Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Συστηματων Μεταδοσης Πληροφοριας
και Τεχνολογιας Υλικων
Εργαστηριο Ευφυων Επικοινωνιων και Δικτυων Ευρειας Ζωνης

Αλγόριθμοι Σχεδίασης Δικτύων Τοπολογίας Δέντρου με Ποικίλους Περιορισμούς Χωρητικότητας

Διδακτορικη Διατριβη

Χρίστος Α. Παππάς

Αθήνα, Απρίλιος 2013

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Συστηματων Μεταδοσης Πληροφοριας
και Τεχνολογιας Υλικων
Εργαστηριο Ευφυων Επικοινωνιων και Δικτυων Ευρειας Ζωνης

Αλγόριθμοι Σχεδίασης Δικτύων Τοπολογίας Δέντρου με Ποικίλους Περιορισμούς Χωρητικότητας

Διδακτορικη Διατριβη

του

Χρίστου Α. Παππά

Συμβουλευτική Επιτροπή: Βενιέρης Ιάκωβος
Κακλαμάνη Δήμητρα-Θεοδώρα
Θεολόγου Μιχαήλ

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 22^{α} Απριλίου 2013.

.....	.....	.....
Βενιέρης Ιάκωβος	Κακλαμάνη Δήμητρα-Θεοδώρα	Θεολόγου Μιχαήλ
Καθηγητής Ε.Μ.Π.	Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.
.....	.....	.....
Παπαβασιλείου Συμεών	Ουζούνογλου Νικόλαος	Στασινόπουλος Γεώργιος
Αναπληρωτής Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

.....
Κακλαμάνης Χρήστος
Καθηγητής Πανεπιστημίου Πατρών

Αθήνα, Απρίλιος 2013

.....

Χρίστος Α. Παππάς

Διδάκτωρ Ε.Μ.Π.

Περίληψη

Κατά τη σχεδίαση κεντρικοποιημένων δικτύων συχνά προκύπτει η ανάγκη για τη εύρεση δέντρων ελάχιστου κόστους. Ένα πρόβλημα που έχει μελετηθεί εκτενώς στη βιβλιογραφία είναι το πρόβλημα εύρεσης Ελάχιστου Δέντρου Επικάλυψης με Περιορισμό Χωρητικότητας (Capacitated Minimum Spanning Tree ή CMST). Στο πρόβλημα CMST στόχος είναι να σχεδιαστεί δίκτυο τοπολογίας δέντρου ελάχιστου κόστους, το οποίο να εξυπηρετεί την προώθηση της κίνησης που παράγει ένα σύνολο τερματικών κόμβων προς ένα κεντρικό κόμβο, με τον περιορισμό η συνολική κίνηση σε οποιαδήποτε ζεύξη να μην υπερβαίνει μία ενιαία προκαθορισμένη τιμή-χωρητικότητα. Ωστόσο, κατά τη σχεδίαση πραγματικών δικτύων συχνά επιλέγεται η εγκατάσταση ζεύξεων διαφορετικών χωρητικοτήτων. Γενικεύοντας το πρόβλημα CMST, έτσι ώστε να υπάρχει η δυνατότητα επιλογής από μία γκάμα διαφορετικών τύπων ζεύξεων, οι οποίοι διαφοροποιούνται μεταξύ τους ως προς τη χωρητικότητα αλλά και το κόστος, οδηγούμαστε στο πρόβλημα εύρεσης Ελάχιστου Δέντρου Επικάλυψης με Περιορισμούς Χωρητικότητας Πολλαπλών Επιπέδων (Multi-Level Capacitated Minimum Spanning Tree ή MLCMST). Η έρευνα γύρω από το πρόβλημα MLCMST είχε μέχρι σήμερα περιοριστεί σε μία συγκεκριμένη κλάση στιγμιότυπων όπου η παραγόμενη από κάθε κόμβο κίνηση είναι μοναδιαία αλλά και η μέγιστη επιτρεπτή χωρητικότητα λαμβάνει χαμηλές τιμές.

Η παρούσα διδακτορική διατριβή έχει ως αντικείμενο τη μελέτη του προβλήματος MLCMST και την ανάδειξη αλγορίθμων που να αντιμετωπίζουν ένα ευρύ φάσμα στιγμιότυπων του. Αρχικά εξετάζεται η δυνατότητα επίλυσης προβλημάτων με τεχνικές μεικτού ακέραιου γραμμικού προγραμματισμού. Η πλήρης επίλυση των γραμμικών μοντέλων μέσα σε λογικά χρονικά πλαίσια αποδεικνύεται δυνατή μόνο για στιγμιότυπα περιορισμένου μεγέθους. Σε μεγαλύτερα προβλήματα, και δεδομένου ότι το πρόβλημα MLCMST ανήκει στη κλάση των NP-complete προβλημάτων, η προσπάθεια αναπόφευκτα μετατοπίζεται στην εύρεση ποιοτικών, αλλά όχι απαραίτητα βέλτιστων λύσεων. Βασιζόμενοι σε προηγούμενες εργασίες στον τομέα παρουσιάζουμε ευρετικούς αλγορίθμους αναβαθμίσεων, με στόχο την αντιμετώπιση στιγμιότυπων ποικίλων χαρακτηριστικών και μεγεθών. Εν συνεχεία, οι προτεινόμενοι αλγόριθμοι αναβαθμίσεων ενσωματώνονται σε αλγόριθμο Διακλάδωσης και Αποκοπής (Branch and Cut) δημοφιλούς πακέτου βελτιστοποίησης. Τέλος, εξετάζεται η εφαρμογή εξελικτικών αλγορίθμων

στο πρόβλημα. Σε αυτή την προσέγγιση οι προτεινόμενοι αλγόριθμοι αναβαθμίσεων αξιοποιούνται ως τροφοδότες λύσεων καλής ποιότητας κατά την αρχικοποίηση των πληθυσμών.

Λέξεις κλειδιά: σχεδίαση δικτύων, δέντρα επικάλυψης ελάχιστου κόστους, συνδυαστική βελτιστοποίηση, μεικτός ακέραιος γραμμικός προγραμματισμός, ευρετικοί αλγόριθμοι, εξελικτικοί αλγόριθμοι

Abstract

Designing centralized networks often involves finding minimum cost spanning trees. One of the well-known problems that has been examined extensively in the literature is the Capacitated Minimum Spanning Tree (CMST) problem. In the CMST problem we are given a set of terminal nodes producing constant traffic that must be transferred to a central node. The goal is to design a minimum cost tree network where the flow on each link shall not exceed a predefined capacity. However, in many real world cases links of different capacities may be provided. A generalization of the CMST problem which considers a set of different types of links, each with its own capacity and cost, has been introduced as the Multi-Level Capacitated Minimum Spanning Tree (MLCMST) problem. To this day, research on the MLCMST problem has been restricted to a specific class of instances involving unary traffic demands and low maximum capacity values.

The goal of the present thesis is to study the MLCMST problem and suggest algorithms addressing a variety of problem instances. At first, mixed integer linear programming techniques are applied and restricted size instances are solved to optimality within reasonable time limits. Since the MLCMST problem is NP-complete, the effort regarding larger problems is shifted to finding good, but not necessarily optimal, solutions. Based on work previously done in the field, upgrade heuristics are presented, addressing a variety of instances. In addition, the proposed heuristics are integrated with the Branch and Cut algorithm of a popular optimization package. Finally, the application of evolutionary algorithms on the MLCMST problem is examined. Heuristic solutions are used as seeds at the initialization phase of the evolutionary populations.

Keywords: network design, minimum cost spanning trees, combinatorial optimization, mixed integer linear programming, heuristics, evolutionary algorithms

Ευχαριστίες

Με το κείμενο της παρούσας διατριβής ολοκληρώνεται μία ερευνητική προσπάθεια περίπου έξι ετών. Στο τέλος της πορείας αυτής, θα ήθελα να ευχαριστήσω τον κ. Ιάκωβο Βενιέρη, καθηγητή ΕΜΠ, επιβλέποντά μου κατά την εκπόνηση της διδακτορικής έρευνας, για την εμπιστοσύνη που μου έδειξε και την καθοδήγησή του κατά τις ερευνητικές μου δραστηριότητες προς την περάτωση της διατριβής. Θα ήθελα ακόμα να ευχαριστήσω την κα. Δήμητρα Θεοδώρα Κακλαμάνη, καθηγήτρια ΕΜΠ, και τον κ. Μιχαήλ Θεολόγου, καθηγητή ΕΜΠ, μέλη της τριμελούς συμβουλευτικής επιτροπής. Επίσης, ένα μεγάλο ευχαριστώ οφείλω στους φίλους και συνοδοιπόρους όλα αυτά τα χρόνια στο Εργαστήριο Ευφυών Επικοινωνιών και Δικτύων Ευρείας Ζώνης, καθώς επίσης και στην οικογένεια μου για την αγάπη και τη στήριξη τους.

Η παρούσα διδακτορική διατριβή υποστηρίχθηκε οικονομικά από τη Συγκλητική Επιτροπή Βασικής Έρευνας του ΕΜΠ μέσω του έργου ''Ανάπτυξη Μεταευρετικών Αλγορίθμων για τη Βέλτιστη Σχεδίαση Ευρυζωνικών Δικτύων Επικοινωνιών'' στο πλαίσιο του Προγράμματος Ενίσχυσης Βασικής Έρευνας 2009 για το χρονικό διάστημα 12/2009 - 11/2011.

Περιεχόμενα

Σχήματα

Πίνακες

Αλγόριθμοι

Συντομογραφίες

B&B	Branch and Bound
B&C	Branch and Cut
CIMCF	Capacity Indexed Multi Commodity Formulation
CISCF	Capacity Indexed Single Commodity Formulation
CMST	Capacitated Minimum Spanning Tree
DCMST	Degree Constrained Minimum Spanning Tree
EA	Evolutionary Algorithm
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
LP	Linear Programming
MCF	Multi Commodity Formulation
MES	Multiple Exchanges Search
MILP	Mixed Integer Linear Programming
MLCMST	Multi Level Capacitated Minimum Spanning Tree
OCSTP	Optimum Commmunication Spanning Tree
PSO	Particle Swarm Optimization
QMST	Quadratic Minimum Spanning Tree
RSMES	Randomized Start Multiple Exchanges Search
SCF	Single Commodity Formulation
UH	Upgrade Heuristic

Κεφάλαιο 1

Εισαγωγή

1.1 Ορισμός του προβλήματος

Ένα πρόβλημα ελάχιστου δέντρου επικάλυψης που έχει εξεταστεί αναλυτικά στη βιβλιογραφία είναι το πρόβλημα Ελάχιστου Δέντρου Επικάλυψης με Περιορισμό Χωρητικότητας (Capacitated Minimum Spanning Tree, CMST) (Esau and Williams, 1966). Στο πρόβλημα CMST δίνεται ένας γράφος $G = (V, E)$ όπου μία από τις κορυφές ορίζεται ως ρίζα r και για κάθε ακμή $\{i, j\} \in E$ ορίζεται ένα κόστος C_{ij}. Το ζητούμενο είναι να βρεθεί το δέντρο ελαχίστου κόστους $T = (V, E_T)$ με μοναδικό περιορισμό το πλήθος των κόμβων σε κάθε υποδέντρο του T να μην υπερβαίνει ένα όριο Z. Γενικεύοντας, μπορούν να αντιστοιχιστούν βάρη F_i σε κάθε κορυφή $i \in V \setminus \{r\}$, οπότε πλέον ο περιορισμός επιβάλλεται στο άθροισμα των βαρών των κόμβων κάθε υποδέντρου.

Μπορούμε να θεωρήσουμε το πρόβλημα CMST ως πρόβλημα σχεδίασης δικτύου, το οποίο να εξυπηρετεί την προώθηση της κίνησης (F_i) που παράγει ένα σύνολο τερματικών κόμβων ($i \in V \setminus \{r\}$) προς ένα κεντρικό κόμβο (r), με τον περιορισμό η συνολική κίνηση σε κάθε ζεύξη να μην υπερβαίνει μία προκαθορισμένη χωρητικότητα (Z). Θεωρώντας ότι η εγκατάσταση ζεύξης μεταξύ δύο κόμβων i, j κοστίζει C_{ij}, στόχος είναι η εύρεση της τοπολογίας δέντρου με το ελάχιστο συνολικό κόστος ζεύξεων. Στην ουσία η διατύπωση αυτή αντιμετωπίζει το CMST ως ένα πρόβλημα σχεδίασης ενός δικτύου με πολλές πηγές (τερματικοί κόμβοι) και έναν παραλήπτη (κεντρικός κόμβος). Μπορούμε όμως να διατυπώσουμε το CMST και ως πρόβλημα με μία πηγή και πολλούς παραλήπτες αν αντιστρέψουμε του ρόλους, δηλαδή να θεωρήσουμε ότι ο κεντρικός κόμβος παράγει κίνηση την οποία πρέπει να κατευθύνουμε προς τους τερματικούς κόμβους.

Ωστόσο, σε πρακτικά προβλήματα πολύ συχνά καλούμαστε να επιλέξουμε από μία γκάμα χωρητικοτήτων. Μία φυσική λοιπόν επέκταση του προβλήματος CMST είναι να θεωρήσουμε ότι για

κάθε ζεύξη του δικτύου υπάρχει δυνατότητα επιλογής του τύπου της ζεύξης, από ένα σύνολο διαθέσιμων τύπωνν, οι οποίοι διαφέρουν τόσο σε χωρητικότητα όσο και κόστος. Το πρόβλημα αυτό παρουσιάστηκε από τους Gamvros et al. (2002) ως πρόβλημα Ελάχιστου Δέντρου Επικάλυψης με Περιορισμούς Χωρητικότητας Πολλαπλών Επιπέδων (Multi-Level Capacitated Minimum Spanning Tree ή MLCMST).

Πρόβλημα 1.1: Το πρόβλημα MLCMST

Δίνεται κατευθυνόμενος γράφος $G(V, E)$ με σύνολο κόμβων V και σύνολο ακμών E. Ένας κόμβος, έστω r, ορίζεται ως κεντρικός κόμβος ενώ οι υπόλοιποι ως τερματικοί κόμβοι. Κάθε τερματικός κόμβος $i \in V \setminus \{r\}$ παράγει κίνηση (ακέραιου) όγκου $F_i > 0$ με προοριζόμενη προς τον κεντρικό κόμβο r. Επίσης, δίνεται σύνολο τύπων ζεύξεων $\Lambda = \{1, 2, \ldots, L\}$ με χωρητικότητες $Z^1 < Z^2 < \ldots < Z^L$. Μεταξύ ενός ζεύγους $(i, j) \in E$ μπορεί να εγκατασταθεί ζεύξη, έστω τύπου $l \in \Lambda$. με κόστος C_{ij}^l. Ζητούμενο είναι να σχεδιαστεί δίκτυο δεντρικής τοπολογίας ελάχιστου κόστους που να διοχετεύει την κίνηση προς το κεντρικό κόμβο. Η ροή κίνησης σε κάθε ζεύξη θα πρέπει να μην υπερβαίνει τη χωρητικότητα του εγκατεστημένου τύπου.

Στη γενικότερη των περιπτώσεων το σύνολο E περιλαμβάνει όλα τα πιθανά διατεταγμένα ζεύγη κόμβων με την προϋπόθεση $i \neq j$. Επίσης, εφόσον ο κεντρικός κόμβος συγκεντρώνει όλες τις ροές δεν έχει νόημα να ορίσουμε ζεύξεις εξερχόμενες από αυτόν. Συνεπώς,

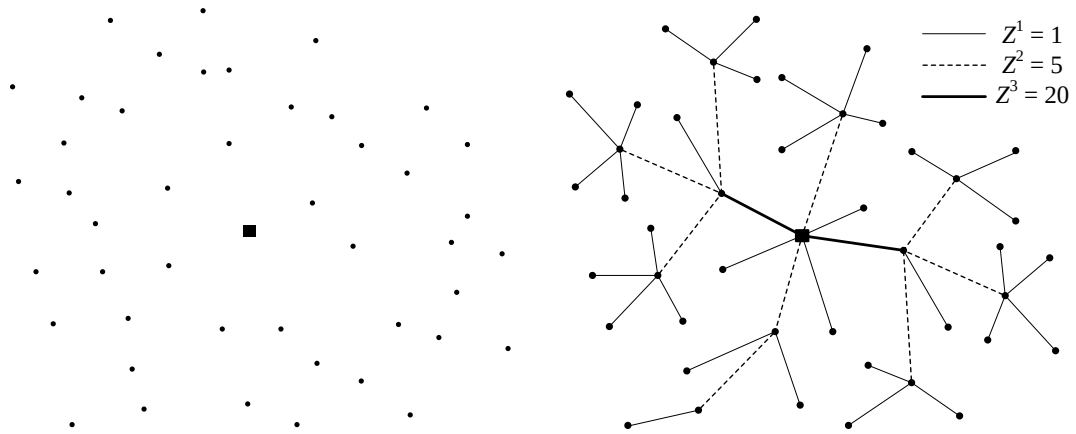
$$E \subseteq \{(i, j) : i \in V, j \in V \setminus \{i, r\}\}. \tag{1.1}$$

Ο παραπάνω ορισμός του προβλήματος δεν προσδιορίζει τη φύση της συνάρτησης κόστους C_{ij}^l. Ωστόσο, η εφαρμογή του προβλήματος MLCMST στη σχεδίαση πραγματικών δικτύων υπαγορεύει συγκεκριμένους περιορισμούς. Καταρχήν, είναι ρεαλιστικό να υποθέσουμε ότι οι τύποι ζεύξεων μεγαλύτερης χωρητικότητας θα έχουν και υψηλότερο κόστος. Αυτό συνεπάγεται ότι για κάθε διατεταγμένο ζεύγος κόμβων $(i, j) \in E$ ισχύει:

$$C_{ij}^1 < C_{ij}^2 < \ldots < C_{ij}^L. \tag{1.2}$$

Επίσης, συνήθως οι συναρτήσεις κόστους τηρούν αυστηρές οικονομίες κλίμακας. Με άλλα λόγια, καθώς μεταβαίνουμε σε τύπους ζεύξεων μεγαλύτερης χωρητικότητας το ανά μονάδα χωρητικότητας κόστος μειώνεται. Συνεπώς είναι λογικό να υποθέσουμε ότι

$$\frac{C_{ij}^1}{Z^1} \geq \frac{C_{ij}^2}{Z^2} \geq \ldots \geq \frac{C_{ij}^L}{Z^L}. \tag{1.3}$$



(α΄) Σύνολο κόμβων *(β΄) Εφικτή λύση*

Σχήμα 1.1: Ελάχιστο Δέντρου Επικάλυψης με Περιορισμούς Χωρητικότητας Πολλαπλών Επιπέδων

Στη βιβλιογραφία έχουν μελετηθεί στιγμιότυπα του προβλήματος MLCMST στα οποία η κίνηση F_i που παράγει κάθε τερματικός κόμβος είναι ίση με 1. Θα αναφερόμαστε στην ειδική αυτή περίπτωση τους προβλήματος ως *μοναδιαία*. Αντίθετα, στιγμιότυπα για τα οποία δεν ισχύει κατ ανάγκην ο παραπάνω περιορισμός θα αναφέρονται ως *μη μοναδιαία*.

1.2 Δυσκολία του προβλήματος MLCMST

Το πρόβλημα MLCMST ανήκει στην ευρύτερη κατηγορία προβλημάτων συνδυαστικής βελτιστοποίησης (combinatorial optimization), δηλαδή προβλημάτων εύρεσης της βέλτιστης λύσης από ένα πεπερασμένο, συνήθως πολύ μεγάλο όμως, σύνολο εφικτών λύσεων. Για το πρόβλημα MLCMST ειδικότερα, ο χώρος λύσεων ισούται με το σύνολο των δέντρων επικάλυψης (spanning trees) που μπορούν να οριστούν πάνω στο σύνολο των κόμβων. Ο Cayley (1889) απέδειξε ότι ο αριθμός δέντρων επικάλυψης ενός πλήρη γράφου με n κόμβους είναι $n^{(n-2)}$. Το γεγονός αυτό καθιστά την εύρεση της βέλτιστης λύσης μέσω εξαντλητικής αναζήτησης πρακτικά εφικτή μόνο για πάρα πολύ μικρά δίκτυα.

Επίσης, ο Papadimitriou (1978) έδειξε ότι το πρόβλημα CMST ανήκει στην κατηγορία των NP-complete προβλημάτων. Όμως, οποιοδήποτε στιγμιότυπο του προβλήματος CMST μπορεί να εκφραστεί και ως στιγμιότυπο του MLCMST με ένα μόνο διαθέσιμο τύπο ζεύξεων. Με άλλα λόγια το πρόβλημα CMST *ανάγεται* στο πρόβλημα MLCMST. Αυτό συνεπάγεται ότι το πρόβλημα MLCMST είναι τουλάχιστον τόσο δύσκολο όσο το CMST και άρα είναι επίσης NP-complete.

Μία ειδική κατηγορία προβλημάτων συνδυαστικής βελτιστοποίησης, στην οποία ανήκει και το πρόβλημα MLCMST, αποτελούν τα προβλήματα εκείνα που μπορούν να διατυπωθούν ως προβλήματα

μεικτού ακεραίου γραμμικού προγραμματισμού (mixed integer linear problems, MILP). Για τα προβλήματα αυτά έχουν αναπτυχθεί τεχνικές που επιτυγχάνουν χρόνους επίλυσης πολύ χαμηλότερους από την εξαντλητική αναζήτηση. Κοινό χαρακτηριστικό αυτών των αλγορίθμων είναι ο εντοπισμός υποσυνόλων του χώρου λύσεων που περιλαμβάνουν αποκλειστικά μη-βέλτιστες λύσεις. Τα συγκεκριμένα υποσύνολα δεν χρειάζεται να εξερευνηθούν και συνεπώς επιτυγχάνεται εξοικονόμηση υπολογιστικού χρόνου.

Ωστόσο, η πλήρης επίλυση των MILP μοντέλων μέσα σε λογικά χρονικά πλαίσια συνήθως επιτυγχάνεται μόνο για στιγμιότυπα περιορισμένου μεγέθους. Σε μεγαλύτερα στιγμιότυπα η προσπάθεια αναπόφευκτα επικεντρώνεται στην εύρεση καλών, αλλά όχι απαραίτητα ολικά βέλτιστων, λύσεων κυρίως μέσω ευρετικών αλγορίθμων (heuristics). Μπορούμε να διαχωρίσουμε τους ευρετικούς αλγορίθμους σε δύο κύριες κατηγορίες: σε αυτούς που έχουν σχεδιαστεί ειδικά για το συγκεκριμένο πρόβλημα (problem-specific heuristics) και στους μεταευρετικούς (metaheuristics). Οι μεταευρετικοί είναι γενικές στρατηγικές αντιμετώπισης προβλημάτων βελτιστοποίησης και συνήθως η εφαρμογή τους απαιτεί λίγες ή καθόλου πληροφορίες για το πρόβλημα που βελτιστοποιείται.

Τέλος, ιδιαίτερα χρήσιμη είναι συχνά η εύρεση καλών κάτω ορίων των βέλτιστων λύσεων. Τα κάτω όρια χρησιμοποιούνται στην αξιολόγηση εφικτών λύσεων παρέχοντάς μας μία εκτίμηση της απόστασης τους από τις βέλτιστες. Ένας συνήθης τρόπος εύρεσης κάτω ορίου σε ένα πρόβλημα MILP είναι μέσω της επίλυσης της γραμμικής χαλάρωσής του.

1.3 Συναφή προβλήματα

Η κατασκευή δέντρου με περιορισμούς χωρητικότητας ανακύπτει στο πλαίσιο σχεδίασης κεντρικοποιημένων δικτύων. Σε αυτά τα προβλήματα στόχος είναι η εύρεση δέντρων επικάλυψης ελαχίστου κόστους έτσι ώστε ο όγκος κίνησης σε κάθε ακμή να περιορίζεται από χωρητικότητες. Οι Esau and Williams (1966); Rothfarb and Goldstein (1971); Gavish (1982, 1991) αναφέρουν εφαρμογές σε δίκτυα επικοινωνιών.

Το πρόβλημα CMST παρουσιάστηκε από τους Esau and Williams (1966) ενώ ο Papadimitriou (1978) έδειξε ότι ανήκει στην κατηγορία των NP-Hard προβλημάτων. Αλγόριθμοι για το πρόβλημα CMST έχουν παρουσιαστεί από τους Gavish (1982, 1983, 1991); Gouveia (1993, 1995); Hall (1996); Amberg et al. (1996); Sharaiha et al. (1997); Gouveia and Martins (2000, 2005); de Souza et al. (2004); Martins (2007); Uchoa et al. (2007). Ο Gavish (1982, 1991) περιγράφει μία παραλλαγή του προβλήματος CMST όπου ο περιορισμός χωρητικότητας είναι διαφορετικός για κάθε ζευγάρι κόμβων.

Ένα πρόβλημα στενά συνδεδεμένο με το MLCMST είναι το πρόβλημα local access network design (LAND) (Berger et al., 2000), το οποίο είχε παρουσιαστεί και νωρίτερα από τους Salman et al. (1997) ως πρόβλημα single-sink buy-at-bulk network design. Το πρόβλημα LAND διαφοροποιείται σε σχέση με το MLCMST στο ότι δεν περιορίζεται σε λύσεις-δέντρα και επίσης επιτρέπει την τοποθέτηση πολλαπλών τύπων σε μία ζεύξη. Αλγόριθμοι για το πρόβλημα LAND έχουν παρουσιαστεί από τους Salman et al. (1997, 2000, 2008); Berger et al. (2000); Garg et al. (2001)

Προβλήματα δέντρων επικάλυψης ελαχίστου κόστους τα οποία έχουν αντιμετωπιστεί με εξελικτικούς αλγορίθμους παρατίθενται στην εισαγωγή του κεφαλαίου 6.

1.4 Διάρθρωση της διατριβής

Η παρούσα διατριβή αποτελείται από συνολικά επτά κεφάλαια. Πέρα από το παρόν εισαγωγικό κεφάλαιο, το περιεχόμενο των υπολοίπων κεφαλαίων συνοψίζεται ως εξής.

- Στο δεύτερο κεφάλαιο αρχικά γίνεται μία αναδρομή στους αλγορίθμους που έχουν προταθεί μέχρι σήμερα για το πρόβλημα MLCMST. Εν συνεχεία, αναλύονται οι ιδιαιτερότητες του στιγμιότυπων που έχουν εξεταστεί στη βιβλιογραφία αλλά και η αδυναμία των αλγορίθμων αυτών να αντιμετωπίσουν γενικότερα στιγμιότυπα. Τέλος, περιγράφεται η κατασκευή νέων στιγμιότυπων αναφοράς καλύπτοντας ένα ευρύτερο φάσμα περιπτώσεων, τα οποία θα αποτελέσουν και τη βάση για την αξιολόγηση των αλγορίθμων που προτείνονται στα υπόλοιπα κεφάλαια της διατριβής.

- Το τρίτο κεφάλαιο επικεντρώνεται στην προσέγγιση του προβλήματος MLCMST ως πρόβλημα μεικτού ακέραιου γραμμικού προγραμματισμού (MILP). Αφού περιγραφούν συνοπτικά οι τεχνικές αντιμετώπισης προβλημάτων MILP αναλύονται οι διαφορετικές γραμμικές διατυπώσεις του προβλήματος. Ακολούθως παρουσιάζονται αριθμητικά αποτελέσματα από την επίλυση των γραμμικών μοντέλων με χρήση δημοφιλούς πακέτου λογισμικού βελτιστοποίησης. Τα αποτελέσματα επικεντρώνονται στην κλιμάκωση των χρονικών απαιτήσεων για την πλήρη επίλυση των στιγμιότυπων αναφοράς. Επίσης, εξετάζεται η δυνατότητα εξαγωγής κάτω ορίων καλής ποιότητας από τις γραμμικές χαλαρώσεις των μοντέλων.

- Στο τέταρτο κεφάλαιο παρουσιάζονται οι προτεινόμενοι ευρετικοί αλγόριθμοι αναβαθμίσεων. Οι αλγόριθμοι αυτοί βασίζονται σε διαδοχικές βελτιώσεις των λύσεων μέσω της αναβάθμισης του ρόλου κάποιων κόμβων. Η ιδέα των αναβαθμίσεων προκύπτει από αλγόριθμο της βιβλιογραφίας, ο οποίος όμως περιοριζόταν στη μοναδιαία περίπτωση. Αφού επεκτείνουμε το αλγόριθμο της βιβλιογραφίας ώστε να μπορεί να εφαρμοστεί και σε μη μοναδιαία στιγμιότυπα, παρουσιάζουμε

δύο ακόμα αλγορίθμους με στόχο την εύρεση καλύτερων λύσεων, ειδικά σε περιπτώσεις όπου η μέγιστη προσφερόμενη χωρητικότητα παίρνει μεγάλες τιμές. Στα πλαίσια των αλγορίθμων προκύπτει ένα υποπρόβλημα, αυτό της βέλτιστης επιλογής των κόμβων που θα επανασυνδεθούν σε έναν αναβαθμιζόμενο κόμβο. Αναλύονται οι διαφορετικές εκδοχές του προβλήματος και προτείνονται διαδικασίες δυναμικού προγραμματισμού που τις επιλύουν.

- Στο πέμπτο κεφάλαιο παρουσιάζεται η ενσωμάτωση των αναβαθμίσεων κόμβων στον αλγόριθμο Οριοθέτησης και Αποκοπής (Branch & Cut, B&C) δημοφιλούς πακέτου λογισμικού. Οι αλγόριθμοι αναβαθμίσεων μετασχηματίζονται κατάλληλα ώστε να μπορούν να εκτελεστούν ως ευρετικές διαδικασίες σε κάθε κόμβο του δέντρου Οριοθέτησης και Αποκοπής. Κατ' αυτόν τον τρόπο επιχειρείται η ενίσχυση της δυνατότητας του αλγορίθμου B&C να ανακαλύπτει εφικτές λύσεις καλής ποιότητας.

- Στο έκτο κεφάλαιο εξετάζεται η δυνατότητα να αντιμετωπιστεί το πρόβλημα MLCMST με Εξελικτικούς Αλγορίθμους (Evolutionary Algorithms). Εξετάζονται εξελικτικές διαδικασίες και αναπαραστάσεις δέντρων που έχουν εφαρμοστεί με επιτυχία σε συναφή προβλήματα εύρεσης δέντρων επικάλυψης ελαχίστου κόστους (minimum cost spannning tree problems). Οι ευρετικές λύσεις αναβαθμίσεων αξιοποιούνται στην αρχικοποίηση του πληθυσμού.

- Το έβδομο και τελευταίο κεφάλαιο αποτελεί τον επίλογο της διατριβής, όπου υπογραμμίζονται κάποια βασικά συμπεράσματα και σκιαγραφούνται κάποιες προτάσεις για μελλοντική έρευνα στο συγκεκριμένο ερευνητικό πεδίο.

Κεφάλαιο 2

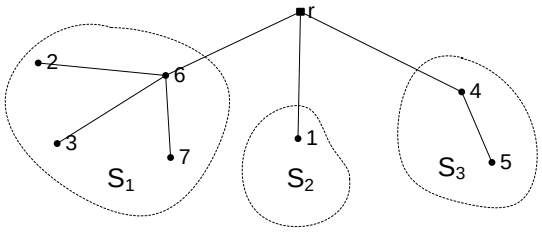
Αλγόριθμοι για το πρόβλημα MLCMST στη βιβλιογραφία

2.1 Ορισμοί

Προτού παρουσιάσουμε τους ευρετικούς αλγορίθμους που έχουν προταθεί στη βιβλιογραφία για το πρόβλημα MLCMST, εισάγουμε κάποιους βασικούς ορισμούς για την καλύτερη κατανόηση τους.

Θεωρούμε την κάθε εφικτή λύση του προβλήματος MLCMST ως ένα κατευθυνόμενο δέντρο (arborescence) $T(V, E_T)$ όπου οι ζεύξεις-ακμές κατευθύνονται προς τον κεντρικό κόμβο r. Επίσης θα συμβολίζουμε με V^* το σύνολο των τερματικών κόμβων, δηλαδή $V^* = V \setminus \{r\}$. Για κάθε $i \in V^*$ ορίζουμε τα εξής:

- p_i: ο γονέας του i. Αυτός είναι ο μόνος κόμβος $j \in V$ για τον οποίο ισχύει $(i, j) \in E_T$.
- P_i: το σύνολο των τερματικών κόμβων που περιλαμβάνονται στο μονοπάτι από τον κόμβο i προς τον κεντρικό κόμβο r. Το σύνολο αυτό περιλαμβάνει τον κόμβο i αλλά δε περιλαμβάνει τον κεντρικό κόμβο (αφού δεν είναι τερματικός). Το σύνολο P_i μπορεί να οριστεί αναδρομικά ως εξής: $P_i = \begin{cases} \{i\} & \text{if } p_i = r \\ \{i\} \cup P_{p_i} & \text{else} \end{cases}$
- V_i: αποτελείται από τους κόμβους που περιλαμβάνονται στο υποδέντρο που ορίζει ο i, δηλαδή τον κόμβο i, τα παιδιά του i, τα παιδιά των παιδιών τους, και ούτω καθεξής. $V_i = \{j \in V : i \in P_j\}$
- t_i: ο όγκος της κίνησης που διασχίζει τη ζεύξη (i, p_i). Αυτός ισούται με το σύνολο της κίνησης που παράγεται στο υποδέντρο που ορίζει ο i, δηλαδή $t_i = \sum_{j \in V_i} F_j$.
- λ_i: ο τύπος της ζεύξης (i, p_i). Για να είναι εφικτή μία λύση θα πρέπει για κάθε τερματικό κόμβο να ισχύει ο περιορισμός χωρητικότητας $t_i \leq Z^{\lambda_i}$.



Σχήμα 2.1: Παράδειγμα διαμερισμού με βάση τα κύρια υποδέντρα

Με τον όρο *κόμβος-φύλλο* αναφερόμαστε στο υποσύνολο των κόμβων, οι οποίοι στη συγκεκριμένη λύση δεν έχουν απογόνους και θα το συμβολίζουμε με V_S. Για ένα κόμβο-φύλλο i ισχύουν $V_i = \{i\}$ και $t_i = F_i$.

Έστω $i_1, i_2, \ldots, i_K$ οι K απευθείας απόγονοι του r σε μία λύση T. Τότε τα σύνολα $\{S_k = V_{i_k}\}_{k=1,\ldots,K}$ αποτελούν διαμερισμό (partition) του συνόλου V^* των τερματικών κόμβων. Επίσης, έστω J_k ο υπογράφος του T που αποτελείται από τους κόμβους $S_k \cup \{r\}$ και τις μεταξύ τους ζεύξεις στο T. Ο J_k είναι δέντρο και θα αναφερόμαστε σε αυτό ώς το k-στο *κύριο υποδέντρο* του T. Για ένα τερματικό κόμβο i θα συμβολίζουμε με $J[i]$ το κύριο υποδέντρο στο οποίο ανήκει και $S[i]$ τον αντίστοιχο σύνολο τερματικών κόμβων. Το σχήμα 2.1 απεικονίζει ένα παράδειγμα διαμερισμού με βάση τα κύρια υποδέντρα.

Τέλος, για τις ανάγκες σύγκρισης λύσεων χρησιμοποιείται συχνά ως μετρική η μέση σχετική απόσταση (relative gap), ή αλλιώς ποσοστιαία απόσταση (percentage gap), και η μέση τιμής της. Έστω f και g οι τιμές κόστους δύο λύσεων. Η σχετική απόσταση του f από το g ορίζεται ως $gap(f, g) = \frac{f-g}{g}$ και θα εκφράζεται ως ποσοστό %. Έστω $\mathbf{f} = (f_1, f_2 \ldots, f_n)$ και $\mathbf{g} = (g_1, g_2 \ldots, g_n)$ διανύσματα όπου το καθένα περιλαμβάνει τιμές κόστους λύσεων για μία σειρά από n διαφορετικά στιγμιότυπα. Η μέση ποσοστιαία διαφορά του $\mathbf{f}$ από το $\mathbf{g}$ ορίζεται ως

$$gap(\mathbf{f}, \mathbf{g}) = \frac{\sum_{i=1}^{n} {}^{(f_i - g_i)}/_{g_i}}{n}.$$

και θα εκφράζεται επίσης ως ποσοστό %.

2.2 Ευρετικός αλγόριθμος αναβάθμισης ζεύξεων των Gamvros et al. (2002, 2006)

Οι Gamvros et al. (2002, 2006) παρουσίασαν έναν ευρετικό αλγόριθμο για τη μοναδιαία περίπτωση του MLCMST. Ο αλγόριθμος χαρακτηρίζεται ως ευρετικός αλγόριθμος *εξοικονομήσεων* (savings), καθότι βασίζεται στην πραγματοποίηση διαδοχικών τροποποιήσεων του δέντρου, όπου κάθε τροπο-

ποίηση μειώνει περαιτέρω το συνολικό κόστος του δέντρου. Οι εξοικονομήσεις κόστους επιτυγχάνονται μέσω *αναβαθμίσεων ζεύξεων*.

Η διαδικασία αναβάθμισης μίας ζεύξης περιλαμβάνει καταρχήν την αύξηση της χωρητικότητας μίας υπάρχουσας ζεύξης αλλάζοντας τον τύπο της ζεύξης. Προφανώς μία τέτοια ενέργεια αυξάνει το συνολικό κόστος του δέντρου. Έστω (i, p_i) η αναβαθμιζόμενη ζεύξη μίας εφικτής λύσης, τότε η μετατροπή της ζεύξης σε τύπου l αυξάνει το συνολικό κόστος του δέντρου κατά $C^l_{ip_i} - C^{\lambda_i}_{ip_i}$. Όμως πλέον στη ζεύξη (i, p_i) υπάρχει πλεονάζουσα χωρητικότητα (τουλάχιστον $Z^l - Z^{\lambda_i}$), η οποία μπορεί να χρησιμοποιηθεί για τη δρομολόγηση κίνησης άλλων τερματικών κόμβων-φύλλων του δικτύου. Για ένα κόμβο-φύλλο $j \in V_S$ η αποσύνδεση από τον πατέρα του p_j και η επανασύνδεση του στον i προκαλεί μεταβολή κόστους ίση με $d_{ij} = C^{\lambda_j}_{ji} - C^{\lambda_j}_{jp_j}$ και είναι επικερδής αν $d_{ij} < 0$. Συνεπώς, η επανασύνδεση ενός συνόλου κόμβων $H \subseteq V_S$ επιφέρει κέρδη ίσα με $\sum_{j \in H} d_{ij}$. Ωστόσο, το σύνολο περιορίζεται από τους περιορισμούς χωρητικότητας των ζεύξεων στο μονοπάτι P_i. Σύμφωνα με τα παραπάνω, αν επιλέξουμε το βέλτιστο σύνολο $H \subseteq V_S$ η συνολική μεταβολή κόστους από μία αναβάθμιση είναι:

$$D^l_i = C^l_{ip_i} - C^{\lambda_i}_{ip_i} + \min_{\substack{\{H: H \subset S,\\ d_{ij}<0 \quad \forall j \in H,\\ t'_m \le Z^\lambda_i \quad \forall m \in P_i,\\ t'_i \le Z^l\}}} \sum_{j \in H} d_{ij},$$

όπου t'_m η κίνηση κατά μήκος της ζεύξης (m, p_m) μετά την αναβάθμιση. Αν $D^l_i < 0$ τότε η αναβάθμιση είναι επικερδής.

Ο αλγόριθμος των Gamvros et al. (2002, 2006), στο εξής GUH (Gamvros' Upgrade Heuristic), εκκινεί από ένα δέντρο τοπολογίας αστέρα, όπου κάθε τερματικός κόμβος συνδέεται με τον κεντρικό κόμβο με ζεύξη όσο το δυνατόν μικρότερης δυνατής χωρητικότητας. Με βάση τους ορισμούς που δώσαμε στην αρχή του κεφαλαίου, για κάθε τερματικό κόμβο $i \in V^*$ μίας αρχικής εφικτής λύσης $T^0(V, E_{T^0})$ τοπολογίας αστέρα ισχύουν τα εξής: $p^0_i = r$, $P^0_i = \{i\}$, $V^0_i = \{i\}$, $t^0_i = F_i$. Στη συνέχεια ο αλγόριθμος υπολογίζει για κάθε τερματικό κόμβο i τη μεταβολή το κόστους D^l_i για την περίπτωση που ο κόμβος i αναβαθμιστεί σε ζεύξη μέγιστης χωρητικότητας (τύπου $l = L$). Αφού εξεταστούν όλες αναβαθμίσεις προς τον τύπο L για όλους τους τερματικούς κόμβους, επιλέγεται η πιο επικερδής αναβάθμιση και υλοποιείται. Η διαδικασία εύρεσης αναβαθμίσεων και υλοποίησης της πιο επικερδούς επαναλαμβάνεται εξαιρώντας κάθε φορά τους κόμβους που αναβαθμίζονται από περαιτέρω αναβαθμίσεις. Όταν πλέον δε μπορούν να βρεθούν άλλες επικερδείς αναβαθμίσεις προς τον τύπο L, ο αλγόριθμος συνεχίζει αναζητώντας αναβαθμίσεις προς τον τύπο $L-1$, έπειτα προς τον τύπο $L-2$ κ.ο.κ. Ο αλγόριθμος τερματίζει όταν εξεταστούν όλοι οι τύποι ζεύξεων.

Ο περιορισμός του αλγορίθμου GUH σε προβλήματα μοναδιαίων απαιτήσεων κίνησης έγκειται στη διαδικασία προσδιορισμού του βέλτιστου συνόλου προς επανασύνδεση κόμβων H. Στην περίπτωση

αυτή η κίνηση που φέρουν οι κόμβοι $j \in V_S$ είναι $t_j = F_j = 1$, με άλλα λόγια η ενσωμάτωση οποιουδήποτε κόμβου $j \in V_S$ στο σύνολο H μειώνει κατά μία μονάδα τη διαθέσιμη χωρητικότητα στην αναβαθμιζόμενη ζεύξη. Συνεπώς, για τη εύρεση του βέλτιστου συνόλου αρκεί απλά να ταξινομήσουμε αρχικά τους κόμβους $j \in V_S$ με βάση το κέρδος επανασύνδεσης d_{ij}, και έπειτα με βάση τη σειρά ταξινόμησης να ελέγχουμε και εισάγουμε διαδοχικά κόμβους στο σύνολο H μέχρι η διαθέσιμη χωρητικότητα να εξαντληθεί. Κατά τον έλεγχο εισαγωγής ενός κόμβου στο σύνολο H, σιγουρευόμαστε ότι δεν παραβιάζεται κάποιος περιορισμός χωρητικότητας στο μονοπάτι P_i.

2.3 Αλγόριθμοι τοπικής αναζήτησης με πολλαπλές ανταλλαγές κόμβων των Gamvros et al. (2006)

Οι Gamvros et al. (2006) παρουσίασαν ένα μηχανισμό βελτίωσης εφικτών λύσεων του προβλήματος MLCMST μέσω *πολλαπλών ανταλλαγών κόμβων*. Στα πλαίσια του μηχανισμού ορίζονται δύο τύποι ανταλλαγών

- κυκλικές ανταλλαγές (cyclic exchanges): Μία κυκλική ανταλλαγή δηλώνεται ως $i_1 - i_2 - \ldots - i_n - i_1$, όπου οι κόμβοι $i_1, i_2, \ldots, i_n$ ανήκουν σε διαφορετικά κύρια υποδέντρα, και αντιπροσωπεύει τις ακόλουθες αλλαγές: ο κόμβος i_1 μετακινείται στο κύριο υποδέντρο $J[i_2]$, ο i_2 στο $J[i_3]$, και ούτω καθεξής, και τέλος ο κόμβος i_n μετακινείται στο κύριο υποδέντρο $J[i_1]$.
- ανταλλαγές μονοπάτια (path exchanges): Μία ανταλλαγή μονοπάτι δηλώνεται ώς $i_1 - i_2 - \ldots - i_n$, όπου οι κόμβοι $i_1, i_2, \ldots, i_n$ ανήκουν σε διαφορετικά κύρια υποδέντρα. Οι αλλαγές που αντιπροσωπεύει μία ανταλλαγή μονοπάτι είναι οι ίδιες με αυτές της αντίστοιχης κυκλική ανταλλαγής με τη διαφορά ότι δεν έχουμε μετακίνηση από το κύριο υποδέντρο $J[i_n]$ προς το κύριο υποδέντρο $J[i_1]$.

Ο μηχανισμός εκκινεί από μία εφικτή λύση και συνεχίζει βρίσκοντας και υλοποιώντας την πιο επικερδή ανταλλαγή έως ότου δεν είναι δυνατόν να βρεθεί ανταλλαγή που θα επιφέρει μείωση κόστους. Για τη εύρεση της βέλτιστης ανταλλαγής σε κάθε βήμα κατασκευάζεται ένας *γράφος βελτιώσεων* (Ahuja et al., 1998, 2001).

Για την αποτίμηση του κέρδους κάθε ανταλλαγής απαιτείται ο υπολογισμός των κύριων υποδέντρων που προκύπτουν μετά τις ανταλλαγές. Δεδομένου ότι ο υπολογισμός των βέλτιστων κύριων υποδέντρων χρησιμοποιώντας μεθόδους MILP είναι υπολογιστικά χρονοβόρα, χρησιμοποιείται ο ευρετικός αλγόριθμος αναβάθμισης ζεύξεων για την κατασκευή των κύριων υποδέντρων.

Οι Gamvros et al. (2006) προτείνουν δύο αλγορίθμους που να υλοποιούν το μηχανισμό ανταλλαγών. Κατά τον πρώτο αλγόριθμο κατασκευάζεται μία αρχική λύση χρησιμοποιώντας τον αλγόριθμο GUH

και ακολούθως εφαρμόζεται ο μηχανισμός ανταλλαγών. Θα αναφερόμαστε στο εξής σε αυτόν τον αλγόριθμο ως MES (multiple exchanges search).

Κατά το δεύτερο αλγόριθμο κατασκευάζονται δέκα διαφορετικές αρχικές λύσεις οι οποίες προκύπτουν από την εφαρμογή του αλγόριθμου GUH σε *διαταραγμένες* εκδόσεις του αρχικού προβλήματος. Η διατάραξη γίνεται πολλαπλασιάζοντας τις τιμές C_{ij}^l του προβλήματος με μία τυχαία μεταβλητή ομοιόμορφα κατανεμημένη στο εύρος $[0.7, 1.3]$. Οι δέκα αρχικές λύσεις βελτιώνονται εφαρμόζοντας στην κάθε μία ξεχωριστά τον μηχανισμό ανταλλαγών και η διαδικασία επιστρέφει την καλύτερη από αυτές. Θα αναφερόμαστε στο εξής σε αυτόν τον αλγόριθμο ως RSMES (randomized start multiple exchanges search).

2.4 Υβριδικός γενετικός αλγόριθμος των Gamvros et al. (2006)

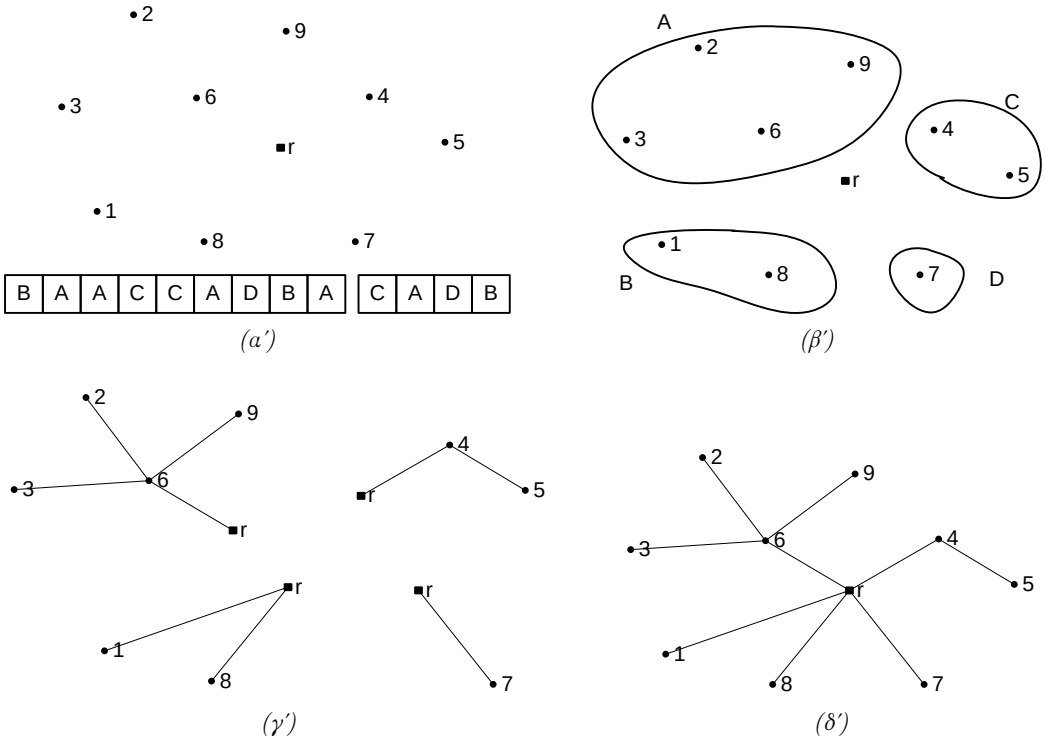
Οι Gamvros et al. (2006) παρουσίασαν έναν υβριδικό γενετικό αλγόριθμο ο οποίος διαχωρίζει το πρόβλημα MLCMST σε δύο μέρη: ένα πρόβλημα ομαδοποίησης των κόμβων στα κύρια υποδέντρα του δικτύου και στη βέλτιστη σχεδίαση των υποδέντρων.

Ο γενετικός αλγόριθμος χρησιμοποιεί μία αναπαράσταση που προτάθηκε από τον Falkenauer (1996) για προβλήματα τοποθέτησης σε κάδους (bin packing). Ένα χρωμόσωμα χωρίζεται σε δύο μέρη, το μέρος αντικειμένων (item part) και το μέρος ομάδων (group part). Το μέρος αντικειμένων έχει μήκος ίσο με το πλήθος των τερματικών κόμβων του προβλήματος. Η τιμή κάθε στοιχείου του χρωμοσώματος αντιπροσωπεύει το κύριο υποδέντρο στο οποίο ανήκει ο αντίστοιχος κόμβος. Το μέρος ομάδων έχει μεταβλητό μήκος και περιλαμβάνει μία λίστα με τα κύρια υποδέντρα του δικτύου. Η επιλογή των χαρακτήρων που χρησιμοποιούνται για να δηλωθούν οι ομάδες, καθώς και η σειρά με την οποία δηλώνονται (στο μέρος ομάδων) δεν επηρεάζουν το διαμερισμό που αναπαριστά το χρωμόσωμα. Κατά την αποκωδικοποίηση ενός χρωμοσώματος, αφού αντιστοιχιστούν οι τερματικοί κόμβοι στα σύνολα $S[i]$, εφαρμόζεται ο αλγόριθμος GUH για τα υποπροβλήματα MLCMST που ορίζουν τα σύνολα $S[i] \cup \{r\}$. Η εξαγόμενη λύση προκύπτει από την ένωση των κύριων υποδέντρων και άρα έχει κόστος ίσο με το άθροισμα του κόστους των κύριων υποδέντρων. Στο σχήμα 2.2 απεικονίζεται η διαδικασία αποκωδικοποίησης ενός χρωμοσώματος.

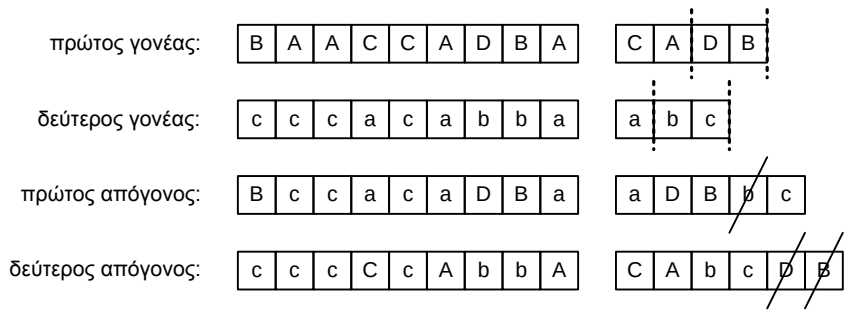
Τα βήματα του γενετικού αλγορίθμου φαίνονται στον αλγόριθμο 2.1. Για την αρχικοποίηση των διανυσμάτων χρησιμοποιούνται ο αλγόριθμος των Esau and Williams (1966) για το μισό πληθυσμό, και ο ευρετικός αλγόριθμος αναβάθμισης ζεύξεων για τον άλλο μισό. Επειδή και οι δυο αυτοί αλγόριθμοι δίνουν μοναδικές λύσεις και εν γένει είναι επιθυμητό οι πληθυσμοί να χαρακτηρίζονται

Αλγόριθμος 2.1: Βήματα του υβριδικού γενετικού αλγόριθμου

1 $t \leftarrow 0$
2 αρχικοποίηση του πληθυσμού $P(t)$
3 αποτίμηση του κόστους για κάθε χρωμόσωμα του $P(t)$
4 **repeat**
5 $\quad t \leftarrow t + 1$
6 $\quad$ επιλογή r γονέων από τον $P(t-1)$, διασταύρωση των r γονέων, και παραγωγή r απογόνων η οποίοι εισάγονται στον $P(t)$
7 $\quad$ επιλογή $(popsize - r - m)$ χρωμοσωμάτων του $P(t-1)$ και απευθείας εισαγωγή τους στον $P(t)$
8 $\quad$ εφαρμογή μετάλλαξης στα m καλύτερα χρωμοσώματα του $P(t-1)$ και εισαγωγή τους στον $P(t)$
9 $\quad$ αποτίμηση του κόστους για κάθε χρωμόσωμα του $P(t)$
10 **until** *κριτήριο τερματισμού*



Σχήμα 2.2: Αναπαράσταση ομάδων - Παράδειγμα αποκωδικοποίησης



Σχήμα 2.3: Αναπαράσταση ομάδων - Παράδειγμα εφαρμογής τελεστή αναπαραγωγής

από ποικιλομορφία, δημιουργούνται διαφορετικά διανύσματα εφαρμόζοντας τους δύο αλγόριθμους σε διαταραγμένα στιγμιότυπα του προβλήματος (βλέπε προηγούμενη ενότητα). Η διαδικασία επιλογής γίνεται με χρήση του μηχανισμού ρουλέτας (roullette-wheel selection, Michalewicz (1996)) και την εφαρμογή αποκοπής σίγμα (sigma truncation, Michalewicz (1996)) στις τιμές κόστους. Ο τελεστής μετάλλαξης που χρησιμοποιείται βελτιώνει τα χρωμοσώματα εφαρμόζοντας πολλαπλές ανταλλαγών κόμβων (βλέπε προηγούμενη ενότητα).

Για τη δημιουργία απογόνων χρησιμοποιείται ένας ειδικός τελεστής αναπαραγωγής, ο οποίος εκτελεί τα παρακάτω βήματα:

1. Διαλέγει τυχαία δύο σημεία τομής στο μέρος ομάδων των δύο χρωμοσωμάτων-γονέων.
2. Εισάγει τα τις τιμές μεταξύ των δύο σημείων τομής του πρώτου γονέα ακριβώς πριν το πρώτο σημείο τομής του δεύτερου γονέα.
3. Ενημερώνει το μέρος αντικειμένων ως εξής: Εάν το αντικείμενο ανήκει σε ομάδα του πρώτου γονέα που εισήχθη στονδεύτερο τότε αντιστοιχίζεται σε αυτήν, αλλιώς αντιστοιχίζεται στην ομάδα που ορίζεται στο δεύτερο γονέα. Εάν οδηγηθούμε σε ομάδες χωρίς αντικείμενα τότε αυτές οι ομάδες διαγράφονται
4. Εάν μία ομάδα του απογόνου έχει λιγότερα από k αντικείμενα τότε αυτά αντιστοιχίζονται σε άλλες ομάδες.

Τα παραπάνω βήματα εκτελούνται δύο φορές αντιστρέφοντας τους ρόλους των ζευγαριών κατά τη δεύτερη εκτέλεση και άρα δίνουν δύο απογόνους. Στο σχήμα 2.3 απεικονίζεται η διαδικασία αναπαραγωγής

2.5 Μέθοδος άπληστης τυχαιοποιημένης προσαρμόσιμης αναζήτησης (GRASP) των Martins et al. (2009)

Οι Martins et al. (2009) πρότειναν ένα αλγόριθμο άπληστης τυχαιοποιημένης προσαρμόσιμης αναζήτησης για το πρόβλημα MLCMST. Η μέθοδος άπληστης τυχαιοποιημένης προσαρμόσιμης αναζήτησης (greedy randomized adaptive search procedure – GRASP Feo and Resende (1989, 1995)) είναι μία ευρετική μέθοδος, η οποία βασίζεται στην εκτέλεση ενός πλήθους ανεξάρτητων επαναλήψεων. Η καλύτερη λύση μεταξύ όλων των επαναλήψεων αποτελεί την έξοδο του αλγορίθμου. Κάθε επανάληψη αποτελείται από δύο φάσεις: τη φάση κατασκευής και τη φάση τοπικής αναζήτησης. Κατά τη φάση κατασκευής δημιουργείται μία εφικτή λύση ενώ κατά τη φάση τοπικής αναζήτησης η μέθοδος εξερευνά τη γειτονιά της αρχικής λύσης μέχρι να βρεθεί τοπικό ακρότατο. Τα βασικά βήματα της μεθόδου GRASP φαίνονται στον αλγόριθμο 2.2.

Αλγόριθμος 2.2: Γενική μέθοδος GRASP

```
1  x* ← ∅
2  f* ← ∞
3  repeat
4  |  Δημιουργία μία άπληστης τυχαιοποιημένης λύσης x
5  |  Εύρεση του τοπικού βέλτιστου x_l ξεκινώντας από την x
6  |  if f(x_l) < f* then
7  |  |  x* ← x_l
8  |  |  f* ← f(x_l)
9  |  end
10 until κριτήριο τερματισμού
11 return x*
```

Η κατασκευή της αρχικής λύσης γίνεται με διαδοχικές εισαγωγές στοιχείων σε αυτήν. Το στοιχείο που εισάγεται σε κάθε βήμα στη λύση επιλέγεται από από μία λίστα περιορισμένων υποψηφίων (restricted candidate list, RCL). Η RCL αποτελείται από τα στοιχεία των οποίων ενδεχόμενη εισαγωγή τους στη τρέχουσα λύση θα επέφερε τις μικρότερες αυξήσεις στο κόστος (εξ ου και ο χαρακτηρισμός 'άπληστη'). Η επιλογή του στοιχείου γίνεται τυχαία από την RCL (εξ ου και ο χαρακτηρισμός 'τυχαιοποιημένη'). Μετά την εισαγωγή ενός στοιχείου, το κόστος εισαγωγής κάθε στοιχείου επαναϋπολογίζεται και κατασκευάζεται νέα RCL για το επόμενο βήμα (εξ ου και ο χαρακτηρισμός 'προσαρμόσιμη'). Ακολουθεί η περιγραφή των δύο φάσεων του GRASP αλγορίθμου των Martins et al. (2009).

Φάση κατασκευής

Η φάση κατασκευής εκκινεί προσδιορίζοντας τον πλήθος K των κύριων υποδέντρων της αρχικής λύσης. Για την περίπτωση μοναδιαίων απαιτήσεων κίνησης: $K = \lceil \frac{|V^*|}{w} \rceil$, όπου $w \geq Z^L$ παράμετρος εισόδου. Επίσης ορίζεται ένα βοηθητικό σύνολο X, στο οποίο διατηρεί τους κόμβους που απομένουν να εισαχθούν στη λύση (αρχικά $X = V^*$). Κατόπιν κατασκευάζουμε ξεχωριστά καθένα από τα K κύρια υποδέντρα με τον εξής τρόπο: Θεωρούμε S_k το σύνολο των τερματικών κόμβων που θα απαρτίζουν το k υποδέντρο. Αφαιρούμε τυχαία ένα κόμβο i από το σύνολο X και τον τοποθετούμε στο S_k. Όσο $X \neq \emptyset$ και $|S_k| < w$ εισάγουμε διαδοχικά κόμβους στο σύνολο S_k. Η επιλογή των κόμβων που εισάγονται στο S_k γίνεται τυχαία από RCL, την οποία κατασκευάζουμε σε κάθε βήμα με κόμβους από το σύνολο X. Έστω d_j η ελάχιστη απόσταση του κόμβου $j \in X$ από κάποιον κόμβο του S_k και $d_{min} = \min\{d_j, j \in X\}$, $d_{max} = \max\{d_j, j \in X\}$. Η RCL ορίζεται ως εξής:

$$\text{RCL} = \{j \in X : d_j \leq d_{min} + \alpha(d_{max} - d_{min})\}, \tag{2.1}$$

Αλγόριθμος 2.3: GRASP: φάση κατασκευής

$X \leftarrow V^*$
for $k \leftarrow 1$ **to** K **do**
  $i \leftarrow$ τυχαίος κόμβος από το X
  $S_k \leftarrow \{i\}$
  $X \leftarrow X \setminus \{i\}$
  while $X \neq \emptyset$ **and** $|S_k| < w$ **do**
    υπολογισμός των τιμών d_j
    δημιουργία της RCL σύμφωνα με τη σχέση (2.1)
    $m \leftarrow$ τυχαίος κόμβος από την RCL
    $S_k \leftarrow S_k \cup \{m\}$
    $X \leftarrow X \setminus \{m\}$
  end
  $J_k \leftarrow$ δέντρο που προκύπτει από επίλυση με MILP τεχνικές του MLCMST υποπροβλήματος που ορίζεται από το σύνολο $S_k \cup \{r\}$
end
return λύση που προκύπτει ενώνοντας τα J_k.

Αλγόριθμος 2.4: GRASP: φάση τοπικής αναζήτησης

$X \leftarrow V^* \setminus V_S$
while $X \neq \emptyset$ **do**
  $i \leftarrow$ τυχαίος κόμβος από το X
  $P \leftarrow S[i] \cup \{r\}$
  $\gamma \leftarrow c(J[i])$
  $Y \leftarrow V^* \setminus P$
  $h \leftarrow$ τυχαία τιμή από το διάστημα $[\underline{h}, \overline{h}]$
  while $|P| < h$ **and** $Y \neq \emptyset$ **do**
    $j \leftarrow$ τυχαίος κόμβος από το Y
    if *υπάρχει κόμβος* $u \in S[j]$ *τέτοιος ώστε* $C_{ui}^{\lambda_u} < C_{up_u}^{\lambda_u}$
    and $|S[j]| + |P| \leq h$ **then**
      $P \leftarrow P \cup S[j]$
      $\gamma \leftarrow \gamma + c(J[j])$
    end
    $Y \leftarrow \setminus S[j]$
  end
  if $|P| > |S[i]| + 1$ **then**
    Επίλυση του υποπροβλήματος που ορίζεται από το σύνολο P, και έστω T' η βέλτιστη λύση
    if $c(T') < \gamma$ **then**
      Ενημέρωση της τρέχουσας λύσης T με τις ακμές τις T'
      Έστω I το σύνολο των κόμβων-φύλλων της $'$ και $\overline{I}$ το σύνολο των υπόλοιπων τερματικών κόμβων του P
      $X \leftarrow (X \setminus I) \cup \overline{I}$
    else
      $X \leftarrow X \setminus S[i]$
    end
  else
    $X \leftarrow X \setminus \{i\}$
  end
end

όπου α παράμετρος με τιμές στο εύρος $[0, 1]$. Αφού τοποθετήσουμε w κόμβους στο S_k, κατασκευάζουμε το κύριο υποδέντρο J_k επιλύοντας το MLCMST πρόβλημα που ορίζεται από το σύνολο $J_k \cup \{r\}$ χρησιμοποιώντας MILP τεχνικές. Η παραπάνω διαδικασία επαναλαμβάνεται και για τα υπόλοιπα κύρια υποδέντρα.

Οι συγγραφείς επέλεξαν να διαλέγουν τυχαίες τιμές από το εύρος $[0.1, 0.4]$ για την παράμετρο α. Επίσης έθεσαν στην παράμετρο w την τιμή 10, ίση δηλαδή με την τιμή της μέγιστη χωρητικότητας Z^l στα προβλήματα που εξέτασαν.

Φάση τοπικής αναζήτησης

Κατά τη φάση τοπικής αναζήτησης επιχειρείται η βελτίωση της λύσης αναδιατάσσοντας κόμβους που βρίσκονται σε διαφορετικά κύρια υποδέντρα. Τα βήματα της φάσης τοπικής αναζήτησης φαίνονται στον αλγόριθμο 2.4. Κάθε βήμα της τοπικής αναζήτησης περιλαμβάνει δύο υποφάσεις:

1. την κατασκευή ενός συνόλου P το οποίο θα ορίζει ένα MLCMST υποπρόβλημα και θα περιλαμβάνει του κόμβους τουλάχιστον δύο κύριων υποδέντρων και
2. την επίλυση του υποπροβλήματος με MILP τεχνικές και ενημέρωση της τρέχουσας λύσης T αν προκύπτουν βελτιώσεις.

Για την κατασκευή του συνόλου P αρχικά δημιουργούμε ένα βοηθητικό σύνολο X στο οποίο τοποθετούμε τους τερματικούς κόμβους που δεν είναι φύλλα στην τρέχουσα λύση T. Σε κάθε βήμα της τοπικής αναζήτησης επιλέγεται τυχαία ένα κόμβος i από το γύρω από τον οποίο επιχειρείται το σύνολο P. Στη μεταβλητή γ κρατείται το συνολικό κόστος του υποδέντρου που ορίζουν οι κόμβοι του P. Αρχικά στο σύνολο P τοποθετούνται οι κόμβοι που βρίσκονται στο ίδιο κύριο υποδέντρο με τον i, δηλαδή $S[i] \cup r$ και συνεπώς στη μεταβλητή γ τίθεται η τιμή του κόστους του υποδέντρου $J[i]$. Ακολούθως δημιουργείται ένα σύνολο Y υποψήφιων κόμβων για εισαγωγή στο P, στο οποίο τοποθετούνται όλοι οι τερματικοί κόμβοι εκτός του P. Η παράμετρος h, που ορίζει το μέγεθος του υποπροβλήματος ($|P|$), επιλέγεται τυχαία από ένα εύρος $[\underline{h}, \overline{h}]$. Στη συνέχεια επιχειρείται η εισαγωγή και άλλων κύριων υποδέντρων στο P. Η διαδικασία επιλέγει τυχαία ένα κόμβο j από το σύνολο Y και ελέγχει εάν υπάρχει κάποιος κόμβος $u \in S[j]$, του οποίου η σύνδεση με τον i θα κόστιζε λιγότερο απ ότι με τον p_u. Οι κόμβοι του $S[j]$ εισάγονται στο P αν ικανοποιείται η παραπάνω συνθήκη και δεν παραβιάζεται η παράμετρος h. Επίσης ενημερώνεται η τιμή της γ προσθέτοντας σε αυτήν το κόστος του υποδέντρου $J[j]$. Εν συνεχεία οι κόμβοι του $S[j]$ αφαιρούνται από το σύνολο Y και η παραπάνω διαδικασία προσπάθειας εισαγωγής υποδέντρων στο P συνεχίζεται μέχρι να εξαντληθεί το σύνολο Y ή η πληθικότητα του P να ισούται με h.

Εάν το σύνολο P που προέκυψε περιλαμβάνει μόνο τους κόμβους $S[i] \cup \{r\}$ τότε το υποδέντρο $S[i]$ δε μπορεί να συνδυαστεί με κάποιο άλλο υποδέντρο όποτε ο κόμβος i αφαιρείται από το X και εκκινείται νέο βήμα της τοπικής αναζήτησης. Σε αντίθετη περίπτωση υπολογίζεται η βέλτιστη λύση T' του υποπροβλήματος που ορίζεται από το σύνολο P. Αν $c(T') < \gamma$, τότε ενημερώνουμε την τρέχουσα λύση T με τις ακμές της T' και ανανεώνουμε το σύνολο X ως εξής: Αφαιρούμε από το X τους κόμβους-φύλλα (I) της T' και εισάγουμε στο τους υπόλοιπους τερματικούς κόμβους μη φύλλα ($\overline{I}$) του P. Αν αντίθετα $c(T') = \gamma$ τότε απλά αφαιρούμε τους κόμβους του $S[i]$ από το σύνολο X.

Οι συγγραφείς χρησιμοποίησαν τις ακόλουθες τιμές για τις παραμέτρους της φάσης τοπικής αναζήτησης στα στιγμιότυπα που εξέτασαν: $\underline{h} = 16, \overline{h} = 31$.

2.6 Εφαρμογές μεταευρετικών διαδικασιών σε πραγματικά προβλήματα

Οι Rothlauf et al. (2000) πρότειναν την -νέα τότε- κωδικοποίηση δέντρων Netkeys για γενετικούς και εξελικτικούς αλγορίθμους (αναλυτικότερα για κωδικοποιήσεις δέντρων στο κεφ. 6). Για την αξιολόγηση των επιδόσεων των Netkeys κατασκεύασαν τέσσερα προβλήματα δικτύων τοπολογίας δέντρου. Ένα από αυτά τα προβλήματα αφορούσε μία εταιρία με 15 υποκαταστήματα τα οποία είχαν συγκεκριμένες ανάγκες επικοινωνίας με τα κεντρικά γραφεία. Κατά το σχεδιασμό του δικτύου τοπολογίας δέντρου δινόταν η δυνατότητα επιλογής από τρεις γραμμές διαφορετικών χωρητικοτήτων, η κοστολόγηση των οποίων προέκυπτε από την τιμολογιακή πολιτική της Deutsche Telecom το 1996. Το κόστος μίας ζεύξης αποτελείτο από ένα σταθερό και ένα εξαρτώμενο από την απόσταση μέρος Και τα δύο μέρη εξαρτιόνταν από τη χωρητικότητα της ζεύξης. Το πραγματικό αυτό πρόβλημα στην ουσία είναι στιγμιότυπο του MLCMST.

Οι Papagianni et al. (2008, 2009a) προσέγγισαν το παραπάνω πρόβλημα με μία παραλλαγή της μεθόδου Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swatm Optimization, PSO). Η PSO είναι μία μεταευρετική τεχνική βασισμένη σε έναν πληθυσμό υποψήφιων λύσεων, τα σωματίδια, οι οποίες κινούνται στο χώρο λύσεων. Σε κάθε κύκλο του αλγορίθμου ένα σωματίδιο μεταβαίνει σε μία νέα 'θέση', η οποία στην ουσία είναι ένα διάνυσμα λύσης. Η νέα θέση προκύπτει κάθε φορά από την προσθήκη ενός διανύσματος 'ταχύτητας' στο διάνυσμα θέσης. Στην αρχική εκδοχή της PSO (Kennedy and Eberhart, 1995) η ταχύτητα των σωματιδίων ενημερώνεται σε κάθε νέο κύκλο σύμφωνα με την καλύτερη θέση που έχει ανακαλύψει το ίδιο το σωματίδιο (personal best: $\hat{\mathbf{x}}$) αλλά και την καλύτερη θέση που έχει ανακαλύψει οποιαδήποτε σωματίδιο (global best: $\mathbf{x}^*$). Το *ίοστό*

στοιχείο της νεάς ταχύτητας ορίζεται ως εξής:

$$v'_i = v_i + c_1 r_1(\hat{x}_i - x_i) + c_2 r_2(x^*_i - x_i)$$

όπου v_i η τιμή του στοιχείου ταχύτητας στον προηγούμενο κύκλο, c_1 και c_2 σταθεροί συντελεστές, r_1 και r_2 τυχαίες τιμές στο διάστημα $[0, 1]$ και x_i η τιμή του αντίστοιχου στοιχείου θέσης. Το iοστό στοιχείο της νεάς θέσης ορίζεται ως εξής:

$$x'_i = x_i + v'_i$$

Εναλλακτικά του global best μπορεί να χρησιμοποιηθεί το local best δηλαδή η καλύτερη θέση που έχει ανακαλύψει ένα σωματίδιο στη 'γειτονιά' του τρέχοντος σωματιδίου. Στο συγκεκριμένο πρόβλημα εφαρμόστηκε η παραλλαγή Attractive and Repulsive PSO (Riget and Vesterstroem, 2002) σε συνδυασμό με τη χρησιμοποίηση τελεστή μετάλλαξης (mutation).

Οι Papagianni et al. (2009b,c) παρουσίασαν μία παραλλαγή της PSO στην οποία αντί του global best επιλέγεται ως οδηγός τυχαία ένα από τα personal best που έχει ανακαλύψει το συγκεκριμένο σωματίδιο. Η συγκεκριμένη παραλλαγή εφαρμόστηκε σε δύο στιγμιότυπα, 15 και 29 τερματικών κόμβων αντίστοιχα. Οι κόμβοι των στιγμιότυπων επιλέχθηκαν από ένα σύνολο τοποθεσιών από μία γεωγραφική περιοχή της Ελλάδας ενώ η κοστολόγηση βασίστηκε στην τιμολογιακή πολιτική του ΟΤΕ για τις μισθωμένες γραμμές.

2.7 Επιδόσεις στο σύνολο στιγμιότυπων των Gamvros et al. (2006)

Οι αλγόριθμοι που περιγράφηκαν στις ενότητες 2.2-2.5 έχουν αξιολογηθεί από τους συγγραφείς τους με υπολογιστικά πειράματα πάνω σε ένα σύνολο στιγμιότυπων του προβλήματος MLCMST το οποίο κατασκεύασαν οι Gamvros et al. (2006). Τα στιγμιότυπα περιελάμβαναν σύνολα κόμβων ποικίλων μεγεθών (20, 30, 50, 100 και 150 τερματικών κόμβων) και διαφορετικών τοποθετήσεων του κεντρικού κόμβου (στο κέντρο, στην άκρη, σε τυχαία θέση). Για κάθε συνδυασμό μεγέθους προβλήματος και τοποθέτησης κεντρικού κόμβου κατασκευάστηκαν 50 διαφορετικά σύνολα κόμβων με τους τερματικούς κόμβους να τοποθετούνται τυχαία σε ένα πλέγμα 20x20. Το κόστος για μία ζεύξη ορίστηκε ως το γινόμενο της ευκλείδειας απόστασης μεταξύ των δύο άκρων επί ενός συντελεστή c^l χαρακτηριστικού του τύπου ζεύξης, ο οποίος εκφράζει το κόστος ανά μονάδα απόστασης. Επίσης, η κάθε τερματικός παρήγε κίνηση ίση με 1 σε όλα τα στιγμιότυπα ενώ υπήρχαν τρεις διαθέσιμοι τύποι

ζεύξεων με χωρητικότητες $Z^1 = 1$, $Z^2 = 3$ και $Z^3 = 10$ μονάδες κίνησης και συντελεστές κόστους $c^1 = 1$, $c^2 = 2$, και $c^3 = 6$ αντίστοιχα.

Τα υπολογιστικά πειράματα χωρίζονταν σε δύο κατηγορίες: σε προβλήματα μικρού μεγέθους όπου ήταν δυνατό να βρεθούν οι βέλτιστες λύσεις σε εύλογο χρονικό διάστημα με τη χρήση MILP τεχνικών και σε προβλήματα μεγαλύτερου μεγέθους όπου κάτι τέτοιο δεν ήταν δυνατόν. Η αποτίμηση της επίδοσης των αλγορίθμων για τα μικρά στιγμιότυπα έγινε κυρίως υπολογίζοντας τη μέση ποσοστιαία απόσταση (average percentage gap) από τις βέλτιστες τιμές. Σε μεγαλύτερα προβλήματα όπου δεν ήταν δυνατόν να βρεθούν οι βέλτιστες τιμές οι αποτίμηση έγινε με βάση κάτω όρια τα οποία έχουν υπολογιστεί από τη γραμμική χαλάρωση των SCF2 διατυπώσεων (βλέπε σελ. 64 - στην εργασία των Gamvros et al. (2006) αναφέρεται ως ESCF). Οι μέσες ποσοστιαίες αποστάσεις από τα κάτω όρια (lb_i) ισούται με:

Τα αποτελέσματα εκτελέσεων που παραθέτουμε προέρχονται από τους Gamvros et al. (2006) για τους αλγορίθμους GUH, MES, RSMES και GA ενώ τα αντίστοιχα αποτελέσματα για τον GRASP από τους Martins et al. (2009). Ο πίνακας 2.1 απεικονίζονται οι τιμές gap_{opt} για μικρά στιγμιότυπα ενώ ο πίνακας 2.2 τις τιμές gap_{lb} για όλα τα στιγμιότυπα. Οι χρόνοι εκτέλεσης των αλγορίθμων φαίνονται στον πίνακα 2.3.

Τα στιγμιότυπα στα οποία βασίστηκαν τα υπολογιστικά πειράματα χαρακτηρίζονται από δυο βασικές ιδιαιτερότητες:

- δεν εξετάζεται η μη μοναδιαία περίπτωση και
- εξετάζεται ένα πολύ συγκεκριμένο σύνολο τύπων ζεύξεων όπου η μέγιστη δυνατή χωρητικότητα Z^L ισούται με 10 μονάδες κίνησης.

Θα αναλύσουμε τώρα τις δυσκολίες αντιμετώπισης στιγμιότυπων όπου αίρονται οι παραπάνω περιορισμοί με του ευρετικούς αλγορίθμους της βιβλιογραφίας.

Μη μοναδιαία περίπτωση

Ο αλγόριθμος αναβαθμίσεων, όπως παρουσιάστηκε από τους Gamvros et al. (2006), μπορεί να αντιμετωπίσει μόνο στιγμιότυπα όπου η παραγόμενη από κάθε τερματικό κόμβο κίνηση είναι μοναδιαία. Ο περιορισμός έγκειται στη διαδικασία εύρεσης του βέλτιστου συνόλου H κόμβων που επανασυνδέονται κατά την εφαρμογή μίας αναβάθμισης. Οι Gamvros et al. (2006) πρότειναν ως μελλοντική εργασία την ανάπτυξη γρήγορων ευρετικών διαδικασιών για τον αντιμετώπιση του προβλήματος σακιδίου (knapsack) που προκύπτει στη γενική περίπτωση του προβλήματος MLCMST (βλέπε κεφ. 2.2, σελ. 32). Δεδομένου ότι οι δύο αλγόριθμοι πολλαπλών ανταλλαγών (MES, RSMS), αλλά και ο υβριδικός γενετικός αλγόριθμος που πρότειναν οι ίδιοι συγγραφείς βασίζονται στον αλγόριθμο

problem set	GUH	MES	RSMES	GA
20c	3.47 %	0.76 %	0.27 %	0.17 %
20e	5.45 %	3.66 %	0.86 %	0.46 %
20r	3.25 %	1.05 %	0.47 %	0.12 %
30c	5.00 %	1.81 %	0.85 %	0.27 %

Πίνακας 2.1: Στιγμότυπα των Gamvros et al. (2006): Αποστάσεις από τις βέλτιστες τιμές

problem set	GUH	MES	RSMES	GA	GRASP
20c	10.24 %	7.35 %	6.83 %	6.73 %	
20e	11.56 %	9.67 %	6.70 %	6.29 %	
20r	10.58 %	8.22 %	7.60 %	7.23 %	
30c	12.17 %	8.77 %	7.75 %	7.12 %	
30e	9.98 %	6.57 %	5.57 %	5.69 %	
30r	11.02 %	7.61 %	6.88 %	6.64 %	
50c	11.15 %	8.15 %	7.56 %	7.30 %	7.13 %
50e	8.14 %	5.15 %	4.77 %	4.69 %	4.21 %
50r	9.53 %	6.47 %	6.01 %	5.78 %	5.22 %
100c	9.93 %	6.64 %	6.45 %	6.38 %	5.48 %
100e	6.16 %	3.85 %	3.96 %	3.80 %	
100r	8.52 %	5.47 %	5.55 %	5.39 %	
150c	8.52 %	5.51 %			4.86 %
150e	5.05 %	3.24 %			
150r	6.75 %	4.37 %			

Πίνακας 2.2: Στιγμότυπα των Gamvros et al. (2006): Αποστάσεις από τα κάτω όρια

problem set	GUH	MES	RSMES	GA	GRASP
20c	<0.01	0.45	5.34	11.43	
20e	<0.01	0.94	12.24	25.37	
20r	<0.01	0.74	9.52	15.52	
30c	<0.01	2.38	27.18	26.49	
30e	<0.01	6.14	91.89	76.81	
30r	<0.01	3.96	56.19	51.67	
50c	<0.01	17.31	256.61	170.57	1891.73
50e	<0.01	29.86	668.14	369.08	13141.27
50r	<0.01	19.82	384.40	225.58	2358.31
100c	0.04	194.16	3557.17	2503.97	1829.98
100e	0.04	284.91	6982.16	2974.92	
100r	0.03	237.84	4748.49	2155.69	
150c	0.09	807.94	6827.744		
150e	0.10	1015.32			
150r	0.09	833.67			

Πίνακας 2.3: Στιγμότυπα των Gamvros et al. (2006): Μέσοι χρόνοι εκτέλεσης (s)

αναβαθμίσεων, οι αλγόριθμοι αυτοί δεν μπορούν να εφαρμοστούν άμεσα στη γενική περίπτωση.

Επίσης, η εφαρμογή μίας πολλαπλής ανταλλαγής (κυκλική ανταλλαγή ή ανταλλαγή μονοπάτι) δε θα έχει τα ίδια αποτελέσματα όταν εφαρμοστεί σε ένα μη μοναδιαίο στιγμιότυπο. Μετά από την εφαρμογή μίας πολλαπλής ανταλλαγής ο αριθμός των κόμβων σε κάθε υποδέντρο παραμένει σταθερός, με εξαίρεση το πρώτο και τελευταίο υποδέντρο μίας ανταλλαγής-μονοπατιού όπου έχουμε μοναδιαία μείωση και αύξηση αντίστοιχα. Σε ένα πρόβλημα μοναδιαίων απαιτήσεων, η παραπάνω διαπίστωση συνεπάγεται ότι η συνολική παραγόμενη κίνηση σε ένα κύριο υποδέντρο παραμένει αμετάβλητη κατά την ανταλλαγή και άρα και η χωρητικότητα της ζεύξης που συνδέει το υποδέντρο με τον κεντρικό κόμβο δε χρειάζεται να μεταβληθεί. Αντίθετα, στη μη μοναδιαία περίπτωση μία πολλαπλή ανταλλαγή εν γένει θα επέφερε αλλαγές στην συνολική κίνηση κάθε κύριου υποδέντρου και πιθανόν και αλλαγές στου τύπους των ζεύξεων έτσι ώστε να μην παραβιάζονται οι περιορισμοί χωρητικότητας. Συνεπώς είναι αμφίβολο αν οι μηχανισμοί πολλαπλής ανταλλαγής θα έχουν την ίδια αποτελεσματικότητα σε μη μοναδιαία στιγμιότυπα.

Μέγιστη τιμή χωρητικότητας

Η τιμή της παραμέτρου Z^L ενός προβλήματος MLCMST επηρεάζει σε μεγάλο βαθμό τη δομή των εφικτών λύσεων του προβλήματος. Δεδομένου ότι οποιαδήποτε ζεύξη μίας εφικτής λύσης δε μπορεί να μεταφέρει περισσότερες από Z^L μονάδες κίνησης, οδηγούμαστε στο συμπέρασμα ότι η κίνηση που παράγεται σε ένα κύριο υποδέντρο S_i μίας εφικτής λύσης του προβλήματος δε μπορεί να ξεπερνά την τιμή Z^L, δηλαδή $\sum_{j \in S_i} F_j \leq Z^L$. Συνεπώς το πλήθος των κόμβων που απαρτίζουν ένα κύριο υποδέντρο είναι το πολύ ίσο με Z^L και το πλήθος των κύριων υποδέντρων μίας εφικτής λύσης είναι κατ ελάχιστον $K = \lceil \frac{|V^*|}{Z^L} \rceil$. Οι παραπάνω διαπιστώσεις συνεπάγονται ότι σε προβλήματα με μικρές τιμές Z^L οι εφικτές λύσεις θα έχουν αρκετά στο πλήθος και μικρά στο μέγεθος κύρια υποδέντρα και άρα περιορίζονται σε τοπολογίες που θα μοιάζουν με αστέρα (star-like topologies). Για παράδειγμα μία λύση στιγμιότυπου 150 τερματικών κόμβων με $Z^L = 10$ θα έχει τουλάχιστον 15 κύρια υποδέντρα, όπου το καθένα θα αποτελείται από το πολύ 10 κόμβους.

Η δομή της λύσης ενός προβλήματος με χαμηλή τιμή Z^L ευνοεί τη αντιμετώπιση του ως πρόβλημα καταρχήν ομαδοποίησης των κόμβων σε κύρια υποδέντρα και δευτερευόντως βελτιστοποίησης των υποδέντρων. Ο γενετικός αλγόριθμος των Gamvros et al. (2006), εφαρμόζει την παραπάνω στρατηγική βελτιστοποιώντας κάθε υποδέντρο ξεχωριστά με τον αλγόριθμο GUH και καταλήγει σε πολύ καλύτερες λύσεις από αυτές που δίνει η απλή εφαρμογή του GUH. Αντιθέτως σε προβλήματα όπου οι βέλτιστες λύσεις συνίστανται σε λίγα κύρια υποδέντρα ο ρόλος της ομαδοποίησης μειώνεται όποτε αναμένεται να έχουμε και μικρότερες βελτιώσεις έναντι του απλού GUH.

Παρομοίως, ο αλγόριθμος GRASP των Martins et al. (2009) βασίζεται στην επίλυση με ακριβείς μεθόδους υποπροβλημάτων MLCMST που προκύπτουν ενώνοντας δύο ή περισσότερα κύρια υποδέντρα. Για τα στιγμιότυπα της βιβλιογραφίας, όπου $Z^L = 10$, ένα υποπρόβλημα με 2 υποδέντρα θα αποτελείται από το πολύ 21 κόμβους, μέγεθος που είναι αντιμετωπίσιμο με MILP τεχνικές. Αντίθετα αν η τιμή Z^L είναι πολύ μεγαλύτερη τότε δεν υπάρχει εγγύηση ότι ο αλγόριθμος μπορεί καν να πλησιάσει τη βέλτιστη ή άλλες καλές λύσεις αν αυτές περιλαμβάνουν κύρια υποδέντρα μεγάλου μεγέθους.

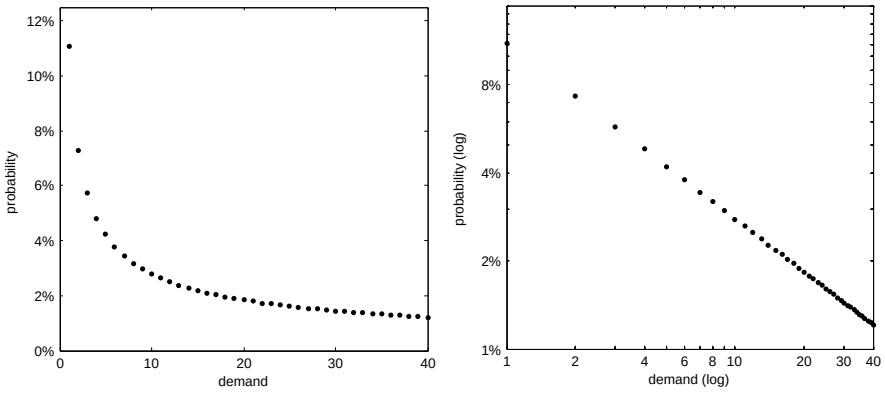
2.8 Κατασκευή στιγμιότυπων αναφοράς

Για τις ανάγκες αξιολόγησης των αλγορίθμων που παρουσιάζονται στο υπόλοιπο της διατριβής κατασκευάσαμε μία σειρά στιγμιότυπων του προβλήματος MLCMST τα οποία υπερβαίνουν τις ιδιαιτερότητες που επισημάνθηκαν για τα στιγμιότυπα των Gamvros et al. (2006). Οποιοδήποτε στιγμιότυπο MLCMST χαρακτηρίζεται από:

- το πλήθος των τερματικών κόμβων,
- τις θέσεις των κόμβων,
- τις απαιτήσεις κίνησης των τερματικών κόμβων,
- τους διαθέσιμους τύπους ζεύξεων.

Σχετικά με τις απαιτήσεις κίνησης εξετάζονται δύο περιπτώσεις. Στην πρώτη περίπτωση η κίνηση κάθε τερματικού κόμβου τίθεται ίση με 1 (μοναδιαία) ενώ στη δεύτερη η τιμή διαφοροποιείται από κόμβο σε κόμβο (μη μοναδιαία). Για τη μη μοναδιαία περίπτωση ορίζουμε καταρχήν στάθμες κίνησης $1, 2, \ldots, 40$. Ακολούθως, αντιστοιχίζουμε σε κάθε τερματικό κόμβο ψευδοτυχαίες τιμές έτσι ώστε η συχνότητα εμφάνισης κάθε στάθμης να ακολουθεί κατανομή Zipf με παράμετρο εκθέτη $\alpha = 0.6$. Στο σχήμα 2.4 απεικονίζεται γραφικά η πιθανότητα εμφάνισης για κάθε στάθμη. Η πιθανότητα βαίνει μειούμενη καθώς κινούμαστε προς υψηλότερες στάθμες. Χαρακτηριστικό δεδομένων που ακολουθούν κατανομή Zipf είναι η απεικόνιση τους σε γράφημα με λογαριθμικούς άξονες να τείνει να σχηματίζει ευθεία (σχήμα 2.4). Επίσης, η μέση τιμή παραγόμενης κίνησης ανά τερματικό κόμβο είναι $F^{NU}_{mean} \simeq 14.05$.

Όσον αφορά τους διαθέσιμους τύπους ζεύξεων κατασκευάστηκαν τρία διαφορετικά σύνολα, τα οποία συνδυαζόμενα με τις δύο περιπτώσεις κίνησης μας δίνουν έξι διαφορετικά *σενάρια* (πίνακας 2.6). Κάθε σενάριο εξετάζει προβλήματα ποικίλων μεγεθών, ενώ για κάθε μέγεθος προβλήματος κατασκευάστηκαν 50 διαφορετικά στιγμιότυπα με τους τερματικούς κόμβους αλλά και τον κεντρικό κόμβο να τοποθετούνται τυχαία σε ένα πλέγμα 4000x4000.



(α′) γραμμικοί άξονες *(β′) λογαριθμικοί άξονες*

Σχήμα 2.4: Κατανομή απαιτήσεων κίνησης τερματικών κόμβων (μη μοναδιαία περίπτωση)

l	Z^l	c^l	c^l/Z^l
1	$1 \times D$	$1 \times D$	1.00
2	$3 \times D$	$2 \times D$	0.67
3	$10 \times D$	$6 \times D$	0.60

Πίνακας 2.4: Περιορισμένο σύνολο τύπων ζεύξεων (RES)

l	Z^l	c^l	c^l/Z^l
1	$1 \times D$	$1 \times D$	1.000
2	$3 \times D$	$2 \times D$	0.667
3	$10 \times D$	$5 \times D$	0.500
4	$30 \times D$	$10 \times D$	0.333
5	$100 \times D$	$20 \times D$	0.200
6	$400 \times D$	$50 \times D$	0.125

Πίνακας 2.5: Εκτεταμένο σύνολο τύπων ζεύξεων (EXT)

σενάριο	παραγόμενη κίνηση	σύνολο τύπων ζεύξεων	Z^L
RES-U	μοναδιαία	Περιορισμένο με $D = 1$ (πίν. 2.4)	10
EXT-U	μοναδιαία	Εκτεταμένο με $D = 1$ (πίν. 2.5)	400
RES-NU	μη μοναδιαία	Περιορισμένο με $D = 15$ (πίν. 2.4)	150
EXT-NU	μη μοναδιαία	Εκτεταμένο με $D = 15$ (πίν. 2.5)	6000

Πίνακας 2.6: Εξεταζόμενα σενάρια

Δεδομένου ότι η μέση παραγόμενη κίνηση είναι σημαντικά υψηλότερη στα μη μοναδιαία σενάρια εξισορροπούμε τις δύο περιπτώσεις εξαρτώντας τις χωρητικότητες των ζεύξεων από ένα συντελεστή D: Θέτουμε $D = 1$ στα μοναδιαία σενάρια και $D = 14$ στα μη μοναδιαία σενάρια. Επίσης και τα δύο σύνολα τύπων επιδεικνύουν οικονομίες κλίμακας, δηλαδή οι τιμές c^l/Z^l βαίνουν μειούμενες.

Το πρώτο σύνολο τύπων ζεύξεων (πίνακας 2.4) αποτελείται από τρείς τύπους. Χαρακτηρίζουμε το σύνολο αυτό ως *περιορισμένο* καθότι η μέγιστη χωρητικότητα είναι ίση με $10D$ και ως εκ τούτου μία ζεύξη μέγιστης χωρητικότητας μπορεί να εξυπηρετήσει την κίνηση περίπου 10 κόμβων. Στη μοναδιαία περίπτωση το περιορισμένο σύνολο τύπων είναι ισοδύναμο με αυτό που χρησιμοποίησαν οι Gamvros et al. (2006).

Το δεύτερο *εκτεταμένο* σύνολο (πίνακας 2.5) περιλαμβάνει περισσότερους τύπους ζεύξεων με το μέγιστο τύπο να προσφέρει χωρητικότητα ίση με $400D$. Είναι αναμενόμενο οι βέλτιστες λύσεις προβλημάτων με λίγους κόμβους μεγέθους να μη περιλαμβάνουν ζεύξεις των υψηλότερων τύπων.

Κεφάλαιο 3

Προσέγγιση μεικτού ακεραίου γραμμικού προγραμματισμού

3.1 Μεικτός Ακέραιος Γραμμικός Προγραμματισμός

Προβλήματα βελτιστοποίησης όπου η αντικείμενη συνάρτηση είναι γραμμική συνάρτηση και ο χώρος λύσεων ορίζεται από ένα πεπερασμένο σύνολο γραμμικών περιορισμών (constraints) (ανισότητες ή ισότητες) πάνω στον $\mathbb{R}^n$ ονομάζονται προβλήματα γραμμικού προγραμματισμού (Linear Programming,LP).

Πρόβλημα 3.1: Γενική διατύπωση προβλήματος γραμμικού προγραμματισμού (LP)

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

όπου $\mathbf{x}$ ένα διάνυσμα άγνωστων μεταβλητών $\mathbf{c}$ και $\mathbf{b}$ διανύσματα γνωστών συντελεστών και A πίνακας γνωστών συντελεστών.

Ο Dantzig (1951) παρουσίασε τον αλγόριθμο simplex για την επίλυση προβλημάτων LP, ο οποίος αποδείχτηκε αρκετά πιο αποδοτικός στην πράξη από παλαιότερες μεθόδους. Ο simplex έχει αποδειχτεί ότι λύνει τυχαία προβλήματα αποδοτικά (συνήθως σε πολυωνυμικό χρόνο Borgwardt (1987)), όμως οι Klee and Minty (1972) κατασκεύασαν μία οικογένεια προβλημάτων των οποίων η επίλυση μέσω του simplex απαιτεί εκθετικό χρόνο. Η πολυπλοκότητα των προβλημάτων LP αποτελούσε ανοιχτό πρόβλημα έως ότου ο Khachiyan (1979) απέδειξε ότι όλα τα προβλήματα LP είναι επιλύσιμα

σε πολυωνυμικό χρόνο μέσω του ελλειψοειδή αλγορίθμου που ο ίδιος πρότεινε. Ωστόσο, ο αλγόριθμος του Khachiyan ήταν αρκετά πιο αργός από τον simplex στα περισσότερα πρακτικά προβλήματα. Ο Karmarkar (1984) εισήγαγε τον πρώτο πολυωνυμικό αλγόριθμο, ο οποίος κατάφερνε να είναι αρκετά αποδοτικός στη μέση περίπτωση. Ο αλγόριθμος του Karmarkar βασιζόταν στη μέθοδο εσωτερικού σημείου και έδωσε το έναυσμα για την ανάπτυξη και άλλων αλγορίθμων εσωτερικού σημείου όπως αυτός που πρότεινε ο Mehrotra (1992). Σήμερα, οι παραλλαγές του simplex έχουνε παρόμοια απόδοση σε σχέση με τους αλγορίθμους εσωτερικού σημείου, με καμία από τις δύο τεχνικές να μην υπερτερεί σε όλα τα προβλήματα (Gondzio and Terlaky (1994); Szabo and Kovacs (2003)).

Στην περίπτωση που ορισμένοι από τους αγνώστους απαιτείται να είναι ακέραιοι τότε το εξεταζόμενο πρόβλημα αποκαλείται πρόβλημα μεικτού ακέραιου γραμμικού προγραμματισμού (mixed integer linear programming - MILP). Ειδικές περιπτώσεις του μεικτού ακέραιου γραμμικού προγραμματισμού είναι ο ακέραιος γραμμικός προγραμματισμός (integer linear programming), όπου όλοι οι άγνωστοι είναι ακέραιοι και ο δυαδικός γραμμικός προγραμματισμός, όπου όλες οι μεταβλητές αγνώστων παίρνουν αυστηρά της τιμές 0 ή 1.

Πρόβλημα 3.2: Γενική διατύπωση προβλήματος μεικτού ακέραιου γραμμικού προγραμματισμού (MILP)

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u} \\ & x_i \in \mathbb{Z}, \quad \forall i \in I \end{aligned}$$

όπου I το σύνολο των δεικτών των ακέραιων αγνώστων.

Σε αντίθεση με τα προβλήματα LP που λύνονται αποδοτικά, τα προβλήματα MILP είναι σε πολλές πρακτικές περιπτώσεις NP-hard. Οι δύο βασικότερες τεχνικές για την επίλυση δύσκολων MILP προβλημάτων είναι η μέθοδος Διακλάδωσης και Οριοθέτησης (Branch and Bound) και η μέθοδος Τεμνόντων Επιπέδων (Cutting Plane). Οι τεχνικές αυτές δε χαρακτηρίζονται ως ευρετικές καθότι δε στοχεύουν στην εύρεση εφικτών λύσεων καλής ποιότητας, αλλά στην εύρεση και απόδειξη της ολικά βέλτιστης λύσης. Η συνδυαστική χρήση των δύο τεχνικών αναφέρεται ως Branch and Cut. Δημοφιλή πακέτα επίλυση προβλημάτων MILP, όπως οι εμπορικοί ILOG CPLEX, Xpress-MP MOSEK, Gurobi, αλλά και οι μη εμπορικοί GLPK, SCIP, CBC, lp_solve, βασίζονται στη μέθοδο Branch and Cut.

Ο Mittelmann δημοσιεύει συγκρίσεις μεταξύ των επιλυτών στην ιστοσελίδα του ανά τακτά χρονικά

	επιλυτές	γεωμετρικός μέσος χρόνων επίλυσης (sec)	ποσοστό προβλημάτων που επιλύθηκαν
εμπορικοί	Gurobi 5.0.0	160.14	88.5%
	Xpress MP 7.3.1	174.92	87.4%
	CPLEX 12.4.0	202.26	86.2%
	SCIP 3.0.0 - CPLEX 12.4.0	539.36	73.6%
μη εμπορικοί	SCIP 3.0.0 - SoPlex 1.7.0	800.26	66.7%
	SCIP 3.0.0 - CLP 1.14.7	848.48	63.2%
	CBC 2.7.7	1611.11	47.1%
	lp_solve 5.5.2	3034.47	5.7%
	GLPK 4.47	3456.74	3.4%

Πίνακας 3.1: Σύγκριση πακέτων επίλυσης προβλημάτων MILP

διαστήματα. Ο πίνακας 3.1 απεικονίζει αποτελέσματα που δημοσιεύτηκαν στις 8/9/2012 και αφορούν τις επιδόσεις των επιλυτών σε 87 MILP προβλήματα με χρονικό όριο επίλυσης τη μία ώρα.

Προτού περιγράψουμε τις δύο μεθόδους είναι απαραίτητο να αναφερθούμε στην έννοια της γραμμικής χαλάρωσης. Η έννοια της χαλάρωσης αναφέρεται στην προσέγγιση ενός δύσκολου προβλήματος με ένα συγγενές πρόβλημα το οποίο λύνεται πιο εύκολα. Η λύση του χαλαρωμένου προβλήματος μας δίνει πληροφορίες για τη λύση του αρχικού προβλήματος. Ως *γραμμική χαλάρωση* ενός προβλήματος MILP ορίζεται το πρόβλημα που προκύπτει αν αφαιρέσουμε τους περιορισμούς ακεραιότητας (integrality constraints), ή με άλλα λόγια εάν αντικαταστήσουμε τους ακέραιους αγνώστους με πραγματικούς. Για παράδειγμα ένας δυαδικός άγνωστος του αρχικού προβλήματος μπορεί να δεχθεί οποιαδήποτε τιμή από το διάστημα $[0, 1]$ στο χαλαρωμένο πρόβλημα. Το χαλαρωμένο πρόβλημα μπορεί να επιλυθεί σχετικά εύκολα με χρήση αλγορίθμων γραμμικού προγραμματισμού δεδομένου ότι είναι πρόβλημα LP.

Έστω P ένα πρόβλημα MILP, LR η γραμμική χαλάρωσή του και $\mathbf{x}_P^{opt}$, $\mathbf{x}_{LR}^{opt}$ αντίστοιχα βέλτιστες λύσεις τους. Οποιαδήποτε εφικτή λύση του P, συμπεριλαμβανομένης και της $\mathbf{x}_P^{opt}$, αποτελεί επίσης εφικτή λύση και του LR. Συνεπώς:

$$f\left(\mathbf{x}_{LR}^{opt}\right) \leq f\left(\mathbf{x}_P^{opt}\right). \tag{3.1}$$

Με άλλα λόγια η βέλτιστη τιμή του LR αποτελεί κάτω όριο της βέλτιστης τιμής του P. Επίσης, εάν η $\mathbf{x}_{LR}^{opt}$ τυχαίνει να είναι ακέραια (δηλαδή οι τιμές όλων των αγνώστων είναι ακέραιες) τότε είναι και βέλτιστη λύση του P.

Η επίλυση των γραμμικών χαλαρώσεων, εκτός του ότι αποτελεί βασικό στοιχείο των μεθόδων Διακλάδωσης και Οριοθέτησης και Τεμνόντων Επιπέδων, μπορεί να χρησιμοποιηθεί για την αξιολόγηση

εφικτών λύσεων. Αν $\mathbf{x}_P$ μία οποιαδήποτε εφικτή λύση του P τότε $f\left(\mathbf{x}_{LR}^{opt}\right) \leq f\left(\mathbf{x}_P^{opt}\right) \leq f\left(\mathbf{x}_P\right)$. Συνεπώς μπορούμε να φράξουμε από πάνω τη σχετική απόσταση της $\mathbf{x}_P$ από τη βέλτιστη ως εξής:

$$\frac{f\left(\mathbf{x}_P\right) - \left(\mathbf{x}_P^{opt}\right)}{f\left(\mathbf{x}_P^{opt}\right)} = \frac{f\left(\mathbf{x}_P\right)}{f\left(\mathbf{x}_P^{opt}\right)} - 1 \leq \frac{f\left(\mathbf{x}_P\right)}{f\left(\mathbf{x}_{LR}^{opt}\right)} - 1 = \frac{f\left(\mathbf{x}_P\right) - f\left(\mathbf{x}_{LR}^{opt}\right)}{f\left(\mathbf{x}_{LR}^{opt}\right)} \tag{3.2}$$

Το παραπάνω μέτρο χρησιμοποιείται σε αρκετές περιπτώσεις αφού συχνά δε γνωρίζουμε τη βέλτιστη λύση.

Οι διαφορετικές γραμμικές διατυπώσεις ενός προβλήματος δίνουν τις ίδιες βέλτιστες λύσεις όμως αυτό δεν ισχύει και για τις χαλαρώσεις τους. Μία χαλάρωση χαρακτηρίζεται 'ισχυρή ή 'αδύναμη' ανάλογα με το βαθμό που η λύση της προσεγγίζει τη λύση του αρχικού προβλήματος.

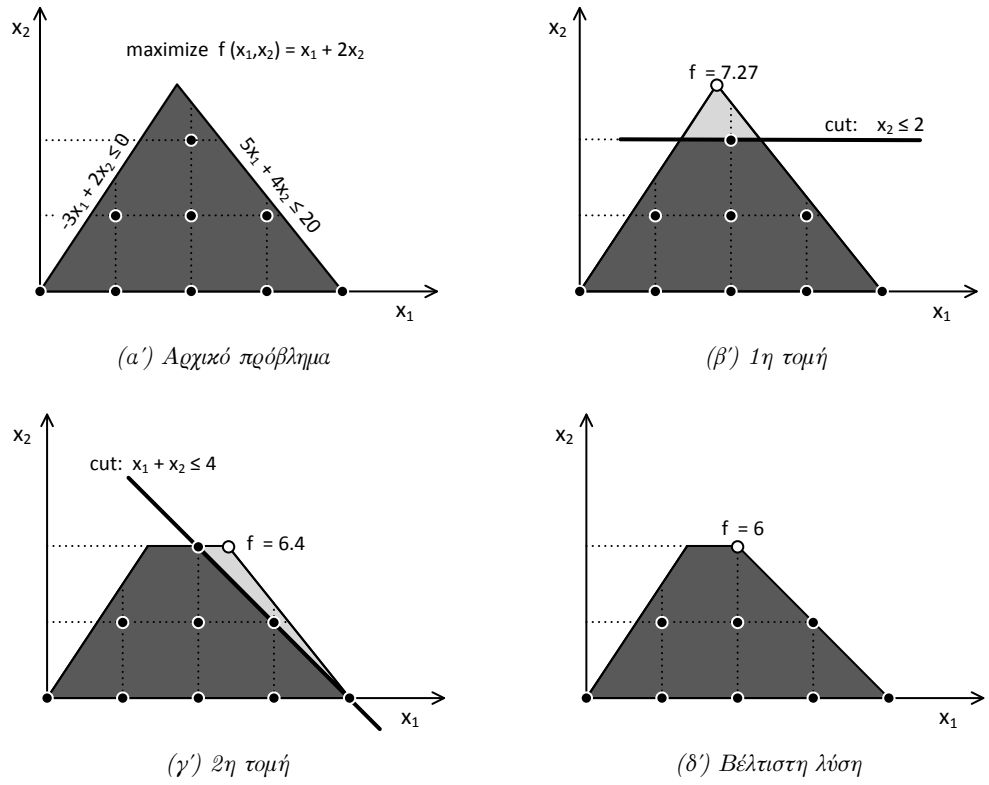
3.1.1 Μέθοδος Τεμνόντων Επιπέδων (Cutting Planes)

Η τεχνική των τεμνόντων επιπέδων είχε προταθεί από τον Gomory (1958) ως μέθοδος για την επίλυση προβλημάτων MILP. Για μεγάλο χρονικό διάστημα θεωρούνταν μη πρακτική μέθοδος, κυρίως λόγω της αριθμητικής αστάθειας και της ανάγκης διενέργειας πολλών επαναλήψεων για να σημειωθεί πρόοδος. Ωστόσο στα μέσα της δεκαετίας του 1990 αναπτύχθηκαν τεχνικές αντιμετώπισης των αριθμητικών ασταθειών και πλέον σήμερα οι τομές αποτελούν βασικό εργαλείο για όλους τους επιλυτές προβλημάτων MILP.

Αρχικά επιλύεται η γραμμική χαλάρωση του προβλήματος και αν η $\mathbf{x}_{LR}^{opt}$ είναι ακέραια τότε είναι και βέλτιστη λύση του αρχικού προβλήματος οπότε σταματάμε. Σε αντίθετη περίπτωση υπάρχει ανισότητα η οποία να διαχωριζει την $\mathbf{x}_{LR}^{opt}$ από οποιαδήποτε εφικτή λύση του P. Η εύρεση μίας τέτοιας ανισότητας αποτελεί το *πρόβλημα διαχωρισμού (separation problem)* ενώ η ανισότητα αυτή αποτελεί μία *τομή (cut)*. Η τομή μπορεί να προστεθεί στη χαλάρωση καθιστώντας ένα τμήμα του χώρου μη εφικτό. Το τμήμα αυτό περιλαμβάνει μόνο μη ακέραιες λύσεις, μεταξύ αυτών και την $\mathbf{x}_{LR}^{opt}$. Η διαδικασία επαναλαμβάνεται 'πετώντας' κάθε φορά τέτοια τμήματα του χώρου μέχρι τελικά να βρεθεί μία ακέραια βέλτιστη λύση.

3.1.2 Μέθοδος Διακλάδωσης και Οριοθέτησης (Branch and Bound)

Θεωρητικά, οποιοδήποτε πρόβλημα με πεπερασμένο χώρο λύσεων μπορεί να επιλυθεί μέσω της μεθόδου εξαντλητικής απαρίθμησης (exhaustive enumeration), δηλαδή εξετάζοντας μία προς μία όλες τις εφικτές λύσεις. Εν γένει, στα προβλήματα MILP ο χώρος λύσεων δεν είναι πεπερασμένος, λόγω των μη ακέραιων αγνώστων, όμως μπορούμε να 'απαριθμήσουμε' τους εφικτούς συνδυασμούς τιμών των ακέραιων μεταβλητών. Εφαρμόζοντας κάθε τέτοιο συνδυασμό τιμών στο αρχικό πρόβλημα



(α′) Αρχικό πρόβλημα *(β′) 1η τομή*

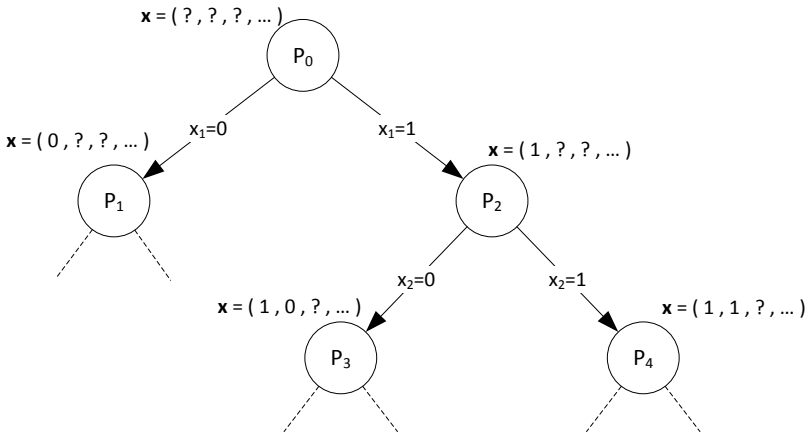
(γ′) 2η τομή *(δ′) Βέλτιστη λύση*

Σχήμα 3.1: Παράδειγμα Εφαρμογής Επίπεδων Τομών

MILP παίρνουμε αντίστοιχα ένα υποπρόβλημα αποτελούμενο μόνο από μη ακέραιους αγνώστους. Στην ουσία δηλαδή τα υποπροβλήματα που προκύπτουν είναι προβλήματα LP τα οποία μπορούμε να τα λύσουμε αποδοτικά με τη βοήθεια ενός επιλυτή LP. Σε πολλές όμως πρακτικές περιπτώσεις το πλήθος των διαφορετικών συνδυασμών είναι πολύ μεγάλο καθιστώντας αδύνατη την εξέταση όλων των λύσεων μέσα σε ένα λογικό χρονικό πλαίσιο.

Η μέθοδος Διακλάδωσης και Οριοθέτησης (Branch and Bound, B&B), που προτάθηκε από τους Land and Doig (1960), επιτυγχάνει τη μείωση του χρόνου επίλυσης αποφεύγοντας την εξερεύνηση τμημάτων του χώρου λύσεων όπου δεν είναι δυνατό να περιλαμβάνουν τη βέλτιστη λύση. Η απόρριψη των 'άχρηστων' συνόλων λύσεων βασίζεται στον υπολογισμό άνω και κάτω ορίων για τη βέλτιστη τιμή κάθε συνόλου.

Για την εξερεύνηση του χώρου αναζήτησης χρησιμοποιείται μία δενδρική δομή. Στη ρίζα του δέντρου B&B τοποθετείται το αρχικό πρόβλημα, ενώ κάθε άλλη ακμή αντιπροσωπεύει ένα υποπρόβλημα του αρχικού προβλήματος. Το δέντρο κατασκευάζεται μέσω της διαδικασίας *διακλάδωσης (branching)* των ακμών του. Μία διακλαδιζόμενη ακμή παράγει δύο ή περισσότερα υποπροβλήματα-ακμές εισάγοντας αλληλοαποκλειόμενους γραμμικούς περιορισμούς. Στην ουσία αν S ο χώρος λύσεων μίας ακμής τότε τα σύνολα λύσεων $S_1, S_2, \ldots, S_n$ των υποπροβλημάτων που προκύπτουν από τη διακλάδωση είναι μεταξύ τους ξένα και καλύπτουν το S. Συνήθως κάθε διακλάδωση αφορά μία μόνο μεταβλητή



Σχήμα 3.2: Δεντρο Διακλάδωσης και Οριοθέτησης

και δίνει δύο παιδιά, π.χ. επιλέγοντας μία δυαδική μεταβλητή x_i, προκύπτουν δύο νέες ακμές με έναν επιπλέον περιορισμό σε σχέση με τον πατέρα τους: $x_i = 0$ και $x_i = 1$ αντίστοιχα. 3.2

Ωστόσο υπάρχουν περιπτώσεις που δεν είναι απαραίτητο να διακλαδωθεί μία ακμή. Τότε λέμε ότι η ακμή *κλαδεύεται (pruning)*. Η ανίχνευση τέτοιων ακμών γίνεται μέσω της επίλυσης των γραμμικών χαλαραρώσεων των υποπροβλημάτων που ορίζουν οι ακμές ως εξής: Κατά την εκτέλεσή της η B&B διατηρεί την επικρατέστερη ή δεσπόζουσα (incumbent) λύση, έστω $\mathbf{x}_P^{inc}$, δηλαδή την καλύτερη λύση που έχει βρεθεί μέχρι στιγμής. Έστω P' το υποπρόβλημα που ορίζει μία ακμή και LR' η γραμμική του χαλάρωση. Εάν η λύση της χαλάρωσης τυχαίνει να είναι ακέραια (δηλαδή οι τιμές όλων των αγνώστων είναι ακέραιες) τότε είναι και βέλτιστη λύση του P', οπότε δε χρειάζεται να προχωρήσουμε σε διακλάδωση του κόμβου. Σε αντίθετη περίπτωση συγκρίνουμε τη βέλτιστη τιμή της χαλάρωσης $f\left(\mathbf{x}_{LR'}^{opt}\right)$ με τη δεσπόζουσα $f\left(\mathbf{x}_P^{inc}\right)$. Αν $f\left(\mathbf{x}_P^{inc}\right) < f\left(\mathbf{x}_{LR'}^{opt}\right)$ τότε λόγω της (3.1) έχουμε ότι $f\left(\mathbf{x}_P^{inc}\right) < f\left(\mathbf{x}_{P'}^{opt}\right)$ και άρα αποκλείεται να βρούμε καλύτερη λύση από τη δεσπόζουσα μέσω της ακμής αυτής. Εάν δεν ισχύει καμία από τις δύο παραπάνω υποθέσεις δε μπορούμε να αποκλείσουμε την ακμή οπότε είμαστε υποχρεωμένοι να τη διακλαδώσουμε.

3.2 Διατυπώσεις του προβλήματος MLCMST

3.2.1 Διατύπωση βασισμένη στη διατήρηση της συνολικής ροής σε κάθε κόμβο

Το πρόβλημα MLCMST μπορεί να διατυπωθεί ως πρόβλημα μεικτού ακέραιου γραμμικού προγραμματισμού. Η πρώτη γραμμική διατύπωση που παρουσιάζουμε αντιμετωπίζει τις εισερχόμενες και

εξερχόμενες ροές σε κάθε κόμβο ενιαία, ανεξαρτήτως κόμβου προέλευσης (Single Commodity Formulation - SCF). Για κάθε ακμή του γράφου G και τύπο ζεύξης ορίζουμε δυαδικές μεταβλητές απόφασης $x_{ij}^p \in \{0,1\}$ οι οποίες υποδεικνύουν εάν έχει εγκατασταθεί, ή όχι, ζεύξη τύπου l στην ακμή (i,j). Επίσης για κάθε ακμή του G ορίζουμε συνεχείς μεταβλητές $f_{ij} \geq 0$ που δηλώνει τη ροή στην ακμή (i,j). Τότε το πρόβλημα MLCMST διατυπώνεται ως εξής:

Πρόβλημα 3.3: Single Commodity Formulation (SCF)

$$\min \quad g = \sum_{(i,j)\in E} \sum_{l=1}^{L} C_{ij}^l x_{ij}^l \tag{3.3α′}$$

$$\text{s.t.} \sum_{j:(i,j)\in E} f_{ij} - \sum_{m:(m,i)\in E} f_{mi} = F_i \qquad \forall i \in V \setminus \{r\}, \tag{3.3β′}$$

$$f_{ij} \leq \sum_{l=1}^{L} Z^l x_{ij}^l \qquad \forall (i,j) \in E, \tag{3.3γ′}$$

$$\sum_{j:(i,j)\in E} \sum_{l=1}^{L} x_{ij}^l = \begin{cases} 0 & \text{if } i = r \\ 1 & \text{else} \end{cases} \qquad \forall i \in V, \tag{3.3δ′}$$

$$x_{ij}^l \in \{0,1\} \qquad \forall (i,j) \in E, l \in \Lambda, \tag{3.3ε′}$$

$$f_{ij} \geq 0 \qquad \forall (i,j) \in E, \tag{3.3στ′}$$

Ο περιορισμός (3.3β′) εκφράζει τη διαφορά εξερχόμενων - εισερχόμενων ροών σε ένα τερματικό κόμβο, η οποία ισούται με τον όγκο κίνησης F_i που παράγει αυτός. Ο περιορισμός (3.3δ′) καταρχήν αποκλείει την ύπαρξη εξερχόμενης ζεύξης από τον κεντρικό κόμβο. Επίσης εξασφαλίζει ότι θα υπάρχει ακριβώς μία εξερχόμενη ζεύξη σε κάθε τερματικό κόμβο και ότι θα επιλεγεί ένας τύπος για αυτή τη ζεύξη. Τέλος ο περιορισμός (3.3γ′) εξασφαλίζει ότι η ροή σε μία ζεύξη δε θα υπερβαίνει τη χωρητικότητα του τύπου ζεύξης που έχει εγκατασταθεί.

Είναι προφανές λοιπόν ότι μία λύση του προβλήματος MLCMST μπορεί να εκφραστεί ως λύση του προβλήματος 3.3 θέτοντας τις κατάλληλες τιμές στις μεταβλητές x_{ij}^l, f_{ij}. Αντίστροφα, για μία λύση του προβλήματος 3.3, $\{\tilde{x}_{ij}^l, \tilde{f}_{ij}\}$ ορίζουμε κατευθυνόμενο γράφο $T = (V, E^T)$ που αντιστοιχίζεται σε αυτή τη λύση με $E^T = \left\{(i,j) | \sum_{l=1}^{L} \tilde{x}_{ij}^l = 1\right\}$. Ακολουθούν προτάσεις, που δείχνουν ότι ο γράφος T είναι πάντα εφικτή λύση του MLCMST και άρα το πρόβλημα 3.3 ισοδυναμεί με το MLCMST. Η μεθοδολογία των αποδείξεων είναι ανάλογη με αυτή που χρησιμοποιήθηκε στο από τον Gavish (1982).

Πρόταση 3.1: Ο γράφος $T = \left(V, E^T\right)$ δε περιλαμβάνει κύκλους

Απόδειξη: Υποθέτουμε ότι οι κόμβοι $(i_1, i_2, \ldots, i_p, i_{p+1} = i_1)$ σχηματίζουν απλό κύκλο **c** στον T.

Για οποιοδήποτε $i_o \in \mathbf{c}$, ισχύει $(i_o, i_{o+1}) \in E^T$ και συνεπώς $\sum_{l=1}^{L} \tilde{x}^l_{i_o i_{o+1}} = 1$. Για κάθε $m \neq i_{o+1}$ προκύπτει από την (3.3δ′) ότι $\sum_{l=1}^{L} \tilde{x}^l_{i_o m} = 0$ και από την (3.3γ′) ότι $\tilde{f}_{i_o m} = 0$. Επίσης από την (3.3δ′) προκύπτει ότι $i_o \neq r$, δηλαδή ο κύκλος αποτελείται μόνο από τερματικούς κόμβους. Εφαρμόζοντας τη σχέση (3.3β′) για $i = i_o$ έχουμε

$$\begin{aligned} F_{i_o} &= \tilde{f}_{i_o i_{o+1}} - \sum_{m:(m,i_o)\in E} \tilde{f}_{m i_o} \\ &= \tilde{f}_{i_o i_{o+1}} - \tilde{f}_{i_{o-1} i_o} - \sum_{m:m\neq i_{o-1},(m,i_o)\in E} \tilde{f}_{m i_o} \end{aligned}$$

και άρα $\tilde{f}_{i_o i_{o+1}} - \tilde{f}_{i_{o-1} i_o} \geq F_{i_o}$. Αθροίζοντας για όλα τα $i \in \mathbf{c}$ έχουμε $0 \geq \sum_{i\in\mathbf{c}} F_i$ το οποίο είναι αδύνατο αφού όλα τα $F_i > 0$, και συνεπώς δεν είναι δυνατόν να υπάρχει απλός κύκλος στον T. □

Από τον περιορισμό 3.3δ′ προκύπτει ότι κάθε τερματικός κόμβος έχει ακριβώς μία εξερχόμενη ακμή στο T. Έχοντας δείξει ότι ο T δε περιλαμβάνει κύκλους οδηγούμαστε στο παρακάτω πόρισμα.

Πρόταση 3.1: Ο γράφος $T = \left(V, E^T\right)$ είναι κατευθυνόμενο δέντρο (arboresence) που καλύπτει όλους τους κόμβους του V με ρίζα τον κεντρικό κόμβο

Πρόταση 3.2: Για κάθε $(i,j) \in E$, εάν $(i,j) \notin E^T$ τότε $\tilde{f}_{ij} = 0$, αλλιώς $\tilde{f}_{ij}$ ισούται με τη ροή κίνησης στη ζεύξη (i,j).

Απόδειξη: Εάν $(i,j) \notin T$ τότε από τον ορισμό του E^T προκύπτει ότι $\sum_{l=1}^{L} \tilde{x}^l_{ij} = 0$. Από τον περιορισμό (3.3γ′) συμπεραίνουμε ότι$\tilde{f}_{ij} = 0$. Εάν $(i,j) \in T$ θεωρούμε το υποδέντρο $T_i = (V_i, E_i^T)$ που προκύπτει αν αφαιρέσουμε την ακμή (i,j) από το T και περιλαμβάνει τον i. Αθροίζοντας τον περιορισμό (3.3β′) για κάθε $k \in V_i$ προκύπτει ότι

$$\begin{aligned} \sum_{k\in V_i} F_k &= \sum_{k\in V_i} \left(\sum_{l:(k,l)\in E} \tilde{f}_{kl} - \sum_{m:(m,k)\in E} \tilde{f}_{mk} \right) \\ &= \sum_{k\in V_i} \left(\sum_{l:(k,l)\in E^T} \tilde{f}_{kl} - \sum_{m:(m,k)\in E^T} \tilde{f}_{mk} \right) \\ &= \sum_{(k,l)\in E^T:k\in V_i} \tilde{f}_{kl} - \sum_{(m,k)\in E^T:k\in V_i} \tilde{f}_{mk} \\ &= \sum_{(k,l)\in E_i^T} \tilde{f}_{kl} + \tilde{f}_{ij} - \sum_{(m,k)\in E_i^T} \tilde{f}_{mk} \\ &= \tilde{f}_{ij}. \end{aligned}$$

Συνεπώς $\tilde{f}_{ij}$ ισούται με το άθροισμα της κίνησης που παράγουν οι κόμβοι του V_i και άρα ισούται με

ροή κίνησης στην ακμή (i, j). □

Άμεση συνέπεια τις παραπάνω πρότασης και του περιορισμού (3.3γ′) είναι το πόρισμα 3.2 το οποίο σε συνδυασμό με το πόρισμα 3.1 μας οδηγεί στο πόρισμα 3.3.

Πρόταση 3.2: Η ροή κίνησης κατά μήκος μία ακμής του T δε ξεπερνά τη χωρητικότητα του τύπου ζεύξης που έχει επιλεγεί.

Πρόταση 3.3: Τα προβλήματα MLCMST και SCF είναι ισοδύναμα.

3.2.2 Διατύπωση βασισμένη στη διατήρηση των επιμέρους ροών σε κάθε κόμβο

Εναλλακτικά μπορούμε να διατυπώσουμε το πρόβλημα MLCMST αναλύοντας κάθε ροή f_{ij} σε επιμέρους ροές με βάση τον κόμβο προέλευσης (Multi Commodity Formulation - MCF). Ορίζουμε δυαδικές μεταβλητές απόφασης y_{ij}^k, οι οποίες υποδεικνύουν εάν η κίνηση που παράγει ο τερματικός κόμβος k αποτελεί μέρος της ροής f_{ij}. Αυτό συμβαίνει όταν $k \in V_i$. Με βάση τον ορισμό των y_{ij}^k προκύπτει ότι

$$f_{ij} = \sum_{k \in V \setminus \{r\}} F_k y_{ij}^k, \quad \forall (i,j) \in E. \tag{3.4}$$

Χρησιμοποιώντας τις μεταβλητές x_{ij}^l και y_{ij}^k διατυπώνουμε το πρόβλημα MLCMST ως εξής:

Πρόβλημα 3.4: Multi Commodity Formulation (MCF)

$$\min \quad g = \sum_{(i,j) \in E} \sum_{l=1}^{L} C_{ij}^l x_{ij}^l \tag{3.5α′}$$

$$\text{s.t.} \sum_{j:(i,j) \in E} y_{ij}^k - \sum_{m:(m,i) \in E} y_{mi}^k = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{else} \end{cases} \qquad \forall i, k \in V \setminus \{r\}, \tag{3.5β′}$$

$$\sum_{k \in V \setminus \{r\}} F_k y_{ij}^k \leq \sum_{l=1}^{L} Z^l x_{ij}^l \qquad \forall (i,j) \in E, \tag{3.5γ′}$$

$$\sum_{j:(i,j) \in E} \sum_{l=1}^{L} x_{ij}^l = \begin{cases} 0 & \text{if } i = r \\ 1 & \text{else} \end{cases} \qquad \forall i \in V, \tag{3.5δ′}$$

$$x_{ij}^l \in \{0,1\} \qquad \forall (i,j) \in E, l \in \Lambda, \tag{3.5ε′}$$

$$y_{ij}^k \in \{0,1\} \qquad \forall (i,j) \in E, k \in V \setminus \{r\}, \tag{3.5στ′}$$

Για δύο κόμβους $i, k \in V \setminus \{r\}$ το άθροισμα $\sum_{j:(i,j) \in E} y_{ij}^k$ υποδεικνύει αν η εξερχόμενη από τον i ροή

περιλαμβάνει την κίνηση που παράγει ο k. Αυτό συμβαίνει όταν $i = k$ ή όταν η κίνηση που παράγει ο k διέρχεται από τον i. Αντίστοιχα, το άθροισμα $\sum_{m:(m,i)\in E} y^k_{mi}$ υποδεικνύει αν η εισερχόμενη στον i ροή περιλαμβάνει την κίνηση που παράγει ο k, πράγμα που συμβαίνει όταν η κίνηση που παράγει ο k διέρχεται από τον i. Ο περιορισμός (3.5β′), ο οποίος αντικαθιστά τον περιορισμό (3.3β′) του SCF, μέσω τη διαφοράς των δύο παραπάνω αθροισμάτων εκφράζει τη σχέση μεταξύ του κόμβου i και της κίνησης που παράγει ο κόμβος k. Αν $i = k$ η διαφορά των αθροισμάτων ισούται με $1-0 = 1$. Σε κάθε άλλη περίπτωση ισούται με 0, είτε αν η κίνηση που παράγει ο k διέρχεται από τον i $(1 - 1 = 0)$, είτε όχι $(0 - 0 = 0)$. Ο περιορισμός (3.5γ′) αντικαθιστά τον περιορισμό (3.3γ′) του SCF εφαρμόζοντας την (3.4), ενώ ο περιορισμός (3.5δ′) είναι ισοδύναμος με τον περιορισμό (3.3δ′).

Μία λύση του προβλήματος MLCMST μπορεί να εκφραστεί ως λύση του προβλήματος 3.4 θέτοντας τις κατάλληλες τιμές στις μεταβλητές x^l_{ij}, y^k_{ij}. Αντίστροφα για κάθε λύση του προβλήματος 3.4, $\dot{x}^l_{ij}$, $\dot{y}^k_{ij}$, θα δείξουμε ότι είναι λύση του προβλήματος 3.3 και άρα και του MLCMST.

Πρόταση 3.3: Κάθε λύση του προβλήματος 3.4, $\dot{x}^l_{ij}$, $\dot{y}^k_{ij}$ ικανοποιεί τους περιορισμούς (3.3β′).

Απόδειξη: Πολλαπλασιάζοντας τη σχέση (3.5β′) με F_k προκύπτει ότι

$$\sum_{j:(i,j)\in E} F_k \dot{y}^k_{ij} - \sum_{m:(m,i)\in E} F_k \dot{y}^k_{mi} = \begin{cases} F_k & \text{if } k = i \\ 0 & \text{else} \end{cases}$$

Αθροίζοντας την παραπάνω σχέση για κάθε $k \in V \setminus \{r\}$ έχουμε

$$\sum_{j:(i,j)\in E} \sum_{k\in V\setminus\{r\}} F_k \dot{y}^k_{ij} - \sum_{m:(m,i)\in E} \sum_{k\in V\setminus\{r\}} F_k \dot{y}^k_{mi} = F_i$$

Εφαρμόζοντας την (3.4) για το πρώτο μέλος της εξίσωσης καταλήγουμε ότι

$$\sum_{j:(i,j)\in E} \dot{f}_{ij} - \sum_{m:(m,i)\in E} \dot{f}_{mi} = F_i \qquad \square$$

Εφαρμόζοντας τη σχέση (3.4) στον περιορισμό (3.5γ′) οδηγούμαστε εύκολα στον περιορισμό (3.3γ′). Έχοντας δείξει ότι κάθε λύση του προβλήματος MCF ικανοποιεί όλες τις συνθήκες του προβλήματος SCF, οδηγούμαστε στο συμπέρασμα ότι αποτελεί λύση και του MLCMST. Συνεπώς καταλήγουμε στο παρακάτω πόρισμα.

Πρόταση 3.4: Τα προβλήματα MLCMST και MCF είναι ισοδύναμα.

3.2.3 Ισοδυναμία των γραμμικών χαλαρώσεων των SCF και MCF

Αίροντας τους περιορισμούς ακεραιότητας των προβλημάτων SCF και MCF παίρνουμε τις γραμμικές χαλαρώσεις τους: SCF-LP και MCF-LP αντίστοιχα. Συγκεκριμένα οι άγνωστοι x_{ij}^l (και των δύο διατυπώσεων) και y_{ij}^k (της MCF-LP) παίρνουν τιμές από το σύνολο $[0, 1]$.

Ο λύσεις των χαλαρώσεων διαφέρουν σε πολλά από τις λύσεις του MLCMST. Δεδομένου ότι πλέον οι x_{ij}^l δεν είναι δυαδικοί χάνεται και η μετάφραση των τιμών τους σε ύπαρξη ή μη τύπου ζεύξης. Διαισθητικά μπορούμε να πούμε ότι εκφράζουν ένα 'ποσοστό' τύπου ζεύξης. Βάσει των περιορισμών (3.3δ΄) και (3.5δ΄) τα εξερχόμενα από ένα κόμβο ποσοστά τύπων ζεύξεων πρέπει να αθροίζουν σε 100%. Τα ποσοστά τύπων ζεύξεων φέρουν και τα αντίστοιχα ποσοστά χωρητικότητας, τα οποία δε μπορεί να υπερβεί η ροή μεταξύ δύο κόμβων. Όσον αφορά τις y_{ij}^k μπορούμε να πούμε ότι εκφράζουν το ποσοστό της κίνησης του k που διοχετεύεται από τον i στον j.

Στη συνέχεια θα δείξουμε ότι οι χαλαρώσεις SCF-LP και MCF-LP είναι ισοδύναμες, δηλαδή οποιαδήποτε από τις δύο διατυπώσεις και αν χρησιμοποιήσουμε θα λάβουμε την ίδια βέλτιστη τιμή στο χαλαρωμένο πρόβλημα. Αρκεί να δείξουμε ότι για οποιαδήποτε εφικτή λύση του SCF-LP υπάρχει εφικτή λύση του αντίστοιχου MCF-LP με το ίδιο κόστος, και το αντίστροφο.

Πρόταση 3.4: Για οποιαδήποτε εφικτή λύση του MCF-LP υπάρχει εφικτή λύση του αντίστοιχου SCF-LP με το ίδιο κόστος.
Απόδειξη: Έστω $\dot{\mathbf{x}} = \{\dot{x}_{ij}^l, \dot{y}_{ij}^k\}$ μία εφικτή λύση του MCF-LP. Κατασκευάζουμε λύση του SCF-LP, $\tilde{\mathbf{x}} = \{\tilde{x}_{ij}^l, \tilde{f}_{ij}\}$, με $\tilde{x}_{ij}^l = \dot{x}_{ij}^l$ και $\tilde{f}_{ij} = \sum_{k \in V \setminus \{r\}} F_k \dot{y}_{ij}^k$. Είναι προφανές ότι οι $\tilde{\mathbf{x}}$ και $\dot{\mathbf{x}}$ έχουν το ίδιο κόστος και ότι η $\tilde{\mathbf{x}}$ ικανοποιεί τις σχέσεις (3.3γ΄) και (3.3δ΄). Όσον αφορά την (3.3β΄) αυτή αποδεικνύεται όπως και η πρόταση 3.3. □

Στη συνέχεια θα αποδείξουμε και το αντίστροφο, δηλαδή: για οποιαδήποτε εφικτή λύση του SCF-LP υπάρχει εφικτή λύση του αντίστοιχου MCF-LP με το ίδιο κόστος. Τα βήματα που θα ακολουθήσουμε είναι τα εξής:

- Θα ορίσουμε το γράφο ροών για κάθε λύση του SCF-LP.
- Θα δείξουμε ότι για οποιαδήποτε εφικτή λύση του SCF-LP, της οποίας ο γράφος ροών περιλαμβάνει κύκλους υπάρχει εφικτή λύση του SCF-LP, της οποίας ο γράφος ροών δεν περιλαμβάνει κύκλους.
- Χρησιμοποιώντας μία λύση του SCF-LP χωρίς κύκλους θα κατασκευάσουμε μία λύση του MCF-LP με το ίδιο κόστος
- Θα δείξουμε ότι η λύση που κατασκευάσαμε είναι εφικτή.

Ορίζουμε ως *γράφο ροών* μία λύσης του SCF-LP τον κατευθυνόμενο γράφο $G(V, E^F)$ όπου $E^F = \{(i,j) : f_{ij} > 0\}$.

Πρόταση 3.5: Για οποιαδήποτε εφικτή λύση του SCF-LP, της οποίας ο γράφος ροών G περιλαμβάνει κύκλους, υπάρχει εφικτή λύση του SCF-LP, της οποίας ο γράφος ροών δεν περιλαμβάνει κύκλους

Απόδειξη: Έστω $i_1, i_2, \ldots, i_o, i_{o+1} = i_1$ κύκλος του G όπου η $f_{i_o i_1}$ είναι η μικρότερη ροή στον κύκλο. Θέτουμε νέες ροές για τις ακμές του κύκλου ώς εξής: $f'_{xy} = f_{xy} - f_{i_o i_1}$. Εφόσον $f'_{i_o,i_1} = 0$ η ακμή (i_o, i_1) δε περιλαμβάνεται στο νέο γράφο ροών G' και άρα ο κύκλος 'έσπασε'. Η νέα λύση που προκύπτει διατηρεί το ίδιο κόστος (αφού δεν αλλάξαμε τα x^l_{ij}) και είναι εφικτή διότι:

- αφαιρώντας ίσο όγκο από τις εξερχόμενες και εισερχόμενες ροές κάθε κόμβου του κύκλου διατηρείται το ισοζύγιο ροών και άρα δεν επηρεάζονται οι συνθήκες (3.3β′)
- οι αλλαγές επέφεραν μόνο μειώσεις στις ροές f_{xy} και άρα δεν παραβιάστηκε κάποια από τις συνθήκες χωρητικότητας (3.3γ′).
- οι συνθήκες (3.3δ′) δεν επηρεάζονται.

Επαναλαμβάνοντας την παραπάνω στρατηγική για κάθε κύκλο καταλήγουμε σε εφικτή λύση χωρίς κύκλους αλλά με το ίδιο κόστος. □

Έστω $\tilde{\mathbf{x}} = \{\tilde{x}^l_{ij},\ \tilde{f}_{ij}\}$ λύση του SCF-LP χωρίς κύκλους. Ο γράφος ροών της λύσης $\tilde{\mathbf{x}}$ είναι ένας DAG (directed acyclic graph). Σε οποιοδήποτε DAG μπορούμε να ορίσουμε διάταξη $\prec$ των κόμβων έτσι ώστε αν $i \succ j$ τότε υπάρχει μοναδικό μονοπάτι από τον j στον i. Επίσης, δεν υπάρχει μονοπάτι από τον i στον j και άρα δεν υπάρχει και ακμή (i,j). Άμεση συνέπεια είναι ότι:

$$i \succ j \Rightarrow \tilde{f}_{ij} = 0 \tag{3.6}$$

Με βάση τη διάταξη $\prec$ και τη λύση $\tilde{\mathbf{x}}$ ορίζουμε $\dot{\mathbf{x}} = \{\dot{x}^l_{ij},\ \dot{y}^k_{ij}\}$ με

$$\dot{x}^l_{ij} = \tilde{x}^l_{ij} \tag{3.7}$$

$$\dot{y}^k_{ij} = \frac{\tilde{f}_{ij}}{\sum\limits_{m:(i,m)\in E} \tilde{f}_{im}} \times \begin{cases} 0 & \text{if } k \succ i \\ 1 & \text{if } k = i \\ \sum\limits_{\substack{m:(m,i)\in E,\\ m\prec i}} \dot{y}^k_{mi} & \text{if } k \prec i \end{cases} \tag{3.8}$$

Από τις (3.6) και (3.8) προκύπτει ότι

$$i \succ j \Rightarrow \dot{y}^k_{ij} = 0. \tag{3.9}$$

Επίσης αν αθροίσουμε την (3.8) για όλα τα $j : (i,j) \in E$ έχουμε ότι

$$\sum_{j:(i,j)\in E} \dot{y}_{ij}^k = \begin{cases} 0 & \text{if } k \succ i \\ 1 & \text{if } k = i \\ \sum\limits_{\substack{m:(m,i)\in E,\\ m\prec i}} \dot{y}_{mi}^k & \text{if } k \prec i \end{cases} \tag{3.10}$$

Πρόταση 3.6: Η $\dot{\mathbf{x}}$ ικανοποιεί τις συνθήκες (3.5β′).

Απόδειξη: Για κάθε $i, k \in V^*$ έχουμε: Άν $k = i$:

$$\sum_{j:(i,j)\in E} \dot{y}_{ij}^k - \sum_{m:(m,i)\in E} \dot{y}_{mi}^k \overset{(3.10)}{=} 1 - 0 = 1$$

Άν $k \prec i$:

$$\sum_{j:(i,j)\in E} \dot{y}_{ij}^k \overset{(3.10)}{=} \sum_{\substack{m:(m,i)\in E,\\ m\prec i}} \dot{y}_{mi}^k \overset{(3.9)}{=} \sum_{m:(m,i)\in E} \dot{y}_{mi}^k$$

Άν $k \succ i$:

$$\sum_{j:(i,j)\in E} \dot{y}_{ij}^k \overset{(3.10)}{=} 0, \qquad \sum_{m:(m,i)\in E} \dot{y}_{mi}^k \overset{(3.9)}{=} \sum_{\substack{m:(m,i)\in E,\\ m\prec i}} \dot{y}_{mi}^k \overset{(3.8)}{=} 0 \qquad \square$$

Πρόταση 3.7: Η $\dot{\mathbf{x}}$ ικανοποιεί τις συνθήκες (3.5γ′).

Απόδειξη: Λόγω της (3.3γ′) αρκεί να δείξουμε ότι $\sum\limits_{k\in V^*} F_k \dot{y}_{ij}^k = \tilde{f}_{ij}$ για οποιαδήποτε ακμή $(i,j) \in E$. Θα δείξουμε την πρόταση με επαγωγή. Έστω i_0 ο ελάχιστος κόμβος σύμφωνα με τη διάταξη $\prec$, δηλαδή $\{m : m \prec i_0\} = \emptyset$. Για $i = i_0$ έχουμε ότι

$$\sum_{k\in V^*} F_{i_0} \dot{y}_{i_0 j}^k \overset{(3.8)}{=} F_{i_0} \dot{y}_{i_0 j}^i \overset{(3.8)}{=} \frac{F_{i_0} \tilde{f}_{i_0 j}}{\sum_{m:(i_0,m)\in E} \tilde{f}_{i_0 m}} \overset{(3.3\beta')}{=} \frac{F_{i_0} \tilde{f}_{i_0 j}}{F_{i_0} + \sum_{m:(m,i_0)\in E} \tilde{f}_{m i_0}} \overset{(3.6)}{=} \frac{F_{i_0} \tilde{f}_{i_0 j}}{F_{i_0} + 0} = \tilde{f}_{i_0 j}.$$

και άρα η πρόταση ισχύει για $i = i_0$. Για οποιοδήποτε $i \succ i_0$ και υποθέτοντας ότι η πρόταση ισχύει για κάθε $m \prec i$ έχουμε ότι

$$\begin{aligned} \sum_{k\in V^*} F_k \dot{y}_{ij}^k &\overset{(3.8)}{=} F_i \dot{y}_{ij}^i + \sum_{\substack{k\in V^*\\ k\prec i}} F_k \dot{y}_{ij}^k \\ &\overset{(3.8)}{=} \frac{F_i \tilde{f}_{ij}}{\sum_{m:(m,i)\in E} \tilde{f}_{im}} + \sum_{\substack{k\in V^*\\ k\prec i}} \left(\frac{F_k \tilde{f}_{ij}}{\sum_{m:(m,i)\in E} \tilde{f}_{im}} \times \sum_{\substack{m:(m,i)\in E,\\ m\prec i}} \dot{y}_{mi}^k \right) \\ &= \frac{\tilde{f}_{ij}}{\sum_{m:(m,i)\in E} \tilde{f}_{im}} \times \left(F_i + \sum_{\substack{m:(m,i)\in E,\ k\in V^*\\ m\prec i \qquad k\prec i}} F_k \dot{y}_{mi}^k \right) \end{aligned}$$

$$\overset{(3.3\beta')}{=} \frac{\tilde{f}_{ij}}{F_i + \sum_{m:(i,m)\in E} \tilde{f}_{mi}} \times \left(F_i + \sum_{m:(m,i)\in E} \sum_{k\in V^*} F_k \dot{y}^k_{mi} \right)$$

$$\underset{\text{υπόθεση}}{\overset{\text{επαγωγική}}{=}} \frac{\tilde{f}_{ij}}{F_i + \sum_{m:(i,m)\in E} \tilde{f}_{mi}} \times \left(F_i + \sum_{m:(m,i)\in E} \tilde{f}_{mi} \right) = \tilde{f}_{ij} \qquad \square$$

3.2.4 Διατυπώσεις με τεχνητούς τύπους ζεύξεων

Οι Martins et al. (2005, 2009) παρουσίασαν μία διατύπωση του MLCMST που χαρακτήρισαν ως *capacity-indexed*. Η διατύπωση αυτή προϋποθέτει ότι $Z^1 = 1$ και οι υπόλοιποι τύποι ζεύξεων προκύπτουν με μοναδιαίες αυξήσεις χωρητικότητας από το 1 έως το Z^L. Άμεση συνέπεια της παραπάνω προϋπόθεσης είναι ο δείκτης ενός τύπου να ισούται με τη χωρητικότητά του (εξ ου και ο χαρακτηρισμός capacity-indexed). Στη γενική περίπτωση όμως η προϋπόθεση αυτή δεν ισχύει, οπότε εισάγουμε $P = Z^L$ τεχνητούς τύπους έτσι ώστε $\bar{Z}^1 = 1, \bar{Z}^2 = 2, \ldots, \bar{Z}^P = Z^L$. Το κόστος εγκατάστασης ενός τεχνητού τύπου p στην ακμή (i, j) ορίζεται ως

$$\bar{C}^p_{ij} = C^q_{ij}, \quad q = \min\{l \in \Lambda : Z^l \geq p\},$$

δηλαδή ισούται με το κόστος του τύπου q του αρχικού προβλήματος, ο οποίος έχει την ελάχιστη δυνατή ίση ή μεγαλύτερη χωρητικότητα από τον p.

Z^l :		2			5				9
C^l_{ij} :		1			2				3
$\bar{Z}^p$:	1	2	3	4	5	6	7	8	9
$\bar{C}^p_{ij}$:	1	1	2	2	2	3	3	3	3

Πίνακας 3.2: Παράδειγμα τεχνητών τύπων

Ο πίνακας 3.2 απεικονίζει την εφαρμογή τεχνητών τύπων σε ένα πρόβλημα με τρεις πραγματικούς τύπους χωρητικότητας $2, 5, 9$ και κόστος εγκατάστασης για συγκεκριμένη ακμή $1, 2, 3$ αντίστοιχα. Το σύνολο των τεχνητών τύπων ζεύξεων, όπως και το αρχικό, μπορεί να καλύψει ζεύξεις με ροή από 1 έως και Z^L. Η διαφοροποίησή τους έγκειται στο γεγονός ότι για οποιαδήποτε ροή $f_{ij} \leq Z^L$ υπάρχει τεχνητός τύπος που να έχει χωρητικότητα ακριβώς ίση με τη ροή. Συνεπώς οι ανισότητες που περιόριζαν την επιλογή τύπου με βάση τη ροή μπορούν να αντικατασταθούν με ισότητες.

Για τις διατυπώσεις SCF και MCF έχουμε τη μετατροπή των ανισοτήτων (3.3γ′) και (3.5γ′) στις παρακάτω ανισότητες αντίστοιχα.

$$f_{ij} = \sum_{p=1}^{P} \bar{Z}^p \bar{x}^p_{ij} \qquad \forall (i, j) \in E$$

$$\sum_{k\in V\setminus\{r\}} F_k y_{ij}^k = \sum_{p=1}^{P} \bar{Z}^p \bar{x}_{ij}^p \qquad \forall (i,j)\in E.$$

Επίσης, οι μεταβλητές f_{ij} της διατύπωσης SCF μπορούν να απαλειφθούν εφαρμόζοντας την παραπάνω ισότητα στον περιορισμό (3.3β′), και άρα να καταλήξουμε στον περιορισμό

$$\sum_{j:(i,j)\in E}\sum_{p=1}^{P} \bar{Z}_{ij}^p \bar{x}_{ij}^p - \sum_{m:(m,i)\in E}\sum_{p=1}^{P} \bar{Z}_{mi}^l \bar{x}_{mi}^p = F_i \qquad \forall i \in V\setminus\{r\}.$$

Με βάση τα παραπάνω το πρόβλημα MLCMST μπορεί επίσης να οριστεί με τις παρακάτω διατυπώσεις.

Πρόβλημα 3.5: Capacity-Indexed Single Commodity Formulation (CISCF)

$$\min \sum_{(i,j)\in E}\sum_{p=1}^{P} \bar{C}_{ij}^p \bar{x}_{ij}^p \tag{3.11α′}$$

$$\text{s.t.} \sum_{j:(i,j)\in E}\sum_{p=1}^{P} \bar{Z}^p \bar{x}_{ij}^p - \sum_{m:(m,i)\in E}\sum_{p=1}^{P} \bar{Z}^l \bar{x}_{mi}^p = F_i \qquad \forall i \in V\setminus\{r\}, \tag{3.11β′}$$

$$\sum_{j:(i,j)\in E}\sum_{p=1}^{P} \bar{x}_{ij}^p = \begin{cases} 0 & \text{if } i = r \\ 1 & \text{else} \end{cases} \qquad \forall i \in V, \tag{3.11γ′}$$

$$\bar{x}_{ij}^p \in \{0,1\} \qquad \forall (i,j)\in E, p\in\{1,2,\ldots,P\}, \tag{3.11δ′}$$

Πρόβλημα 3.6: Capacity-Indexed Multi Commodity Formulation (CIMCF)

$$\min \sum_{(i,j)\in E}\sum_{p=1}^{P} \bar{C}_{ij}^p \bar{x}_{ij}^p \tag{3.12α′}$$

$$\text{s.t.} \sum_{j:(i,j)\in E} y_{ij}^k - \sum_{m:(m,i)\in E} y_{mi}^k = \begin{cases} 1 & \text{if } i = k \\ 0 & \text{else} \end{cases} \qquad \forall i,k \in V\setminus\{r\}, \tag{3.12β′}$$

$$\sum_{k\in V\setminus\{r\}} F_k y_{ij}^k = \sum_{p=1}^{P} \bar{Z}^p \bar{x}_{ij}^p \qquad \forall (i,j)\in E, \tag{3.12γ′}$$

$$\sum_{j:(i,j)\in E}\sum_{p=1}^{P} \bar{x}_{ij}^p = \begin{cases} 0 & \text{if } i = r \\ 1 & \text{else} \end{cases} \qquad \forall i \in V, \tag{3.12δ′}$$

$$\bar{x}_{ij}^p \in \{0,1\} \qquad \forall (i,j)\in E, p\in\{1,2,\ldots,P\}, \tag{3.12ε′}$$

$$y_{ij}^k \in \{0,1\} \qquad \forall (i,j)\in E, k\in V\setminus\{r\}, \tag{3.12στ′}$$

3.2.5 Ενίσχυση των διατυπώσεων

Στην ενότητα 3.1 αναφερθήκαμε στο ρόλο των γραμμικών χαλαρώσεων και στη χρησιμότητα MILP διατυπώσεων που να δίνουν ισχυρές γραμμικές χαλαρώσεις. Οι χαλαρώσεις των διατυπώσεων του προβλήματος MLCMST που παρουσιάσαμε στις προηγούμενες ενότητες μπορούν να ενισχυθούν με την εισαγωγή επιπλέον περιορισμών. Οι περιορισμοί αυτοί θα πρέπει να είναι πλεονάζοντες (redundants) δηλαδή να μην αποκλείουν ακέραιες λύσεις εκτός από αποδεδειγμένα μη βέλτιστες.

Κάθε μία από τις γραμμικές διατυπώσεις του MLCMST περιλαμβάνει περιορισμούς οι οποίοι προσδιορίζουν την ισορροπία εξερχόμενων και εισερχόμενων ροών στους τερματικούς κόμβους (περιορισμοί β'). Μπορούμε να ενισχύσουμε περαιτέρω τις διατυπώσεις εισάγοντας περιορισμούς και για την ισορροπία ροών στον κεντρικό κόμβο. Δεδομένου ότι ο κεντρικός κόμβος πρακτικά συγκεντρώνει όλες τις ροές προκύπτουν οι παρακάτω περιορισμοί.

$$\sum_{m:(m,r)\in E} f_{mr} = \sum_{k\in V\setminus\{r\}} F_k \qquad \text{(for SCF)} \qquad (3.13\alpha')$$

$$\sum_{m:(m,r)\in E}\sum_{p=1}^{P} \bar{Z}^l \bar{x}^p_{mr} = \sum_{k\in V\setminus\{r\}} F_k \qquad \text{(for CISCF)} \qquad (3.13\beta')$$

$$\sum_{(m,r)\in E} y^k_{mr} = 1 \qquad \forall k \in V\setminus\{r\} \qquad \text{(for MCF, CIMCF)} \qquad (3.13\gamma')$$

Ακολούθως, παρατηρούμε ότι μία λύση του προβλήματος δεν είναι δυνατόν να περιλαμβάνει δύο ζεύξεις αντίθετης κατεύθυνσης μεταξύ δυο κόμβων. Αυτό εξασφαλίζεται, από τη μη ύπαρξη κύκλων στο γράφο-λύση (πρόταση 3.1). Από την παραπάνω διαπίστωση προκύπτει ο παρακάτω επιπλέον περιορισμός.

$$\sum_{l=1}^{L} \left(x^l_{ij} + x^l_{ji}\right) \le 1, \quad \forall i,j \in V : (i,j),(j,i) \in E \qquad \text{(for SCF, MCF)} \qquad (3.14\alpha')$$

$$\sum_{p=1}^{P} \left(x^p_{ij} + x^p_{ji}\right) \le 1, \quad \forall i,j \in V : (i,j),(j,i) \in E \qquad \text{(for CISCF, CIMCF)} \qquad (3.14\beta')$$

Εν συνεχεία θα ενδυναμώσουμε τους περιορισμούς χωρητικότητας των διατυπώσεων. Υποθέτουμε ότι έχουμε εγκαταστήσει μία ζεύξη τύπου l στην ακμή (i,j). Αυτό συνεπάγεται ότι υπάρχει ροή στη ζεύξη αυτή, αφού τουλάχιστον η κίνηση που παράγει ο κόμβος i τη διατρέχει, δηλαδή $f_{ij} \ge 1$. Στην περίπτωση που $l > 1$ τότε είναι ασφαλές να υποθέσουμε ότι $f_{ij} \ge Z^{l-1} + 1$. Το παραπάνω ισχύει διότι αν η ροή f_{ij} ήταν μικρότερη ή ίση με Z^{l-1}, τότε θα μπορούσαμε να χρησιμοποιήσουμε τον τύπο $l-1$ αντί του l και συνεπώς η λύση που εξετάζουμε είναι μη βέλτιστη. Οι παραπάνω παρατηρήσεις

οδηγούν στους παρακάτω περιορισμούς για τις διατυπώσεις SCF και MCF αντίστοιχα.

$$f_{ij} \geq x_{ij}^1 + \sum_{l=2}^{L} \left(Z^{l-1}+1\right) x_{ij}^l \qquad \forall (i,j) \in E \qquad \text{(for SCF)} \qquad (3.15\alpha')$$

$$\sum_{k \in V \setminus \{r\}} F_k y_{ij}^k \geq x_{ij}^1 + \sum_{l=2}^{L} \left(Z^{l-1}+1\right) x_{ij}^l \qquad \forall (i,j) \in E. \qquad \text{(for MCF)} \qquad (3.15\beta')$$

Η ροή μίας ζεύξης δε μπορεί να υπερβαίνει τη χωρητικότητα του τύπου ζεύξης που έχει εγκατασταθεί σε αυτή. Ωστόσο εαν η ζεύξη, έστω (i,j), έχει κατεύθυνση προς ένα τερματικό κόμβο μπορούμε να περιορίσουμε περεταίρω την τιμή της ροής. Παρατηρούμε ότι η ροή στη ζεύξη (i,j) δε μπορεί να υπερβαίνει την τιμή $Z^L - F_j$. Κάτι τέτοιο θα ήταν αδύνατο αφού θα συνεπαγόταν ότι η ροή στην εξερχόμενη από τον j ζεύξη θα ήταν μεγαλύτερη από Z^L, μίας που θα περιλάμβανε περιλαμβάνει τη ροή της (i,j) αλλά και την παραγόμενη από τον j κίνηση. Συνεπώς εάν η ζεύξη (i,j) είναι τύπου l, η ροής σε αυτή θα είναι το πολύ $\min\{Z^l, Z^L - F_j\}$. Η διαπίστωση οδηγεί σε αντικατάσταση των περιορισμών (3.3γ′) του SCF και (3.5γ′) του MCF για τις περιπτώσεις όπου $j \neq r$ με τις (3.16α′) και (3.16β′). Στην περίπτωση διατυπώσεων με τεχνητούς τύπους θα πρέπει να αποκλείσουμε τους τύπους από τον $P - F_j + 1$ μέχρι και τον P (περιορισμό (3.16γ′)).

$$f_{ij} \leq \sum_{l=1}^{L} \min\{Z^l, Z^L - F_j\} x_{ij}^l \quad \forall (i,j) \in E, j \neq r \qquad \text{(for SCF)} \quad (3.16\alpha')$$

$$\sum_{k \in V \setminus \{r\}} F_k y_{ij}^k \leq \sum_{l=1}^{L} \min\{Z^l, Z^L - F_j\} x_{ij}^l \quad \forall (i,j) \in E, j \neq r \qquad \text{(for MCF)} \quad (3.16\beta')$$

$$\sum_{p=P-F_j+1}^{P} x_{ij}^p = 0 \quad \forall (i,j) \in E, j \neq r \quad \text{(for CISCF, CIMCF)} \quad (3.16\gamma')$$

Οι διατυπώσεις MCF και CIMCF μπορούν να ενισχυθούν περεταίρω από την παρατήρηση ότι η κίνηση που παράγει ένας κόμβος, έστω k, μπορεί να περνά από ένα κόμβο i προς ένα κόμβο j μόνο εάν υπάρχει εγκατεστημένη ζεύξη μεταξύ των i,j. Το σκεπτικό αυτό μας δίνει τον παρακάτω περιορισμό για κάθε ζεύγος i,j:

$$y_{ij}^k \leq \sum_{l=1}^{L} x_{ij}^l \quad \forall (i,j) \in E, k \in V \setminus \{r\}. \qquad \text{(for MCF, CIMCF)} \qquad (3.17)$$

Πρόταση 3.8: Κάθε λύση του προβλήματος MCF, ικανοποιεί τη σχέση (3.17).

Απόδειξη: Έστω ότι υπάρχει τριάδα i', j', k' για την οποία η σχέση (3.17) δεν ισχύει, δηλαδή $\sum_{l=1}^{L} x_{i'j'}^l < y_{i'j'}^{k'}$. Αυτό συνεπάγεται ότι $y_{i'j'}^{k'} = 1$ (i) και $\sum_{l=1}^{L} x_{i'j'}^l = 0$ (ii). Εφαρμόζοντας τη

σχέση (3.5γ′) για τα i', j' έχουμε ότι

$$\sum_{l=1}^{L} Z^l x^l_{i'j'} \geq \sum_{k \in V \setminus \{r\}} F_k y^k_{i'j'} \overset{(i)}{\geq} F_{k'}.$$

Όμως, λόγω της (ii) έχουμε ότι $\sum_{l=1}^{L} Z^l x^l_{i'j'} = 0$ και άρα $0 \geq F_{k'}$. Άτοπο. □

3.3 Υπολογιστικά αποτελέσματα

Στόχος των πειραμάτων είναι η αξιολόγηση των διαφορετικών διατυπώσεων του προβλήματος ML-CMST που παρουσιάστηκαν σε αυτό το κεφάλαιο καθώς επίσης και η επίδραση των ενισχύσεων στις διατυπώσεις.

Στον πίνακα 3.3 παρουσιάζονται οι εξεταζόμενες διατυπώσεις. Σε αυτές περιλαμβάνονται καταρχήν οι τέσσερεις 'βασικές' διατυπώσεις: SCF, MCF, CISCF και CIMCF. Εν συνεχεία χωρίζουμε τους περιορισμούς ισχυροποίησης που παρουσιάστηκαν στην ενότητα 3.2.5 σε δύο ομάδες: η πρώτη ομάδα περιλαμβάνει τους περιορισμούς (3.13), (3.14), (3.15) και (3.16) ενώ η δεύτερη τους περιορισμούς (3.17). Προσθέτοντας σε κάθε μία από τις βασικές διατυπώσεις τους περιορισμούς της πρώτης ομάδας προκύπτουν οι αντίστοιχες ισχυροποιημένες διατυπώσεις SCF2, MCF2, CISCF2 και CIMCF2. Εν συνεχεία, προσθέτοντας και τους περιορισμούς τις δεύτερης ομάδας στις MCF2 και CIMCF2 προκύπτουν οι διατυπώσεις MCF3 και CIMCF3 αντίστοιχα.

	εξεταζόμενες διατυπώσεις			
Βασικές διατυπώσεις :	SCF	MCF	CISCF	CIMCF
+ περιορισμοί (3.13), (3.14), (3.15), (3.16):	SCF2	MCF2	CISCF2	CIMCF2
+ περιορισμοί (3.17):	-	MCF3	-	CIMCF3

Πίνακας 3.3: Εξεταζόμενες διατυπώσεις του προβλήματος MLCMST

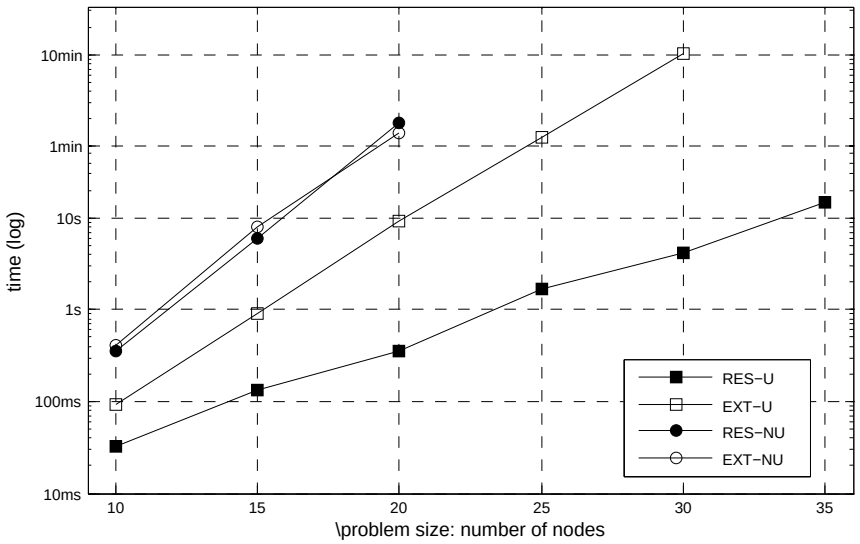
Επιλέγουμε να διαχωρίσουμε τους περιορισμούς (3.17) από τους υπόλοιπους για δύο λόγους. Αφενός, διότι εφαρμόζονται μόνο σε διατυπώσεις βασισμένες στη διατήρηση των επιμέρους ροών σε κάθε κόμβο (multi commodity formulations). Αφετέρου, διότι η αύξηση που προκαλούν στο μέγεθος των γραμμικών μοντέλων είναι δραματικά υψηλότερη - σε ένα πρόβλημα με N τερματικούς κόμβους η πρώτη ομάδα περιλαμβάνει $\mathcal{O}(N^2)$ πρόσθετους περιορισμούς ενώ η δεύτερη $\mathcal{O}(N^3)$.

Για τα υπολογιστικά πειράματα του κεφαλαίου χρησιμοποιήθηκε το πακέτο βελτιστοποίησης ILOG CPLEX (Academic Research Edition v12.1). Οι κλήσεις έγιναν μέσω της διεπαφής της CPLEX για Java, η στρατηγική που επιλέχθηκε ήταν η Δυναμική Αναζήτηση (βλέπε κεφ. 5) και σε κάθε εκτέλεση τέθηκε όριο μνήμης 1280 Mbytes.

σενάριο	RES-U						RES-NU		
διατύπωση	$N=10$	$N=15$	$N=20$	$N=25$	$N=30$	$N=35$	$N=10$	$N=15$	$N=20$
SCF	53	638	3.250	15.425	52.679	-	354	5.921	209.510
SCF2	49	390	1.562	9.780	31.399	150.460	391	6.819	110.292
MCF	163	3.870	104.779	-	-	-	468	13.721	518.908
MCF2	269	6.912	105.387	-	-	-	531	20.897	-
MCF3	285	19.336	-	-	-	-	641	33.141	-
CISCF	36	133	355	1.674	4.203	14.715	-	-	-
CISCF2	33	132	380	2.106	4.395	14.746	62.047	-	-
CIMCF	315	6.281	-	-	-	-	6.365	-	-
CIMCF2	199	6.005	-	-	-	-	5.979	-	-
CIMCF3	293	14.974	-	-	-	-	7.662	-	-
best	33	132	355	1.674	4.203	14.715	354	5.921	110.292

σενάριο	EXT-U					EXT-NU		
διατύπωση	$N=10$	$N=15$	$N=20$	$N=25$	$N=30$	$N=10$	$N=15$	$N=20$
SCF	110	1.751	12.151	76.484	-	407	7.993	82.970
SCF2	93	894	9.179	75.891	618.769	424	7.952	97.267
MCF	211	5.324	198.307	-	-	426	20.694	469.691
MCF2	483	9.474	272.515	-	-	734	29.018	776.749
MCF3	493	18.135	-	-	-	576	38.119	-
CISCF	2.057	5.991	-	-	-	-	-	-
CISCF2	2.046	7.859	-	-	-	-	-	-
CIMCF	278	6.622	-	-	-	13.567	-	-
CIMCF2	370	11.242	-	-	-	11.805	-	-
CIMCF3	567	20.416	-	-	-	-	-	-
best	93	894	9.179	75.891	618.769	407	7.952	82.970

Πίνακας 3.4: Μέσοι χρόνοι επίλυσης MILP μοντέλων (ms)



Σχήμα 3.3: Μέσοι χρόνοι επίλυσης MILP προβλημάτων (βέλτιστες τιμές μεταξύ των διατυπώσεων)

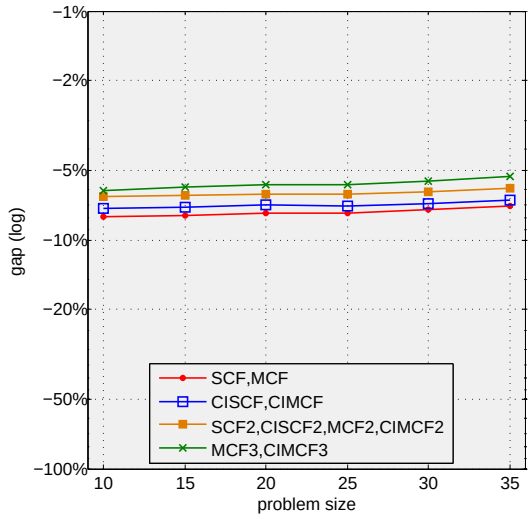
Επίλυση των μοντέλων μεικτού ακέραιου γραμμικού προγραμματισμού

Σε πρώτη φάση εξετάζουμε σε ποιο βαθμό και με τι υπολογιστικό κόστος είναι δυνατή η πλήρης επίλυση προβλημάτων MLCMST μέσω της CPLEX. Για κάθε ζεύγος σεναρίου (κεφ. 2.8) και εξεταζόμενης διατύπωσης ακολουθούμε την εξής διαδικασία: Ξεκινάμε επιχειρώντας να λύσουμε όλα τα στιγμιότυπα του μικρότερου μεγέθους ($N = 10$). Εφόσον επιλύθηκαν όλα τα προβλήματα συνεχίζουμε με το σύνολο στιγμιότυπων του αμέσως μεγαλύτερου μεγέθους. Σε περίπτωση που δεν είναι δυνατή η επίλυση κάποιου στιγμιότυπου λόγω εξάντλησης της διαθέσιμης μνήμης τερματίζεται η διαδικασία για το συγκεκριμένο ζεύγος σεναρίου-διατύπωσης. Οι μέσοι χρόνοι επίλυσης των 50 στιγμιότυπων για κάθε σύνολο στιγμιότυπων και διατύπωση παρατίθενται στον πίνακα 3.4. Με '-' δηλώνεται συνδυασμός για τον οποίο δεν κατέστει δυνατή η επίλυση όλων των στιγμιότυπων του συνόλου.

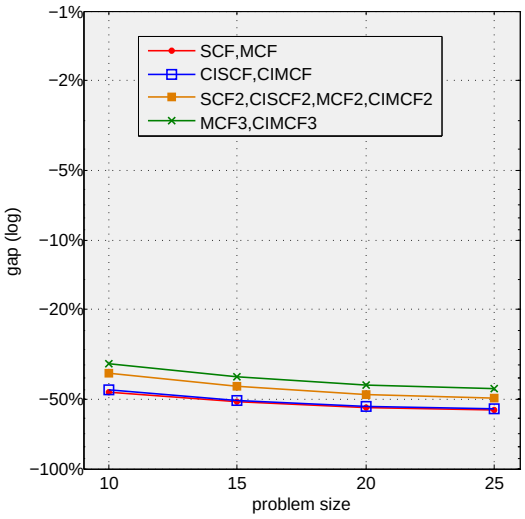
Παρατηρούμε καταρχήν ότι το σενάριο RES-U αποδείχτηκε πιο 'εύκολο' καθότι το μέγιστο μέγεθος προβλημάτων για το οποίο επιλύθηκαν όλα τα στιγμιότυπα από τουλάχιστον μία διατύπωση ήταν 35. Αντίστοιχα, στο EXT-U φτάσαμε στους 30 κόμβους ενώ στα δύο μη μοναδιαία σενάρια μη μοναδιαία, RES-NU και EXT-NU, φτάσαμε μόνο στους 20 κόμβους. Παρόμοιο συμπέρασμα μπορούμε να εξάγουμε και από τους χρόνους εκτέλεσης όπου σχεδόν για όλους τους συνδυασμούς διατύπωσης-μεγέθους οι χρόνοι επίλυσης για το RES-U είναι χαμηλότεροι από αυτούς των άλλων σεναρίων με τους RES-NU και EXT-NU να εμφανίζουν και πάλι τα χειρότερα αποτελέσματα.

Συγκρίνοντας τους χρόνους μεταξύ των διατυπώσεων παρατηρούμε ότι τα αποτελέσματα των τεσσάρων βασικών διατυπώσεων εμφανίζουν σε γενικές γραμμές μικρές αποκλίσεις από τα αντίστοιχα των πρώτων ισχυροποιήσεων τους (xCF2). Επίσης, το ζεύγος SCF, SCF2 επιτυγχάνει τους καλύτερους χρόνους για οποιαδήποτε μέγεθος προβλήματος σε όλα τα σενάρια εκτός από το RES-U όπου υπερισχύει το ζεύγος CISCF, CISCF2.

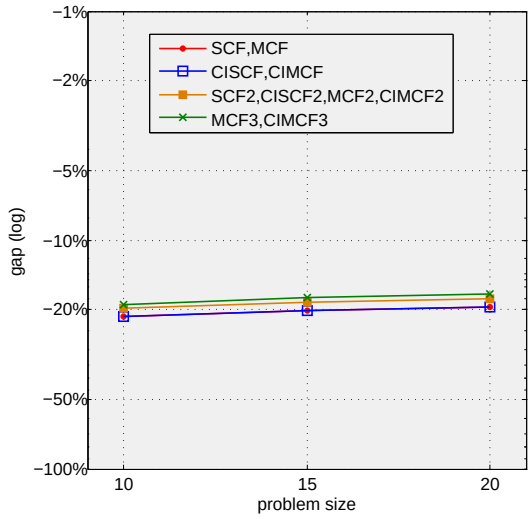
Όλες οι capacity-indexed διατυπώσεις δίνουν συγκρίσιμα και σε πολλές περιπτώσεις καλύτερα αποτελέσματα από τις αντίστοιχες μη capacity-indexed διατυπώσεις στο σενάριο RES-U. Η εικόνα αυτή όμως αντιστρέφεται στα άλλα σενάρια. Ειδικότερα, στο σενάριο EXT-U δεν είναι επιτυγχάνεται η επίλυση προβλημάτων μεγαλύτερων των 15 κόμβων ενώ στα NU σενάρια επιλύονται μόνο τα στιγμιότυπα 10 κόμβων. Η επιδείνωση αυτή αποδίδεται στο μέγεθος των διατυπώσεων αυτών. Οι capacity-indexed διατυπώσεις δημιουργούν τεχνητούς τύπους ζεύξεων όσες και οι δυνατές στάθμες χωρητικότητας στο πρόβλημα δηλαδή Z^L στο πλήθος (πίνακας 2.6). Αυτό σημαίνει ότι έχουμε μία αύξηση του μεγέθους του μοντέλου σε σχέση με την αντίστοιχη μη capacity-indexed διατύπωση. Στο σενάριο RES-U η αύξηση αυτή είναι μικρή καθότι από 3 πραγματικούς τύπους κατασκευάζονται



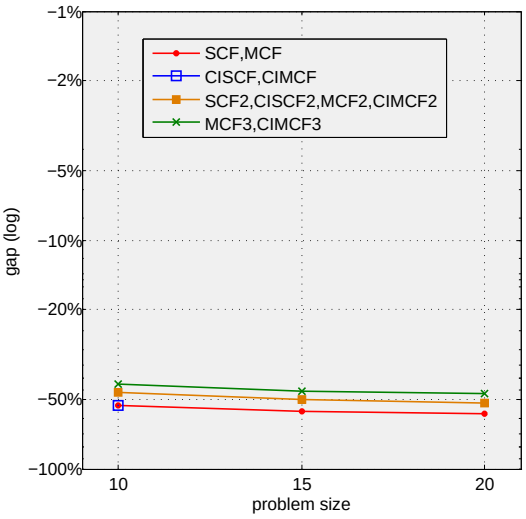
(α′) σενάριο RES-U



(β′) σενάριο EXT-U



(γ′) σενάριο RES-NU



(δ′) σενάριο EXT-NU

Σχήμα 3.4: Αποστάσεις κάτω ορίων από τις βέλτιστες τιμές

10 τεχνητοί. Στα άλλα τρία σενάρια όμως η αναλογία αυτή επιδεινώνεται με αποκορύφωμα το σενάριο EXT-NU όπου οι 6 πραγματικοί τύποι αντικαθίστανται από 6000 τεχνητούς.

Στο σχήμα 3.3 απεικονίζονται οι μέσοι χρόνοι επίλυσης για κάθε σενάριο συνάρτηση του πλήθους των κόμβων. Για κάθε σημείο έχει επιλεγεί ο χαμηλότερος χρόνος μεταξύ των διαφορετικών διατυπώσεων Η αύξηση των χρόνων επίλυσης είναι εμφανώς εκθετική. Πέντε επιπλέον κόμβοι φαίνεται να αυξάνουν το χρόνο επίλυσης κατά περίπου 3,3 φορές για το σενάριο RES-U. Οι αντίστοιχες τιμές για τα σενάριο EXT-U είναι 10 και για τα RES-NU και EXT-NU περίπου 15.

Επίλυση των γραμμικών χαλαρώσεων των προβλημάτων

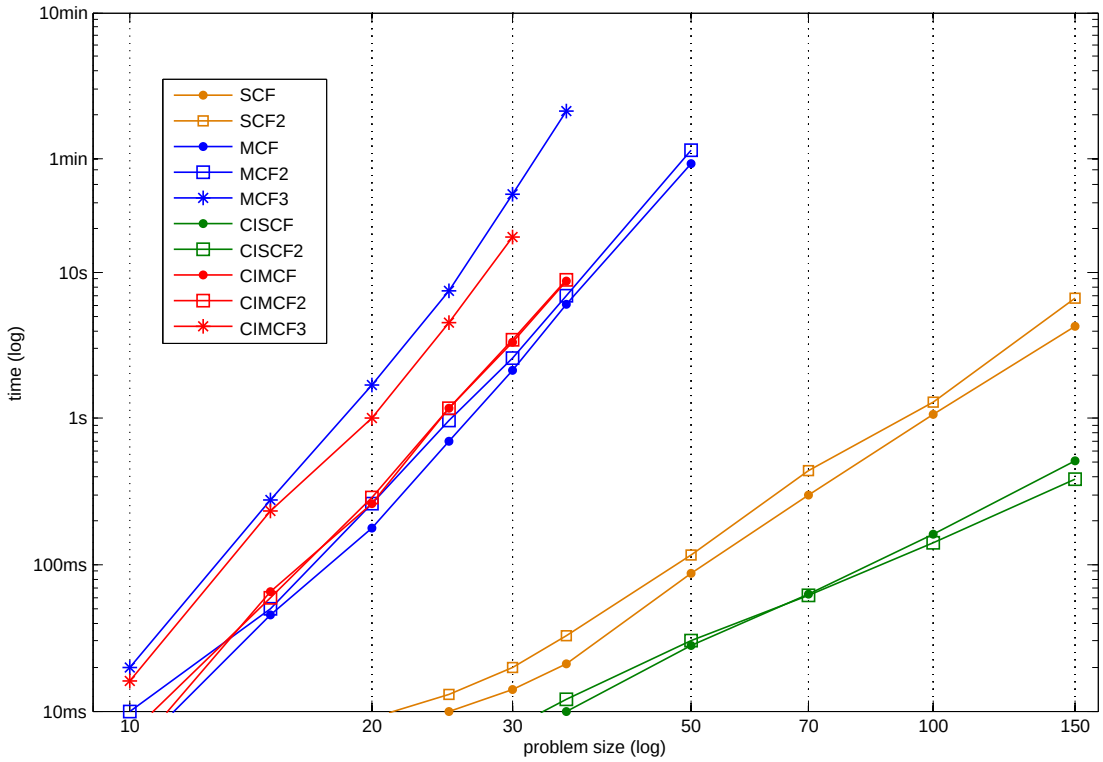
Εν συνεχεία επιχειρήθηκε η επίλυση των γραμμικών χαλαρώσεων των προβλημάτων ώστε να εξαχθούν κάτω όρια (LBs) των βέλτιστων τιμών. Προκειμένου να αξιολογήσουμε την ποιότητα των εξαχθέντων ορίων υπολογίσαμε τις σχετικές αποστάσεις τους από τις βέλτιστες τιμές για τα μεγέθη όπου οι βέλτιστες λύσεις έχουν βρεθεί. Οι μέσες σχετικές αποστάσεις από τις βέλτιστες τιμές απεικονίζονται στο σχήμα 3.4 [1].

Παρατηρούμε καταρχήν ότι ανεξαρτήτως σεναρίου αρκετές διατυπώσεις δίνουν μεταξύ τους τα ίδια κάτω όρια. Με βάση τη διαπίστωση αυτή οι διατυπώσεις μπορούν να χωριστούν σε 4 ομάδες. Όπως δείξαμε στο κεφάλαιο 3.2.3, μία SCF διατύπωση δίνει τις ίδιες τιμές με την αντίστοιχη της MCF για οποιοδήποτε σενάριο και μέγεθος. Το γεγονός αυτό επιβεβαιώνεται από τα πειραματικά αποτελέσματα.

Οι αποστάσεις των κάτω ορίων από τις βέλτιστες για το σενάριο RES-U κυμαίνονται μεταξύ 5% και 8%. Αντίθετα στα υπόλοιπα σενάρια δεν εξάγονται κάτω όρια καλής ποιότητας. Για το RES-NU υπερβαίνουν ακόμα και στην καλύτερη των περιπτώσεων το 17%, ενώ για τα RES-NU και EXT-NU οι αποστάσεις υπερβαίνουν το 34% και 42% αντίστοιχα. Οι καμπύλες δε φαίνεται να επηρεάζονται σημαντικά από τα μεγέθη των προβλημάτων και είναι λογικό να συμπεράνουμε ότι δε θα έχουμε μεγάλες διαφοροποιήσεις σε μεγαλύτερα προβλήματα. Συμπερασματικά, τα κάτω όρια σε όλα τα σενάρια πλην RES-U είναι πρακτικά μη αξιοποιήσιμα και δε θα έχει ιδιαίτερο νόημα η σύγκριση ευρετικών λύσεων με αυτά.

Όσον αφορά τους χρόνους επίλυσης των γραμμικών χαλαρώσεων, στο σχήμα 3.5 απεικονίζονται οι μέσες τιμές για το σενάριο RES-U [2]. Για τις MCF διατυπώσεις δε κατέστει δυνατή η επίλυση αρκετών συνόλων στιγμιότυπων με περισσότερους από 30 κόμβους. Παρατηρούμε ότι οι δύο CISCF διατυπώσεις δίνουν τους καλύτερους χρόνους, με τις αντίστοιχες SCF να ακολουθούν.

[1] Οι τιμές των γραφημάτων παρατίθενται στον πίνακα Α΄.1 του Παραρτήματος.
[2] Οι χρόνοι επίλυσης των γραμμικών χαλαρώσεων για όλα τα σενάρια παρατίθενται στον πίνακα Α΄.2 του Παραρτήματος.



Σχήμα 3.5: Μέσοι χρόνοι επίλυσης γραμμικών χαλαρώσεων για το σενάριο RES-U

Κεφάλαιο 4

Ευρετικοί αλγόριθμοι αναβαθμίσεων

4.1 Αναβαθμίσεις κόμβων

Μία αναβάθμιση δεν είναι παρά ένα σύνολο τροποποιήσεων, οι οποίες εφαρμοζόμενες στην εκάστοτε τρέχουσα λύση $T(V, E_T)$ δίνουν μία διαφορετική εφικτή λύση $T'(V, E'_T)$. Συγκεκριμένα μία αναβάθμιση συνίσταται στην αντικατάσταση του εγκατεστημένου τύπου σε μία ζεύξη με τύπο μεγαλύτερης χωρητικότητας και την επαναδρομολόγηση άλλων κόμβων μέσω της ζεύξης αυτής. Στόχος είναι η εύρεση *επικερδών* αναβαθμίσεων, δηλαδή αναβαθμίσεων που μειώνουν το συνολικό κόστος.

Για την περιγραφή των αναβαθμίσεων θα χρησιμοποιήσουμε τους ορισμούς της ενότητας 2.1 ως προς την τρέχουσα λύση T. Έστω $i \in V^*$ ένας τερματικός κόμβος του προβλήματος. Η ζεύξη μεταξύ του κόμβου i και του πατέρα του p_i είναι τύπου λ_i και κοστίζει $C_{ip_i}^{\lambda_i}$. Αντικαθιστώντας τον τύπο της συγκεκριμένης ζεύξης με άλλο τύπο $l > \lambda_i$ μεγαλύτερης χωρητικότητας το κόστος αυξάνεται κατά

$$C_{ip_i}^{l} - C_{ip_i}^{\lambda_i},$$

όμως η ζεύξη πλέον έχει ένα πλεόνασμα χωρητικότητας, το οποίο θα μπορεί να χρησιμοποιηθεί για τη δρομολόγηση της κίνησης άλλων τερματικών κόμβων. Αν για παράδειγμα αντικαταστήσουμε τη ζεύξη ενός τερματικού κόμβου j προς τον πατέρα του p_j με ζεύξη προς τον i ίσης χωρητικότητας θα έχουμε μία μεταβολή του κόστους

$$d_{ij} = C_{ji}^{\lambda_j} - C_{jp_j}^{\lambda_j}.$$

Αν $d_{ij} < 0$ τότε η επανασύνδεση του j στον i χαρακτηρίζεται επικερδής. Επανασυνδέοντας ένα σύνολο κόμβων $H \subset V^*$, αντί ενός μεμονωμένου κόμβου, στον i επιτυγχάνουμε μεταβολή κόστους

ίση με $\sum_{j\in H} d_{ij}$, και άρα η συνολική μεταβολή από την αναβάθμιση -αλλαγή τύπου ζεύξης (i, p_i) και επανασυνδέσεις- ισούται με

$$D_{iH}^{l} = \left(C_{ip_i}^{l} - C_{ip_i}^{\lambda_i}\right) + \sum_{j\in H} d_{ij} \tag{4.1}$$

$$= \left(C_{ip_i}^{l} - C_{ip_i}^{\lambda_i}\right) + \sum_{j\in H} \left(C_{ji}^{\lambda_j} - C_{jp_j}^{\lambda_j}\right). \tag{4.2}$$

Αν $D_{iH}^{l} < 0$ τότε η αναβάθμιση χαρακτηρίζεται επικερδής.

Η παραπάνω ανάλυση του οφέλους μίας αναβάθμισης προϋποθέτει ότι οι τύποι των υπολοίπων ζεύξεων της λύσης παραμένουν αμετάβλητοι. Για να εξασφαλιστεί αυτό αρκεί η προσθήκη κίνησης κατά μήκος του μονοπατιού P_i να μην οδηγήσει σε παραβίαση των περιορισμών χωρητικότητας κατά μήκος του μονοπατιού P_i. Έστω $m_1, m_2, \ldots, m_K = i$ οι κόμβοι του P_i διατεταγμένοι κατά σειρά από τον απευθείας απόγονο του r ως τον i και dt_{m_k} η αύξηση της κίνησης στους αντίστοιχους κόμβους μετά τις επανασυνδέσεις. Η αύξηση της κίνησης θα πρέπει να μην υπερβαίνει τα παρακάτω όρια W_k:

$$dt_{m_k} \leq W_k = \begin{cases} Z^{l} - t_{m_k} & \text{if } k = K \\ Z^{\lambda_{m_k}} - t_{m_k} & \text{else} \end{cases} \tag{4.3}$$

Όλοι οι αλγόριθμοι αναβαθμίσεων που παρουσιάζονται σε αυτό το κεφάλαιο ακολουθούν κοινή στρατηγική (αλγ 4.1). Οι αλγόριθμοι εκκινούν από μία εφικτή λύση τοπολογίας αστέρα, όπου κάθε τερματικός κόμβος συνδέεται με τον κεντρικό κόμβο με ζεύξη όσο το δυνατόν μικρότερης δυνατής χωρητικότητας. Συνεπώς για την αρχική λύση $T^0(V, E_{T^0})$ ισχύουν τα εξής: $p_i^0 = r$, $t_i^0 = F_i$, $\lambda_i^0 = \min\{l \in \Lambda : Z^l \geq F_i\}$. Κατόπιν, αναζητούνται επικερδείς αναβαθμίσεις και εφόσον βρεθούν τέτοιες υλοποιείται εκείνη με το υψηλότερο όφελός. Η διαδικασία αναζήτησης-υλοποίησης επαναλαμβάνεται έως ότου φτάσουμε σε τοπικό ακρότατο δηλαδή μέχρι να μην είναι πλέον δυνατή η εύρεση επικερδούς αναβάθμισης. Το αποτέλεσμα της διαδοχικής εφαρμογής επικερδών αναβαθμίσεων είναι η μετάβαση σε λύσεις όλο και μικρότερου κόστους.

Αλγόριθμος 4.1: Βασική στρατηγική ευρετικών αλγορίθμων αναβαθμίσεων

```
1  create initial star solution
2  repeat
3  |  foreach valid (i, l) combination do
4  |  |  create set H from a candidate set U
5  |  |  compute D_iH^l
6  |  end
7  |  if at least one profitable D_iH^l found then
8  |  |  implement most profitable upgrade;
9  |  end
10 until no profitable D_iH^l found
```

Οι διαφοροποιήσεις μεταξύ των αλγορίθμων εντοπίζονται κυρίως στη διαδικασία αναζήτησης επικερδών αναβαθμίσεων. Σύμφωνα με τον ορισμό που δώσαμε παραπάνω μία αναβάθμιση χαρακτηρίζεται από τον αναβαθμιζόμενο κόμβο i, τον τύπο ζεύξης l που εγκαθιστούμε και το σύνολο H των κόμβων που επανασυνδέουμε. Σε κάθε κύκλο της κύριας διαδικασίας εξετάζονται ένα σύνολο συνδυασμών (i, l). Για κάθε τέτοιο συνδυασμό υπολογίζεται ένα μοναδικό σύνολο H από ένα σύνολο υποψήφιων κόμβων U και η αντίστοιχη εξοικονόμηση D^l_{iH}.

Συνοψίζοντας, η αλγόριθμοι αναβαθμίσεων διαφοροποιούνται ως προς:

- τα ζεύγη i και l που εξετάζονται σε κάθε κύκλο,
- το σύνολο των υποψήφιων κόμβων U (για κάθε ζεύγος i, l) και
- τον τρόπο επιλογής των κόμβων του U που θα περιληφθούν στο H.

Στόχος της διαδικασίας κατασκευής του συνόλου H είναι η μεγιστοποίηση της εξοικονόμησης D^l_{iH} για τα δεδομένα i, j. Το συγκεκριμένο πρόβλημα 'θυμίζει' το διακριτό πρόβλημα σακιδίου: από ένα σύνολο αντικείμενων (κόμβων), τα οποία φέρουν βάρος (κίνηση) και αξία (όφελος επανασύνδεσης) πρέπει να βρούμε το σύνολο που μεγιστοποιεί τη συνολική αξία με δεδομένο ένα περιορισμό στο συνολικό βάρος (χωρητικότητα αναβαθμιζόμενης ζεύξης). Όμως η εύρεση του βέλτιστου H διαφοροποιείται από το κλασσικό πρόβλημα σακιδίου στο ότι τα αντικείμενα-κόμβοι δεν επιβαρύνουν μόνο ένα περιορισμό-ζεύξη αλλά πολλαπλούς, καθότι πρέπει να διασφαλιστούν οι περιορισμοί χωρητικότητας για όλες τις ζεύξεις στο P_i σύμφωνα με τη σχέση (4.3). Στην ουσία δηλαδή έχουμε έναν περιορισμό για κάθε κόμβο στο P_i. Όμως η εισαγωγή ενός κόμβου έστω j στο σύνολο H δεν επηρεάζει με τον ίδιο τρόπο όλες τις ζεύξεις στο P_i. Για παράδειγμα, οι κοινοί πρόγονοι των κόμβων j και i $(P_j \cup P_i)$ δε θα επιβαρυνθούν από την επανασύνδεση του κόμβου j αφού ήδη η κίνηση του τους διαπερνά.

Για να εξάγουμε την υπολογιστική πολυπλοκότητα των αλγορίθμων αναβάθμισης βρίσκουμε αρχικά άνω όρια για το μέγεθος συγκεκριμένων συνόλων του προβλήματος. Έστω ότι $N = |V^*|$. Δεδομένου ότι κάθε κόμβος παράγει κατ ελάχιστον μοναδιαία κίνηση το μέγεθος του οποιουδήποτε συνόλου προς επανασύνδεση κόμβων H φράσσεται από την τιμή Z^L. Για τον ίδιο λόγο, για δεδομένο κόμβο i, ο αριθμός των κόμβων στο μονοπάτι P_i δε μπορεί να ξεπερνάει την τιμή Z^L. Δεδομένου ότι τα σύνολα H, P_i και V_S είναι υποσύνολα του V^* έχουμε:

$$|H| \leq M, \quad |P_i| \leq M, \quad |V_S| \leq N$$

όπου $M = \min\left(Z^L, N\right)$. Τέλος θεωρούμε ότι $L << M$ και άρα

$$\mathcal{O}(L + M) = \mathcal{O}(M).$$

4.2 Το Διακριτό Πρόβλημα Σακιδίου

Το διακριτό πρόβλημα σακιδίου (discrete knapsack problem) είναι ένα από τα πιο μελετημένα προβλήματα συνδυαστικής βελτιστοποίησης, με πολλές πρακτικές εφαρμογές. Έστω ότι έχουμε στη διάθεσή μας ένα σύνολο αντικειμένων, όπου το κάθε αντικείμενο χαρακτηρίζεται από το βάρος του και την αξία του. Επίσης έχουμε ένα σακίδιο στο οποίο μπορούμε να εισάγουμε αντικείμενα αρκεί το συνολικό βάρος τους να μην υπερβαίνει το προκαθορισμένο όριο βάρους του σακιδίου. Το ζητούμενο είναι να βρούμε ποια αντικείμενα πρέπει να εισάγουμε στο σακίδιο έτσι ώστε να μεγιστοποιήσουμε τη συνολική αξία των αντικειμένων εντός του σακιδίου.

Το διακριτό πρόβλημα σακιδίου μπορεί να διατυπωθεί ως πρόβλημα IP ως εξής: Έστω n αντικείμενα, όπου v_i η αξία και w_i το βάρος του iοστού αντικειμένου. Επίσης έστω W το μέγιστο επιτρεπόμενο βάρος στο σακίδιο:

Πρόβλημα 4.1: Διακριτό πρόβλημα σακιδίου

$$\begin{aligned} &\text{maximize} \sum_{i=1}^{n} v_i x_i \\ &\text{subject to} \sum_{i=1}^{n} w_i x_i \leq W \\ &\qquad x_i \in \{0,1\} \qquad\qquad \forall i \in \{0,1,\ldots,n\} \end{aligned}$$

Το διακριτό πρόβλημα σακιδίου ανήκει στην κατηγορία των NP-hard προβλημάτων (Pisinger (1995).

4.2.1 σχέση με το συνεχές πρόβλημα σακιδίου

Το συνεχές πρόβλημα σακιδίου προκύπτει από το διακριτό αν επιτρέψουμε στους αγνώστους x_i να πάρουν τιμές από το διάστημα $[0,1]$ και η βέλτιστη λύση μπορεί να βρεθεί πολύ εύκολα (Dantzig, 1957) ως εξής: Αρχικά διατάσσουμε τα αντικείμενα σε φθίνουσα σειρά με βάση το λόγο οφέλους-προς-βάρος, v_i/w_i. Για χάρη ευκολίας θεωρούμε στο εξής ότι οι δείκτες των αντικειμένων ανταποκρίνονται στη νέα διάταξη. Κατόπιν, εισάγουμε αντικείμενα διαδοχικά στο σακίδιο, μέχρι να φτάσουμε στο αντικείμενο, έστω με δείκτη b, που δε χωράει εξ ολοκλήρου στο σακίδιο.

$$b = \min\left\{ j : \sum_{i=1}^{j} w_i > W \right\}$$

Εισάγοντας το κλάσμα $f = \left(W - \sum_{i=1}^{b-1} w_i\right) / w_b$ του αντικειμένου b που απαιτείται για να φτάσουμε στο όριο W καταλήγουμε στη βέλτιστη λύση του συνεχούς προβλήματος

$$\overline{x_i} = \begin{cases} 1 & \text{if } i < b \\ f & \text{if } i = b \\ 0 & \text{else} \end{cases}, \tag{4.5}$$

η οποία έχει συνολική αξία

$$\overline{v} = \sum_{i=1}^{b-1} v_i + f v_b. \tag{4.6}$$

Η τιμή $\overline{v}$ αποτελεί άνω φράγμα για τη βέλτιστη λύση του αντίστοιχου διακριτού προβλήματος. Επίσης από την παραπάνω ανάλυση προκύπτει η προφανής εφικτή λύση του διακριτού προβλήματος όπου

$$\underline{x_i} = \begin{cases} 1 & \text{if } i < b \\ 0 & \text{else} \end{cases}, \tag{4.7}$$

με συνολική αξία $\underline{v} = \sum_{i=1}^{b-1} v_i$. Συνεπώς για τη βέλτιστη τιμή v^* του διακριτού προβλήματος ισχύει

$$\sum_{i=1}^{b-1} v_i \leq v^* \leq \sum_{i=1}^{b-1} v_i + f v_b. \tag{4.8}$$

Σύμφωνα με τους (Balas and Zemel, 1980) η βέλτιστη λύση του διακριτού προβλήματος $\mathbf{x}^*$ προκύπτει εφαρμόζοντας συνήθως λίγες αλλαγές στη λύση $\underline{\mathbf{x}}$ γύρω από το αντικείμενο b.

Η διαδικασία υπολογισμού των λύσεων $\underline{x_i}$ και $\overline{x_i}$ κυριαρχείται από την ταξινόμηση των αντικειμένων, η οποία απαιτεί $\mathcal{O}(n \log n)$ βήματα.

4.2.2 ειδική περίπτωση μοναδιαίων βαρών

Στην ειδική περίπτωση όπου όλα τα αντικείμενα έχουν το ίδιο βάρος $w_i = 1$ η εύρεση της βέλτιστης λύσης είναι πολύ απλή. Στην ουσία ο περιορισμός βάρους μετασχηματίζεται σε περιορισμό στο πλήθος των αντικειμένων. Η βέλτιστη λύση προκύπτει επιλέγοντας τα W πιο 'ακριβά' αντικείμενα, δηλαδή $\mathbf{x}^* = \underline{\mathbf{x}}$. Η εύρεση του βέλτιστου συνόλου απαιτεί $\mathcal{O}(n \log n)$ βήματα.

4.2.3 επίλυση με δυναμικό προγραμματισμό

Ο δυναμικός προγραμματισμός είναι μία μέθοδος επίλυσης σύνθετων προβλημάτων μέσω της ανάλυσής τους σε απλούστερα υποπροβλήματα. Βασικό στοιχείο της μεθόδου είναι η επίλυση κάθε υποπροβλήματος μόνο μία φορά και η καταγραφή της λύσης σε πίνακα, εξοικονομώντας έτσι χρόνο από πιθανούς περιττούς επαναϋπολογισμούς. Οι τιμές του πίνακα αξιοποιούνται για την επίλυση όλο και μεγαλύτερης τάξης υποπροβλημάτων μέχρι να επιλύσουμε το αρχικό πρόβλημα (προσέγγιση

από-κάτω-προς-τα-πάνω).

Για την επίλυση ενός προβλήματος σακιδίου κατασκευάζουμε υποπροβλήματα με λιγότερα διαθέσιμα αντικείμενα και με χαμηλότερο όριο χωρητικότητας. Για κάθε $i \in \{0, 1, \ldots, n\}$ και $w \in \{0, 1, \ldots, W\}$ ορίζουμε ως P_{iw} το υποπρόβλημα με όριο χωρητικότητας w και διαθέσιμα αντικείμενα τα $\{1, \ldots, i\}$. Στον πίνακα A διαστάσεων $(n+1)\times(W+1)$ αποθηκεύουμε τις βέλτιστες τιμές των υποπροβλημάτων. Η τιμή $A[n, W]$ θα είναι και η βέλτιστη για το αρχικό πρόβλημα.

Η βέλτιστη αξία των υποπροβλημάτων με $i = 0$ ή $w = 0$ είναι προφανώς 0. Για την επίλυση υποπροβλημάτων P_{iw} με $i, w > 0$ εξετάζουμε εάν είναι προτιμότερο, ή όχι, να εισάγουμε το αντικείμενο i στο σακίδιο βασιζόμενοι στα τιμές που έχουμε υπολογίσει για υποπροβλήματα $i - 1$ αντικειμένων: Για να είναι δυνατή η εισαγωγή του αντικειμένου i θα πρέπει $w_i \leq w$. Επίσης, τα υπόλοιπα αντικείμενα του σακιδίου θα πρέπει να μην υπερβαίνουν το όριο βάρους $w - w_i$ και άρα η μέγιστη αξία που μπορεί να επιτευχθεί αν όντως συμπεριλάβουμε το i είναι $v_i + A[i-1, w-w_i]$. Αν αντίθετα το αντικείμενο i δεν εισαχθεί στο σακίδιο τότε η βέλτιστη αξία στο σακίδιο είναι $A[i-1, w]$. Συνεπώς κάθε στοιχείο $A[i, w]$ μπορεί να υπολογιστεί με βάση την παρακάτω αναδρομική σχέση

$$A[i,w] = \begin{cases} 0 & \text{if } i = 0 \text{ or } w = 0 \\ A[i-1,w] & \text{else if } w \geq w_i \\ \max\left(A[i-1,w], v_i + A[i-1, w-w_i]\right) & \text{else} \end{cases} . \tag{4.9}$$

Ο πίνακας A μας δίνει τη βέλτιστη αξία που μπορεί να έχει το σακίδιο, όχι όμως και από ποια αντικείμενα θα περιλαμβάνει αυτό. Για το λόγο αυτό παράλληλα με τον A συμπληρώνουμε και έναν επιπλέον πίνακα B ως εξής: Εάν κατά τον υπολογισμό του στοιχείου $A[i, w]$ αποφασίσουμε να συμπεριλάβουμε το αντικείμενο i στη λύση τότε θέτουμε $K[i, w] = 1$ ενώ σε αντίθετη περίπτωση θέτουμε $K[i, w] = 0$. Αφού έχουμε γεμίσει και τους δύο αυτούς πίνακες, βρίσκουμε το βέλτιστο σύνολο αντικειμένων διασχίζοντας τον πίνακα B από-πάνω-προς-τα-κάτω (αλγ 4.2).

Ο υπολογισμός του πίνακα A απαιτεί $\mathcal{O}(nW)$ βήματα, ενώ η αντίστροφη διάσχιση του πίνακα B μόλις $\mathcal{O}(n)$, συνεπώς η χρονική πολυπλοκότητα του αλγορίθμου 4.2 είναι $\mathcal{O}(nW)$. Ο αλγόριθμος είναι ψευδο-πολυωνυμικός, και όχι πολυωνυμικός, καθότι δε φράσσεται αποκλειστικά από το μέγεθος της εισόδου (πλήθος αντικειμένων n) αλλά και από την αριθμητική τιμή της εισόδου (όριο βάρους W).

Αλγόριθμος 4.2: Επίλυση διακριτού προβλήματος σακιδίου με δυναμικό προγραμματισμό

```
for w ← 0 to W do
    A[0, w] ← 0
end
for i ← 1 to n do
    for w ← 0 to W do
        if w_i ≤ w and v_i + A[i − 1, w − w_i] > A[i − 1, w] then
            A[i, w] ← v_i + A[i − 1, w − w_i]
            B[i, w] ← 1
        else
            A[i, w] ← A[i − 1, w]
            B[i, w] ← 0
        end
    end
end
S ← ∅
w ← W
for i ← n downto 1 do
    if B[i, w] = 1 then
        S ← S ∪ {i}
        w ← w − w_i
    end
end
return S
```

4.3 Ο αλγόριθμος GUH για μοναδιαία στιγμιότυπα

Οι Gamvros et al. (2002, 2006) παρουσίασαν τον πρώτο ευρετικό αλγόριθμο για το πρόβλημα ML-CMST, ο οποίος όμως περιορίζεται σε στιγμιότυπα όπου η παραγόμενη κίνηση από κάθε κόμβο είναι μοναδιαία. Ο αλγόριθμος GUH (αλγ. 4.3) εκκινεί εξετάζοντας αναβαθμίσεις προς τον τύπο μέγιστης χωρητικότητας ($l \leftarrow L$). Σε κάθε κύκλο της φάσης αναζήτησης όλοι οι τερματικοί κόμβοι που δεν έχουν ήδη αναβαθμιστεί (V_S) θεωρούνται ως υποψήφιοι προς αναβάθμιση. Για κάθε υποψήφιο κόμβο κατασκευάζεται ένα σύνολο (H) κόμβων προς επανασύνδεση και εξετάζεται η αντίστοιχη αναβάθμιση D^l_{iH}. Αν βρεθούν επικερδείς αναβαθμίσεις τότε υλοποιείται η πιο επικερδής από αυτές. Όταν δεν είναι δυνατόν να βρεθούν επικερδείς αναβαθμίσεις προς τον τύπο l ο αλγόριθμος δεν υλοποιεί κάποια αναβάθμιση αλλά από τον επόμενο κύκλο εξετάζει αναβαθμίσεις προς τον αμέσως μικρότερο τύπο ($l \leftarrow l - 1$). Όταν πλέον έχουν εξαντληθεί οι αναβαθμίσεις και για τον τύπο με τη μικρότερη χωρητικότητα ο αλγόριθμος τερματίζει.

Οι Gamvros et al. (2002, 2006) προτείνουν τον ακόλουθο μηχανισμό για την κατασκευή των συνόλων H: Κατασκευάζουμε αρχικά το σύνολο U αρχικά ως υποψήφιους κόμβους τους κόμβους εκείνους που δεν έχουν προηγουμένως αναβαθμιστεί και των οποίων η επανασύνδεση θα ήταν επικερδής ($d_{ij} < 0$). Οι υποψήφιοι κόμβοι εισάγονται σε ταξινομημένη λίστα q_i κατά φθίνουσα σειρά με βάση το όφελος $|d_{ij}|$. Ακολούθως, εξετάζουμε κάθε κόμβο της q_i ξεχωριστά για να τον εισάγουμε στο H

Αλγόριθμος 4.3: Κύρια διαδικασία αλγορίθμου GUH

```
1  Create initial star solution
2  l ← L
3  V_S ← V*
4  repeat
5  |  foreach i ∈ V_S do
6  |  |  if l ≥ λ_i then
7  |  |  |  Create set H and compute D^l_iH
8  |  |  end
9  |  end
10 |  if min{D^l_iH} < 0 then
11 |  |  implement upgrade and reconnections for min{D^l_iH}
12 |  |  V_S ← V_S \ {i}
13 |  else
14 |  |  l ← l − 1
15 |  end
16 until l = 0
```

Αλγόριθμος 4.4: Διαδικασία κατασκευής συνόλου H για τον αλγόριθμο GUH

```
1  H ← ∅
2  foreach m ∈ P_i do  t'_m ← t_m
3  for o = 1 to  |q_i| do
4  |  j ← q_i[o]
5  |  add ← true
6  |  foreach m ∈ P_i \ P_j do
7  |  |  if t'_m + t_j > Z^{λ_m} then
8  |  |  |  add ← false
9  |  |  |  break
10 |  |  end
11 |  end
12 |  if add = true then
13 |  |  H ← H ∪ {j}
14 |  |  foreach m ∈ P_i \ P_j do
15 |  |  |  t'_m ← t'_m + t_j
16 |  |  end
17 |  end
18 end
```

(αλγ. 4.4). Για ένας εισαχθεί ένας κόμβος $j \in U$ στο H θα πρέπει να διασφαλιστούν οι περιορισμοί χωρητικότητας (4.3) των κόμβων που θα επιβαρυνθούν με την κίνηση του j δηλαδή οι κόμβοι του συνόλου $P_i \setminus P_j$.

Η χρονική πολυπλοκότητα ενός κύκλου του κύριου βρόχου κυριαρχείται από τη διαδικασία ανίχνευσης επικερδών αναβαθμίσεων και ειδικότερα από το μηχανισμό υπολογισμού των συνόλων H. Ο υπολογισμός αυτός απαιτεί καταρχήν $\mathcal{O}(N \log N)$ βήματα για την κατασκευή της ταξινομημένης λίστας q_i. Επίσης δεδομένου ότι για κάθε κόμβο θα πρέπει να ελεγχθούν το πολύ $|P_i| \leq M$ το πλήθος περιορισμοί και στη χειρότερη περίπτωση θα εξεταστούν όλοι οι υποψήφιοι κόμβοι (N), απαιτούνται επιπλέον $\mathcal{O}(NM)$ βήματα. Συνεπώς, η πολυπλοκότητα κατασκευής ενός συνόλου H είναι $\mathcal{O}\left(N(M + \log N)\right)$ και άρα ενα κύκλος του κύριου βρόχου απαιτεί

$$\mathcal{O}\left(N^2(M + \log N)\right)$$

βήματα. Δεδομένου ότι ο κύριος βρόχος θα εκτελεστεί το πολύ $L + N$ φορές, η χρονική πολυπλοκότητα του GUH είναι

$$\mathcal{O}\left(N^3(M + \log N)\right).$$

4.4 Επιλέγει ο GUH το βέλτιστο σύνολο επανασυνδεόμενων κόμβων;

Οι Gamvros et al. (2002, 2006) σημειώνουν, χωρίς να αποδεικνύουν, ότι ο αλγόριθμος 4.4 εξασφαλίζει ότι θα επιλεγεί το σύνολο που μεγιστοποιεί το κέρδος από τις επανασυνδέσεις στα πλαίσια μίας αναβάθμισης. Στην ενότητα αυτή θα ορίσουμε τυπικά το πρόβλημα και θα δείξουμε ότι ο παραπάνω ισχυρισμός είναι αληθής.

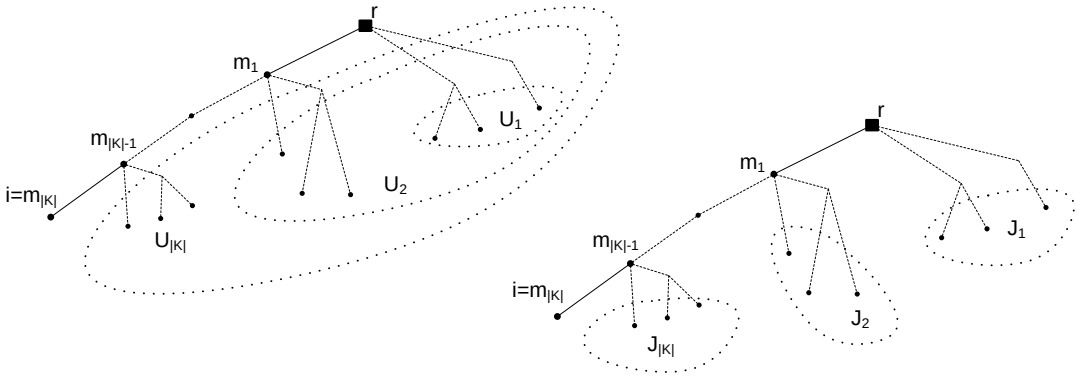
Έστω U το σύνολο των υποψήφιων για επανασύνδεση κόμβων, ορίζουμε τα υποσύνολα U_k ως εξής:

$$U_k = \{j \in U : m_k \notin P_j\}. \tag{4.10}$$

Κάθε σύνολο U_k περιλαμβάνει τους κόμβους του U που *δεν* είναι απόγονοι του m_k, δηλαδή του κόμβους εκείνους που θα επιβαρύνουν την αντίστοιχη ζεύξη αν εισαχθούν στο σύνολο H.

Πρόταση 4.1: $U_1 \subseteq U_2 \subseteq \ldots \subseteq U_{K-1} \subseteq U_K$.

Απόδειξη: Έστω $k < K$ και $j \in U_k$. Άρα $m_k \notin P_j$. Αν υποθέσουμε ότι $m_{k+1} \in P_j$ τότε επειδή $m_k = p_{m_{k+1}}$ έχουμε ότι $m_{k+1} \in P_j$ το οποίο είναι άτοπο. Άρα $m_{k+1} \notin P_j \Rightarrow j \in U_{k+1} \Rightarrow U_k \subseteq$



Σχήμα 4.1: Διαμέριση του συνόλου των υποψήφιων προς επανασύνδεση κόμβων

U_{k+1}. □

Πρόταση 4.2: $U_K = U$.

Απόδειξη: Ο κόμβος $m_K = i$ δεν έχει αναβαθμιστεί προηγουμένως και άρα δεν έχει και απογόνους. Συνεπώς, $m_K \notin P_j$ για κάθε $j \in U$. □

Επίσης, ορίζουμε συνάρτηση $\kappa : U \to K$ ως εξής:

$$\kappa(j) = \min \{k : j \in U_k\}, \tag{4.11}$$

και με βάση αυτή κατασκευάζουμε διαμέριση [1] του συνόλου U ως εξής:

$$J_k = \{j \in U : \kappa(j) = k\}. \tag{4.12}$$

Άμεσες συνέπειες των παραπάνω ορισμών και της πρότασης 4.1 είναι οι ακόλουθες προτάσεις:

Πρόταση 4.3: $J_k = U_k \setminus U_{k-1}$ (θεωρώντας ότι $U_0 = \emptyset$).

Πρόταση 4.4: Για οποιοδήποτε $j \in U$: αν $k < \kappa(j)$ τότε $j \notin U_k$, ενώ αν $k \geq \kappa(j)$ τότε $j \in U_k$.

Το εξεταζόμενο πρόβλημα αποτελεί παραλλαγή του προβλήματος σακιδίου. Αντί όμως ενός σακιδίου πλέον έχουμε K σακίδια. Για κάθε σακίδιο k υπάρχει το αντίστοιχο σύνολο διαθέσιμων αντικειμένων-κόμβων J_k. Τα αντικείμενα έχουν μία ιδιότυπη σχέση με τα σακίδια: η εισαγωγή ενός αντικειμένου του J_k επιβαρύνει τα σακίδια $k' \geq k$. Εναλλακτικά μπορούμε να το οπτικοποιήσουμε ως εξής: Κάθε σακίδιο k βρίσκεται εντός του σακιδίου $k' = k+1$ (με εξαίρεση το σακίδιο K). Υπό αυτή την έννοια

[1] Τα J_k αποτελούν διαμέριση του U δηλαδή είναι ξένα ανά δύο μεταξύ τους και έχουν ένωση το U

η εισαγωγή ενός αντικειμένου σε κάποιο σακίδιο αναπόφευκτα επιβαρύνει και τα εξωτερικά σε αυτό σακίδια.

Δεδομένου ότι οι υποψήφιοι κόμβοι δεν έχουν προηγουμένως αναβαθμιστεί, η κίνηση που τους διασχίζει είναι ίση με την κίνηση που παράγουν οι ίδιοι δηλαδή $t_j = F_j$. Επίσης εφόσον εξετάζουμε μοναδιαία στιγμιότυπα έχουμε ότι $t_j = 1$ και άρα το όριο W_k ενός σακιδίου-ζεύξης εκφράζει το πλήθος των κόμβων που μπορούν να επιλεγούν από το σύνολο U_k.

Πρόβλημα 4.2: Εύρεση βέλτιστου συνόλου επανασυνδεόμενων κόμβων στα πλαίσια του αλγόριθμου GUH

$$\underset{H}{\text{maximize}} \quad \sum_{j \in H} |d_{ij}| \tag{4.13α′}$$

$$\text{subject to} \quad |U_k \cap H| \leq W_k \qquad \forall k \in \{1, \ldots, K\} \tag{4.13β′}$$

$$U_1 \subseteq U_2 \subseteq \ldots \subseteq U_K \tag{4.13γ′}$$

$$H \subseteq U_K \tag{4.13δ′}$$

Πρόταση 4.5: Ο αλγόριθμος 4.4 δίνει τη βέλτιστη λύση για το πρόβλημα 4.2 .
Απόδειξη: Ορίζουμε $f(H) = \sum_{j \in H} |d_{ij}|$ και $r_k(H) = W_k - |U_k \cap H|$ για οποιαδήποτε $H \subseteq U_K, k \in \{1, \ldots, K\}$. Έστω $\hat{H}$ το σύνολο που επιστρέφει ο αλγόριθμος 4.4. Αρκεί να δείξουμε ότι δεν υπάρχει σύνολο H' με $f(H') > f(\hat{H})$ και $r_k(H') \geq 0$ για κάθε $k \in \{1, \ldots, K\}$.

Έστω $A = H' \setminus \hat{H}$ και $B = \hat{H} \setminus H'$. Το σύνολο A δε μπορεί να είναι κενό διότι σε αντίθετη περίπτωση θα ίσχυε $f(H') \leq f(\hat{H})$. Έστω $a_1, \ldots, a_n$ οι κόμβοι του A με $|d_{ia_1}| \geq \ldots \geq |d_{ia_n}|$. Κάθε κόμβος a_x αποκλείστηκε από το σύνολο $\hat{H}$ διότι η εισαγωγή του σε αυτό θα οδηγούσε σε παραβίαση τουλάχιστον ενός από του περιορισμούς (4.13β′) με $k \geq \kappa(a_x)$. Θεωρούμε k_x το υψηλότερο από τα k για το οποία παραβιάζεται περιορισμός κατά την εξέταση του a_x από τον αλγόριθμο 4.4.

Ο τρόπος που διατάξαμε τους κόμβους του A αντικατοπτρίζει και τη σειρά με την οποία εξετάστηκαν και αποκλείστηκαν από το σύνολο $\hat{H}$. Έστω $a_x, a_y \in A$ με $x < y$. Αν $\kappa(a_y) > k_x$ τότε προφανώς $k_y > k_x$ Αν αντίθετα $\kappa(a_y) \leq k_x$ τότε $a_y \in U_{k_x}$ και εφόσον ο a_y εξετάστηκε μετά τον a_x θα παραβίαζε και αυτός τον περιορισμό k_x. Συνεπώς σε κάθε περίπτωση $k_y \geq k_x$

Εφόσον $\hat{H}$ εφικτή λύση τότε $r_k(\hat{H}) \geq 0$ για κάθε $k \in \{1, \ldots, K\}$. Επίσης από τον ορισμό των k_x προκύπτει ότι $r_{k_x}(\hat{H}) = 0$. Η εισαγωγή του a_1 στο σύνολο $\hat{H}$ συνεπάγεται παραβίαση του περιορισμού για $k = k_1$ (ενδεχομένως και για άλλα $k < k_1$). Για να αποκατασταθεί η εφικτότητα της λύσης θα πρέπει να αφαιρεθεί κόμβος $b_1 \in U_{k_1} \cap \hat{H}$. Όμως, δεν είναι δυνατόν ο b_1 να εξετάστηκε από τον αλγόριθμο 4.4 μετά τον a_1 διότι τότε θα είχε σίγουρα απορριφθεί και αυτός από το $\hat{H}$. Συνεπώς

$|d_{ib_1}| \geq |d_{ia_1}|$. Μετά την προσθήκη του a_1 και την αφαίρεση του b_1 από το $\hat{H}$ έχουμε ότι

$$r_{k_1}\left(\hat{H} \cup \{a_1\} \setminus \{b_1\}\right) = r_{k_1}(\hat{H}) = 0$$
$$r_k\left(\hat{H} \cup \{a_1\} \setminus \{b_1\}\right) = r_k(\hat{H}) \qquad \forall k > k_1$$

Εφόσον $r_{k_2}(\hat{H}) = 0$ και $k_2 \geq k_1$ από την παραπάνω πρόταση προκύπτει ότι $r_{k_2}(\hat{H} \cup \{a_1\} \setminus \{b_1\}) = 0$ και άρα προσθέτοντας και τον κόμβο a_2 στη λύση έχουμε παραβίαση του περιορισμού για $k = k_2$. Όπως και προηγουμένως θα πρέπει να αφαιρέσουμε κόμβο $b_2 \in U_{k_2} \cap \hat{H}$, για τον οποίο θα ισχύει $|d_{ib_2}| \geq |d_{ia_2}|$. Μετά την προσθήκη του a_2 και την αφαίρεση του b_2 από τη λύση θα έχουμε

$$r_{k_2}\left(\hat{H} \cup \{a_1, a_2\} \setminus \{b_1, b_2\}\right) = r_{k_2}(\hat{H}) = 0$$
$$r_k\left(\hat{H} \cup \{a_1, a_2\} \setminus \{b_1, b_2\}\right) = r_k(\hat{H}) \qquad \forall k > k_2$$

Επεκτείνοντας τον παραπάνω συλλογισμό φτάνουμε στο συμπέρασμα ότι για κάθε κόμβο $a_x \in A$ θα πρέπει να υπάρχει και αντίστοιχος κόμβος $b_x \in B$ με $|d_{ib_x}| \geq |d_{ia_x}|$ προκείμενου η H' να είναι εφικτή. Όμως $f(H') = f(\hat{H}) + f(A) - f(B) \leq f(\hat{H})$. Άτοπο. □

4.5 Ο αλγόριθμος UH1

Στην ενότητα αυτή θα παρουσιάσουμε τροποποιήσεις στον τρόπο προσδιορισμού του συνόλου των επανασυνδεόμενων κόμβων οι οποίες αφενός μεν βελτιώνουν τους χρόνους εκτέλεσης και την υπολογιστική πολυπλοκότητα του αλγορίθμου GUH, αφετέρου δε επιτρέπουν την αντιμετώπιση και μη μοναδιαίων στιγμιότυπων. Στο εξής θα αναφερόμαστε στον τροποποιημένο αλγόριθμο ως UH1 ώστε να μη γίνεται σύγχυση με τον αρχικό αλγόριθμο GUH των Gamvros et al. (2002, 2006).

Η βασική στρατηγική του αλγορίθμου GUH είναι προφανές ότι μπορεί να εφαρμοστεί και σε μη μοναδιαία στιγμιότυπα. Ως ευρετικός μπορεί να χρησιμοποιηθεί ο αλγόριθμος 4.5 με την ταξινόμηση όμως των κόμβων να γίνεται με βάση το λόγο οφέλους-προς-επιβάρυνση $|d_{ij}|/t_j$, και όχι απλά το όφελος $|d_{ij}|$. Όπως και με το κλασσικό πρόβλημα σακιδίου, αναμένεται η εξαγόμενη λύση να έχει μικρές διαφοροποιήσεις από τη βέλτιστη.

Κατά την εξέταση ενός κόμβου j από τον αλγ ...ελέγχεται το υπόλοιπο χωρητικότητας για τους κόμβους m_k με $k \geq \kappa(j)$. Η πραγματοποίηση των ελέγχων προϋποθέτει την ενημέρωση των αντίστοιχων υπολοίπων μετά από κάθε προσθήκη κόμβου στο σύνολο H. Η εξέταση μπορεί να ολοκληρωθεί κάνοντας ένα μόνο έλεγχο αντί για $K + 1 - \kappa(j)$ ελέγχους ως εξής (αλγ. 4.5): Δεδομένου ότι ο εξεταζόμενος κόμβος θα επιβαρύνει ισόποσα τους κόμβους m_k με $k \geq \kappa(j)$ αρκεί να ελέγξουμε αν η

Αλγόριθμος 4.5: Διαδικασία κατασκευής συνόλου H για τον αλγόριθμο UH1

```
1  H ← ∅
2  r_K ← W_K
3  for k ← K − 1 downto 1 do
4  |  r_k ← min{r_{k+1}, W_k}
5  end
6  for o ← 1 to |q_i| do
7  |  j ← q_i[o]
8  |  if r_{κ(j)} ≥ t_j then
9  |  |  H ← H ∪ {j}
10 |  |  for k ← K downto 1 do
11 |  |  |  if k ≥ κ(j) then
12 |  |  |  |  r_k ← r_k − t_j
13 |  |  |  else
14 |  |  |  |  r_k ← min{r_k, r_{κ(j)}}
15 |  |  |  end
16 |  |  end
17 |  |  if r_K = 0 then
18 |  |  |  return H
19 |  |  end
20 |  end
21 end
22 return H
```

τιμή t_j υπερβαίνει ή όχι το ελάχιστο των υπολοίπων χωρητικότητας (r_k) των εμπλεκόμενων κόμβων. Η αλλαγή αυτή επιβάλει την ενημέρωση των μεταβλητών r_k μετά από κάθε προσθήκη κόμβου.

Η κατασκευή ενός συνόλου H από τον GUH, προϋποθέτει την κατασκευή της αντίστοιχης ταξινομημένης λίστας q_i. Ωστόσο, δεν είναι απαραίτητο να κατασκευάζεται η αντίστοιχη λίστα q_i κάθε φορά εκ του μηδενός. Αντ' αυτού, προτείνουμε την κατασκευή λιστών q_i για όλους τους τερματικούς κόμβους $i \in V^*$ κατά την εκκίνηση του αλγορίθμου και την ενημέρωση τους μετά από κάθε υλοποίηση αναβάθμισης.

Οι προτεινόμενες τροποποιήσεις ελαφρύνουν τον υπολογισμό ενός συνόλου H, ο οποίος απαιτούσε αρχικά $\mathcal{O}(N \log N + NM)$ βήματα. Το πρώτο σκέλος το αθροίσματος αφορούσε την κατασκευή της ταξινομημένης λίστας q_i και πλέον απαλείφεται. Το δεύτερο σκέλος αφορούσε την εξέταση των υποψήφιων κόμβων. Με τον αλγόριθμο 4.5 θα χρειαστεί ένας έλεγχος για κάθε υποψήφιο κόμβο στην χειρότερη περίπτωση ($\mathcal{O}(N)$), ενώ για κάθε κόμβο που εισάγεται στο θα πρέπει να ενημερωθούν οι τιμές r_k $\left(\mathcal{O}(M^2)\right)$. Συνεπώς ο υπολογισμός ενός συνόλου H μετά τις τροποποιήσεις θα απαιτεί $\mathcal{O}(N + M^2)$ βήματα.

Η εύρεση της βέλτιστης αναβάθμισης για ένα κύκλο του κύριου βρόχου θα κοστίσει το πολύ $\mathcal{O}((N + M^2))$, όμως θα πρέπει μετά την υλοποίηση της να ενημερώσουμε τις τιμές των ταξινομημένων λιστών q_i. Δεδομένου ότι μία λίστα q_i περιλαμβάνει το πολύ V^* κόμβους, οι διαδικασίες αφαίρεσης και προσθήκης ενός στοιχείου κοστίζουν αμφότερες $\mathcal{O}(\log N)$ αν η λίστα υποστηρίζεται

από ένα ισορροπημένο δυαδικό δέντρο αναζήτησης (balanced BST). Άρα, η ενημέρωση (είτε προσθήκη, είτε αφαίρεση, είτε και τα δύο) των τιμών των $\mathcal{O}(M)$ επανασυνδεόμενων κόμβων στις $\mathcal{O}(N)$ λίστες κοστίζει συνολικά $\mathcal{O}(NM \log N)$.

Συνεπώς, ένας κύκλος του κύριου βρόχου απαιτεί

$$\mathcal{O}\left(N(M \log N + N + M^2)\right)$$

βήματα και άρα η χρονική πολυπλοκότητα του UH μετά τις βελτιώσεις είναι ίση με

$$\mathcal{O}\left(N^2(M \log N + N + M^2)\right).$$

4.6 Ο αλγόριθμος UH2

Το κίνητρο για την περαιτέρω τροποποίηση του GUH προέκυψε από τη μελέτη των εξαγόμενων από αυτόν λύσεων. Σε περιπτώσεις όπου το κόστος μίας εξαγόμενης λύσης απείχε πολύ από αυτό της βέλτιστης, αυτό συνήθως οφειλόταν στην ύπαρξη σε αυτήν κεντρικών ζεύξεων μεγαλύτερης χωρητικότητας από τις αντίστοιχες της βέλτιστης. Η αδυναμία του GUH, εντοπίστηκε στο γεγονός ότι εξετάζουν αναβαθμίσεις προς ένα τύπο κάθε φορά ξεκινώντας από τον $l = L$ και εν συνεχεία μειώνει κάθε φορά το δείκτη l κατά μία μονάδα. Όταν για παράδειγμα υλοποιείται μία αναβάθμιση προς κάποιον τύπο l_1, ενδέχεται η αναβάθμιση προς κάποιον άλλο τύπο $l_2 < l_1$ για τον ίδιο κόμβο να είναι περισσότερο επικερδής. Σε αυτή την περίπτωση η 'προτιμότερη' αναβάθμιση προς τον l_2 αγνοείται.

Με τον UH2 (αλγόριθμος 4.6) επιχειρούμε να αποφύγουμε το παραπάνω φαινόμενο επιτρέποντας στον αλγόριθμο να εξετάζει αναβαθμίσεις προς όλους τους τύπους ζεύξεων σε ένα κύκλο του κύριου βρόγχου. Η αλλαγή αυτή αυξάνει το μέγιστο αριθμό εξεταζόμενων αναβαθμίσεων σε κάθε επανάληψη σε LN (από N στον UH). Συνεπώς μία επανάληψη του κύριου βρόγχου απαιτεί το πολύ

$$\mathcal{O}(LN(M \log N + N + M^2)),$$

βήματα, και συνεπώς η χρονική πολυπλοκότητα του UH2 είναι

$$\mathcal{O}(LN^2(M \log N + N + M^2)).$$

4.7 Ο αλγόριθμος UH3

Με τον αλγόριθμο UH3 (αλγ. 4.7) επεκτείνουμε περαιτέρω την ιδέα των αναβαθμίσεων, επιτρέποντας σε κόμβους που έχουν ήδη αναβαθμιστεί να επανα-αναβαθμιστούν. Ήτοι, σε κάθε επανάληψη του κυρίου βρόχου εξετάζουμε το ενδεχόμενο αναβάθμισης για οποιονδήποτε τερματικό κόμβο ($i \in V^*$). Επίσης, το σύνολο U, των υποψήφιων προς επανασύνδεση κόμβων, δε περιορίζεται πλέον σε κόμβους που δεν έχουν προηγουμένως αναβαθμιστεί (κόμβοι-φύλλα, V_S). Αντ' αυτού, επιτρέπεται η επανασύνδεση οποιουδήποτε τερματικού κόμβου προς ένα αναβαθμιζόμενο κόμβο i, αρκεί να μη δημιουργείται κύκλος.

Μπορούμε να χωρίσουμε τους τερματικούς κόμβους του προβλήματος στα εξής τρία ξένα υποσύνολα: α) τους πρόγονους του i, β) τους απόγονους του i και γ) όλους τους υπόλοιπους τερματικούς κόμβους. Οι πρόγονοι του i δε μπορούν να επανασυνδεθούν προς τον i καθώς κάτι τέτοιο θα δημιουργούσε κύκλο. Όσον αφορά τους απόγονους του i, η κίνηση που παράγουν οι κόμβοι αυτοί ήδη διαπερνά τον i και άρα μπορούν να επανασυνδεθούν άμεσα χωρίς να επιβαρύνουν κάποια ζεύξη στο P_i. Συνεπώς το πρόβλημα επικεντρώνεται στη βέλτιστη επιλογή μεταξύ των τερματικών κόμβων που δεν είναι ούτε πρόγονοι, ούτε απόγονοι του i.

Επίσης, παρατηρούμε ότι η επιβάρυνση για ένα κόμβο m_k του P_i δεν ισούται απαραίτητα με το άθροισμα της κίνησης των κόμβων που επιλέχθηκαν από το U_k. Για παράδειγμα έστω j_1 και j_2 δύο τερματικοί κόμβοι με τον j_2 να είναι απόγονος του j_1. Η κίνηση t_{j_2} διέρχεται από τον j_1 και άρα αποτελεί μέρος της t_{j_1}. Συνεπώς εάν επανασυνδέσουμε τους δύο αυτούς κόμβους προς τον i η επιβάρυνση στον i θα είναι ίση με j_1. Συμπερασματικά, ένας επανασυνδεόμενος κόμβος δεν επιβαρύνει εάν ταυτόχρονα επανασυνδέουμε και κάποιον πρόγονό του. Η αλληλεξάρτηση μεταξύ των υποψήφιων κόμβων επιβάλλει την τροποποίηση της διαδικασίας επιλογής των κόμβων που θα επανασυνδεθούν.

Ο ευρετικός αλγόριθμος που προτείνουμε (αλγ. 4.8) ακολουθεί τη βασική ιδέα του αλγορίθμου 4.5, δηλαδή επιλέγουμε όσο το δυνατόν περισσότερους κόμβους, δίνοντας προτεραιότητα σε εκείνους με υψηλότερο λόγο οφέλους-προς-επιβάρυνση $|d_{ij}|/t_j$, όμως διαφοροποιείται στο ότι μετά από κάθε εισαγωγή κόμβου επαναξιολογούμε την κίνηση στους εναπομείναντες υποψήφιους κόμβους. Επειδή η επαναξιολόγηση αυτή έχει ισχή μόνο για την εξεταζόμενη αναβάθμιση, η οποία θα υλοποιηθεί μόνο εάν είναι και η βέλτιστη, καταγράφουμε τις αλλαγές σε μία βοηθητική λίστα q'_i (γραμμή 2). Η επανασύνδεση ενός κόμβου j με τον i επιφέρει μείωση ίση με t_j στην κίνηση όλων των κόμβων στο μονοπάτι P_j και άρα οι τιμές τους στην ταξινομημένη λίστα q'_i πρέπει να ενημερωθούν. (γραμμές 8-10). Επίσης, όλοι οι απόγονοι του j καθίστανται (έμμεσοι) απόγονοι του i. Όσοι από αυτούς $(V_j \setminus \{j\})$ περιλαμβάνονται στη λίστα q'_i (δηλαδή πιθανή επανασύνδεση τους με τον i είναι επικερδής) μπορούν να εισαχθούν στο σύνολο H απευθείας δεδομένου ότι δε θα μεταβληθεί η κίνηση κατά

Αλγόριθμος 4.6: Κύρια διαδικασία αλγορίθμου UH2

```
Create initial star solution
V_S ← V*
repeat
    foreach i ∈ V_S do
        for l = λ_i to L do
            Create set H and compute D^l_iH
        end
    end
    if min{D^l_iH} < 0 then
        implement upgrade and reconnections for min{D^l_iH}
        V_S ← V_S \ {i}
    end
until min{D^l_iH} ≥ 0
```

Αλγόριθμος 4.7: Κύρια διαδικασία αλγορίθμου UH3

```
Create initial star solution
repeat
    foreach i ∈ V* do
        for l = λ_i to L do
            Create set H and compute D^l_iH
        end
    end
    if min{D^l_iH} < 0 then
        implement upgrade and reconnections for min{D^l_iH}
    end
until min{D^l_iH} ≥ 0
```

Αλγόριθμος 4.8: Διαδικασία κατασκευής συνόλου H για τον αλγόριθμο UH3

```
H ← ∅
q'_i ← copy of q_i excluding nodes in P_i
while q'_i not empty do
    j ← pop first element of q'_i
    if adding j in H does not violate any constraint in P_i then
        add j in H
        update traffic on all m in P_i
        foreach m ∈ P_j \ {j} do
            update m's entry in q'_i
        end
        foreach k ∈ V_j \ {j} do
            if k ∈ q'_i then
                add k in H and remove it from q'_i
            end
        end
    end
end
```

μήκος του P_i (γραμμές 11-15).

Οι προαναφερθείσες ενημερώσεις της λίστας q'_i απαιτούν τουλάχιστον $M \log N$ βήματα για κάθε εισαγωγή κόμβου στο σύνολο H. Συνεπώς, το υπολογιστικό κόστος για όλες τις ενημερώσεις της q'_i κατά τον υπολογισμό ενός συνόλου H είναι το πολύ $\mathcal{O}\left(M^2 \log N\right)$. Αν σε αυτό προσθέσουμε και το κόστος ελέγχου των περιορισμών χωρητικότητας $\mathcal{O}(MN)$, η προκύπτουσα πολυπλοκότητα για τον υπολογισμό ενός συνόλου H είναι $\mathcal{O}\left(M\left(N + M \log N\right)\right)$. Αυτό συνεπάγεται ότι τα βήματα που απαιτούνται για μία επανάληψη του κύριου βρόχου είναι το πολύ

$$\mathcal{O}\left(NLM\left(N + M \log N\right)\right).$$

4.8 Βέλτιστη επιλογή επανασυνδεόμενων κόμβων: οι αλγόριθμοι UH1-DP, UH2-DP

Οι αλγόριθμοι UH1, UH2 και UH3 που παρουσιάσαμε στις προηγούμενες ενότητες απευθύνονται σε στιγμιότυπα του προβλήματος MLCMST ανεξάρτητα από τις τιμές της παραγόμενης κίνησης σε κάθε τερματικό κόμβο. Και οι τρεις αυτοί αλγόριθμοι επιλέγουν τους κόμβους που επανασυνδέονται στα πλαίσια μίας αναβάθμισης με βάση το λόγο οφέλους-προς-επιβάρυνση (αλγ 4.4 για τους UH1, UH2 και 4.8 για τον UH3). Η στρατηγική αυτή δίνει τα βέλτιστα σύνολα κόμβων μόνο στη περίπτωση που εφαρμόζουμε τον αλγόριθμο UH1 ή UH2 σε μοναδιαία στιγμιότυπα. Στην ενότητα αυτή παρουσιάζουμε αλγόριθμο δυναμικού προγραμματισμού που να δίνει βέλτιστα σύνολα επανασυνδεόμενων κόμβων για αναβαθμίσεις των UH1, UH2. Αντίστοιχα στην επόμενη ενότητα παρουσιάζουμε αλγόριθμο δυναμικού προγραμματισμού που να δίνει βέλτιστα σύνολα επανασυνδεόμενων κόμβων για αναβαθμίσεις του UH3. Για το σχεδιασμό των παρουσιαζόμενων αλγορίθμων βασιστήκαμε στον αλγόριθμο 4.2 για το πρόβλημα σακιδίου.

Τα βήματα που θα ακολουθήσουμε είναι τα εξής: Αρχικά θα διατυπώσουμε το πρόβλημα (πρόβλημα 4.3) έτσι ώστε να 'μοιάζει' με το πρόβλημα σακιδίου. Ακολούθως, θα κατασκευάσουμε ένα βοηθητικό πρόβλημα, στο οποίο αν τεθούν συγκεκριμένες τιμές σε παραμέτρους του είναι ισοδύναμο με το πρόβλημα 4.3. Εν συνεχεία θα δείξουμε ότι οποιοδήποτε στιγμιότυπο του βοηθητικού προβλήματος μπορεί να λυθεί μέσω αναδρομικής σχέσης. Τέλος, αξιοποιώντας την αναδρομική σχέση θα κατασκευάσουμε τον αντίστοιχο αλγόριθμο δυναμικού προγραμματισμού.

Η περιγραφή του προβλήματος βασίζεται στα σύνολα U_k και J_k τα οποία ορίσαμε στην ενότητα 4.4. Στους UH1 και UH2 οι επανασυνδεόμενοι τερματικοί κόμβοι είναι όλοι κόμβοι-φύλλα. Αυτό συνεπάγεται ότι φέρουν μόνο την κίνηση που παράγουν οι ίδιοι. Άρα ένα σύνολο H επανασυνδεόμενων

κόμβων είναι εφικτό αν και μόνον αν

$$\sum_{j \in U_k \cap H} t_j = \sum_{j \in U_k \cap H} F_j \leq W_k \qquad \forall k \in \{1, \ldots, K\}. \tag{4.14}$$

Επίσης, παρατηρούμε ότι για οποιαδήποτε $k_1, k_2 \in \{1, \ldots K\}$ με $k_1 < k_2$ ισχύει ότι

$$\sum_{j \in U_{k_1}} t_j x_j \leq \sum_{j \in U_{k_2}} t_j x_j \leq W_{k_2}.$$

Συνεπώς για οποιοδήποτε $k \in K$ το άθροισμα $\sum_{j \in U_k} t_j x_j$ θα είναι μικρότερο από τα όρια W_k, W_{k+1}, $\ldots, W_K$ και μπορούμε να αντικαταστήσουμε τα όρια W_k με τα

$$W'_k = \min \{W_k, W_{k+1}, \ldots, W_K\}. \tag{4.15}$$

Άμεση συνέπεια του ορισμού των νέων ορίων είναι ότι

$$W_1 \leq W_2 \leq \ldots \leq W_{K-1} \leq W_K. \tag{4.16}$$

Κατόπιν, τοποθετούμε τους κόμβους του U σε διάταξη $(j_1, j_2, \ldots, j_{|U|})$ με βάση τις τιμές της συνάρτησης κ, δηλαδή

$$\kappa(j_a) \leq \kappa(j_b) \qquad \forall a, b : 1 \leq a < b \leq |U|. \tag{4.17}$$

Με άλλα λόγια τοποθετούμε πρώτα τους κόμβους που ανήκουν στο σύνολο J_1, ακολούθως αυτούς του J_2, και ούτω καθεξής. Επίσης, ορίζουμε $v_a = |d_{ij_a}|$ και $w_a = t_{j_a}$ για κάθε $a \in \{1, \ldots, |U|\}$. Τέλος ορίζουμε $u_k = |U_k|$ για κάθε $k \in K$. Το πρόβλημα μπορεί πλεόν να επαναδιατυπωθεί ως εξής:

Πρόβλημα 4.3: Πρόβλημα βέλτιστης επιλογής επανασυνδεόμενων κόμβων

$$\text{maximize} \sum_{a=1}^{u_K} v_a x_a \tag{4.18α′}$$

$$\text{subject to} \sum_{a=1}^{u_k} w_a x_a \leq W_k \qquad \forall k \in \{1, \ldots, K\} \tag{4.18β′}$$

$$0 \leq W_1 \leq W_2 \leq \ldots \leq W_K \tag{4.18γ′}$$

$$0 \leq u_1 \leq u_2 \leq \ldots \leq u_K \tag{4.18δ′}$$

$$x_a \in \{0, 1\} \qquad \forall a \in \{1, \ldots, u_K\} \tag{4.18ε′}$$

Οι δυαδικές μεταβλητές x_a προσδιορίζουν αν ο κόμβος j_a εισάγεται στο σύνολο H.

Ακολούθως, κατασκευάζουμε ένα νέο βοηθητικό πρόβλημα (4.4) παραλλάσσοντας το πρόβλημα 4.3 ως εξής:

- περιορίζουμε τα διαθέσιμα αντικείμενα στα πρώτα u αντικείμενα
- εισάγουμε έναν επιπλέον περιορισμό χωρητικότητας w στο συνολικό βάρος των αντικειμένων που θα επιλέξουμε

Πρόβλημα 4.4: Βοηθητικό πρόβλημα $\Pi(u, w)$

$$\text{maximize} \sum_{a=1}^{u} v_a x_a \tag{4.19α′}$$

$$\text{subject to} \sum_{a=1}^{\min\{u,u_k\}} w_a x_a \leq \min\{w, W_k\} \qquad \forall k \in \{1, \ldots, K\} \tag{4.19β′}$$

$$0 \leq W_1 \leq W_2 \leq \ldots \leq W_K \tag{4.19γ′}$$

$$0 \leq u_1 \leq u_2 \leq \ldots \leq u_K \tag{4.19δ′}$$

$$0 < u \leq u_K \tag{4.19ε′}$$

$$x_a \in \{0, 1\} \qquad \forall a \in \{1, \ldots, u\} \tag{4.19στ′}$$

Είναι προφανές ότι το πρόβλημα 4.3 αποτελεί ειδική περίπτωση του προβλήματος 4.4 για $w = W_K$ και $u = u_K$.

Για δεδομένο u ορίζουμε

$$k' = \kappa(j_u). \tag{4.20}$$

Άμεση συνέπεια είναι ότι για κάθε k_1, k_2 με $k_1 < k' \leq k_2$ ισχύει ότι

$$u_{k_1} < u \leq u_{k'} \leq u_{k_2} \tag{4.21}$$

$$W_{k_1} < W_{k'} \leq W_{k_2} \tag{4.22}$$

Θα δείξουμε ότι η βέλτιστη λύση, έστω $\mathbf{x}^*(u, w)$ με τιμή $f(u, w)$, για το πρόβλημα $\Pi(u, w)$ δίνεται από την παρακάτω αναδρομική σχέση.

$$\mathbf{x}^*(u,w) = \begin{cases} \mathbf{0} & \text{if } u = 0 \text{ or } w = 0 \\ \mathbf{x}^*(u, W_{k'}) & \text{else if } W_{k'} < w \\ [\mathbf{x}^*(u-1, w-w_u), 1] & \text{else if } \begin{cases} w_u < w \leq W_{k'} \text{ and} \\ v_u + f(u-1, w-w_u) > f(u-1,w) \end{cases} \\ [\mathbf{x}^*(u-1,w), 0] & \text{else} \end{cases} \tag{4.23}$$

Πρόταση 4.6: $\mathbf{x}^*(0,w) = \mathbf{x}^*(u,0) = \mathbf{0}$ και $f(0,w) = f(u,0) = 0$.

Πρόταση 4.7: Αν $w > W_{k'}$ τότε $\mathbf{x}^*(u,w) = \mathbf{x}^*(u, W_{k'})$

Απόδειξη: Αρκεί να δείξουμε ότι τα προβλήματα $\Pi(u,w)$ και $\Pi(u, W_{k'})$ έχουν το ίδιο σύνολο εφικτών λύσεων.

$\Leftarrow$: $\min\{W_{k'}, W_k\} \leq \min\{w, W_k\}$ για οποιαδήποτε k και άρα κάθε λύση του $\Pi(u, W_{k'})$ είναι λύση και του $\Pi(u,w)$.

$\Rightarrow$: για κάθε λύση του $\Pi(u,w)$ και $k \in \{1,\ldots,K\}$ έχουμε ότι:

- Αν $k < k'$ τότε $W_k \leq W_{k'} < w \Rightarrow \min\{w, W_k\} = W_k = \min\{W_{k'}, W_k\}$.
- Αν $k \geq k'$ τότε $u \leq u_{k'} \leq u_k \Rightarrow \min\{u, u_k\} = u = \min\{u, u_{k'}\}$ και άρα

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a = \sum_{a=1}^{\min\{u,u_{k'}\}} w_a x_a \leq \min\{w, W_{k'}\} = W_{k'} = \min\{W_{k'}, W_k\}.$$

Συνεπώς $\sum_{a=1}^{\min\{u,u_k\}} w_a x_a \leq \min\{W_{k'}, W_k\}$ για κάθε $\forall k \in \{1,\ldots,K\}$ και άρα είναι και λύση του $\Pi(u, W_{k'})$. □

Στην περίπτωση που $w \leq W_{k'}$ ελέγχουμε αν το βάρος του u υπερβαίνει το όριο w. Αν $w_u > w$ τότε δεν υπάρχει εφικτή λύση που να περιλαμβάνει τον u και άρα η βέλτιστη λύση είναι η $[\mathbf{x}^*(u-1,w), 0]$. Αν αντίθετα $w_u \leq w$ θα πρέπει να επιλέξουμε την καλύτερη λύση, μεταξύ της $[\mathbf{x}^*(u-1,w), 0]$ και της βέλτιστης απο τις λύσεις που περιλαμβάνουν το u. Από τις προτάσεις 4.8 και 4.9 προκύπτει ότι η βέλτιστη λύση αν $w_u < w \leq W_{k'}$ και $x_u = 1$ είναι η $[\mathbf{x}^*(u-1, w-w_k), 1]$.

Πρόταση 4.8: Αν $w_u < w \leq W_{k'}$ και $(x_1,\ldots,x_{u-1},1)$ εφικτή λύση του $\Pi(u,w)$ τότε $(x_1,\ldots,x_{u-1})$ είναι εφικτή λύση του $\Pi(u-1, w-w_u)$.

Απόδειξη:

- Για $k \geq k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a = \sum_{a=1}^{u} w_a x_a = \sum_{a=1}^{u-1} w_a x_a + w_u = \sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a + w_u.$$

Εφαρμόζοντας στην (4.19β′) προκύπτει ότι

$$\sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a \leq \min\{w, W_k\} - w_u \overset{(w \leq W_{k'} \leq W_k)}{=} \min\{w - w_u, W_k\}. \tag{4.24}$$

- Για $k < k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a \overset{(u_k \leq u-1 < u)}{=} \sum_{a=1}^{\min\{u,u_k\}} w_a x_a \overset{(4.19\beta')}{\leq} W_k.$$

Επίσης,

$$\sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a \overset{(u_k \leq u_{k'})}{\leq} \sum_{a=1}^{\min\{u-1,u_{k'}\}} w_a x_a \overset{(4.24) \text{ για } k'=k}{\leq} w - w_u.$$

Συνεπώς $\sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a \leq \min\{w - w_u, W_k\}$ για κάθε $k \in \{1, \ldots, K\}$ και άρα η $(x_1, \ldots, x_{u-1})$ είναι εφικτή λύση του $\Pi(u-1, w-w_u)$. □

Πρόταση 4.9: Αν $w_u < w \leq W_{k'}$ και $(x_1, \ldots, x_{u-1})$ εφικτή λύση του $\Pi(u-1, w-w_u)$ τότε $(x_1, \ldots, x_{u-1}, 1)$ είναι εφικτή λύση του $\Pi(u, w)$.

Απόδειξη:

- Για $k \geq k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a = \sum_{a=1}^{u} w_a x_a = \sum_{a=1}^{u-1} w_a x_a + w_u = \sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a + w_u$$
$$\overset{(4.19\beta')}{\leq} \min\{w - w_u, W_k\} + w_u \overset{w \leq W_{k'} \leq W_k}{=} \min\{w, W_k\}$$

- Για $k < k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a \overset{(u_k \leq u-1 < u)}{=} \sum_{a=1}^{\min\{u-1,u_k\}} w_a x_a \overset{(4.19\beta')}{\leq} \min\{w - w_u, W_k\} \leq \min\{w, W_k\}$$

Συνεπώς $\sum_{a=1}^{\min\{u,u_k\}} w_a x_a \leq \min\{w, W_k\}$ για κάθε $k \in \{1, \ldots, K\}$ και άρα η $(x_1, \ldots, x_{u-1}, 1)$ είναι εφικτή λύση του $\Pi(u, w)$. □

Αλγόριθμος 4.9: Διαδικασία κατασκευής συνόλου για τους αλγορίθμους UH1-DP και UH2-DP

```
1  for u ← 0 to u_K do
2  |   for w ← 0 to W_K do
3  |   |   if u = 0 or w = 0 then
4  |   |   |   A[u, w] ← 0
5  |   |   else if w > W_{k'[u]} then
6  |   |   |   A[u, w] ← A[u, W_{k'[u]}]
7  |   |   else if w_u ≤ w and v_u + A[u − 1, w − w_u] > A[u − 1, w] then
8  |   |   |   A[u, w] ← v_u + A[u − 1, w − w_u]
9  |   |   |   B[u, w] ← 1
10 |   |   else
11 |   |   |   A[u, w] ← A[u − 1, w]
12 |   |   |   B[u, w] ← 0
13 |   |   end
14 |   end
15 end
16 w ← W_K
17 for u ← u_K downto 1 do
18 |   w ← min{w, W_{k'[u]}}
19 |   if B[u, w] = 1 then
20 |   |   x[u] ← 1
21 |   |   w ← w − w_u
22 |   else
23 |   |   x[u] ← 0
24 |   end
25 end
26 return x
```

Με βάση την αναδρομική σχέση (4.23) κατασκευάζουμε το αλγόριθμο δυναμικού προγραμματισμού 4.9. Σε πρώτη φάση (γραμμές 1-15) συμπληρώνουμε δύο βοηθητικούς πίνακες A και B διαστάσεων $u_K \times W_K$. Σε κάθε $A[u, w]$ αποθηκεύουμε τη βέλτιστη τιμη για το πρόβλημα $\Pi(u, w)$ ενώ στις δυαδικές μεταβλητές $B[u, w]$ δηλώνεται αν ο κόμβος u περιλαμβάνεται στη βέλτιστη λύση του προβλήματος $\Pi(u, w)$. Κατόπιν (γραμμές 16-25), κατασκευάζουμε το διάνυσμα της βέλτιστης λύσης του $\Pi(u_K, W_K)$ διασχίζοντας τον πίνακα B από-πάνω-προς-τα-κάτω όσον αφορά τις τιμές u.

Ορίζουμε ως UH1-DP και UH2-DP τις παραλλαγές των UH1 και UH2 αντίστοιχα που χρησιμοποιούν τον αλγόριθμο 4.9 για την επιλογή των επανασυνδεόμενων κόμβων μίας αναβάθμισης. Όσον αφορά τον UH1-DP σε κάθε κύκλο του κύριου βρόχου και για κάθε εξεταζόμενο κόμβο υπολογίζεται ένα σύνολο επανασυνδεόμενων κόμβων και άρα εκτελούμε μία φορά τον αλγόριθμο 4.9. Ο αλγόριθμος 4.9 κυριαρχείται από τη διαδικασία συμπλήρωσης των βοηθητικών πινάκων A και B. Δεδομένου ότι το πλήθος των υποψήφιων κόμβων είναι το πολύ N και το όριο W_k δε μπορεί να υπερβαίνει τη μέγιστη χωρητικότητα Z^L, η κατασκευή ενός συνόλου H απαιτεί $\mathcal{O}(NZ^L)$ βήματα.

Ο UH2-DP από την άλλη δεν περιορίζει τις εξεταζόμενες αναβαθμίσεις προς ένα συγκεκριμένο τύπο ζεύξης, αλλά προς οποιονδήποτε τύπο $l \in \{\lambda_i, \ldots, L\}$. Συνεπώς, πρέπει να υπολογιστούν

περισσότερα του ενός συνόλα H ανά αναβαθμιζόμενο κόμβο i. Παρατηρούμε όμως ότι για δεδομένο i τα προβλήματα επιλογής επανασυνδεόμενων κόμβων για τις διάφορες τιμές του l διαφοροποιούνται μόνο ως προς τις τιμές Z^l. Βάση της σχέσης (4.3) η διαφοροποίηση αυτή επηρεάζει μόνο το τελευταίο από τα όρια W_K δηλαδή το όριο w στο πρόβλημα 4.4. Συνεπώς αρκεί να υπολογίσουμε μία μόνο φορά τους βοηθητικούς πίνακες A και B για το μέγιστο W_K $(l = L)$ και να εξάγουμε τις λύσεις για κάθε πρόβλημα ξεχωριστά από τον B. φροντίζοντας κάθε φορά να ξεκινάμε τη διάσχιση του από την κατάλληλη θέση w. Συνολικά λοιπόν για κάθε εξεταζόμενο κόμβο απαιτούνται το πολύ Z^L βήματα για τη συμπλήρωση των βοηθητικών πινάκων και L βήματα για τις διασχίσεις του B (L διασχίσεις, βήματα ανά διάσχιση) επομένως η χρονική πολυπλοκότητα είναι και πάλι $\mathcal{O}(NZ^L)$.

Συμπερασματικά, μία επανάληψη του κύριου βρόγχου των UH1-DP και UH2-DP απαιτεί το πολύ

$$\mathcal{O}(N^2 Z^L),$$

βήματα, ενώ η συνολική χρονική πολυπλοκότητα των UH1-DP και UH2-DP είναι

$$\mathcal{O}(N^3 Z^L).$$

4.9 Ο αλγόριθμος UH3-DP

Ο ευρετικός αλγόριθμος UH3 επιτρέπει την επανασύνδεση ήδη αναβαθμισμένων κόμβων, αντίθετα με τους UH1 και UH2 οι οποίοι περιορίζονται σε κόμβους-φύλλα. Το γεγονός αυτό δυσκολεύει τη βέλτιστη επιλογή των επανασυνδεόμενων κόμβων καθώς πλέον η επιβάρυνση για τους κόμβους m_k του P_i δε θα ισούται απαραίτητα με το άθροισμα της κίνησης των επανασυνδεόμενων κόμβων. Όπως δείξαμε στην ενότητα 4.7 ένας επανασυνδεόμενος κόμβος θα επιβαρύνει μόνο εάν ταυτόχρονα *δεν* επανασυνδέουμε και κάποιον πρόγονό του. Συνεπώς στην περίπτωση του UH3 ο περιορισμός (4.14) αντικαθίσταται με τον

$$\sum_{j:H\cap U_k\cap P_j=\{j\}} t_j \leq W_k.$$

Όπως και για τους UH1-DP και UH2-DP, έτσι και για τον UH3-DP τον οποίο θα παρουσιάσουμε σε αυτή την ενότητα αναδιατάσσουμε τους κόμβους σύμφωνα με τη σχέση 4.17, δηλαδή με βάση τα σύνολα J_m στα οποία ανήκουν. Πρώτοι τοποθετούνται οι κόμβοι του συνόλου J_1, ακολουθούν αυτοί του J_2, και ούτω καθεξής. Ωστόσο, η διάταξη των κόμβων του ίδιου J_m δε μας είναι αδιάφορη. Επιλέγουμε να διατάξουμε τους κόμβους αυτούς με βάση τη σειρά μεταδιατεταγμένης διάσχισης (post-order traversal). Θα αναφερόμαστε στο εξής στους κόμβους του U ως $j_1, j_2, \ldots, j_{|U|}$ με βάση τη θέση τους στη νέα διάταξη.

Η μεταδιατεταγμένη διάσχιση εξασφαλίζει ότι ένα κόμβος έπεται οποιουδήποτε απογόνου του. Επίσης, οι $|V_{j_a} - 1|$ απόγονοι ενός κόμβου j_a [2] βρίσκονται στις ακριβώς προηγούμενες από αυτόν θέσεις Άρα αν ορίσουμε ως $g_a = a - |V_{j_a}|$ τότε ισχύει η παρακάτω πρόταση.

Πρόταση 4.10: j_b απόγονος του j_a αν και μόνον αν $g_a < b < a$.

Το σχήμα 4.2 απεικονίζει την εφαρμογή της διάταξης σε παράδειγμα ενώ ο πίνακας 4.1 τις τιμές g_a για το παράδειγμα.

Παρατηρούμε ότι εάν ένας κόμβος περιλαμβάνεται στη βέλτιστη λύση τότε θα περιλαμβάνονται και όλοι οι απόγονοί του αφού η προσθήκη τους θα γίνει χωρίς επιβάρυνση. Ως εκ τούτου, επιλέγουμε να εκφράσουμε τις εφικτές λύσεις του προβλήματος με διάνυσμα δυαδικών μεταβλητών x_a για κάθε $a \in \{1, \ldots, |U|\}$ όπου $x_a = 1$ αν ο κόμβος j_a περιλαμβάνεται στο σύνολο H χωρίς να περιλαμβάνεται κάποιος πρόγονός του, δηλαδή μόνο στη περίπτωση που επιβαρύνει τα σακίδια-ζεύξεις. Το πρόβλημα πλέον μπορεί να διατυπωθεί ως εξής:

Πρόβλημα 4.5: Πρόβλημα εύρεσης βέλτιστου συνόλου επανασυνδεόμενων κόμβων για τον αλγόριθμο UH3

$$\text{maximize} \sum_{a=1}^{u_K} v_a x_a \tag{4.25α′}$$

$$\text{subject to} \sum_{a=1}^{u_k} w_a x_a \leq W_k \qquad \forall k \in \{1, \ldots, K\} \tag{4.25β′}$$

$$x_a x_b \leq 0 \qquad \forall a \in \{1, \ldots, u_K\}, b \in \{g_a, \ldots, a-1\} \tag{4.25γ′}$$

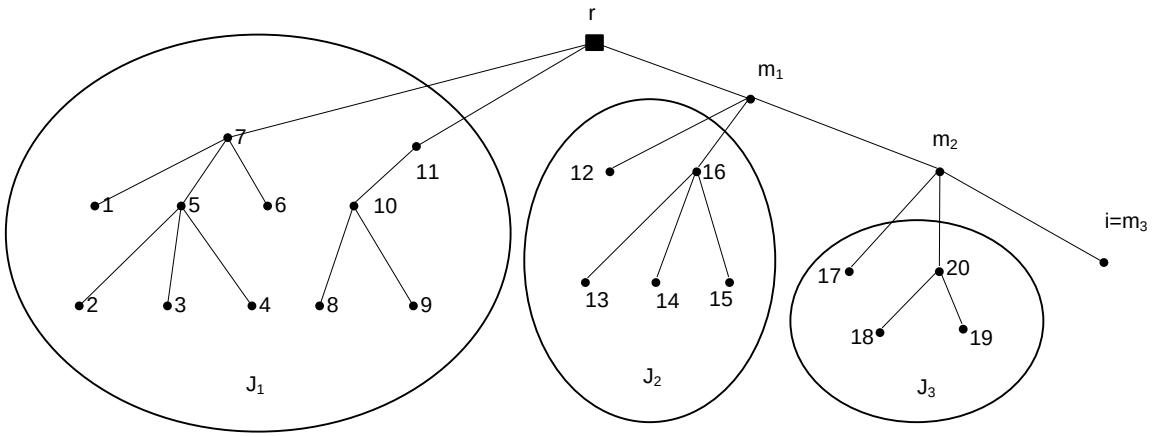
$$0 \leq W_1 \leq W_2 \leq \ldots \leq W_K \tag{4.25δ′}$$

$$0 \leq u_1 \leq u_2 \leq \ldots \leq u_K \tag{4.25ε′}$$

$$x_a \in \{0, 1\} \qquad \forall a \in \{1, \ldots, u_K\} \tag{4.25στ′}$$

όπου $v_a = \sum_{b=g_a+1}^{a} |d_{ij_b}|$ το όφελος από την επιλογή του κόμβου a και τον απογόνων του, $w_a = t_{j_a}$ και $u_k = |U_k|$ για κάθε $k \in K$.′ Κατ αναλογία με την προηγούμενη ενότητα κατασκευάζουμε ένα νέο βοηθητικό πρόβλημα (4.6) παραλλάσσοντας το πρόβλημα 4.5 έτσι ώστε να περιορίσουμε τα διαθέσιμα αντικείμενα στα πρώτα u αντικείμενα και έχουμε έναν επιπλέον περιορισμό χωρητικότητας w στο συνολικό βάρος των αντικειμένων που θα επιλέξουμε.

[2] το σύνολο V_x όπως το ορίσαμε στο κεφάλαιο 1 περιλαμβάνει τον κόμβο x και όλους του απογόνους του.



Σχήμα 4.2: Παράδειγμα ταξινόμησης υποψήφιων προς επανασύνδεση κόμβων για τον UH3-DP

a	$\lvert V_{j_a}\rvert$	g_a	απόγονοι	a	$\lvert V_{j_a}\rvert$	g_a	απόγονοι
1	1	0	$\emptyset$	11	4	7	$\{8, 9, 10\}$
2	1	1	$\emptyset$	12	1	11	$\emptyset$
3	1	2	$\emptyset$	13	1	12	$\emptyset$
4	1	3	$\emptyset$	14	1	13	$\emptyset$
5	4	1	$\{2, 3, 4\}$	15	1	14	$\emptyset$
6	1	5	$\emptyset$	16	4	12	$\{13, 14, 15\}$
7	7	0	$\{1, 2, 3, 4, 5, 6\}$	17	1	16	$\emptyset$
8	1	7	$\emptyset$	18	1	17	$\emptyset$
9	1	8	$\emptyset$	19	1	18	$\emptyset$
10	3	7	$\{8, 9\}$	20	3	17	$\{18, 19\}$

Πίνακας 4.1: Παράδειγμα ταξινόμησης κόμβων για τον UH3-DP

Πρόβλημα 4.6: Βοηθητικό πρόβλημα $\Pi'(u, w)$

$$\text{maximize} \sum_{a=1}^{u} v_a x_a \tag{4.26α′}$$

$$\text{subject to} \sum_{a=1}^{\min\{u,u_k\}} w_a x_a \leq \min\{w, W_k\} \qquad \forall k \in \{1, \ldots, K\} \tag{4.26β′}$$

$$x_a x_b \leq 0 \qquad \forall a \in \{1, \ldots, u\}, b \in \{g_a, \ldots, a-1\} \tag{4.26γ′}$$

$$0 \leq W_1 \leq W_2 \leq \ldots \leq W_K \tag{4.26δ′}$$

$$0 \leq u_1 \leq u_2 \leq \ldots \leq u_K \tag{4.26ε′}$$

$$0 < u \leq u_K \tag{4.26στ′}$$

$$x_a \in \{0, 1\} \qquad \forall a \in \{1, \ldots, u\} \tag{4.26ζ′}$$

Θα δείξουμε ότι η βέλτιστη λύση, έστω $\mathbf{x}^*(u, w)$ με τιμή $f(u, w)$, για το πρόβλημα $\Pi'(u, w)$ δίνεται από την παρακάτω αναδρομική σχέση.

$$\mathbf{x}^*(u,w) = \begin{cases} \mathbf{0} & \text{if } u = 0 \text{ or } w = 0 \\ \mathbf{x}^*(u, W_{k'}) & \text{else if } w > W_{k'} \\ [\mathbf{x}^*(g_u, w - w_u), 0, 0, \ldots, 0, 1] & \text{else if } \begin{cases} w_u \leq w \text{ and} \\ v_u + f(g_u, w - w_u) > f(u-1, w) \end{cases} \\ [\mathbf{x}^*(u-1, w), 0] & \text{else} \end{cases} \tag{4.27}$$

Για την περίπτωση όπου $u = 0$ ή $w = 0$ η λύση είναι προφανώς τετριμμένη (πρόταση 4.6). Επίσης, αν $w > W_{k'}$ ισχύει η πρόταση 4.7. Σε κάθε άλλη περίπτωση θα πρέπει εξετάσουμε αν επιλέγοντας τον κόμβο u μπορούμε να πάρουμε καλύτερη λύση από την $[\mathbf{x}^*(u-1, w), 0]$ η οποία είναι και η βέλτιστη επιλογή αν δε τον επιλέξουμε. Από τις προτάσεις 4.11 και 4.12 προκύπτει ότι η βέλτιστη λύση αν $x_u = 1$ είναι η $[\mathbf{x}^*(g_u, w - w_k), 1]$.

Πρόταση 4.11: Αν $w_u < w \leq W_{k'}$ και $(x_1, \ldots, x_{g_u}, 0, 0, \ldots, 0, 1)$ εφικτή λύση του $\Pi'(u, w)$ τότε $(x_1, \ldots, x_{g_u})$ είναι εφικτή λύση του $\Pi'(g_u, w - w_u)$.

Απόδειξη:

- Για $k \geq k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a = \sum_{a=1}^{u} w_a x_a = \sum_{a=1}^{g_u} w_a x_a + w_u = \sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a + w_u$$

Εφαρμόζοντας στην (4.26β′) προκύπτει ότι

$$\sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a \leq \min\{w, W_k\} - w_u \overset{(w\leq W_{k'}\leq W_k)}{=} \min\{w - w_u, W_k\}. \tag{4.28}$$

- Για $k < k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a \overset{(u_k\leq g_u<u)}{=} \sum_{a=1}^{\min\{u,u_k\}} w_a x_a \overset{(4.26\beta')}{\leq} W_k.$$

Επίσης,

$$\sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a \overset{(u_k\leq u_{k'})}{\leq} \sum_{a=1}^{\min\{g_u,u_{k'}\}} w_a x_a \overset{(4.28)\ \text{για}\ k'=k}{\leq} w - w_u.$$

Συνεπώς $\sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a \leq \min\{w - w_u, W_k\}$ για κάθε $k \in \{1, \ldots, K\}$.

Επίσης, από την υπόθεση η (4.26γ′) ισχύει για κάθε $a \in \{1, \ldots, u\}$ άρα ισχύει και για κάθε $a \in \{1, \ldots, g_u\}$.

Συνεπώς η $(x_1, \ldots, x_{g_u})$ είναι εφικτή λύση του $\Pi'(g_u, w - w_u)$. □

Πρόταση 4.12: Αν $w_u < w \leq W_{k'}$ και $(x_1, \ldots, x_{g_u})$ εφικτή λύση του $\Pi'(g_u, w - w_u)$ τότε η $(x_1, \ldots, x_{g_u}, 0, 0, \ldots, 0, 1)$ είναι εφικτή λύση του $\Pi'(u, w)$.

Απόδειξη:

- Για $k \geq k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a = \sum_{a=1}^{u} w_a x_a = \sum_{a=1}^{g_u} w_a x_a + w_u = \sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a + w_u$$
$$\overset{(4.26\beta')}{\leq} \min\{w - w_u, W_k\} + w_u \overset{w\leq W_{k'}\leq W_k}{=} \min\{w, W_k\}$$

- Για $k < k'$ έχουμε ότι

$$\sum_{a=1}^{\min\{u,u_k\}} w_a x_a \overset{(u_k\leq g_u<u)}{=} \sum_{a=1}^{\min\{g_u,u_k\}} w_a x_a \overset{(4.26\beta')}{\leq} \min\{w - w_u, W_k\} \leq \min\{w, W_k\}$$

Συνεπώς $\sum_{a=1}^{\min\{u,u_k\}} w_a x_a \leq \min\{w, W_k\}$ για κάθε $k \in \{1, \ldots, K\}$.

Επίσης η (4.26γ′) ισχύει

- για κάθε $a \in \{1, \ldots, g_u\}$ από την υπόθεση
- για κάθε $a \in \{g_u, \ldots, i-1\}$ αφού $x_a = 0$
- για κάθε $a = u$ αφού $x_b = 0$ για κάθε $b \in \{g_u, \ldots, i-1\}$

Συνεπώς η $(x_1, \ldots, x_{u-1}, 1)$ είναι εφικτή λύση του $\Pi'(u, w)$. □

Με βάση την αναδρομική σχέση (4.27) κατασκευάζουμε το αλγόριθμο δυναμικού προγραμματισμού 4.10. Σε πρώτη φάση (γραμμές 1-15) συμπληρώνουμε δύο βοηθητικούς πίνακες A και B διαστάσεων $u_K \times W_K$. Σε κάθε $A[u, w]$ αποθηκεύουμε τη βέλτιστη τιμή για το πρόβλημα $\Pi'(u, w)$ ενώ οι δυαδικές μεταβλητές $B[u, w]$ προσδιορίζουν αν $x_u = 1$ στη βέλτιστη λύση του προβλήματος $\Pi'(u, w)$. Κατόπιν (γραμμές 16-28), κατασκευάζουμε το διάνυσμα της βέλτιστης λύσης του $\Pi'(u_K, W_K)$ διασχίζοντας τον πίνακα B από-πάνω-προς-τα-κάτω όσον αφορά τις τιμές u.

Ορίζουμε ως UH3-DP την παραλλαγή του UH3 που χρησιμοποιεί τον αλγόριθμο 4.10 για την επιλογή των επανασυνδεόμενων κόμβων μίας αναβάθμισης. Κατά αντίστοιχο τρόπο με τον UH2-DP, απαιτείται μία επανάληψη της φάσης κατασκευής (NZ^L βήματα) συν $\mathcal{O}(L)$ κλήσεις τη φάσης διάσχισης (N βήματα) ανά εξεταζόμενο κόμβο. Συνεπώς, η χρονική πολυπλοκότητα μίας επανάληψης του κύριου βρόχου του UH3-DP είναι

$$\mathcal{O}\left(N^2 Z^L\right).$$

4.10 Υπολογιστικά Αποτελέσματα

Στην ενότητα αυτή επιχειρείται η αξιολόγηση των προτεινόμενων αλγορίθμων αναβαθμίσεων μέσω της εφαρμογής τους στα στιγμιότυπα αναφοράς (κεφάλαιο 2.8). Ο πίνακας 4.2 συνοψίζει τους τρεις εξεταζόμενους αλγορίθμους αναβαθμίσεων και τις παραλλαγές τους (-DP). Με '✓' σημειώνονται οι περιπτώσεις όπου κατά τον υπολογισμό μίας αναβάθμισης γίνεται η βέλτιστη επιλογή κόμβων που θα επανασυνδεθούν.

Χρόνοι εκτέλεσης

Τα log-log γραφήματα του σχήματος 4.3 [3] απεικονίζουν την εξέλιξη του χρόνου εκτέλεσης συναρτήσει του μεγέθους προβλήματος για κάθε αλγόριθμο και σενάριο. Ένα σημείο των γραφημάτων εκφράζει το μέσο χρόνο εκτέλεσης ενός αλγορίθμου για τα 50 στιγμιότυπα συγκεκριμένου σεναρίου και μεγέθους. Επειδή δε οι χρονομετρήσεις είχαν ακρίβεια 1ms, κάθε εκτέλεση επαναλήφθηκε 20 φορές.

Παρατηρούμε ότι σε όλες τις περιπτώσεις οι καμπύλες προσεγγίζουν ευθείες γραμμές και άρα προσεγγίζονται από συναρτήσεις της μορφής $t = \alpha N^\beta$. Εφαρμόζοντας τη μέθοδο ελαχίστων τετραγώνων και αποδεχόμενοι μόνο τους χρόνους που υπερβαίνουν το 1ms, υπολογίσαμε ότι οι τιμές των εκθε-

[3] Οι τιμές του σχήματος 4.3 παρατίθενται στον πίνακα Α′.3 του Παραρτήματος.

Αλγόριθμος 4.10: Διαδικασία κατασκευής συνόλου H για τον αλγόριθμο UH3-DP

```
for u ← 0 to u_K do
    for w ← 0 to W_K do
        if u = 0 or w = 0 then
            A[u, w] ← 0
        else if w > W_{k'[u]} then
            A[u, w] ← A[u, W_{k'[u]}]
        else if w_u ≤ w and v'_u + A[g_u, w − w_u] > A[u − 1, w] then
            A[u, w] ← v'_u + A[g_u, w − w_u]
            B[u, w] ← 1
        else
            A[u, w] ← A[u − 1, w]
            B[u, w] ← 0
        end
    end
end
w ← W
u ← u_K
repeat
    w ← min{w, W_{k'[u]}}
    if B[u, w] = 1 then
        x[u] ← 1
        w ← w − w_u
        u ← g_u
    else
        x[u] ← 0
        u ← u − 1
    end
until u = 0
return x
```

αλγόριθμος	κύρια διαδικασία	καστασκευή συνόλου επανασυνδεόμενων κόμβων H	βέλτιστο σύνολο H ?	
			μοναδιαία περίπτωση	μη μοναδιαία περίπτωση
UH1	αλγ. 4.3	αλγ. 4.5	✓	×
UH2	αλγ. 4.6	αλγ. 4.5	✓	×
UH3	αλγ. 4.7	αλγ. 4.8	×	×
UH1-DP	αλγ. 4.3	αλγ. 4.9	✓	✓
UH2-DP	αλγ. 4.6	αλγ. 4.9	✓	✓
UH3-DP	αλγ. 4.7	αλγ. 4.10	✓	✓

Πίνακας 4.2: Συγκεντρωτικός πίνακας προτεινόμενων αλγορίθμων αναβαθμίσεων

τών β κυμαίνονται μεταξύ 2,1 και 2,6. Σε κάθε περίπτωση ο συντελεστής προσδιορισμού (coefficient of determination, R^2) ήταν υψηλότερος από $0,995$.

UH1 και UH2 φαίνεται να είναι οι ταχύτεροι αλγόριθμοι με χρόνους 10-25ms στα μεγαλύτερα προβλήματα, αυτά των 150 κόμβων. Οι αντίστοιχοι χρόνοι για τον UH3 κυμαίνονται μεταξύ 70 και 100ms. Η επιβάρυνση αυτή αποδίδεται στο γεγονός ότι κατά τη διαδικασία προσδιορισμού των επανασυνδεόμενων κόμβων ο UH3, σε αντίθεση με τους UH1 και UH2, επαναϋπολογίζει τα οφέλη των υπό εξέταση κόμβων μετά από κάθε επιλογή.

Οι χρόνοι των τριών παραλλαγών -DP φαίνεται να έχουν σημαντικές διαφοροποιήσεις από σενάριο σε σενάριο. Για το σύνολο στιγμιότυπων 150 κόμβων οι χρόνοι κυμαίνονται μεταξύ 50-100ms για το σενάριο RES-U, 100-250ms για τα EXT-U, RES-NU και 990-1400ms για το EXT-NU. Η κλιμάκωση αυτή αποδίδεται στις μεγάλες διαφορές στις τιμές Z^L μεταξύ των σεναρίων (πίνακας 2.6) καθώς κατά τη διαδικασία προσδιορισμού των επανασυνδεόμενων κόμβων μίας αναβάθμισης κατασκευάζονται βοηθητικοί πίνακες διαστάσεων $|U| \times Z^L$ (όπου $|U|$ το πλήθος των υποψήφιων κόμβων). Ο αλγόριθμος UH3-DP δίνει τους χειρότερους χρόνους σε κάθε περίπτωση.

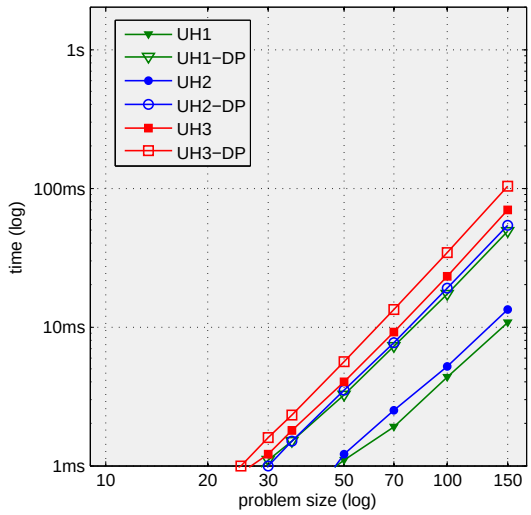
Εξαγόμενες λύσεις

Για την αξιολόγηση των εξαγόμενων λύσεων των αλγορίθμων αναβαθμίσεων καταρχήν υπολογίζουμε τις σχετικές αποστάσεις από τις βέλτιστες λύσεις για τα προβλήματα με λίγους κόμβους όπου καταφέραμε να επιλύσουμε με τεχνικές μεικτού ακέραιου γραμμικού προγραμματισμού στο προηγούμενο κεφάλαιο. Οι μέσες σχετικές αποστάσεις για τα προβλήματα αυτά απεικονίζονται στα γραφήματα του σχήματος 4.4 [4].

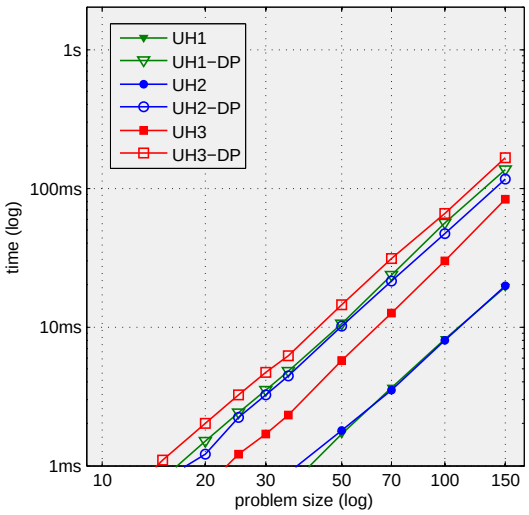
Όσον αφορά στα μεγαλύτερα προβλήματα του σεναρίου RES-U αξιοποιούμε τα κάτω όρια των βέλτιστων λύσεων που προέκυψαν από την επίλυση των γραμμικών χαλαρώσεων της διατύπωσης CISCF2. Οι μέσες σχετικές αποστάσεις των ευρετικών λύσεων από τα κάτω όρια για τα προβλήματα αυτά παρατίθενται στον πίνακα 4.3. Οι τιμές του πίνακα στην ουσία εκφράζουν τη μέγιστη απόσταση που μπορεί να έχουν οι ευρετικές λύσεις από τις βέλτιστες. Στα υπόλοιπα σενάρια δε μπορούμε να έχουμε αξιόπιστη εκτίμηση της δυνατότητας προσέγγισης των βέλτιστων λύσεων για τα μεγαλύτερα προβλήματα οπότε περιοριζόμαστε σε σύγκριση των αλγορίθμων αναβαθμίσεων μεταξύ τους. Στο σχήμα 4.5 απεικονίζονται οι μέσες σχετικές αποστάσεις των ευρετικών λύσεων από εκείνες του UH3-DP ο οποίος δίνει σε όλες τις περιπτώσεις τις καλύτερες λύσεις. [5].

Τον UH3-DP ακολουθούν κατά σειρά με σχετικά μικρές διαφορές οι UH3, UH2-DP και UH2 ενώ οι UH1-DP και UH1 φαίνεται να δίνουν με διαφορά τις χειρότερες τιμές. Επίσης παρατηρούμε ότι

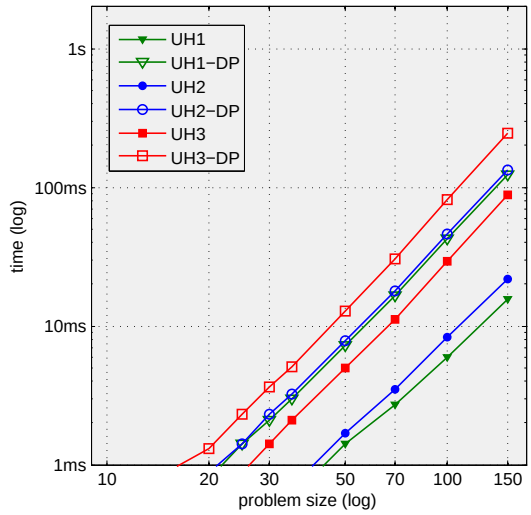
[4] Οι τιμές του σχήματος 4.4 παρατίθενται στον πίνακα Α′.4 του Παραρτήματος.
[5] Οι τιμές του σχήματος 4.5 παρατίθενται στον πίνακα Α′.5 του Παραρτήματος.



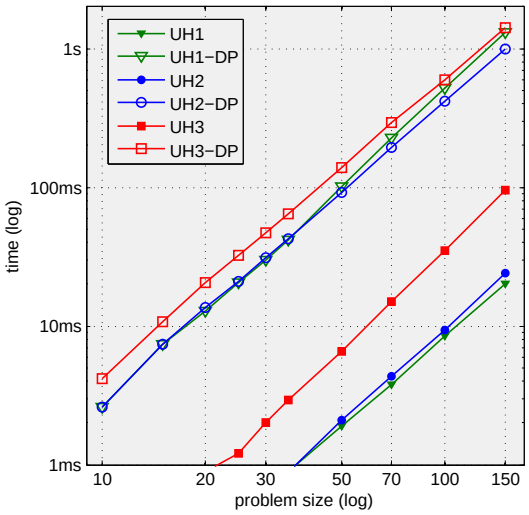
(α′) σενάριο RES-U



(β′) σενάριο EXT-U



(γ′) σενάριο RES-NU

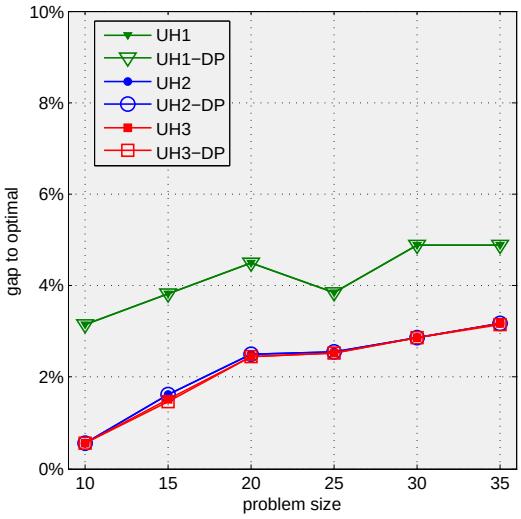


(δ′) σενάριο EXT-NU

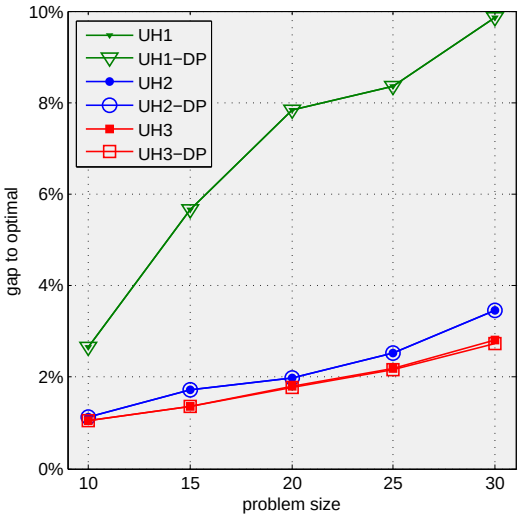
Σχήμα 4.3: Χρόνοι εκτέλεσης ευρετικών αλγορίθμων αναβαθμίσεων

size	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
50	10.27 %	10.27 %	8.97 %	8.97 %	8.94 %	8.94 %
70	9.30 %	9.30 %	8.40 %	8.40 %	8.38 %	8.37 %
100	8.12 %	8.12 %	7.44 %	7.44 %	7.42 %	7.41 %
150	7.19 %	7.19 %	6.66 %	6.66 %	6.66 %	6.65 %

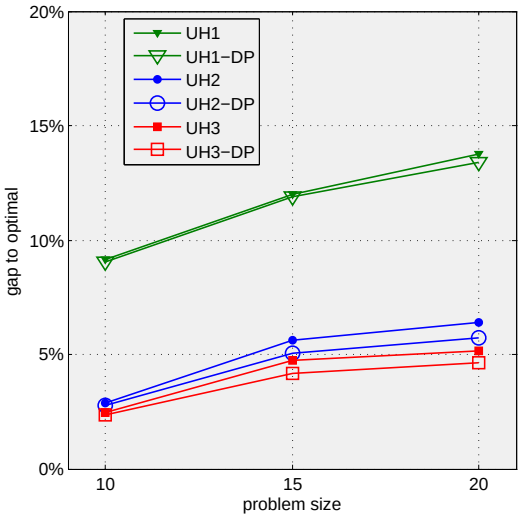
Πίνακας 4.3: Αποστάσεις ευρετικών λύσεων από τα κάτω όρια για το σενάριο RES-U



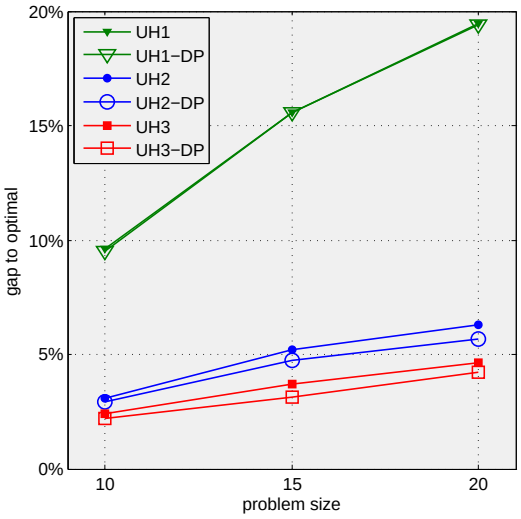
(α′) σενάριο RES-U



(β′) σενάριο EXT-U

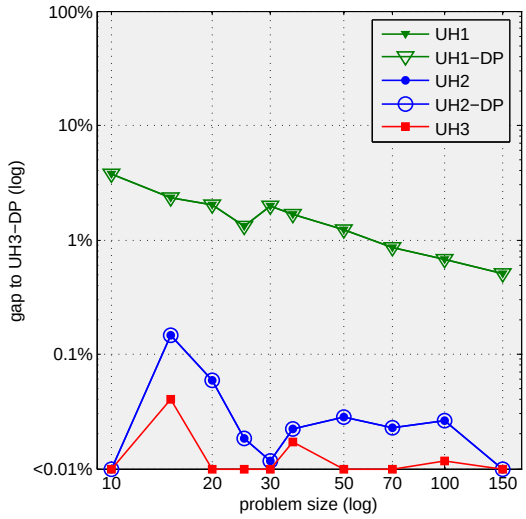


(γ′) σενάριο RES-NU

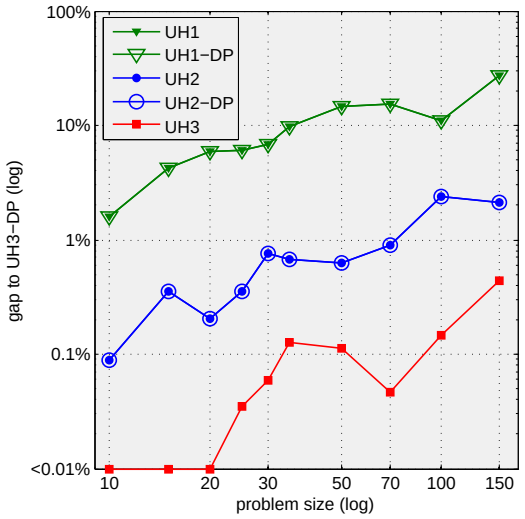


(δ′) σενάριο EXT-NU

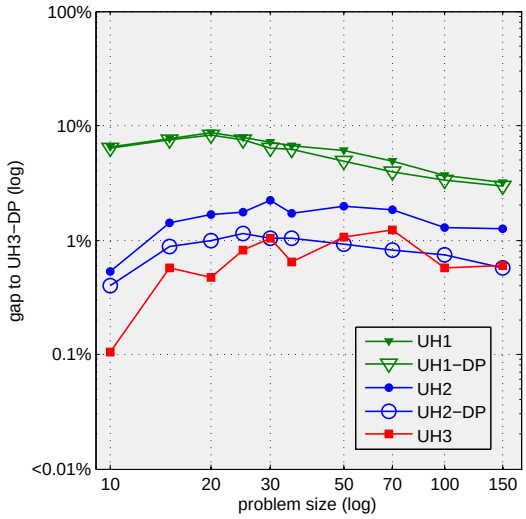
Σχήμα 4.4: Αποστάσεις ευρετικών λύσεων από τις βέλτιστες



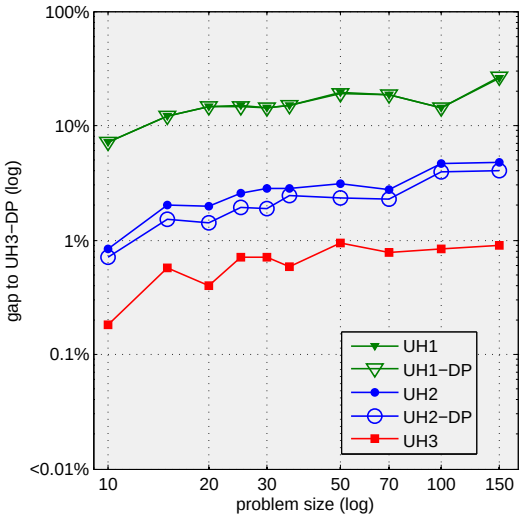
(α′) σενάριο RES-U



(β′) σενάριο EXT-U



(γ′) σενάριο RES-NU



(δ′) σενάριο EXT-NU

Σχήμα 4.5: Αποστάσεις ευρετικών λύσεων από τις λύσεις του UH3-DP

οι τρεις βασικοί αλγόριθμοι δίνουν, έστω και οριακά σε κάποιες περιπτώσεις, χειρότερες λύσεις από τις παραλλαγές τους UHx-DP. Εξαίρεση αποτελούν οι UH1 και UH2 στα δύο μοναδιαία σενάρια όπου δίνουν πανομοιότυπες λύσεις με τους UH1-DP και UH2-DP αντίστοιχα. Η ταύτιση αυτή ήταν αναμενόμενη καθότι και οι δύο προσεγγίσεις υπολογίζουν βέλτιστα το σύνολο επανασυνδεόμενων κόμβων σε μοναδιαία σενάρια.

Αναφορικά με τη προσέγγιση των βέλτιστων λύσεων στα μικρά προβλήματα όπου γνωρίζουμε τις βέλτιστες τιμές, ο UH3-DP σημειώνει αποστάσεις που δεν υπερβαίνουν το 5%. Εντούτοις, οι αποστάσεις φαίνεται να διευρύνονται καθώς αυξάνουμε το μέγεθος των προβλημάτων. Ειδικότερα για το σενάριο RES-U, οι τιμές του πίνακα 4.3 εξασφαλίζουν ότι ότι οι ποσοστιαίες αποστάσεις από τις βέλτιστες τιμές δε μπορεί παρά να είναι μονοψήφιες ακόμα και για τα μεγαλύτερα προβλήματα.

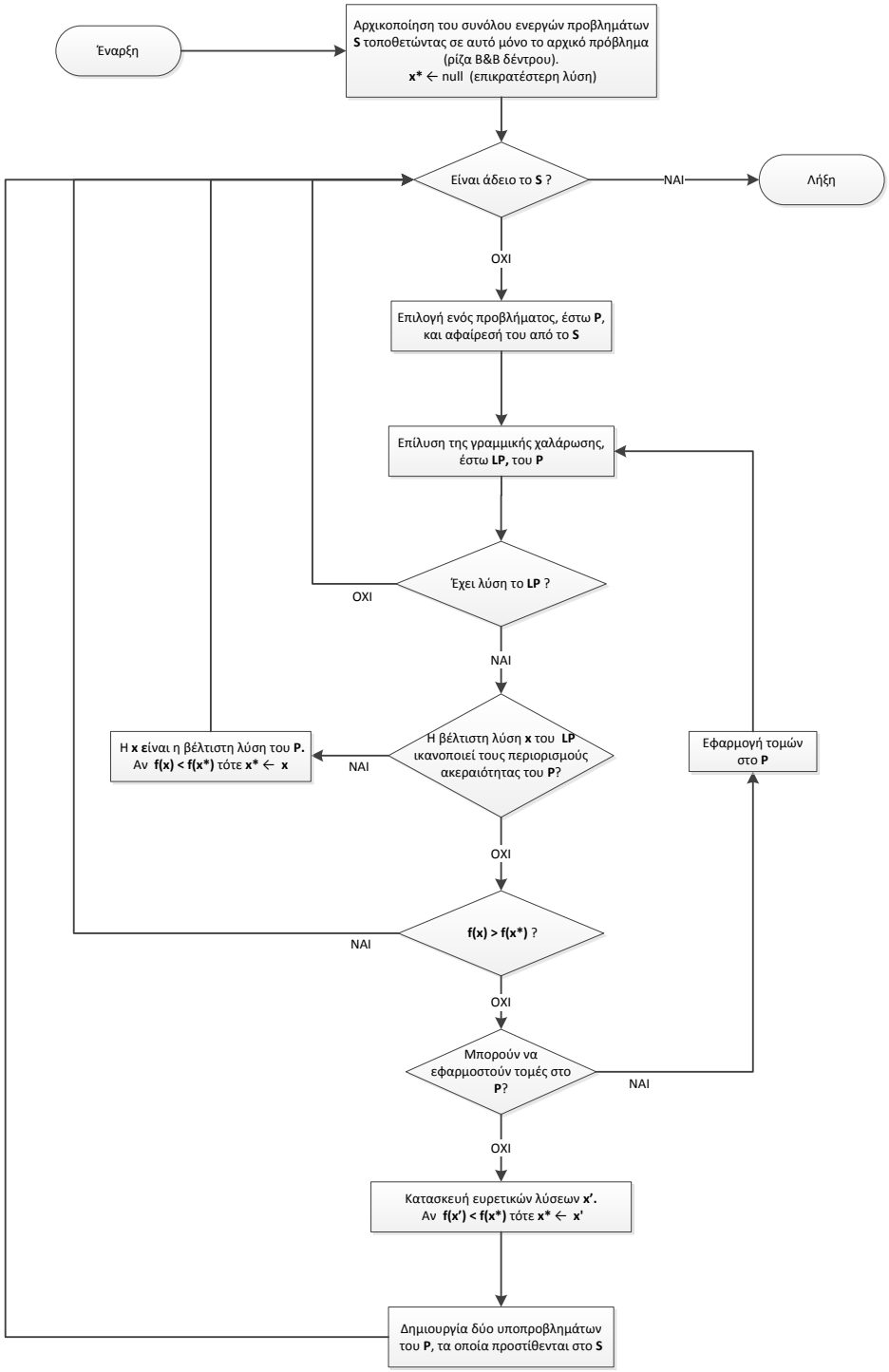
Κεφάλαιο 5

Ενσωμάτωση αναβαθμίσεων σε αλγόριθμο Διακλάδωσης και Αποκοπής

5.1 Το πακέτο λογισμικού βελτιστοποίησης CPLEX

Το πακέτο λογισμικού IBM ILOG CPLEX Optimization Studio, το οποία συχνά αναφέρεται και πιο απλά ως ILOG CPLEX, προσφέρει βιβλιοθήκες για την επίλυση προβλημάτων βελτιστοποίησης. Το όνομα προέκυψε από την αρχική υλοποίηση της μεθόδου επίλυσης προβλημάτων γραμμικού προγραμματισμού Simplex στη γλώσσα προγραμματισμού C, ωστόσο σήμερα υποστηρίζονται και άλλοι τύποι προβλημάτων. Συγκεκριμένα, επιλύονται προβλήματα με γραμμικούς ή τετραγωνικούς περιορισμούς όπου η αντικειμενική συνάρτηση είναι γραμμική ή κυρτή τετραγωνική. Οι μεταβλητές του μοντέλου μπορεί να δηλωθούν ως συνεχείς ή να περιοριστούν περεταίρω ώστε να δέχονται μόνο ακέραιες τιμές. Επίσης, μέσω του στρώματος μοντελοποίησης Concert προσφέρονται πλέον διεπαφές με άλλες γλώσσες προγραμματισμού όπως οι C++, C# και Java.

Όσον αφορά στην επίλυση προβλημάτων μεικτού ακέραιου γραμμικού προγραμματισμού το πακέτο CPLEX προσφέρει έναν αλγόριθμο Διακλάδωσης και Αποκοπής (Branch and Cut, B&C). Από την έκδοση 11.0 και μετά προσφέρεται 'μία νέα αλγοριθμική προσέγγιση', γνωστή ως Δυναμική Αναζήτηση (Dynamic Search), η οποία σύμφωνα με το εγχειρίδιο χρήσης του πακέτου (ILOG CPLEX V12.1 - User's Manual for CPLEX) βρίσκει εφικτές και βέλτιστες λύσεις πιο γρήγορα από το συμβατικό αλγόριθμο B&C σε πολλά προβλήματα. Ο αλγόριθμος Δυναμικής Αναζήτησης αποτελείται από τα ίδια δομικά στοιχεία με τον B&C όμως η ακριβής λειτουργία του δεν είναι γνωστή. Συν τοις άλλοις, η Δυναμική Αναζήτηση, σε αντίθεση με το συμβατικό B&C, δεν επιτρέπει την τροποποίηση βημάτων της διαδικασίας βελτιστοποίησης μέσω των Επανακλήσεων Ελέγχου. Για το λόγο αυτό οι ενσωμάτωσεις αλγορίθμων αναβαθμίσεων εφαρμόστηκαν στο προσφερόμενο αλγόριθμο B&C.



Σχήμα 5.1: Ο Αλγόριθμος Διακλάδωσης και Αποκοπής του πακέτου CPLEX

5.2 Ο αλγόριθμος B&C του πακέτου λογισμικού CPLEX

Ο αλγόριθμος Branch and Cut (B&C) του πακέτου λογισμικού CPLEX (αλγ. 5.1) διαχειρίζεται ένα δέντρο αναζήτησης. Κάθε κόμβος του δέντρου αυτού αναπαριστά ένα υποπρόβλημα MILP το οποίο θα εξεταστεί. Κατά την εκτέλεση του ο αλγόριθμος διατηρεί στη μνήμη την *επικρατέστερη* (x^*) από τις εφικτές λύσεις που έχουν βρεθεί καθώς επίσης και μία λίστα με τα *ενεργά* υποπροβλήματα (S), δηλαδή τα υποπροβλήματα που δεν έχουν ακόμα εξεταστεί. Κατά την εκκίνηση θεωρείται ως μοναδικό ενεργό το αρχικό πρόβλημα.

Σε κάθε επανάληψη του κύριου βρόχου επιλέγεται και εξετάζεται ένα υποπρόβλημα (P) από τη λίστα ενεργών υποπροβλημάτων. Κατά κανόνα επιλέγεται ένας από τους δύο νέους κόμβους εάν στην προηγούμενη επανάληψη είχαμε διακλάδωση. Σε αντίθετη περίπτωση επιλέγεται ο κόμβος με το βέλτιστο όριο γραμμικής χαλάρωσης.

Αφού επιλυθεί η γραμμική χαλάρωση (LP) του επιλεγμένου προβλήματος γίνονται οι εξής έλεγχοι.

- Αν δεν υπάρχει εφικτή λύση για το LP συμπεραίνουμε ότι ούτε το P μπορεί να δώσει κάποια εφικτή λύση.
- Αν η βέλτιστη λύση του LP ικανοποιεί τους περιορισμούς ακεραιότητας του P τότε αποτελεί βέλτιστη λύση και για το P. Συνεπώς ενημερώνεται η επικρατέστερη λύση εάν αυτή βελτιώνεται.
- Αν η βέλτιστη λύση του LP είναι χειρότερη από την τρέχουσα επικρατέστερη τότε οποιαδήποτε και αν είναι η βέλτιστη λύση του P δεν είναι δυνατόν να βελτιώνει την επικρατέστερη. Στην περίπτωση ατή έχουμε το *κλάδεμα* (pruning) του κόμβου.

Σε οποιαδήποτε από τις παραπάνω περιπτώσεις οι δυνατότητες του συγκεκριμένου υποπροβλήματος έχουν εξαντληθεί οπότε τερματίζεται η τρέχουσα επανάληψη του βρόχου.

Εν συνεχεία, ενδέχεται να αναζητηθούν τομές. Οι τομές (κεφ. 3.1.1) είναι περιορισμοί που προστίθενται στο ενεργό μοντέλο αποκλείοντας περιοχές με μη ακέραιες λύσεις. Η προσθήκη τομών συνήθως μειώνει τον αριθμό των κόμβων που θα χρειαστεί για να επιλυθεί το πρόβλημα. Τομές μπορούν να εφαρμοστούν σε οποιοδήποτε κόμβο, ωστόσο δίνεται μεγαλύτερη έμφαση στην εύρεση τομών στο αρχικό πρόβλημα καθώς αυτές θα ισχύουν και σε οποιοδήποτε υποπρόβλημα. Στην περίπτωση που εφαρμοστεί κάποια τομή τότε το τροποποιημένο LP πρόβλημα επιλύεται και επαναλαμβάνονται οι παραπάνω έλεγχοι.

Ο αλγόριθμος επιχειρεί ανά τακτές επαναλήψεις να κατασκευάσει εφικτές λύσεις από την μερική λύση που ορίζει ο τρέχων κόμβος με στόχο τη βελτίωση της επικρατέστερης. Το όφελος από τη βελτίωση της επικρατέστερης λύσης είναι διττό. Αφενός, σε περιπτώσεις όπου η διαδικασία branch and bound δε μπορεί να ολοκληρωθεί σε λογικά χρονικά πλαίσια υποχρεωτικά θα συμβιβαστούμε με

την επικρατέστερη λύση τη χρονική στιγμή που θα διακόψουμε τη διαδικασία. Αφετέρου, αυξάνονται οι περιπτώσεις κλαδέματος άχρηστων κλαδιών και άρα οδηγούμαστε σε μείωση του συνολικού χρόνου εκτέλεσης της διαδικασίας.

Σε περίπτωση που ακόμα και μετά από τυχών προσθήκες τομών στο μοντέλο δεν έχουμε οδηγηθεί σε κάποια από τις προαναφερθείσες συνθήκες τερματισμού της επανάληψης γίνεται διακλάδωση του P. Για τη διακλάδωση επιλέγεται μία ακέραια μεταβλητή της οποίας η τιμή δεν έχει ακόμα προσδιοριστεί και διαχωρίζεται το σύνολο τιμών της σε δύο ξένα σύνολα. Κατά τον τρόπο αυτό παράγονται δύο υποπροβλήματα του P τα οποία προστίθενται στο S.

5.3 Δυνατότητες παραμετροποίησης του αλγορίθμου B&C

Το πακέτο CPLEX προσφέρει δυνατότητες παραμετροποίησης του αλγορίθμου Branch and Cut. Σε πρώτο επίπεδο είναι δυνατή η επιλογή στρατηγικών σε επιμέρους διαδικασίες του αλγορίθμου (ILOG CPLEX V12.1 - Parameters Reference Manual).

Για παράδειγμα, όσον αφορά στη διαδικασία επιλογής κόμβου στην περίπτωση που ο τελευταίος κόμβος δεν έδωσε απογόνους δίνονται οι εξής εναλλακτικές στρατηγικές. Σύμφωνα με την προεπιλεγμένη διαδικασία σε κάθε επανάληψη επιλέγεται ο κόμβος με το βέλτιστο όριο γραμμικής χαλάρωσης η οποία δίνει συνήθως και τους χαμηλότερους συνολικούς χρόνους εκτέλεσης. Εναλλακτικά μπορεί να επιλέγεται ο κόμβος που εκτιμάται ότι προσεγγίζει ευκολότερα μία εφικτή λύση καλής ποιότητας, στρατηγική που προτιμάται σε περιπτώσεις όπου η εύρεση εφικτών λύσεων είναι δύσκολη, ή διάσχιση του δέντρου διακλάδωσης κατά βάθος. Η διάσχιση κατά βάθος συνήθως κρατάει τη λίστα των ενεργών κόμβων σε μικρά μεγέθη όμως παρόλο που ο χρόνος επίλυσης των επιμέρους προβλημάτων βαίνει μειούμενος καθώς πηγαίνουμε σε μεγαλύτερο βάθος, ο συνολικός χρόνος εκτέλεσης χειροτερεύει σημαντικά.

Μεταξύ άλλων, δίνεται επίσης η δυνατότητα ρύθμισης της συχνότητας εφαρμογής ευρετικών διαδικασιών, της έμφασης σε διάφορους τύπους τομών και της ιεράρχησης μεταβλητών για τη διακλάδωση. Ακόμα μπορεί να επιλεγεί εάν ο αλγόριθμος θα δώσει έμφαση στην εύρεση όσο το δυνατόν καλύτερης εφικτής λύσης ή στη γρηγορότερη απόδειξης της βέλτιστης.

Σε δεύτερο επίπεδο επιτρέπεται η επίβλεψη και καθοδήγηση του επιλυτή δίνοντας τη δυνατότητα να εκτελεστούν κομμάτια κώδικα του χρήστη σε διάφορα στάδια του αλγορίθμου Branch and Cut. Η εκτέλεση του κώδικα χρήστη γίνεται μέσω της υλοποίησης *επανακλήσεων* (callbacks). Υπάρχουν δύο βασικοί τύποι επανακλήσεων: οι πληροφοριακές επανακλήσεις (informational and query callbacks),

που δίνουν πρόσβαση σε πληροφορίες για την πρόοδο της βελτιστοποίησης, και οι επανακλήσεις ελέγχου (control callbacks), οι οποίες δίνουν τη δυνατότητα ελέγχου της διαδικασίας branch & cut. Οι επανακλήσεις ελέγχου κατηγοριοποιούνται με βάση το βήμα του αλγορίθμου Branch and Cut στο οποίο εκτελούνται:

- node callback: επιτρέπει τον ορισμό του επόμενου κόμβου που θα εξεταστεί.
- solve callback: επιτρέπει την τροποποίηση της διαδικασίας επίλυσης των γραμμικών χαλαρώσεων.
- user cut callback: επιτρέπει την προσθήκη ειδικών τομών για το συγκεκριμένο πρόβλημα.
- lazy constraint callback: επιτρέπει την προσθήκη περιορισμών, οι οποίοι όμως δε θα λαμβάνονται υπόψιν κατά την επίλυση των γραμμικών χαλαρώσεων, παρά μόνο εάν παραβιάζονται από την εξαγόμενη λύση.
- heuristic callback (επανάκληση ευρετικής διαδικασίας): επιτρέπει την εκτέλεση ευρετικού αλγορίθμου με στόχο την εύρεση νέας επικρατούσας λύσης.
- branch callback: επιτρέπει την τροποποίηση της διαδικασίας διακλάδωσης.
- incumbent callback: επιτρέπει την εξέταση ή/και την απόρριψη μίας νέας πιθανής επικρατέστερης λύσης.

5.4 Υλοποίηση επανάκλησης ευρετικής διαδικασίας

Η βασική ιδέα πίσω από το συνδυασμό του αλγορίθμου B&C με έναν αλγόριθμο αναβαθμίσεων είναι η αξιοποίηση από τον πρώτο των ευρετικών λύσεων καλής ποιότητας που παράγει ο δεύτερος. Μία αρχική προσέγγιση θα ήταν η τροφοδότηση του αλγορίθμου B&C κατά την εκκίνηση του με την ευρετική λύση ενός αλγορίθμου αναβαθμίσεων, Η στρατηγική της εκκίνησης μίας κατά κανόνα χρονοβόρας διαδικασίας -εν προκειμένω του αλγορίθμου B&C- λαμβάνοντας υπόψιν τα αποτελέσματα μίας άλλης συντομότερης διαδικασίας -ενός αλγορίθμου αναβαθμίσεων- αναφέρεται και ως warm start και υποστηρίζεται από το πακέτο CPLEX.

Ωστόσο, η κατασκευή ευρετικών λύσεων σε κάθε εξεταζόμενο κόμβο του δέντρου B&C ενδεχομένως θα οδηγούσε σε ταχύτερη βελτίωση της επικρατούσας λύσης. Για να το επιτύχουμε αυτό απαιτήθηκε η τροποποίηση των αλγορίθμων αναβαθμίσεων ώστε να παράγουν λύσεις που να είναι εφικτές και για τα αντίστοιχα υποπροβλήματα. Οι κλήσεις των τροποποιημένων αλγορίθμων αναβαθμίσεων γίνονται μέσω της υλοποίησης μίας 'επανάκλησης ευρετικής διαδικασίας' του πακέτου CPLEX.

Στην περίπτωση του προβλήματος MLCMST η διατύπωση SCF2 εμφανίζεται να δίνει τους καλύτερους χρόνους επίλυσης μεταξύ των MILP διατυπώσεων που παρουσιάστηκαν στο κεφ. 3 οπότε την επιλέγουμε για την υλοποίηση της ενσωμάτωσης. Ένα διάνυσμα λύσης της SCF2 περιλαμβάνει τις

δυαδικές μεταβλητές x_{ij}^l, οι οποίες προσδιορίζουν αν έχει εγκατασταθεί μία ζεύξη τύπου l από τον κόμβο i προς τον κόμβο j, και τις συνεχείς μεταβλητές f_{ij}, οι οποίες προσδιορίζουν τη ροή κίνησης τον κόμβο i προς τον κόμβο j. Δεδομένου ότι οι διακλαδώσεις γίνονται πάντα σε μη συνεχείς μεταβλητές θα επιλέγεται υποχρεωτικά μία από τις x_{ij}^l με τον ένα κλάδο να παίρνει την τιμή 0 και τον άλλο την τιμή 1. Συνεπώς, κατά την κατασκευή μίας ευρετικής λύσης θα είναι υποχρεωτική η ενσωμάτωση κάποιων ζεύξεων (για τις $x_{ij}^l = 1$) ενώ θα απαγορεύεται η επιλογή κάποιων άλλων (για τις $x_{ij}^l = 0$).

Οι αλγόριθμοι αναβαθμίσεων τροποποιήθηκαν ώστε οι εξαγόμενες λύσεις να τηρούν τους επιβαλλόμενους περιορισμούς. Οι αλλαγές είναι ενιαίες για όλους του αλγορίθμους και είναι στην κατεύθυνση της διατήρησης της εφικτότητας της λύσης σε όλα τα ενδιάμεσα βήματα.

Συγκεκριμένα, για κάθε επιβαλλόμενο $y_{ij}^l = 1$ κάνουμε τις εξής αλλαγές:

1. Κατά τη φάση αρχικοποίησης και αφού κατασκευαστεί το αρχικό δίκτυο τοπολογίας αστέρα επανασυνδέουμε τον κόμβο i προς τον κόμβο j με ζεύξη τύπου l. Με αυτό τον τρόπο εξασφαλίζεται ότι έχουμε μία εφικτή λύση προτού εφαρμόσουμε οποιαδήποτε αναβάθμιση.
2. Ο κόμβος i αποκλείεται από οποιαδήποτε αναβάθμιση με την εξαίρεση αναβαθμίσεων προς τον τύπο l. Δεδομένου ότι ο τύπος αυτός έχει ήδη εγκατασταθεί στη ζεύξη κατά τη φάση αρχικοποίησης μία τέτοια αναβάθμιση στην ουσία δεν αλλάζει τον τύπο της ζεύξης όμως επιτρέπει την επανασύνδεση άλλων κόμβων προς τον i. Η εξέταση τέτοιων αναβαθμίσεων είναι πολύ σημαντική καθότι έτσι δίνεται η δυνατότητα αξιοποίησης της χωρητικότητας του επιβαλλόμενου τύπου.

Για κάθε επιβαλλόμενο $y_{ij}^l = 0$ κάνουμε τις εξής αλλαγές:

1. Αποκλείονται αναβαθμίσεις του κόμβου i προς τον τύπο l εάν ο πατέρας του i είναι ο j.
2. Αποκλείονται επανασυνδέσεις του κόμβου i προς τον j εάν η ζεύξη του i προς τον πατέρα του είναι τύπου l.

Προκειμένου να εξοικονομηθεί υπολογιστικός χρόνος αποφεύγουμε να εκτελέσουμε τους αλγορίθμους αναβαθμίσεων σε κάθε κόμβο του δέντρου B&C. Μετά τη διακλάδωση ενός κόμβου η ευρετική λύση που κατασκευάσαμε γι αυτόν θα είναι μεν εφικτή για τον έναν του απόγονο αλλά θα παραβιάζει το νέο περιορισμό στον έτερο απόγονο. Όσον αφορά στον 'εφικτό' απόγονο η ευρετική λύση που θα παίρναμε θα ήταν πιθανότατα πανομοιότυπη με εκείνη του γονέα του εφόσον εισάγεται μόνο ένας επιπλέον περιορισμός, τον οποίο η λύση αυτή σίγουρα δεν παραβιάζει. Για το λόγο αυτό επιλέγουμε να μην εκτελεστεί η ευρετική διαδικασία αλλά να κληρονομηθεί η ευρετική λύση του γονέα. Αντίθετα ο αλγόριθμος αναβαθμίσεων θα δώσει σίγουρα διαφορετική λύση στον 'μη εφικτό' απόγονο και άρα η εκτέλεση του σε αυτόν επιβάλλεται.

5.5 Υπολογιστικά Αποτελέσματα

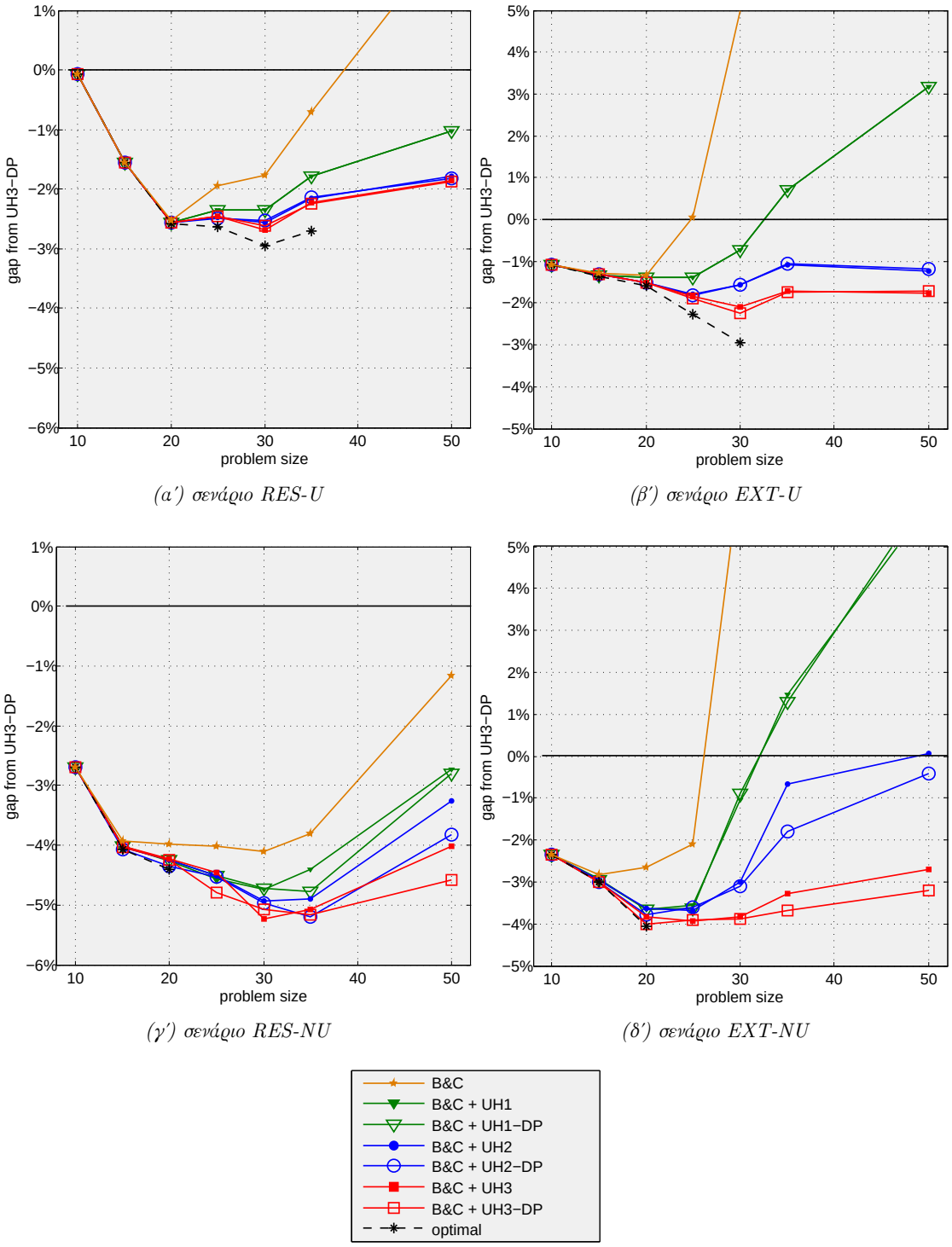
Για καθέναν από τους έξι αλγορίθμους αναβαθμίσεων (UH1, UH2, UH3, UH1-DP, UH2-DP, UH3-DP) υλοποιήθηκε η αντίστοιχη ευρετική επανάκληση. Αντίστοιχα ορίζουμε έξι διαφορετικές παραλλαγές τους αλγορίθμου B&C, μία για κάθε υλοποιημένη ευρετική επανάκληση οι οποίες συμβολίζονται στα αποτελέσματα ως 'B&C + UHx'. Εν συνεχεία εφαρμόσαμε τις έξι αυτές παραλλαγές καθώς και την απλή -χωρίς κάποια ευρετική επανάκληση- σε όλα τα στιγμιότυπα αναφοράς μέχρι και 50 τερματικών κόμβων θέτοντας σε κάθε εκτέλεση ως συνθήκη τερματισμού την εξέταση το πολύ 1000 κόμβων του δέντρου B&C. Στα γραφήματα του σχήματος 5.2 απεικονίζονται οι μέσες αποστάσεις των λύσεων κάθε παραλλαγής του B&C από τις λύσεις του αλγορίθμου UH3-DP [1]. Ως σημείο αναφοράς επιλέχθηκε ο UH3-DP καθότι έδωσε τις καλύτερες λύσεις μεταξύ των αλγορίθμων αναβαθμίσεων. Για λόγους σύγκρισης στα γραφήματα προστέθηκε και η καμπύλη των βέλτιστων τιμών για τα μεγέθη που αυτές έχουν υπολογιστεί (κεφ 3).

Μία πρώτη παρατήρηση είναι ότι σε πολύ μικρά προβλήματα -10, 15 και σε κάποια σενάρια και 20 τερματικών κόμβων- το όριο που έχει τεθεί αποδεικνύεται αρκετά 'χαλαρό' καθότι και οι επτά εξεταζόμενες παραλλαγές καταφέρνουν να προσεγγίσουν τις βέλτιστες τιμές. Καθώς όμως το μέγεθος τον προβλημάτων αυξάνεται παρατηρούνται όλο και μεγαλύτερες διαφοροποιήσεις, γεγονός που αποδίδεται στο ρόλο των διαφορετικών ευρετικών επανακλήσεων. Από κάποια μεγέθη και μετά η απλή εκδοχή του B&C αποτυγχάνει πλήρως να προσεγγίσει τις λύσεις του UH3-DP ενώ σε κάποια προβλήματα 50 κόμβων δε καταφέρνει καν να βρει κάποια εφικτή λύση.

Αναφορικά με τις παραλλαγές του B&C που χρησιμοποιούν ευρετικές επανακλήσεις, οι διαφοροποιήσεις μεταξύ των καμπύλων είναι σε κάποιο βαθμό ανάλογες με εκείνες μεταξύ των αλγορίθμων αναβαθμίσεων στο προηγούμενο κεφάλαιο. Κυριότερη παρατήρηση είναι ότι η παραλλαγή που αξιοποιεί τον UH3-DP δίνει κατά κανόνα τις καλύτερες λύσεις. Επίσης, επιβεβαιώνεται και πάλι η αδυναμία του UH1, αλλά και του UH1-DP, στα σενάρια που χρησιμοποιείται το επεκτεταμένο σύνολο τύπων ζεύξεων.

Η παραλλαγή που αξιοποιεί τον UH3-DP επιλέγεται ως επικρατέστερη για να εφαρμοστεί και στα μεγαλύτερα προβλήματα 70, 100 και 150 τερματικών κόμβων. Στις εκτελέσεις τέθηκε και πάλι ως συνθήκη τερματισμού η εξέταση 1000 B&C κόμβων. Στον πίνακα 5.1 παρατίθενται οι βελτιώσεις στις εξαγόμενες λύσεις σε σχέση με τον αλγόριθμο UH3-DP ενώ το σχήμα 5.3 απεικονίζει τους μέσους χρόνους εκτέλεσης για κάθε σενάριο [2]. Η αύξουσα πορεία των καμπυλών χρόνων εκτέλεσης είναι αναμενόμενη δεδομένου ότι καθώς αυξάνεται το μέγεθος των προβλημάτων αυξάνεται και ο

[1] Οι τιμές των καμπυλών του σχήματος 5.2 παρατίθενται στον πίνακα Α΄.6 του Παραρτήματος.
[2] Οι τιμές των καμπυλών του σχήματος 5.3 παρατίθενται στον πίνακα Α΄.7 του Παραρτήματος.



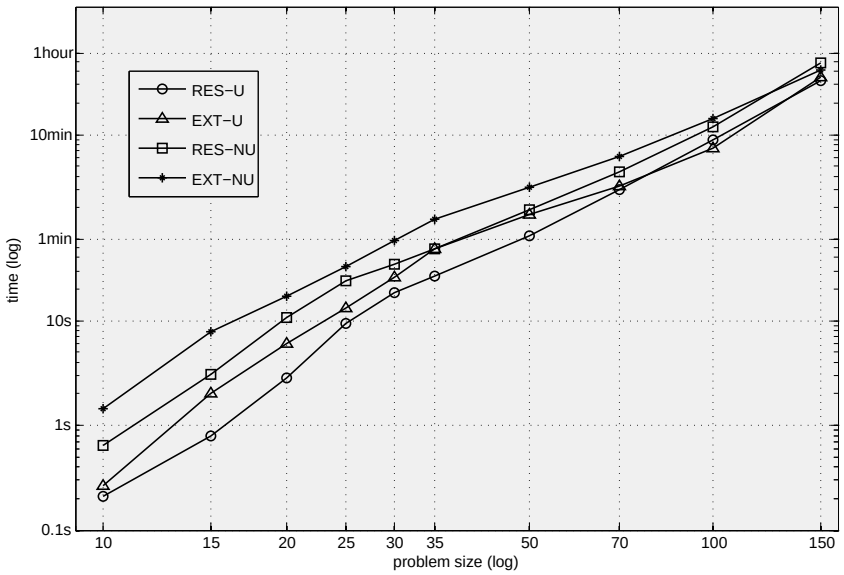
(α′) σενάριο RES-U *(β′) σενάριο EXT-U*

(γ′) σενάριο RES-NU *(δ′) σενάριο EXT-NU*

Σχήμα 5.2: Αποστάσεις λύσεων παραλλαγών του αλγόριθμου Διακλάδωσης και Αποκοπής με ευρετικές επαναλήσεις από τις λύσεις του UH3-DP

size	RES-U	EXT-U	RES-NU	EXT-NU
10	-0.08 %	-1.10 %	-2.69 %	-2.34 %
15	-1.56 %	-1.31 %	-4.03 %	-3.00 %
20	-2.56 %	-1.51 %	-4.25 %	-4.01 %
25	-2.45 %	-1.88 %	-4.80 %	-3.90 %
30	-2.62 %	-2.25 %	-5.07 %	-3.88 %
35	-2.24 %	-1.73 %	-5.17 %	-3.67 %
50	-1.86 %	-1.72 %	-4.57 %	-3.21 %
70	-1.77 %	-1.46 %	-3.09 %	-2.41 %
100	-0.97 %	-1.46 %	-2.64 %	-2.51 %
150	-0.67 %	-1.19 %	-1.82 %	-2.06 %

Πίνακας 5.1: Αποστάσεις λύσεων αλγορίθμου B&C με εφαρμογή της ευρετικής επανάκλησης UH3-DP από τις λύσεις του αλγορίθμου αναβαθμίσεων UH3-DP



Σχήμα 5.3: Χρόνοι εκτέλεσης παραλλαγών αλγόριθμου Διακλάδωσης και Αποκοπής

χρόνος επίλυσης των γραμμικών χαλαρώσεων των εξεταζόμενων B&C κόμβων.

Όσον αφορά στις εξαγόμενες λύσεις, παρατηρούμε βελτιώσεις που υπερβαίνουν και το 5% σε κάποιες περιπτώσεις, ωστόσο βαίνουν σταθερά μειούμενες στα μεγαλύτερα προβλήματα. Η επιδείνωση αυτή αποδίδεται στη συνθήκη τερματισμού που τέθηκε. Το όριο εξέτασης 1000 B&C κόμβων είναι υπερεπαρκές για τα μικρότερα προβλήματα όπου επιτυγχάνονται σχεδόν βέλτιστες λύσεις. Ωστόσο, καθώς αυξάνεται το μέγεθος των προβλημάτων οι 1000 B&C κόμβοι αντιπροσωπεύουν όλο και μικρότερο ποσοστό του συνολικού αριθμού κόμβων που θα χρειαζόταν να επισκεφτούμε για να επιλύσουμε το πρόβλημα. Υποθέτουμε ότι αυξάνοντας το όριο αυτό στα μεγαλύτερα προβλήματα θα επιτυγχάναμε καλύτερες λύσεις, όμως η αλλαγή αυτή θα είχε αναπόφευκτα αντίκτυπο στους χρόνους εκτέλεσης.

Κεφάλαιο 6

Εξελικτικοί Αλγόριθμοι

Οι μεταευρετικές διαδικασίες (metaheuristics) αποτελούν ένα σύνολο γενικών στρατηγικών που μπορούν να εφαρμοστούν σε ένα ευρύ σύνολο διαφορετικών προβλημάτων με σχετικά λίγες ή καθόλου τροποποιήσεις. Υποκατηγορία των μεταευρετικών διαδικασιών αποτελούν οι εξελικτικοί αλγόριθμοι (evolutionary algorithms, EA). Οι εξελικτικοί αλγόριθμοι χρησιμοποιούν μηχανισμούς εμπνευσμένους από τη βιολογική εξέλιξη: αναπαραγωγή, μετάλλαξη, διασταύρωση, και φυσική επιλογή. Οι υποψήφιες λύσεις ενός προβλήματος παίζουν το ρόλο ατόμων σε ένα πληθυσμό, και οι συνάρτηση κόστους καθορίζει το περιβάλλον μέσα στο οποίο 'ζουν' οι λύσεις. Η εξέλιξη του πληθυσμού πραγματοποιείται μέσω της επαναλαμβανόμενης εφαρμογής των παραπάνω τελεστών. Οι λύσεις κατά την εκτέλεση των αλγορίθμων αναπαριστώνται με διανύσματα. Η αντιστοίχηση μεταξύ διανυσμάτων και λύσεων συνήθως αναφέρεται ως αναπαράσταση της λύσης (solution representation) ή κωδικοποίηση λύσης (solution encoding). Η διαδικασία εξέλιξης ενός πληθυσμού κατά μία γενιά προϋποθέτει αρχικά τη μετατροπή των διανυσμάτων του πληθυσμού σε λύσεις του προβλήματος μέσω μίας συνάρτησης αποκωδικοποίησης (decoding function) $f : X \to Y$, όπου X ο χώρος των διανυσμάτων και Y ο χώρος των λύσεων.

Εξελικτικοί αλγόριθμοι έχουν εφαρμοστεί εκτενώς σε προβλήματα ελάχιστων δέντρων επικάλυψης (spanning tree problems) . Ενδεικτικά παραθέτουμε τις εργασίες των Palmer and Kershenbaum (1994); Palmer et al. (1995); Gaube and Rothlauf (2001); Li and Bouchebaba (2000); Li (2001); Rothlauf et al. (2002, 2003); Soak et al. (2006); Soak (2006); Hoang et al. (2010) για το πρόβλημα Optimum Communication Spanning Tree (OCSTP). Επίσης, υπάρχουν αρκετές εργασίες στα προβλήματα Degree Constrained Minimum Spanning Tree (DCMST) (Zhou and Gen, 1997; Knowles and Corne, 2000; Raidl, 2000; Krishnamoorthy et al., 2001; Soak et al., 2004, 2006) και Quadratic Minimum Spanning Tree (QMST) (Zhou and Gen, 1997, 1998; Soak et al., 2006).

6.1 Αναπαραστάσεις δέντρων

Σημαντική παράμετρος για την επίδοση των αλγορίθμων αποτελεί η επιλογή της συνάρτησης αποκωδικοποίησης που θα χρησιμοποιηθεί, η οποία συνήθως στα προβλήματα βελτιστοποίησης επικαλύπτοντος δέντρου αναφέρεται ως αναπαράσταση δέντρου (tree representation) ή κωδικοποίηση δέντρου (tree encoding). Για την αξιολόγηση της αποτελεσματικότητας διαφορετικών αναπαραστάσεων δέντρων πρέπει να λάβουμε υπόψη σε πιο βαθμό αυτές πληρούν κάποια χαρακτηριστικά. Βασικότερα αυτών των χαρακτηριστικών είναι σύμφωνα με τους Palmer and Kershenbaum (1994); Lin and Gen (2006); Carrano et al. (2007) τα εξής:

- *Κάλυψη (Coverage)*: Είναι απαραίτητη η πλήρης κάλυψη του συνόλου των δυνατών δέντρων, δηλαδή η μη ύπαρξη δέντρου το οποίο να μη μπορεί να προκύψει από κάποιο κωδικοποιημένο διάνυσμα μέσω της f. Αυτό συμβαίνει όταν η f είναι 'επί' δηλαδή για κάθε $y \in Y$ υπάρχει $x \in X$ τέτοιο ώστε $f(x) = y$.
- *Αμεροληψία (Unbiasedness)*: Είναι καλό η κωδικοποίηση να αναπαριστά όλα τα πιθανά δέντρα με το ίδιο αριθμό κωδικοποιημένων διανυσμάτων. Στην ειδική περίπτωση που κάθε δέντρο αντιστοιχίζεται σε ένα ακριβώς κωδικοποιημένο διανυσμάτων τότε η κωδικοποίηση χαρακτηρίζεται από *μοναδικότητα (uniqueness)* και η συνάρτηση f είναι ''ένα προς ένα''.
- *Εφικτότητα (Feasibility)*: Όλα τα πιθανά διανύσματα θα πρέπει να αντιστοιχίζονται σε λύσεις. Στις περιπτώσεις που η εφικτότητα δεν τηρείται εφαρμόζονται μηχανισμοί διόρθωσης των διανυσμάτων ώστε να γίνει εφικτή η αποκωδικοποίηση. Συνήθως η εφικτότητα εκφράζεται με το ποσοστό των διανυσμάτων που αντιστοιχίζεται σε εφικτές λύσεις.
- *Κληρονομικότητα (Heritability)*: Εκφράζει το βαθμό όπου το δέντρο που προκύπτει από την εφαρμογή ενός γενετικού τελεστή διασταύρωσης έχει κοινά στοιχεία με του γονείς του.
- *Τοπικότητα (Locality)*: Με υψηλή τοπικότητα χαρακτηρίζονται αναπαραστάσεις όπου η εφαρμογή ενός γενετικού τελεστή μετάλλαξης, ή μία μικρή μετατόπιση του διανύσματος θέσης ενός σωματιδίου, επιφέρει μικρές αλλαγές σε ένα δέντρο.
- *Χρόνος*: Οι διαδικασίες της αποκωδικοποίησης, της εφαρμογής γενετικών τελεστών και μετατόπισης ενός σωματιδίου θα πρέπει να μην κοστίζουν πολύ σε υπολογιστικό χρόνο. Ο υπολογιστικό χρόνος επηρεάζεται σε μεγάλο βαθμό από το μήκος των κωδικοποιημένων διανυσμάτων.

Οι Palmer and Kershenbaum (1994) σημειώνουν ότι: ''Ιδεατά μία αναπαράσταση δέντρων θα έπρεπε να συγκεντρώνει όλα τα παραπάνω χαρακτηριστικά. Δυστυχώς, οι περισσότερες αναπαραστάσεις ανταλλάσσουν μερικά από τα αυτά τα επιθυμητά γνωρίσματα με άλλα.'' Στις επόμενες ενότητες θα αναφερθούμε συνοπτικά σε μερικές δημοφιλείς αναπαραστάσεις δέντρων. Για την ανάλυση μας θεωρούμε ότι ο γράφος G για τον οποίο αναζητούμε επικαλύπτον δέντρο είναι πλήρης και με σύνολο

κόμβων $\{1, \ldots, n\}$.

6.1.1 Χαρακτηριστικό διάνυσμα

Το χαρακτηριστικό διάνυσμα (Davis et al., 1993; Piggott and Suraweera, 1995) περιλαμβάνει δυαδικές μεταβλητές όπου καθεμία αντιπροσωπεύει την παρουσία, ή απουσία, μία ακμής του G από το δέντρο. Το μήκος ενός τέτοιου διανύσματος είναι $n(n-1)/2$, όσες δηλαδή και οι ακμές του γράφου G. Είναι προφανές ότι οποιοδήποτε δέντρο μπορεί να περιγραφεί (μοναδικά) εάν θέσουμε 1 στις μεταβλητές του διανύσματος που αντιστοιχούν στις ακμές του δέντρου. Συνεπώς η κωδικοποίηση αυτή καλύπτει το σύνολο των δέντρων, είναι αμερόληπτη (1-1). Επίσης, η κωδικοποίηση αυτή χαρακτηρίζεται από κληρονομικότητα και υψηλή τοπικότητα.

Εν τούτοις, ένα διάνυσμα μπορεί να περιγράψει, εκτός από δέντρα, οποιοδήποτε υπογράφο του G και άρα δεν πληρείται η εφικτότητα. Μάλιστα, το πλήθος των δυνατών υπογράφων που μπορούν να περιγραφούν είναι $2^{n(n-1)/2}$. Η πιθανότητα ένα τυχαίο διάνυσμα να αντιστοιχίζεται σε δέντρο γίνεται απείρως μικρή καθώς το n αυξάνεται. Στην πραγματικότητα είναι της τάξης $2^{-n(n/2-\log_2 n)}$ (Palmer and Kershenbaum, 1994). Ακόμα, το μεγάλο μήκος του διανύσματος κοστίζει ακριβά σε υπολογιστικό χρόνο. Οποιαδήποτε διαδικασία που προϋποθέτει το ``διάβασμα'' του διανύσματος έχει κατ' ελάχιστο πολυπλοκότητα $\mathcal{O}(n^2)$.

6.1.2 Κωδικοποίηση προκατόχων (Predecessor encoding)

Μία πιο συμπαγής αναπαράσταση δέντρων είναι η κωδικοποίηση προκατόχων (Chou et al., 2001; Raidl and Drexel, 2000; Chu et al., 1999), στην οποία ένας κόμβος έχει οριστεί ως ρίζα του δέντρου και το διάνυσμα περιέχει μία μεταβλητή για κάθε άλλο κόμβο (πλην της ρίζας). Η κάθε μεταβλητή προσδιορίζει τον προκάτοχο ενός κόμβου στο μονοπάτι από αυτόν προς τη ρίζα του δέντρου. Ένα κωδικοποιημένο διάνυσμα έχει μήκος $(n-1)$ και άρα πολυπλοκότητα αποκωδικοποίησης $\mathcal{O}(n)$, κάτι που είναι μεγάλη βελτίωση σε σχέση με την κωδικοποίηση χαρακτηριστικού διανύσματος. Επίσης η κωδικοποίηση προκατόχων διατηρεί τα πλεονεκτήματα το χαρακτηριστικού διαστήματος καθώς καλύπτει το σύνολο των δέντρων, είναι αμερόληπτη (1-1) και χαρακτηρίζεται από υψηλή κληρονομικότητα και τοπικότητα.

Σημαντικό μειονέκτημα και αυτής της κωδικοποίησης είναι ότι δεν περιορίζεται μόνο σε δέντρα. Βελτίωση σε σχέση με την κωδικοποίηση χαρακτηριστικού διανύσματος είναι το γεγονός ότι η πιθανότητα ένα τυχαίο διάνυσμα να αντιστοιχίζεται σε δέντρο είναι $1/n$.

Κωδικοποίηση	Μήκος	Πεδίο τιμών	Εφαρμογή τελεστή	Αποκωδικοποίηση	Πλήρης Κάλυψη	Εφικτότητα (πιθανότητα)	Προκατάληψη (bias)	Κληρονομικότητα / Τοπικότητα
Χαρακτηριστικό διάνυσμα	$n(n-1)/2$	$\{0,1\}$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$	ναι	χείριστη: $2^{-n(n/2-\log_2 n)}$	όχι	υψηλή
Κωδικοποίηση προκατόχων	$n-1$	$\{1,\ldots,n\}$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	ναι	κακή: $1/n$	όχι	υψηλή
Prüfer	$n-2$	$\{1,\ldots,n\}$	$\mathcal{O}(n)$	$\mathcal{O}(n\log n)$	ναι	ναι	όχι	χαμηλη
Netkeys	$n(n-1)/2$	$[0,1]$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2\log n)$	ναι	ναι	χαμηλη	υψηλή
Edge sets με ειδικούς τελεστές	$2(n-1)$	$\{1,\ldots,n\}$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	ναι	ναι	χαμηλή	υψηλή
EW (CB-TCR) με ειδικούς τελεστές	$2(n-1)$	$\{1,\ldots,n\}$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	ναι	ναι	ναι: ακμές μικρότερης απόστασης	υψηλή

Πίνακας 6.1: Δημοφιλείς αναπαραστάσεις δέντρων

6.1.3 Ακολουθία Prüfer

Ο Cayley (1889) απέδειξε την ακόλουθη σχέση: ο αριθμός δέντρων επικάλυψης ενός πλήρη γράφου με n κόμβους είναι $n^{(n-2)}$. Ο Prüfer (1918) παρουσίασε μία απλή απόδειξη της σχέσης, αντιστοιχίζοντας ένα προς ένα το σύνολο των πιθανών δέντρων με μία ακολουθία $n-2$ ακεραίων με τιμές από 1 έως n. Η ακολουθία αυτή είναι γνωστή ως ακολουθία Prüfer, αριθμοί Prüfer, ή κωδικοποίηση Prüfer.

Το γεγονός ότι η κωδικοποίηση αυτή είναι αμερόληπτη, δίνει μόνο δέντρα και καλύπτει εξ ολοκλήρου το σύνολο των δέντρων, έχει οδηγήσει πολλούς ερευνητές στη χρησιμοποίηση της σε πληθώρα προβλημάτων. Επίσης η αποκωδικοποίηση μίας ακολουθίας, παρόλο που απαιτεί πιο σύνθετες ενέργειες από αυτές των δύο προηγούμενων κωδικοποιήσεων, απαιτεί χρόνο $\mathcal{O}(n \log n)$. Το μεγάλο μειονέκτημα αυτής της κωδικοποίησης είναι ότι έχει κακή τοπικότητα και κληρονομικότητα (Palmer and Kershenbaum, 1994; Rothlauf and Goldberg, 2000; Gottlieb et al., 2001) καθώς μία μικρή αλλαγή σε μία ακολουθία μπορεί να οδηγήσει σε ένα δέντρο που θα διαφέρει κατά πολύ από το αρχικό.

6.1.4 Network random keys (Netkeys)

Οι Rothlauf et al. (2000, 2002, 2003) παρουσίασαν μία κωδικοποίηση βασισμένη σε τυχαία κλειδιά δίνοντάς της το όνομα Network random keys ή Netkeys. Σύμφωνα με αυτήν, ένα κωδικοποιημένο διάνυσμα περιλαμβάνει μία πραγματική μεταβλητή ανά ακμή του γράφου G, η οποία αντιπροσωπεύει ένα βάρος για την ακμή. Κατά την αποκωδικοποίηση εφαρμόζεται ο αλγόριθμος του Kruskal (1956) για τα βάρη αυτά και προκύπτει δέντρο ελαχίστου βάρους. Η αναπαράσταση δίνει πάντα δέντρα και μπορεί να προκύψει οποιοδήποτε δέντρο εάν τεθούν οι κατάλληλες τιμές. Εντούτοις, χαρακτηρίζεται από υψηλό πλεονασμό (redundancy), καθώς ο αριθμός των διαφορετικών διανυσμάτων που αναπαριστούν ένα δέντρο είναι μεγάλος (θεωρητικά άπειρος). Το γεγονός αυτό επιβαρύνει την κληρονομικότητα και την τοπικότητα. Η πολυπλοκότητα εφαρμογής ενός γενετικού τελεστή (ή μετατόπιση σωματιδίου) είναι κατ' ελάχιστο $\mathcal{O}(n^2)$ αφού το μήκος ενός διανύσματος είναι $n(n-1)/2$. Η αποκωδικοποίηση κοστίζει ακόμα περισσότερο, $\mathcal{O}(n^2 \log n)$, δεδομένου ότι για την εφαρμογή του αλγορίθμου του Kruskal απαιτείται η ταξινόμηση του διανύσματος.

6.1.5 Σύνολα Ακμών (Edge Sets)

Η κωδικοποίηση συνόλων ακμών προτάθηκε από τους Raidl (2000); Raidl and Julstrom (2003) βασίζεται στην απευθείας περιγραφή των ακμών που αποτελούν ένα δέντρο. Σε αντίθεση με το χαρακτηριστικό διάνυσμα, όπου για κάθε ακμή του γράφου μία δυαδική μεταβλητή δηλώνει αν περιλαμβάνεται στο δέντρο ή όχι, στα σύνολα ακμών έχουμε $n-1$ ζευγάρια μεταβλητών (όσες δηλαδή

και οι ακμές σε ένα δέντρο), όπου κάθε ζευγάρι δηλώνει τους δύο κόμβους μίας ακμής του δέντρου. Όπως και η κωδικοποίηση χαρακτηριστικού διανύσματος, τα σύνολα ακμών καλύπτει το σύνολο των δέντρων, είναι αμερόληπτη, και χαρακτηρίζεται από υψηλή κληρονομικότητα και τοπικότητα. Η διαφορά στο μήκος ενός διανύσματος συνόλου ακμών σε σχέση με ένα χαρακτηριστικό διάνυσμα δίνει πολύ χαμηλότερους χρόνους αποκωδικοποίησης: $\mathcal{O}(n)$ έναντι $\mathcal{O}(n^2)$.

Η δημιουργία ενός διανύσματος μήκους $2(n-1)$ ακεραίων με τυχαίες τιμές 1 έως n είναι εμφανές ότι δε θα έδινε δέντρα, παρά μόνο σε πολύ λίγες περιπτώσεις. Για το λόγο αυτό οι Raidl (2000); Raidl and Julstrom (2003) παρουσίασαν ειδικούς τελεστές αρχικοποίησης, διασταύρωσης και μετάλλαξης, οι οποίοι εγγυώνται ότι τα παραγόμενα διανύσματα θα αποκωδικοποιούνται πάντα σε δέντρα.

6.1.6 Παράθυρο Ακμών (Edge Window)

Ο Soak (2006) πρότεινε κωδικοποίηση με $2(n-1)$ ακέραιες μεταβλητές, όπου κάθε μεταβλητή αντιπροσωπεύει ένα κόμβο (τιμές από 1 έως n). Κάθε κόμβος πρέπει να εμφανίζεται τουλάχιστον μία φορά στο διάνυσμα. Από κάθε ζευγάρι γειτονικών κόμβων στο διάνυσμα εξάγουμε μία υποψήφια ακμή, άρα έχουμε συνολικά $2(n-1)$ υποψήφιες ακμές. Προφανώς η εισαγωγή όλων των υποψήφιων ακμών στη λύση δε δίνουν δέντρο οπότε απαιτείται η εφαρμογή μίας Ρουτίνας Κατασκευής Δέντρου (Tree Construction Routine, TCR) η οποία τελικά θα αξιοποιεί κάποιες από αυτές. Η ρουτίνα προτάθηκε ονομάστηκε Cycle Breaking TCR (CB-TCR). Το δέντρο κατασκευάζεται με τη διαδοχική εισαγωγή των υποψήφιων ακμών στη λύση, τηρώντας τη σειρά τους στο διάνυσμα. Όταν δημιουργηθεί κύκλος τότε τον 'σπάμε' απομακρύνοντας την ακμή του κύκλου με τη μεγαλύτερη απόσταση.

Η κωδικοποίηση Παράθυρου Ακμών δίνει πάντα δέντρα (εφικτότητα), καλύπτει πλήρως το σύνολο των δέντρων και χαρακτηρίζεται από υψηλή τοπικότητα και κληρονομικότητα. Από την άλλη δεν είναι αμερόληπτη καθότι ευνοεί ζεύξεις μικρότερης απόστασης κατά τη διαδικασία σπασίματος κύκλων.

6.2 Εξελικτικοί αλγόριθμοι για το πρόβλημα MLCMST

Από τις αναπαραστάσεις που παρουσιάστηκαν στην προηγούμενη ενότητα επιλέχτηκαν για να εξεταστούν στο πρόβλημα MLCMST οι Netkeys και τα Σύνολα Ακμών καθώς έχουν δώσει καλά αποτελέσματα σε αρκετά προβλήματα. Κάθε κωδικοποίηση συνοδεύεται από αντίστοιχους τελεστές αρχικοποίησης, διασταύρωσης και μετάλλαξης. Όσον αφορά ειδικά στην αρχικοποίηση, επιλέγουμε να εκκινήσουμε από τυχαίες λύσεις με εξαίρεση το πρώτο διάνυσμα του πληθυσμού στο οποίο εισάγουμε τη λύση που δίνει ο αλγόριθμος αναβαθμίσεων UH3-DP. Συνδυάζοντας τις ανωτέρω κωδικοποιήσεις με δύο διαφορετικές εξελικτικές διαδικασίες, καταλήγουμε σε τέσσερεις αλγορίθμους: NK1, NK2,

ES1, ES2.

Εδώ θα πρέπει να σημειώσουμε ότι η αποκωδικοποίηση ενός διανύσματος -οποιασδήποτε αναπαράστασης- μας δίνει ένα επικαλύπτον δέντρο, όμως δε μας παρέχει κάποια πληροφορία για τους εγκατεστημένους τύπους ζεύξεων. Ωστόσο, αφού υπολογίσουμε τις ροές κίνησης στο δέντρο είναι προφανές ότι η βέλτιστη στρατηγική είναι αυτή της επιλογής του χαμηλότερου τύπου από εκείνους που ικανοποιούν τον περιορισμό χωρητικότητας σε κάθε ακμή. Εάν η ροή σε μία ή περισσότερες ακμές ενός δέντρου υπερβαίνει την τιμή Z^L η λύση είναι μη εφικτή. Χειριζόμαστε τις λύσεις αυτές θέτοντας πολύ υψηλή τιμή κόστους (penalty) στις προβληματικές ακμές.

6.2.1 Πρώτη εξελικτική διαδικασία

Η πρώτη εξελικτική διαδικασία (αλγ. 6.1) προτάθηκε από τον Raidl (2000). Ο νέος πληθυσμός παράγεται ως εξής: Αρχικά επιλέγονται κόμβοι από τον προηγούμενο πληθυσμό μέσω δυαδικού τουρνουά (binary tournament). Από τους επιλεγμένους κόμβους προκύπτουν νέα διανύσματα με εφαρμογή τελεστών διασταύρωσης (με πιθανότητα p_c) και μετάλλαξης (με πιθανότητα p_m). Από τα νέα διανύσματα αφαιρούνται τα διπλότυπα (duplicates). Ο νέος πληθυσμός προκύπτει αντικαθιστώντας τα χειρότερα διανύσματα του παλιού πληθυσμού με τα νέα διανύσματα.

6.2.2 Δεύτερη εξελικτική διαδικασία

Η δεύτερη εξελικτική διαδικασία (αλγ. 6.2) προτάθηκε από τον Soak (2006). Ο διαδικασία παράγει τη νέα γενιά του πληθυσμού ($P(t)$) από την προηγούμενη ($P(t-1)$) με τα εξής βήματα: Καταρχήν, εφαρμόζεται τελεστής επιλογής στον πληθυσμό $P(t-1)$, από τον οποίο προκύπτει μία αρχική έκδοση του $P(t)$. Ακολούθως, τα μέλη του $P(t)$ χωρίζονται σε ζευγάρια και με πιθανότητα p_c ένα ζευγάρι παράγει έναν απόγονο (τελεστής διασταύρωσης). Κάθε απόγονος αντικαθιστά στον $P(t)$ το χειρότερο από τους δύο γονείς του. Τέλος, με πιθανότητα p_m εφαρμόζεται τελεστής μετάλλαξης στα στοιχεία του $P(t)$.

Ως διαδικασία επιλογής χρησιμοποιήθηκε ο τελεστής Real World Tournament Selection (RWTS) που πρότεινε ο Soak (2006). Ο τελεστής RWTS δίνει έναν ενδιάμεσο πληθυσμό I, ο οποίος απαρτίζεται από αντίγραφα ορισμένων ατόμων του τρέχοντος πληθυσμού P, με τον εξής τρόπο: Καταρχήν όλα τα διανύσματα του P χωρίζονται σε ζευγάρια και αντίγραφο του καλύτερου διανύσματος κάθε ζεύγους, δηλαδή εκείνου με το μικρότερο κόστος, τοποθετείται στον πληθυσμό I. Τα διανύσματα αυτά ονομάζονται 'νικητές επιπέδου 1'. Επαναλαμβάνοντας την παραπάνω διαδικασία για τους 'νικητές επιπέδου 1' παράγονται 'νικητές επιπέδου 2', οι οποίοι επίσης εισάγονται στον I. Η διαδικασία επαναλαμβάνεται κάθε φορά παράγοντας 'νικητές επιπέδου x' από τους νικητές 'νικητές επιπέδου

Αλγόριθμος 6.1: Βήματα πρώτου εξελικτικού αλγορίθμου

```
1  t ← 0
2  αρχικοποίηση του πληθυσμού P(t)
3  αποτίμηση του κόστους κάθε διανύσματος του P(t)
4  repeat
5  |  t ← t + 1
6  |  S ← επιλογή κόμβων μέσω δυαδικού τουρνουά από τον P(t − 1)
7  |  O ← δημιουργία απογόνων από ποσοστό p_c του S
8  |  M ← δημιουργία μεταλλαγμένων διανυσμάτων από ποσοστό p_m του S
9  |  N ← O ∪ M
10 |  αφαίρεση διπλοτύπων (duplicates) από το N
11 |  δημιουργία του P(t) αντικαθιστώντας τα χειρότερα διανύσματα του P(t − 1) με τα
   |  διανύσματα του N
12 |  αποτίμηση του κόστους των νέων διανυσμάτων του P(t)
13 until κριτήριο τερματισμού
```

Αλγόριθμος 6.2: Βήματα δεύτερου εξελικτικού αλγορίθμου

```
1  t ← 0
2  αρχικοποίηση του πληθυσμού P(t)
3  αποτίμηση του κόστους κάθε διανύσματος του P(t)
4  repeat
5  |  t ← t + 1
6  |  δημιουργία του P(t) με επιλογή στοιχείων από τον P(t − 1) μέσω RWTS
7  |  χωρισμός των στοιχείων του P(t) σε ζευγάρια
8  |  παραγωγή απογόνων από ποσοστό p_c των ζευγαριών
9  |  κάθε απόγονος αντικαθιστά το χειρότερο γονέα του στον P(t)
10 |  σε ποσοστό p_m των ατόμων του P(t) εφαρμόζεται μετάλλαξη
11 |  αποτίμηση του κόστους κάθε διανύσματος του P(t)
12 until κριτήριο τερματισμού
```

$x + 1'$, μέχρι να απομείνει μόνο ένας νικητής. Ο πληθυσμός I συμπληρώνεται με επιπλέον αντίγραφα του τελικού νικητή ώστε να έχει το ίδιο μέγεθος με τον P.

6.2.3 Τελεστές για την κωδικοποίηση Netkeys

Οποιοδήποτε αποκωδικοποίηση διανύσματος από την Netkeys δίνει δέντρο. Για το λόγο αυτό, η Netkeys συνήθως συνδυάζεται με γενικούς (generic) τελεστές και όχι συγκεκριμένους για το πρόβλημα (problem specific). Για τη διαδικασία διασταύρωσης χρησιμοποιήθηκε ο τελεστής ομοιόμορφης διασταύρωσης (uniform crossover), ενώ για τη διαδικασία μετάλλαξης χρησιμοποιήθηκε ο τελεστής μετάλλαξης ανταλλαγής (swap mutation). Κατά την εκκίνηση, όλα τα διανύσματα αρχικοποιήθηκαν με τυχαίες τιμές από το εύρος $[0, 1]$, πλην του πρώτου διανύσματος στο οποίο εισάγαμε τη λύση που δίνει ο αλγόριθμος αναβαθμίσεων UH3-DP.

Στην Netkeys δεν ορίζεται η αντίστροφη της συνάρτησης αποκωδικοποίησης, καθώς ένα δέντρο μπορεί να παραχθεί από διαφορετικά διανύσματα λύσης. Συνεπώς, επιλέγουμε εμείς τις αρχικές τιμές του πρώτου διανύσματος έτσι ώστε να εξασφαλίζεται ότι θα δώσουν το ζητούμενο δέντρο. Έστω x_i το στοιχείο του διανύσματος που αντιστοιχίζεται στην iοστή ακμή του γράφου G, τότε θέτουμε $x_i = 0.8r_i + 0.2y_i$ όπου r_i τυχαία τιμή στο διάστημα $[0, 1]$ και x_i δυαδική μεταβλητή με $d_i = 1$ αν και μόνο αν η iοστή ακμή του G περιλαμβάνεται στο ζητούμενο δέντρο.

6.2.4 Τελεστές για την κωδικοποίηση Συνόλων Ακμών

Τα σύνολα ακμών απαιτούν ειδικούς τελεστές τυχαιοποιημένης αρχικοποίησης, διασταύρωσης και μετάλλαξης έτσι ώστε να δίνουν εφικτές λύσεις (Raidl and Julstrom, 2003). Το πρώτο διάνυσμα αρχικοποιείται με το σύνολο ακμών της λύσης που δίνει ο UH3-DP. Για τα υπόλοιπα διανύσματα η αρχικοποίηση γίνεται με τη διαδικασία *KruskalRST*. Η εν λόγω διαδικασία εκκινεί από κενό σύνολο ακμών. Εν συνεχεία επιλέγονται τυχαία ακμές και εισάγονται στη λύση εφόσον δε δημιουργούν κύκλο. Η διαδικασία τερματίζεται όταν έχει παραχθεί επικαλύπτον δέντρο του γράφου.

Για την παραγωγή απογόνων χρησιμοποιείται η εξής διαδικασία: Έστω E_1 και E_2 τα σύνολα ακμών των δύο δέντρων-γονέων. Αρχικοποιούμε το σύνολο ακμών E' του απογόνου εισάγοντας τις ακμές $E_1 \cap E_2$. Εν συνεχεία, επιλέγουμε διαδοχικά τυχαίες ακμές από το σύνολο $E' \setminus (E_1 \cap E_2)$ και τις εισάγουμε στο E' αν δε προκύπτει κύκλος. Η διαδικασία τερματίζεται όταν έχει παραχθεί επικαλύπτον δέντρο του γράφου.

Η μετάλλαξη πραγματοποιείται με αντικατάσταση μίας ακμής της τρέχουσας με μία άλλη: Αρχικά επιλέγεται τυχαία μία ακμή (από αυτές που δεν περιλαμβάνονται στην τρέχουσα λύση) και εισάγεται

στη λύση. Η εισαγωγή επιπλέον ακμής δημιουργεί κύκλο στο γράφο. Η λύση γίνεται πάλι δέντρο απομακρύνοντας μία από τις ακμές του κύκλου.

6.3 Υπολογιστικά Αποτελέσματα

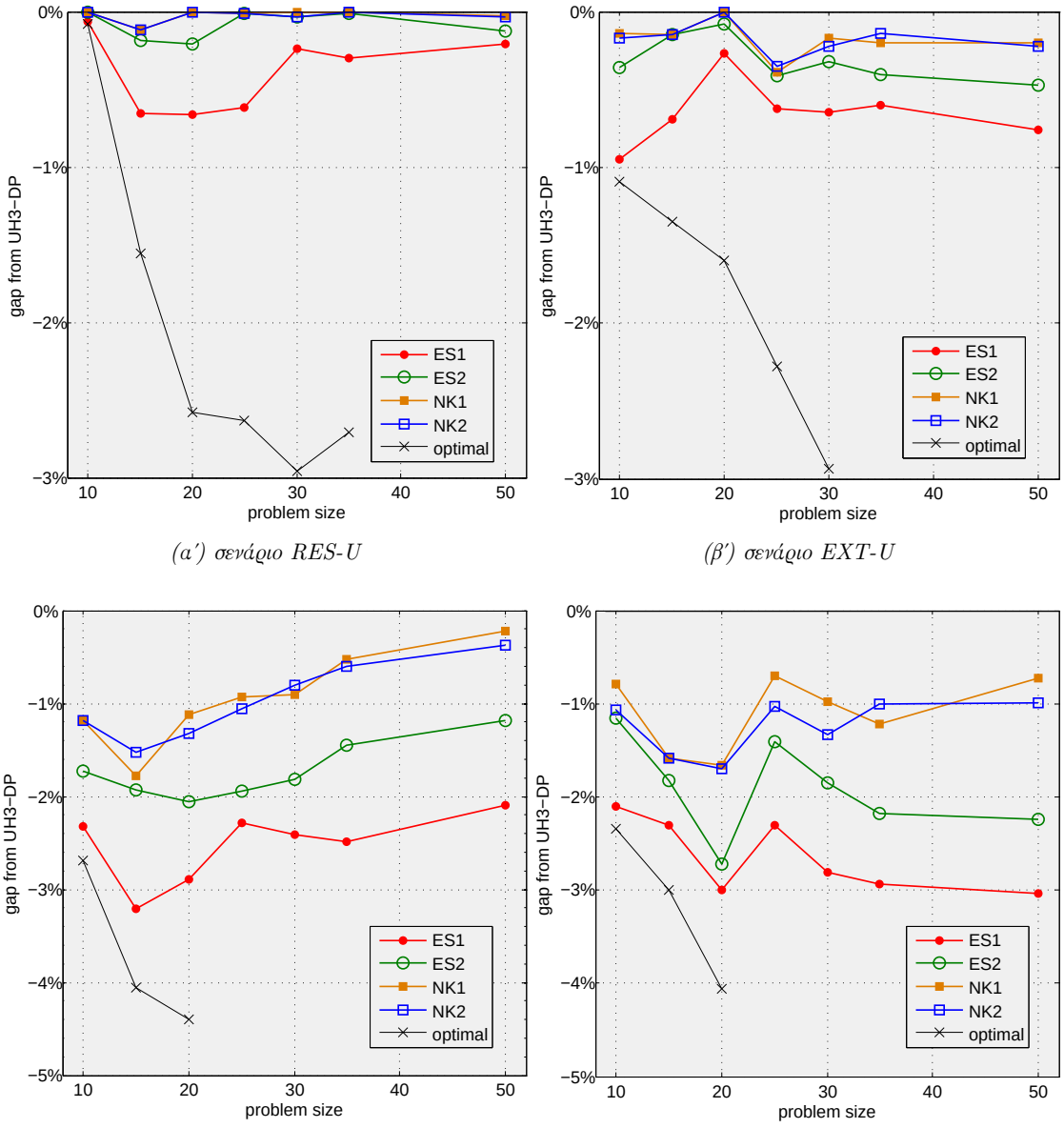
Για την υλοποίηση των εξελικτικών αλγορίθμων βασιστήκαμε στη βιβλιοθήκη γενετικών και εξελικτικών αλγορίθμων JGAP για Java. Και στους τέσσερεις εξεταζόμενους αλγορίθμους οι τιμές που τέθηκαν στους συντελεστές μετάλλαξης και διασταύρωσης ήταν $p_c = p_m = 0.6$. Τέλος όσον αφορά στο μέγεθος του πληθυσμού και το πλήθος των γενεών επιλέχθηκε κλιμάκωση ανάλογη με το μέγεθος των στιγμιότυπων. Για ένα πρόβλημα N τερματικών κόμβων κατασκευάζεται πληθυσμός $15N$ διανυσμάτων και η εκτέλεση τερματίζεται μετά από $10N$ γενεές.

Αρχικά κάθε αλγόριθμος εφαρμόστηκε σε όλα τα στιγμιότυπα αναφοράς έως και 50 τερματικών κόμβων. Δεδομένου ότι οι αρχικοί πληθυσμοί τροφοδοτούνται με τις λύσεις του ευρετικού UH3-DP αναμένουμε σε κάθε περίπτωση μόνο βελτιώσεις σε σχέση με αυτές. Τα γραφήματα του σχήματος 6.1 δίνουν τη δυνατότητα σύγκρισης των εξαγόμενων λύσεων απεικονίζοντας τις μέσες αποστάσεις των λύσεων των εξεταζόμενων αλγορίθμων από εκείνες του αλγορίθμου UH3-DP [1]. Παρατηρούμε οι αλγόριθμοι που χρησιμοποιούν την αναπαράσταση Συνόλων Ακμών δίνουν καλύτερες λύσεις από εκείνους αντίστοιχους που χρησιμοποιούν την Netkeys, με τον ES1, να είναι ο επικρατέστερος όλων. Οι βελτιώσεις του ES1 σε σχέση με τον UH3-DP φαίνεται να μην υπερβαίνουν το 1% στα μοναδιαία σενάρια ενώ στα μη μοναδιαία είναι της τάξης 2-3%.

Στο γράφημα του σχήματος 6.2 απεικονίζεται οι χρόνοι εκτέλεσης για τους τέσσερεις αλγορίθμους [2]. Τα σημεία κάθε καμπύλης εκφράζουν το μέσο χρόνο εκτέλεσης μεταξύ των στιγμιότυπων ίδιου μεγέθους και των τεσσάρων σεναρίων. Η παρατηρούμενη διαφορά στην κλήση των καμπυλών δύο κωδικοποιήσεων αποδίδεται στην πολυπλοκότητα της διαδικασίας αποκωδικοποίησης των διανυσμάτων, που είναι και η κυρίαρχη διαδικασία. Η αποκωδικοποίηση ενός διανύσματος της Netkeys απαιτεί $\mathcal{O}\left(N^2 \log N\right)$ βήματα. Εφόσον θα γίνουν $10N \times 15N$ τέτοιες αποκωδικοποιήσεις, η υπολογιστική συνολική επιβάρυνση για μία εκτέλεση του εξελικτικού αλγορίθμου της τάξης των $N^4 \log N$ βημάτων. Από την άλλη, η αποκωδικοποίηση ενός διανύσματος Συνόλων Ακμών απαιτεί $\mathcal{O}\left(N\right)$ βήματα άρα η συνολική επιβάρυνση για μία εκτέλεση του εξελικτικού αλγορίθμου θα είναι της τάξης των N^3 βημάτων.

Ο αλγόριθμος ES1, ως επικρατέστερος, επιλέγεται για να εφαρμοστεί και στα μεγαλύτερα στιγμιότυπα: 70, 100 και 150 τερματικών κόμβων. Στον πίνακα 6.2 παρατίθενται οι αποστάσεις των

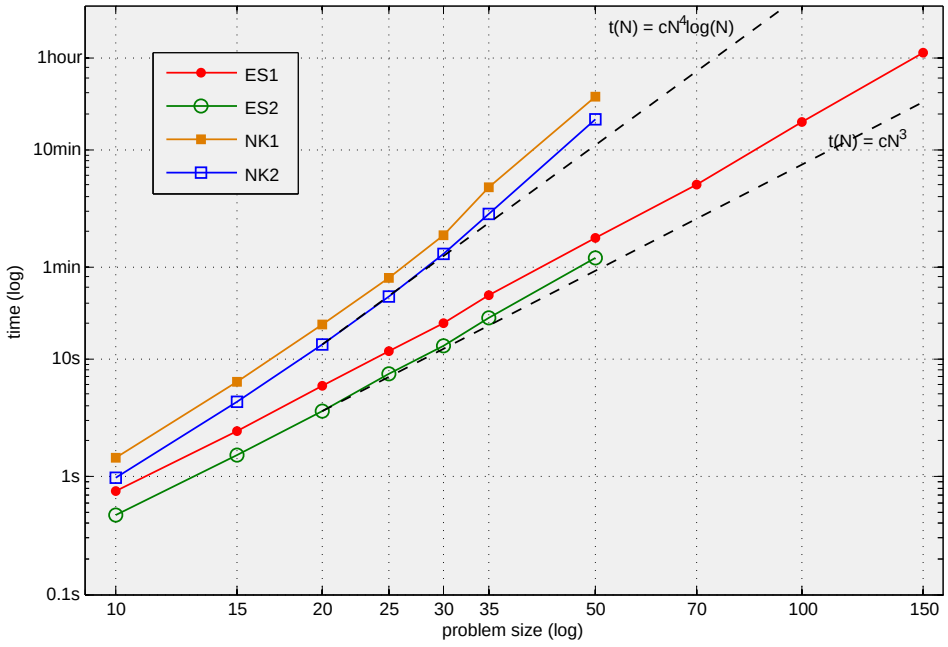
[1] Οι τιμές των γραφημάτων του σχήματος 6.1 παρατίθενται στον πίνακα Α΄.9 του Παραρτήματος.
[2] Οι τιμές του σχήματος 6.2 παρατίθενται στον πίνακα Α΄.8 του Παραρτήματος.



(α′) σενάριο RES-U *(β′) σενάριο EXT-U*

(γ′) σενάριο RES-NU *(δ′) σενάριο EXT-NU*

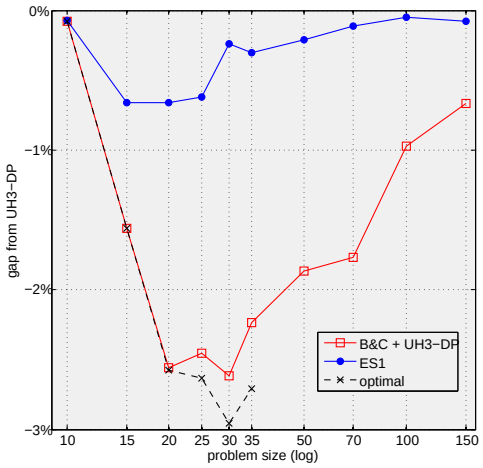
Σχήμα 6.1: Αποστάσεις λύσεων εξελικτικών αλγορίθμων από τις λύσεις του UH3-DP



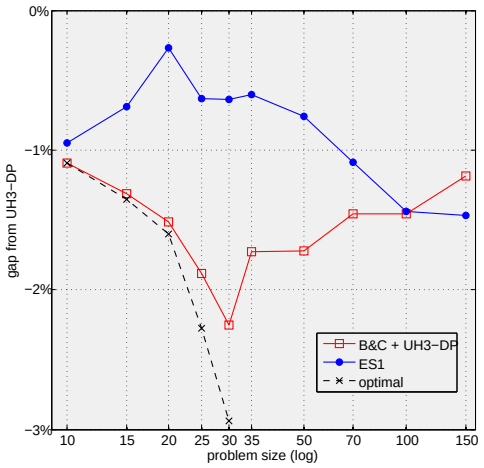
Σχήμα 6.2: Χρόνοι εκτέλεσης εξελικτικών αλγορίθμων

size	RES-U	EXT-U	RES-NU	EXT-NU
10	-0.07 %	-0.95 %	-2.33 %	-2.10 %
15	-0.66 %	-0.69 %	-3.22 %	-2.30 %
20	-0.66 %	-0.27 %	-2.89 %	-2.99 %
25	-0.62 %	-0.63 %	-2.29 %	-2.31 %
30	-0.24 %	-0.64 %	-2.41 %	-2.81 %
35	-0.30 %	-0.60 %	-2.49 %	-2.94 %
50	-0.21 %	-0.76 %	-2.10 %	-3.04 %
70	-0.11 %	-1.09 %	-1.61 %	-2.52 %
100	-0.05 %	-1.44 %	-1.24 %	-2.85 %
150	-0.08 %	-1.47 %	-1.10 %	-2.91 %

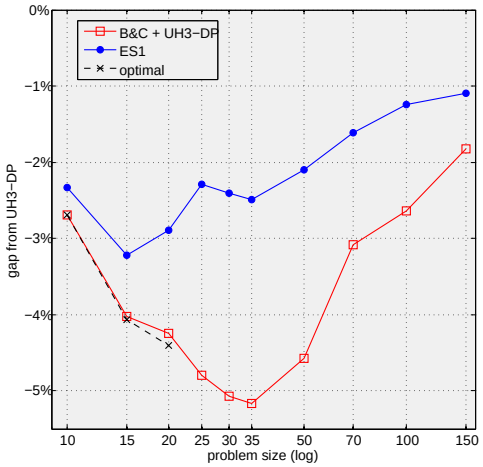
Πίνακας 6.2: Αποστάσεις λύσεων εξελικτικού αλγορίθμου ES1 από τις λύσεις του αλγόριθμου αναβαθμίσεων UH3-DP



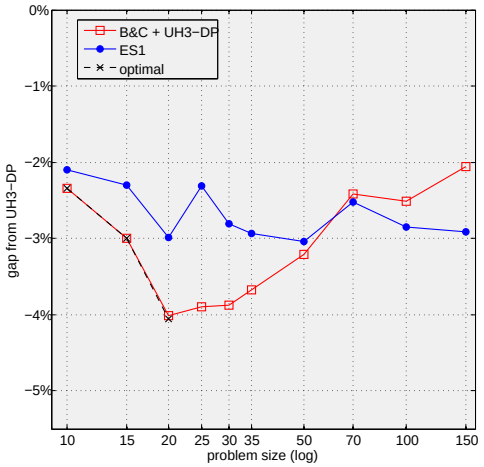
(α′) σενάριο RES-U



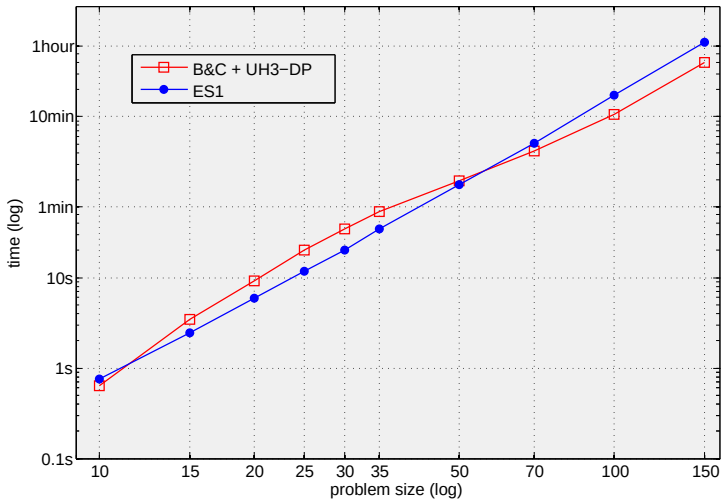
(β′) σενάριο EXT-U



(γ′) σενάριο RES-NU



(δ′) σενάριο EXT-NU



(ε′) μέσοι χρόνοι εκτέλεσης

Σχήμα 6.3: Σύγκριση μεταξύ εξελικτικού αλγόριθμου ES1 και αλγόριθμου Διακλάδωσης και Αποκοπής με ευρετική επανάκληση του UH3-DP

εξαγόμενων λύσεων από εκείνες του UH3-DP. Σε γενικές γραμμές οι βελτιώσεις σε σχέση με τον UH3-DP είναι υψηλότερες στα μη μοναδιαία σενάρια, όπως άλλωστε παρατηρήθηκε και στο σχήμα 6.1. Επίσης, στα δύο 'περιορισμένα' σενάρια οι τιμές των βελτιώσεων βαίνουν κατά κανόνα μειούμενες.

Ουσιαστικότερα συμπεράσματα μπορούμε να εξάγουμε από τη σύγκριση μεταξύ του εξελικτικού ES1 και του αλγόριθμου B&C με ευρετική επανάκληση του UH3-DP καθώς οι χρόνοι εκτέλεσης των δύο αλγορίθμων είναι συγκρίσιμοι. Στα γραφήματα του σχήματος 6.3, απεικονίζονται οι αποστάσεις των αλγορίθμων από τον ευρετικό UH3-DP για κάθε σενάριο και οι μέσοι χρόνοι εκτέλεσης τους. Παρατηρούμε καταρχήν ότι ο αλγόριθμος B&C δίνει κατά κανόνα καλύτερες λύσεις στις περισσότερες των περιπτώσεων. Ωστόσο, όπως παρατηρήσαμε και στο σχολιασμό των υπολογιστικών αποτελεσμάτων του προηγούμενου κεφαλαίου, οι βελτιώσεις που επιτυγχάνει ο αλγόριθμος B&C φθίνουν σημαντικά καθώς αυξάνεται το πλήθος των κόμβων. Ως αποτέλεσμα, στα μεγαλύτερα προβλήματα των 'εκτεταμένων' σεναρίων οι λύσεις του εξελικτικού αλγορίθμου υπερτερούν έναντι εκείνων του αλγορίθμου B&C.

Κεφάλαιο 7

Σύνοψη και συμπεράσματα / Προτάσεις μελλοντικής έρευνας

Στο παρόν κεφάλαιο παρουσιάζονται τα σημαντικότερα συμπεράσματα από τα αποτελέσματα της παρούσας διδακτορικής διατριβής, ενώ παρατίθενται και προτάσεις για ενδεχόμενη μελλοντική εργασία.

7.1 Σύνοψη και συμπεράσματα

Η παρούσα διατριβή επικεντρώθηκε στη μελέτη του προβλήματος MLCMST και την ανάδειξη αλγορίθμων που να αντιμετωπίζουν ένα ευρύ φάσμα στιγμιότυπων του. Συνοπτικά:

- Αφού έγινε μία αναδρομή στους αλγορίθμους που έχουν προταθεί μέχρι σήμερα για το πρόβλημα MLCMST, αναλύθηκαν οι ιδιαιτερότητες των στιγμιότυπων που έχουν εξεταστεί στη βιβλιογραφία αλλά και η αδυναμία των αλγορίθμων αυτών να αντιμετωπίσουν γενικότερα στιγμιότυπα. Συγκεκριμένα, υπογραμμίστηκε η αδυναμία να αντιμετωπιστούν στιγμιότυπα που δε χαρακτηρίζονται Κατ' ανάγκην από α) χαμηλές τιμές μέγιστης επιτρεπτής χωρητικότητας (Z^L) ή β) μοναδιαίες απαιτήσεις κίνησης σε κάθε τερματικό κόμβο.
- Εξετάστηκε η δυνατότητα επίλυσης του προβλήματος με τεχνικές μεικτού ακέραιου γραμμικού προγραμματισμού. Η πλήρης επίλυση των προβλημάτων κατέστει δυνατή μόνο για σχετικά μικρά στιγμιότυπα: μέχρι 20-35 κόμβους ανάλογα με την περίπτωση. Επίσης, τα κάτω όρια που προέκυψαν από την επίλυση των γραμμικών χαλαρώσεων των μοντέλων απείχαν πολύ από τις βέλτιστες τιμές -πλην της ειδικής περίπτωσης που περιγράφηκε παραπάνω- και ως εκ τούτου δε μπορούν να αξιοποιηθούν για την αξιολόγηση εφικτών λύσεων.
- Προτάθηκαν ευρετικοί αλγορίθμοι αναβαθμίσεων με στόχο την εύρεση λύσεων καλής ποιότη-

τας. Συγκεκριμένα:

- Παρουσιάστηκε επέκταση αλγορίθμου της βιβλιογραφίας ώστε να είναι δυνατή η εφαρμογή σε μη μοναδιαία στιγμιότυπα.
- Γενικεύτηκε η ιδέα των αναβαθμίσεων και παρουσιάστηκαν δύο νέοι αλγόριθμοι με στόχο την αντιμετώπιση περιπτώσεων (υψηλές τιμές Z^L) όπου ο επεκτεταμένος αλγόριθμος της βιβλιογραφίας δε κατάφερνε να δώσει λύσεις καλής ποιότητας.
- Μελετήθηκε το υποπρόβλημα επιλογής των κόμβων που θα επανασυνδέονται στα πλαίσια μίας αναβάθμισης. Παρουσιάστηκαν διαδικασίες δυναμικού προγραμματισμού για τη βέλτιστη επιλογή σε καθένα από τους τρεις αλγορίθμους αναβαθμίσεων.

Οι προταθέντες αλγόριθμοι αναβαθμίσεων έδωσαν λύσεις καλύτερης ποιότητας από τον επεκτεταμένο αλγόριθμο της βιβλιογραφίας σε όλα τα εξετασθέντα σενάρια. Επίσης, οι αλγόριθμοι αναβαθμίσεων χαρακτηρίζονται ως αρκετά αποδοτικοί δεδομένου ότι οι εκτελέσεις δεν υπερέβησαν σε διάρκεια το 1.5s στο μέγιστο εξετασθέν μέγεθος 150 κόμβων.

- Οι πολύ χαμηλές υπολογιστικές απαιτήσεις των αλγορίθμων αναβαθμίσεων καθιστούν δυνατή την ενσωμάτωση τους σε άλλες, κατά κανόνα πιο χρονοβόρες, διαδικασίες. Σε αυτή την κατεύθυνση εξετάστηκαν δύο προσεγγίσεις οι οποίες δίνουν περεταίρω βελτιώσεις:
  - Τροποποιημένες εκδόσεις των αλγορίθμων αναβαθμίσεων ενσωματώθηκαν ως ευρετικές διαδικασίες σε αλγόριθμο Διακλάδωσης και Αποκοπής δημοφιλούς πακέτου βελτιστοποίησης. Κατ' αυτόν τον τρόπο βελτιώθηκε η δυνατότητα του αλγορίθμου Διακλάδωσης και Αποκοπής να ανακαλύπτει εφικτές λύσεις.
  - Εξετάστηκε η εφαρμογή εξελικτικών αλγορίθμων βασισμένων στις αναπαραστάσεις δέντρων Συνόλων Ακμών και Netkeys. Οι προτεινόμενοι αλγόριθμοι αναβαθμίσεων χρησιμοποιήθηκαν ως τροφοδότες λύσεων καλής ποιότητας κατά την αρχικοποίηση των πληθυσμών.

7.2 Προτάσεις μελλοντικής έρευνας

Οι προτάσεις για μελλοντική έρευνα στα πλαίσια της παρουσιαζόμενης διατριβής θα μπορούσαν να συνοψιστούν στα εξής σημεία:

- Περεταίρω αξιοποίηση των αλγορίθμων αναβαθμίσεων στα πλαίσια του προβλήματος MLCMST ως συστατικά υβριδικών διαδικασιών. Για παράδειγμα, οι αλγόριθμοι αναβαθμίσεων θα μπορούσαν να χρησιμοποιηθούν στα πλαίσια του αλγορίθμου GRASP των Martins et al. (2009) αντί της πλήρους επίλυσης των μοντέλων MILP των υποπροβλημάτων που προκύπτουν.

- Εξέταση της ιδέας των αναβαθμίσεων και σε άλλα προβλήματα δέντρων επικάλυψης με περιορισμούς χωρητικότητας. Εκτός από προβλήματα ελαχιστοποίησης κόστους θα μπορούσε να εξεταστεί η εφαρμογή σε Προβλήματα πολυστοχικής βελτιστοποίησης, όπως αυτό που εξέτασαν οι Papagianni et al. (2008, 2009a) όπου εκτός από την ελαχιστοποίηση του κόστους κατασκευής ενός δικτύου επικοινωνιών ζητούμενη είναι και η ελαχιστοποίηση της καθυστέρησης από άκρο σε άκρο ανά κλάση υπηρεσίας.
- Ανάπτυξη μαθηματικών μοντέλων στην κατεύθυνση εύρεσης κάτω ορίων που να προσεγγίζουν τις βέλτιστες λύσεις σε ικανοποιητικό βαθμό.

Βιβλιογραφία

Achterberg, T., Berthold, T., Koch, T., and Wolter, K. (2008). Constraint integer programming : A new approach to integrate cp and mip. *LNCS*, 5015:6--20.

Ahuja, R. K., Orlin, J. B., and Sharma, D. (1998). New neighborhood search structures for the capacitated minimum spanning tree problem. Working papers WP 4040-98., Massachusetts Institute of Technology (MIT), Sloan School of Management.

Ahuja, R. K., Orlin, J. B., and Sharma, D. (2001). Multi-exchange neighborhood structures for the capacitated minimum spanning tree problem. *Mathematical Programming*, 91:71--97.

Amberg, A., Domschke, W., and Voß, S. (1996). Capacitated minimum spanning trees: algorithms using intelligent search. *Combin. Optim. Theory Practice*, 1:9–40.

Balas, E. and Zemel, E. (1980). An algorithm for large zero-one knapsack problems. *Oper. Res.*, 28:1130--1154.

Berger, D., Gendron, B., Potvin, J.-Y., Raghavan, S., and Soriano, P. (2000). Tabu search for a network loading problem with multiple facilities. *Journal of Heuristics*, 6:253--267.

Borgwardt, K. (1987). *The simplex method: a probabilistic analysis.* Algorithms and combinatorics. Springer.

Carrano, E., Fonseca, C., Takahashi, R., Pimenta, L., and Neto, O. (2007). A preliminary comparison of tree encoding schemes for evolutionary algorithms. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 1969 --1974.

Cayley, A. (1889). A theorem on trees. *Quart. Journal of Mathematics*, 23:376--378.

CBC: part of Computational Infrastructure for Operations Research - COIN-OR (COIN-OR Foundation, Inc.). website: https://projects.coin-or.org/Cbc.

Chou, H., Premkumar, G., and Chu, C.-H. (2001). Genetic algorithms for communications network design - an empirical study of the factors that influence performance. *Evolutionary Computation, IEEE Transactions on*, 5(3):236--249.

Chu, C.-h., Premkumar, G., Chou, C., Sun, J., and Ofchina, P. R. (1999). Dynamic degree constrained network design: a genetic algorithm approach. In *Proceedings GECCO-99. Genetic and Evolutionary Computation Conference. Eighth International Conference on Genetic Algorithms (ICGA-99) and the Fourth Annual Genetic Programming Conference (GP99*, pages 141--148.

Dantzig, G. (1957). Discrete variable extremum problems. *Oper. Res.*, 5:266--277.

Dantzig, G. B. (1951). *Maximization of a Linear Function of Variables Subject to Linear Inequalities, in Activity Analysis of Production and Allocation*, chapter XXI. Wiley, New York.

Davis, L., Orvosh, D., Cox, A., and Qiu, Y. (1993). A genetic algorithm for survivable network design. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 408--415, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

de Souza, M. C., Duhamel, C., and Ribeiro, C. C. (2004). *A GRASP heuristic for the capacitated minimum spanning tree problem using a memory-based local search strategy*, pages 627--657. Kluwer Academic Publishers.

Esau, L. R. and Williams, K. C. (1966). On teleprocessing system design: part ii a method for approximating the optimal network. *IBM Syst. J.*, 5:142--147.

Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2:5--30.

Feo, T. A. and Resende, M. G. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8(2):67 -- 71.

Feo, T. A. and Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109--133.

Gamvros, I., Golden, B., and Raghavan, S. (2006). The multilevel capacitated minimum spanning tree problem. *INFORMS J. Comput.*, 18(3):348--365.

Gamvros, I., Raghavan, S., and Golden, B. (2002). An evolutionary approach for the multilevel capacitated minimum spanning tree. Technical report, Institute for Systems Research, University of Maryland.

Garg, N., Khandekar, R., Konjevod, G., Ravi, R., Salman, F. S., and II, A. S. (2001). On the integrality gap of a natural formulation of the single-sink buy-at-bulk network design problem. In *IPCO*, pages 170--184.

Gaube, T. and Rothlauf, F. (2001). The link and node biased encoding revisited: Bias and adjustment of parameters. In *EvoWorkshops'01*, pages 1--10.

Gavish, B. (1982). Topological design of centralized computer networks - formulations and algorithms. *Networks*, 12:355--377.

Gavish, B. (1983). Formulations and algorithms for the capacitated minimal directed tree problem. *J. ACM*, 30:118--132.

Gavish, B. (1991). Topological design of telecommunication networks-local access design methods. *Annals of Operations Research*, 33:17--71.

GLPK: GNU Linear Programming Kit (GNU General Public License). website: http://www.gnu.org/software/glpk/.

Gomory, R. E. (1958). Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64(5):275--278.

Gondzio, J. and Terlaky, T. (1994). A computational view of interior-point methods for linear programming. In *IN: ADVANCES IN LINEAR AND INTEGER PROGRAMMING*, pages 103--144. University Press.

Gottlieb, J., Julstrom, B. A., Raidl, G. R., and Rothlauf, F. (2001). Prufer numbers: A poor representation of spanning trees for evolutionary search. In *Genetic and Evolutionary Computation Conference*.

Gouveia, L. (1993). A comparison of directed formulations for the capacitated minimal spanning tree problem. *Telecommunication Systems*, 1:51--76.

Gouveia, L. (1995). A 2n constraint formulation for the capacitated minimal spanning tree problem. *Oper. Res.*, 43:130--141.

Gouveia, L. and Martins, P. (2000). A hierarchy of hop-indexed models for the capacitated minimum spanning tree problem. *Networks*, 35(1):1--16.

Gouveia, L. and Martins, P. (2005). The capacitated minimum spanning tree problem: revisiting hop-indexed formulations. *Comput. Oper. Res.*, 32:2435--2452.

Gurobi (Gurobi Optimization). website: http://www.gurobi.com/products/gurobi-optimizer/gurobi-overview.

Hall, L. (1996). Experience with a cutting plane approach for the capacitated spanning tree problem. *INFORMS J. Comput.*, 8:219--234.

Hoang, A., Le, V., and Nguyen, N. (2010). A novel particle swarm optimization-based algorithm for the optimal communication spanning tree problem. In *Communication Software and Networks, 2010. ICCSN '10. Second International Conference on*, pages 232 --236.

ILOG CPLEX (IBM, corp.). website: http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/.

ILOG CPLEX V12.1 - Parameters Reference Manual (IBM, corp.). retrieved from ftp://ftp.software.ibm.com/software/websphere/ilog/docs/optimization/cplex/ps_refparameterscplex.pdf.

ILOG CPLEX V12.1 - User's Manual for CPLEX (IBM, corp.). retrieved from ftp://public.dhe.ibm.com/software/websphere/ilog/docs/optimization/cplex/ps_usrmancplex.pdf.

JGAP - Java Genetic Algorithms and Genetic Programming Package. (GNU General Public License). website: http://jgap.sf.net/.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373--396.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942--1948 vol.4.

Khachiyan, L. (1979). A polynomial algorithm in linear programming. *Sov. Math. Dokl.*, 20(4):191--194.

Klee, V. and Minty, G. J. (1972). How good is the simplex algorithm? In Shisha, O., editor, *Inequalities*, volume III, pages 159--175. Academic Press, New York.

Knowles, J. and Corne, D. (2000). A new evolutionary approach to the degree-constrained minimum spanning tree problem. *Evolutionary Computation, IEEE Transactions on*, 4(2):125 --134.

Krishnamoorthy, M., Ernst, A. T., and Sharaiha, Y. M. (2001). Comparison of algorithms for the degree constrained minimum spanning tree. *Journal of Heuristics*, 7:587--611.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48--50.

Land, A. H. and Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497--520.

Li, Y. (2001). An effective implementation of a direct spanning tree representation in gas. In *Proceedings of the EvoWorkshops on Applications of Evolutionary Computing*, pages 11--19, London, UK, UK. Springer-Verlag.

Li, Y. and Bouchebaba, Y. (2000). A new genetic algorithm for the optimal communication spanning tree problem. In *Artificial Evolution*, volume 1829 of *LNCS*, pages 162--173. Springer Berlin / Heidelberg.

Lin, L. and Gen, M. (2006). Node-based genetic algorithm for communication spanning tree problem. *IEICE TRANSACTIONS on Communications*, E89-B(4):1091–1098.

lp_solve (GNU General Public License). website: http://lpsolve.sourceforge.net/.

Martins, A. X., Souza, M. C., Souza, M. J., and Toffolo, T. A. (2009). Grasp with hybrid heuristic-subproblem optimization for the multi-level capacitated minimum spanning tree problem. *J. Heuristics*, 15:133--151.

Martins, A. X., Souza, M. J. F., and Souza, M. C. (2005). Modelos matemáticos para o problema da árvore geradora mínima capacitada em níveis. In *Anais do XXXVII Simpósio Brasileiro de Pesquisa Operacional*, page 1971–1982, Gramado, Brazil.

Martins, P. (2007). Enhanced second order algorithm applied to the capacitated minimum spanning tree problem. *Comput. Oper. Res.*, 34:2495--2519.

Mehrotra, S. (1992). On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575--601.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, New York.

Mittelmann, H. (webpage). Benchmarks for optimization software. retrieved from http://plato.asu.edu/bench.html on 13/9/2012.

MOSEK (MOSEK ApS). website: http://mosek.com/products/mosek/.

Orlowski, S., Pióro, M., Tomaszewski, A., and Wessäly, R. (2007). SNDlib 1.0--Survivable Network Design Library. In *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*. http://sndlib.zib.de, extended version accepted in Networks, 2009.

Orlowski, S., Pióro, M., Tomaszewski, A., and Wessäly, R. (2010). SNDlib 1.0--Survivable Network Design Library. *Networks*, 55(3):276--286.

O.T.E.-S.A. (2011). Main telecommunication tariffs.

Palmer, C. and Kershenbaum, A. (1994). Representing trees in genetic algorithms. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 379--384.

Palmer, C. C., Kershenbaum, A., Hummel, S. F., Slyke, M. V., and Boorstyn, R. R. (1995). An approach to a problem in network design using genetic algorithms.

Papadimitriou, C. H. (1978). The complexity of the capacitated tree problem. *Networks*, 8(3):217--230.

Papagianni, C., Papadopoulos, K., Pappas, C., Tselikas, N., Kaklamani, D., and Venieris, I. (2008). Communication network design using particle swarm optimization. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 915 -- 920.

Papagianni, C., Papadopoulos, K., Pappas, C., Tselikas, N., Kaklamani, D., and Venieris, I. (2009a). Particle swarm optimization for communication network design. *Analele Universitatii de Vest din Timisoara*, 47(2):65 -- 94.

Papagianni, C., Pappas, C., Lefkaditis, N., and Venieris, I. (2009b). A particle swarm optimization approach to the multi level capacitated minimum spanning tree problem. *Mathematica Balkanica*, 23(3-4):303 -- 324.

Papagianni, C., Pappas, C., Lefkaditis, N., and Venieris, I. (2009c). Particle swarm optimization for the multi level capacitated minimum spanning tree. In *Computer Science and Information Technology, 2009. IMCSIT '09. International Multiconference on*, pages 765 --770.

Pappas, C., Anadiotis, A.-C., Papagianni, C., and Venieris, I. (2013). Heuristics for the multi-level capacitated minimum spanning tree problem. *Optimization Letters*, pages 1--12.

Piggott, P. and Suraweera, F. (1995). Encoding graphs for genetic algorithms: An investigation using the minimum spanning tree problem. In *Selected papers from the AI'93 and AI'94 Workshops on Evolutionary Computation, Process in Evolutionary Computation*, pages 305--314, London, UK. Springer-Verlag.

Pisinger, D. (1995). *Algorithms for Knapsack Problems*. PhD thesis, University of Copenhagen.

Prüfer, H. (1918). Neuer beweis eines satzes über permutationen. *Arch. Math. Phys.*, 27:742–744.

Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, pages 1389--1401.

Raidl, G. (2000). An efficient evolutionary algorithm for the degree-constrained minimum spanning tree problem. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 1, pages 104 --111.

Raidl, G. and Julstrom, B. (2003). Edge sets: an effective evolutionary coding of spanning trees. *Evolutionary Computation, IEEE Transactions on*, 7(3):225 -- 239.

Raidl, G. R. and Drexel, C. (2000). A predecessor coding in an evolutionary algorithm for the capacitated minimum spanning tree problem. In *Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference, pages 309–316, Las Vegas, NV*, pages 309--316.

Riget, J. and Vesterstroem, J. (2002). A diversity-guided particle swarm optimizer - the ARPSO. *EVALife Technical Report*, (2).

Rothfarb, B. and Goldstein, M. (1971). The one-terminal telpak problem. *Operations Research*, 19(1):156--169.

Rothlauf, F., Gerstacker, J., and Heinzl, A. (2003). On the optimal communication spanning tree problem.

Rothlauf, F. and Goldberg, D. (2000). Pruefer numbers and genetic algorithms: A lesson on how the low locality of an encoding can harm the performance of gas. In *Parallel Problem Solving from Nature PPSN VI*, volume 1917 of *Lecture Notes in Computer Science*, pages 395--404. Springer Berlin / Heidelberg.

Rothlauf, F., Goldberg, D., and Heinzl, A. (2000). Network random keys - a tree network representation scheme for genetic and evolutionary algorithms. Technical Report 8, University of Bayreuth, Germany.

Rothlauf, F., Goldberg, D., and Heinzl, A. (2002). Network random keys - a tree network representation scheme for genetic and evolutionary algorithms. *Evolutionary Computation*, 10(1):75--97.

Salman, F. S., Cheriyan, J., Ravi, R., and Subramanian, S. (1997). Buy-at-bulk network design: Approximating the single-sink edge installation problem. In *SODA*, pages 619--628.

Salman, F. S., Cheriyan, J., Ravi, R., and Subramanian, S. (2000). Approximating the single-sink link-installation problem in network design. *SIAM J. on Optimization*, 11:595--610.

Salman, F. S., Ravi, R., and Hooker, J. N. (2008). Solving the capacitated local access network design problem. *INFORMS J. Comput.*, 20(2):243--254.

SCIP: Solving Constraint Integer Programs (Zuse Institute Berlin). website: http://scip.zib.de/.

Sharaiha, Y. M., Gendreau, M., Laporte, G., and Osman, I. H. (1997). A tabu search algorithm for the capacitated shortest spanning tree problem. *Networks*, 29(3):161--171.

Soak, S.-M. (2006). A new evolutionary approach for the optimal communication spanning tree problem. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E89-A:2882--2893.

Soak, S.-M., Corne, D., and Ahn, B.-H. (2004). A new encoding for the degree constrained minimum spanning tree problem. In *Knowledge-Based Intelligent Information and Engineering Systems*, volume 3213 of *LNCS*, pages 952--958. Springer Berlin / Heidelberg.

Soak, S.-M., Corne, D. W., and Ahn, B.-H. (2006). The edge-window-decoder representation for tree-based problems. *IEEE Transactions on Evolutionary Computation*, 10(2):124--144.

Szabo, Z. and Kovacs, M. (2003). On interior-point methods and simplex method in linear programming. *Analele Stiintifice ale Universitatii "Ovidius" Constanta, Seria Matematica*, 11(2):155--162.

Uchoa, E., Fukasawa, R., Lysgaard, J., Pessoa, A., de Aragao, M. P., and Andrade, D. (2007). Robust branch-cut-and-price for the capacitated minimum spanning tree problem over a large extended formulation. *Math. Program.*, 112:443--472.

Uchoa, E., Toffolo, T. A. M., de Souza, M. C., and Martins, A. X. (2009). Branch-and-cut and grasp with hybrid local search for the multi-level capacitated minimum spanning tree problem. In *INOC 2009*, Pisa, Italy.

Xpress-MP (FICO). website: http://www.fico.com/en/Products/DMTools/Pages/FICO-Xpress-Optimization-Suite.aspx.

Zhou, G. and Gen, M. (1997). A note on genetic algorithms for degree-constrained spanning tree problems. *Networks*, 30(2):91--95.

Zhou, G. and Gen, M. (1998). An effective genetic algorithm approach to the quadratic minimum spanning tree problem. *Computers and Operations Research*, 25(3):229 -- 237.

Δημοσιεύσεις

Διεθνή επιστημονικά περιοδικά:

J1. **Pappas, C.**, Anadiotis, A.-C., Papagianni, C., and Venieris, I.: Heuristics for the multi-level capacitated minimum spanning tree problem. *Optimization Letters*, pages 1–12, doi: 10.1007/s11590-013-0607-8OI, 2013.

J2. Papagianni, C., **Pappas, C.**, Lefkaditis, N., and Venieris, I.: A particle swarm optimization approach to the multi level capacitated minimum spanning tree problem. *Mathematica Balkanica*, 23(3- 4):303 – 324, 2009.

J3. Papagianni, C., Papadopoulos, K., **Pappas, C.**, Tselikas, N., Kaklamani, D., and Venieris, I.: Particle swarm optimization for communication network design. *Analele Universitatii de Vest din Timisoara*, 47(2):65 – 94, 2009.

Πρακτικά διεθνών επιστημονικών συνεδρίων:

C1. Papagianni, C., **Pappas, C.**, Lefkaditis, N., and Venieris, I.: Particle swarm optimization for the multi level capacitated minimum spanning tree. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 765 – 770, October, 2009.

C2. Papadopoulos, K., Papagianni, C., **Pappas, C.**, Kaklamani, D., and Venieris, I.: Beam Array Optimization For Smart Antenna Systems Using Stochastic Algorithms. In *European Conference on Antennas and Propagation(EuCAP)*, pages 23 - 27, Berlin, March 2009.

C3. Papagianni, C., Karagiannis, G., Tselikas, N., Sfakianakis, E., Chochliouros, I., Kabilafkas, D., Cinkler, T., Westberg, L., Sjodin, P., Hidell, M., de Groot, S., Kontos, T., Katsigiannis, C., **Pappas, C.**, Antonakopoulou, A., Venieris, I.: Supporting end-to-end resource virtualization for Web 2.0 applications using Service Oriented Architecture, In *2nd IEEE Workshop on Enabling the Future Service-Oriented Internet*, New Orleans, November, 2008.

C4. Kapitsaki, G., Kateros, D., **Pappas, C.**, Tselikas, N., and Venieris, I.: Model-Driven Development of Composite Web Applications, In *Proceedings of the 10th International Conference on Information Integration and Web-based Applications and Services (iiWAS08)*, November, 2008.

C5. Papagianni, C., Papadopoulos, K., **Pappas, C.**, Tselikas, N., Kaklamani, D., and Venieris, I.: Communication network design using particle swarm optimization. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 915 – 920, October, 2008.

Παράρτημα Α΄

Εκτενή υπολογιστικά αποτελέσματα

	σενάριο RES-U						σενάριο RES-NU		
διατύπωση	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 35$	$N = 10$	$N = 15$	$N = 20$
SCF	-7,89%	-7,79%	-7,63%	-7,61%	-7,41%	-7,16%	-21,56%	-20,39%	-19,57%
SCF2	-6,48%	-6,36%	-6,28%	-6,34%	-6,19%	-5,95%	-19,96%	-18,80%	-17,99%
MCF	-7,89%	-7,79%	-7,63%	-7,61%	-7,41%	-7,16%	-21,56%	-20,39%	-19,57%
MCF2	-6,48%	-6,36%	-6,28%	-6,34%	-6,19%	-5,95%	-19,96%	-18,80%	-17,99%
MCF3	-6,07%	-5,89%	-5,75%	-5,76%	-5,52%	-5,26%	-19,19%	-17,93%	-17,13%
CISCF	-7,30%	-7,23%	-7,05%	-7,08%	-6,97%	-6,74%	-21,52%	-20,33%	-19,51%
CISCF2	-6,48%	-6,36%	-6,28%	-6,34%	-6,19%	-5,95%	-19,96%	-18,80%	-17,99%
CIMCF	-7,30%	-7,23%	-7,05%	-7,08%	-6,97%	-6,74%	-21,52%	-20,33%	-19,51%
CIMCF2	-6,48%	-6,36%	-6,28%	-6,34%	-6,19%	-5,95%	-19,96%	-18,80%	-17,99%
CIMCF3	-6,07%	-5,89%	-5,75%	-5,76%	-5,52%	-	-19,19%	-17,93%	-17,13%

	σενάριο EXT-U					σενάριο EXT-NU		
διατύπωση	$N = 10$	$N = 15$	$N = 20$	$N = 25$	$N = 30$	$N = 10$	$N = 15$	$N = 20$
SCF	-46,14%	-51,14%	-54,06%	-55,44%	-56,34%	-52,56%	-56,09%	-57,23%
SCF2	-38,40%	-43,56%	-47,40%	-49,13%	-50,63%	-46,08%	-49,61%	-51,30%
MCF	-46,14%	-51,14%	-54,06%	-55,44%	-56,34%	-52,56%	-56,09%	-57,23%
MCF2	-38,40%	-43,56%	-47,40%	-49,13%	-50,63%	-46,08%	-49,61%	-51,30%
MCF3	-34,87%	-39,67%	-43,14%	-44,86%	-46,20%	-42,56%	-45,64%	-46,98%
CISCF	-45,29%	-50,30%	-53,29%	-54,72%	-55,67%	-52,53%	-	-
CISCF2	-38,40%	-43,56%	-47,40%	-49,13%	-50,63%	-46,08%	-	-
CIMCF	-45,29%	-50,30%	-53,29%	-54,72%	-55,67%	-52,53%	-	-
CIMCF2	-38,40%	-43,56%	-47,40%	-49,13%	-50,63%	-46,08%	-	-
CIMCF3	-34,87%	-39,67%	-43,14%	-	-	-	-	-

Πίνακας Α΄.1: Αποστάσεις κάτω ορίων από βέλτιστες τιμές

σενάριο RES-U

	διατύπωση									
μέγεθος	SCF	SCF2	MCF	MCF2	MCF3	CISCF	CISCF2	CIMCF	CIMCF2	CIMCF3
10	2	4	5	10	20	1	3	5	7	16
15	4	5	45	50	276	2	3	65	60	235
20	5	9	177	263	1685	4	4	262	289	1008
25	10	13	703	969	7461	5	8	1180	1169	4510
30	14	20	2156	2576	34204	7	8	3324	3456	17644
35	21	33	6079	6888	126626	10	12	8670	8983	-
50	87	117	55349	69051	-	28	30	-	-	-
70	301	443	-	-	-	63	62	-	-	-
100	1079	1306	-	-	-	161	142	-	-	-
150	4270	6642	-	-	-	513	383	-	-	-

σενάριο EXT-U

	διατύπωση									
μέγεθος	SCF	SCF2	MCF	MCF2	MCF3	CISCF	CISCF2	CIMCF	CIMCF2	CIMCF3
10	2	3	4	11	19	32	37	88	103	170
15	8	7	47	76	256	80	101	493	604	2579
20	7	10	171	334	2128	168	191	2119	2584	20687
25	9	18	740	1361	12174	274	309	8837	10657	-
30	17	30	2304	3819	52698	393	464	-	-	-
35	26	48	7309	12136	178570	561	664	-	-	-
50	98	192	82415	154014	-	1286	1549	-	-	-
70	442	757	-	-	-	-	-	-	-	-
100	2368	3601	-	-	-	-	-	-	-	-
150	40043	24316	-	-	-	-	-	-	-	-

σενάριο RES-NU

	διατύπωση									
μέγεθος	SCF	SCF2	MCF	MCF2	MCF3	CISCF	CISCF2	CIMCF	CIMCF2	CIMCF3
10	3	4	9	10	36	10	14	23	30	131
15	5	7	43	52	575	32	37	181	181	2142
20	5	8	198	241	3948	57	71	853	883	17794
25	10	15	891	1039	21352	97	119	3589	3927	-
30	16	24	2232	3111	86767	150	185	15235	14275	-
35	27	40	5832	8244	280138	219	269	-	-	-
50	89	137	68863	98079	-	557	655	-	-	-
70	316	404	-	-	-	1461	1507	-	-	-
100	1120	1201	-	-	-	-	-	-	-	-
150	5046	5732	-	-	-	-	-	-	-	-

σενάριο EXT-NU

	διατύπωση									
μέγεθος	SCF	SCF2	MCF	MCF2	MCF3	CISCF	CISCF2	CIMCF	CIMCF2	CIMCF3
10	5	3	6	8	18	689	848	1685	2012	-
15	4	6	36	50	247	-	-	-	-	-
20	9	10	207	272	2083	-	-	-	-	-
25	13	17	883	1193	11802	-	-	-	-	-
30	18	28	2776	4203	56646	-	-	-	-	-
35	32	46	8486	12704	-	-	-	-	-	-
50	109	189	137404	185025	-	-	-	-	-	-
70	762	731	-	-	-	-	-	-	-	-
100	6430	3396	-	-	-	-	-	-	-	-
150	59824	22401	-	-	-	-	-	-	-	-

Πίνακας Α΄.2: Χρόνοι επίλυσης γραμμικών χαλαρώσεων (ms)

		αλγόριθμος αναβαθμίσεων					
σενάριο	μέγεθος	UH1	UH2	UH3	UH1-DP	UH2-DP	UH3-DP
RES-U	10	1	1	1	1	1	1
	15	1	1	1	1	1	1
	20	1	1	1	1	1	1
	25	1	1	1	1	1	1
	30	1	2	1	1	2	2
	35	1	2	1	2	2	3
	50	2	4	2	4	4	6
	70	2	8	3	8	10	14
	100	5	18	6	19	23	34
	150	11	50	14	54	70	104
EXT-U	10	1	1	1	1	1	1
	15	1	1	1	1	1	2
	20	1	2	1	2	1	2
	25	1	3	1	3	2	4
	30	1	4	1	4	2	5
	35	1	5	1	5	3	7
	50	2	11	2	11	6	15
	70	4	24	4	22	13	31
	100	9	56	8	48	31	67
	150	20	136	20	116	83	166
RES-NU	10	1	1	1	1	1	1
	15	1	1	1	1	1	1
	20	1	1	1	1	1	2
	25	1	2	1	2	1	3
	30	1	3	1	3	2	4
	35	1	3	1	4	3	6
	50	2	8	2	8	5	13
	70	3	17	4	18	12	31
	100	6	43	9	47	29	82
	150	16	124	22	133	88	247
EXT-NU	10	1	3	1	3	1	5
	15	1	8	1	8	1	11
	20	1	13	1	14	1	21
	25	1	21	1	21	2	33
	30	1	31	1	31	2	48
	35	1	42	1	43	3	65
	50	2	102	3	92	7	140
	70	4	227	5	193	15	291
	100	9	521	10	420	35	593
	150	21	1317	24	991	96	1410

Πίνακας Α΄.3: Χρόνοι εκτέλεσης ευρετικών αλγορίθμων αναβαθμίσεων (ms)

		αλγόριθμος αναβαθμίσεων					
σενάριο	μεγεθος	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
RES-U	10	3,14%	3,14%	0,56%	0,56%	0,55%	0,55%
	15	3,82%	3,82%	1,62%	1,62%	1,51%	1,47%
	20	4,50%	4,50%	2,50%	2,50%	2,45%	2,44%
	25	3,84%	3,84%	2,54%	2,54%	2,52%	2,52%
	30	4,88%	4,88%	2,86%	2,86%	2,85%	2,85%
	35	4,89%	4,89%	3,18%	3,18%	3,17%	3,15%
EXT-U	10	2,65%	2,65%	1,13%	1,13%	1,04%	1,04%
	15	5,65%	5,65%	1,71%	1,71%	1,35%	1,35%
	20	7,84%	7,84%	1,98%	1,98%	1,79%	1,78%
	25	8,35%	8,35%	2,53%	2,53%	2,20%	2,17%
	30	9,78%	9,78%	3,59%	3,59%	2,87%	2,81%
RES-NU	10	9,12%	9,03%	2,89%	2,76%	2,46%	2,36%
	15	11,98%	11,90%	5,62%	5,06%	4,73%	4,15%
	20	13,77%	13,38%	6,38%	5,70%	5,14%	4,65%
EXT-NU	10	9,61%	9,50%	3,06%	2,92%	2,39%	2,21%
	15	15,57%	15,56%	5,21%	4,73%	3,72%	3,14%
	20	19,45%	19,41%	6,27%	5,68%	4,63%	4,21%

Πίνακας Α΄.4: Αποστάσεις ευρετικών λύσεων από τις βέλτιστες

		αλγόριθμος αναβαθμίσεων				
σενάριο	μέγεθος	UH1	UH1-DP	UH2	UH2-DP	UH3
RES-U	10	2.58 %	2.58 %	0.01 %	0.01 %	0.00 %
	15	2.33 %	2.33 %	0.15 %	0.15 %	0.04 %
	20	2.03 %	2.03 %	0.06 %	0.06 %	0.00 %
	25	1.30 %	1.30 %	0.02 %	0.02 %	0.00 %
	30	1.98 %	1.98 %	0.01 %	0.01 %	0.00 %
	35	1.68 %	1.68 %	0.02 %	0.02 %	0.02 %
	50	1.23 %	1.23 %	0.03 %	0.03 %	0.00 %
	70	0.86 %	0.86 %	0.02 %	0.02 %	0.00 %
	100	0.67 %	0.67 %	0.03 %	0.03 %	0.01 %
	150	0.50 %	0.50 %	0.01 %	0.01 %	0.00 %
EXT-U	10	1.59 %	1.59 %	0.09 %	0.09 %	0.00 %
	15	4.25 %	4.25 %	0.36 %	0.36 %	0.00 %
	20	5.95 %	5.95 %	0.20 %	0.20 %	0.01 %
	25	6.06 %	6.06 %	0.36 %	0.36 %	0.04 %
	30	6.80 %	6.80 %	0.75 %	0.75 %	0.06 %
	35	9.71 %	9.71 %	0.67 %	0.67 %	0.13 %
	50	14.68 %	14.68 %	0.62 %	0.62 %	0.11 %
	70	15.52 %	15.52 %	0.90 %	0.90 %	0.05 %
	100	11.00 %	11.00 %	2.39 %	2.39 %	0.15 %
	150	27.00 %	27.00 %	2.14 %	2.14 %	0.44 %
RES-NU	10	6.50 %	6.40 %	0.53 %	0.40 %	0.10 %
	15	7.61 %	7.52 %	1.42 %	0.87 %	0.56 %
	20	8.68 %	8.30 %	1.66 %	1.00 %	0.47 %
	25	7.85 %	7.47 %	1.76 %	1.13 %	0.82 %
	30	7.20 %	6.32 %	2.21 %	1.04 %	1.05 %
	35	6.59 %	6.24 %	1.71 %	1.05 %	0.64 %
	50	6.04 %	4.90 %	1.97 %	0.92 %	1.07 %
	70	4.86 %	3.94 %	1.84 %	0.82 %	1.22 %
	100	3.70 %	3.35 %	1.30 %	0.74 %	0.56 %
	150	3.16 %	2.95 %	1.25 %	0.57 %	0.60 %
EXT-NU	10	7.22 %	7.11 %	0.84 %	0.70 %	0.18 %
	15	12.11 %	12.10 %	2.00 %	1.53 %	0.57 %
	20	14.57 %	14.53 %	1.98 %	1.41 %	0.40 %
	25	14.96 %	14.77 %	2.55 %	1.95 %	0.71 %
	30	14.31 %	14.15 %	2.82 %	1.88 %	0.70 %
	35	15.07 %	14.92 %	2.85 %	2.42 %	0.59 %
	50	19.35 %	19.27 %	3.13 %	2.32 %	0.95 %
	70	18.46 %	18.38 %	2.75 %	2.26 %	0.78 %
	100	14.41 %	14.33 %	4.69 %	3.95 %	0.83 %
	150	26.10 %	26.69 %	4.76 %	4.07 %	0.89 %

Πίνακας Α΄.5: Αποστάσεις ευρετικών λύσεων από τις λύσεις του UH3-DP

σενάριο RES-U

	B&C + ευρετική επανάκληση						
μέγεθος	χωρίς ευρ. επανάκληση	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
10	-0.08 %	-0.08 %	-0.08 %	-0.08 %	-0.08 %	-0.08 %	-0.08 %
15	-1.56 %	-1.56 %	-1.56 %	-1.56 %	-1.56 %	-1.56 %	-1.56 %
20	-2.53 %	-2.56 %	-2.56 %	-2.56 %	-2.56 %	-2.56 %	-2.56 %
25	-1.94 %	-2.35 %	-2.35 %	-2.48 %	-2.50 %	-2.46 %	-2.45 %
30	-1.76 %	-2.35 %	-2.35 %	-2.57 %	-2.53 %	-2.68 %	-2.62 %
35	-0.71 %	-1.78 %	-1.78 %	-2.15 %	-2.14 %	-2.23 %	-2.24 %
50	2.30 %	-1.03 %	-1.03 %	-1.78 %	-1.82 %	-1.86 %	-1.86 %

σενάριο EXT-U

	B&C + ευρετική επανάκληση						
μέγεθος	χωρίς ευρ. επανάκληση	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
10	-1.10 %	-1.10 %	-1.10 %	-1.10 %	-1.10 %	-1.10 %	-1.10 %
15	-1.29 %	-1.33 %	-1.33 %	-1.31 %	-1.31 %	-1.31 %	-1.31 %
20	-1.34 %	-1.38 %	-1.38 %	-1.52 %	-1.52 %	-1.52 %	-1.51 %
25	0.06 %	-1.38 %	-1.38 %	-1.79 %	-1.82 %	-1.83 %	-1.88 %
30	5.04 %	-0.74 %	-0.74 %	-1.56 %	-1.55 %	-2.08 %	-2.25 %
35	10.72 %	0.70 %	0.70 %	-1.07 %	-1.06 %	-1.72 %	-1.73 %
50	38.32 %	3.18 %	3.18 %	-1.24 %	-1.18 %	-1.76 %	-1.72 %

σενάριο RES-NU

	B&C + ευρετική επανάκληση						
μέγεθος	χωρίς ευρ. επανάκληση	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
10	-2.69 %	-2.69 %	-2.69 %	-2.69 %	-2.69 %	-2.69 %	-2.69 %
15	-3.92 %	-4.02 %	-4.02 %	-4.03 %	-4.06 %	-4.02 %	-4.03 %
20	-3.99 %	-4.27 %	-4.24 %	-4.25 %	-4.34 %	-4.23 %	-4.25 %
25	-4.02 %	-4.57 %	-4.51 %	-4.52 %	-4.53 %	-4.47 %	-4.80 %
30	-4.11 %	-4.74 %	-4.72 %	-4.93 %	-4.98 %	-5.24 %	-5.07 %
35	-3.80 %	-4.41 %	-4.78 %	-4.89 %	-5.20 %	-5.08 %	-5.17 %
50	-1.16 %	-2.73 %	-2.80 %	-3.26 %	-3.83 %	-4.02 %	-4.57 %

σενάριο EXT-NU

	B&C + ευρετική επανάκληση						
μέγεθος	χωρίς ευρ. επανάκληση	UH1	UH1-DP	UH2	UH2-DP	UH3	UH3-DP
10	-2.34 %	-2.34 %	-2.34 %	-2.34 %	-2.34 %	-2.34 %	-2.34 %
15	-2.82 %	-2.92 %	-2.96 %	-2.94 %	-3.00 %	-3.00 %	-3.00 %
20	-2.66 %	-3.64 %	-3.65 %	-3.63 %	-3.79 %	-3.83 %	-4.01 %
25	-2.09 %	-3.55 %	-3.64 %	-3.67 %	-3.61 %	-3.93 %	-3.90 %
30	6.82 %	-1.05 %	-0.89 %	-2.99 %	-3.11 %	-3.84 %	-3.88 %
35	24.87 %	1.48 %	1.30 %	-0.68 %	-1.81 %	-3.28 %	-3.67 %
50	34.58 %	5.99 %	6.28 %	0.06 %	-0.41 %	-2.69 %	-3.21 %

Πίνακας Α΄.6: Αποστάσεις λύσεων παραλλαγών του αλγόριθμου Διακλάδωσης και Αποκοπής με ευρετικές επανακλήσεις από τις λύσεις του UH3-DP

μέγεθος	RES-U	RES-NU	EXT-U	EXT-NU	μέση τιμή
10	0.2	0.3	0.6	1.5	0.6
15	0.8	2.0	3.1	7.8	3.4
20	2.8	6.1	10.9	16.9	9.2
25	9.5	13.1	24.3	32.9	19.9
30	18.3	26.0	34.9	59.0	34.6
35	27.0	49.0	48.0	92.8	54.2
50	65.3	103.4	115.8	187.2	118.0
70	178.5	191.3	264.5	369.4	250.9
100	535.2	449.1	720.0	853.1	639.3
150	1968.5	2116.6	2871.8	2494.3	2362.8

Πίνακας Α΄.7: Μέσοι χρόνοι εκτέλεσης αλγόριθμου Διακλάδωσης και Αποκοπής με επανάκληση του ευρετικού UH3-DP (s)

μέγεθος	ES1	ES2	NK1	NK2
10	0.8	0.5	1.5	1.0
15	2.5	1.5	6.4	4.3
20	5.9	3.6	19.4	13.2
25	11.6	7.5	48.9	33.6
30	20.0	12.9	112.2	77.9
35	34.5	22.5	282.1	169.1
50	106.0	71.0	1,676.2	1,067.5
70	301.3	-	-	-
100	1,031.4	-	-	-
150	3,987.3	-	-	-

Πίνακας Α΄.8: Μέσοι χρόνοι εκτέλεσης εξελικτικών αλγορίθμων (s)

		εξελικτικός αλγόριθμος			
σενάριο	μέγεθος	ES1	ES2	NK1	NK2
RES-U	10	-0.07 %	0.00 %	0.00 %	0.00 %
	15	-0.66 %	-0.18 %	-0.12 %	-0.12 %
	20	-0.66 %	-0.21 %	0.00 %	0.00 %
	25	-0.62 %	-0.01 %	-0.01 %	-0.01 %
	30	-0.24 %	-0.04 %	0.00 %	-0.04 %
	35	-0.30 %	-0.01 %	0.00 %	0.00 %
	50	-0.21 %	-0.12 %	-0.02 %	-0.03 %
EXT-U	10	-0.95 %	-0.36 %	-0.14 %	-0.17 %
	15	-0.69 %	-0.15 %	-0.15 %	-0.15 %
	20	-0.27 %	-0.08 %	0.00 %	0.00 %
	25	-0.63 %	-0.41 %	-0.39 %	-0.35 %
	30	-0.64 %	-0.32 %	-0.17 %	-0.22 %
	35	-0.60 %	-0.40 %	-0.20 %	-0.14 %
	50	-0.76 %	-0.47 %	-0.20 %	-0.22 %
RES-NU	10	-2.33 %	-1.73 %	-1.18 %	-1.19 %
	15	-3.22 %	-1.93 %	-1.78 %	-1.53 %
	20	-2.89 %	-2.06 %	-1.12 %	-1.32 %
	25	-2.29 %	-1.95 %	-0.93 %	-1.05 %
	30	-2.41 %	-1.81 %	-0.90 %	-0.80 %
	35	-2.49 %	-1.46 %	-0.52 %	-0.60 %
	50	-2.10 %	-1.18 %	-0.22 %	-0.38 %
EXT-NU	10	-2.10 %	-1.15 %	-0.79 %	-1.07 %
	15	-2.30 %	-1.82 %	-1.59 %	-1.58 %
	20	-2.99 %	-2.72 %	-1.66 %	-1.70 %
	25	-2.31 %	-1.41 %	-0.70 %	-1.03 %
	30	-2.81 %	-1.85 %	-0.98 %	-1.33 %
	35	-2.94 %	-2.18 %	-1.22 %	-1.00 %
	50	-3.04 %	-2.25 %	-0.72 %	-0.99 %

Πίνακας Α΄.9: Αποστάσεις λύσεων εξελικτικών αλγορίθμων από τις λύσεις του UH3-DP