανόνα ΔΙΑΓΡΑΦΗΣ και καταλήγουμε σε μια ενιαία ανισότητα της οποίας η z-κανονική μορφή είναι:

$$-4 \leq z.$$

Αυτήν τη στιγμή εφαρμόζουμε τον κανόνα z-ΔΙΑΓΡΑΦΗΣ. Καμία ανισότητα δεν είναι στην z≤-κανονική μορφή ή στη z-κανονική μορφή, επομένως αυτή η εφαρμογή κανόνα ισοδυναμεί με την διαγραφή της παραπάνω ανισότητας και καταλήγουμε στο κενό σύνολο. Αυτό, όπως θα δούμε, υπονοεί ότι το αρχικό σύνολο ανισοτήτων (4.7) είναι σταθερό.

(ii) Εξετάστε τώρα το ακόλουθο σύνολο γραμμικών ανισοτήτων:

$$\begin{aligned}
x + z &\leq x + z + 1 \\
y + 3z + 6 &\leq x + y \\
-y + 2z + 6 &\leq x - y \\
x + y &\leq -2y + 2 \\
x + z &\leq 2y + z + 3 \\
x + 2y &\leq x + z + 1 \\
x + y &\leq x + z + 1
\end{aligned}$$

Η πρώτη ανισότητα κανονικοποιείται στο —1 ≤ 0, έτσι χρησιμοποιώντας τον κανόνα ΔΙΑΓΡΑΦΗΣ μπορούμε να την διαγράψουμε. Ο μετασχηματισμός κάθε μιας από τις υπόλοιπες έξι ανισότητες στη x-κανονική μορφή παράγει το ακόλουθο σύνολο:

$$\begin{aligned}
3z + 6 &\leq x \\
2z + 6 &\leq x \\
x &\leq -3y + 2 \\
x &\leq 2y + 3 \\
2y - z - 1 &\leq 0 \\
y - z - 1 &\leq 0
\end{aligned}$$

Επομένως οι δύο πρώτες ανισότητες είναι στη ≤x-κανονική μορφή, οι επόμενες δύο στη x≤- κανονική μορφή και οι δύο τελευταίες στη x-κανονική μορφή. Ως εκ τούτου χρησιμοποιώντας τον κανόνα x-ΔΙΑΓΡΑΦΗΣ λαμβάνουμε το ακόλουθο σύνολο ανισοτήτων στο οποίο δεν εμφανίζεται το x:

$$\begin{aligned}
3z + 6 &\leq -3y + 2 \\
3z + 6 &\leq 2y + 3 \\
2z + 6 &\leq -3y + 2 \\
2z + 6 &\leq 2y + 3 \\
2y - z - 1 &\leq 0 \\
y - z - 1 &\leq 0
\end{aligned}$$

Μετασχηματίζοντας κάθε μιας από αυτές τις έξι ανισότητες στην y-κανονική μορφή λαμβάνουμε το ακόλουθο σύνολο:

$$\begin{aligned} y &\leq -z - \tfrac{4}{3} \\ \tfrac{3}{2}z + \tfrac{3}{2} &\leq y \\ y &\leq -\tfrac{2}{3}z - \tfrac{4}{3} \\ z + \tfrac{3}{2} &\leq y \\ y &\leq \tfrac{1}{2}z + \tfrac{1}{2} \\ y &\leq z + 1 \end{aligned}$$

Έτσι χρησιμοποιώντας τον κανόνα y-ΔΙΑΓΡΑΦΗΣ λαμβάνουμε ένα σύνολο οκτώ ανισοτήτων. Μία από αυτές, ως αποτέλεσμα των ανισοτήτων $z + 3/2 \leq y$ και $y \leq z + 1$ είναι η $z + 3/2 \leq z + 1$ που κανονικοποιείται στο $½ \leq 0$. Έτσι εφαρμόζοντας τον κανόνα ΑΠΟΤΥΧΙΑΣ εισάγουμε τον λανθασμένο περιορισμό $\perp$ που παράγει μια αποτυχημένη παραγωγή. Αυτό σημαίνει, όπως θα δούμε στο τέλος αυτού του μέρους, ότι το αρχικό σύνολο ανισοτήτων είναι σταθερό.

Στο υπόλοιπο αυτού του μέρους διευκρινίζουμε τη ιδιότητα του αλγόριθμου FOURIER-MOTZKIN ELIMINATION. Για αυτόν τον σκοπό ας επιστρέψουμε στον κανόνα x-ELIMINATION. Επειδή το συμπέρασμα αυτού του κανόνα δεν αναφέρεται στο x, αυτός ο κανόνας δεν διατηρεί την ισοδυναμία. Ο λόγος είναι ότι το συμπέρασμα δεν βάζει καθόλου όρους στην μεταβλητή x, ενώ η το εισαγωγικό μέρος το κάνει. Θεωρήστε για παράδειγμα τις εξισώσεις $y \leq x$ και $x \leq z$. Στη συνέχεια μετά την εφαρμογή του κανόνα x-ELIMINATION λαμβάνουμε την ανισότητα $y \leq z$ που ικανοποιείται από οποιοδήποτε τριάδα x, y, z έτσι ώστε η $y \leq z$ ισχύει, μ' αυτόν τον τρόπο με μία αυθαίρετη x. Εντούτοις, αυτός ο κανόνας ωφελείται από μία πιό 'ανίσχυρη' ιδιότητα.

Λήμμα 4.42 (INEQ)

(i) Οι κανόνες ΔΙΑΓΡΑΦΗΣ και ΑΠΟΤΥΧΙΑΣ διατηρούν την ισοδυναμία.

(ii) Κάθε γενική εφαρμογή του κανόνα x-ΔΙΑΦΡΑΦΗΣ είναι ισοδυναμία που διατηρεί w.r.t. την ακολουθία των μεταβλητών που υπάρχουν στο συμπέρασμα του κανόνα. Συνεπώς, αυτός ο κανόνας διατηρεί την ισοδυναμία.

Απόδειξη (i) Η αξίωση είναι άμεση.

(ii) Πρώτα, σημειώνουμε ότι ο κανόνας

$$\frac{LI}{{}^{\leq}x(LI),\ x^{\leq}(LI),\ \bar{x}(LI)}$$

είναι κανόνας διατηρήσιμης ισοδυναμίας αφού κάθε γραμμικό ανισότητας l_i είναι ισοδύναμο με τη κανονική του μορφή w.r.t. στην ακολουθία των μεταβλητών του I_i. Έτσι αρκεί να αποδείξουμε ότι ο κανόνας

$$\frac{{}^{\le}x(LI),\ x^{\le}(LI),\ \bar{x}(LI)}{{}^{\le}x(LI)\cdot x^{\le}(LI),\ \bar{x}(LI)}$$

είναι κανόνας διατηρήσιμης ισοδυναμίας w.r.t. την ακολουθία των μεταβλητών που υπάρχουν στο συμπέρασμα του κανόνα, το οποίο είναι w.r.t. στην ακολουθία των μεταβλητών μεταξύ

$$Var({}^{\le}x(LI),\ x^{\le}(LI),\ \bar{x}(LI)) - \{x\}.$$

Έτσι εξετάστε μια λύση d στην αρχή αυτού του κανόνα. Ικανοποιεί κάθε ανισότητα s ≤ x στο ≤x(LI) και κάθε ανισότητα x ≤ t στο x≤(LI), επομένως με τη μεταβατικότητα του ≤ ικανοποιεί κάθε ανισότητα s≤t στο ≤ x(LI) . x≤ (≤LI). Ως εκ τούτου το d είναι μια λύση στο συμπέρασμα αυτού του κανόνα.

Αντιθέτως, εξετάστε μια λύση d στο συμπέρασμα αυτού του κανόνα. Πρέπει να την επεκτείνουμε, αναθέτοντας μια τιμή στο x, σε μια λύση στην αρχή του κανόνα. Μπορούμε να υποθέσουμε ότι το ≤x(LI) ή το x≤(LI) είναι μη- άδειο , δεδομένου ότι διαφορετικά συμπίπτουν η αρχή και το συμπέρασμα του παραπάνω κανόνα. Προκύπτουν τρεις περιπτώσεις.

Περίπτωση 1. Το ≤x(LI) είναι μη- κενό και το x≤ (LI) είναι κενό.

Κατόπιν ο παραπάνω κανόνας ανάγεται σε

$$\frac{{}^{\le}x(LI),\ \bar{x}(LI)}{\bar{x}(LI)}$$

Επεκτείνετε την d με έναν αυθαίρετο τρόπο στις μεταβλητές στο Var((≤x(LI)) — {x} . Αφήστε

s_o := max (t \t≤x E ≤ x(LI)).

Τώρα ορίστε στο x την τιμή του s_o που ορίζεται από την παραπάνω προέκταση της λύσης d. Αυτή η τελική προέκταση της d ικανοποιεί όλες τις ανισότητες στο ≤x(LI).

Περίπτωση 2. Το ≤x(LI) είναι κενό και το x≤(LI) είναι μη- κενό.

Κατόπιν ο παραπάνω κανόνας ανάγεται σε

$$\frac{x^{\le}(LI),\ \bar{x}(LI)}{\bar{x}(LI)}$$

Επεκτείνετε την d με έναν αυθαίρετο τρόπο στις μεταβλητές στο Var(x≤((LI))— {x}.Αφήστε:

$$t_0 := min(s \mid x \le s \in x^{\le}(LI)).$$

Τώρα ορίστε στο x την τιμή του t_o ορισμένη από την παραπάνω προέκταση της λύσης d. Αυτή η τελική επέκταση της d ικανοποιεί όλες τις ανισότητες στο $x^{\leq}$ (LI).

Περίπτωση 3. Το $^{\leq}x(LI)$ και το $x^{\leq}(LI)$ είναι μη- κενά.

Η d ορίζει μια τιμή σε όλες τις μεταβλητές στο $^{\leq}x(LI)$. $x^{\leq}(LI)$, επομένως ορίζει μοναδικές τιμές στους όρους s_0 από την Περίπτωση 2 και t_0 από την Περίπτωση 3. Αυτές οι τιμές αντιστοιχούν στις γραμμικές εκφράσεις s_1 και t_1 σε κανονική μορφή έτσι ώστε $s_1 \leq x$ Ε $^{\leq}x(LI)$ και $x \leq t_1$ Ε $x^{\leq}(LI)$. Έτσι $s_1 \leq t_1$ Ε $^{\leq}x(LI)$. $x^{\leq}(LI)$. Ως εκ τούτου η d είναι μια λύση στη $s_1 \leq t_1$.

Όπως στην Περίπτωση 2 επεκτείνετε την d θέτοντας στο x την τιμή του s_0, δηλ., την τιμή του s_1. Όπως στην Περίπτωση 2 η επέκταση e της d που προκύπτει ικανοποιεί όλες τις ανισότητες στο $^{\leq}x(LI)$. Δεδομένου ότι το e έπειτα ικανοποιεί την ανισότητα $x \leq t_1$, ικανοποιεί επίσης όλες τις ανισότητες στο $x^{\leq}(LI)$. (Εναλλακτικά, μπορούμε να επεκτείνουμε την d θέτοντας στο x την τιμή του t_0.)

Αυτή η απόδειξη δείχνει επίσης ότι ο κανόνας Χ-ΔΙΑΓΡΑΦΗΣ είναι κανόνας διατηρήσιμης σταθερότητας . Πράγματι, δείξαμε ότι εάν η πρόταση(αρχή) κανόνα έχει μια λύση, σ' αυτή την περίπτωση αποτελεί επίσης μια λύση για το συμπέρασμα κανόνα, και αντιθέτως, εάν το συμπέρασμα κανόνα έχει μια λύση, τότε μπορεί να επεκταθεί σε μια λύση της πρότασης κανόνα.

Αυτό μας οδηγεί στο ακόλουθο συμπέρασμα.

Θεώρημα 4.43 (INEQ) *Ο αλγόριθμος FOURIER-MOTZKIN ELIMINATION πάντα ολοκληρώνεται. Εάν το αρχικό πεπερασμένο σύνολο γραμμικών ανισοτήτων είναι σταθερό, κατόπιν κάθε εκτέλεση του αλγόριθμου ολοκληρώνεται με το κενό σύνολο περιορισμών ειδάλλως κάθε εκτέλεση ολοκληρώνεται με ένα σύνολο που περιέχει τον λανθασμένο περιορισμό* $\perp$.

Απόδειξη Ο τερματισμός είναι μια άμεση συνέπεια του γεγονότος ότι υπάρχουν τελικά πολλές μεταβλητές στο αρχικό ορισμένο σύνολο LI των ανισοτήτων.

Από την φόρμα των κανόνων του συστήματος INEQ το τελικό σύνολο περιορισμών F είναι κενό ή περιέχει τον λανθασμένο περιορισμό $\perp$. Από το Λήμμα 4.42 INEQ τα CSPs που διαμορφώνουν την εξεταζόμενη παραγωγή είναι όλα σταθερά ή όλα μεταβλητά. Έτσι εάν το αρχικό σύνολο γραμμικών ανισοτήτων είναι σταθερό, κατόπιν το τελικό σύνολο περιορισμών είναι επίσης σταθερό, επομένως είναι κενό. Και εάν το αρχικό σύνολο γραμμικών ανισοτήτων είναι μεταβλητό, κατόπιν το τελικό σύνολο περιορισμών είναι επίσης μεταβλητό, συνεπώς περιέχει τον λανθασμένο περιορισμό $\perp$.

Κατά συνέπεια ο αλγόριθμος FOURIER-MOTZKIN ELIMINATION μας επιτρέπει να ορίσουμε τη σταθερότητα ενός πεπερασμένου συνόλου γραμμικών ανισοτήτων. Έτσι, σύμφωνα με την ορολογία μας, αυτός ο αλγόριθμος δεν είναι ένας πλήρης επιλυτής περιορισμών. Εντούτοις, κάποιος μπορεί να τροποποιήσει αυτόν τον αλγόριθμο σε έναν που μας επιτρέπει να μειώσουμε το αρχικό σύνολο ανισοτήτων σε ένα ισοδύναμο σύνολο σε μια κατάλληλα ορισμένη λυμένη μορφή. Για αυτόν τον σκοπό χρησιμοποιούμε τους όρους s_0 και t_0 που ορίζονται στην απόδειξη του Λήμματος 4.42 INEQ και προσθέτουμε στο συμπέρασμα του κανόνα x- ELIMINATION καμία, μια ή και τις δύο ανισότητες $s_0 \leq x$, $x \leq t_0$ ανάλογα με την αναλυτική περίπτωση που παρέχεται στην απόδειξη του Λήμματος 4.42 INEQ. Σημειώστε ότι οι όροι s_0 και t_0 ορίζονται σε μια προέκταση της εξεταζόμενης γλώσσας που λαμβάνεται με την προσθήκη των λειτουργιών min και max. Κατόπιν ο αλγόριθμος που προκύπτει ολοκληρώνεται με το σύνολο τέτοιων προστιθέμενων ανισοτήτων, τις οποίες βλέπουμε ως λυμένη μορφή, ή με ένα σύνολο που περιέχει τον λανθασμένο περιορισμό $\perp$. Από αυτήν την λυμένη μορφή μπορούμε να παράγουμε όλες τις λύσεις στο αρχικό σύνολο ανισοτήτων.

Ο αλγόριθμος FOURIER-MOTZKIN ELIMINATION μας επιτρέπει επίσης να βρούμε τις βέλτιστες λύσεις σε ένα πεπερασμένο σύνολο γραμμικών ανισοτήτων οι οποίες υπάγονται σε μια γραμμική αντικειμενική λειτουργία. Για να δούμε πώς θεωρήστε ως παράδειγμα το πρόβλημα εύρεσης μιας λύσης στο ακόλουθο σύνολο ανισοτήτων:

$$\begin{array}{rcl} 0 & \leq & x \\ -x-y & \leq & 2 \\ -x+y & \leq & 3 \\ x+2y & \leq & 6 \\ 0 & \leq & y \end{array}$$

για το οποίο η τιμή της αντικειμενικής λειτουργίας—2x —y + 2 είναι ελάχιστη.

Για να το λύσουμε εισάγουμε μια νέα μεταβλητή, z, και προσθέτουμε την ακόλουθη ανισότητα στις παραπάνω:

$$-2x-y+2 \quad \leq \quad z.$$

Αυτό γίνεται εν συνεχεία το σύνολο ανισοτήτων που εξετάζεται στο Παράδειγμα 4.41 (i). Εκεί μειώσαμε αυτό το σύνολο ανισοτήτων στην ενιαία ανισότητα —4 ≤ z. Έτσι η ελάχιστη πιθανή τιμή για το z είναι το —4. Προχωρώντας προς τα πίσω μπορούμε τώρα να βρούμε τις τιμές για τα x και y για τα οποία z = —4. Δηλαδή, αντικαθιστάμε το z από το —4 στις εξισώσεις (4.8), και το ίδιο στο y ≤ z + 4. Αυτό, μαζί με την ανισότητα 0 ≤ y του (4.8), παράγει y = 0. Τώρα αντικαθιστώντας το y από το 0 και το z από το —4 στο αρχικό σύνολο ανισοτήτων (4.7) παράγει συγκεκριμένα —x ≤ —6 και x ≤ 6 από τα οποία εμείς καταλήξαμε ότι x = 6.

Δυστυχώς, ο αλγόριθμος FOURIER-MOTZKIN ELIMINATION δεν είναι αποδοτικός. Για να το δούμε αυτό ρίχνουμε μια καλύτερη ματιά στον κανόνα x-ΔΙΑΓΡΑΦΗΣ. Κάθε γενική εφαρμογή αυτού του κανόνα διαγράφει από το τρέχον σύνολο ανισοτήτων όλες τις ανισότητες που περιέχουν το x. Εντούτοις, αυτό συμβαίνει με κόστος την εισαγωγή ενός ενδεχομένως μεγάλου συνόλου ανισοτήτων που δεν περιέχουν το x. Πράγματι, ο αριθμός στοιχείων συνόλου του συνόλου $\leq$x(LI) . x$\leq$(LI) είναι το γινόμενο των αριθμών στοιχείων συνόλων των συνόλων $\leq$x(LI) και x$\leq$(LI). Στη χειρότερη περίπτωση η συσσωρευμένη αύξηση του αριθμού ανισοτήτων μπορεί να οδηγήσει σε έναν διπλά εκθετικό αριθμό ανισοτήτων.

Μια πρακτικότερη προσέγγιση παρέχεται από το αλγόριθμο SIMPLEX. Αυτός ο αλγόριθμος μας επιτρέπει να υπολογίσουμε τις βέλτιστες λύσεις σε ένα πεπερασμένο σύνολο γραμμικών ανισοτήτων οι οποίες υπάγονται σε μία γραμμική αντικειμενική λειτουργία μετασχηματίζοντας κατ' επανάληψη μια λύση στο εξεταζόμενο σύνολο γραμμικών ανισοτήτων σε μια με μια καλύτερη τιμή αντικειμενικής λειτουργίας. Σε ορισμένα σύνολα προβλήματος αυτός ο αλγόριθμος τρέχει σε εκθετικό χρόνο. Εντούτοις, στην πράξη είναι εντυπωσιακά αποδοτικός. Συνεπώς, χρησιμοποιείται σε διάφορα πλήρη προγράμματα προγραμματισμού γραμμικών και συστήματα προγραμματισμού περιορισμών. Μια συζήτηση του SIMPLEX αλγόριθμου θα μας πήγαινε πάρα πολύ μακριά.

4.5 Περίληψη

Σε αυτό το κεφάλαιο συζητήσαμε για τρεις πλήρεις επιλυτές περιορισμών. Ασχολήθηκαν με:

- τις εξισώσεις όρου,
- τις γραμμικές εξισώσεις πέρα από τους πραγματικούς αριθμούς, και
- τις γραμμικές ανισότητες πέρα από τους πραγματικούς.

Στην παρουσίασή μας στοχεύσαμε σε μια ομοιόμορφη παρουσίαση αυτών των επιλυτών. Συνεπώς, τους εξετάσαμε χρησιμοποιώντας ένα απλό θεωρητικό πλαίσιο απόδειξης. Αυτή η παρουσίαση μας επέτρεψε να διευκρινίσουμε τις ομοιότητες μεταξύ αυτών των επιλυτών.

5

Αναζήτηση

Σκοπός των αλγόριθμων διάδοσης περιορισμών είναι να επιτευχθεί μία μορφή τοπικής σταθερότητας σε ένα CSP. Παραδείγματος χάριν, όπως ήδη σημειώνεται στην περίπτωση του Προβλήματος των n Βασιλισσών, όπου n ≥ 4, τίποτα δεν κερδίζεται με την χρησιμοποίηση ενός αλγορίθμου τοξοειδής σταθερότητας επειδή αυτό το CSP είναι ήδη τοξοειδές σταθερό. Σε τέτοιες καταστάσεις πρόοδος μπορεί να επιτευχθεί μόνο με το διαχωρισμό του τρέχοντος CSP P σε δύο ή περισσότερα CSPs η ένωση των οποίων είναι ισοδύναμη με το P βάση του Καθορισμού 3.3. Γενικά μια τέτοια διάσπαση πετυχαίνεται με το διαχωρισμό μιας περιοχής ή με το διαχωρισμό ενός περιορισμού.

Έτσι το γενικό σχέδιο αποτελείται από μια εναλλασσόμενη χρήση της διάδοσης περιορισμών και του διαχωρισμού. Αυτό οδηγεί σε αυτό που ονομάζουμε δέντρα αναζήτησης. Σκοπός αυτού του κεφαλαίου είναι να αναφερθούν τα δέντρα αναζήτησης και οι πιό κοινοί αλγόριθμοι που χρησιμοποιούνται για να τα διερευνήσουν. Καλούμε αυτούς τους αλγόριθμους αλγόριθμους αναζήτησης. Εννοιολογικά, διαβάζοντας αυτό το κεφάλαιο, είναι χρήσιμο να έχουμε κατά νου το ακόλουθο σλόγκαν:

Αλγόριθμος αναζήτησης = Δέντρο Αναζήτησης + Αλγόριθμος Διερεύνησης.

Η παραπάνω εξίσωση υποτίθεται πως υποδηλώνει ότι κάθε αλγόριθμος αναζήτησης μπορεί να αντιμετωπισθεί ως αλγόριθμος εξερεύνησης ενός δέντρου αναζήτησης και όχι ότι κάθε τέτοιος αλγόριθμος κατασκευάζει αρχικά ένα δέντρο αναζήτησης και έπειτα το διερευνά. Στην πραγματικότητα, τα εξεταζόμενα δέντρα αναζήτησης κατασκευάζονται "συνεχώς", κατά τη διάρκεια εκτέλεσης του αλγόριθμου διερεύνησης.

Στην παρουσίασή μας δεν εκμεταλλευόμαστε πλήρως αυτήν την εξίσωση, δεδομένου ότι περιορίζουμε την παρουσίασή μας στους **top-down αλγόριθμους αναζήτησης**, οι οποίοι αποκαλούνται και **depth-first αλγόρίθμοι αναζήτησης**. Σε αυτούς τους αλγορίθμους κάποιος επεκτείνει κατ' επανάληψη έναν κόμβο στο χαμηλότερο επίπεδο στο δέντρο έως ότου προκύψει μια αποτυχία, επάνω στην οποία κάποιος επιστρέφει (αντίστροφες διαδρομές) σε πιό υψηλό επίπεδο στο οποίο κάποιος επαναλαμβάνει την επέκταση των κόμβων. Στο επόμενο κεφάλαιο, δίνουμε μια σύντομη περιγραφή των εναλλακτικών μορφών των αλγόριθμων αναζήτησης.

Αρχίζουμε την παρουσίασή μας εισάγοντας στο επόμενο τμήμα μια πολύ γενική έννοια ενός δέντρου αναζήτησης που είναι επαρκής για να περιγράψει την εναλλαγή μεταξύ των αυθαίρετων μορφών διάδοσης περιορισμών και του διαχωρισμού. Έπειτα στην Παράγραφο 5.2 συζητάμε για τον πιό κοινό τύπο δέντρων αναζήτησης, τα οποία καλούμε δέντρα μαρκαρίσματος. Ισχύουν χρησιμοποιώντας τη μέθοδο μαρκαρίσματος της Παραγράφου 3.2 ως διαχωρισμό και μια μέθοδο μείωσης περιοχών ως διάδοση περιορισμών. Για να επεξηγήσουμε αυτά τα δέντρα στην Παράγραφο 5.3 συζητάμε λεπτομερώς για τρεις τύπους δέντρων μαρκαρίσματος για το CSP που αντιπροσωπεύει το παζλ SEND + MORE = MONEY του Παραδείγματος 2.1.

Έπειτα, στην Παράγραφο 5.4, κάνουμε λόγο για τα δέντρα μαρκαρίσματος που προκύπτουν από την επιλογή τριών συγκεκριμένων μορφών διάδοσης περιορισμών. Με αυτόν τον τρόπο πετυχαίνουμε

- forward checking,
- partial look ahead, και
- διατηρήσιμης τοξοειδούς σταθερότητας (MAC), αποκαλούμενη μερικές φορές full look ahead,

δέντρα αναζήτησης.

Μετά από αυτήν την έκθεση των δέντρων αναζήτησης προχωράμε στην παρουσίαση των αλγόριθμων αναζήτησης, πιό συγκεκριμένα των top-down αλγορίθμων αναζήτησης. Στην Παράγραφο 5.5 παρουσιάζουμε στη συνέχεια

- τον BACKTRACK-FREE αλγόριθμος αναζήτησης,
- τον BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμο αναζήτησης,
- τον BACKTRACK αλγόριθμος αναζήτησης, και
- τον BACKTRACK WITH CONSTRAINT PROPAGATION αλγόριθμο αναζήτησης,

Κατόπιν, στην Παράγραφο 5.6 αναφέρουμε τρεις ειδικεύσεις του τελευταίου αλγόριθμου αναζήτησης, για τα

- forward checking,
- partial look ahead, και
- MAC

δέντρα αναζήτησης. Στην Παράγραφο 5.7 στρέφουμε την προσοχή μας στους αλγόριθμους αναζήτησης για τα περιορισμένα προβλήματα βελτιστοποίησης που εισάγονται στη Παράγραφο 2.6. Συζητάμε στη συνέχεια για

- τον BRANCH AND BOUND αλγόριθμος αναζήτησης,
- τον BRANCH AND BOUND WITH CONSTRAINT PROPAGATION αλγόριθμο αναζήτησης, και
- τον BRANCH AND BOUND WITH CONSTRAINT PROPAGATION AND COST αλγόριθμο αναζήτησης.

Έπειτα, στην Παράγραφο 5.8, συζητάμε για διάφορες ευρετικές μεθόδους που είναι χρήσιμες στα πλαίσια των αλγόριθμων αναζήτησης για τα δέντρα μαρκαρίσματος.

Οι αλγόριθμοι αναζήτησης που αναφέρονται στις Παραγράφους 5.5-5.8 ασχολούνται όλοι με τα δέντρα μαρκαρίσματος. Οι περισσότεροι από αυτούς τους αλγόριθμους μπορούν επίσης να διατυπωθούν για τα αυθαίρετα δέντρα αναζήτησης. Ολοκληρώνουμε την έκθεση εξετάζοντας στην Παράγραφο 5.9 το_παράδειγμα αυτού του αλγόριθμου: μια θεωρητική διακλάδωση και ένας κατευθυνόμενος αλγόριθμος για τα αυθαίρετα δέντρα αναζήτησης.

5.1 Δέντρα αναζήτησης

Κατ' αρχήν, παρέχουμε έναν επίσημο ορισμό του δέντρου αναζήτησης. Γενικά σε αυτό το κεφάλαιο ορίζουμε σε κάθε κόμβο σε ένα δέντρο ένα **επίπεδο** ως εξής. Η ρίζα είναι στο επίπεδο 0 και οι άμεσοι απόγονοι ενός κόμβου στο επίπεδο i είναι όλοι στο επίπεδο i+1.

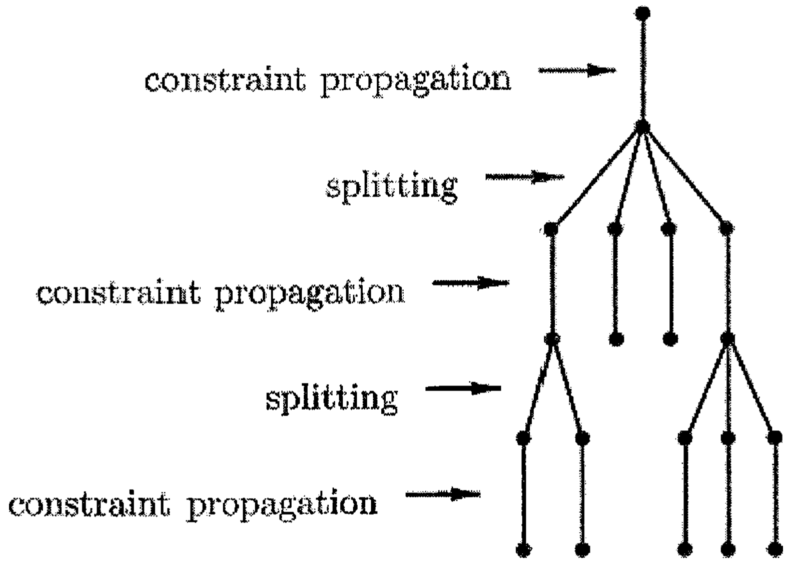
Καθορισμός 5.1 Θεωρήστε ένα CSP P με μια ακολουθία μεταβλητών X. Από ένα **δέντρο αναζήτησης για** το P δηλώνουμε ένα πεπερασμένο δέντρο έτσι ώστε

- οι κόμβοι του να είναι CSPs,
- η ρίζα του να είναι P,
- οι κόμβοι σε άρτιο επίπεδο να έχουν ακριβώς έναν άμεσο απόγονο,
- εάν $P_1,\ldots,P_m$ όπου $m\geq 1$, είναι άμεσοι απόγονοι του P_0, τότε η ένωση του P1,...,Pm είναι ισοδύναμη w.r.t. X στο P.

Έτσι ένα δέντρο αναζήτησης κατά γενική ομολογία είναι μια πολύ γενική έννοια. Η ιδέα είναι ότι:

- στα άρτια επίπεδα εφαρμόζεται η διάδοση περιορισμών στο τρέχον CSP,
- στα μονά επίπεδα εφαρμόζεται ο διαχωρισμός στο τρέχον CSP,

και ότι συνεχίζουμε να εφαρμόζουμε αυτήν την διαδικασία έως ότου φθάσουμε ένα φύλλο που είναι ένα "προφανώς λυμένο" CSP ή φθάσουμε φύλλα εκ των οποίων όλα είναι "προφανώς αποτυχημένα". Ένα παράδειγμα ενός δέντρου αναζήτησης στο οποίο αναφερόμαστε συγκεκριμένα στη διάδοση περιορισμών και στο διαχωρισμό απεικονίζεται στο Σχήμα 5.1



Σχήμα 5.1. Ένα δέντρο αναζήτησης για ένα CSP

Η αναφορά "προφανώς" σημαίνει ότι είναι απλό να παράγουμε όλες τις λύσεις στο CSP, σε περίπτωση που είναι "προφανώς λυμένο", ή να ορίσουμε ότι καμία λύση δεν υπάρχει σ'αυτό, σε περίπτωση που είναι "προφανώς αποτυχημένο". Αυτή είναι κατά γενική ομολογία μια ανακριβής περιγραφή ανάλογη με την έννοια ενός πλήρη επιλυτή που εισάγεται στην αρχή του Κεφαλαίου 4. Όπως θα δούμε στο επόμενο τμήμα διαφορετικές ακριβολογίες της έννοιας ενός "προφανώς αποτυχημένου" CSP οδηγεί σε διάφορους τύπους δέντρων αναζήτησης.

Στον παραπάνω ορισμό η διάδοση περιορισμών και ο διαχωρισμός εφαρμόζονται με έναν εναλλασσόμενο τρόπο. Ένας τέτοιος γενικός ορισμός μας επιτρέπει να διαμορφώσουμε τις αυθαίρετες μορφές διάδοσης περιορισμών και διαχωρισμού. Στην πραγματικότητα, μας δίνει τη δυνατότητα να εφαρμόζουμε σε κάθε κόμβο μια διαφορετική μορφή διάδοσης περιορισμών ή διαχωρισμού. Ο διαχωρισμός, όπως σημειώνεται στην Παράγραφο 3.2, μπορεί να αποτελείται από το διαχωρισμό ενός περιορισμού ή το διαχωρισμό της περιοχής μιας μεταβλητής.

Δεδομένου ότι η ρίζα του δέντρου είναι σε άρτιο επίπεδο, η διάδοση περιορισμών εφαρμόζεται πρώτα στο αρχικό CSP P. Μπαίνουμε στον πειρασμό να συμφωνήσουμε ότι οι κόμβοι σε μονό επίπεδο έχουν δύο τουλάχιστον άμεσους απογόνους και χρησιμοποιούνται όλοι με τη βοήθεια του διαχωρισμού. Εντούτοις, για λόγους που θα γίνουν σαφείς στον επόμενο τμήμα, θέλουμε εδώ επίσης να εισάγουμε την περίπτωση άκρων όταν μειωθεί ο διαχωρισμός σε μια "μη-δράση", αντιπροσωπευόμενος από ένα αντίγραφο του κόμβου.

Το ακόλουθο απλό συμπέρασμα συνοψίζει τη θεμελιώδη ιδιότητα των δέντρων αναζήτησης.

Σημείωση 5.2 (Δέντρο Αναζήτησης) *Θεωρήστε ένα P με μια ακολουθία μεταβλητών X. Υποθέστε ότι $P_1,\ldots,P_m$ είναι τα φύλλα ενός δέντρου αναζήτησης για το P. Κατόπιν η ένωση του $P_1,\ldots,P_m$ είναι ισοδύναμη w.r.t. X στο P.*

Απόδειξη Η απόδειξη παρουσιάζεται ως Άσκηση 5.1.

Αυτή η σημείωση δείχνει ότι δεδομένου ενός CSP P και ενός δέντρου αναζήτησης γι' αυτό, κατά την αναζήτηση μιας λύσης στο P είναι ικανοποιητικό να αναζητηθεί μια λύση σε ένα από το φύλλα CSPs.

Τέλος, σημειώστε ότι δεν θεωρήσαμε τίποτα δεδομένο για τους εσωτερικούς, δηλ., 'χωρίς-φύλλο', κόμβους. Αυτά τα CSPs μπορούν να είναι μεταβλητά ακόμη κι αν απέτυχαν. Τέτοιες καταστάσεις προκύπτουν όταν, παραδείγματος χάριν, το αρχικό CSP είναι μεταβλητό αλλά αυτό το γεγονός εντοπίζεται μόνο με τη βοήθεια της αφαίρεσης που διαμορφώνεται από τη διάδοση περιορισμών και της περιπτωσιολογικής ανάλυσης που διαμορφώνεται από το διαχωρισμό.

Από εδώ και στο εξής υποθέτουμε ότι όλα τα εξεταζόμενα CSPs είναι πεπερασμένα, δηλ., ότι οι περιοχές τους είναι ορισμένες.

5.2 Δέντρα μαρκαρίσματος

Η πιό κοινή μορφή διαχωρισμού αποτελείται από το ***μαρκάρισμα*** της περιοχής μιας μεταβλητής. Αντιστοιχεί στον κανόνα μαρκαρίσματος που αναφέρεται στην Παράγραφο 3.2. Ανεπίσημα, αποτελείται από τη λήψη μιας μεταβλητής, για παράδειγμα x, και το διαχωρισμό της περιοχής της σε ορισμένα μονοσύνολα. Κάθε τέτοιο μονοσύνολο, για παράδειγμα {α} , αντιστοιχεί σε ένα CSP στο οποίο η περιοχή της μεταβλητής x αντικαθίσταται από το {α}. Ισοδυνάμως, κάθε τέτοιο μονοσύνολο σύνολο {α} αντιστοιχεί σε μια ανάθεση της τιμής α στη μεταβλητή x.

Σε αυτό το τμήμα ορίζουμε διάφορους τύπους δέντρων αναζήτησης για πεπερασμένα CSPs. Για κάθε έναν από αυτούς:

•ο διαχωρισμός αποτελείται από το μαρκάρισμα,

•η διάδοση περιορισμών αποτελείται από μια μέθοδο μείωσης περιοχών.

Καλούμε τέτοια δέντρα αναζήτησης ***δέντρα μαρκαρίσματος***. Μια εναλλακτική εξήγηση θα οδηγούσε στα ***δέντρα απαρίθμησης*** που προκύπτουν από τη χρησιμοποίηση του κανόνα απαρίθμησης της Παραγράφου 3.2 αντί του μαρκαρίσματος. Έτσι τα δέντρα απαρίθμησης, σε αντίθεση με τα δέντρα μαρκαρίσματος, είναι δυαδικά. Η εστίασή μας στα δέντρα μαρκαρίσματος αντί των δέντρων απαρίθμησης παρακινείται από το γεγονός ότι στο επόμενο κεφάλαιο θα αναφερθούμε πιο διεξοδικά στο μαρκάρισμα απ' ότι στην απαρίθμηση.

Όπως θα δούμε, σε διάφορες καταστάσεις θα είναι δυνατό να ορίσουμε τα δέντρα μαρκαρίσματος με έναν απλούστερο τρόπο, χωρίς την χρησιμοποίηση των κόμβων που είναι CSPs.

5.2.1 Πλήρη δέντρα μαρκαρίσματος

Ξεκινάμε τη συζήτηση όσον αφορά τα δέντρα μαρκαρίσματος σε περίπτωση που καμία διάδοση περιορισμών δεν είναι διαθέσιμη. Έπειτα οι κόμβοι στα μονά επίπεδα, που αντιστοιχούν στο αποτέλεσμα της διάδοσης περιορισμών, μπορούν να αγνοηθούν και το δέντρο που προκύπτει μπορεί να απλοποιηθεί με την αφαίρεση τους. Περαιτέρω, το δέντρο αναζήτησης μπορεί να οριστεί αμέσως όσον αφορά τα στιγμιαία, χωρίς άμεση εισαγωγή οποιωνδήποτε CSPs. Ο επίσημος ορισμός είναι ο ακόλουθος.

Ορισμός 5.3 Θεωρήστε ένα CSP P. Αφήστε την $x_1,...,x_n$ να είναι η ακολουθία των μεταβλητών του που διατάσσονται γραμμικά με <. Από ένα ***πλήρες δέντρο μαρκαρίσματος που συνδέεται με*** το P ***και*** το < υπολογίζουμε ένα δέντρο έτσι ώστε

- οι άμεσοι απόγονοι της ρίζας να είναι της μορφής (x_1, d),
- οι άμεσοι απόγονοι ενός κόμβου (xj, d), όπου το j Ε [l..n —1], να είναι της μορφής (x_{j+1}, e),
- οι διακλαδώσεις του να ορίζουν όλα τα στιγμιαία με την περιοχή $\{x_1,...,x_n\}$.

Για να διευκρινίσουμε τη σχέση μεταξύ αυτού του ορισμού και του Ορισμού 5.1 ας συνδέσουμε με κάθε πλήρες δέντρο μαρκαρίσματος ένα μοναδικό δέντρο αναζήτησης ως εξής. Ξεκινώντας, συνδέουμε το αρχικό CSP P με τη ρίζα του πλήρες δέντρου μαρκαρίσματος. Έπειτα μαζί με κάθε κόμβο (x_k, d_k) με το μονοπάτι $\{(x_1, d_1),...,(x_k, d_k)\}$ που οδηγεί σ' αυτό συνδέουμε ένα μοναδικό CSP που λαμβάνεται από το P τροποποιώντας τις περιοχές των μεταβλητών $x_1,...,x_k$, αντίστοιχα σε, $\{a_1\},...\{a_k\}$ και περιορίζοντας τους περιορισμούς P σε αυτές τις νέες περιοχές. Τέλος, για να διευθετήσουμε τη διάδοση περιορισμών, που εδώ είναι η "ανεξάρτητη δράση", επεκτείνουμε κάθε κόμβο r σε ένα τόξο με τον κόμβο r σε κάθε άκρο. Έτσι ένα πλήρες δέντρο μαρκαρίσματος μπορεί να αντιμετωπισθεί ως απλουστευμένη αντιπροσώπευση του δέντρου αναζήτησης με την τετριμμένη διάδοση περιορισμών και το μαρκάρισμα ως μέθοδο διαχωρισμού. Ένα πλήρες δέντρο μαρκαρίσματος είναι επίσης ένας "συνεπτυγμένος" τρόπος αντιπροσώπευσης όλων των πιθανών στιγμιαίων.

Στην περίπτωση των πλήρη δέντρων μαρκαρίσματος και τα δύο CSPs "προφανώς λυμένο " και "προφανώς αποτυχημένο" αντιστοιχούν σε όλες τις περιοχές που είναι ορισμένα μονοσύνολα. Έτσι ανιχνεύουμε τη θέση ενός CSP μόνο στην περίπτωση που έχουμε εφαρμόσει το μαρκάρισμα σε όλες τις μεταβλητές του.

Σημειώστε δέντρο μαρκαρίσματος που συνδέεται με ένα CSP εξαρτάται από τον τρόπο που διατάσσουμε τις μεταβλητές του.

Παράδειγμα 5.4 Θεωρήστε το ακόλουθο CSP:

$$\langle x < y, y < z \; ; \; x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\}\rangle.$$

Εάν διατάξουμε τις μεταβλητές του ως x< y< z, τότε παίρνουμε το πλήρες δέντρο μαρκαρίσματος που απεικονίζεται στο Σχήμα 5.2, ενώ για τη σειρά x< z< y παίρνουμε το πλήρες δέντρο μαρκαρίσματος που απεικονίζεται στο Σχήμα 5.3.

Σημειώστε ότι το πρώτο πλήρες δέντρο μαρκαρίσματος έχει 28 κόμβους, ενώ το δεύτερο 31 κόμβους. Εν αντιθέσει, και οι δύο έχουν τον ίδιο αριθμό φύλλων, δηλαδή 18. Έτσι το μέγεθος του πλήρες δέντρου μαρκαρίσματος εξαρτάται από την σειρά των μεταβλητών. Στην πραγματικότητα, μπορούμε να είμαστε ακριβέστεροι.

Σημείωση 5.5 (Μαρκάρισμα) *Ορίστε ένα CSP με μη-άδειες περιοχές. Αφήστε την $x_1,\ldots,x_n$ να είναι η ακολουθία μεταβλητών του CSP που διατάσσεται γραμμικά με < και $D_1,\ldots,D_n$ οι αντίστοιχες περιοχές.*

(i) Ο αριθμός των κόμβων στοπ πλήρες δέντρο μαρκαρίσματος που συνδέονται με < είναι ίσος με

$$1 + \Sigma_{i=1}^{n}(\Pi_{j=1}^{i}|D_j|),$$

με όπου το |A| δείχνει τον αριθμό στοιχείων συνόλου του ορισμένου A.

(ii) Ο αριθμός των φύλλων στο πλήρες δέντρο μαρκαρίσματος που συνδέονται με < είναι ίσος με

$$\Pi_{j=1}^{n}|D_j|,$$

έτσι δεν εξαρτάται από τη σειρά των μεταβλητών.

(iii) Το δέντρο μαρκαρίσματος έχει τον μικρότερο αριθμό κόμβων εάν οι μεταβλητές διατάσσονται από τα μεγέθη των περιοχών τους στην

αυξανόμενη σειρά.

Απόδειξη

(i) Θυμηθείτε ότι υποθέσαμε πως η ρίζα ενός δέντρου είναι στο επίπεδο 0 και οι άμεσοι απόγονοι ενός κόμβου στο επιπεδο i είναι όλοι στο επίπεδο i+1. Μία άμεση συνέπεια από την εισαγωγή δείχνει ότι ο αριθμός των κόμβων στο επίπεδο i είναι $\Pi^{i}_{i=1} = |D_J|$.

Αυτό αποδεικνύει την αξίωση.

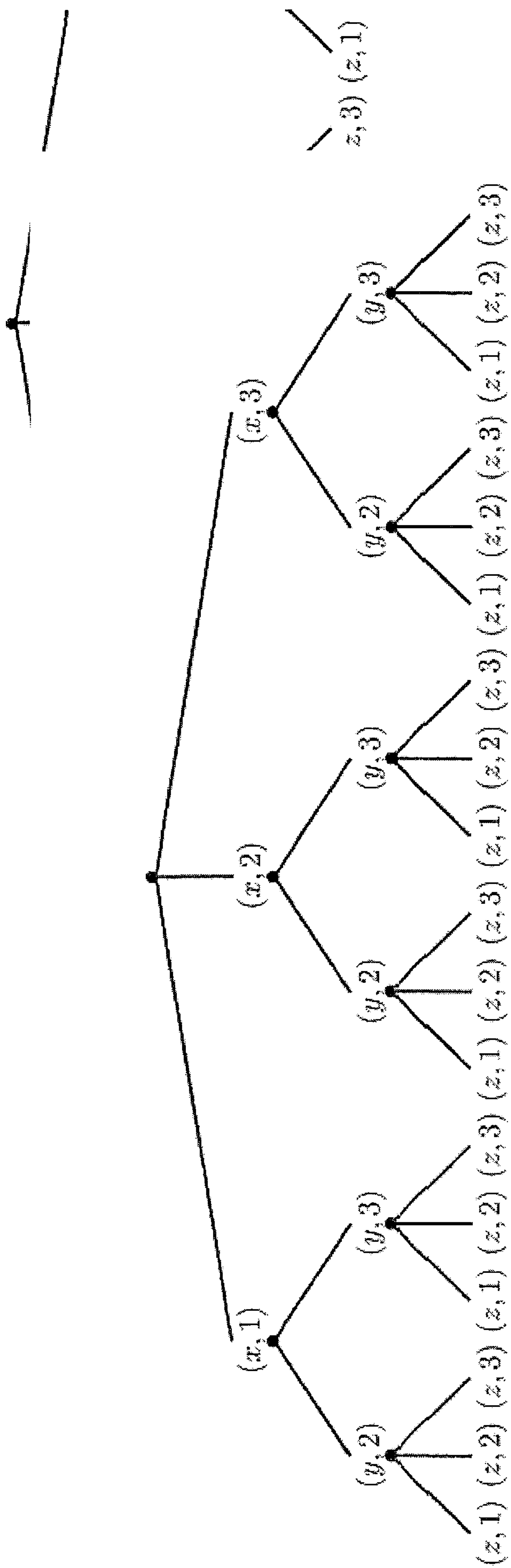
(ii) Όλα τα φύλλα βρίσκονται στο επίπεδο n, έτσι η απόδειξη (i) μας παρέχει τον αριθμό τους

(iii) Εάν αλληλομεταθέσουμε στη σειρά < τις μεταβλητές x_{j+1} και x_{j+2}, τότε ο αριθμός των κόμβων στο δέντρο μαρκαρίσματος αλλάζει με

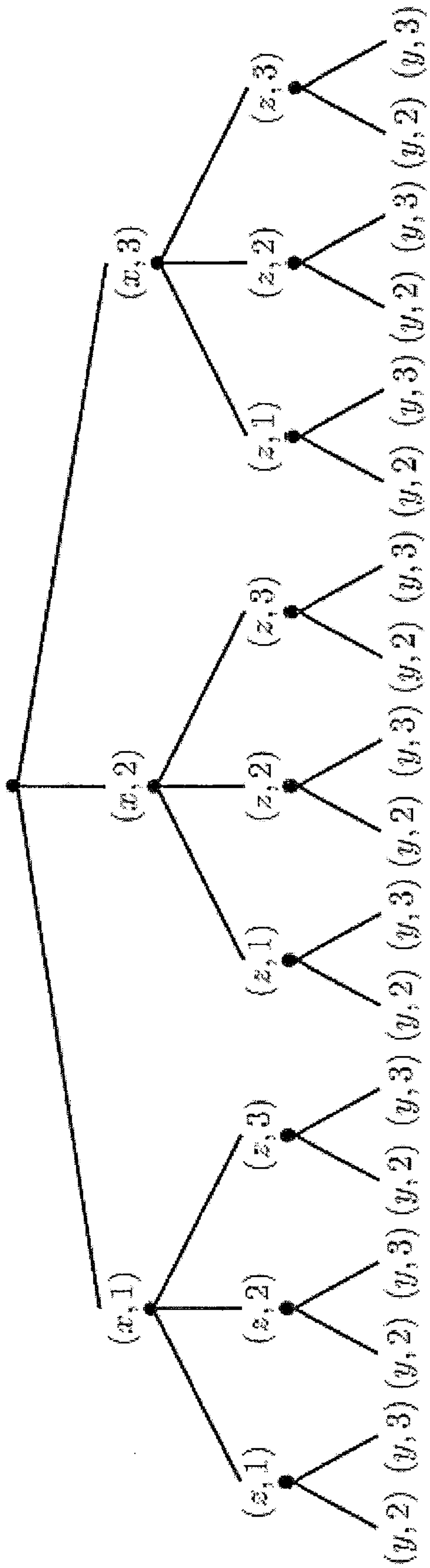
$$(\Pi^{i}_{j=1}|D_j|) \cdot (|D_{j+2}| - |D_{j+1}|).$$

Κατά συνέπεια εάν $|D_{J+2}| < |D_{J+1}|$ έπειτα μετά από αυτήν την αλληλομετάθεση το μέγεθος του δέντρου αναζήτησης μειώνεται. Με άλλα λόγια, εάν για κάποιο j Ε [0..n —2] έχουμε $|D_{J+1}| > |D_{J+2}|$,τότε το μέγεθος του δέντρου αναζήτησης δεν είναι ελάχιστο. Αυτό αποδεικνύει η αξίωση.

Αυτή η παρατήρηση υποδηλώνει ότι όταν δεν υπάρχουν άλλες πληροφορίες είναι χρήσιμο να επιλεχτούν πρώτα οι μεταβλητές με τις μικρότερες περιοχές στους αλγόριθμους αναζήτησης.



Σχήμα 5.2 Το πλήρες δέντρο μαρκαρίσματος για τη σειρά x<y<z



Σχήμα 5.3 Το πλήρες δέντρο μαρκαρίσματος για τη σειρά x<z<y

5.2.2 Μειωμένα δέντρα μαρκαρίσματος

Την ανάθεση μιας τιμής σε μια μεταβλητή θα επιθυμούσαμε να την κάνουμε κατά τέτοιο τρόπο ώστε να είναι σταθερή μέχρι τώρα η στιγμιαία κατασκευή. Αυτό οδηγεί στα δέντρα μαρκαρίσματος τα οποία ονομάζουμε μειωμένα δέντρα μαρκαρίσματος. Επίσης αυτά τα δέντρα μπορούν να οριστούν σε σχέση με τα στιγμιαία. Ο ακριβής ορισμός απαιτεί την εισαγωγή μιας βοηθητικής έννοιας.

Ορισμός 5.6 Λέμε ότι ένα στιγμιαίο I είναι **κατά μήκος της σειράς** $x_1,\ldots x_n$ εάν η περιοχή του είναι της μορφής $\{x_1,\ldots x_j\}$ για κάποιο j Ε [1..n].

Θυμηθείτε τώρα ότι, δεδομένου ενός CSP, ένα στιγμιαίο με την περιοχή X είναι σταθερό εάν ικανοποιεί όλους τους περιορισμούς στις υποακολουθίες των μεταβλητών από το X.

Ορισμός 5.7 Θεωρήστε ένα CSP P με μη-άδειες περιοχές. Αφήστε την $x_1,\ldots x_n$ να είναι η ακολουθία των μεταβλητών του που διατάσσονται γραμμικά με <. Από ένα ***μειωμένο δέντρο μαρκαρίσματος συνδεδεμένο με*** το P ***και*** το < υπολογίζουμε ένα δέντρο έτσι ώστε

- οι άμεσοι απόγονοι της ρίζας να είναι της μορφής (x_1, d),
- οι άμεσοι απόγονοι ενός κόμβου (x_j, d) $_T$ όπου το j Ε [1..n —1], να είναι της μορφής (x_{j+1}, e),
- οι διακλαδώσεις του να ορίζουν όλα τα σταθερά στιγμιαία κατά μήκος της σειράς $x_1,\ldots x_n$.

Για να συσχετίσουμε τα μειωμένα δέντρα μαρκαρίσματος με τα δέντρα αναζήτησης προχωράμε ομοίως όπως στην περίπτωση των πλήρη δέντρων μαρκαρίσματος. Δηλαδή μαζί με κάθε κόμβο (x_k, d_k) με το μονοπάτι $\{(x_1, d_1),\ldots,(x_k, d_k)\}$που οδηγεί σ' αυτό συνδέουμε ένα μοναδικό CSP που ορίζεται από το P και τις περιοχές $\{\alpha_1\},\ldots\{\alpha_k\}$ των μεταβλητών $x_1,\ldots,x_k$ και επεκτείνουμε στη συνέχεια κάθε κόμβο σε ένα τόξο. Είναι εύκολο να δούμε ότι το δέντρο που προκύπτει είναι ένα δέντρο αναζήτησης.

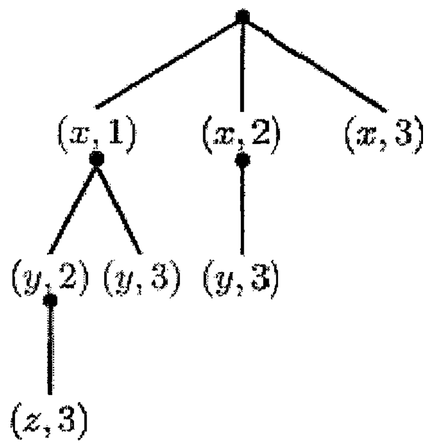
Έτσι ένα μειωμένο δέντρο μαρκαρίσματος μπορεί να αντιμετωπισθεί ως απλουστευμένη αντιπροσώπευση του δέντρου αναζήτησης στο οποίο ο διαχωρισμός συνίσταται από το μαρκάρισμα που περιορίζεται σε εκείνες τις τιμές για τις οποίες η στιγμιαία κατασκευή παραμένει μέχρι τώρα σταθερή. Το γεγονός ότι εξετάζουμε μόνο σταθερά στιγμιαία εκφράζει την άποψη ότι τώρα εξετάζουμε ένα CSP ως "προφανώς αποτυχημένο" νωρίτερα απ' ότι στα πλήρη δέντρα μαρκαρίσματος, δηλαδή προς το παρόν παράγεται ένα σταθερό στιγμιαίο που δεν μπορεί να επεκταθεί σταθερά στην επόμενη μεταβλητή.

Ακριβώς όπως το πλήρες δέντρο μαρκαρίσματος, το μειωμένο δέντρο μαρκαρίσματος που συνδέεται με ένα CSP εξαρτάται από τον τρόπο που διατάσσουμε τις μεταβλητές του. Για τις διαφορετικές σειρές μεταβλητών μπορούμε να λάβουμε θεμελιωδώς διαφορετικά δέντρα.

Παράδειγμα 5.8 Θεωρήστε πάλι το CSP του Παραδείγματος 5.4, έτσι ώστε

$$\langle x < y, y < z \,;\, x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\}\rangle$$

και τις δύο ήδη εξεταζόμενες σειρές μεταβλητών: x < y < z και x < z < y . Για την πρώτη παίρνουμε το μειωμένο δέντρο μαρκαρίσματος που απεικονίζεται στο Σχήμα 5.4, ενώ για τη δεύτερη παίρνουμε το μειωμένο δέντρο μαρκαρίσματος που απεικονίζεται στο Σχήμα 5.5.

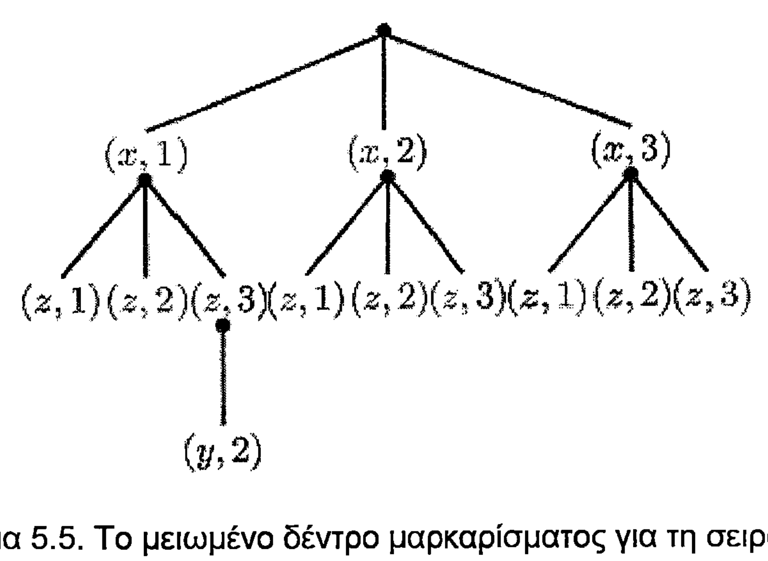


Σχήμα 5.4. Το μειωμένο δέντρο μαρκαρίσματος για τη σειρά x<y<z.

Το μέγεθος του μειωμένου δέντρου μαρκαρίσματος εξαρτάται όχι μόνο από το μέγεθος των περιοχών των μεταβλητών αλλά και από τους εξεταζόμενους περιορισμούς. Επομένως, γενικά είναι αρκετά περίπλοκο να υπολογιστεί το μέγεθός του.

Σημειώστε ότι στους ορισμούς ενός πλήρες δέντρου μαρκαρίσματος και ενός μειωμένου δέντρου μαρκαρίσματος αγνοήσαμε τη σειρά μεταξύ των άμεσων απογόνων. Φυσικά, στα σχήματα όπως τα παραπάνω δεν είναι δυνατό να τους υπεξαιρέσουμε από μια τέτοια σειρά. Διατάξαμε σε αυτά τους άμεσους απογόνους σύμφωνα με την τιμή των επιλεγμένων στοιχείων.

Γενικά, η σειρά μεταξύ των άμεσων απογόνων έχει σημασία για τους αλγόριθμους αναζήτησης που υιοθετούνται επειδή επισκέπτονται τους κόμβους με μια συγκεκριμένη σειρά. Εντούτοις, εκτός από τους backtrack-free(ελεύθερης-αντίστροφης διαδρομής) αλγόριθμους αναζήτησης, η έκβαση των



Σχήμα 5.5. Το μειωμένο δέντρο μαρκαρίσματος για τη σειρά x<z<y

εξεταζόμενων αλγόριθμων δεν βασίζεται σε αυτή τη σειρά. Επομένως, σε κάθε έναν από αυτούς τα στοιχεία θα επιλεχτούν με έναν μη-ντετερμινιστικό τρόπο.

5.2.3 *Prop* δέντρα μαρκαρίσματος

Τώρα προχωράμε προς την εξέταση των δέντρων μαρκαρίσματος παρουσία μιας διάδοσης περιορισμών. Έχει επιπτώσεις στις περιοχές των όχι ακόμα εξεταζόμενων μεταβλητών. Η ιδέα είναι ότι η διάδοση περιορισμών μας επιτρέπει να ανιχνεύσουμε νωρίτερα ότι παράγεται ένα "προφανώς αποτυχημένο" CSP και συνεπώς τα δέντρα αναζήτησης που προκύπτουν είναι γενικά μικρότερα από αυτά που αναφέρονται στις προηγούμενες υποενότητες.

Για να εξετάσουμε αυτά τα δέντρα δεν μπορούμε να χρησιμοποιήσουμε άλλο σωστά στιγμιαία. Παρ' όλα αυτά, μπορούμε να αντιπροσωπεύσουμε κάθε κόμβο από μια ακολουθία περιοχών των εξεταζόμενων μεταβλητών, χωρίς να κάνουμε συχνή χρήση των CSPs. Το CSP που αντιστοιχεί σε αυτό τον κόμβο ορίζεται μεμονωμένα από αυτές τις εκφράσεις περιοχών και το αρχικό CSP P: οι περιορισμοί του λαμβάνονται με την χρήση των περιορισμών του P στις απαριθμημένες περιοχές.

Περαιτέρω, δεδομένου ότι επιθυμούμε τώρα να συμπεριλάβουμε μια αυθαίρετη διάδοση περιορισμών υπό μορφή μείωσης περιοχών, δεν διαγράφουμε άλλο τους κόμβους στο μονό επίπεδο. Στην πραγματικότητα, τώρα επιθυμούμε να διαμορφώσουμε το γεγονός ότι διαδοχικά μεταβλητά στιγμιαία συνδυάζονται με τις εφαρμογές της διάδοσης περιορισμών. Από τώρα και στο εξής υποθέτουμε λόγω ευθύτητας ότι η γραμμική σειρά < στις μεταβλητές συμπίπτει με τη σειρά που χρησιμοποιείται στον ορισμό του εξεταζόμενου CSP.

Θεωρήστε ένα CSP P := $\langle C \,;\, x_1 \in D_1,\ldots,x_n \in D_n \rangle$. Υποθέτουμε ότι μια σταθερή μορφή διάδοσης(propagation) περιορισμών prop(i) υπό μορφή μείωσης περιοχών, παραμετροποιείται από μια τιμή i Ε [0..n —1]. Η τιμή i ορίζει την ακολουθία x_{i+1} ,x_n των μεταβλητών στις περιοχές στις οποίες εφαρμόζεται το prop(i).

Δηλαδή, δεδομένης μιας ακολουθίας των τελευταίων περιοχών $E_1,\ldots,E_n$, η διάδοση περιορισμών prop(i) μετασχηματίζει μόνο τις περιοχές $E_{i+1},\ldots,E_n$. Η έκβαση prop (i) εξαρτάται από τους αρχικούς περιορισμούς C του P και από τις περιοχές $E_1,\ldots,E_i$. Επειδή εδώ συζητάμε για τα δέντρα μαρκαρίσματος, προς το παρόν πραγματοποιείται η διάδοση περιορισμών prop(i) και οι περιοχές $E_1,\ldots,E_i$ είναι όλες ορισμένα μονοσύνολα.

Ας δώσουμε τώρα τον επίσημο ορισμό των δέντρων αναζήτησης για τα οποία ενδιαφερόμαστε.

Ορισμός 5.9 Υποθέστε μια σταθερή μορφή διάδοσης περιορισμών prop(i) υπό μορφή μείωσης περιοχής όπως εξηγείται παραπάνω. Θεωρήστε ένα CSP P := $\langle C \,;\, x_1 \in D_1,\ldots,x_n \in D_n \rangle$. Από ένα ***prop δέντρο μαρκαρίσματος που συνδεδεμένο με*** το P υπολογίζουμε ένα δέντρο έτσι ώστε

- οι κόμβοι του να είναι ακολουθίες των εκφράσεων περιοχών $x_1 \in E_1,\ldots,x_n \in E_n$
- η ρίζα του να είναι η ακολουθία $x_1 \in D_1, x_2 \in D_2, x_n \in D_n$,
- κάθε κόμβος σε επίπεδο 2i με i Ε [0..n] να είναι της μορφής

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, \ldots, x_n \in E_n.$$

Εάν i=n, αυτός ο κόμβος να είναι ένα φύλλο. Διαφορετικά, να έχει ακριβώς έναν άμεσο απόγονο,

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E'_{i+1}, \ldots, x_n \in E'_n,$$

όπου $E'_j \subseteq E_j$ για j Ε [i+1..n], που λαμβάνεται με τη βοήθεια της διάδοσης περιορισμών prop(i),

- κάθε κόμβος σε επίπεδο 2i + 1 με i Ε [0..n —1] να είναι της μορφής

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, \ldots, x_n \in E_n.$$

Εάν Ej= $\emptyset$ για κάποιο j Ε [ι+1..n],αυτός ο κόμβος να είναι ένα φύλλο. Διαφορετικά να έχει άμεσους απογόνους της μορφής

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in \{d\}, x_{i+2} \in E_{i+2}, \ldots, x_n \in E_n,$$

για όλα τα d Ε E_{i+1} έτσι ώστε το στιγμιαίο $\{(x_1, d_1),\ldots,(x_i, d_i),(x_{i+1},d)\}$ να είναι σταθερό. Αυτό το σύνολο άμεσων απογόνων μπορεί να είναι ένα ορισμένο μονοσύνολο, ή κενό.

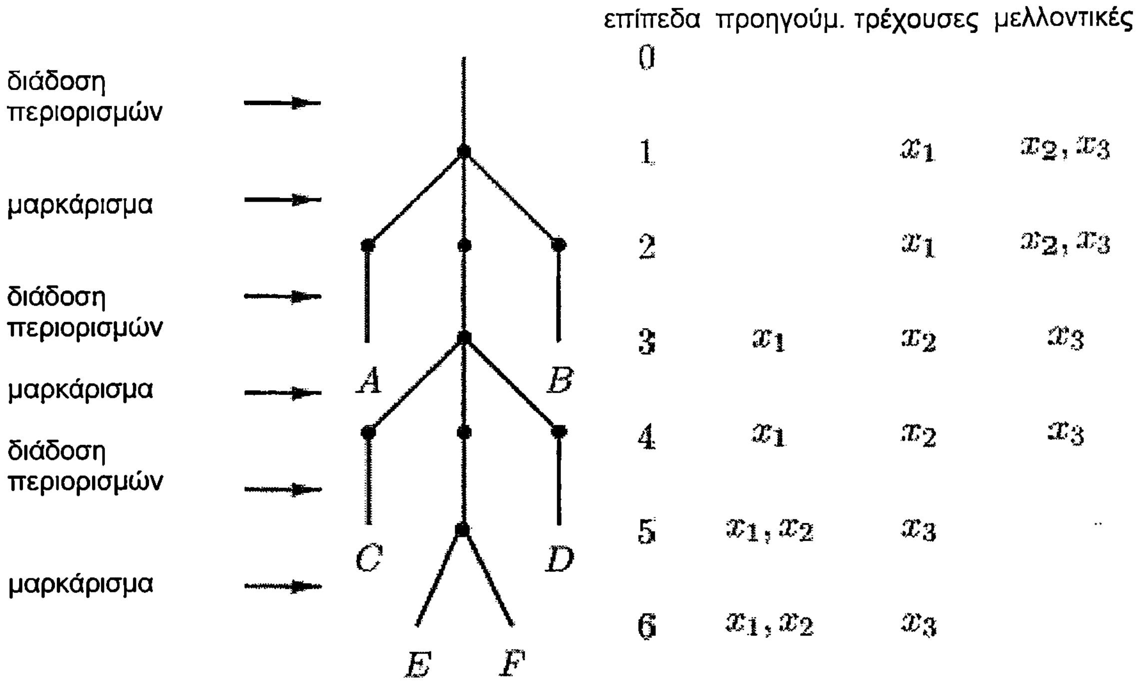
Καλούμε ένα φύλλο ***πετυχημένο κόμβο*** εάν βρίσκεται σε επίπεδο 2n και ***αποτυχημένο κόμβο*** (ή ***αποτυχία***) εάν βρίσκεται σε επίπεδο < 2n.

Ας διευκρινίσουμε τώρα μερικά στοιχεία αυτού του ορισμού. Λαμβάνοντας υπόψη τον κόμβο σε επίπεδο 0, δηλ., τη ρίζα δέντρου, ονομάζουμε $x_1,\ldots,x_n$ τις μελλοντικές μεταβλητές του. Λαμβάνοντας υπόψη έναν κόμβο σε επίπεδο 1 ή 2 ονομάζουμε x_1 την τρέχουσα μεταβλητή του και $x_2,\ldots,x_n$ τις μελλοντικές μεταβλητές του. Έπειτα, λαμβάνοντας υπόψη έναν κόμβο σε επίπεδο 3 ή 4 ονομάζουμε x_1 την προηγούμενη μεταβλητή του, x_2 την τρέχουσα μεταβλητή του και $x_3,\ldots x_n$ τις μελλοντικές μεταβλητές του. Και τα λοιπά.

Σαφέστερα, δεδομένου ενός κόμβου x_1 Ε $E_1,\ldots,x_n$ Ε E_n, σε επίπεδο 2i-1 ≥ 0 ή $2i \geq 0$,

- εάν i Ε [2..n —1], ονομάζουμε $x_1,\ldots,x_{i-1}$ τις **προηγούμενες μεταβλητές** του,
- εάν I Ε [1..n], ονομάζουμε x_i την **τρέχουσα μεταβλητή** του, και
- εάν i Ε [0..n —1], ονομάζουμε $x_{i+1},\ldots,x_n$ τις **μελλοντικές μεταβλητές** του.

Για παράδειγμα θεωρήστε ένα CSP με τρεις μεταβλητές, x_1, x_2 και x_3. Εξετάστε ένα prop δέντρο μαρκαρίσματος συνδεδεμένο με το P που απεικονίζεται στο Σχήμα 5.6 μαζί με τη λίστα των επιπέδων του και την αντιστοιχία των προηγούμενων, των πρόσφατων και των μελλοντικών μεταβλητών. Εδώ δεν λαμβάνουμε υπόψη τη δομή των κόμβων. Σύμφωνα με την παραπάνω ορολογία τα A, B, C και D είναι αποτυχημένοι κόμβοι, ενώ τα E και F είναι πετυχημένοι κόμβοι.



Σχήμα 5.6. Ένα prop δέντρο μαρκαρίσματος

Σημειώστε ότι οι περιοχές των προηγούμενων μεταβλητών είναι όλες ορισμένα μονοσύνολα. Περαιτέρω, εάν ένας κόμβος είναι σε επίπεδο 2i με i E [0..n —1], η διάδοση περιορισμών prop εφαρμόζεται σε αυτό. Ακριβέστερα, το prop(i) εφαρμόζεται στο CSP που ορίζεται μεμονωμένα από το αρχικό CSP και τις εκφράσεις περιοχής του κόμβου. Το αποτέλεσμα του αντιπροσωπεύεται από μια νέα ακολουθία των εκφράσεων περιοχής. Η διάδοση περιορισμών prop(i) επηρεάζει μόνο τις περιοχές των μελλοντικών μεταβλητών. Ένας κόμβος σε άρτιο επίπεδο 2i με i E [0..n] είναι ένα φύλλο εάν και μόνο εάν i = n. Ένας τέτοιος κόμβος αντιστοιχεί σε μια επιτυχία υπό την έννοια ότι ορίζει μια λύση στο εξεταζόμενο CSP.

Με την αύξηση του επιπέδου από 2i σε 2i + 1 η τρέχουσα μεταβλητή αλλάζει από xi σε x_{i+1}. Έτσι σε κάθε άρτιο επίπεδο εξετάζεται ένας κόμβος με μια νέα τρέχουσα μεταβλητή. Ένας κόμβος σε επίπεδο 2i + 1 είναι φύλλο σε μια από τις δύο περιπτώσεις:

- η περιοχή της τρέχουσας μεταβλητής x_{i+1} ή μιας μελλοντικής μεταβλητής είναι κενή,
- το στιγμιαίο $\{(x_1,d_1),\dots,(x_i,d_i)\}$ δεν μπορεί να επεκταθεί σταθερά στη μεταβλητή x_{i+1}

Και στις δύο περιπτώσεις ο κόμβος αντιστοιχεί σε ένα "προφανώς αποτυχημένο" CSP. Η πρώτη πιθανότητα οφείλεται στη χρήση της διάδοσης περιορισμών και είναι νέα όσον αφορά την προηγούμενη υποενότητα στην οποία εξετάσαμε τα μειωμένα δέντρα μαρκαρίσματος.

Εάν ο κόμβος στο επίπεδο 2i + 1 δεν είναι φύλλο, το μαρκάρισμα εφαρμόζεται στην τρέχουσα μεταβλητή x_{i+1} αλλά παραμένουν σταθερά μόνο εκείνα τα στοιχεία που έχουν επιλεχτεί στις περιοχές τους για τα οποία το στιγμιαίο κατασκευάστηκε μέχρι τώρα. Αυτό είναι ανάλογο με την περίπτωση των μειωμένων δέντρων μαρκαρίσματος.

Είναι απλό να δούμε ότι κάθε prop δέντρο μαρκαρίσματος αντιστοιχεί σε ένα μοναδικό δέντρο αναζήτησης. Σε αυτή την αντιστοιχία το prop(i) είναι η διάδοση περιορισμών που χρησιμοποιείται στα άρτια επίπεδα. Το prop(i) διατηρεί την ισοδυναμία w.r.t. το αρχικό σύνολο μεταβλητών και μειώνει τις περιοχές μεταβλητών. Αυτή η αντιστοιχία μας επιτρέπει να βγάλουμε το ακόλουθο σημαντικό συμπέρασμα βάση της Σημείωσης 5.2(Δέντρα Αναζήτησης).

Σημείωση 5.10 (prop Δέντρο Μαρκαρίσματος) *Θεωρήστε ένα πεπερασμένο CSP P και ένα prop δέντρο μαρκαρίσματος T που συνδέεται με το P. Ένας κόμβος της μορφής x_1 E $\{d_1\}$,...,x_n E $\{d_n\}$ είναι ένα φύλλο του T εάν $(d_1,\dots,d_n)$ είναι μια λύση στο P.*

Αυτό σημαίνει ότι δεδομένου_ενός πεπερασμένου CSP P, όλες οι λύσεις σ' αυτό μπορούν να βρεθούν με την διερεύνηση ενός αυθαίρετου prop δέντρου μαρκαρίσματος που συνδέεται με το P. Οι κατάλληλοι αλγόριθμοι θα παρουσιαστούν στην Παράγραφο 5.5. Όπως αναφέρεται στην αρχή αυτού του κεφαλαίου, στην πραγματικότητα αυτοί οι αλγόριθμοι δεν διερευνούν τα prop δέντρα μαρκαρίσματος, αλλά μάλλον τα κατασκευάζουν κατά τη διάρκεια της εκτέλεσής τους. Έπειτα η εκτέλεση αντιστοιχεί σε μια διερεύνηση του κατασκευασμένου prop δέντρου μαρκαρίσματος.

5.3 Ένα παράδειγμα: *SEND + MORE = MONEY*

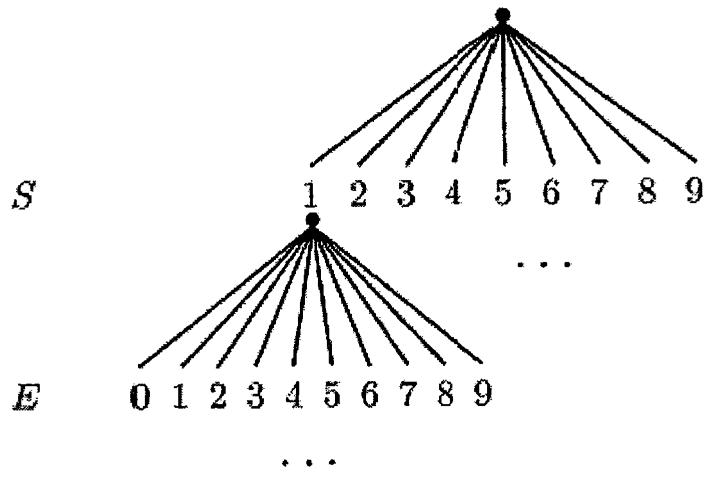
Τα μειωμένα δέντρα μαρκαρίσματος και τα pop δέντρα μαρκαρίσματος μπορούν να είναι ουσιαστικά μικρότερα από τα πλήρη δέντρα μαρκαρίσματος. Για να επεξηγήσουμε το σύνολο της οικονομίας που μπορεί να επιτευχθεί κατά τη χρησιμοποίηση τους επιστρέφουμε τώρα στο γρίφο SEND + MORE = MONEY που εισάγεται στο Παράδειγμα 2.1 του Κεφαλαίου 2.

Ας συζητήσουμε τώρα για τα αντίστοιχα δέντρα μαρκαρίσματος. Έτσι παίρνουμε ως CSP αυτό στο οποίο, έχουμε έναν περιορισμό ισότητας

$$\begin{array}{rrrrrrrrr} & & 1000\cdot S & + & 100\cdot E & + & 10\cdot N & + & D \\ & + & 1000\cdot M & + & 100\cdot O & + & 10\cdot R & + & E \\ = 10000\cdot M & + & 1000\cdot O & + & 100\cdot N & + & 10\cdot E & + & Y \end{array}$$

και 28 απλούς περιορισμούς ανισότητας $x \neq y$ για x, y να εκτείνονται πέρα από το σύνολο {S, E, N, D, M, O, R, Y} όπου το x προηγείται του y στην αλφαβητική σειρά, με την περιοχή [1..9] για το S και M και την περιοχή [0..9] για τις άλλες έξι μεταβλητές.

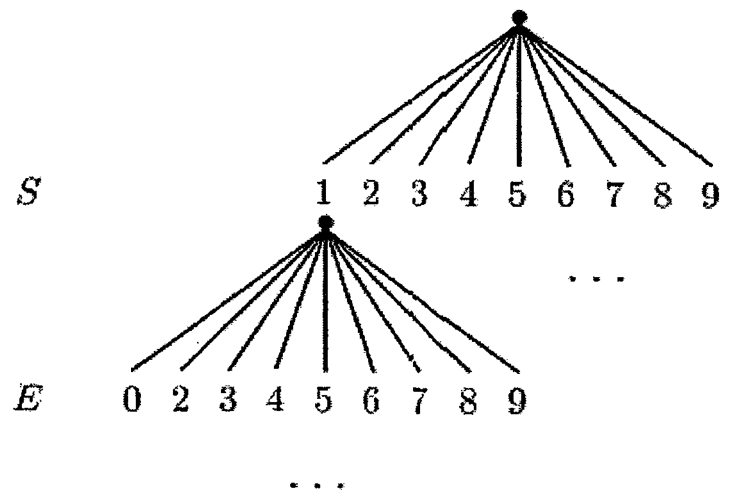
Το πλήρες δέντρο μαρκαρίσματος που συνδέεται με αυτό το CSP και η σειρά που επιδεικνύεται στις μεταβλητές είναι τεράστια: ο συνολικός αριθμός των φύλλων του είναι $9^2 \cdot 10^6$ = 81 000 000. Το αρχικό αποσπασθέν μέρος του απεικονίζεται στο Σχήμα 5.7. Εδώ και παρακάτω χρησιμοποιούμε μερικές αυτεξήγητες απλοποιήσεις στην αντιπροσώπευση των κόμβων.



Σχήμ.5.7. Το πλήρες δέντρο μαρκαρίσματος για το SEND+MORE=MONEY CSP

Το μειωμένο δέντρο μαρκαρίσματος είναι ουσιαστικά μικρότερο. Χρησιμοποιώντας τις εσωτερικές πληροφορίες σύμφωνα με τις οποίες αυτό το CSP έχει ακριβώς μια λύση μπορούμε να καταλήξουμε στο συμπέρασμα ότι ο συνολικός αριθμός φύλλων σε αυτό το δέντρο είναι 10. 9. 8. 7. 6. 5. 4- 2. (9. 8. 7. 6. 5. 4) = 483840. Αυτό αντιπροσωπεύει μία αύξηση 99,4% όσον αφορά το πλήρες δέντρο μαρκαρίσματος. Το αρχικό αποσπασθέν μέρος του απεικονίζεται στο Σχήμα 5.8. Σημειώστε ότι ο αριθμός των άμεσων απογόνων για τον κόμβο που χαρακτηρίζεται από το E τώρα είναι 9.

Το αποτέλεσμα της διάδοσης περιορισμών φαίνεται με έναν εντυπωσιακό τρόπο εξετάζοντας το prop δέντρο μαρκαρίσματος όπου για το prop παίρνουμε τη διάδοση περιορισμών για τους γραμμικούς περιορισμούς πέρα από τα διαστήματα ακέραιων αριθμών, που αναφέρονται στην Παράγραφο 6.4. Παρουσιάσαμε εκεί ότι αυτή η διάδοση περιορισμών μειώνει τις αρχικές περιοχές



Σχήμ. 5.7.Το μειωμένο δέντρο μαρκαρίσματος για το SEND+MORE=MONEY CSP

των ορισμένων μεταβλητών σε

S=9, E $\in$ [4..7], N $\in$ [5..8], D $\in$ [2..8], M $\in$ [2..8], M=1, O=0, R $\in$ [2..8], Y $\in$ [2..8],

όπου X = α είναι η στενογραφία για το X E {α} .

Αυτό σημαίνει ότι αυτή η ακολουθία αναπαράστασης των περιοχών είναι ο άμεσος απόγονος της αρχικής ακολουθίας αναπαράστασης των περιοχών. Τώρα είναι εύκολο να κατασκευαστεί το πραγματικό δέντρο. Απεικονίζεται στο Σχήμα 5.9. Αυτό το prop δέντρο μαρκαρίσματος έχει ακριβώς τέσσερα φύλλα. Το μαρκάρισμα κάθε μεταβλητής εκτός από το E εφαρμόζεται σε ένα ορισμένο μονοσύνολο. Συνεπώς, κάθε κόμβος σε μονό επίπεδο διαφορετικό του 3 δεν έχει κανέναν ή ακριβώς έναν άμεσο απόγονο.

5.4 Περιπτώσεις *prop* δέντρων μαρκαρίσματος

Σε αυτό το τμήμα αντικαθιστούμε τον ορισμό ενός prop δέντρου μαρκαρίσματος από διαδοχικά ισχυρότερες μορφές διάδοσης περιορισμών prop(i) που χρησιμοποιούνται συνήθως στα πλαίσια της αναζήτησης. Για κάθε συγκεκριμένη μορφή της διάδοσης περιορισμών prop(i) πρέπει να διευκρινίσουμε ακριβώς πώς ορίζονται για κάθε χωρίς-φύλλο κόμβο

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, x_n \in E_n$$

οι νέες περιοχές $E'_{i+1}, \ldots, E'_n$ στο επίπεδο 2i.

5.4.1 Forward cheking

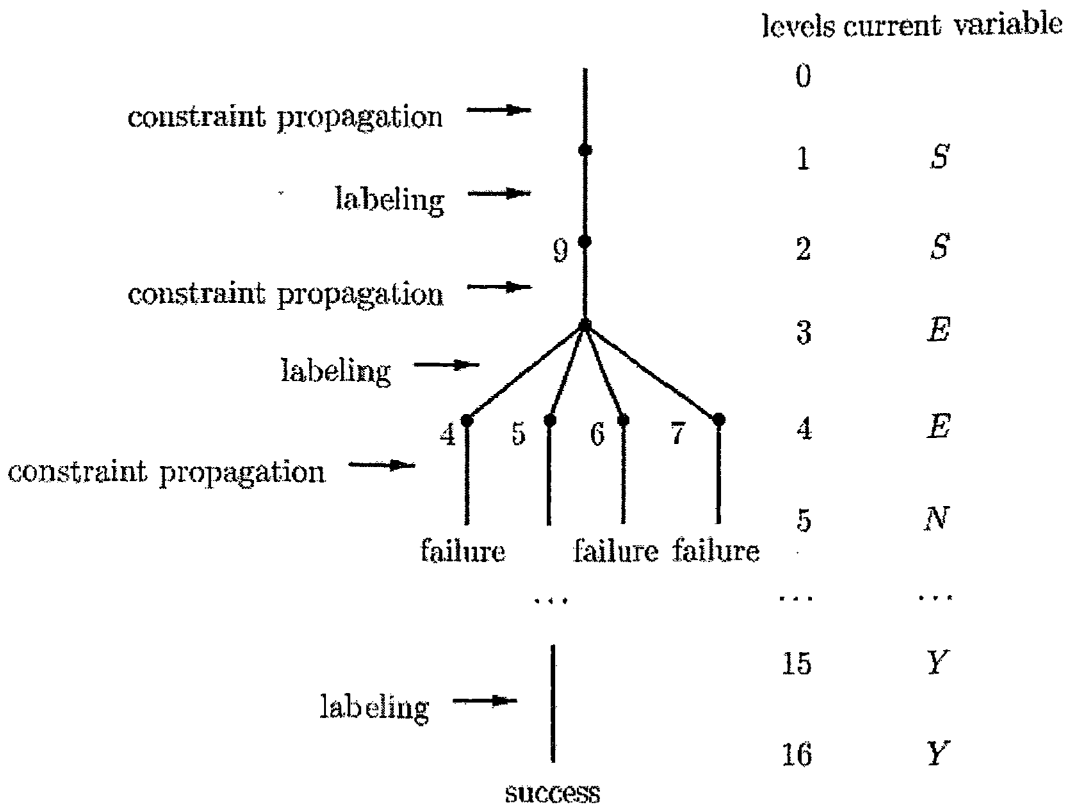
Για να διευκρινιστεί αυτή η μέθοδος διάδοσης περιορισμών θεωρήστε το Πρόβλημα των n Βασιλισσών που συζητήθηκε στο Παράδειγμα 2.2 του Κεφαλαίου 2 για n = 5. Υποθέστε τη διαμόρφωσή του υπό μορφή τυποποιημένου CSP, ώστε είναι να ένα CSP στο οποίο για κάθε ζευγάρι μεταβλητών x, y να υπάρχει ένας μοναδικός περιορισμός στα x, y. Έτσι αυτό το CSP είναι της μορφής

$$\langle C \; ; \; x_1 \in [1..5], \ldots, x_5 \in [1..5] \rangle,$$

όπου για κάθε ζευγάρι μεταβλητών x_i, x_j με $i \neq j$ ο μοναδικός περιορισμός στα x_i, x_j είναι η σύνδεση των τριών ακόλουθων περιορισμών:

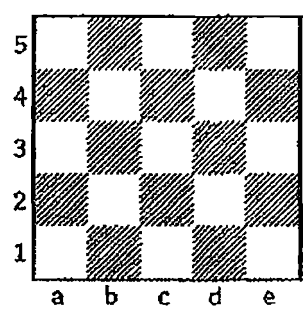
- $x_i \neq x_j$,
- $x_i - x_j \neq i - j$,
- $x_i - x_j \neq j - i$,

που συζητηθήκαν στο Παράδειγμα 2.2.



Σχήμα 5.9. Το prop δέντρο μαρκαρίσματος για το SEND + MORE = MONEY CSP

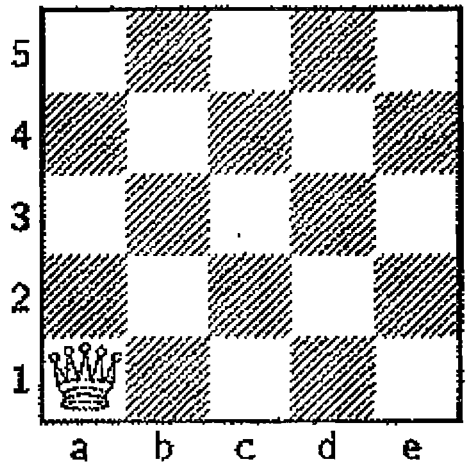
Παρουσιάζουμε ένα τέτοιο CSP ως ένα κενό πίνακα σκακιού διαστάσεων 5 x 5 που απεικονίζεται στο Σχήμα 5.10. Θυμηθείτε ότι στην αντιπροσώπευση αυτού του προβλήματος οι βασίλισσες υποτίθεται ότι τοποθετούνται σε διαφορετικές στήλες, επομένως εδώ στις a - e. Δηλαδή, κάθε στήλη αντιστοιχεί σε μια περιοχή μεταβλητής. Παραδείγματος χάριν η στήλη d αντιστοιχεί στην περιοχή [1..5] της μεταβλητής x_4. Το Σχήμα 5.10 ακολούθως ισοδυναμεί με τη ρίζα ενός prop δέντρου μαρκαρίσματος που συνδέεται με αυτό το CSP.



Σχήμα 5.10. Ο κενός πίνακας σκακιού διαστάσεων 5 x 5

Υποθέστε τώρα ότι τοποθετούμε την πρώτη βασίλισσα στη περιοχή a_1. Την απεικονίζουμε στο Σχήμα 5.11. Αυτός το σχήμα ισοδυναμεί με τον ακόλουθο άμεσο απόγονο της ρίζας στο prop δέντρο μαρκαρίσματος που αναφέρθηκε:

$$x_1 \in \{1\}, x_2 \in [1..5], x_3 \in [1..5], x_4 \in [1..5], x_5 \in [1..5].$$

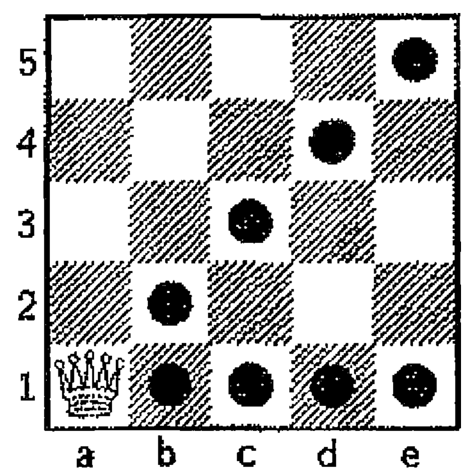


Σχήμα 5.11. Η πρώτη βασίλισσα που τοποθετήθηκε στο a_1

Κατόπιν σημαδεύουμε στις άλλες στήλες τις περιοχές που βρίσκονται υπό επίθεση. Αυτό οδηγεί σε μια κατάσταση που απεικονίζεται στο Σχήμα 5.12. Αυτή η κατάσταση αντιστοιχεί στην αφαίρεση των σημαδεμένων στοιχείων από τις περιοχές των μεταβλητών που αντιπροσωπεύουν τις άλλες βασίλισσες. Σημειώστε ότι κατ' αυτό τον τρόπο καμία λύση δεν χάνεται. Το Σχήμα 5.12 ισοδυναμεί με τον ακόλουθο μοναδικό άμεσο απόγονο του προηγούμενου κόμβου στο prop δέντρο μαρκαρίσματος που συζητήθηκε:

$$x_1 \in \{1\}, x_2 \in \{3,4,5\}, x_3 \in \{2,4,5\}, x_4 \in \{2,3,5\}, x_5 \in \{2,3,4\}.$$

Γενικότερα, εξετάζουμε τις στήλες σε αλφαβητική σειρά και αφού τοποθετήσουμε μια βασίλισσα σε μια περιοχή σημαδεύουμε στις στήλες που εξετάζουν ακόμη τις περιοχές που βρίσκονται υπό επίθεση. Αυτή η διαδικασία μπορεί να τυποποιηθεί ως μορφή



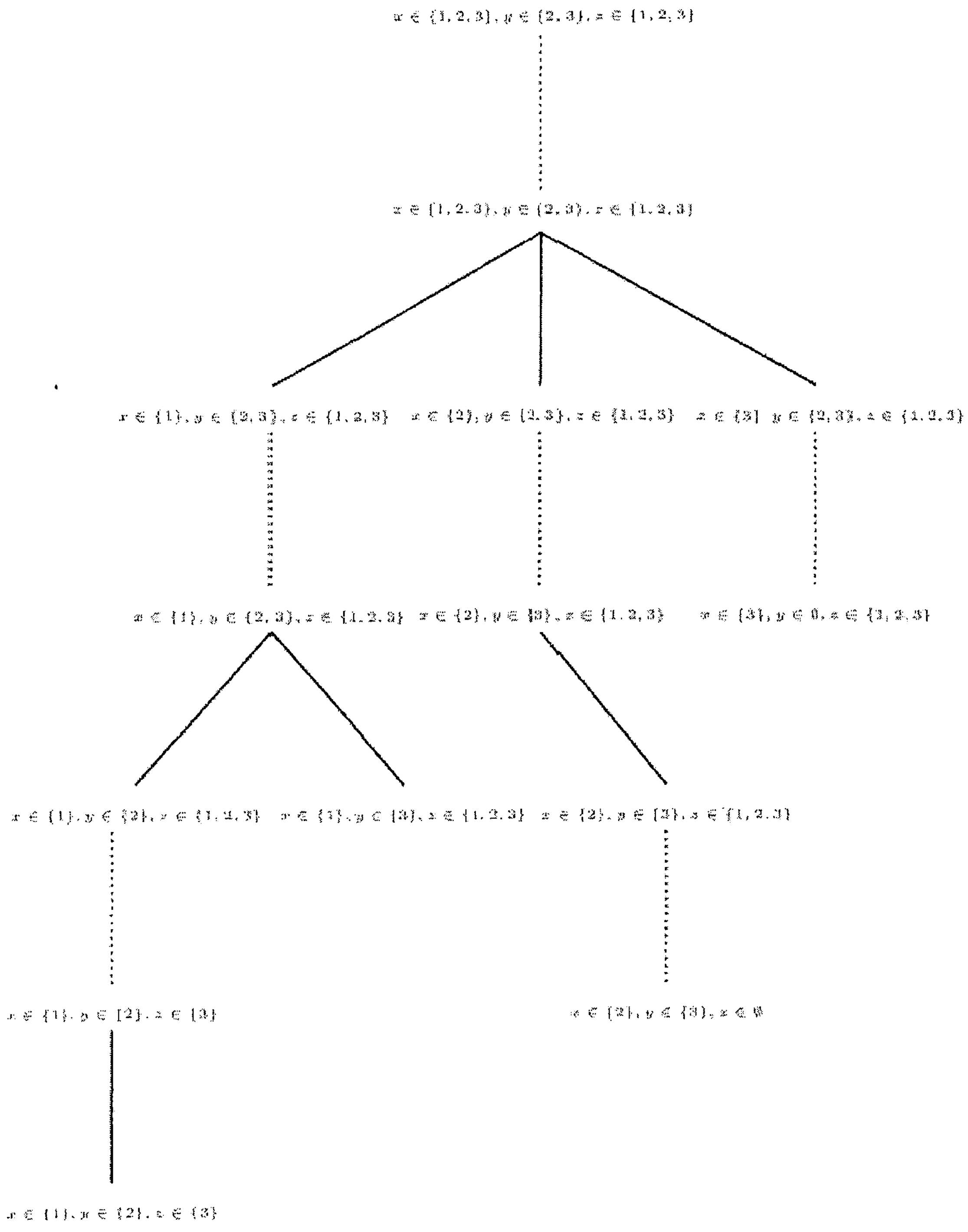
Σχέδιο 5.12. Το αποτέλεσμα του μπροστινού ελέγχου

διάδοσης περιορισμών αποκαλούμενη ***forward checking***(*μπροστινός*

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, x_n \in E_n$$

σε επίπεδο 2i οι περιοχές $E'_{i+1}, \ldots, E'_n$ του μοναδικού άμεσου απογόνου του ορίζονται τοποθετώντας για j Ε [i + 1..n]

$$E'_j := \{e \in E_j \mid \{(x_1, d_1), \ldots, (x_i, d_i), (x_j, e)\} \text{ is consistent}\}. \qquad (5.1)$$



Σχήμα 5.13 Το *FORWARD CHEKING* δέντρο αναζήτησης για το CSP του Παραδείγματος 5.4

Σχήμα 5.13 Το *FORWARD CHEKING* δέντρο αναζήτησης για το CSP του Παραδείγματος 5.4

Έτσι στο *FORWARD CHEKING* δέντρο αναζήτησης, δεδομένης μιας τρέχουσας μεταβλητής x_i αναθεωρείται κάθε περιοχή μιας μελλοντικής μεταβλητής x_j λαμβάνοντας υπόψη τους περιορισμούς στις υποακολουθίες $x_1,\ldots,x_i,x_j$. Για παράδειγμα, στο Σχήμα 5.13 απεικονίζουμε το *FORWARD CHEKING* δέντρο αναζήτησης για το CSP

$$\langle x < y, y < z \; ; \; x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\} \rangle$$

που παρουσιάστηκε στο Παράδειγμα 5.4.

Στο *FORWARD CHEKING* δέντρο αναζήτησης για κάθε μελλοντική μεταβλητή x_j εξετάζονται μόνο οι περιορισμοί που αφορούν το x_j με τις προηγούμενες ή τις τρέχουσες μεταβλητές. Στη συνέχεια λαμβάνοντας υπόψη επίσης τους περιορισμούς στις υποακολουθίες της ακολουθίας $x_{i+1},\ldots,x_n$ των μελλοντικών μεταβλητών παίρνουμε τις ισχυρότερες μορφές διάδοσης περιορισμών. Στις επόμενες δύο υποενότητες ορίζουμε δύο τέτοιους τύπους διάδοσης περιορισμών.

5.4.2 Partial look ahead

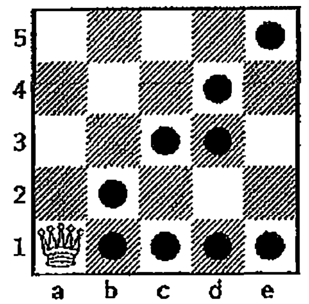
Όπως στην Υποενότητα 5.4.1 έτσι κι εδώ επεξηγούμε τον τρόπο προσέγγισης με τη βοήθεια του προβλήματος των 5 Βασιλισσών. Ας επανεξετάσουμε την κατάσταση που απεικονίζεται στο Σχήμα 5.12, συνεπώς μετά

- στη στήλη a τοποθετείται η βασίλισσα στη περιοχή a_1,
- στις στήλες b-d σημαδεύονται οι περιοχές που βρίσκονται υπό επίθεση

Τώρα στις στήλες b-d σημαδεύουμε κατ' επανάληψη οποιαδήποτε περιοχή με την ακόλουθη ιδιότητα:

εάν μια βασίλισσα τοποθετηθεί σε αυτήν την περιοχή, τότε σε κάποια πιό πρόσφατη στήλη όλες οι ασημάδευτες περιοχές βρίσκονται υπό επίθεση.

Το σημάδεμα κάθε τέτοιας περιοχής αντιστοιχεί στην αφαίρεση του αντίστοιχου στοιχείου από την περιοχή της μεταβλητής που συνδέεται με αυτήν την στήλη. Σημειώστε ότι καμία λύση δεν χάνεται με αυτόν τον τρόπο. Πράγματι, καμία βασίλισσα δεν μπορεί να τοποθετηθεί σε μια σημαδεμένη περιοχή, μιας και διαφορετικά καμία βασίλισσα δεν θα μπορούσε να τοποθετηθεί στην πιό πρόσφατη, "επιθετική", στήλη. Αυτό οδηγεί στην κατάσταση που απεικονίζεται στο Σχήμα 5.14.



Σχήμα 5.14. Το αποτέλεσμα της parial look ahead

Η διαφορά μεταξύ αυτού του σχήματος και του Σχήματος 5.12 είναι ότι τώρα η περιοχή d3 είναι σημαδεμένη. Ο λόγος είναι ότι εάν μια βασίλισσα τοποθετηθεί στην d3, κατόπιν όλες οι ασημάδευτες περιοχές στη στήλη e βρίσκονται υπό επίθεση. Αφότου σημαδευτεί η d3, καμία άλλη περιοχή δεν πρέπει να είναι σημαδεμένη. Πράγματι, αυτήν τη στιγμή η ακόλουθη ιδιότητα ισχύει για τις στήλες b-d:

εάν μια βασίλισσα τοποθετηθεί σε μια ασημάδευτη περιοχή, τότε σε οποιαδήποτε πιό πρόσφατη στήλη κάποια ασημάδευτη περιοχή δεν βρίσκεται υπό επίθεση.

Παραδείγματος χάριν, εάν μια βασίλισσα τοποθετηθεί στην ασημάδευτη περιοχή c4, τότε στη στήλη d η περιοχή d2 δεν βρίσκεται υπό επίθεση και στη στήλη e η περιοχή e3 δεν βρίσκεται υπό επίθεση επίσης. Με άλλα λόγια, εάν μια βασίλισσα τοποθετηθεί στη περιοχή c4, τότε στη στήλη d μπορεί να τοποθετηθεί μια βασίλισσα στη d2 και στη στήλη e μπορεί να τοποθετηθεί μια βασίλισσα στην e3.

Αυτή η διαδικασία μπορεί να τυποποιηθεί επιβάλλοντας τη κατευθυντική τοξοειδή σταθερότητα στο CSP ως αποτέλεσμα της εφαρμογής του μπροστινού ελέγχου. Ολόκληρη η διαδικασία μπορεί να οριστεί ως μια άλλη μορφή διάδοσης περιορισμών, αποκαλούμενη ***partial look ahead*** (*μερική ματιά στο μέλλον)*. Τα αντίστοιχα δέντρα, που ονομάζονται ***PARTIAL LOOK AHEAD*** δέντρα αναζήτησης, ορίζονται ως εξής.

Ορισμός 5.12 Θεωρήστε ένα CSP P := ⟨C ; x_1 E D_1,...,x_n E Dn⟩. Από ένα ***PARTIAL LOOK AHEAD*** δέντρο αναζήτησης σχεδιάζουμε ένα prop δέντρο μαρκαρίσματος που συνδέεται με το P, όπου για κάθε χωρίς-φύλλο κόμβο

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, x_n \in E_n$$

σε επίπεδο 2i οι περιοχές E'_{i+1},...,E'_n του μοναδικού άμεσου απογόνου του ορίζονται αρχικά με τη βοήθεια του μπροστινού ελέγχου, δηλ., χρησιμοποιώντας την (5.1) για j E [i + 1..n]. Εάν όλες τους είναι μη-άδειες, θεωρούμε το CSP

$$\mathcal{P}_i := \langle C' \ ; \ x_{i+1} \in E'_{i+1}, \ldots, x_n \in E'_n \rangle, \qquad (5.2)$$

όπου οι περιορισμοί C΄ ορίζονται μεμονωμένα από τους περιορισμούς του P στις υποακολουθίες x_{i+1},...,x_n και στις περιοχές E'_{i+1},...,E'_n, και επαναπροσδιορίζουν αυτές τις περιοχές με την επιβολή στο P_i της κατευθυντικής τοξοειδής σταθερότητας w.r.t. η σειρά x_{i+1}<...<x_n με τη βοήθεια οποιουδήποτε κατευθυντικής τοξοειδής σταθερότητας αλγόριθμου.

Για παράδειγμα, απεικονίζουμε στο Σχήμα 5.15 το *PARTIAL LOOK AHEAD* δέντρο αναζήτησης για το CSP

$$\langle x < y, y < z \ ; \ x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\} \rangle$$

από το παράδειγμα 5.4. Έτσι αυτό το δέντρο είναι μικρότερο από το αντίστοιχο FORWARD CHECKING δέντρο αναζήτησης που απεικονίζεται στο Σχήμα 5.13. Αυτό οφείλεται στο γεγονός ότι τώρα η διάδοση περιορισμών είναι ισχυρότερη.

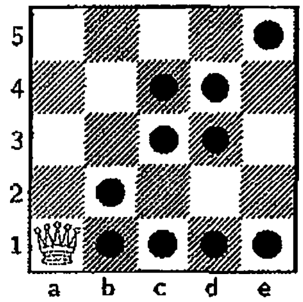
5.4.3 Διατήρηση τοξοειδούς σταθερότητας (MAC)

Μία ακόμα ισχυρότερη μορφή διάδοσης περιορισμών από αυτή που χρησιμοποιείται στα PARTIAL LOOK AHEAD δέντρα αναζήτησης, ισχύει επιβάλλοντας τη τοξοειδή σταθερότητα στις μελλοντικές μεταβλητές αν και ορίζεται μόνο για τους δυαδικούς περιορισμούς. Αυτό οδηγεί στην έννοια που ονομάζεται ***MAC*** μία σύντμηση για την ***Maintaining Arc Consistency***(διατήρηση τοξοειδούς σταθερότητας). Η MAC καλείται εναλλακτικά ***ARC CONSISTENCY LOOK AHEAD***(σταθερή ματιά στο μέλλον) ή ***FULL LOOK AHEAD***(πλήρη ματιά στο μέλλον).

Για να επεξηγήσουμε την έμπνευση πίσω από αυτήν τη μέθοδο ας επιστρέψουμε στο παράδειγμα μας, το Πρόβλημα των 5 Βασιλισσών. Υποθέστε ότι ενισχύουμε τη διαδικασία που αναφέρεται στην Υποενότητα 5.4.2 στην ακόλουθη. Αφού τοποθετήσουμε την πρώτη βασίλισσα στην περιοχή a_1 και μετατοπίσουμε τις περιοχές που βρίσκονται υπό επίθεση στις άλλες στήλες εκτελούμε την ακόλουθη ενέργεια. Σημαδεύουμε κατ' επανάληψη οποιαδήποτε περιοχή στις στήλες b-d με την ακόλουθη ιδιότητα:

εάν μια βασίλισσα τοποθετηθεί σε αυτήν την περιοχή, τότε για κάποια άλλη στήλη από b-d όλες οι ασημάδευτες περιοχές βρίσκονται υπό επίθεση.

Η διαφορά μεταξύ αυτής της διαδικασίας και αυτής που περιγράφεται στην προηγούμενη υποενότητα είναι ότι τώρα σταματάμε την περιοριστική τροποποίηση "πιο πρόσφατα" κατά την επιστροφή στις άλλες στήλες. Αυτό οδηγεί στην κατάσταση που απεικονίζεται στο Σχήμα 5.16.



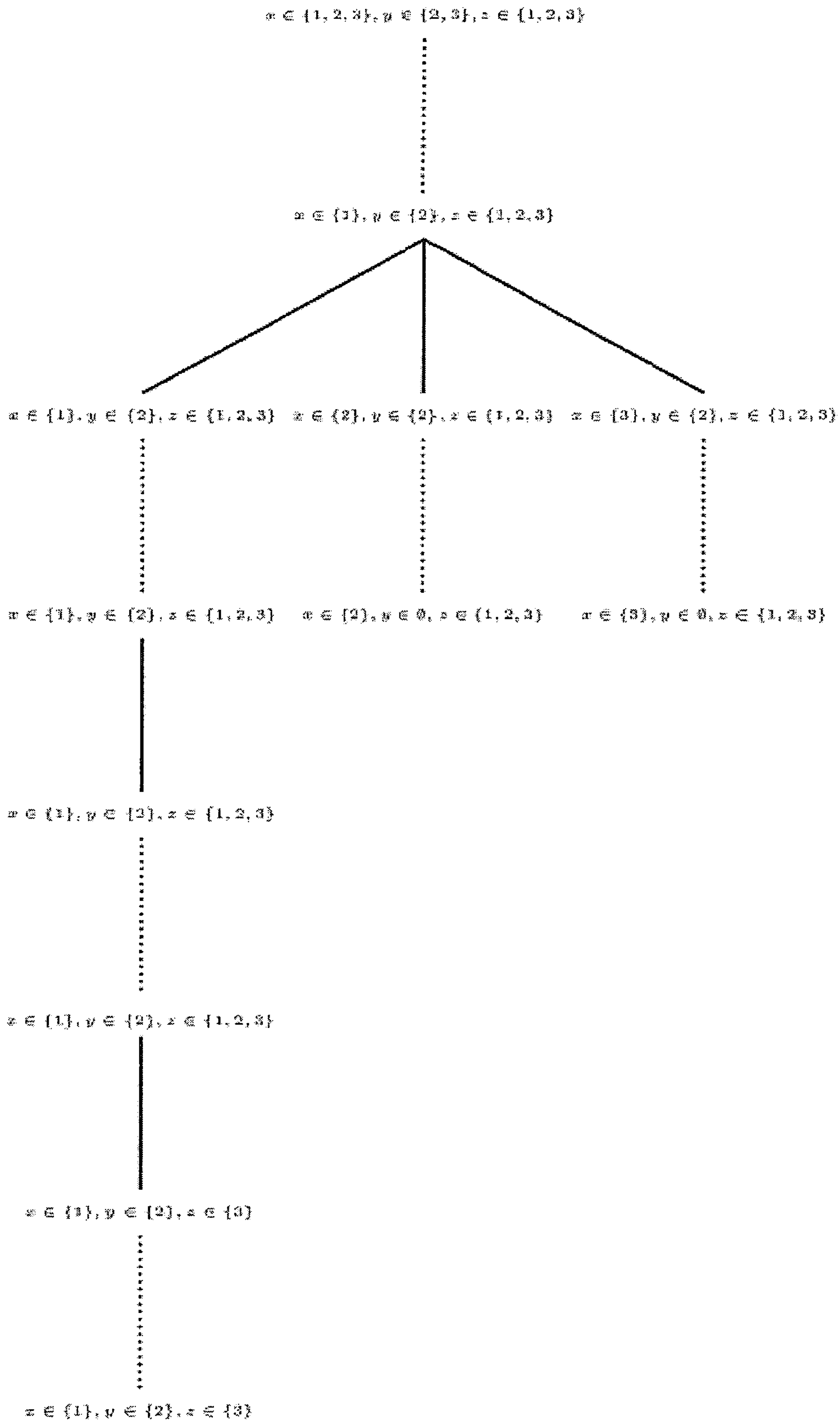
Σχήμα 5.16. Το αποτέλεσμα της διατήρησης τοξοειδής σταθερότητας (MAC)

Η διαφορά μεταξύ αυτού του σχήματος και του Σχήματος 5.14 είναι ότι η περιοχή c4 είναι επίσης σημαδεμένη τώρα. Ο λόγος είναι ότι εάν μια βασίλισσα τοποθετηθεί στην c4, έπειτα όλες οι ασημάδευτες περιοχές στη στήλη b βρίσκονται υπό επίθεση. Στην Υποενότητα 5.4.2 εξετάστηκε μόνο η επίδραση της τοποθέτησης μιας βασίλισσας στις πιό πρόσφατες στήλες κι έτσι δεν αναλύθηκαν στη στήλη b οι συνέπειες της τοποθέτησης μιας βασίλισσας στη στήλη c

Αφού σημαδευτούν οι c4 και d3, καμία άλλη περιοχή δεν πρέπει να σημαδευτεί. Πράγματι, αυτήν τη στιγμή ισχύουν τα εξής για τις στήλες b-d:

εάν μια βασίλισσα τοποθετηθεί σε μια ασημάδευτη περιοχή, τότε σε οποιαδήποτε άλλη στήλη από b-d κάποιες ασημάδευτες περιοχές δεν βρίσκονται υπό επίθεση.

Αυτή η διαδικασία μπορεί να τυποποιηθεί με την επιβολή της τοξοειδούς σταθερότητας στο CSP ως αποτέλεσμα της εφαρμογής του μπροστινού ελέγχου. Τα αντίστοιχα δέντρα, αποκαλούμενα **MAC** δέντρα αναζήτησης, ορίζονται ανάλογα ως *PARTIAL LOOK AHEAD* δέντρα αναζήτησης.



Σχήμα 5.15. Το PARTIAL LOOK AHEAD δέντρο αναζήτησης για το CSP του

Παραδείγματος 5.4

Ορισμός 5.13 Θεωρήστε ένα CSP P := ⟨C ; x_1 E D_1,...,x_n E Dn⟩. Από ένα MAC δέντρο αναζήτησης σχεδιάζουμε ένα prop δέντρο μαρκαρίσματος που συνδέεται με το P, όπου για κάθε χωρίς-φύλλο κόμβο

$$x_1 \in \{d_1\}, \ldots, x_i \in \{d_i\}, x_{i+1} \in E_{i+1}, x_n \in E_n$$

σε επίπεδο 2i οι περιοχές E'_{i+1},...,E'_n του μοναδικού άμεσου απογόνου του είναι αρχικά ορισμένες με τη βοήθεια του μπροστινού ελέγχου, δηλ., χρησιμοποιώντας την (5.1) για j E [i + 1...n] . Εάν όλες τους είναι μη-άδειες, θεωρούμε το CSP

$$\mathcal{P}_i = \langle C' \ ; \ x_{i+1} \in E'_{i+1}, \ldots, x_n \in E'_n \rangle$$

ορισμένο στην (5.2) και μοναδικά ορισμένο από το P και τις περιοχές E'_{i+1},...,E'_n και επαναπροσδιορίζουμε τις περιοχές E'_{i+1},...,E'_n με την επιβολή στο P_i της τοξοειδής σταθερότητας, με τη βοήθεια οποιουδήποτε τοξοειδής σταθερότητας αλγορίθμου.

Για παράδειγμα, ας επιστρέψουμε στο CSP

$$\langle x < y, y < z \ ; \ x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\} \rangle$$

από το Παράδειγμα 5.4. Το MAC δέντρο αναζήτησης για αυτό το CSP μειώνεται σε ένα ενιαίο μονοπάτι που αποτελείται από επτά κόμβους. Απεικονίζεται στο Σχήμα 5.17. Έτσι σε αυτήν την περίπτωση η αρχική εφαρμογή της διάδοσης περιορισμών οδήγησε σε μια μείωση όλων των περιοχών στα ορισμένα μονοσύνολα.

$$x \in \{1,2,3\}, y \in \{2,3\}, z \in \{1,2,3\}$$

$$x \in \{1\}, y \in \{2\}, z \in \{3\}$$

$$\cdots$$

$$x \in \{1\}, y \in \{2\}, z \in \{3\}$$

Σχήμα 5.17. Το MAC δέντρο αναζήτησης για το CSP του Παραδείγματος 5.4

5.5 Αλγόριθμοι αναζήτησης για τα δέντρα μαρκαρίσματος

Τώρα που έχουμε ορίσει τα δέντρα αναζήτησης και τους πιό κοινούς τους τύπους για την περίπτωση του μαρκαρίσματος ας γυρίσουμε στους αλγόριθμους αναζήτησης. Στόχος αυτών των αλγόριθμων είναι να βρεθεί μια λύση ή όλες οι λύσεις σε ένα CSP, ή να αναφερθεί ότι καμία τέτοια λύση δεν υπάρχει. Στη περίπτωση των περιορισμένων προβλημάτων βελτιστοποίησης αυτοί οι αλγόριθμοι ψάχνουν για μια βέλτιστη ή όλες τις βέλτιστες λύσεις. Στη συνέχεια διατυπώνουμε πολλούς αλγόριθμους αναζήτησης για τα δέντρα μαρκαρίσματος. Η στρατηγική που χρησιμοποιείται από τους δύο πρώτους αλγόριθμους είναι σωστή μόνο για συγκεκριμένα πεπερασμένα CSPs, ενώ οι άλλες είναι σωστές για όλα τα πεπερασμένα CSPs. Εννοιολογικά, κάθε ένας από αυτούς τους αλγόριθμους ψάχνει μια λύση πραγματοποιώντας μια πρώτης-δύναμης αναζήτηση σε έναν κατάλληλο τύπο δέντρου μαρκαρίσματος που ορίζεται στα δύο προηγούμενα τμήματα.

Σε όλους τους αλγόριθμους υποθέτουμε για λόγους ευθύτητας ότι η γραμμική σειρά < στις μεταβλητές συμπίπτει με τη σειρά που χρησιμοποιείται στον ορισμό του εξεταζόμενου CSP P := $\langle C \; ; \; x_1 \in D_1, \ldots, x_n \in Dn \rangle$. Επίσης υποθέτουμε δύο περαιτέρω αναλυτικούς τύπους **elements** και **domain** και χρησιμοποιούμε τον πίνακα D τύπου **ARRAY[1..n] OF domain** για να αντιπροσωπεύσουμε την ακολουθία των περιοχών $D_1, \ldots, Dn$. Μια λύση, εάν υπάρχει, παράγεται στον συνολικό πίνακα inst τύπου **ARRAY[1...n] OF elements**. Τέλος, συντέμνουμε την αναφορά

"το στιγμιαίο $\{(x_1, inst[1]), \ldots, (x_{j-1}, inst[j-1]), (x_j, d)\}$ είναι σταθερό"

σε **cons (inst, j, d)**.

Κατά την διάρκεια της συζήτησης των αλγόριθμων υποθέτουμε ότι τα στοιχεία σε κάθε αριθμητική περιοχή επιλέγονται με αυξανόμενη σειρά. Αυτό θα μας επιτρέψει να συσχετίσουμε αυτούς τους αλγόριθμους με τα δέντρα μαρκαρίσματος που εξετάστηκαν. Αυτοί οι αλγόριθμοι γράφονται χρησιμοποιώντας τη σύνταξη MODULA-2 που πρέπει να είναι αυτεξήγητη. Ας υπενθυμίσουμε ότι στην σύνταξη MODULA-2 υπάρχουν δύο τύποι παραμέτρων: η συνηθισμένη δήλωση-με-τιμή παράμετρος, και η δήλωση-με-μεταβλητή παράμετρος, που διακρίνεται από το πρόθεμα **VAR**. Οι τωρινές δήλωση-με-τιμή παράμετροι μπορούν να είναι αυθαίρετες εκφράσεις του δεδομένου τύπου, ενώ οι τωρινές δήλωση-με-μεταβλητή παράμετροι μπορούν να είναι μόνο μεταβλητές του δεδομένου τύπου. Οι πρώτες δεν μπορούν να αλλάξουν, ενώ οι τελευταίες μπορούν. Η δήλωση-με-τιμή παράμετρος διαμορφώνεται στο C χρησιμοποιώντας ως τωρινή παράμετρο έναν δείκτη στην εξεταζόμενη μεταβλητή.

5.5.1 Backtrack-free αναζήτηση

Ξεκινάμε με τον απλούστερο αλγόριθμο, αποκαλούμενο BACKTRACK-FREE(ελεύθερη-αντίστροφη διαδρομή) αλγόριθμο αναζήτησης. Προσπαθεί να κατασκευάσει μια λύση ξεκινώντας από το κενό στιγμιαίο και διαδοχικά προσπαθεί να το επεκτείνει σε ένα σταθερό στιγμιαίο που ορίζεται στην επόμενη μεταβλητή στην υποτιθέμενη γραμμική σειρά. Εάν συνεχίζοντας με αυτόν τον τρόπο γίνει στιγμιαία η τελική μεταβλητή, έχει βρεθεί μια λύση. Διαφορετικά αναφέρεται μια αποτυχία. Ο αλγόριθμος παρουσιάζεται στο Σχήμα 5.18. Διατυπώνεται κατά τέτοιο τρόπο ώστε να είναι εύκολο να τροποποιηθεί για να εξασφαλίσει άλλους αλγορίθμους αναζήτησης.

Ο BACKTRACK-FREE αλγόριθμος αναζήτησης εκτελεί την αναζήτηση στα μειωμένα δέντρα μαρκαρίσματος. Συγκεκριμένα, ισχύει το ακόλουθο συμπέρασμα.

Θεώρημα 5.14 (Backtrack-free Search) *Ο BACKTRACK-FREE αλγόριθμος αναζήτησης κατασκευάζει σε ένα αρχικό τμήμα του πίνακα* **inst** *την αριστερότερη διακλάδωση του αντίστοιχου μειωμένου δέντρου μαρκαρίσματος και θέτει τη μεταβλητή* **success** *ως* **ΑΛΗΘΗ** *εάν αυτή η διακλάδωση αντιπροσωπεύει μια λύση.*

```
MODULE backtrack_free;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
PROCEDURE backtrack_free(j: INTEGER; D: domains;
                              VAR success: BOOLEAN);
BEGIN
  WHILE D[j] <> {} AND NOT success DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      success := (j = n);
      IF NOT success THEN
        j := j+1
      END
    END
  END
END backtrack_free;
BEGIN
  success := FALSE;
  backtrack_free(1,D,success)
END backtrack_free;
```

Σχήμα 5.18. Ο BACKTRACK-FREE αλγόριθμος αναζήτησης

Οι αποδείξεις και τα ανάλογα επόμενα αποτελέσματα προέρχονται από την εισαγωγή στον αριθμό των μεταβλητών. Οι περισσότερες από αυτές τις αποδείξεις είναι απλές. Δίνονται ως Ασκήσεις.

Στην περίπτωση του δέντρου μαρκαρίσματος του Σχήματος 5.4 ο BACKTRACK-FREE αλγόριθμος αναζήτησης κατευθύνεται κατά μήκος της αριστερότερης διακλάδωσης του δέντρου και κατασκευάζει στο **inst** μια λύση που αντιστοιχεί στο σταθερό στιγμιαίο {(x, 1), (y, 2), (z, 3)}. Σημειώστε ότι αυτή η διακλάδωση δεν είναι η αριστερότερη διακλάδωση στο αντίστοιχο πλήρες δέντρο μαρκαρίσματος του Σχήματος 5.2. Στη συνέχεια, στην περίπτωση του δέντρου μαρκαρίσματος του Σχήματος 5.5 ο BACKTRACK-FREE αλγόριθμος παράγει ένα αρχικό τμήμα του inst που αντιστοιχεί στο στιγμιαίο {(x, 1), (z, 1)} και γι' αυτό το λόγο αποτυγχάνει να βρεί μια λύση.

Έτσι γενικά ο BACKTRACK-FREE αλγόριθμος αναζήτησης δεν βρίσκει λύση εάν υπάρχει κάποια. Αφ' ετέρου αυτός ο αλγόριθμος είναι αποδοτικός —τρέχει σε χρόνο $O(\Pi^n_{i=1}|D_i|)$, όπου $|D_i|$ δείχνει τον αριθμό στοιχείων συνόλου της περιοχής D_i. Έτσι είναι ενδιαφέρον να βρεθούν όροι που εγγυώνται ότι ο BACKTRACK-FREE αλγόριθμος αναζήτησης βρίσκει μια λύση εάν υπάρχει. Το ακόλουθο θεώρημα είναι ένα παράδειγμα αυτού του συμπεράσματος.

Θεώρημα 5.15 (Backtrack-free Αναζήτηση II) *Θεωρήστε ένα CSP που είναι ισχυρά (K + 1) σταθερό και η γραφική του παράσταση έχει πλάτος K. Κατόπιν για κάποια διατεταγμένη μεταβλητή ο* BACKTRACK-FREE *αλγόριθμος αναζήτησης κατασκευάζει μια λύση εάν υπάρχει ειδάλλως δηλώνει μια αποτυχία θέτοντας την μεταβλητή* **success** *ως* **ΨΕΥΔΗ**.

Απόδειξη Εάν κάποια περιοχή είναι κενή, σ' αυτή τη περίπτωση δεν υπάρχει καμία λύση και ο BACKTRACK-FREE αλγόριθμος αναζήτησης ολοκληρώνεται θέτοντας την μεταβλητή **success** ως **ΨΕΥΔΗ.** Ειδάλλως η κατασκευή χρησιμοποιείται στην απόδειξη του Θεωρήματος Σταθερότητα 2 που συμπίπτει με μια επιτυχή εκτέλεση του BACKTRACK-FREE αλγόριθμου αναζήτησης.

5.5.2 Backtrack-free αναζήτηση με διάδοση περιορισμών

Είναι εύκολο να τροποποιηθεί ο BACKTRACK-FREE αλγόριθμος αναζήτησης έτσι ώστε μια διάδοση περιορισμών υπό μορφή μείωσης περιοχών να ενσωματώνεται σε αυτόν. Αντιπροσωπεύουμε τη διάδοση περιορισμών συμβολικά με τη βοήθεια της διαδικασίας prop με την ένδειξη

```
PROCEDURE prop(j: INTEGER; VAR D: domains;
               VAR failure: BOOLEAN);
```

Δεδομένου ενός πίνακα D τύπου **ARRAY[1..n] OF domain** που αντιπροσωπεύει την ακολουθία περιοχών των μεταβλητών και του j Ε [1..n], η διαδικασία κλήσης **prop (j, D,failure)** τροποποιεί τις περιοχές D [j+1], ..., D[n] και θέτει τη μεταβλητή **failure** ως **ΑΛΗΘΗ** (ανεξάρτητα από την αρχική τιμή της) εάν μια από αυτές τις περιοχές γίνει κενή. Τροποποιούμε έπειτα τη διαδικασία **backtrack_free** στη διαδικασία **backtrack_free_prop**.Ο αλγόριθμος αναζήτησης που προκύπτει παρουσιάζεται στο Σχήμα 5.19.

Η διαφορά είναι στην παρουσία των κλήσεων **prop (0, D, failure)** και **prop (j, D,failure)** και στη χρήση του τεστ **NOT FAILURE** πριν τη κλήση **backtrack_free_prop (1, D, success)** και στην ανάθεση **j := j+1**. Ο BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμος αναζήτησης εκτελεί την αναζήτηση στα prop δέντρα μαρκαρίσματος.

Σε αντίθεση με τον BACKTRACK-FREE αλγόριθμο αναζήτησης ο BACKTRACK-FREE WITH CONSTRAINT PROPAGATION(ελεύθερη-αντίστροφη διαδρομή με διάδοση περιορισμών) αλγόριθμος αναζήτησης εκτελεί την αναζήτηση σε ένα πιο περίπλοκο δέντρο. Συνεπώς, είναι πιο δύσκολο να περιγράψουμε το αποτέλεσμα αυτού του αλγόριθμου. Ο λόγος είναι ότι, σε αντίθεση με τα μειωμένα δέντρα, στα prop δέντρα μαρκαρίσματος οι αποτυχημένοι κόμβοι παρουσιάζονται συγκεκριμένα: παράγονται ως αποτέλεσμα της διάδοσης περιορισμών. Έτσι για να ορίσουμε το αποτέλεσμα του BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμου αναζήτησης εκτελούμε την επόμενη ακολουθία διαδικασιών στο αντίστοιχο prop δέντρο μαρκαρίσματος:

- αφαιρούμε όλους τους αποτυχημένους κόμβους.
  αυτούς τους αφαιρούμενους κόμβους δεν θα τους επισκεφτεί ο αλγόριθμος,
- αφαιρούμε από το δέντρο που προκύπτει όλα τα φύλλα L με την ακόλουθη ιδιότητα: υπάρχει ένας αμφιθαλής του L που βρίσκεται στα δεξιά του το οποίο δεν είναι φύλλο;
  στο αρχικό δέντρο αυτούς τους αφαιρούμενους κόμβους θα τους επισκεφτεί ο αλγόριθμος, αλλά δεν θα επιλεχτούν δεδομένου ότι ο μοναδικός απόγονός τους στο αρχικό δέντρο είναι ένας αποτυχημένος κόμβος,
- εντοπίζουμε το αριστερότερο φύλλο στο δέντρο. Το ονομάζουμε τελικό κόμβο.

Μπορούμε τώρα να ορίσουμε το επιθυμητό αποτέλεσμα.

Θεώρημα 5.16 (Backtack-free with Constraint Propagation Αναζήτηση) *Ο BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμος αναζήτησης κατασκευάζει σε ένα αρχικό τμήμα του πίνακα* **inst** *ένα στιγμιαίο που αντιστοιχεί στον τελικό κόμβο του αντίστοιχου prop δέντρου μαρκαρίσματος και θέτει τη μεταβλητή* **success** *ως* **ΑΛΗΘΗ** *εάν αυτός ο κόμβος βρίσκεται σε επίπεδο 2n.*

```
MODULE backtrack_free_prop;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
    failure: BOOLEAN;
PROCEDURE backtrack_free_prop(j: INTEGER; D: domains;
                              VAR success: BOOLEAN);
BEGIN
  WHILE D[j] <> {} AND NOT success DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      success := (j = n);
      IF NOT success THEN
        prop(j,D,failure);
        IF NOT failure THEN
          j := j+1
        END
      END
    END
  END
END backtrack_free_prop;
BEGIN
  success := FALSE;
  prop(0,D,failure);
  IF NOT failure THEN
    backtrack_free_prop(1,D,success)
  END
END backtrack_free_prop;
```

Σχήμα. 5.19. O BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμος αναζήτησης

5.5.3 Backtracking

Οι BACKTRACK-FREE και BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμοι αναζήτησης είναι σωστοί μόνο για μια περιορισμένη κατηγορία πεπερασμένων CSPs. Από εδώ και στο εξής αναφέρουμε τους αλγόριθμους αναζήτησης που είναι σωστοί για όλα τα πεπερασμένα CSPs.

Σημειώστε ότι στον BACKTRACK-FREE αλγόριθμο αναζήτησης οι επιλογές των στοιχείων d για τα οποία ισχύει **cons (inst, j ,d)** δεν μπορούν να ανακληθούν: κάθε φορά που βρίσκεται ένα τέτοιο d καταγράφεται μόνιμα στον πίνακα inst. O BACK-TRACK(αντίστροφη-διαδρομή) αλγόριθμος αναζήτησης είναι μια τροποποίηση αυτής της στρατηγικής. Ανεπίσημα, εάν η αναζήτηση "στο πλαίσιο" των κόμβων (j ,d) δεν πετύχει, επιχειρείται μια άλλη επιλογή του d για το επίπεδο j. Αυτό πραγματοποιείται με την αντικατάσταση της ανάθεσης j := j+1 από μια επαναλαμβανόμενη κλήση του αλγόριθμου που αντιπροσωπεύει αυτήν την αναζήτηση "στα πλαίσια" του κόμβου (j ,d). O αλγόριθμος που προκύπτει παρουσιάζεται στο Σχήμα 5.20.

Σημειώστε ότι στον BACKTRACK αλγόριθμο αναζήτησης ο πίνακας D περνά ως κλήση με τιμή παραμέτρου. Αυτό είναι κρίσιμο για την κατάλληλη λειτουργία του αλγόριθμου: έπειτα σε κάθε επίπεδο επιστροφής τροποποιείται μόνο ένας εναλλακτικός τύπος του D. O BACKTRACK αλγόριθμος αναζήτησης εκτελεί την αναζήτηση στα μειωμένα δέντρα μαρκαρίσματος. Το ακόλουθο συμπέρασμα συνοψίζει τη συμπεριφορά του.

Θεώρημα 5.17 (Backtrack-free with Constraint Propagation Αναζήτηση) *Εάν υπάρχει μια λύση, η* BACKTRACK *αναζήτηση κατασκευάζει στον πίνακα* **inst** *την αριστερότερη διακλάδωση του αντίστοιχου μειωμένου δέντρου μαρκαρίσματος που αντιπροσωπεύει μια λύση. Διαφορετικά δηλώνει μια αποτυχία θέτοντας τη μεταβλητή* **success** *ως* **FALSE**.

```
MODULE backtrack;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
PROCEDURE backtrack(j: INTEGER; D: domains;
                    VAR success: BOOLEAN);
BEGIN
  WHILE D[j] <> {} AND NOT success DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      success := (j = n);
      IF NOT success THEN
        backtrack(j+1,D,success)
      END
```

```
    END
  END
END backtrack;
BEGIN
  success := FALSE;
  backtrack(1,D,success)
END backtrack;
```

Σχήμα 5.20. Ο BACKTRACK αλγόριθμος αναζήτησης

5.5.4 Backtracking με διάδοση περιορισμών

Τροποποιούμε τώρα τον BACKTRACK αλγόριθμο αναζήτησης έτσι ώστε μια διάδοση περιορισμών υπό μορφή μείωσης περιοχών να ενσωματώνεται σε αυτόν. Για αυτόν τον σκοπό προχωράμε ομοίως, όπως στην Υποενότητα 5.5.2 και χρησιμοποιούμε την prop διαδικασία με την ένδειξη

```
PROCEDURE prop(j: INTEGER; VAR D: domains;
               VAR failure: BOOLEAN);
```

Η κατάλληλη τροποποίηση του BACKTRACK αλγόριθμου αναζήτησης παράγει τον αλγόριθμο που παρουσιάζεται στο Σχήμα 5.21.

Έτσι στον αλγόριθμο αναζήτησης χρησιμοποιούμε τη διάδοση περιορισμών που ενσωματώνεται στην διαδικασία κλήσης prop (j, D, failure) η οποία προηγείται και της αρχικής και της επαναλαμβανόμενης κλήσης στην κύρια διαδικασία backtrack_prop. Αυτές οι κλήσεις εισάγονται μόνο εάν η μεταβλητή failure δεν τεθεί ως ΑΛΗΘΗΣ. Η συμπεριφορά αυτού του αλγορίθμου αναζήτησης συνοψίζεται στο ακόλουθο συμπέρασμα ανάλογο του Θεωρήματος 5.17 (με την BACKTRACK Αναζήτηση).

Θεώρημα 5.18 (Backtrack with Constraint Propagation Αναζήτηση) *Εάν υπάρχει μια η λύση, ο* BACKTRACK WITH CONSTRAINT PROPAGATION *αλγόριθμος αναζήτησης, κατασκευάζει στον πίνακα* **inst** *το αριστερότερο φύλλο του αντίστοιχου prop δέντρου μαρκαρίσματος που αντιπροσωπεύει μια λύση. Διαφορετικά δηλώνει μια αποτυχία θέτοντας τη μεταβλητή* **success** *ως* **ΨΕΥΔΗ**.

```
MODULE backtrack_prop;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
    failure: BOOLEAN;
PROCEDURE backtrack_prop(j: INTEGER; D: domains;
                         VAR success: BOOLEAN);
BEGIN
  WHILE D[j] <> {} AND NOT success DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      success := (j = n);
      IF NOT success THEN
        prop(j,D,failure);
        IF NOT failure THEN
          backtrack_prop(j+1,D,success)
        END
      END
    END
  END
END backtrack_prop;
BEGIN
  success := FALSE;
  prop(0,D,failure);
  IF NOT failure THEN
    backtrack_prop(1,D,success)
  END
END backtrack_prop;
```

Σχήμα 5.21. Ο BACKTRACK WITH CONSTRAINT PROPAGATION αλγόριθμος αναζήτησης

5.6 Περιπτώσεις αντίστροφης διαδρομής με διάδοση περιορισμών

Θα χρησιμοποιήσουμε τώρα τον τελευταίο αλγόριθμο αναζήτησης ως πρότυπο για να παράγει τρεις συγκεκριμένους αλγόριθμους αναζήτησης που συσχετίζονται με τα *FORWARD CHEKING, PARTIAL LOOK AHEAD* και *MAC* δέντρα αναζήτησης που ορίζονται στην Παράγραφο 5.4. Ανάλογες ειδικεύσεις του BACKTRACK-FREE WITH CONSTRAINT PROPAGATION αλγόριθμου αναζήτησης βρίσκονται στην διαθεσιμότητα του στον αναγνώστη.

5.6.1 Forward checking

Ο πρώτος, αποκαλούμενος FORWARD CHECKING αλγόριθμος αναζήτησης, είναι μια ειδίκευση του BACKTRACK WITH CONSTRAINT PROPAGATION αλγόριθμου αναζήτησης στον οποίο η διάδοση περιορισμών αποτελείται από τον μπροστινό έλεγχο όπως εξηγήθηκε στην Υποενότητα 5.4.1. Για να τον ορίσουμε ακριβέστερα χρησιμοποιούμε την ακόλουθη βοηθητική διαδικασία **revise** που χρησιμοποιεί τον γενικό πίνακα **inst**:

```
PROCEDURE revise(j,k: INTEGER; VAR D: domains);
BEGIN
  D[k] := {d ∈ D[k] | {(x1,inst[1]),...,(xj,inst[j]),(xk,d)} is
                        a consistent instantiation}
END revise;
```

The prop procedure is now defined as follows.

```
PROCEDURE prop(j: INTEGER; VAR D: domains;
               VAR failure: BOOLEAN);
VAR k: INTEGER;
BEGIN
  failure := FALSE;
  k := j+1;
  WHILE k <> n+1 AND NOT failure DO
    revise(j,k,D);
    failure := (D[k] = {});
    k := k+1
  END
END prop;
```

Έτσι στην περίπτωση που η περιοχή μιας μελλοντικής μεταβλητής x_k γίνει κενή ως αποτέλεσμα της κλήσης revise (j, k, D), η μεταβλητή **failure** τίθεται ως **ΑΛΗΘΗΣ** και ,ως μια βελτιστοποίηση, οι διορθώσεις των επόμενων μελλοντικών μεταβλητών εγκαταλείπονται. Σε αυτή τη περίπτωση η κλήση prop (j, D, failure) στην πραγματικότητα δεν υπολογίζει τις νέες περιοχές όλων των μελλοντικών μεταβλητών όπως ορίζονται στην περίπτωση του *FORWARD CHECKING* δέντρου αναζήτησης, αλλά δεν έχει σημασία δεδομένου ότι σε αυτή τη περίπτωση πραγματοποιείται οπωσδήποτε μια αντίστροφη διαδρομή στον αλγόριθμο αναζήτησης.

Ο FORWARD CHECKING αλγόριθμος αναζήτησης που προκύπτει εκτελεί την *FORWARD CHECKING* αναζήτηση στο δέντρο αναζήτησης. Το ακόλουθο συμπέρασμα συνοψίζει τη συμπεριφορά του.

Θεώρημα 5.19 (Forward Checking Αναζήτηση) *Εάν υπάρχει μια λύση, ο* FORWARD CHECKING *αλγόριθμος αναζήτησης κατασκευάζει στον πίνακα* **inst** *το αριστερότερο φύλλο του αντίστοιχου FORWARD CHECKING δέντρου που αντιπροσωπεύει μια λύση. Διαφορετικά δηλώνει μια αποτυχία θέτοντας τη μεταβλητή* **success** *ως* **ΨΕΥΔΗ**.

5.6.2 Partial look ahead

Ο επόμενος αλγόριθμος, αποκαλούμενος PARTIAL LOOK AHEAD αλγόριθμος αναζήτησης, είναι μια άλλη ειδίκευση του BACKTRACK WITH CONSTRAINT PROPAGATION αλγόριθμου αναζήτησης στον οποίο η διάδοση περιορισμών αποτελείται από τον αλγόριθμο μερικής ματιάς στο μέλλον όπως εξηγείται στην Υποενότητα 5.4.2.

Περιοριζόμαστε στην παρουσίασή του αλγόριθμου αναζήτησης για τα CSPs μόνο με δυαδικούς περιορισμούς. Η επέκταση σε αυθαίρετα CSPs είναι απλή αν και κάποιος πρέπει να αναθεωρήσει με κατάλληλο τρόπο τον αλγόριθμο **DARC** της παραγράφου 7.6 που επιβάλλει την κατευθυντική τοξοειδή σταθερότητα. Παρακάτω χρησιμοποιούμε άλλη, υποδεέστερη, τροποποίηση του αλγόριθμου DARC στην οποία ο αλγόριθμος επίσης ολοκληρώνεται, μόλις μια περιοχή μεταβλητής γίνει κενή. Σε αυτή την περίπτωση η μεταβλητή **failure** τίθεται ως **ΑΛΗΘΗΣ**. Στον παρακάτω αλγόριθμο αυτή η τροποποίηση του αλγόριθμου DARC έχει την ακόλουθη ένδειξη:

```
PROCEDURE darc(i: INTEGER; VAR D: domains;
               VAR failure: BOOLEAN);
```

Η κλήση **darc (i, D, failure)** επικαλείται την προαναφερθείσα τροποποίηση του αλγόριθμου DARC για το CSP που περιορίζεται στις μεταβλητές $x_i,\ldots,x_n$. Στην prop διαδικασία αυτή η διαδικασία υλοποιείται με την πραγματική παράμετρο j+1. Η κατάλληλη prop διαδικασία ορίζεται τώρα ως εξής.

```
PROCEDURE prop(j: INTEGER; VAR D: domains;
               VAR failure: BOOLEAN);
VAR k: INTEGER;
BEGIN
  failure := FALSE;
  k := j+1;
  WHILE k <> n+1 AND NOT failure DO
    revise(j,k,D);
    failure := (D[k] = {});
    k := k+1
  END;
  IF NOT failure THEN
    darc(j+1,D,failure)
  END
END prop;
```

Ο PARTIAL LOOK AHEAD αλγόριθμος αναζήτησης που προκύπτει συνδέεται με το *PARTIAL LOOK AHEAD* δέντρο αναζήτησης με ανάλογο τρόπο όπως αυτός που διατυπώθηκε στο Θεώρημα 5.19 που αναφέρεται στην FORWARD CHECKING αναζήτηση.

5.6.3 Maintaining arc consistency (MAC)

Ο τελικός αλγόριθμος, αποκαλούμενος MAC αλγόριθμος αναζήτησης, ή ***full look ahead*** αλγόριθμος αναζήτησης, είναι μια ειδίκευση του BACKTRACK WITH CONSTRAINT PROPAGATION αλγόριθμου αναζήτησης στον οποίο η διάδοση περιορισμών αποτελείται από την *MAC*, δηλ., τη *FULL LOOK AHEAD* διαδικασία, όπως εξηγείται στην Υποενότητα 5.4.3.

Για να γίνουμε σαφέστεροι εδώ χρησιμοποιούμε μια τροποποίηση του αλγορίθμου **ARC** της Παραγράφου 7.4 στην οποία ο αλγόριθμος ολοκληρώνεται αμέσως μόλις μια περιοχή μεταβλητής γίνει κενή. Σε αυτή την περίπτωση η μεταβλητή **failure** τίθεται ως **ΑΛΗΘΗΣ**. Στον αλγόριθμο αυτή η τροποποίηση του ARC αλγόριθμου έχει την ακόλουθη ένδειξη:

```
PROCEDURE prop(j: INTEGER; VAR D: domains;
                    VAR failure: BOOLEAN);
```

Η κλήση **arc (i, d, failure)** επικαλείται την τροποποίηση του ARC αλγόριθμου την οποία μόλις περιγράψαμε για το CSP που περιορίζεται στις μεταβλητές $x_i,\ldots,x_n$. Στη prop διαδικασία αυτή η διαδικασία επικαλείται με την πραγματική παράμετρο j+1. Η κατάλληλη prop διαδικασία υλοποιείται με τον ίδιο τρόπο όπως στην προηγούμενη υποενότητα: αρκεί να αντικαταστήσουμε την κλήση **darc (j+1, D, failure)** από την **arc (j+1, D, failure)**.

Ο αλγόριθμος αναζήτησης MAC που προκύπτει συνδέεται με το *MAC* δέντρο αναζήτησης με ανάλογο τρόπο όπως αυτός που διατυπώθηκε στο Θεώρημα 5.19 που αναφέρεται στην Αναζήτηση FORWARD CHECKING.

5.6.4 Ψάχνοντας για όλες τις λύσεις

Είναι απλό να τροποποιήσουμε όλους τους αλγόριθμους των δύο τελευταίων τμημάτων σε αυτούς που ένας αλγόριθμος αναζητά όλες τις λύσεις αντί μίας. Για αυτόν τον σκοπό αρκεί σε κάθε αλγόριθμο να αφαιρέσουμε τον όρο **NOT success** από τον βρόγχο **WHILE** και να προσθέσουμε μετά από την ανάθεση **success := (j = n)** τη γραμμή

```
IF success THEN PRINT(inst) END
```

όπου **PRINT** είναι μια διαδικασία με την αναμενόμενη έννοια. Περαιτέρω, η μεταβλητή **success** δεν είναι πλέον αναγκαία, έτσι κάποια μπορεί να την αντικαταστήσει με μια άμεση αναφορά στον έλεγχο j = n. Στην περίπτωση του BACKTRACK αλγόριθμου αναζήτησης της Υποενότητας 5.5.3 αυτές οι τροποποιήσεις οδηγούν στον αλγόριθμο που παρουσιάζεται στο Σχήμα 5.22.

Ο BACKTRACK-ALL(ολική-αντίστροφη διαδρομή) αλγόριθμος αναζήτησης εκτελεί την αναζήτηση στο μειωμένο δέντρο μαρκαρίσματος. Η συμπεριφορά του συνοψίζεται από το ακόλουθο θεώρημα.

Θεώρημα 5.20 (Backtrack-all Αναζήτηση) Ο BACKTRACK-ALL *αλγόριθμος αναζήτησης επισημαίνει όλες τις περιπτώσεις του πίνακα inst που αντιπροσωπεύουν μια λύση.*

Στο επόμενο τμήμα χρησιμοποιούμε αυτόν τον αλγόριθμο αναζήτησης ως βάση για την εξέλιξη των αλγόριθμων στα περιορισμένα προβλήματα βελτιστοποίησης.

5.7 Αλγόριθμοι αναζήτησης για πεπερασμένα περιορισμένα προβλήματα βελτιστοποίησης

Στρέφουμε τώρα την προσοχή μας στα περιορισμένα προβλήματα βελτιστοποίησης που εισάγονται αρχικά στην Παράγραφο 2.6. Έτσι θεωρούμε ένα CSP P := $\langle C ; x_1 \in D_1,\ldots,x_n \in Dn \rangle$ και θεωρούμε μια λετουργία obj : Sol$\rightarrow$R από το σύνολο όλων των λύσεων Sol στο P προς το σύνολο των πραγματικών αριθμών R. Επιδιώκουμε μια λύση d στο P για την οποία η τιμή obj (d) είναι μέγιστη.

```
MODULE backtrack_all;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
PROCEDURE backtrack_all(j: INTEGER; D: domains);
BEGIN
  WHILE D[j] <> {} DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      IF j = n THEN
        PRINT(inst)
      ELSE
        backtrack_all(j+1,D)
      END
    END
  END
END backtrack_all;
BEGIN
  backtrack_all(1,D)
END backtrack_all;
```

Σχήμα 5.22. BACKTRACK-ALL αλγόριθμος αναζήτησης

Θυμηθείτε από την Παράγραφο 3.2 ότι στις χαρακτηριστικές περιπτώσεις του περιορισμένου προβλήματος βελτιστοποίησης έχουμε επίσης στη διάθεσή μας μια ευρετική λειτουργία h που μας επιτρέπει να προσεγγίσουμε αποτελεσματικά την τιμή της obj από παραπάνω. Η ιδέα είναι ότι η λειτουργία h μπορεί να υπολογιστεί πριν όλες οι μεταβλητές γίνουν εντελώς στιγμιαίες και ότι η τιμή που προκύπτει παρέχει ένα **ανώτερο όριο** στις τιμές της obj σε όλες τις λύσεις που επεκτείνουν το τρέχον στιγμιαίο. Αυτό το ανώτερο όριο μπορεί έπειτα να χρησιμοποιηθεί στα στιγμιαία που δεν μπορούν να επεκταθούν σε λύσεις με μεγαλύτερη τιμή της αντικειμενικής λειτουργίας obj από αυτή που μέχρι τώρα γνωρίζουμε. Γενικά, η επιλογή της ευρετικής λειτουργίας h απαιτεί καλές ιδέες στο εξεταζόμενο περιορισμένο πρόβλημα βελτιστοποίησης. Σε αυτό που ακολουθεί υποθέτουμε την ύπαρξη μιας λειτουργίας

$$h : \mathcal{P}(D_1) \times \cdots \times \mathcal{P}(D_n) \rightarrow \mathcal{R} \cup \{\infty\}$$

από το σύνολο όλων των ορισμένων ακολουθιών $E_1,\dots,E_n$ έτσι ώστε $E_1 \subseteq D_1,\dots, E_n \subseteq D_n$, στο σύνολο των πραγματικών αριθμών να αυξάνεται με το ∞ το οποίο ικανοποιεί τις ακόλουθες δύο ιδιότητες, όπου θεωρούμε ότι η σχέση υποσυνόλων επεκτείνεται στις ακολουθίες συνόλων:

Μονοτονία Εάν $\overline{E}_1 \subseteq \overline{E}_2$, τοτε $h(\overline{E}_1) \leq h(\overline{E}_2)$,

Όριο obj $(d_1\dots,d_n) \leq h\ \{d_1\},\dots,\{d_n\})$.

Αυτές οι ιδιότητες είναι προφανώς ειδικεύσεις των αντίστοιχων δύο ιδιοτήτων που έχουν τεθεί ως αξίωμα στην Παράγραφο 3.2 σε μια κατάσταση στην οποία ο διαχωρισμός αποτελείται από το μαρκάρισμα.

Παρακάτω παρουσιάζουμε τρεις αλγόριθμους αναζήτησης που μας επιτρέπουν να ψάξουμε μια λύση στο συνεπώς διατυπωμένο περιορισμένο πρόβλημα βελτιστοποίησης. Περιορίζουμε την προσοχή μας στα πεπερασμένα CSPs και υποθέτουμε ότι ο διαχωρισμός αποτελείται από το μαρκάρισμα. Αυτοί οι αλγόριθμοι χρησιμοποιούν την ίδια αρχή, αποκαλούμενη ***branch και bound***, η οποία εξηγεί τα ονόματά τους. Αντιπροσωπεύουμε σ' αυτούς τις λειτουργίες obj και h με τη βοήθεια των διαδικασιών obj και h με τις δηλώσεις

```
PROCEDURE obj ( inst: instantiation ) : REAL;


PROCEDURE h ( inst: instantiation; j : INTEGER; D: domains) : REAL;
```

Σκοπός είναι η κλήση h (inst, j, D) να επιστρέφει την τιμή της λειτουργίας h στην ακολουθία ({ inst[1]}, ..., {inst[j]}, D[j+1],..., D[n]).

5.7.1 Branch και Bound

Ο πρώτος αλγόριθμος, αποκαλούμενος BRANCH ΚΑΙ BOUND αλγόριθμος αναζήτησης, είναι μια τροποποίηση του BACKTRACK-ALL αλγόριθμου αναζήτησης της Υποενότητας 5.6.4 και παρουσιάζεται στο Σχήμα 5.23.

Ας συνοψίσουμε τις διαφορές μεταξύ του BRANCH ΚΑΙ BOUND αλγόριθμου αναζήτησης και του BACKTRACK-ALL αλγόριθμου αναζήτησης. Κατ' αρχήν, χρησιμοποιούμε στη διαδικασία δύο πρόσθετες παραμέτρους, την **solution**, για να καταγράψουμε την καλύτερη λύση, και την **bound**, για να διατηρήσουμε την πιο γνωστή τιμή του ανώτατου ορίου που ψάχνουμε.

Δεύτερον, όταν αντιμετωπίζεται μια λύση, κι αυτό συμβαίνει όταν ισχύει j = n, δεν επισημαίνεται. Αντ' αυτού, αν προκύψει μια λύση με μία υψηλότερη τιμή της λειτουργίας obj, κι αυτό συμβαίνει εάν ισχύει **obj (inst) > bound**, η μεταβλητή bound ενημερώνεται και αυτή η λύση καταγράφεται στη μεταβλητή solution.

Περαιτέρω, η επαναλαμβανόμενη κλήση εισάγεται μόνο εάν πετύχει η δοκιμή **h (inst, j ,D) > bound**. Σκοπός της είναι να ελέγξει εάν αυτή η κλήση μπορεί να παράγει μια λύση με μία υψηλότερη τιμή της λειτουργίας obj από αυτή που καταγράφεται στη μεταβλητή bound. Για να το θέσουμε αρνητικά, εάν h (inst, j,D) <= bound, τότε λόγω των υποθέσεων της **Μονοτονίας** και του **Ορίου** καμία λύση που επεκτείνει το τρέχον στιγμιαίο δεν μπορεί να παράγει μια τιμή της λειτουργίας obj υψηλότερης από αυτή που καταγράφεται στη μεταβλητή bound. Η εκτέλεση του αλγόριθμου αρχίζει μονογράφωντας τη μεταβλητή solution σε κάποια τιμή NIL που υποτίθεται ότι είναι διαφορετική από όλες τις λύσεις, και τη μεταβλητή bound στη μικρότερη πιθανή τιμή, η οποία αντιπροσωπεύεται από το **-infinity**. Το ακόλουθο θεώρημα συνοψίζει τη συμπεριφορά αυτού του αλγόριθμου.

Θεώρημα 5.21 (Branch και Bound Αναζήτηση) *Εάν υπάρχει μια λύση,* ο BRANCH ΚΑΙ BOUND *αλγόριθμος αναζήτησης κατασκευάζει στον πίνακα solution μια λύση με τη μεγαλύτερη τιμή της obj. Διαφορετικά περιγράφει μια αποτυχία θέτοντας τη μεταβλητή* **solution** *ως* **NIL**.

```
MODULE branch_and_bound;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
PROCEDURE branch_and_bound(j: INTEGER; D: domains;
                           VAR solution: instantiation;
                           VAR bound: REAL);
BEGIN
  WHILE D[j] <> {} DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      IF j = n THEN
        IF obj(inst) > bound THEN
          bound := obj(inst);
          solution := inst
        END
      ELSE
        IF h(inst,j,D) > bound THEN
          branch_and_bound(j+1,D,solution,bound)
        END
      END
    END
  END
END branch_and_bound;
BEGIN
  solution := NIL;
  bound := -infinity;
  branch_and_bound(1,D,solution,bound)
  END
END branch_and_bound;
```

Σχήμα 5.23. Branch και Bound αλγόριθμος αναζήτησης

5.7.2 Branch και Bound με διάδοση περιορισμών

Έπειτα, τροποποιούμε τον BRANCH KAI BOUND αλγόριθμο αναζήτησης ενσωματώνοντας σε αυτόν μια διάδοση περιορισμών. Όπως στην περίπτωση των αλγόριθμων αναζήτησης που παρουσιάζονται στο προηγούμενο τμήμα, έτσι κι εδώ τον παριστάνουμε μέσω της prop διαδικασίας με την δήλωση

```
PROCEDURE prop (j : INTEGER ;  VAR D : domains ;
                    VAR failure :  BOOLEAN) ;
```

Τα κατάλληλα αποτελέσματα τροποποίησης στον αλγόριθμο παρουσιάζονται στο Σχήμα 5.24.

Έτσι ενσωματώσαμε στον BRANCH AND BOUND WITH CONSTRAINT PROPAGATION αλγόριθμο αναζήτησης τις ίδιες τροποποιήσεις με αυτές που χρησιμοποιήσαμε κατά το μετασχηματισμό του BACKTRACK αλγόριθμου αναζήτησης στον BACK-TRACK WITH CONSTRAINT PROPAGATION αλγόριθμο αναζήτησης. Δηλαδή, η επαναλαμβανόμενη κλήση τώρα εισάγεται μόνο εάν η διάδοση περιορισμών, που αντιπροσωπεύεται από την κλήση prop (j, D, failure), δεν θέσει τη μεταβλητή failure ως ΑΛΗΘΗ και όταν πετύχει η δοκιμή h (inst, j, D) > bound. Αυτός ο αλγόριθμος ικανοποιεί την ίδια ιδιότητα με αυτήν που δηλώνεται στο Θεώρημα 5.21που αναφέρεται στην BRANCH AND BOUND Αναζήτηση.

5.7.3 Branch and Bound με διάδοση και κόστος περιορισμών

Στην τελική τροποποίηση του BRANCH AND BOUND αλγόριθμου αναζήτησης ενσωματώνουμε σε αυτόν, εκτός από τη διάδοση περιορισμών, μια ***δυναμική τροποποίηση*** του εξεταζόμενου CSP. Πιό συγκεκριμένα, όταν πετύχει η δοκιμή obj (inst) > bound, εκτός από την ενημέρωση των μεταβλητών bound και solution προσθέτουμε επίσης τον περιορισμό obj $(x_1,\ldots, x_n)$ > bound στο σύνολο των εξεταζόμενων περιορισμών. Υποθέτουμε εδώ ότι η λειτουργία obj είναι ευπροσδιόριστη στη γλώσσα που χρησιμοποιείται για να αντιπροσωπεύσει τους περιορισμούς και ότι ο obj $(x_1,\ldots,x_n)$ > bound είναι ένας επιτρεπτός συντακτικά περιορισμός. Αυτή η περίπτωση ισχύει παραδείγματος χάριν όταν εξετάζουμε τους γραμμικούς περιορισμούς και όταν η λειτουργία obj είναι γραμμική.

Η ιδέα είναι πως όταν πετύχει η δοκιμή obj (inst) > bound, από εδώ και στο εξής πρέπει να ψάχνουμε μόνο για τις λύσεις με την τιμή obj που είναι μεγαλύτερη από μία τιμή που μόλις βρέθηκε και που καταγράφηκε στη μεταβλητή bound.

```
MODULE branch_and_bound_prop;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
    failure: BOOLEAN;
PROCEDURE branch_and_bound_prop(j: INTEGER; D: domains;
                                VAR solution: instantiation;
                                VAR bound: REAL);
BEGIN
  WHILE D[j] <> {} DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      IF j = n THEN
        IF obj(inst) > bound THEN
          bound := obj(inst);
          solution := inst
        END
      ELSE
        prop(j,D,failure);
        IF NOT failure THEN
          IF h(inst,j,D) > bound THEN
            branch_and_bound_prop(j+1,D,solution,bound)
          END
        END
      END
    END
  END
END branch_and_bound_prop;
BEGIN
  solution := NIL;
  bound := -infinity;
  prop(0,D,failure);
  IF NOT failure THEN
    branch_and_bound_prop(1,D,solution,bound)
  END
END branch_and_bound_prop;
```

Σχήμα 5.24. BRANCH AND BOUND WITH CONSTRAINT PROPAGATION αλγόριθμος αναζήτησης

Δηλαδή μπορούμε να επιβάλουμε τον περιορισμό obj $(x_1,\ldots,x_n)$ > bound στο σύνολο των παραγόμενων λύσεων που υπάρχουν ακόμα. Αυτό οδηγεί στον αλγόριθμο που παρουσιάζεται στο Σχήμα 5.25.

Για να διατυπώσουμε τη δυναμική τροποποίηση του συνόλου περιορισμών χρησιμοποιούμε στη διαδικασία branch_and_bound_prop_obj του BRANCH AND BOUND WITH CONSTRAINT PROPAGATION AND COST PROPAGATION αλγόριθμου αναζήτησης μια νέα παράμετρο C ενός ακόμη μη αναλυμένου τύπου περιορισμών που αντιπροσωπεύει το σύνολο των εξεταζόμενων περιορισμών. Σημειώστε ότι η C καταφράφεται ως κλήση μέσω μιας παραμέτρου. Με αυτόν τον τρόπο οι αλλαγές στο σύνολο περιορισμών είναι μόνιμες. Όπως με τον εξεταζόμενο αλγόριθμο προηγουμένως, ο BRANCH AND BOUND WITH CONSTRAINT PROPAGATION AND COST PROPAGATION αλγόριθμος αναζήτησης ικανοποιεί την ίδια ιδιότητα με αυτήν που δηλώνεται στο Θεώρημα 5.21 που αναφέρεται στην BRANCH AND BOUND Αναζήτηση.

Οι δύο τελευταίοι αλγόριθμοι αναζήτησης μπορούν να γίνουν στιγμιαίοι με έναν απλό τρόπο, μέσω μιας αυθαίρετης διάδοσης περιορισμών υπό μορφή μείωσης περιοχών. Ειδικότερα, μπορούν γίνουν στιγμιαίοι μέσω της διάδοσης περιορισμών που υιοθετείται στους FORWARD CHECKING, PARTIAL LOOK AHEAD και MAC αλγόριθμους αναζήτησης. Αυτό οδηγεί στους αλγορίθμους αναζήτησης που συνδυάζουν τα χαρακτηριστικά γνωρίσματα που εισάγονται στα δύο τελευταία τμήματα.

5.8 Ευρετικές λύσεις για τους αλγόριθμους αναζήτησης

Κατά την διάρκεια της συζήτησης των αλγόριθμων αναζήτησης στα δύο τελευταία τμήματα δεχτήκαμε ότι ορίζεται η διάδοση περιορισμών και η διασπασμένη μέθοδος. Εν τούτοις παραμένουν μερικές επιλογές ακόμη. Έχουν σχέση με την επιλογή των μεταβλητών και την επιλογή των τιμών στις περιοχές μεταβλητών. Γενικά είναι πολύ δύσκολο, εάν όχι αδύνατο, να προβλεφθεί η επίδρασή τους στην αποδοτικότητα της αναζήτησης μιας λύσης σε ένα δεδομένο CSP. Ως εκ τούτου,_λόγω της απουσίας συμπληρωματικών πληροφοριών, αυτές οι επιλογές ορίζονται με τη βοήθεια διάφορων ευρετικών λύσεων η αποτελεσματικότητα των οποίων αξιολογείται εμπειρικά.

Ας εξετάσουμε τώρα αυτές τις επιλογές για την περίπτωση των δέντρων μαρκαρίσματος. Μπορούν να ενσωματωθούν απλά σε οποιοδήποτε αλγόριθμο αναζήτησης για τον οποίο γίνεται λόγος στα δύο προηγούμενα τμήματα.

5.8.1 Επιλογή μεταβλητής

Η συγκεκριμένη απόφαση αφορά την επιλογή της επόμενης μεταβλητής για το μαρκάρισμα. Προτάθηκαν δύο απλές ευρετικές λύσεις:

•Επιλέξτε μια μεταβλητή με τη μικρότερη περιοχή.

Αυτή η επιλογή βασίζεται στη Σημείωση 5.5(Μαρκάρισμα) σύμφωνα με την οποία το μέγεθος του πλήρες δέντρου μαρκαρίσματος είναι μικρότερο όταν οι μεταβλητές διατάσσονται σύμφωνα με τα μεγέθη περιοχών τους στην αυξανόμενη σειρά.

```
MODULE branch_and_bound_prop_obj;
TYPE domains =  ARRAY [1..n] OF domain;
     instantiation = ARRAY [1..n] OF elements;
VAR inst: instantiation;
    failure: BOOLEAN;
PROCEDURE branch_and_bound_prop_obj(j: INTEGER;
              VAR C: constraints; D: domains;
              VAR solution: instantiation; VAR bound: REAL);
BEGIN
  WHILE D[j] <> {} DO
    choose d from D[j];
    D[j] := D[j] - {d};
    IF cons(inst,j,d) THEN
      inst[j] := d;
      IF j = n THEN
        IF obj(inst) > bound THEN
          bound := obj(inst);
          solution := inst;
          C := C ∪ {obj(x1,...,xn) > bound}
        END
      ELSE
        prop(j,D,failure);
        IF NOT failure THEN
          IF h(inst,j,D) > bound THEN
            branch_and_bound_prop_obj(j+1,C,D,solution,bound)
          END
        END
      END
    END
  END
END branch_and_bound_prop_obj;
BEGIN
  solution := NIL;
  bound := -infinity;
  prop(0,D,failure);
  IF NOT failure THEN
    branch_and_bound_prop_obj(1,C,D,solution,bound)
  END
END branch_and_bound_prop_obj;
```

Σχήμα 5.25 BRANCH AND BOUND WITH CONSTRAINT PROPAGATION AND COST PROPAGATION αλγόριθμος αναζήτησης

• Επιλέξτε την ***περισσότερο περιορισμένη*** μεταβλητή, δηλ., μια μεταβλητή που εμφανίζεται στο μεγαλύτερο αριθμό περιορισμών.
Μια αιτιολόγηση για αυτήν την επιλογή παρέχεται από μια παρατήρηση που αναφέρει ότι ένα στιγμιαίο μιας τέτοιας μεταβλητής πρέπει να παράγει μια αποτελεσματικότερη μείωση περιοχών με τη βοήθεια της διάδοσης περιορισμών. Παραδείγματος χάριν, μπορεί να μετατρέψει μια μη γραμμική εξίσωση σε μια γραμμική.

Επιπλέον, εάν όλες οι περιοχές μεταβλητών είναι αριθμητικές, μπορούμε να χρησιμοποιήσουμε τις ακόλουθες ευρετικές λύσεις:

• Επιλέξτε μια μεταβλητή με τη μικρότερη διαφορά μεταξύ των ορίων περιοχής της.
Σημειώστε ότι δεν είναι απαραίτητο να είναι μια μεταβλητή με τη μικρότερη περιοχή: πάρτε παραδείγματος χάριν τη μεταβλητή x με την περιοχή {0, 1, 2} και τη μεταβλητή y με την περιοχή {0, 100}.

Και οι τρεις ευρετικές λύσεις είναι περιπτώσεις της αποκαλούμενης ***first-failed*** αρχής σύμφωνα με την οποία όταν βρεθεί κάποιος αντιμέτωπος με μια επιλογή αρχικά δοκιμάζει μια εναλλακτική λύση που το πιο πιθανόν είναι να οδηγεί σε αποτυχία.

Είναι σημαντικό να σημειωθεί ότι οι παραπάνω αρχές επιλογής είναι ***δυναμικές*** υπό την έννοια ότι σε κάθε επίπεδο η επιλογή γίνεται εκ νέου. Τώρα, για να κρατήσουμε τον ορισμό απλό, υποθέσαμε ότι τα δέντρα μαρκαρίσματος και τα prop δέντρα μαρκαρίσματος ορίζονται αναφορικά με ***μια σταθερή εκ των προτέρων*** σειρά μεταβλητών. Για να ερμηνευτεί κατάλληλα αυτός ο ορισμός για τη δυναμική αρχή επιλογής πρέπει να τροποποιηθεί έτσι ώστε να ορίζεται δυναμικά η νέα τρέχουσα μεταβλητή σε κάθε μονό επίπεδο.

5.8.2 Επιλογή τιμής

Κατά την παρουσίαση των αλγόριθμων αναζήτησης υποθέσαμε ότι τα στοιχεία σε κάθε περιοχή μεταβλητής επιλέγονται με έναν μη-ντετερμινιστικό τρόπο. Στην πράξη προκύπτουν διάφορες απλές επιλογές. Στην περίπτωση των αλγόριθμων αναζήτησης για τα περιορισμένα προβλήματα βελτιστοποίησης είναι χρήσιμες οι ακόλουθες ευρετικές λύσεις:

• επιλέξτε μια τιμή για την οποία η ευρετική λειτουργία παράγει το υψηλότερο αποτέλεσμα.

Εάν η ευρετική λειτουργία είναι σωστά επιλεγμένη (δηλαδή προσεγγίζει όχι απόλυτα καλά τη λειτουργία obj), τότε μια τέτοια τιμή είναι "η περισσότερο ενθαρρυντική". Για να είμαστε πιο συγκεκριμένοι, στον BRANCH AND BOUND αλγόριθμο αναζήτησης και στις δύο τροποποιήσεις του, υποθέτοντας ότι j < n , επιλέγουμε ένα στοιχείο d από την περιοχή D[j] για το οποίο ισχύει cons (inst, j ,d) και για το οποίο, μετά από την ανάθεση inst[j] := d, η τιμή του h (inst, j,D) είναι η υψηλότερη. Έτσι αυτές είναι οι ευρετικές κλήσεις για έναν απλό συμπληρωματικό υπολογισμό.

Περαιτέρω, κατά την διάρκεια της συζήτησης των αριθμητικών περιοχών, προκύπτουν οι ακόλουθες απλές επιλογές κατά την επιλογή της επόμενης τιμής σε μια περιοχή:

- επιλέξτε τη μικρότερη τιμή,
- επιλέξτε τη μεγαλύτερη τιμή,
- επιλέξτε τη μεσαία τιμή

Οι δύο πρώτες ευρετικές λύσεις είναι προφανείς. Η τρίτη παρακινείται από την παρατήρηση ότι σε μερικές περιπτώσεις η επιλογή της μεσαίας τιμής μπορεί να οδηγήσει σε μια αποτελεσματικότερη διάδοση περιορισμών. Παραδείγματος χάριν, έχει διαπιστωθεί ότι αυτή η ευρετική λύση είναι χρήσιμη για το CSP τυποποιώντας το Πρόβλημα των n Βασιλισσών που παρουσιάστηκε στο Παράδειγμα 2.2.

5.9 Ένας abstract branch and bound αλγόριθμος

Ξεκινήσαμε αυτό το κεφάλαιο με τον ορισμό των αυθαίρετων δέντρων αναζήτησης στην Παράγραφο 5.1. Κατόπιν, αφού συζητήσαμε για τα δέντρα μαρκαρίσματος παρουσιάσαμε τους αλγόριθμους αναζήτησης. Ας κλείσουμε τώρα αυτόν τον κύκλο αναφέροντας εν συντομία τους αλγόριθμους αναζήτησης και πιο συγκεκριμένα τα αυθαίρετα δέντρα αναζήτησης. Σε αυτά τα δέντρα η διάδοση περιορισμών διαμορφώνεται με τη βοήθεια μιας μετάβασης από ζυγά σε μονά επίπεδα. Έτσι πρέπει να επιστήσουμε την προσοχή μας σε τρεις αλγορίθμους αναζήτησης, στον BACKTRACK-FREE WITH CONSTRAINT PROPAGATION, στον BACKTRACK WITH CONSTRAINT PROPAGATION και τέλος στον BRANCH AND BOUND WITH CONSTRAINT PROPAGATION. Παρουσιάζουμε τώρα τον τελευταίο αλγόριθμο σε μια συνοπτική μορφή δίνοντας την παρουσίαση των δύο άλλων, απλούστερων, ως Άσκηση 5.11.

Ορίζουμε ένα πεπερασμένο δέντρο αναζήτησης T και το αναπαριστάνουμε με τη βοήθεια ενός πίνακα **children** ενός περαιτέρω αναλυμένου τύπου **searchtree** όπου δεδομένου ενός CSP P αποδίδει στο **children [P]** το σύνολο των άμεσων απογόνων του στο T. Επίσης, θεωρούμε έναν ακόμη αναλυμένο τύπο CSP. Επί πλέον, θεωρούμε μια λειτουργία **next** η οποία θα εφαρμοστεί σε ένα CSP σε μονό επίπεδο για να έχει πρόσβαση στον μοναδικό άμεσο απόγονό, χρησιμοποιώντας την οδηγία **R := next (P).**

Ακόμα υποθέτουμε ότι ένα CSP είναι ένα φύλλο του T εάν και μόνο εάν είναι "προφανώς λυμένο" ή "προφανώς αποτυχημένο". Για να εξετάσουμε τη κατάσταση ενός CSP θεωρούμε ότι υπάρχουν δύο διαδικασίες, η **solved** που επιστρέφει την πραγματική τιμή της δήλωσης "το CSP" είναι "προφανώς λυμένο" και η **failed** που επιστρέφει την πραγματική τιμή της δήλωσης "το CSP" είναι "προφανώς αποτυχημένο"

Τέλος, διαμορφώνουμε τη βελτιστοποίηση θεωρώντας την ύπαρξη μιας λειτουργίας obj όπου δεδομένου ενός "προφανώς λυμένου" CSP επιστρέφει την αντικειμενική τιμή του που είναι μια πραγματική τιμή. Στο δέντρο αναζήτησης T ψάχνουμε για ένα "προφανώς λυμένο" CSP με μια μέγιστη τιμή obj. Για αυτόν τον σκοπό χρησιμοποιούμε μια ευρετική λειτουργία h που ορίζει σε κάθε CSP P μία πραγματική τιμή στο T και θεωρούμε δύο ιδιότητες των λειτουργιών obj και του h:

- εάν το R είναι άμεσος απόγονος του P, τότε $h(R) \leq h(P)$,
- εάν το P είναι ένα "προφανώς λυμένο" CSP, τότε $obj(P) \leq h(P)$.

Ο αλγόριθμος παρουσιάζεται στο Σχήμα 5.26. Για την κατάλληλη λειτουργία του είναι ουσιαστικό ότι η παράμετρος **children** παραβλέπεται από την τιμή, έτσι ώστε οι τροποποιήσεις που εκτελούνται κατά τη διάρκεια της επαναλαμβανόμενης κλήσης να πραγματοποιούνται μόνο στην τοπική εκδοχή της.

Το ακόλουθο θεώρημα συνοψίζει τη συμπεριφορά αυτού του αλγόριθμου.

Θεώρημα 5.22 (Abstract Branch and Bound Αναζήτηση) Καλούμε ένα CSP P "προφανώς λυμένο" εάν ισχύει το **solved(P)**. Εάν στο εξεταζόμενο δέντρο αναζήτησης με τη ρίζα **Pinit** υπάρχει ένα "προφανώς λυμένο" CSP, ο ABSTRACT BRANCH AND BOUND αλγόριθμος αναζήτησης επιστρέφει στο P αυτό το CSP με τη μεγαλύτερη τιμή obj. Διαφορετικά δηλώνει μια αποτυχία θέτοντας τη μεταβλητή **solution** ως **NIL**.

Αυτό το θεώρημα ισχύει για αυθαίρετα CSPs, ειδικότερα για αυτά με τις άπειρες περιοχές. Ο ABSTRACT BRANCH AND BOUND αλγόριθμος αναζήτησης μπορεί να ειδικευτεί με πολλούς τρόπους επιλέγοντας τους ορισμούς ενός "προφανώς λυμένου" και ενός "προφανώς αποτυχημένου" CSP, με την επιλογή της διασπασμένης μεθόδου, της μορφής διάδοσης περιορισμών, της λειτουργίας obj, και της ευρετικής λειτουργίας.

Δεδομένης μιας επεξήγησης της χρήσης του ας επιστρέψουμε στην προσέγγιση του υπολογισμού των βέλτιστων λύσεων στους πολυωνυμικούς περιορισμούς στα διαστήματα ακέραιων αριθμών υποκείμενων σε μια πολυωνυμική αντικειμενική λειτουργία που αναφέραμε στην Παράγραφο 3.4. Εκεί χρησιμοποιήσαμε μια συγκεκριμένη λειτουργία obj, ορίσαμε μια συγκεκριμένη διασπασμένη μέθοδο υπό μορφή διχοτόμησης, και ορίσαμε τέλος τη διάδοση περιορισμών και την ευρετική λειτουργία h χρησιμοποιώντας αριθμητικό διάστημα στους ακέραιους αριθμούς. Για να ολοκληρώσουμε τις λεπτομέρειες του ABSTRACT BRANCH AND BOUND αλγόριθμου αναζήτησης ορίζουμε ένα CSP "προφανώς λυμένο" εάν λύνεται καθώς και όλες τις περιοχές του μονοσύνολα, και "προφανώς αποτυχημένο" εάν αποτύχει.

Σημειώστε επίσης ότι εξετάζοντας τα πεπερασμένα CSPs με το διαχωρισμό που ορίζεται ως μαρκάρισμα και ορίζοντας ως "προφανώς λυμένο" " ένα λυμένο CSP με τις περιοχές να είναι μονοσύνολα" λαμβάνουμε το περιορισμένο πρόβλημα βελτιστοποίησης για πεπερασμένα CSPs που ήδη αναφέρθηκε και τον BRANCH AND BOUND WITH CONSTRAINT PROPOGATION αλγόριθμο αναζήτησης.

Είναι δύσκολο να ορίσουμε ευρετικές λύσεις για τα αυθαίρετα δέντρα αναζήτησης. Αυτό μπορεί να γίνει για κάποιες ειδικές περιπτώσεις διαχωρισμού και διάδοσης περιορισμών. Παραδείγματος χάριν, η ευρετική λύση μετά από την επιλογή μεταβλητής που ορίζεται στην Παράγραφο 5.8 μπορεί επίσης να χρησιμοποιηθεί όταν ο διαχωρισμός αποτελείται από τη διχοτόμηση της περιοχής της τρέχουσας μεταβλητής με μια αριθμητική περιοχή χρησιμοποιώντας μια μεσαία τιμή μεταξύ των ορίων.

```
MODULE abstract_branch_and_bound;
PROCEDURE abstract_branch_and_bound(children: searchtree;
                    VAR solution: CSP; VAR bound: REAL);
BEGIN
  WHILE children[P] <> {} DO
    choose R from children[P];
    children[P] := children[P] - {R};
    IF NOT failed(R) THEN
      P := R;
      IF solved(P) THEN
        IF obj(P) > bound THEN
          bound := obj(P);
          solution := P
        END
      ELSE
        P := next(P);
        IF NOT failed(P) THEN
          IF h(P) > bound THEN
            abstract_branch_and_bound(children,solution,bound)
          END
        END
      END
    END
  END
END abstract_branch_and_bound;
BEGIN
  solution := NIL;
  bound := -infinity;
  P := next(Pinit);
  IF NOT failed(P) THEN
    abstract_branch_and_bound(children,solution,bound)
  END
END abstract_branch_and_bound;
```

Σχήμα 5.26. Ο ABSTRACT BRUNCH ANDBOUND αλγόριθμος αναζήτησης

5.10 Περίληψη

Σε αυτό το κεφάλαιο μελετήσαμε τους αλγόριθμους αναζήτησης. Για αυτόν τον σκοπό παρουσιάσαμε αρχικά τα δέντρα αναζήτησης που προκύπτουν από μια εναλλασσόμενη χρήση της διάδοσης περιορισμών και του διαχωρισμού. Κατόπιν παρουσιάσαμε τους αλγόριθμους αναζήτησης ως αλγόριθμους που διερευνούν τα δέντρα αναζήτησης.

Το μεγαλύτερο μέρος της παρουσίασης ασχολήθηκε με τα δέντρα αναζήτησης που αποκτήθηκαν όταν χρησιμοποιήθηκε το μαρκάρισμα ως μέθοδος διαχωρισμού. Τα ονομάσαμε δέντρα μαρκαρίσματος. Παρουσιάσαμε τρεις τύπους δέντρων μαρκαρίσματος που προκύπτουν από την επιλογή μιας συγκεκριμένης μορφής διάδοσης περιορισμών:

- forward checking,
- partial look ahead και
- διατήρηση τοξοειδούς σταθερότητας (MAC),

και παρουσιάσαμε διάφορους αλγόριθμους αναζήτησης για αυτά τα δέντρα, δηλαδή

- τον backtrack-free αλγόριθμο αναζήτησης,
- τον backtrack αλγόριθμο αναζήτησης,

και τις τροποποιήσεις τους που περιλαμβάνουν τη διάδοση περιορισμών.

Επίσης, αναφέραμε διάφορους αλγόριθμους αναζήτησης για τα περιορισμένα προβλήματα βελτιστοποίησης, στόχος των οποίων είναι η εύρεση μιας βέλτιστης λύσης. Αυτοί οι αλγόριθμοι στηρίζονται στην

- αρχή branch and bound.

Παρουσιάσαμε τους αλγόριθμους αναζήτησης για τα δέντρα μαρκαρίσματος με έναν ενιαίο τρόπο, μέσω διαδοχικών υποδεέστερων τροποποιήσεων. Αυτό μας επέτρεψε να διευκρινίσουμε πώς αυτοί οι αλγόριθμοι αλληλοσχετίζονταν. Εξηγήσαμε επίσης πώς μπορούν να συνδυαστούν οι αλγόριθμοι αναζήτησης με τα δέντρα μαρκαρίσματος με διάφορες ευρετικές λύσεις που έχουν σχέση με την επιλογή της μεταβλητής και της τιμής. Τέλος, υποστηρίξαμε ότι οι βασικοί αλγόριθμοι αναζήτησης μπορούν επίσης να παρουσιαστούν για τα αυθαίρετα δέντρα αναζήτησης.

ΒΙΒΛΙΟΓΡΑΦΙΑ

K. R. APT

[2003] Principles of Constraint Programming, CWI, Amsterdam, The Netherlands, Cambridge University Press, UK.

A. AGGOUN AND N. BELDIOEANU

[1993] Extending CHIP in order to solve complex scheduling and placement problems, Mathematical and Computer Modelling, 17, pp. 57-73.

K. R. APT

[1998] A proof theoretic view of constraint programming, Fundamenta Informaticae, 33, pp. 263-293. Available via http://arXiv.org/archive/cs/.

K. R. APT

[1999] The essence of constraint propagation, Theoretical Computer Science, 221, pp. 179-210. Available via http://arXiv.org/archive/cs/.

K. R. APT

[2000a] The role of commutativity in constraint propagation algorithms, ACM Transactions on Programming Languages and Systems, 22, pp. 1002-1036. Available via http://arXiv.org/archive/cs/.

K. R. APT

[2000b] Some remarks on Boolean constraint propagation, in: New Trends in Constraints, K. R. Apt, A. C. Kakas, E. Monfroy, and F. Rossi, eds., vol. 1865 of Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 91-107. Available via http://arXiv.org/archive/cs/.

K. R. APT AND P. ZOETEWEIJ

[2003] A comparative study of arithmetic constraints on integer intervals, in: Proceedings of the 2003 BRCIM Workshop on Constraints, MTA SZ-TAKI Available via http://www.cwi.nl/~apt.

F. BAADER AND T. NIPKOW

[1998] Term Rewriting and All That, Cambridge University Press, Cambridge, UK.

F. BAADER AND J. II. SIEKMANN

[1994] Unification theory, in: Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 2, Deduction Methodologies, D. M. Gabbay, C. J. Hogger, and J. A. Robinson, eds., Oxford University Preys, pp. 41-125.

A. B. BABICHEV, O. B. KADYROVA, T. P. KAHEVAROVA, A. S. LESHCHENKO, AND A. L. SEMENOV

[1993] UniCalc, a novel approach to solving systems of algebraic equations, Interval Computations, N2, pp. 29-47.

F. BACCHUS AND P. VAN BEEK

[1998] On the conversion between non-binary and binary constraint satisfaction problems, in: AAAI-98: Proceedings of the 15th National Conference on Artificial Intelligence, AAAI Press, Menlo Park.

R. BARTAK

[2002] On-line guide to constraint programming. Available via http://kti. ms.mff.cuni.cz/"bartak/constraints/index.html.

N. BELDICEANU

[2000] Global constraints as graph properties on structured network of elementary constaints of the same type, Tech. Rep. T2000-01, Swedish Institute of Computer Science (S1CS). Available via http://www.sics.se/isl/ cps/.

F. BENHAMOU

[1996] Heterogeneous constraint solving, in: Proceeding of the Fifth International Conference on Algebraic and Logic Programming (ALP 96), M. Hanus and M. Rodriguez-Artalejo, eds., Lecture Notes in Computer Science 1139, Springer-Verlag, Berlin, pp. 62-76.

F. BENHAMOU AND A. COLMERAUER

[1993] eds., Constraint Logic Programming: Selected Research, The MIT Press.

F. BENHAMOU, F. GOUALARD, AND L. GRANVILLIERS

[1997] Programming with the Declic language, in: Proceedings of the Second Workshop on Interval Constraints (October 1997, Port-Jefferson, NY).

F. BENHAMOU, F. GOUALARD, AND L. GRANVILLIERS

[2000] Interval constraints: Results and perspectives, in: New Trends in Constraints, K. R. Apt, A. C. Kakas, E. Monfroy, and F. Rossi, eds., vol. 1865 of Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 1-16.

F. BENHAMOU, D. A. MCALLESTER, AND P. VAN HENTENRYCK

[1994] CLP (intervals) revisited, in: Proceedings of the 1994 International Logic Programming Symposium, M. Bruynooghe, ed., The MIT Press, pp. 124-138.

F. BENHAMOU AND W. OLDER

[1997] Applying interval arithmetic to real, integer and Boolean constraints, Journal of Logic Programming, 32, pp. 1-24.

C. BESSIERE, P. MESEGUER, E. C. FREUDER, AND J. LARROSA

[2002] On forward checking for non-binary constraint satisfaction, Artificial Intelligence, 141, pp. 205-224.

C. BESSIERE AND J. C. REGIN

[2001] Refining the basic constraint propagation algorithm, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-01), Morgan Kaufmann, Seattle, WA, USA, pp. 309-315.

S. BISTARELLI, H. FARGIER, U. MONTANARI, F. ROSSI, T. SCHIEX, AND G. VERPAILME

[1996] Semiring-based CSPs and valued CSPs: basic properties and comparison, in: Over-Constrained Systems, M. Jampel, E. C. Freuder, and M. J. Maher, eds., vol. 1106 of Lecture Notes in Computer Science, Springer, pp. I11-150.

S. BISTARELLI, U. MONTANARI, AND F. ROSSI

[1997] Semiring-based constraint satisfaction and optimization, Journal of the ACM, 44, pp. 201-236.

N, BLEUZEN-GUERNALEC AND A. COLMERAUER

[2000] Optimal narrowing of a block of sortings in optimal time, Constraints, 5, pp. 85 118.

A. BOCKMAYR AND T. KASPE

[1998] A unifying framework for integer and finite domain constraint programming, INFORMS Journal on Completing, 10, pp. 287-300.

A. BORNING AND B. FREEMAN-BENSON

[1998] Ultraviolet: a constraint satisfaction algorithm for interactive graphics, Constraints, 3, pp. 9-32.

A. BORNING, B. FREEMAN-BENSON, AND M. WILSON

[1992] Constraint hierarchies, Lisp and Symbolic Computation, 5, pp. 223-270.

P. BOROVANSKY, C. KIRCHNER, H. KIRCHNER, P.-E. MOREAU, AND C. RINGEISSEN

[1998] An Overview of ELAN, in: Proceedings of the Second International Workshop on Rewriting Logic and its Applications, C. Kirchner and H. Kirchner, eds., vol. 15 of Electronic Notes in Theoretical Computer Science, Elsevier, Pont-à-Mousson (France), September.

R. BOSCH

[1999] Peaceably coexisting armies of queens, OPTIMA (Newsletter of the Mathematical Programming Society), 62, pp. 6-9.

T. BY

[1997] Line Labelling by Meta-programming, Tech. Rep. CS-97-07, University of Sheffield.

Y. CASEAU, P.-X. JOSSET, AND F. LABURTHE [2002] CLAIRE: Combining sets, search and rules to better express algorithms, Theory and Practice of Logic Programming, 2, pp. 769-805.

C. CASTRO

[1998] Building constraint satisfaction problem solvers using rewrite rules and strategies, Fundamenta Informaticae, 33, pp. 263-293.

P. CODOGNET AND D. DIAZ

[1996] A simple arid efficient Boolean constraint solver for constraint logic programming, Journal of Automated Reasoning, 17, pp. 97-128.

J. COHEN

[1990] Constraint logic programming languages, Communications of the ACM, 33, pp. 52-68.

A. G. COHN AND S. M. HAZARIKA

[2001] Qualitative spatial representation and reasoning: an overview, Fundamenta Informaticae, 46, pp. 1-29.

A. COLMERAUER

[1990] An introduction to Prolog III, Communications of the ACM, 33, pp. 69 90.

A. COLMERAUER

[1993] Naive solving of non-linear constraints, in: Constraint Logic Programming: Selected Research, P. Benhamou and A. Colmerauer, eds., The MIT Press, pp. 89-112.

A. COLMERAUER

[2001] Solving the multiplication constraint in several approximation spaces, in: 17th International Conference on Logic Programming, (ICLP 2001), P. Codognet, ed., vol. 2237 of Lecture Notes in Computer Science, Springer-Verlag, p. 1. The transparencies of the lecture are at http://www.lim.univ-mrs.fr/~colmer/Transparents/Paphos01/ paphosps.zip.

K. DARBY-DOWMAN AND J. LITTLE

[1998] Properties of some combinatorial optimization problems and their effect on the performance of integer programming and constraint logic programming, INFORMS Journal on Computing, 10, pp. 276-286.

A. J. DAVENPORT, E. TSANG, C. J. WANG, AND K. ZHU

[1994] GENET: A connectionist architecture for solving constraint satisfaction problems by iterative improvement, in: National Conference on Artificial Intelligence, pp. 325-330.

R. DECHTER AND P. VAN BEEK

[1997] Local and global relational consistency, Theoretical Computer Science, 173, pp. 283-308.

R. DECHTER AND D. FROST

[2002] Backjump-based backtracking for constraint satisfaction problems, Artificial Intelligence, 136, pp. 147-188.

A. DOLLAS, W. T. RANKIN, AND D. MCCRACKEN

[1998] A new algorithm for Golomb ruler derivation and proof of the 19 mark rule, IEEE Transactions on Information Theory, pp. 379-382.

D. DUBOIS, II. FARGIER, AND H. PRADE

[1993] The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction, in: Proceedings 2nd IEEE Conference on Fuzzy Sets, San Francisco, Mar.

M. EGENHOFER

[1991] Reasoning about binary topological relations, in: Proceedings of the 2nd International Symposium on Large Spatial Databases (SSD), O. Günther and H.-J. Schek, eds., vol. 525 of Lecture Notes in Computer Science, Springer-Verlag, pp. 143-160.

M. EGENHOFER

[1994a] Deriving the composition of binary topological relations, Journal of Visual Languages and Computing, 5, pp. 133-149. Cited on page 51.

M. EGENHOFER

[1994b] Pre-processing queries with spatial constraints, Photogrammetric Engineering & Remote Sensing, 60, pp. 783-790.

A. E. EIBEN AND Z. RUTTKAY

[1997] Constraint satisfaction problems, in: Handbook of Evolutionary Computation, T. Back, D. B. Fogel, and Z. Michalewicz, eds., Institute of Physics Publishing and Oxford University Press, Bristol, New York, pp.

C5.7:l-8.

F. FAGES, J. FOWLER, AND T. SOLA

[1998] Experiments in reactive constraint logic programming, Journal of Logic Programming, 37, pp. 185-212.

P. FLENER, A. FRISCH, B. HNICH, Z. KIZILTAN, I. MIGUEL, J. PEARSON, AND T. WALSH

[2002] Breaking row and column symmetries in matrix models, in: Proceedings of the Eighth International Conference on Principles and Practice of Constraint Programming (CP '02), vol. 2470 of Lecture Notes in Computer Science, Springer-Verlag, pp. 462-477.

P. FLENER AND J. PEARSON

[2002] eds., SymCon'02: The Second International Workshop on Symmetry in Constraint Satisfaction Problems. Available via http://www.it.uu.se/research/group/astra/SymCon02.

R. FOURER, D. M. GAY, AND B. W. KERNIGHAM

[1993] AMPL: A Modeling Language for Mathematical Programming, The Scientific Press Series.

E. C. FREUDER AND R, WALLACE

[1992] Partial constraint satisfaction, Artificial Intelligence, 58, pp. 21-70.

T. FRUHWIRTH

[1995] Constraint Handling Rules, in: Constraint Programming: Basics and Trends, A. Podelski, ed., LNCS 910, Springer-Verlag, pp. 90-107. (Chatillon-sur-Seine Spring School, France, May 1994).

T. FRUHWIRTH

[1998] Theory and practice of Constraint Handling Rules, Journal of Logic Programming, 37, pp. 95 138. Special Issue on Constraint Logic Programming (P. J. Stuckey and K. Marriot, Eds.).

M. GARDNER

[1997] The Last Recreations: Hydras, Eggs, and Other Mathematical Mystifications, Copernicus Books.

R. GENNARI

[2000] Arc consistency via subsumed functions, in: Proceedings of Computational Logic 2000 (CL2000), J. Lloyd, ed., Lecture Notes in Artificial Intelligence 1861, Springer-Verlag, Berlin, pp. 358-372.

R. GENNARI

[2001] General schema for constraint propagation, Joint Bulletin of the Novosibirsk Computing Center and Institute of Informatics Systems. Series: Computer Science, 16, pp. 25-40.

L. GRANVILLIERS, E. MONFROY, AND F. BENHAMOU

[2001] Symbolic-interval cooperation in constraint programming, in: Proceedings of the 26th ACM international Symposium, on Symbolic and Algebraic Computation (ISSAC 2001), ACM Press, pp. 150-166.

E. HANSEN

[1992] Global Optimization Using Interval Analysis, Marcel Dekker.

W. HARVEY AND P. J. STUCKEY

[2003] Improving linear constraint propagation by changing constraint representation, Constraints, 8, pp. 173-207.

W. D. HARVEY AND M. L. GINSBERG

[1995] Limited discrepancy search, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95); Vol. 1, C. S. Mellish, ed., Morgan Kaufmann, Montreal, Quebec, Canada, pp. 607-615.

M. HENZ, Y. F. LIM, S. C. LUA, X. P. SHI, J. P. WALSER, AND R. H. C. YAP

[2003] Solving hierarchical constraints over finite domains with local search, Annals of Mathematics and Artificial Intelligence. To appear.

T. J. HICKEY, Q. JU, AND M. H. VAN EMDEN

[2001] Interval arithmetic: from principles to implementation, Journal of the ACM, 48, pp. 1038-1068.

T. J. HICKEY, M. H. VAN EMDEN, AND H. WU

[1998] A unified framework for interval constraints and interval arithmetic, in: Proceedings of the Fourth International Conference on Principles and Practice of Constraint Programming (CP'98), M. J. Maher and J.-F. Puget, eds., vol. 1520 of Lecture Notes in Computer Science, Springer-Verlag, pp. 250-264.

W. J. VAN HOEVE

[2001] The all different constraint: A survey, November. Submitted for publication. Available via http://www.cwi. nl/~wjvh/papers/alldiff.ps.gz.

E. ILOG

[2003] Ilog white papers. Available via http://www.ilog.com/products/ optimization/papers.cfm.

J.-L. IMBERT

[1995][Fourier Elimination: which to choose, in: Principles and Practice of Constraint Programming, P. Van Hentenryck and V. Saraswat, eds., MIT Press, pp. 245-268.

J. JAFFAR AND M. J. MAUER

[1994] Constraint logic programming: a survey, Journal of Logic Programming, 19/20, pp. 503-581.

J. JAFFAR, M. J. MAKER, P. J. STUCKEY, AND R. H. C. YAP

[1993] Projecting CLP(R) constraints, New Generation Computing, 11, pp. 449-469.

J. JAFFAR, S. MICHAYOV, P. J. STUCKEY, AND R. H. C. YAP

[1992] The CLP(R) language and system, ACM Transactions on Programming Languages and Systems, 14, pp. 339-395.

M. JAMPEL, E. C. FREUDER, AND M. J. MAHER

[1996] eds., Over-Constrained Systems, vol. 1106 of Lecture Notes in Computer Science, Springer.

C. KIRCHNER AND C. RINGEISSEN

[1998] Rule-based constraint programming, Fundamenta Informaticae, 34, pp. 225-262.

F. LABURTHE AND Y. CASEAU

[2002] SALSA: A language for search algorithms, Constraints, 7, pp. 255-288.

O. LHOMME

[1993] Consistency techniques for numeric CSPs, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-93), pp. 232-238.

A. MAOKWORTH

[1992] Constraint satisfaction, in: Encyclopedia of Artificial Intelligence, S. C. Shapiro, ed., John Wiley and Sons, pp. 285-293, Volume 1.

K. MARRIOTT AND P. J. STUCKEY

[1998] Programming With Constraints: An Introduction, The MIT Press.

D. MCALLESTE

[1990] Truth maintenance, in: AAAI-90: Proceedings 8th National Conference on Artificial Intelligence, pp. 1109-1116.

K. MCALOON AND C. TRETKOFF

[1996] Optimization and Computational, John Wiley and Sons.

P. MESEGUER, N. BOUHMALA, T. BOUZOUBAA, M. IRGENS, AND M. SANCHEZ

[2003] Current approaches for solving over-constrained problems, Constraints, 8, pp. 9-39.

Z. MICHALEWIGZ

[1996] Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin, third ed.

L. MICHEL AND P. VAN HENTENRYCK

[2002a] A constraint-based architecture for local search, in: Proceedings of the 17th ACM conference on Object-oriented programming, systems, languages, and applications, ACM Press, pp. 83-100.

L. MICHEL AND P. VAN HENTENRYCK

[2002b] A decomposition-based implementation of search strategies. To appear in ACM Transactions on Computational Logic. Available via http:// www.acm.org/tocl.

S. MINTON, M. D. JOHNSTON, A. B. PHILIPS, AND P. LAIRD

[1992] A heuristic repair method for constraint satisfaction and scheduling problems, Artificial Intelligence, 58, pp. 161-205.

E. MONFROY AND J.-H. RETY

[1999] Chaotic iteration for distributed constraint propagation, in: Proceedings of the 14th ACM Symposium on Applied Computing, ACM SAC'99, Scientific Computing Track, J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, and G. Lamont, eds., ACM Press, San Antonio, Texas, USA, March, pp. 19-24.

U. MONTANARI AND F. R.OSSI

[1991] Constraint relaxation may be perfect, Artificial Intelligence, 48, pp. 143 -170.

W. OLDER, G. SWINKELS, AND-M. II. VAN EMDEN

[1995] Getting to the real problem: experience with BNR Prolog in OR, in: Proceedings of the Third International Conference on the Practical Application of Prolog (PAP '95), L. Sterling, ed., Alinmead Software Ltd, pp. 465-478.

W. OLDER AND A. VELLINO

[1993] Constraint arithmetic on real intervals, in: Constraint Logic Programming: Selected Research, F. Benhamou and A. Colmerauer, eds., The MIT Press, pp. 175-195.

W. H. PRESS, B. P. FLANNERY, S. A. TEUKOLSKY, AND W. T. VETTBRLING

[1992] Numerical Recipes in C : The Art of Scientific Computing, Cambridge University Press, 2nd ed.

D. A. RANDELL, A. G. COHN, AND Z. CUI

[1992] Computing transitivity tables: A challenge for automated theorem provers, in: Proceedings CADE 11, Springer, Berlin.

D. RATZ

[1996] Inclusion Isotone Extended Interval Arithmetic, tech. rep., University of Karlsruhe. Report No. D-76128 Karlsruhe.

J. C. REGIN

[1994] A filtering algorithm for constraints of difference in CSPs, in: AAAI-94: Proceedings of the 12th National Conference on Artificial Intelligence, pp. 362-367.

J. A. ROBINSON

[1992] Logic and logic programming, Communications of ACM, 35, pp. 40-65.

H. RUDOVA

[2001] Soft scheduling, in: Proceedings of the 2001 ERCIM Workshop on Constraints, Charles University in Prague, Faculty of Mathematics and Physics. Available via http://arXiv.org/abs/cs.AI/0106004.

S. RUSSELL AND P. NORVIG

[2003] Artifical Intelligence: A Modern Approach, Prentice-Hall, Englewood Cliffs, NJ, second ed.

Z. RUTTKAY

[1994] Fuzzy constraint satisfaction, in: Proceedings 1st IEEE Conference on Evolutionary Computing, Orlando, pp. 542-547.

D. SABIN AND E. C. FREUDER

[1994] Contradicting conventional wisdom in constraint satisfaction, in: Proceedings ECAI'94, Amsterdam, pp. 125-129.

D. SAM-HAROUD AND B. FALTINGS

[1996] Consistency techniques for continuous constraints, Constraints, 1, pp. 85-118.

V, SARASWAT

[1993] Concurrent Constraint Programming, The MIT Press.

V. SARASWAT, M. RINARD, AND P. PANANGADEN

[1991] Semantic foundations of concurrent constraint programming, in: Proceedings of the Eighteenth Annual ACM Symposium on Principles of Programming Languages (POPL'91), pp. 333-352.

A. SCHAERF, M. CADOLI, AND M. LENZERJNI

[2000] LOCAL++: A C++ framework for combinatorial search problems, Software Practice and Experience, pp. 233-257.

T. SCHIEX, H. FARGTER, AND G. VERFAILLIE

[1995] Valued constraint satisfaction problems: hard and easy problems, in: IJCAP95, Montreal, Canada, pp. 631-637.

C. SCHULTE

[1997] Programming constraint inference engines, in: Proceedings of the Third International Conference on Principles and Practice of Constraint Programming (CP '97), G. Smolka, ed., vol. 1330 of Lecture Notes in Computer Science, Springer-Verlag, pp. 519-533.

C. SCHULTE

[2002] Programming Constraint Services, vol. 2302 of Lecture Notes in Computer Science, Springer.

C. SCHULTE AND G. SMOLKA

[2002] Finite domain constraint programming in Oz. A tutorial, August. Version 1.2.4 (20020829). Available via http://www.mozart-oz.org/documentation/fdt/index.html.

E. SCHWALB AND L. VILA

[1998] Temporal constraints: a survey, Constraints, 3, pp. 129-149.

H. SIMONIS

[1996] A problem classification scheme for finite domain constraint solving, in: Proceedings of the CP'96 workshop on Constraint Programming Applications, Cambridge MA, pp. 1-26. Available via http://www.cs.wfu.edu/~burg/CP96/cp96.html.

B. M. SMITH, K. STERGIOU, AND T. WALSH

[2000] Using auxiliary variables and implied constraints to model non-binary problems, in: AAAI-00: Proceedings National Conference on Artificial Intelligence, pp. 182-187.

G. SMOLKA

[1995] The Oz programming model, in: Computer Science Today, J. van Leeuwen, ed., vol. 1000 of Lecture Notes in Computer Science, Springer-Verlag, pp. 324-343.

V. TELERMAN AND D. USHAKOV

[1996] Data types in subdefinite models, in: Artificial Intelligence and Symbolic Mathematical Computations, Lecture Notes in Computer Science 1138, Springer-Verlag, Berlin, pp. 305-319.

M. TRICK

[2001] A dynamic programming approach for consistency and propagation for knapsack constraints, in: Proceedings of the Third International Workshop on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR-01), pp. 113-124. Available via http://www.icparc.ic.ac.uk/cpAI0R01/.

E. TSANG

[1993] Foundations of Constraint Satisfaction, Academic Press.

P. VAN BEEK

[1992] Reasoning about qualitative temporal information, Artificial Intelligence, 58, pp. 297-326.

P. VAN BEEK AND R. DECHTER

[1995] On the minimality and global consistency of row-convex constraint networks, Journal of the ACM, 42, pp. 543-561.

M. H. VAN EMDEN

[1997] Value constraints in the CLP scheme, Constraints, 2, pp. 163-184.

M. H. VAN EMDEN

[1999] The logic programming paradigm in numerical computation, in: The Logic Programming Paradigm, K. R. Apt, V. W. Marek, M. Truszezynski, and D. S. Warren, eds., Springer-Verlag, pp. 257-276.

P. VAN HENTENRYCK

[1997] Numerica: A Modeling Language for Global Optimization, The MIT Press.

P. VAN HENTENRYCK

[1999] The OPL Optimization Programming Language, The MIT Press.

P. VAN HENTENRYCK, L. PERRON, AND J.-F. PUGET

[2000] Search and strategies in OPL, ACM Transactions on Computational Logic, 1, pp. 285-320.

P. VAN HENTENRYCK AND V. SARASWAT

[1996] Strategic directions in constraint programming, A CM Computing Surveys, 28, pp. 701-726.

P. VAN HENTENRYCK, V. SARASWAT, AND Y. DEVILLE

[1998] Design, implementation, and evaluation of the constraint language cc(fd), Journal of Logic Programming, 37, pp. 139-164. Special Issue on Constraint Logic Programming (P. J. Stuckey and K. Marriot, Eds.).

M. WALLACE

[2002] Constraint logic programming, in: Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowatski, Part I, A. C. Kakas and F. Sadri, eds., vol. 2407 of Lecture Notes in Computer Science, Springer, pp. 512-532.

M. WALLACE, S. NOVELLO, AND J. SOHIMPF

[1997] BCD'PS0: A Platform for Constraint Logic Programming, tech. rep., IC-Parc, Imperial College, London. Available via http: //www. icparc. ic. ac.uk/eclipse/reports/index.html.

M. WALLACE AND J. SCHIMPF

[2002] Finding the right hybrid algorithm - a combinatorial meta-problem, Annals of Mathematics and Artificial Intelligence, 34, pp. 259-269.

T. WALSH

[2001] Relational Consistencies, Tech. Rep. APES-28-2001, APES Research Group, January. Available from http://www.dcs.st~and.ac.uk/apes/apesreports.html.

M. J. WESTER

[1999] ed., Computer Algebra Systems, John Wiley and Sons.

P. H. WINSTON

[1992] Artificial Intelligence, Addison-Wesley, third ed.

L. A. WOLSEY

[1998] Integer Programming, John Wiley and Sons.

H. ZHANG AND M. STICKEL

[1996] An efficient algorithm for unit propagation, in: Proceedings of the

Fourth International Symposium on Artificial Intelligence and Mathematics, Ft. Lauderdale, Florida.

Y. ZHANG AND R. H. C. YAP

[2001] Making AC-3 an optimal algorithm, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-01), Morgan Kaufmann, Seattle, WA, USA, pp. 316-321.