Συγκριτική Μελέτη Μεταευρετικών Αλγορίθμων Βελτιστοποίησης για Υπολογισμό Κυκλικών Πινάκων Στάθμισης

Κωνσταντίνος Καλτσάς

Master Thesis

– ♦ –

Ιωάννινα, Φεβρουάριος 2015

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον κύριο Κωνσταντίνο Παρσόπουλο ο οποίος με καθοδήγησε σε όλη την διάρκεια των μεταπτυχιακών σπουδών μου. Τόσο με τα μαθήματα που δίδαξε, τα οποία με έκαναν να επιλέξω να ασχοληθώ με τον κλάδο της Βελτιστοποίησης, όσο και κατά την διάρκεια της υλοποίησης και συγγραφής της παρούσας εργασίας, όπου η βοήθεια που μου παρείχε ήταν καταλυτική στην άρτια έκβασή της.

Επίσης θα ήθελα να ευχαριστήσω τους κκ. Ι. Λαγαρή και Η. Κοτσιρέα για τα εποικοδομητικά σχόλιά τους.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

ΠΕΡΙΛΗΨΗ

Κωνσταντίνος Καλτσάς του Γεωργίου και της Άννας. MSc, Τμήμα Μηχανικών Η/Υ & Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Φεβρουάριος, 2015. Συγκριτική Μελέτη Μεταευρετικών Αλγορίθμων Βελτιστοποίησης για Υπολογισμό Κυκλικών Πινάκων Στάθμισης. Επιβλέπων: Κωνσταντίνος Παρσόπουλος

Στην παρούσα διατριβή γίνεται εκτενής μελέτη της απόδοσης πέντε μεταευρετικών αλγορίθμων στην επίλυση προβλημάτων εύρεσης κυκλικών πινάκων στάθμισης (CW matrices), καθώς και σύγκριση των αποτελεσμάτων που προέκυψαν.

Οι αλγόριθμοι που μελετήθηκαν είναι οι :

(1) Tabu Search
(2) Tabu Search με Path Relinking
(3) Variable Neighborhood Search
(4) Iterated Gradient Descent
(5) Random Search

Τα προβλήματα που επιλύθηκαν είναι τα *CW(n,k)* για $n \in [1,101]$ και $k \in [1,10]$. Για την επίλυση των προβλημάτων αυτών χρησιμοποιήθηκαν οι αλγόριθμοι που αναφέρθηκαν παραπάνω, οι οποίοι αναλύθηκαν ως εξής: για κάθε αλγόριθμο υπολογίστηκαν στατιστικά στοιχεία όπως το πλήθος επιτυχιών, ο μέσος απαιτούμενος χρόνος, και η τυπική απόκλιση. Σύμφωνα με αυτά έγινε στη συνέχεια σύγκριση των αλγορίθμων, και προέκυψαν ενδιαφέροντα συμπεράσματα, τα οποία θα αναλύσουμε σε επόμενα κεφάλαια.

ABSTRACT

Konstantinos Kaltsas, MSc, Department of Computer Science & Engineering, University of Ioannina, Greece. February, 2015. A Comparative Study of Metaheuristic Optimization Algorithms for the Computation of Circulant Weighing Matrices. Thesis Supervisor: Konstantinos Parsopoulos.

In this master thesis I study the performance of five metaheuristc algorithms in solving problems of Circulant Weighing (CW) matrices. Also I conduct comparisons of the algorithms based on the obtained results.

The algorithms that were implemented and studied are the following:

(1) Tabu Search
(2) Tabu Search με Path Relinking
(3) Variable Neighborhood Search
(4) Iterated Gradient Descent
(5) Random Search

The problems that were solved are of the type CW(n,k) for $n \in [1,101]$ and $k \in [1,10]$. In order to solve these problems, the aforementioned algorithms were employed. For each algorithm statistics, such as the number of success, average time and standard deviation were recorded. Based on these, comparisons of the algorithms were conducted, deriving useful conclusions.

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

Στην παρούσα εργασία μελετάται η αποδοτικότητα ενός αριθμού μεταευρετικών αλγορίθμων βελτιστοποίησης για την επίλυση ενός συνόλου προβλημάτων, των οποίων γνωρίζουμε την ύπαρξη λύσης, καθώς και η σύγκριση και ανάλυση αυτών. Στόχος είναι να κατανοήσουμε βαθύτερα ποιές τεχνικές είναι πιο κατάλληλες και αποδοτικές για τα συγκεκριμένα προβλήματα.

Τα προβλήματα που θα επιχειρήσουμε να λύσουμε χαρακτηρίζονται ως πολύπλοκα, καθώς παράγονται από συνδυαστικούς πίνακες. *Συνδυαστικοί* ονομάζονται οι πίνακες οι οποίοι παρουσιάζουν συγκεκριμένες συνδυαστικές ιδιότητες, όπως, π.χ. σταθερά αθροίσματα γραμμών ή στηλών ή δομικά χαρακτηριστικά όπως οι *κυκλικοί πίνακες*. Οι τελευταίοι έχουν ευρύ φάσμα εφαρμογών σε διάφορους τομείς της Πληροφορικής όπως είναι η Θεωρία Κωδίκων, οι Τηλεπικοινωνίες, οι Κβαντικοί υπολογιστές, η Κρυπτογραφία, η Επεξεργασία Σήματος και Εικόνας, κ.α. Για το λόγο αυτό, προκύπτει μια πληθώρα προβλημάτων με πολλές μεταβλητές, τα οποία παρουσιάζουν δυσκολία στην επίλυσή τους.

Η εύρεση συνδυαστικών πινάκων, καθώς και συγκεκριμένες κατηγορίες αυτών, απασχολεί την επιστημονική κοινότητα εδώ και πολλά χρόνια. Για την επίλυση τέτοιου είδους προβλημάτων έχουν υλοποιηθεί αλγόριθμοι που βασίζονται σε τεχνικές από διάφορους κλάδους των Μαθηματικών, όπως η Θεωρία Ομάδων, η Θεωρία Αριθμών και η Γραμμική Άλγεβρα. Το φάσμα αλγορίθμων που θα μας απασχολήσει εδώ είναι αλγόριθμοι που βασίζονται σε τεχνικές βελτιστοποίησης και πιο συγκεκριμένα, σε μεταευρετικές τεχνικές βελτιστοποίησης.

Οι αλγόριθμοι οι οποίοι υλοποιήθηκαν και χρησιμοποιήθηκαν για την επίλυση των προβλημάτων ανήκουν στους μεταευρετικούς αλγορίθμους. Ο όρος *μεταευρετικοί* (metaheuristics), αναφέρεται σε μεθόδους επίλυσης που κάνουν δυνατή μια αλληλεπίδραση μεταξύ των τοπικών διαδικασιών βελτίωσης και των υψηλότερου επιπέδου στρατηγικών. Ως αποτέλεσμα, ο αλγόριθμος μπορεί να ξεφύγει από μια τοπική λύση, και να διεξάγει μια αναζήτηση περισσότερο ανεξάρτητη από εκείνη που πραγματοποιούν οι ευρετικοί αλγόριθμοι. Εξαπλώνεται, δηλαδή, σε μία ευρύτερη περιοχή για την αναζήτηση των βέλτιστων λύσεων. Οι μεταευρετικοί αλγόριθμοι

χρησιμοποιήθηκαν και ως μέθοδοι που επέτρεψαν τη μεταφορά αναζήτησης λύσης από το τοπικό, σε κάποια άλλη γειτονιά. Μια συστηματικότερη και ειδικότερη ανάλυση όλων των αλγορίθμων που χρησιμοποιήθηκαν στην έρευνα θα επιχειρηθεί στην συνέχεια, όπου θα παρουσιαστεί και η δομή και λειτουργία τους. Εκεί θα παρουσιάσουμε τα χαρακτηριστικά της Tabu Search και πως η χρήση μνήμης βοηθά στην αποφυγή τοπικών ελαχίστων, την δημιουργία καινούριων λύσεων από ήδη γνωστές καλές λύσεις με την Path Relinking, την Iterated Gradient Descent και κατά πόσο η χρήση μνήμης βοηθά στην εύρεση μονοπατιών με κατεύθυνση προς μικρότερες τιμές, τεχνικές συστηματικής αλλαγής γειτονιάς με την Variable Neighborhood Search. Τέλος θα αναφερθούν τα χαρακτηριστικά της απλής τυχαίας αναζήτησης για λόγους τυπικούς, καθώς κάθε μεταευρετική μέθοδος για να έχει λόγο ύπαρξης θα πρέπει να επιτυγχάνει απόδοση τουλάχιστον ίση με αυτήν.

Την έρευνά μας θα απασχολήσουν οι κυκλικοί πίνακες στάθμισης (). Μία κατηγορία συνδυαστικών πινάκων που απασχολεί έντονα την επιστημονική κοινότητα. Θα παρουσιαστούν τα χαρακτηριστικά αυτών και πως μπορούν να οριστούν με την χρήση περιοδικών συναρτήσεων αυτοσυσχέτισης. Με βάση αυτά, θα περιγραφεί η διαδικαδία μοντελοποίησης του προβλήματος εύρεσης κυκλικών πινάκων στάθμισης.

Στην συνέχεια αναλύεται η διαδικασία εκτέλεσης πειραμάτων για κάθε αλγόριθμο, καθώς και τα αποτελέσματα που προέκυψαν. Για κάθε έναν παρουσιάζονται στοιχεία όπως το πλήθος επιτυχιών, ο μέσος απαιτούμενος χρόνος και η τυπική απόκλιση. Όλα τα στοιχεία που προέκυψαν παρατίθενται σε μορφή πινάκων και γραφημάτων στο παράρτημα της διατριβής. Με χρήση αυτών έγινε εκτενής μελέτη και σύγκριση των αλγορίθμων και προέκυψαν ενδιαφέροντα συμπεράσματα τα οποία αναλύονται στα επόμενα κεφάλαια. Τέλος θα γίνει κατάταξη των αλγορίθμων για κάθε πρόβλημα που ερευνήσαμε τόσο με βάση των αριθμό των επιτυχιών όσο και με βάση τον χρόνο επίλυσης. Με βάση αυτά θα επιχειρηθεί να γίνει και μία γενικότερη κατάταξη των αλγορίθμων όσον αφορά την αποδοτικότητά τους και την ταχύτητα επίλυσης.

ΚΕΦΑΛΑΙΟ 2. ΚΥΚΛΙΚΟΙ ΠΙΝΑΚΕΣ ΣΤΑΘΜΙΣΗΣ

Στις επόμενες παραγράφους περιγράφονται οι κυκλικοί πίνακες στάθμισης, οι περιοδικές συναρτήσεις αυτοσυσχέτισης και η διαδικασία μοντελοποίησης του προβλήματος εύρεσης κυκλικών πινάκων στάθμισης.

2.1. Κυκλικοί Πίνακες Στάθμισης

Ένας *πίνακας στάθμισης* $W = W(n, k)$, διάστασης n και βάρους k, είναι ένας τετραγωνικός πίνακας W διάστασης $n \times n$, με τιμές $w_{i,j} \in \{-1, 0, +1\}$ τέτοιος ώστε:

$$WW^T = kI_n$$

όπου W^T είναι ο ανάστροφος του W και I_n είναι ο $n \times n$ ταυτοτικός πίνακας.

Ένας *κυκλικός πίνακας στάθμισης*, (Circulant Weighing matrix) [3] με συμβολισμό $CW(n, k)$, είναι μια ειδική περίπτωση πίνακα στάθμισης, στην οποία κάθε γραμμή (εκτός της πρώτης) αποτελεί μια κυκλική μετατόπιση προς τα δεξιά κατά μια θέση της προηγούμενης γραμμής.

Ένα βασικό παράδειγμα πίνακα κυκλικής στάθμισης είναι ο I_n:

$$I_1 = \begin{bmatrix}1\end{bmatrix},\ I_2 = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ I_3 = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix},\ \cdots,\ I_n = \begin{bmatrix}1 & 0 & 0 & \cdots & 0\\ 0 & 1 & 0 & \cdots & 0\\ 0 & 0 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & 1\end{bmatrix}$$

Ένα άλλο παράδειγμα πίνακα κυκλικής στάθμισης είναι το ακόλουθο:

$$CW(7,4) = \begin{pmatrix} -1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & -1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & -1 \end{pmatrix}.$$

2.2. Περιοδικές Συναρτήσεις Αυτοσυσχέτισης

Ένας μεγάλος αριθμός συνδυαστικών πινάκων, καθώς και διάφορες κατηγορίες αυτών, μπορούν να οριστούν με τη χρήση των *Περιοδικών Συναρτήσεων Αυτοσυσχέτισης* PAF [3]. Αυτές οι συναρτήσεις μελετήθηκαν πολύ και η σημαντικότητά τους έχει αναγνωριστεί από την επιστημονική κοινότητα που ασχολείται με θέματα Συνδυαστικής, ειδικά για τις περιπτώσεις χρήσης τους στους συνδυαστικούς πίνακες.

Η PAF για μια ακολουθία A=($a_1 a_2 \ldots a_n$) διάστασης *n*, όριζεται ως ακολούθως:

$$\mathrm{PAF_A}(s) = \sum_{i=1}^{n} a_i\, a_j \ , \ i = 0, \ldots, n\text{-}1$$

όπου,

$$j = (i + s) \bmod n$$

Η συσχέτιση που υπάρχει ανάμεσα στις PAF και στους κυκλικούς πίνακες μπορεί να γίνει πιο κατανοητή αν θεωρήσουμε ένα διάνυσμα A και έναν κυκλικό πίνακα, του οποίου η πρώτη γραμμή είναι το A. Τότε ισχύει ότι το $\mathrm{PAF_A}(i)$ είναι ίσο με το εσωτερικό γινόμενο της πρώτης γραμμής του πίνακα με την i+1 γραμμή του.

2.3. Μοντελοποίηση Προβλήματος Εύρεσης Κυκλικών Πινάκων Στάθμισης

Ένα χαρακτηριστικό που έχουν οι κυκλικοί πίνακες στάθμισης είναι ότι οι συνιστώσες κάθε γραμμής του πίνακα παίρνουν 0, 1 και -1. Έτσι, το πρόβλημα εύρεσης ενός κυκλικού πίνακα *CW (n, k)* ανάγεται στην εύρεση ενός διανύσματος (ακολουθίας), οι συνιστώσες του οποίου παίρνουν τιμές 0, 1, -1:

$$A=(a_1,a_2,...a_n),$$

όπου,

$$a_i = 0 \text{ ή } +1 \text{ ή } -1, \ \forall i = 1, \ldots, n,$$

Ένα επιπλέον χαρακτηριστικό που έχουν οι κυκλικοί πίνακες στάθμισης είναι ότι σε κάθε γραμμή η συχνότητα εμφάνισης των 0, 1 και -1 είναι συγκεκριμένη και εξαρτάται από τον αντίστοιχο πίνακα που ψάχνουμε να βρούμε. Συγκεκριμένα το πλήθος των αριθμών αυτών ορίζεται από τις σχέσεις που ακολουθούν. Συνεπώς, οι οποίες καθορίζουν και τους περιορισμούς του προβλήματος που καλούμαστε να επιλύσουμε:

$$\text{πλήθος των } 0 \ : \ n - k^2,$$
$$\text{πλήθος των } +1 : \ k \ \frac{(k+1)}{2},$$
$$\text{πλήθος των } -1 \ : \ k \ \frac{(k-1)}{2}$$

Επιπλέον, μία βασική ιδιότητα των κυκλικών πινάκων στάθμισης που θα μας απασχολήσουν είναι ότι τα (PAF) που ορίσαμε παραπάνω έχουν όλα σταθερό άθροισμα:

$$PAF_A(s) = \sum_{i=1}^{n} a_i \, a_{i} = 0 , \quad \forall i = 1, \ldots, \left[\frac{n}{!}\right] , \qquad (1)$$

όπου,

$$j = (i + s) \ mod \ n ,$$

και

$$\left[\frac{n}{!}\right] = \begin{cases} \frac{n}{!}\,, & \alpha\nu!!\ \ \acute{\alpha}\rho\tau\iota o\varsigma\,, \\ \frac{n-1}{2}\,, & \alpha\nu\ n\ \pi\varepsilon\rho\iota\tau\tau\acute{o}\varsigma. \end{cases}$$

Έστω $s \in [1, \frac{n}{2}]$, τότε η ακολουθία-λύση πρέπει να μηδενίζει όλες τις s εξισώσεις που ορίζει η Σχέση (1).

Αυτό μπορεί να γραφτεί ισοδύναμα ως πρόβλημα συνδυαστικής βελτιστοποίησης, όπου καλούμαστε να βρούμε την κατάλληλη ακολουθία A που ελαχιστοποιεί την αντικειμενική συνάρτηση:

$$\mathrm{F(A)} = \sum_{s=1}^{\left[\frac{n}{2}\right]} |PAF_A(s)|.$$

Προφανώς για μία λύση A^* θα ισχύει ότι $\mathrm{F}(\mathrm{A}^*) = 0$.

ΚΕΦΑΛΑΙΟ 3. ΜΕΤΑΕΥΡΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ

Στις επόμενες παραγράφους περιγράφονται οι μεταευρετικοί αλγόριθμοι βελτιστοποίησης που χρησιμοποιήθηκαν στην παρούσα μελέτη.

3.1. Tabu Search

Βασική αρχή της Tabu Search (TS) αποτελεί η συνέχεια της Local Search (τοπική αναζήτηση) όταν συναντά ένα τοπικό ελάχιστο (ή μέγιστο), επιτρέποντας τις κινήσεις μη-βελτίωσης της λύσης. Η επιστροφή σε λύσεις που έχουμε ήδη επισκεφθεί, εμποδίζεται με τη χρήση μιας μνήμης που ονομάζεται tabu list. Η tabu list διατηρεί ένα ιστορικό των πρόσφατων λύσεων που έχουμε επισκεφτεί.

Συμβολισμοί

- S , η τρέχουσα υποψήφια λύση
- S^* , η καλύτερη λύση που έχουμε βρει μέχρι στιγμής
- f^* , η τιμή της S^*
- $N(S)$, η γειτονιά της S
- $\tilde{N}(S)$, το αποδεκτό υποσύνολο της N(S) (π.χ. χωρίς τις λύσεις της tabu list ή περιορισμών του προβλήματος)
- T , η tabu list

Βασικά βήματα του αλγορίθμου

1. **Αρχικοποίηση**
   Επιλογή μιας αρχικής λύσης
   $S \leftarrow S_0$, $f^* \leftarrow f(S_0)$, $S^* \leftarrow S_0$, $T \leftarrow \varnothing$

2. **Αναζήτηση**
   Όσο δεν ικανοποιούνται τα κριτήρια τερματισμού επανέλαβε
   select S in $argmin_{S' \in \tilde{N}(S)}[f(S')]$
   if $f(S) < f^*$, thenset $f^* \leftarrow f(S)$, $S^* \leftarrow S$;
   record tabu for the current move in T
   (delete oldest entry if necessary)

Κριτήρια τερματισμού:

(1) Μετά από ένα σταθερό αριθμό επαναλήψεων του αλγορίθμου ή συναρτησιακών υπολογισμών ή CPU time

(2) Μετά από έναν αριθμό επαναλήψεων χωρίς βελτίωση της καλύτερης ευρεθείσας τιμής της αντικειμενικής συνάρτησης

(3) Όταν ο αλγόριθμος βρει τιμή που έχει τεθεί από την αρχή ως τιμή τερματισμού

Αναλυτική Περιγραφή

Το βασικό χαρακτηριστικό της TS είναι η χρήση μνήμης για την αποφυγή της επίσκεψης λύσεων που έχουν ήδη βρεθεί. Όπως σε όλους τους αλγορίθμους, έτσι και στην TS ξεκινάμε με την αρχικοποίηση του αλγορίθμου σε μια τυχαία αρχική συνθήκη-διάνυσμα, η οποία πρέπει να πληρεί τις βασικές απαιτήσεις του προβλήματος. Στην περίπτωσή μας, βασική απαίτηση είναι ο συγκεκριμένος αριθμός των 0, 1 και -1, όπως προβλέπονται από τη μοντελοποίηση του προβλήματος που περιγράφηκε προηγουμένως (σελ.5). Σε κάθε περίπτωση επιλογής τυχαίων αριθμών χρησιμοποιούμε μία γεννήτρια τυχαίων αριθμών, η οποία παίρνει σαν παράμετρο τον χρόνο του επεξεργαστή, αποφεύγοντας έτσι την επανάληψη ίδιων αριθμών.

Κατά την αρχικοποίηση του αλγορίθμου θέτουμε την αρχική λύση που πήραμε ως την καλύτερη ως τώρα, *xbest*, και ως την καλύτερη νέα, *xnew*. Στο σημείο

αυτό είναι σημαντικό να διευκρινίσουμε τη διαφορά ανάμεσα στα δύο αυτά διανύσματα για αποφυγή σύγχυσης στην συνέχεια της ανάλυσης του αλγορίθμου. Το διάνυσμα *xbest* αντιστοιχεί στην καλύτερη λύση που βρήκε ως τώρα ο αλγόριθμος μετά από το σύνολο των επαναλήψεων που έχουν ήδη γίνει. Το διάνυσμα *xnew* αντιστοιχεί στην καλύτερη λύση που μας έδωσε ο αλγόριθμος στην τρέχουσα επανάληψη, η οποία μπορεί να αλλάξει στην επόμενη, και δεν είναι η καλύτερη x που έχει βρει ο αλγόριθμος γενικά, αλλά η καλύτερη της τρέχουσας επανάληψης και μόνο.

Η εύρεση της λύσης του προβλήματος (*xbest*) βασίζεται στην παραγωγή μεταθέσεων της τρέχουσας λύσης που έχει βρει ο αλγόριθμος (*xnew*). Σε κάθε επανάληψη του αλγορίθμου βρίσκουμε την μετάθεση που έχει δύο μόλις διαφορετικές συνιστώσες της τρέχουσας καλύτερης τιμής (*xnew*), δεν βρίσκεται στην tabu list, και έχει την καλύτερη συναρτησιακή τιμή ανάμεσα σε όλες τις μεταθέσεις που πληρούν αυτό τον περιορισμό. Το διάνυσμα αυτό πλέον αποτελεί την τρέχουσα καλύτερη λύση του αλγορίθμου (*xnew*).

Στο σημείο αυτό, μετά την εύρεση της νέας καλύτερης λύσης (*xnew*), αφαιρούμε από την tabu list την παλιότερη λύση που έχουμε εισάγει και την αντικαθιστούμε από την νέα *xnew*. Με την χρήση της tabu list, ο αλγόριθμος έχει την δυνατότητα να αποφύγει τοπικά ελάχιστα, χωρίς να εμφανίζει κυκλική συμπεριφορά, ώστε να επεκτείνει την αναζήτηση και σε γειτονιές άλλων λύσεων.

Συνεπώς γίνεται πολύ εύκολα κατανοητό ότι το μέγεθος της tabu list που θα επιλεγεί, παίζει πολύ σημαντικό ρόλο στην βαρύτητα που θα δίνει ο αλγόριθμος στην αναζήτηση νέων λύσεων στην ίδια γειτονιά ή στην αναζήτηση νέων. Αυξάνοντας το μέγεθος της tabu list, αποφεύγουμε για περισσότερες επαναλήψεις γειτονιές που έχουμε ήδη επισκεφθεί. Σε αντίθετη περίπτωση, δίνουμε τη δυνατότητα στον αλγόριθμο να μένει για περισσότερες επαναλήψεις στην ίδια γειτονιά και να ψάχνει τοπικά, διακινδυνεύοντας, με τον τρόπο αυτό, να εγκλωβιστούμε σε τοπικό ελάχιστο και να έχουμε κυκλική συμπεριφορά.

Μετά την αφαίρεση από την tabu list της παλαιότερης καταχώρησής της, εισάγεται η *xnew*, ο αλγόριθμος ελέγχει πλέον αν η τρέχουσα καλύτερη τιμή (*xnew*) έχει και καλύτερη συναρτησιακή τιμή από την τιμή της συνολικά καλύτερης λύσης *xbest* που έχει βρει. Στην περίπτωση που συμβαίνει αυτό, θέτουμε ως νέα *xbest* την τρέχουσα *xnew*.

Ο αλγόριθμος σταματά τις επαναλήψεις και μας επιστρέφει τη λύση που βρήκε, είτε όταν βρει την βέλτιστη λύση *xbest* που έχει συναρτησιακή τιμή 0, είτε όταν ο αριθμός των επαναλήψεων (ή των συναρτησιακών υπολογισμών) που έχει κάνει είναι ίσος με το μέγιστο όριο που έχουμε ορίσει κατά την υλοποίηση. Σε αυτό το σημείο, δύο σημαντικά πράγματα οφείλουν να επισημανθούν αναφορικά με την λειτουργία του αλγορίθμου. Το πρώτο και σημαντικότερο, το οποίο καθιστά την TS πολύ αποδοτική (τουλάχιστον στα προβλήματα Συνδυαστικής Βελτιστοποίησης που μελετάμε εδώ), είναι το γεγονός ότι η τρέχουσα καλύτερη λύση του αλγορίθμου (*xnew*) στην επανάληψη *i*+1 δεν συνεπάγεται ότι είναι καλύτερη από την *xnew* της προηγούμενης επανάληψης *i* του αλγορίθμου. Αυτό είναι πολύ σημαντικό για προβλήματα με πολλά τοπικά ελάχιστα, καθώς επιτρέπει στην TS να ψάχνει σε νέες γειτονιές. Έτσι, η TS καθίσταται ένας πολύ αποδοτικός αλγόριθμος και για αναζητήσεις σε μεγάλους χώρους αναζήτησης και γειτονιές με πολλά τοπικά ελάχιστα. Για το λόγο αυτό χρησιμοποιείται και σε συνδυασμό με άλλους αλγορίθμους, όπως θα δούμε και στη συνέχεια αυτής της μελέτης.

Η δεύτερη επισήμανση που πρέπει να γίνει αφορά στην υλοποίηση του αλγορίθμου, για να αποφύγουμε την περίπτωση συνεχούς εύρεσης λύσεων που δεν βελτιώνουν την *xbest*. Συγκεκριμένα, χρησιμοποποιήσαμε έναν μηχανισμό επανακκίνησης, ο οποίος παίρνει μια νέα τυχαία λύση ως *xnew*, στην περίπτωση που ο αλγόριθμος κάνει έναν αριθμό από επαναλήψεις χωρίς να βρει καλύτερη *xbest*. Ο αριθμός αυτός είναι σημαντικός για την TS διότι την καθιστά ικανή να ξεκινήσει αναζήτηση του χώρου όταν έχει “εγκλωβιστεί” σε περιοχές που δεν δίνουν καλύτερες λύσεις.

Έτσι, κάθε φορά που ο αλγόριθμος βρίσκει μια καλύτερη *xbest*, ξεκινάμε να μετράμε από την αρχή τις μη-βελτιώσεις της *xbest*, διαφορετικά συνεχίζουμε τις επαναλήψεις με τον τρόπο λειτουργίας του αλγορίθμου, μετρώντας τις επαναλήψεις της μη-βελτίωσης και παίρνοντας τυχαία *xnew*, όταν ο αριθμός γίνει ίσος με τον αριθμό που έχουμε ορίσει.

Η τιμή που θα ορίσουμε παίζει σημαίνοντα ρόλο στην λειτουργία του αλγορίθμου διότι, στην περίπτωση που είναι ιδιαίτερα μικρή, θα επανεκκινούμε πολύ συχνά τον αλγόριθμο, περιορίζοντας την αναζήτηση σε μικρές περιοχές/γειτονιές. Στην περίπτωση που είναι πολύ μεγάλη, δίνουμε την δυνατότητα στον αλγόριθμο να ψάχνει με περισσότερες επαναλήψεις, διαγράφοντας μεγαλύτερες τροχιές στο χώρο.

Παραμετροποίηση και ιδιαιτερότητες του αλγορίθμου

Όπως αναφέραμε ήδη στην περιγραφή του αλγορίθμου, η TS έχει κάποιες παραμέτρους για τις οποίες απαιτείται μια προεργασία ώστε να βρεθούν οι κατάλληλες τιμές τους.

Για την απλούστερη περιγραφή και κατανόηση της παραπάνω διαδικασίας, παρατίθενται παρακάτω οι παράμετροι της TS:

(1) Ο μέγιστος αριθμός επαναλήψεων του αλγορίθμου (iterations_max)
(2) Το μέγεθος της tabu list (list_length)
(3) Μέγιστος αριθμός επαναλήψεων μη βελτίωσης της καλύτερης λύσης *xbest* (iterations_reset)

Για την εύρεση των καταλληλότερων τιμών των παραμέτρων, πραγματοποιήθηκε ένας αριθμός πειραμάτων για διάφορες τιμές τους με σκοπό να εκτιμηθεί η απόδοση του αλγορίθμου για καθεμιά.

Επειδή ο αριθμός των προβλημάτων ήταν σχετικά μεγάλος (194 προβλήματα), η διαδικασία εύρεσης των κατάλληλων παραμέτρων ήταν χρονοβόρα, διότι κάθε πρόβλημα αντιμετωπίστηκε ανεξάρτητα. Για παράδειγμα μπορούμε να σκεφτούμε ότι δεν θα ήταν αποδοτικό να έχουμε μεγάλο μέγεθος list_length για ένα πρόβλημα μικρής διάστασης, π.χ. το *CW*(2,1), διότι το σύνολο των διαθέσιμων λύσεων σε μια γειτονιά θα βρισκόταν διαρκώς στην tabu list, με συνέπεια να μην λειτουργεί αποδοτικά ο αλγόριθμος. Το αντίστροφο θα συνέβαινε σε ένα πρόβλημα μεγάλης διάστασης, π.χ. στο *CW*(100,2). Δηλαδή δεν θα έπρεπε να χρησιμοποιήσουμε μικρό list_length, διότι ο αριθμός τον διαθέσιμων μεταθέσεων θα ήταν μεγάλος με συνέπεια ενδεχόμενη κυκλική κίνηση του αλγορίθμου.

Η διαδικασία εύρεσης των κατάλληλων παραμέτρων για την TS ήταν η ακόλουθη. Για το μέγιστο αριθμό επαναλήψεων του αλγορίθμου (iterations_max), έγιναν πειράματα με 1000, 10000 και 100000 επαναλήψεις. Τα πειράματα έδειξαν ότι οι 10000 επαναλήψεις ήταν ικανοποιητικός αριθμός, τις περισσότερες φορές ήταν αρκετά μεγαλύτερος από τις αναγκαίες επαναλήψεις. Για την συγκεκριμένη παράμετρο δεν υπήρχε λόγος να γίνουν ξεχωριστά πειράματα ανά πρόβλημα καθώς,

αν ο αλγόριθμος έβρισκε την λύση, θα σταματούσε σε κάθε περίπτωση τις επαναλήψεις και δεν θα χρησιμοποιούσε τις υπόλοιπες διαθέσιμες. Έτσι προτιμήθηκε να γίνουν πειράματα σε προβλήματα με μεγαλύτερες διαστάσεις και δυσκολία επίλυσης και κρατήθηκε η τιμή που ήταν πιο αποδοτική σε αυτά.

Για την δεύτερη παράμετρο (list_length), τα δεδομένα ήταν τελείως διαφορετικά. Όπως αναφέρθηκε παραπάνω (φέρνοντας και το παράδειγμα με τις μικρές και μεγάλες διαστάσεις και πως επηρεάζει αυτό το μέγεθος που πρέπει να έχει η tabu list) δεν θα μπορούσαμε να έχουμε την ίδια τιμή για το μέγεθος της μνήμης που χρησιμοποιούμε σε όλα τα προβλήματα.

Κατά συνέπεια, για αυτή την παράμετρο απαιτήθηκαν τα περισσότερα πειράματα. Πιο συγκεκριμένα, για κάθε ένα από τα προβλήματα που ερευνήθηκαν, έγιναν πειράματα θέτοντας την τιμή της list_length από 1 έως την διάσταση του προβλήματος. Για να είναι πιο αξιόπιστα τα αποτελέσματα, έγιναν 5 επαναλήψεις για κάθε μια τιμή της list_length που δοκιμάστηκε, και κρατήθηκε ο μέσος όρος των επιτυχιών κάθε φορά. Στην περίπτωση μη επιτυχίας, αποθηκευόταν το σφάλμα (δηλαδή η απόκλιση από την τιμή 0 που είναι η συναρτησιακή τιμή της λύσης του προβλήματος). Έτσι, ακόμα και στις περιπτώσεις που δεν εντοπιζόταν λύση, διατηρούνταν οι περισσότερα υποσχόμενες τιμές των παραμέτρων.

Η διαδικασία αυτή ήταν πιο χρονοβόρα από την διαδικασία εύρεσης των παραμέτρων και εμφάνισε τα περισσότερα προβλήματα. Ένα βασικό πρόβλημα το οποίο έπρεπε να αντιμετωπιστεί, ήταν η κατάλληλη υλοποίηση του αλγορίθμου, έτσι ώστε να καταγράφει τις περιπτώσεις όπου η tabu list ήταν πολύ μεγάλη για το συγκεκριμένο πρόβλημα.

Μετά το πέρας αυτής της διαδικασίας και την μελέτη των αποτελεσμάτων, φάνηκε ότι η TS είχε απόδοση άμεσα συσχετιζόμενη με την διάσταση του προβλήματος. Συγκεκριμένα, έδινε τις καλύτερες τιμές στις περιπτώσεις όπου η list_length (το μέγεθος της μνήμης) ήταν ίση με το μέγεθος της διάστασης του προβλήματος.

Η τελευταία παράμετρος για την οποία εκπονήθηκαν πειράματα ήταν η iterations_reset, δηλαδή ο επιτρεπόμενος αριθμός διαδοχικών επαναλήψεων του αλγορίθμου χωρίς βελτίωση της *xbest*. Ο ίδιος τρόπος διεξαγωγής πειραμάτων που ακολουθήθηκε για την list_length, έλαβε χώρα και για την iterations_reset. Δηλαδή, έγιναν 5 πειράματα για κάθε πρόβλημα, με την iterations_reset να παίρνει τιμές 100, 500 και 1000, αλλά και χωρίς την χρήση παραμέτρου. Τα πειράματα έδειξαν ότι στις

περιπτώσεις όπου επανεκκινούσαμε τον αλγόριθμο σε τυχαία τιμή μετά από 100 επαναλήψεις χωρίς βελτίωση της *xbest*, ο αλγόριθμος έδινε καλύτερες λύσεις από εκείνες που έδινε προηγουμένως.

Συνοπτικά, οι καταλληλότερες τιμές των παραμέτρων που βρέθηκαν με την παραπάνω διαδικασία ήταν οι ακόλουθες:

(1) iterations_max = 10000
(2) list_length = dimension
(3) iterations_reset = 100

Αυτές οι τιμές υιοθετήθηκαν στην πειραματική μελέτη που θα παρουσιαστεί παρακάτω.

3.2. Tabu Search με Path Relinking

Η Path Relinking (PR) είναι μια τεχνική που μας επιτρέπει να εξερευνήσουμε τα μονοπάτια που συνδέουν καλές λύσεις που βρέθηκαν από ευρετικές μεθόδους. Με την PR, γίνεται αναζήτηση στο συντομότερο μονοπάτι ανάμεσα σε δύο ή περισσότερες ευρεθείσες λύσεις. Η καλύτερη λύση που θα βρεθεί στο μονοπάτι χρησιμοποιείται ως νέα αρχική συνθήκη στον αλγόριθμο. Η PR χρησιμοποιείται κατά κανόνα με αλγορίθμους όπως η TS και το Scatter Search. Στην περίπτωσή μας συνδυάστηκε επιτυχώς με την TS.

Αναλυτική Περιγραφή:

Έστω ένας μη-κατευθυνόμενος γράφος $G=(S,M)$ όπου οι κόμβοι S αντιστοιχούν σε λύσεις του προβλήματος και οι ακμές M στις κινήσεις στην δομή της γειτονιάς. Δηλαδή, $(i, j) \in M$ αν και μόνο αν $i \in S$, $j \in S$, $j \in N(i)$, και $i \in N(j)$, όπου $N(s)$ συμβολίζει την γειτονιά μια λύσης $s \in S$. Η PR τυπικά εφαρμόζεται μεταξύ δύο ευρεθείσων λύσεων. Η μία λύση ονομάζεται *initial solution* και η άλλη *guiding solution.* Ένα ή περισσότερα μονοπάτια του πεδίου των λύσεων που ενώνουν τις δύο αυτές λύσεις εξερευνώνται για καλύτερες λύσεις. Στις καλύτερες λύσεις των μονοπατιών εφαρμόζεται τοπική αναζήτηση (στην περίπτωσή μας TS), ώστε να εντοπιστεί η κοντινότερη τοπικά βέλτιστη λύση.

Έστω $s \in S$ βρίσκεται στο μονοπάτι μεταξύ των *initial* και *guiding solution.* Περιορίζουμε τις αποδεκτές λύσεις της γειτονιάς $N(s)$ του s που θα ακολουθήσουν το μονοπάτι, σε αυτές που μοιάζουν περισσότερο στην g (guiding solution) παρά στο s. Αυτό επιτυγχάνεται επιλέγοντας κατάλληλες κινήσεις από το s, που οδηγούν σε λύσεις κατά το δυνατόν πιο όμοιες με την *guiding solution*. Ως αποτέλεσμα, οι επιλεχθείσες κινήσεις δίνουν σε κάθε Βήμα λύσεις που υιοθετούν ένα χαρακτηριστικό της *guiding solution,* το οποίο δεν έχει το *s.*

Υπάρχουν πολλές εκδοχές της PR. Στην περίπτωσή μας έχουμε χρησιμοποιήσει την mixed PR (mPR). Σε αυτή, ανάμεσα σε δύο πιθανές λύσεις s και t ξεκινάμε δύο μονοπάτια ταυτόχρονα, ένα από την s και ένα από την t. Αυτά τα δύο μονοπάτια συναντώνται σε μία λύση $r \in S$, με αποτέλεσμα να συνδέονται τα s και t με ένα μονοπάτι. Η βασική δομή του αλγορίθμου για δυαδικά διανύσματα παρουσιάζεται παρακάτω. Οι απαραίτητες αλλαγές που έγιναν πάνω στον αλγόριθμο ώστε να προσαρμοστεί στα προβλήματα που επιλύσαμε, δίνονται αναλυτικά στην συνέχεια.

1 MixedPathRelinking
2 $\Delta \leftarrow \{j=1,...,n : x^s_j \neq x^t_j\}$;
3 $x^* \leftarrow \text{argmin}\{z(x^s), z(x^t)\}$;
4 $z^* \leftarrow \min\{z(x^s), z(x^t)\}$;
5 $x \leftarrow x^s$;
6 **while** $|\Delta|>1$ **do**
7 $l^* \leftarrow \text{argmin}\{z(x \oplus l) : l \in \Delta\}$;
8 $\Delta \leftarrow \Delta \setminus \{l^*\}$;
9 $x_l \leftarrow 1 - x_l$;
10 **if** $z(x) < z^*$ **then**
11 $x^* \leftarrow x$;
12 $z^* \leftarrow z(x)$;
13 **end**
14 $x^s \leftarrow x^t$;
15 $x^t \leftarrow x$;
16 **end**
17 $x \leftarrow \text{LocalSearch}(x)$;
18 **return** x ;

Αναλυτική Περιγραφή:

Όπως όλοι οι αλγόριθμοι, έτσι και στην περίπτωση της Tabu Search με Path Relinking (TS-PR) ξεκινάμε τον αλγόριθμό μας αρχικοποιώντας τον. Στην περίπτωση της mPR, που χρησιμοποιήθηκε, απαιτούνται δύο τυχαία αρχικά διανύσματα/λύσεις, και όχι ένα όπως στη απλή PR. Αυτά αντιστοιχούν στις λύσεις x^s και x^t που αναφέραμε πιο πάνω. Τα διανύσματα θα πρέπει οπωσδήποτε να πληρούν τις συνθήκες του προβλήματος. Στην περίπτωση των προβλημάτων επιλύουμε, οι βασικοί περιορισμοί είναι ο αριθμός των 0, 1 και -1 στις ακολουθίες, όπως αναφέραμε σε προηγούμενο κεφάλαιο. Στην περίπτωση που προκύψουν δύο ίδια διανύσματα, όπως είναι πιθανό σε προβλήματα σχετικά μικρής διάστασης, αρχικοποιείται εκ νέου η μια από τις δύο λύσεις. Όπως και στην απλή TS, σε κάθε περίπτωση επιλογής τυχαίων αριθμών χρησιμοποιείται μια γεννήτρια τυχαίων αριθμών, η οποία παίρνει ως παράμετρο τον χρόνο του επεξεργαστή, αποφεύγοντας έτσι την επανάληψη ίδιων αριθμών.

Όπως και στην TS, σε κάθε επανάληψη του αλγορίθμου βρίσκουμε την καλύτερη λύση της τρέχουσας επανάληψης, την οποία συμβολίζουμε με *xnew*. Κάθε φορά που βρίσκουμε μία καλύτερη λύση ενημερώνουμε την καλύτερη έως τώρα λύση του αλγορίθμου, την οποία ονομάζουμε *xbest*. Πιο συγκεκριμένα, μετά την αρχικοποίηση των δύο διανυσμάτων x^s και x^t, ο αλγόριθμος εντοπίζει τις συνιστώσες στις οποίες τα δύο διανύσματα διαφέρουν και αποθηκεύει τις αντίστοιχες θέσεις. Ας ονομάσουμε *error_num* τον αριθμό των διαφορών, και *error* το δυαδικό διάνυσμα που δείχνει τις θέσεις όπου τα δύο vectors x^s και x^t διαφέρουν. Όπως είναι αναμενόμενο, θέτουμε 1 στις θέσεις του *error vector* όπου τα x^s και x^t διαφέρουν και 0 εκεί όπου έχουν την ίδια τιμή. Με αυτό τον τρόπο, όπως αναφέρθηκε στην αρχή του κεφαλαίου, μπορούμε να βρούμε τις διαφορές ανάμεσα στις δύο λύσεις και να δημιουργήσουμε ένα μονοπάτι που τις ενώνει.

Αφού έχουμε βρει τον αριθμό και τις θέσεις των διαφορών ανάμεσα στις δύο λύσεις, ελέγχουμε ποια από τις δύο λύσεις έχει καλύτερη συναρτησιακή τιμή. Θέτουμε, λοιπόν, σαν *xnew* (καλύτερη λύση της τρέχουσας επανάληψης του αλγορίθμου) αυτή με την καλύτερη συναρτησιακή τιμή (x^s ή x^t). Στην περίπτωση που βρισκόμαστε στην πρώτη επανάληψη του αλγορίθμου, θέτουμε επίσης *xbest* (καλύτερη τιμή του αλγορίθμου) ίση με την συγκεκριμένη. Αν όχι, τότε ελέγχουμε κατά πόσο έχουμε βρει κάποια καλύτερη λύση από την *xbest* και στην περίπτωση

αυτή την ενημερώνουμε. Οποιαδήποτε στιγμή ο αλγόριθμος βρει λύση με τιμή μηδέν, σταματούν οι επαναλήψεις και ο αλγόριθμος επιστρέφει την λύση.

Το επόμενο βήμα του αλγορίθμου είναι το πιο σημαντικό καθώς και το πιο πολύπλοκο στην λειτουργία του. Πρόκειται για το τμήμα του αλγορίθμου που δημιουργεί τα δύο μονοπάτια από τα οποία θα προκύψει η νέα λύση από τις x^s και x^t που έχουμε ήδη.

Ο αλγόριθμος επαναλαμβάνει την ίδια διαδικασία όσο το *error_num* (διαφορά ανάμεσα στα x^s και x^t) συνεχίζει να είναι μεγαλύτερο του 2. Επειδή επιλέξαμε την mPR ο αλγόριθμος κάνει ένα βήμα διαδοχικά από την x^s και ένα από την x^t. Ο ακριβής τρόπος λειτουργίας περιγράφεται αναλυτικά στη συνέχεια.

Σε κάθε επανάληψη επίσης ελέγχουμε αν η νέα λύση που δημιουργήθηκε έχει καλύτερη συναρτησιακή τιμή από την βέλτιστη και ενημερώνουμε ανάλογα την *xbest*. Στην περίπτωση που η συναρτησιακή τιμή του νέου σημείου είναι 0, ο αλγόριθμος σταματά και επιστρέφει την λύση.

Αφού έχουμε βρει ένα νέο σημείο προς επίσκεψη από τις x^s και x^t, ενημερώνουμε την *xnew* και κάνουμε τοπική αναζήτηση χρησιμοποιώντας την TS (την οποία περιγράψαμε στο προηγούμενο κεφάλαιο). Μετά την εφαρμογή της TS θα έχουμε ένα τοπικό ελάχιστο της συγκεκριμένης περιοχής αναζήτησης (γειτονιά). Εάν η TS βρει την βέλτιστη λύση με συναρτησιακή τιμή 0, ο αλγόριθμος τερματίζεται και επιστρέφει την λύση, διαφορετικά επιστρέφει στην mPR και ενημερώνει την *xnew* με την νέα λύση που βρήκε.

Σε αυτό το σημείο έχει ολοκληρωθεί ένας πλήρης κύκλος του αλγορίθμου. Όμως πριν περάσουμε στον επόμενο κύκλο του αλγορίθμου γίνονται κάποιες απαραίτητες ενέργειες που είναι αναγκαίες.

Ο αλγόριθμος PR βασίζεται στην εύρεση νέων λύσεων από ήδη υπάρχουσες καλές λύσεις. Για τον λόγο αυτό, μετά το πέρας κάθε επανάληψης, καθοριζόταν ως x^t η καλύτερη λύση που βρήκε ο αλγόριθμος έως το σημείο εκείνο (*xbest*), ενώ ως x^s χρησιμοποιούνταν η καλύτερη λύση της τρέχουσας επανάληψης *xnew*. Τα πλεονεκτήματα και μειονεκτήματα αυτής της επιλογής αναλύονται στην συνέχεια. Στην περίπτωση που η *xbest* ήταν ίδια με την *xnew*, τότε ο αλγόριθμος εκκινούσε την x^t από την *xbest* και έπαιρνε ένα τυχαίο διάνυσμα ως x^s.

Τέλος, για την αποφυγή εγκλωβισμού σε τοπικά ελάχιστα, υιοθετήθηκε η ίδια στρατηγική όπως και στην TS. Δηλαδή ορίστηκε ένας μέγιστος αριθμός επιτρεπτών επαναλήψεων του αλγορίθμου χωρίς βελτίωση της καλύτερης λύσης *xbest*. Έτσι,

στην περίπτωση που ο αλγόριθμος δεν βρει καλύτερη λύση για τόσες συνεχόμενες επαναλήψεις όσες ο αριθμός που ορίσαμε, δεν ακολουθείται η διαδικασία της προηγούμενης παραγράφου, αλλά αρχικοποιούμε τα x^s και x^t προσδίδοντας στοχαστικότητα δίνοντάς τους τυχαίες τιμές που πληρούν τις συνθήκες του προβλήματος, όπως αυτές προβλέπονται από την μοντελοποίησή του που περιγράφηκε στην αρχή της διατριβής.

Ο αλγόριθμος συνεχίζει τις επαναλήψεις όσο ο αριθμός επαναλήψεων δεν ξεπερνά τον μέγιστο αριθμό και όσο η συναρτησιακή τιμή της *xbest* δεν είναι ίση με 0.

Παραμετροποίηση και ιδιαιτερότητες του αλγορίθμου

Όπως αναφέραμε παραπάνω, η TS-PR έχει έναν αριθμό παραμέτρων. Για αυτές έπρεπε να γίνει ένας αριθμός δοκιμών για τον καθορισμό των τιμών τους, ώστε να γίνει ο αλγόριθμος πιο αποδοτικός. Επίσης, απαιτήθηκαν αλλαγές στην βασική δομή της mPR που αφορά κυρίως σε δυαδικά διανύσματα. Όλα αυτά θα τα δούμε ένα προς ένα στη συνέχεια αυτής της ενότητας.

Ας ξεκινήσουμε με τις αλλαγές που έπρεπε να γίνουν στην δομή της mPR. Το βασικότερο πρόβλημα, λόγω της ιδιαιτερότητας των προβλημάτων που καλούμαστε να λύσουμε, είναι ότι σε αντίθεση με την έκδοση του αλγορίθμου για δυαδικά διανύσματα, εδώ δεν έχουμε μόνο δύο τιμές (0 ή 1) για την κάθε συνιστώσα του διανύσματος, αλλά τρεις τιμές, 0, 1 και -1. Οπότε κάθε φορά που έπρεπε να αλλάξει η τιμή της συνιστώσας του τρέχοντος διανύσματος, έπρεπε να αποφασιστεί ποια από τις άλλες δύο θα είναι η νέα τιμή. Σε αυτό το σημείο δεν πρέπει να παραβλέψουμε τη βασική αρχή της PR, δηλαδή ότι σε κάθε επανάληψη δίνεται η δυνατότητα να πραγματοποιήσουμε μια αλλαγή είτε στο x^s είτε στο x^t, (με την προϋπόθεση η αλλαγή να γίνεται πάντα εναλλάξ ανά επανάληψη). Επίσης, ένας βασικός περιορισμός που θέτει το ίδιο το πρόβλημα είναι ο συγκεκριμμένος αριθμός των μηδενικών 0, 1 και -1 που μπορεί να έχει κάθε λύση. Σύμφωνα με αυτό, δεν ενδείκνυται η απλή αλλαγή της τιμής μίας συνιστώσας όπως γίνεται στην βασική δομή της PR. Αντιθέτως, στην περίπτωσή μας, χρησιμοποιήθηκαν τα διανύσματα x^s και x^t, καθώς και το δυαδικό

διάνυσμα σφάλματος που δείχνει τις θέσεις, στις οποίες οι λύσεις x^s και x^t διαφέρουν. Στις θέσεις όπου το *error* έχει τιμή 1, οι συνιστώσες των x^s και x^t είναι διαφορετικές, όπως αναφέραμε παραπάνω στην ανάλυση του αλγορίθμου. Έτσι, κρίθηκε αναγκαίο ότι έπρεπε να γίνει μία μετάθεση έτσι ώστε να αλλαχθεί η τιμή μιας επιπλέον συνιστώσας της λύσης, ώστε να μην υπάρχει παραβίαση των συνθηκών του προβλήματος σχετικά με τον αριθμό των 0, 1 και -1.

Το πρόβλημα που προέκυψε, ήταν το ποιες θα ήταν οι δύο αυτές συνιστώσες των οποίων τις τιμές έπρεπε να μεταθέσουμε. Την επιλογή αυτή δυσκόλεψε περαιτέρω η ίδια η αρχή της PR, δηλαδή ότι δεν επιτρέπεται η ταυτόχρονη αλλαγή της x^s με εκείνη της x^t, αλλά η διαδικασία της αλλαγής αυτών είναι αναγκαίο να γίνεται διαδοχικά. Το γενονός αυτό δικαιολογεί την αναφορά σε δύο συνιστώσες και όχι σε μία μόνο στην περιγραφή του *error_num*. Αυτό θα γίνει πιο κατανοητό στην συνέχεια. Ας θεωρήσουμε, λοιπόν, ότι βρισκόμαστε σε κάποια επανάληψη της δημιουργίας του μονοπατιού, στην οποία θέλουμε να αλλάξουμε την x^s έτσι ώστε να υιοθετήσει κάποιο χαρακτηριστικό από την x^t. Θα ξεκινήσουμε εντοπίζοντας ποιες συνιστώσες του *error* έχουν τιμή 1, δηλαδή τις συνιστώσες εκείνες όπου τα x^s και x^t διαφέρουν. Για παράδειγμα ας θεωρήσουμε ότι το *error* έχει τιμές 1 στις συνιστώσες i και j. Αυτό κατ' επέκταση σημαίνει ότι οι συνιστώσες $x^s[i]$ και $x^t[i]$ δεν είναι ίδιες, όπως επίσης και οι $x^s[j]$ και $x^t[j]$. Δηλαδή $x^s[i]\neq x^t[i]$ και $x^s[j]\neq x^t[j]$. Επίσης, αυτό που προσπαθεί να κάνει η PR είναι η αλλαγή μίας από τις συνιστώσες της x^s (εδώ την $x^s[i]$) με την τιμή της αντίστοιχης συνιστώσας της x^t (δηλαδή την $x^t[i]$) δηλαδή $x^s[i]\leftarrow x^t[i]$. Εφόσον όμως και τα δύο διανύσματα (x^s και x^t) έχουν ίδιο αριθμό 0, 1 και -1, θα υπάρχει σίγουρα μία συνιστώσα της $x^s[k]$ όπου η αντίστοιχη $x^t[k]$ θα έχει διαφορετική τιμή (δηλαδή $x^s[k]\neq x^t[k]$). Εφόσον το *error_num* είναι μεγαλύτερο του 2, τότε θα έχουμε περισσότερες των δύο συνιστωσών όπου το *error* θα έχει την τιμή 1. Στην περίπτωση που οι συνιστώσες αυτές είναι μόλις δύο, έστω ότι συμβαίνει αυτό στην περίπτωση που περιγράφουμε, τότε η $x^t[i]$ θα είναι ίδια με την $x^s[j]$ ($x^t[i]=x^s[j]$) και συνεπώς, αν κάνουμε μία μετάθεση στην x^s αλλάζοντας την τιμή της $x^s[i]$ με την $x^s[j]$ και το αντίστροφο (δηλαδή $x^s[i]\leftarrow x^s[j]$ και $x^s[j]\leftarrow x^s[i]$), τότε το νέο διάνυσμα που προκύπτει από την x^s θα έχει υιοθετήσει ένα χαρακτηριστικό της x^t και θα έχουν αλλάξει δύο συνιστώσες της x^s.

Όπως είναι λογικό, στις περισσότερες των περιπτώσεων δεν θα έχουμε μόνο δύο διαφορετικές συνιστώσες. Το παραπάνω παράδειγμα χρησιμοποιείται για να γίνει

απλά πιο κατανοητή η διαδικασία. Στην περίπτωση που οι διαφορετικές συνιστώσες είναι περισσότερες των δύο, τότε ψάχνουμε να βρούμε σύμφωνα με το παράδειγμα πιο πάνω, την συνιστώσα $x^s[k]$, η οποία σύμφωνα με το *error* είναι διαφορετική από την αντίστοιχη $x^t[k]$, (δηλαδή $x^s[k] \neq x^t[k]$) και είναι ίδια με την $x^t[i]$ (δηλαδή $x^s[k]=x^t[i]$). Αφού βρούμε αυτή τη συνιστώσα, τότε κάνουμε την μετάθεση (δηλαδή $x^s[i] \leftarrow x^s[k]$ και $x^s[k] \leftarrow x^s[i]$) ώστε να δούμε αν αυτή η μετάθεση είναι εκείνη που μας δίνει την καλύτερη συναρτησιακή τιμή. Κάνουμε την ίδια διαδικασία ξεκινώντας και από τις άλλες συνιστώσες όπου το *error* είναι 1, ελέγχοντας και για αυτές την συναρτησιακή τιμή των μεταθέσεων που μπορούν να δημιουργηθούν. Αφού βρούμε ποια μετάθεση από όλα τα πιθανά σενάρια μας δίνει την καλύτερη συναρτησιακή τιμή, τελικά την κρατάμε. Στην επόμενη επανάληψη ελέγχονται και πάλι οι θέσεις στις οποίες είναι διαφορετικές οι συνιστώσες, ενημερώνεται το *error* και συνεχίζεται η διαδικασία με τον ίδιο τρόπο.

Ένα παράδειγμα για να γίνει πιο κατανοητή η διαδικασία είναι το ακόλουθο, για το πρόβλημα $CW(6,2)$:

x^s:

0	0	1	1	1	-1

x^t:

1	0	-1	1	1	0

error:

1	0	1	0	0	1

Τα διανύσματα έχουν 3 θέσεις με διαφορετική τιμή, τις 0, 2 και 5.

Αν ξεκινήσουμε θέλοντας να αλλάξουμε την x^s, έχουμε 3 πιθανές περιπτώσεις:

1) $x^s[0] \leftarrow x^t[0]$ δηλαδή μετάθεση των $x^s[0]$ με $x^s[2]$
2) $x^s[2] \leftarrow x^t[2]$ δηλαδή μετάθεση των $x^s[2]$ με $x^s[5]$
3) $x^s[5] \leftarrow x^t[5]$ δηλαδή μετάθεση των $x^s[5]$ με $x^s[0]$

Για τα τρία σενάρια, οι μεταθέσεις που προκύπτουν είναι:

1	0	0	1	1	-1

0	0	-1	1	1	1

-1	0	1	1	1	0

Υπολογίζουμε την συναρτησιακή τιμή γι' αυτά τα τρία διανύσματα και κρατάμε αυτό που έχει την καλύτερη συναρτησιακή τιμή.

Ένα άλλο σημαντικό σημείο του αλγορίθμου, είναι ο τρόπος που επιλέγεται σε κάθε επανάληψη το x^t και το x^s. Επειδή η PR απαιτεί την δημιουργία μίας νέας λύσης από δύο άλλες καλές λύσεις, επιλέχθηκε να χρησιμοποιείται ως x^t το *xbest*. Για την x^s, μετά από σχετικά πειράματα που έγιναν με τον αλγόριθμο, χρησιμοποιήθηκε η πιο πρόσφατη *xnew* (δηλαδή η καλύτερη λύση της τρέχουσας επανάληψης του αλγορίθμου). Αυτή η επιλογή έγινε ώστε να αλλάζει συχνά η μία από τις δύο λύσεις καθώς, όπως έδειξαν τα πειράματα, η xnew ανανεώνεται αρκετά συχνά. Μια άλλη περίπτωση που εξετάστηκε ήταν η χρήση ενός ιστορικού προηγούμενων καλών λύσεων και η δοκιμή ανάμεσα σε αυτές μίας τυχαίας κάθε φορά. Αυτό δεν φάνηκε να είναι περισσότερο αποδοτικό, οπότε προτιμήθηκε η επιλογή της τελευταίας xnew.

Στο ίδιο σημείο του αλγορίθμου, επειδή επιλέχθηκε η x^s να γίνεται ίση με την *xnew*, πριν ξεκινήσει η νέα επανάληψη, γίνεται πάντα μια αξιολόγηση ώστε να μην είναι ποτέ ίδια η *xbest* με την *xnew*. Σε αντίθετη περίπτωση, η x^s αρχικοποιείται σε ένα τυχαίο διάνυσμα που ικανοποιεί τις συνθήκες του προβλήματος. Εκτός από αυτό, έγιναν πειράματα και για την περίπτωση να κρατάει ο αλγόριθμος ένα ιστορικό καλών προηγούμενων λύσεων και να επιλέγεται κάποια από αυτές τυχαία, αλλά φάνηκε ότι η τυχαία αρχικοποίηση βοηθούσε τον αλγόριθμο στην εξερεύνηση νέων μονοπατιών, αυξάνοντας την αποδοτικότητά του.

Στο σημείο αυτό θα αναλύσουμε τις παραμέτρους που χρησιμοποίησε ο αλγόριθμος και τα πειράματα που έγιναν για να βρούμε τις κατάλληλες τιμές τους. Για ευκολότερη ανάγνωση και κατανόηση της διαδικασίας παρατίθενται παρακάτω οι παράμετροι με τα ονόματα στα οποία θα αναφερόμαστε σε αυτές:

1) Ο μέγιστος αριθμός επαναλήψεων του αλγορίθμου (iterations_max)
2) Το μέγεθος της tabu list (list_length)
3) Ο μέγιστος αριθμός επαναλήψεων της Tabu (tabu_iterations)
4) μέγιστος αριθμός επαναλήψεων χωρίς εύρεση καλύτερης λύσης (*xbest*) για επανεκκίνηση του αλγορίθμου (iterations_reset)

Η διαδικασία που ακολουθήθηκε, όπως και στην περίπτωση της TS, ήταν η ακόλουθη. Για τον μέγιστο αριθμό επαναλήψεων του αλγορίθμου (iterations_max) έγιναν πειράματα με 1000, 10000, 50000 και 100000 επαναλήψεις. Τα πειράματα έδειξαν ότι οι 10000 επαναλήψεις ήταν ο καταλληλότερος αριθμός. Για την ακρίβεια, φάνηκε ότι η TS-PR στα περισσότερα προβλήματα δεν χρειαζόταν περισσότερες επαναλήψεις για την εύρεση λύσης. Για την συγκεκριμένη παράμετρο δεν υπήρχε λόγος να γίνουν ξεχωριστά πειράματα για την εύρεση της καλύτερης τιμής ανά πρόβλημα. Αν ο αλγόριθμος έβρισκε την λύση σταματούσε έτσι κι αλλιώς τις επαναλήψεις και δεν χρησιμοποιούσε τις υπόλοιπες διαθέσιμες. Έτσι, προτιμήθηκε να γίνουν πειράματα σε προβλήματα μεγαλύτερης διάστασης και δυσκολίας και κρατήθηκε η καλύτερη τιμή σε αυτά.

Για την δεύτερη παράμετρο που χρησιμοποιήθηκε (list_length), τα δεδομένα ήταν διαφορετικά. Όπως αναφέρθηκε και πιο πάνω (φέρνοντας και το παράδειγμα με τις μικρές και μεγάλες διαστάσεις και πως επηρρεάζει το μέγεθος της tabu list), δεν θα μπορούσαμε να έχουμε την ίδια τιμή για το μέγεθος της μνήμης σε όλα τα προβλήματα, διότι αυτό θα έκανε τον αλγόριθμο να δουλεύει σε κάποιες μόνο περιπτώσεις προβλημάτων. Έτσι μετά από πειράματα που έγιναν και περιγράφονται αναλυτικά στο κεφάλαιο της TS καταλήξαμε πως η TS δουλεύει με τον βέλτιστο δυνατό τρόπο όταν το μέγεθος της μνήμης (list_length) είναι ίσο με την διάσταση του εκάστοτε προβλήματος.

Για την τρίτη παράμετρο, τον μέγιστο αριθμό επαναλήψεων της TS (tabu_iterations), απαιτήθηκε εκτενής μελέτη καθώς, όσο μεγαλύτερος αριθμός επαναλήψεων χρησιμοποιούνταν τόσο πιο εκτενής τοπική αναζήτηση θα γινόταν.

Έγιναν πειράματα με 100, 500 και 1000 επαναλήψεις, 5 στον αριθμό για το καθένα από αυτά, σε προβλήματα μεγάλης διάστασης και δυσκολίας. Από τα αποτελέσματα φάνηκε ότι στην περίπτωση που η TS έκανε 1000 επαναλήψεις, τα αποτελέσματα ήταν καλύτερα. Αυτό συνιστούσε ότι είχαμε καλύτερα αποτελέσματα όταν επικεντρωνόμασταν σε τοπική αναζήτηση στο σημείο που προέκυπτε από την PR.

Η τελευταία παράμετρος για την οποία έγιναν πειράματα είναι η iterations_reset, δηλαδή ο μέγιστος αριθμός διαδοχικών επαναλήψεων του αλγορίθμου χωρίς βελτίωση της *xbest*. Έγιναν πειράματα στο σύνολο των προβλημάτων, με τον ίδιο τρόπο όπως και προηγουμένως για την list_length. Δηλαδή, 5 πειράματα για κάθε πρόβλημα, με την iterations_reset να παίρνει τιμές 10, 100, 500 και 1000 αλλά και χωρίς την χρήση παραμέτρου. Φάνηκε ότι στις περιπτώσεις που αρχικοποιούσαμε τον αλγόριθμο από τυχαία τιμή μετά από 10 επαναλήψεις του αλγορίθμου χωρίς βελτίωση της *xbest*, ο αλγόριθμος έδινε καλύτερες λύσεις.

Συνοπτικά, οι τιμές των παραμέτρων που χρησιμοποιήσαμε είναι:

1. iterations_max = 10000
2. list_length = dimension
3. tabu_iterations = 1000
4. iterations_reset = 10

3.3. Variable Neighborhood Search

Η Variable Neighborhood Search (VNS) είναι αλγόριθμος για την επίλυση προβλημάτων συνδυαστικής και καθολικής βελτιστοποίησης. Η βασική ιδέα είναι η συστηματική αλλαγή γειτονιάς. Η VNS βασίζεται σε τρεις βασικές παραδοχές:

1) Ένα τοπικό ελάχιστο σε μία γειτονιά δεν είναι απαραίτητα τοπικό ελάχιστο και σε άλλη γειτονιά.
2) Ένα καθολικό ελάχιστο είναι τοπικό ελάχιστο σε όλες τις πιθανές γειτονιές.
3) Σε πολλά προβλήματα, τα τοπικά ελάχιστα μίας ή περισσοτέρων γειτονιών N_k είναι σχετικά κοντά το ένα στο άλλο.

Η βασική δομή του αλγορίθμου είναι η ακόλουθη:

Algorithm Basic VNS
Function BVNS(x, k_{max} , t_{max})
1 $t \leftarrow 0$
2 while $t < t_{max}$ **do**
3 $k \leftarrow 1$
4 **repeat**
5 $x' \leftarrow$ Shake(x, k) //Shaking
6 $x'' \leftarrow$ BestImprovement(x') // Local search
7 $x,k \leftarrow$ NeighborhoodChange(x,x'',k) // Change neighborhood
until $k = k_{max}$
8 $t \leftarrow$ CpuTime()
return x

Οι βασικές συναρτήσεις είναι οι ακόλουθες:

Shake

Function Shake(x, k)
1 $w \leftarrow [1+\text{Rand}(0, 1) \times |N_k(x)|]$
2 $x' \leftarrow x^w$
return x'

Αυτή η συνάρτηση επιστρέφει ένα τυχαίο σημείο της N_k.

Local Search

Function BestImprovement(x)
1 repeat
2 $x' \leftarrow x$
3 $x \leftarrow arg\min_{y \in N(x)} f(y)$
until($f(x) \geq f(x')$)
return x

Αυτή η συνάρτηση αντιστοιχεί στην τοπική αναζήτηση στην γειτονιά *N(x)*.

Change Neighborhood

Function NeighborhoodChange (x, x', k)
1 if $f(x') < f(x)$ **then**
2 $x \leftarrow x'$ //Makeamove
3 $k \leftarrow 1$ // Initial neighborhood
else
4 $k \leftarrow k + 1$ // Next neighborhood
return x , k

Αυτή η συνάρτηση κάνει αλλαγή στην γειτονιά.

Αναλυτική Περιγραφή:

Το βασικό χαρακτηριστικό της VNS είναι η συστηματική αλλαγή γειτονιάς. Με αυτό τον τρόπο η VNS ψάχνει στις γειτονιές των πιθανών λύσεων για την καλύτερη λύση του προβλήματος. Όπως σε όλους τους αλγορίθμους, έτσι και στην VNS, ξεκινάμε τον αλγόριθμό μας με την αρχικοποίηση, παίρνοντας ένα τυχαίο αρχικό διάνυσμα το οποίο πληρεί τις συνθήκες του προβλήματος, δηλαδή τον περιορισμό στον αριθμό των 0, 1 και -1, όπως προβλέπονται από τη μοντελοποίηση του προβλήματος που περιγράφηκε στην αρχή της διατριβής. Αμέσως μετά την αρχικοποίηση του αλγορίθμου θέτουμε το αρχικό διάνυσμα ως το καλύτερο του αλγορίθμου.

Για την παρουσίαση του αλγορίθμου θα χρησιμοποιήσουμε παρακάτω τους ακόλουθους συμβολισμούς: με x συμβολίζεται το διάνυσμα του οποίου τις γειτονιές εξερευνούμε σε κάθε επανάληψη, x’ το τυχαίο διάνυσμα που μας επιστρέφει η

συνάρτηση Shake, ενώ *x*'' είναι το διάνυσμα που μας επιστρέφει η συνάρτηση BestImprovement μετά την τοπική αναζήτηση.

Σε αντίθεση με τους άλλους αλγόριθμους που μελετήθηκαν στην διατριβή, η VNS είναι ο μοναδικός που δεν έχει ως κριτήριο τερματισμού τον αριθμό επαναλήψεων του αλγορίθμου. Ο συγκεκριμένος αλγόριθμος επειδή μελετάει και εναλλάσσει αρκετές γειτονιές και σημεία, κρίθηκε προτιμότερο να έχει σαν κριτήριο τερματισμού τον χρόνο εκτέλεσης (σε millisecond). Έτσι, οποιαδήποτε στιγμή ξεπεραστεί ο χρόνος που έχει οριστεί από τον χρήστη, ο αλγόριθμος σταματά τις επαναλήψεις και επιστρέφει την καλύτερη λύση που έχει βρει ως εκείνη την στιγμή. Περισσότερα για την επιλογή του χρόνου, τα πειράματα που έγιναν για την επιλογή αυτής της παραμέτρου, καθώς και άλλες πληροφορίες αναφέρονται στην συνέχεια.

Η Basic VNS (BVNS) εκδοχή του αλγορίθμου που υλοποιήθηκε συνδυάζει ντετερμινιστικές και στοχαστικές αλλαγές της γειτονιάς. Το ντετερμινιστικό κομμάτι εκπροσωπείται από την τοπική αναζήτηση, δηλαδή την συνάρτηση BestImprovement που ανεφέρθηκε πιο πάνω. Το στοχαστικό κομμάτι αντιστοιχεί στην συνάρτηση Shake, που αναφέρθηκε επίσης πιο πάνω. Για πληρέστερη ανάλυση και κατανόηση του αλγορίθμου, είναι καλό να αναλύσουμε περαιτέρω τα τμήματα από τα οποία αποτελείται. Δηλαδή τις συναρτήσεις Shake, BestImprovement και NeighborhoodChange, οι ψευδοκώδικες των οποίων δόθηκαν αναλυτικά παραπάνω.

Σε κάθε κύκλο του αλγορίθμου, καλείται πρώτα η Shake, μετά η BestImprovement και τέλος η NeighborhoodChange. Οπότε, αμέσως μετά την την αρχικοποίηση και την ανάθεση τιμής στην *xbest* ξεκινούν οι επαναλήψεις του αλγορίθμου, οι οποίες διαρκούν μέχρι ο χρόνος εκτέλεσης να ξεπεράσει τον μέγιστο χρόνο που έθεσε ο χρήστης.

Η αρχική ενέργεια του αλγορίθμου, είναι η κλήση της συνάρτησης Shake. Η βασική λειτουργία αυτής της συνάρτησης είναι να δώσει ένα καινούριο τυχαίο διάνυσμα από την γειτονιά του *x*, την οποία εξερευνούμε εκείνη την στιγμή. Ας συμβολίσουμε την γειτονιά αυτή με *k*, όπως και στον ψευδοκώδικα του αλγόριθμου. Προφανώς, κάθε φορά που ξεκινάει μία επανάληψη του αλγορίθμου πάντα ξεκινάμε από την πρώτη γειτονιά. Στην συνέχεια της ανάλυσης του αλγορίθμου θα δείξουμε πώς αλλάζουμε γειτονιά αναζήτησης. Το νέο, λοιπόν, τυχαίο διάνυσμα που μας επιστρέφει η Shake το ονομάζουμε *x*'. Ο λόγος για τον οποίο χρησιμοποιούμε την Shake είναι για να δώσει τυχαιότητα στα δείγματα προς αναζήτηση. Η Shake δεν χρησιμοποιείται πάντοτε στην επίλυση από την VNS, αλλά είναι πολύ χρήσιμη στις

περιπτώσεις προβλημάτων με πολύ μεγάλο χώρο αναζήτησης όπου η εξαντλητική αναζήτηση των πιθανών λύσεων είναι ανέφικτη. Τα προβλήματα που επιλύθηκαν ανήκουν σε αυτή την κατηγορία και συνεπώς, η χρήση της κρίθηκε αναγκαία.

Το δεύτερο βήμα του αλγορίθμου είναι η κλήση της BestImprovement. Με αυτή τη συνάρτηση γίνεται τοπική αναζήτηση στη γειτονιά του *x*' (το τυχαίο διάνυσμα που μας έδωσε η Shake), αναζητώντας το διάνυσμα με την καλύτερη συναρτησιακή τιμή. Το διάνυσμα που προκύπτει από αυτή τη διαδικασία, το ονομάζουμε *x*''. Η σημαντικότητα αυτού του βήματος έγκειται στο γεγονός ότι η BVNS βασίζεται σε δύο διαδικασίες. Στην στοχαστική, που περιγράφηκε μέσω της Shake, και στην ντετερμινιστική, την οποία επιτελεί η BestImprovement. Μέσω αυτής επιτυγχάνεται η καλύτερη αναζήτηση στις γειτονιές ευρεθέντων λύσεων.

Το τελικό στάδιο της VNS είναι η αλλαγή γειτονιάς, εφόσον αυτό κριθεί χρήσιμο. Μετά την εύρεση της *x*'', καλείται η συνάρτηση ChangeNeighborhood. Κατά την διαδικασία αυτή, ο αλγόριθμος ελέγχει κατά πόσο το διάνυσμα *x*'' που προέκυψε από τις Shake και BestImprovement, έχει καλύτερη συναρτησιακή τιμή. Στην περίπτωση που η συναρτησιακή τιμή του *x*'' είναι καλύτερη του x, ξεκινάμε μια νέα επανάληψη για τον αλγόριθμο, θέτοντας το *x* ίσο με *x*'', και ψάχνουμε πλέον στην πρώτη γειτονιά του νέου *x* (δηλαδή θέτουμε $x \leftarrow x''$ και $k \leftarrow 1$). Αντίθετα, αν η συναρτησιακή τιμή του *x*'' δεν είναι καλύτερη από του *x*, τότε συνεχίζουμε να ψάχνουμε τις γειτονιές του *x* που έχουμε, αλλά αυτή τη φορά ψάχνουμε στην επόμενη γειτονιά (k+1). Ασφαλώς ο αλγόριθμος πρέπει να έχει έναν ανώτατο αριθμό γειτονιών στις οποίες θα ψάξει, και αν ξεπεραστεί αυτός τότε να ξεκινήσει μια νέα επανάληψη από ένα νέο αρχικό σημείο και από την πρώτη του γειτονιά. Ο μέγιστος αριθμός γειτονιάς προς αναζήτηση θα αναλυθεί περαιτέρω παρακάτω και αποτελεί μια από τις δύο βασικότερες παραμέτρους για τις οποίες χρειάστηκαν πειράματα για να βρεθεί η κατάλληλη τιμή τους.

Όταν ο αλγόριθμος ολοκληρώσει την διαδικασία με ChangeNeighborhood, ελέγχει αν βρέθηκε καλύτερη λύση και αν εξαντλήθηκε ο αριθμός γειτονιών. Αν δεν έχει εξαντληθεί, προχωράει στην επόμενη γειτονιά. Σε αντίθετη περίπτωση ξεκινά μια νέα επανάληψη, όπως αναφέραμε και πιο πάνω.

Παραμετροποίηση και ιδιαιτερότητες του αλγορίθμου

‘Ενα βασικό στοιχείο της υλοποίησης της VNS αφορά στην μορφή των γειτονιών και στον τρόπο παραγωγής και μετάβασης από μια γειτονιά σε άλλη. Μετά από αρκετή ανάλυση, επιλέχθηκε οι γειτονιές ενός διανύσματος να αποτελούνται από τις μεταθέσεις του. Έτσι, καθορίστηκε η πρώτη γειτονιά να αποτελείται από τις μεταθέσεις του διανύσματος που διαφέρουν σε δύο συνιστώσες, η δεύτερη γειτονιά σε τρεις συνιστώσεις κλπ. Επίσης, έπρεπε να γίνει πολύ προσεκτικά η δημιουργία των γειτονιών χωρίς να επαναλαμβάνονται κάποια διανύσματα ενώ, όπως θα φανεί πιο κάτω, ο χειρισμός κατά την δημιουργία των γειτονιών είχε ισχυρή εξάρτηση από την διάσταση του προβλήματος.

Όπως σε όλους τους αλγορίθμους λοιπόν, έτσι και στην BVNS που υλοποιήθηκε, έγινε μελέτη για την εύρεση τιμών των παραμέτρων.
Οι παράμετροι ήταν οι ακόλουθοι:

1. *kMax* : μέγιστος αριθμός γειτονιών.
2. *tMax* : μέγιστος χρόνος εκτέλεσης του αλγορίθμου.

Η πρώτη από τις δύο παραμέτρους είναι ο μέγιστος αριθμός γειτονιών. Όπως αναφέραμε και στην ανάλυση, όταν ο αλγόριθμος δεν καταφέρει να βρει καλύτερη τιμή στο τέλος μιας επανάληψης, ξεκινάει την επόμενη επανάληψη από την επόμενη γειτονιά. Η *kMax* είναι η μέγιστη γειτονιά μέχρι την οποία μπορεί ο αλγόριθμος να ψάχνει. Όπως είναι λογικό, η παράμετρος αυτή εξαρτάται από την διάσταση του προβλήματος. Άλλωστε, δεν θα μπορούσαμε να ορίσουμε για ένα πρόβλημα μεγαλύτερη γειτονιά από το μέγεθος της διάστασης, καθώς δεν θα υπήρχε σύμφωνα με τον ορισμό της. Συνεπώς, απαιτήθηκε σημαντική πειραματική μελέτη ώστε η τιμή που παίρνει η παράμετρος σε κάθε πρόβλημα να οδηγεί στην εύρεση καλών λύσεων.

Η διαδικασία που ακολουθήθηκε ήταν η εξής. Επιλέχθηκε να δοκιμαστεί η λειτουργία του αλγορίθμου δίνοντας στην *kMax* τιμές από 1 έως 10. Για κάθε πρόβλημα έγιναν 5 πειράματα για κάθε μία από αυτές τις τιμές και κρατήθηκαν οι μέσοι όροι των επιτυχημένων πειραμάτων, καθώς και οι χρόνοι που απαιτούνταν. Σε

προβλήματα με διάσταση μικρότερη του 10 ο αλγόριθμος έπαιρνε την τιμή της διάστασης του προβλήματος μειωμένης κατά ένα.

Μετά από την συλλογή και ανάλυση των αποτελεσμάτων, προέκυψε ότι στο σύνολο των προβλημάτων ο αριθμός των επιτυχημένων αποτελεσμάτων δεν μεταβαλλόταν σημαντικά για *kMax*=5 και πάνω. Σε αντίθεση, ο χρόνος που χρειαζόταν αυξανόταν σημαντικά όσο το *kMax* γινόταν μεγαλύτερο. Συνεπώς, κρίθηκε ότι στα προβλήματα με διάσταση μεγαλύτερη του πέντε 5 ο αριθμός αυτός ήταν ικανοποιητικός. Στα υπόλοιπα προβλήματα (μικρότερης διάστασης), επιλέχθηκε η *kMax* να παίρνει την τιμή της διάστασης μειωμένης κατά ένα, καθώς εκεί φαινόταν ο αλγόριθμος να αποδίδει καλύτερα και ο χρόνος που χρειαζόταν δεν ήταν απαγορευτικός.

Η δεύτερη παράμετρος που μελετήθηκε είναι ο χρόνος εκτέλεσης του αλγορίθμου. Όπως ήδη αναφέραμε, η VNS είναι ο μόνος αλγόριθμος από αυτούς που ερευνούμε στην διατριβή που δεν έχει ως κριτήριο τερματισμού τον αριθμό επαναλήψεων του αλγορίθμου. Εδώ το κριτήριό μας ήταν ο χρόνος εκτέλεσης (σε milliseconds) που θέτει ο χρήστης. Για την εύρεση της κατάλληλης τιμής, χρησιμοποιήθηκαν τα αποτελέσματα που προέκυψαν από το προηγούμενο βήμα για την εύρεση της *kMax*. Πιο συγκεκριμένα, φάνηκε ότι στα περισσότερα προβλήματα, ο χρόνος που χρειαζόταν για τα προβλήματα ήταν μεταξύ 2 και 5 λεπτών. Οπότε και εδώ ακολουθήθηκε η ίδια διαδικασία. Έγιναν δηλαδή για κάθε ένα από τα προβλήματα 5 πειράματα με κάθε μία από τις τιμές {2,3,4,5} λεπτά και κρατήθηκαν στατιστικά επιτυχιών για κάθε πρόβλημα. Έτσι, μετά την ανάλυση αυτών φάνηκε ότι τα 3 λεπτά ήταν αρκετά για τον αλγόριθμο. Μάλιστα, σε ορισμένες περιπτώσεις φάνηκε να ήταν περισσότερα από τον απαιτούμενο χρόνο.

Συνοπτικά οι τιμές των παραμέτρων που χρησιμοποιήσαμε είναι:

- $kMax = \begin{cases} dim - 1\,, & \text{αν } dim \leq 5, \\ 5\,, & \text{αν } dim > 5. \end{cases}$
- *tMax* = 180 seconds.

3.4. Iterated Gradient Descent

Η υλοποίηση της Iterated Gradient Descent (IGD) βασίστηκε στην υλοποίηση της TS. Η διαφορά ανάμεσα στην TS και την IGD είναι ότι, ενώ η TS επιτρέπει τις ανοδικές κινήσεις μη-βελτίωσης της λύσης, η IGD δεν τις επιτρέπει και σταματά την αναζήτηση όταν βρει τοπικό ελάχιστο. Η υπόλοιπη υλοποίηση είναι ίδια με της TS.

Το βασικό χαρακτηριστικό τις IGD είναι ότι συνεχίζει την αναζήτηση λύσης όσο βρίσκει διανύσματα για τα οποία η αντικειμενική συνάρτηση δίνει καλύτερες τιμές, αλλά σταματάει αμέσως μόλις βρει μια τιμή η οποία είναι χειρότερη από αυτήν που βρήκε ήδη. Μπορούμε να φανταστούμε την IGD σαν μια διαδικασία που ακολουθεί ένα μονοπάτι τιμών που έχει κατεύθυνση προς μικρότερες τιμές διαρκώς, ενώ μόλις βρει μια τιμή που κάνει την κατεύθυνση του μονοπατιού να αλλάξει προς τιμές μεγαλύτερες, τότε σταματά.

Η υλοποίηση της IGD βασίστηκε στην υλοποίηση της TS, η οποία τροποποιήθηκε ώστε να σέβεται την βασική αρχή της IGD. Το αρχικό στάδιο είναι και εδώ η αρχικοποίηση του αλγορίθμου, παίρνοντας ένα τυχαίο αρχικό διάνυσμα. Αυτό πρέπει να πληρεί τις συνθήκες του προβλήματος, δηλαδή να είναι κατάλληλος αριθμός των 0, 1 και -1, όπως προβλέπεται από την μοντελοποίηση του προβλήματος που περιγράφηκε στην αρχή της διατριβής. Στην IGD χρησιμοποιούμε τρία βασικά διανύσματα, τα *xbest*, *xnew* και *xtemp*. Το *xbest* συμβολίζει την καλύτερη λύση που έχει βρει ο αλγόριθμος έως τώρα. Η *xnew* συμβολίζει την καλύτερη έως τώρα λύση του μονοπατιού που ξεκινήσαμε να εξερευνούμε. Τέλος, το *xtemp*, είναι η νέα μετάθεση που παράγουμε κάθε φορά από την *xnew* κατά την διαδικασία εύρεσης νέων λύσεων.

Στην πρώτη επανάληψη του αλγορίθμου, ξεκινάμε από το τυχαίο διάνυσμα που πήραμε, *xnew*, και ψάχνουμε να βρούμε την μετάθεσή του που έχει δύο [2] συνιστώσες διαφορετικές από την *xnew*, δίνει την καλύτερη συναρτησιακή τιμή και δεν βρίσκεται στην tabu list. Το διάνυσμα που βρίσκουμε το ονομάζουμε *xtemp*. Όπως προδίδει και το όνομά του, είναι ένα διάνυσμα το οποίο θα αξιολογηθεί και από την τιμή του, θα εξαρτηθεί η περαιτέρω συμπεριφορά του αλγορίθμου. Το σημείο αυτό είναι σημαντικό για τον αλγόριθμο διότι κάνει την IGD να έχει διαφορετική λειτουργία από την TS. Σε αυτό το σημείο ελέγχουμε αν η συναρτησιακή τιμή του

καινούριου διανύσματος, *xtemp* είναι καλύτερη από την συναρτησιακή τιμή του διανύσματος *xnew*.

Στην περίπτωση που βρούμε καλύτερη τιμή, θέτουμε σαν νέο *xnew* το *xtemp*, και προχωράμε στο επόμενο βήμα του αλγορίθμου. Εδώ αφαιρούμε την παλιότερη λύση που έχουμε καταχωρήσει στην tabu list και προσθέτουμε την νέα *xnew*, ώστε να μην την επισκεφτούμε ξανά στις επόμενες επαναλήψεις του αλγορίθμου. Αμέσως μετά ελέγχουμε τα κριτήρια τερματισμού και, εφόσον δεν ισχύουν, επιστρέφουμε στην επιλογή νέας μετάθεσης από το νέο *xnew*. Όπως γίνεται κατανοητό, σε αυτή την περίπτωση βρισκόμαστε σε ένα μονοπάτι όπου τα διανύσματα μας δίνουν όλο και καλύτερες συναρτησιακές τιμές. Έχει, δηλαδή, φθίνουσα πορεία.

Στην περίπτωση που η *xtemp* δεν δίνει καλύτερη λύση, τότε παίρνουμε ένα τυχαίο διάνυσμα, το όποιο θέτουμε ως *xnew*. Ακολούθως, αναζητούμε μία νέα μετάθεση *xtemp* από την *xnew,* με τα χαρακτηριστικά που αναφέραμε πιο πάνω, και συνεχίζουμε με αυτό τον τρόπο τις επαναλήψεις του αλγορίθμου. Την περίπτωση αυτή μπορούμε να την φανταστούμε ως ένα μονοπάτι που έχει ακολουθήσει μία διαδρομή καλύτερων τιμών και κάποια στιγμή αλλάζει η κατεύθυνσή του προς κάποια μεγαλύτερη τιμή.

Τα παραπάνω αποκαλύπτουν την δομή του αλγορίθμου, σύμφωνα με την οποία, κάθε φορά ξεκινάμε από ένα τυχαίο σημείο και ακολουθούμε το μονοπάτι με την πιο απότομη κλίση που δημιουργείται προς καλύτερες λύσεις. Όταν το μονοπάτι σταματήσει να δίνει καλύτερες λύσεις (τοπικό ελάχιστο), ξεκινάμε από καινούριο τυχαίο σημείο προσπαθώντας να εξερευνήσουμε ένα νέο μονοπάτι που πιθανότατα οδηγεί σε άλλο τοπικό ακρότατο.

Όπως και σε άλλες περιπτώσεις αλγορίθμων, έτσι και στην IGD χρησιμοποιούμε έναν μέγιστο αριθμό επαναλήψεων χωρίς εύρεση καλύτερης λύσης (xbest) για επανεκκίνηση του αλγορίθμου. Σε αυτή την περίπτωση, στέλνουμε τον αλγόριθμο να ξεκινήσει από κάποια τυχαίο διάνυσμα και πάλι.

Τέλος, οφείλουμε να επισημάνουμε ότι σε οποιοδήποτε σημείο παράγεται κάποια καινούρια λύση από τον αλγόριθμο, πάντοτε ελέγχεται κατά πόσο αυτή είναι καλύτερη από την καλύτερη λύση που έχει βρει γενικά ο αλγόριθμος (*xbest*).

Παραμετροποίηση και ιδιαιτερότητες του αλγορίθμου

Όπως και στους υπόλοιπους αλγορίθμους, έτσι και στην IGD, χρησιμοποιήθηκε ένας αριθμός παραμέτρων για την καλύτερη λειτουργία του αλγορίθμου. Για τις παραμέτρους αυτές έπρεπε να γίνει ένας αριθμός πειραμάτων με σκοπό να βρεθούν οι κατάλληλες τιμές που θα έκαναν τον αλγόριθμο πιο αποδοτικό.

Για την καλύτερη και ευκολότερη ανάλυση και αναφορά στις παραμέτρους που χρησιμοποιήθηκαν, αυτές παρατίθενται αμέσως με τα ακόλουθα ονόματα:

(1) Ο μέγιστος αριθμός επαναλήψεων του αλγορίθμου (iterations_max)
(2) Το μέγεθος της tabu list (list_length)
(3) Ο μέγιστος αριθμός επαναλήψεων χωρίς εύρεση καλύτερης λύσης *xbest* για επανεκκίνηση του αλγορίθμου (iterations_reset)

Λόγω του μεγάλου αριθμού των προβλημάτων (194 προβλήματα) η διαδικασία εύρεσης των κατάλληλων παραμέτρων για κάθε πρόβλημα ήταν μια χρονοβόρα διαδικασία. Όπως περιγράψαμε π.χ. για την list_length στην διαδικασία εύρεσης παραμέτρων την TS είναι λογικό να μην έχουμε το ίδιο μέγεθος για την tabu list σε όλα τα προβλήματα.

Έτσι, για τις παραμέτρους που αναφέραμε πιο πάνω, ξεκινώντας από την iterations_max, έγινε ένας αριθμός πειραμάτων για τιμές 1000 , 10000 και 100000. Μετά από το σύνολο των πειραμάτων φάνηκε πως ο αλγόριθμος είχε ικανοποιητική απόδοση για 10000. Σε πολλά προβλήματα ο αλγόριθμος έβρισκε την λύση μετά από πολύ μικρότερο αριθμό επαναλήψεων από εκείνον που είχε οριστεί ως μέγιστος. Η διαδικασία που ακολουθήθηκε και εδώ ήταν παρόμοια με εκείνη των υπολοίπων αλγορίθμων. Έτσι, επιλέχθηκε το μεγαλύτερο πλήθος των πειραμάτων για την συγκεκριμένη παράμετρο να γίνει για προβλήματα μεγάλης διάστασης που είναι πιο χρονοβόρα στην επίλυσή τους, καθώς αυτά χρειάζονται και μεγαλύτερο αριθμό επαναλήψεων.

Για την δεύτερη παράμετρο (list_length) και την εύρεση της κατάλληλης τιμής της ανά πρόβλημα έγινε εκτενής μελέτη και ανάλυση στο αντίστοιχο κεφάλαιο της TS όπου δίνονται περισσότερες πληροφορίες. Το τελικό συμπέρασμα ήταν ότι η

list_length εξαρτάται άμεσα από την διάσταση του προβλήματος και οι καλύτερες λύσεις λαμβάνονται όταν αυτή έχει μέγεθος ίσο με την διάσταση του προβλήματος.

Για την iterations_reset έγιναν 5 πειράματα ανά πρόβλημα και ανά τιμή, για τις τιμές 50, 100, 500 και 1000 επαναλήψεις, χωρίς βελτίωση της *xbest*. Τα στοιχεία που προέκυψαν από τα πειράματα έδειξαν ότι η τιμή 100 έδινε τα καλύτερα αποτελέσματα για τον αλγόριθμο.

Τέλος, σε αυτό το σημείο πρέπει να τονιστεί ότι η σημαντικότητα της χρήσης μνήμης (tabu list) στην IGD, έγκειται στο ότι με την χρήση της μνήμης αποφεύγεται η κυκλική επίσκεψη σε ίδιες λύσεις.

Συνοπτικά οι τιμές των παραμέτρων που χρησιμοποιήσαμε είναι:

(1) iterations_max= 10000
(2) list_length= ίδια με την διάσταση του προβλήματος
(3) iterations_reset= 100

3.5. Random Search

Η Random Search (RS) είναι η απλή τυχαία αναζήτηση. Δεν χρησιμοποιεί μνήμη όπως στους προηγούμενους αλγόριθμους που περιγράφηκαν, ούτε γίνεται αναζήτηση σε γειτονιές όπως στην VNS. Στην RS παράγουμε τυχαία διανύσματα που πληρούν τις συνθήκες του προβλήματος και εξετάζουμε τις συναρτησιακές τιμές τους.

Η βασική δομή της απλής RS περιγράφεται από τα ακόλουθα βήματα:

1. Αρχικοποίηση σε τυχαίο σημείο *x* στον χώρο αναζήτησης.
2. Μέχρι να ικανοποιηθεί κάποιο κριτήριο τερματισμού (π.χ. αριθμός επαναλήψεων του αλγορίθμου) επαναλαμβάνουμε:
   a. Παίρνουμε ένα τυχαίο σημείο *y* στον χώρο αναζήτησης
   b. Αν f(y)<f(x) τότε μετακινούμαστε στο σημείο *y* θέτοντας *x = y*
3. Τώρα το *x* έχει την καλύτερη τιμή έως τώρα της αντικειμενικής συνάρτησης f

Αναλυτική Περιγραφή

Ο αλγόριθμος της RS είναι, συγκριτικά με του υπόλοιπους, πολύ απλούστερος στην υλοποίησή του και λειτουργεί με τον τρόπο που περιγράφεται πιο αναλυτικά στη συνέχεια.

Σε κάθε επανάληψη του αλγορίθμου επαναλαμβάνουμε τα εξής βήματα. Παίρνουμε ένα τυχαίο διάνυσμα που υπακούει στους περιορισμούς του προβλήματος, όπως περιγράφονται από τη μοντελοποίησή του. Δηλαδή να έχει συγκεκριμένο αριθμό 0, 1 και -1.

Στην περίπτωση που είμαστε στην πρώτη επανάληψη του αλγορίθμου, θέτουμε το αρχικό διάνυσμα *x* ως το καλύτερο που έχει βρει ο αλγόριθμος (*xbest*). Αν όχι, τότε ελέγχουμε την συναρτησιακή τιμή του νέου διανύσματος (*xnew*) και, στην περίπτωση που αυτό είναι καλύτερο, τότε θέτουμε το διάνυσμα *xnew* ως το καλύτερο που έχει βρει ο αλγόριθμος έως τώρα, (*xbest*).

Στην επόμενη επανάληψη του αλγορίθμου παίρνουμε ξάνα ένα τυχαίο διάνυσμα και συνεχίζουμε με τον ίδιο τρόπο την λειτουργία του αλγορίθμου ως την στιγμή που θα ολοκληρωθούν οι διαθέσιμες επαναλήψεις ή βρεθεί η λύση του προβλήματος.

Η χρήση της RS με τους υπόλοιπους αλγορίθμους γίνεται για λόγους τυπικούς διότι κάθε μεταευρετική μέθοδος που εφαρμόζεται σε κάποιο πρόβλημα θα πρέπει να επιτυγχάνει απόδοση τουλάχιστον ίση με αυτή της RS για να έχει λόγο ύπαρξης. Άρα η RS αποτελεί το κάτω όριο απόδοσης για να γίνει αποδεκτός ένας αλγόριθμος.

Όπως είναι λογικό, το γεγονός αυτό κάνει τον αλγόριθμο να μην επηρεάζεται από τοπικά ελάχιστα κατά την αναζήτησή του και να ψάχνει στο σύνολο του χώρου αναζήτησης. Όμως λόγω του ότι κατά την παραγωγή των νέων λύσεων δεν αξιοποιούμε πληροφορίες από προηγούμενες επαναλήψεις του αλγορίθμου, θα μπορούσαμε να βρισκόμαστε πολύ κοντά σε μια πολύ καλή λύση του προβλήματος και να μην την επισκεφθούμε ποτέ.

Παραμετροποίηση και ιδιαιτερότητες του αλγορίθμου

Από τη φύση της η RS δεν είναι αλγόριθμος με πολλούς παραμέτρους. Όπως και στους υπόλοιπους, επιλέχθηκε να τεθεί σαν κριτήριο τερματισμού του αλγορίθμου ο μέγιστος αριθμός επαναλήψεων του αλγορίθμου (iterations_max).

Για την περίπτωση της RS έγιναν πειράματα για 1000 , 10000 και 100000 επαναλήψεις. Φάνηκε ότι ο αριθμός των 10000 επαναλήψεων ήταν επαρκής, και σε πολλά προβλήματα που βρέθηκε λύση, ο αριθμός των υπολογισμών που έγιναν ήταν αρκετά μικρότερος από αυτόν.

ΚΕΦΑΛΑΙΟ 4. ΣΥΓΚΡΙΣΗ ΑΛΓΟΡΙΘΜΩΝ

Στις επόμενες παραγράφους αναλύεται η διαδικασία εκτέλεσης πειραμάτων, συγκρίνονται τα αποτελέσματα και παρουσιάζονται τα συμπεράσματα που προέκυψαν.

4.1. Περιγραφή πειραμάτων

Μετά την επιλογή των κατάλληλων παραμέτρων για τον εκάστοτε αλγόριθμο και την τελική υλοποίησή του, ακολούθησαν τα πειράματα για την μελέτη αποδοτικότητας ανά πρόβλημα.

Όπως και στη διαδικασία για την επιλογή των παραμέτρων, έτσι και εδώ, στην τελική φάση των πειραμάτων ακολουθήθηκε η ίδια μέθοδος για τα πειράματα. Για κάθε ένα από τα 194 προβλήματα έγιναν 30 ανεξάρτητα πειράματα ανα αλγόριθμο χρησιμοποιώντας τις αντίστοιχες παραμέτρους που προέκυψαν από την ανάλυση προηγούμενων κεφαλαίων της διατριβής.

Τα στοιχεία που κατεγράφησαν μετά την ολοκλήρωση κάθε πειράματος για κάθε πρόβλημα ήταν:

(1) Η επιτυχία του αλγορίθμου στην εύρεση λύσης.
(2) Ο απαιτούμενος χρόνος για να βρει τη λύση (milliseconds).
(3) Στην περίπτωση που δεν βρήκε τη λύση, την αντίστοιχη απόκλιση από αυτή.
(4) Ο αριθμός των επαναλήψεων που χρειάστηκαν.

Από το σύνολο των δεδομένων που συλλέχθηκαν, προέκυψε για κάθε πρόβλημα το ποσοστό επιτυχίας του αλγορίθμου καθώς και ο μέσος χρόνος που

χρειάστηκε ο αλγόριθμος για τα επιτυχή πειράματα σε milliseconds, καθώς και την τυπική απόκλιση του χρόνου (σε milliseconds). Όλα αυτά τα δεδομένα παρατίθενται στους πίνακες του παραρτήματος Α.

Επίσης, για καλύτερη οπτικοποίηση των αποτελεσμάτων, έγινε ένας αριθμός γραφημάτων όπου είναι ευδιάκριτη η αλλαγή στην απόδοση του αλγορίθμου όσο δυσκολεύουν τα προβλήματα. Τα γραφήματα αυτά χωρίζονται σε τρεις κατηγορίες.

(1) Γραφήματα αριθμού επιτυχιών (Παράρτημα Α)
(2) Γραφήματα μέσου απαιτούμενου χρόνου (Παράρτημα Β)
(3) Γραφήματα τυπικής απόκλισης χρόνου (Παράρτημα Γ)

Για την καλύτερη οργάνωση των δεδομένων, τα τελικά γραφήματα δίνονται σε δεκάδες προβλημάτων σύμφωνα με τη διάσταση. Έτσι, υπάρχουν 10 γραφήματα των επιτυχιών στα οποία το πρώτο αφορά τα προβλήματα *CW(n,k)* όπου $n \in [1,10]$ και $k \in [1,10]$, το δεύτερο γράφημα για $n \in [11,20]$ κ.λ.π., και παρουσιάζουν με μορφή τρισδιάστατων ραμβογραμμάτων το σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα.

Τα γραφήματα του μέσου απαιτούμενου χρόνου δημιουργήθηκαν με την ίδια λογική. Το μόνο που αξίζει να επισημανθεί στις περιπτώσεις των χρόνων, είναι το γεγονός ότι, λόγω της συχνά μεγάλης απόκλισης στους χρόνους ανάμεσα στα προβλήματα που παρουσιαζόταν στο ίδιο γράφημα, (π.χ. περιπτώσεις όπου ένα πρόβλημα είχε μέσο χρόνο μερικά milliseconds ενώ κάποιο άλλο χιλιάδες milliseconds), κρίθηκε σκόπιμο να αναπαρασταθεί λογαριθμικά ο χρόνος, ώστε να είναι ευδιάκριτες τόσο οι μικρές όσο και οι μεγάλες τιμές στο ίδιο γράφημα.

Τέλος, δίνονται δεκάδες γραφημάτων για την τυπική απόκλιση όπως και παραπάνω.Τα γραφήματα αυτά παρουσιάζουν χαρακτηριστική ομοιομορφία στο σύνολο των πειραμάτων ανά πρόβλημα. Όλα τα παραπάνω γραφήματα δίνονται στα Παραρτήματα Β, Γ και Δ.

Όλα τα πειράματα, τόσο για την εύρεση των παραμέτρων όσο και τα τελικά πειράματα, υλοποιήθηκαν σε μηχανές με Ubuntu Linux 14.04 και επεξεργαστές Intel Core i7 2.8GHz, 8GB RAM. Η χρονική διάρκεια του συνόλου των πειραμάτων ήταν περίπου 3 μήνες.

4.2. Σύγκριση αποτελεσμάτων ως προς την αποδοτικότητα

Με βάση τα στοιχεία που έχουμε συλλέξει από τα πειράματα, θα επιχειρήσουμε να αναλύσουμε τα δεδομένα που προέκυψαν. Συγκεκριμένα, θα επικεντρωθούμε στον συνολικό αριθμό των προβλημάτων που κατάφερε να επιλύσει ο κάθε αλγόριθμος, καθώς και στα ποσοστά επιτυχίας που είχε ανά πρόβλημα. Ο στόχος μας στην εν λόγω ανάλυση είναι να κατατάξουμε τους αλγορίθμους ανά πρόβλημα σε φθίνουσα σειρά με βάση τα ποσοστά επιτυχίας που είχαν. Με βάση τις κατατάξεις που θα προκύψουν, θα βγάλουμε ένα συγκεντρωτικό αποτέλεσμα για κάθε αλγόριθμο, το οποίο θα περιγράφει τον αριθμό των προβλημάτων όπου ήταν καλύτερος, χειρότερος ή ισοδύναμος με κάποιον ή κάποιους από τους υπόλοιπους.

Το πρώτο βήμα στην διαδικασία που ακολουθήσαμε ήταν η παραγωγή ενός πίνακα, ο οποίος περιέχει το σύνολο των προβλημάτων. Με βάση τα δεδομένα που συλλέξαμε ανά αλγόριθμο, θα κατατάξουμε σε φθίνουσα σειρά τους αλγορίθμους με βάση τα ποσοστά επιτυχίας για έκαστο πρόβλημα. Για την παραγωγή αυτού του πίνακα χρησιμοποιήθηκαν οι δέκα πίνακες κάθε αλγορίθμου του Παραρτήματος Α.

Τα αποτελέσματα που προέκυψαν δίνονται στους πίνακες του Παραρτήματος Ε. Σε αυτούς, κάθε κελί του πίνακα αποτελείται από δύο γραμμές. Η πρώτη γραμμή αφορά στην κατάταξη των αλγορίθμων ως προς τα ποσοστά επιτυχίας που είχαν στο συγκεκριμένο πρόβλημα. Η δεύτερη γραμμή αφορά στην κατάταξη των αλγορίθμων ως προς τον μέσο χρόνο επίλυσης του προβλήματος. Για παράδειγμα, η ακολουθία TS,TS-PR,IGD,RS,VNS δηλώνει ότι η TS έχει το μεγαλύτερο ποσοστό επιτυχίας, ακολουθεί η TS-PR κλπ. Επειδή βέβαια υπήρχαν και περιπτώσεις όπου δύο αλγόριθμοι είχαν ίδιο αριθμό επιτυχιών, τότε καταγράφονταν εντός παρένθεσης, π.χ. (TS,TS-PR),IGD,RS,VNS δηλαδή εδώ η TS και η TS-PR έχουν ίδιο αριθμό επιτυχιών, συνεπώς παίρνουν την πρώτη θέση και ακολουθούνται κατά σειρά από τους υπόλοιπους, όπου η IGD παίρνει την δεύτερη θέση κλπ. Στην περίπτωση που κάποιος αλγόριθμος δεν έδωσε λύση, δεν αναφέρεται το νούμερό του. Αντί αυτού, σημειώνεται μία παύλα. Έτσι π.χ. TS,---- σημαίνει ότι η TS κατατάσσεται πρώτη, ενώ όλοι οι υπόλοιποι αλγόριθμοι δεν βρήκαν λύση. Τέλος, σημειώνεται ότι στα

προβλήματα όπου όλοι οι αλγόριθμοι δεν βρήκαν λύση ή δεν υπάρχει γενικά λύση σημειώνεται στο κελί μία παύλα.

Με το τέλος της καταγραφής των αποτελεσμάτων αυτού του πίνακα, ακολούθησε το επόμενο βήμα, στο οποίο έγινε καταμέτρηση των εμφανίσεων του αλγορίθμου σε κάθε θέση κατάταξης. Σε πόσα προβλήματα π.χ. η TS βγήκε πρώτη στην κατάταξη, σε πόσα δεύτερη κλπ. Με την ολοκλήρωση αυτής της διαδικασίας συγκεντρώθηκαν τα τελικά στοιχεία, τα οποία και παρουσιάζονται στον πίνακα που ακολουθεί.

Alorithm Rank	**Tabu**	**Tabu (με Path Relinking)**	**Iterated Gradient Descent**	**Random Search**	**Variable Neighborhood Search**
1	171	164	156	148	119
2	3	4	9	7	30
3		2	4	4	6
4					4
Προβλήματα που επιλύθηκαν	174	170	169	159	159

Όπως φαίνεται από τον παραπάνω πίνακα, κάθε στήλη περιλαμβάνει τα αποτελέσματα ενός αλγορίθμου, ενώ σε κάθε γραμμή αναφέρεται ο αριθμός των προβλημάτων για κάθε αλγόριθμο, στα οποία βρέθηκε σε αυτή την θέση κατάταξης. Στην τελευταία γραμμή κάθε στήλης, αναφέρεται ο συνολικός αριθμός των προβλημάτων που κατάφερε να επιλύσει ο αντίστοιχος αλγόριθμος. Όπως είναι λογικό, το άθροισμα όλων των γραμμών μιας στήλης (εκτός της τελευταίας), μας δίνει τον συνολικό αριθμό των προβλημάτων τα οποία έλυσε ο αλγόριθμος. Εύλογα μπορεί να αναρωτηθεί κανείς βλέποντας τον πίνακα, γιατί το πλήθος των προβλημάτων σε κάθε γραμμή υπερβαίνει τον συνολικό αριθμό των προβλημάτων που ερευνήσαμε. Αυτό συμβαίνει διότι σε πολλά προβλήματα, όπως αναφέραμε και πιο πάνω, τα ποσοστά επιτυχίας των αλγορίθμων ήταν ίδια. Έτσι, στην κατάταξη που δημιουργούσαμε για εκείνο το πρόβλημα, οι αλγόριθμοι κατατάχθηκαν στην ίδια θέση. Επίσης όπως φαίνεται από τα δεδομένα του πίνακα δεν υπάρχει κανένας

αλγόριθμος που να κατατάσσεται στην πέμπτη θέση. Αυτό οφείλεται στο γεγονός ότι σε όλα τα προβλήματα που μελετήθηκαν υπήρχαν ισοβαθμίες στην κατάταξη.

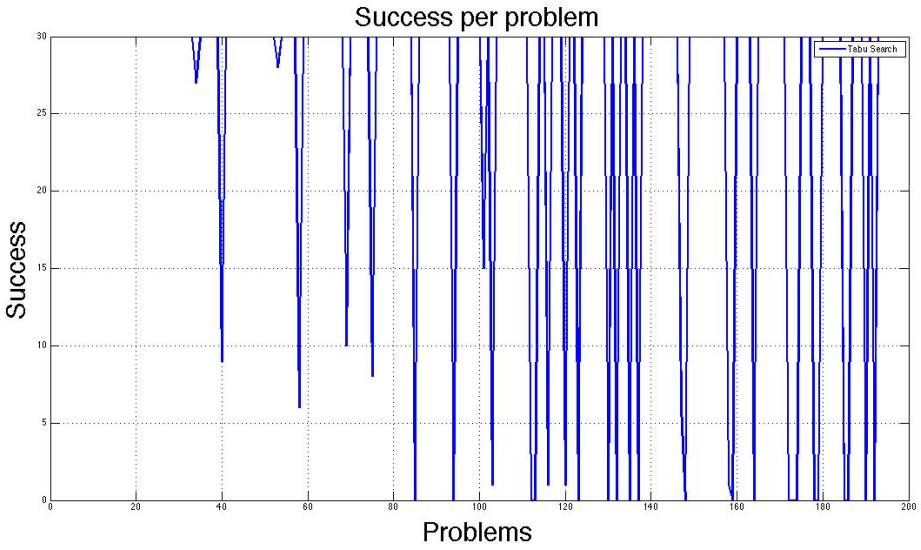
Από τα δεδομένα που προέκυψαν, φαίνεται ότι ο αλγόριθμος που επίλυσε τον μεγαλύτερο αριθμό προβλημάτων είναι η TS. Αμέσως μετά ακολουθεί η TS-PR και τρίτη η IGD. Οι άλλοι δύο αλγόριθμοι έχουν επιλύσει τον ίδιο ακριβώς αριθμό προβλημάτων. Είναι επίσης πολύ σημαντικό το γεγονός ότι η TS είναι και ο αλγόριθμος που κατατάχθηκε τις περισσότερες φορές στην πρώτη θέση. Αυτό σημαίνει ότι στα περισσότερα προβλήματα που ερευνήσαμε, η TS είχε το μεγαλύτερο ποσοστό επιτυχίας (επιτυχημένων πειραμάτων). Με μια πιο προσεκτική παρατήρηση των δεδομένων του πίνακα, μπορούμε να διακρίνουμε ότι η ίδια κατάταξη ισχύει και για τα ποσοστά επιτυχίας που είχαν οι αλγόριθμοι στα προβλήματα. Δηλαδή η TS είχε στα περισσότερα προβλήματα το μεγαλύτερο ποσοστό επιτυχίας, ακολουθεί η TS-PR, τρίτη η IGD, και τέλος η RS και η VNS. Αυτό δείχνει ότι οι αλγόριθμοι που κατάφεραν να λύσουν περισσότερα προβλήματα, ήταν και πιο αποδοτικοί γενικά στο σύνολο των προβλημάτων, έχοντας τα μεγαλύτερα ποσοστά επιτυχίας.

Με μια μελέτη των επιμέρους πινάκων αποτελεσμάτων για κάθε αλγόριθμο (πίνακες Παραρτήματος Α), φαίνεται ότι ο λόγος για τον οποίο οι τρεις πρώτοι αλγόριθμοι επίλυσαν μεγαλύτερο αριθμό προβλημάτων, είναι γιατί τα κατάφεραν καλύτερα σε προβλήματα μεγαλύτερης δυσκολίας. Το κοινό χαρακτηριστικό που έχουν οι τρεις πρώτοι αλγόριθμοι είναι ότι χρησιμοποιούν εξωτερική μνήμη. Αυτό έχει ως αποτέλεσμα να περιορίζεται αρκετά η πιθανότητα επίσκεψης στις ίδιες λύσεις κατά την αναζήτηση του αλγορίθμου. Επίσης βασικό χαρακτηριστικό και των τριών αλγορίθμων είναι το γεγονός ότι δημιουργούν μια καινούρια υποψήφια λύση από προηγούμενες ή προηγούμενη λύση, που επισκεύθηκε ο αλγόριθμος με αποτέλεσμα οι αναζητήσεις που γίνονται να έχουν μια συνέχεια. Τα δύο αυτά χαρακτηρηστικά τους φαίνεται να είναι πολύ σημαντικά στα προβλήματα μεγαλύτερης δυσκολίας, τα οποία μπορούν να επιλύσουν ευκολότερα οι τρεις αυτοί αλγόριθμοι.

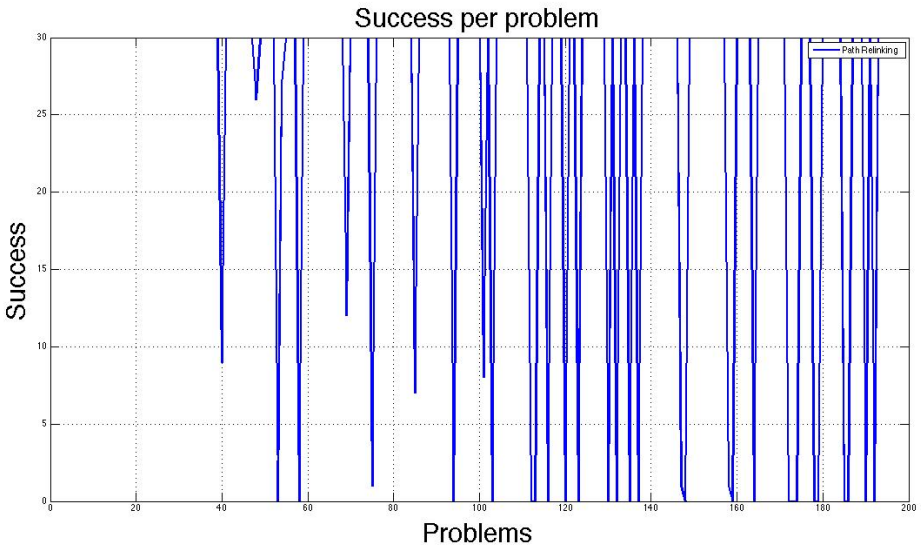
Στους τελευταίους δύο αλγορίθμους, φαίνεται να παίζει σημαντικότερο ρόλο η τυχαιότητα και η μη χρήση ιστορικού. Το γεγονός, δηλαδή, ότι η RS ψάχνει για τιμές σε διάφορες περιοχές του χώρου αναζήτησης λόγω διαρκών τυχαίων αρχικοποιήσεων, ενώ η VNS ψάχνει σε εναλλασόμενες γειτονιές. Και οι δύο βέβαια δεν κρατούν ιστορικό των προηγούμενων επισκέψεων και αυτό φαίνεται να επηρεάζει κατά πολύ την επίδοσή τους στα προβλήματα με μεγαλύτερη δυσκολία.

Αυτό φαίνεται και από τα ποσοστά επιτυχίας τους, όπως αυτά παρουσιάζονται στους πίνακες του Παραρτήματος Α.

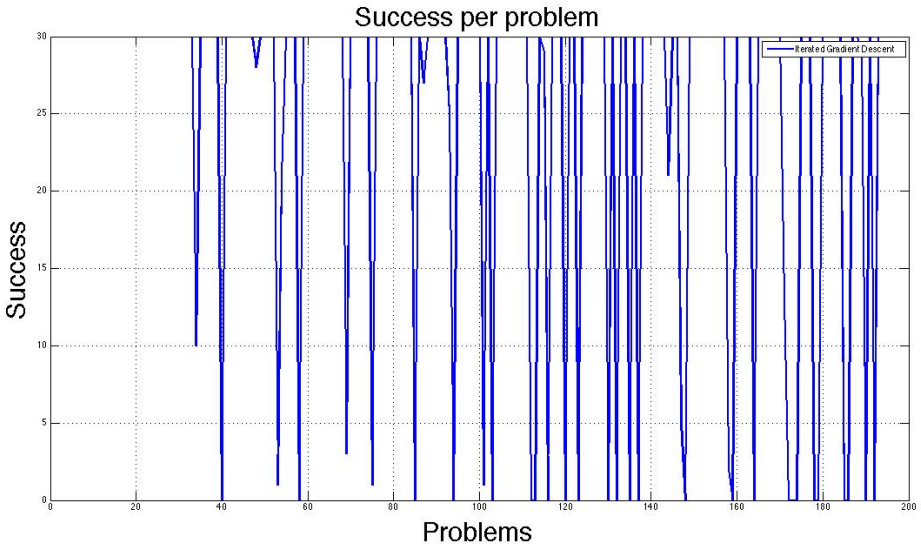
Ένα άλλο συμπέρασμα που προκύπτει από το σύνολο των πειραμάτων που έγιναν είναι ότι η απόδοση και των πέντε αλγορίθμων επιρρεάζεται περισσότερο από την αύξηση του βάρους *k* ενός προβλήματος παρά από την αύξηση της διάστασης *n* αυτού. Οι διακιμάνσεις της απόδοσης του αλγορίθμου σε συνάρτηση με την αύξηση της δυσκολίας των προβλημάτων *CW(n,k)* φαίνεται πιο αναλυτικά στα γραφήματα που ακολουθούν.



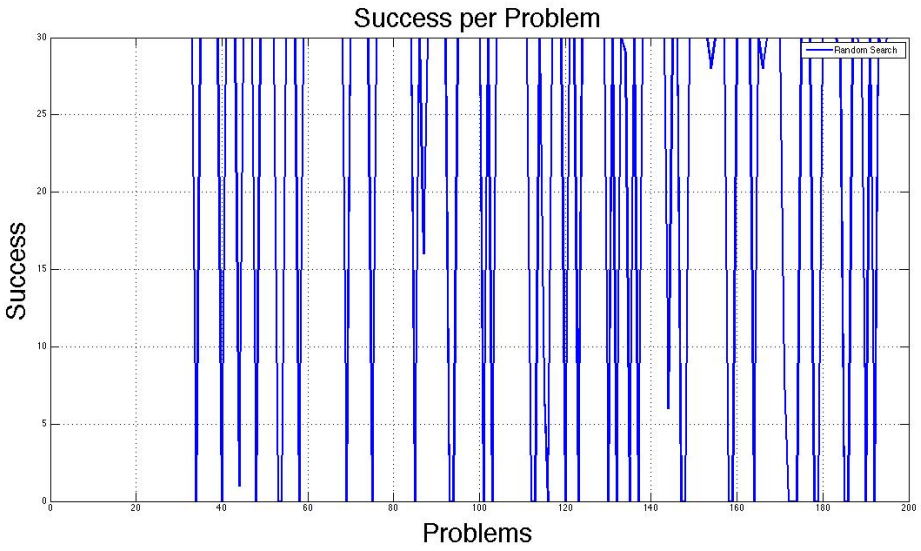
TS


Success per problem
Path Relinking
Success
Problems
0
5
10
15
20
25
30
0
20
40
60
80
100
120
140
160
180
200

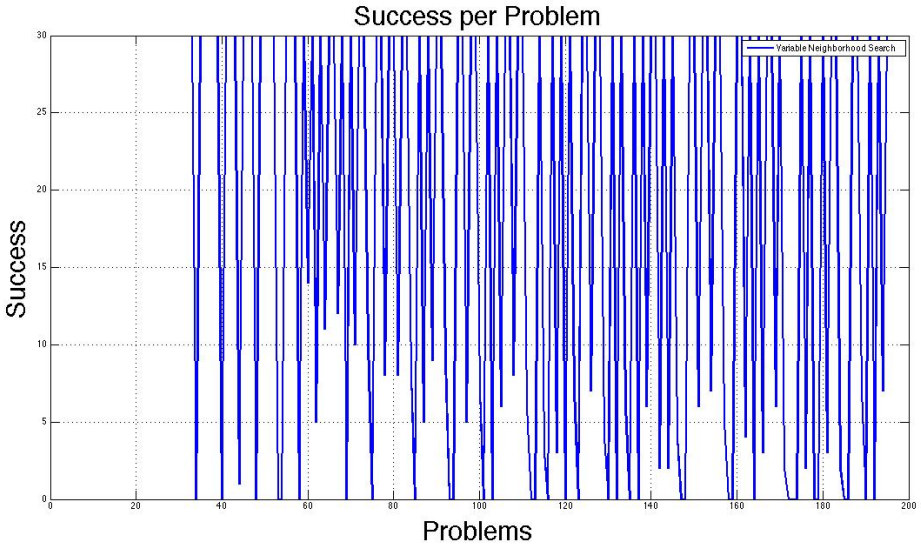
TS-PR


Success per problem
Iterated Gradient Descent
Success
Problems
0
5
10
15
20
25
30
0
20
40
60
80
100
120
140
160
180
200

IGD



RS



VNS

Στον κατακόρυφο άξονα των γραφημάτων αναπαριστούμε τον αριθμό των επιτυχημένων πειραμάτων με μέγιστο αριθμό τα 30 (δηλαδή 100% επιτυχία). Στον οριζόντιο άξονα αναπαριστούμε τα προβλήματα που ερευνήσαμε. Η σειρά διάταξης των προβλημάτων έγινε για κάθε *n* και για κάθε *k*. Έτσι στο οριζόντιο άξονα βρίσκονται τα προβλήματα που ερευνήσαμε κατά σειρά CW(1,1) ,.., CW(1,10), CW(2,1) ,…, CW(2,10) ,..., CW(101) ,.., CW(101,10).

Τα παραπάνω γραφήματα επιβεβαιώνουν τα συμπεράσμα που προέκυψαν από τους πίνακες που αναφέραμε παραπάνω.

4.3. Σύγκριση αποτελεσμάτων ως προς τον χρόνο επίλυσης

Ας επικεντρωθούμε τώρα στον μέσο χρόνο που χρειαζόταν ο κάθε αλγόριθμος για την επίλυση κάθε προβλήματος. Στόχος μας είναι να κατατάξουμε τους αλγορίθμους αρχικά ανά πρόβλημα, σε αύξουσα σειρά με βάση τον χρόνο επίλυσης (σε millisecond) που κατά μέσο όρο χρειαζόταν. Στη συνέχεια, με βάση τις κατατάξεις που προκύπτουν θα βγάλουμε ένα συνολικό αποτέλεσμα για κάθε αλγόριθμο, το οποίο περιγράφει τον αριθμό των προβλημάτων όπου ήταν ταχύτερος ή όχι με κάποιον ή κάποιους από τους υπόλοιπους. Τα αποτελέσματα θα δοθούν και σε αυτή την περίπτωση σε έναν πίνακα κατάταξης.

Για την παραγωγή αυτού του πίνακα χρησιμοποιήθηκαν και σε αυτή την ανάλυση οι δέκα πίνακες κάθε αλγορίθμου του Παραρτήματος Α. Εκτός από τον μέσο χρόνο επίλυσης, στους πίνακες του Παραρτήματος Α δίνονται επίσης και οι τυπικές αποκλίσεις των χρόνων που προέκυψαν από τα πειράματα. Επίσης, για την οπτικοποίηση των αποτελεσμάτων παρατίθενται τα αποτελέσματα αυτά και με την μορφή τρισδιάστατων ραμβρογραμμάτων στα γραφήματα του Παραρτήματος Β, τα οποία περιλαμβάνουν 10 γραφήματα μέσου χρόνου και 10 γραφήματα τυπικής απόκλισης ανά αλγόριθμο.

Τα αποτελέσματα που προέκυψαν δίνονται στους πίνακες του Παραρτήματος Ε. Ο τρόπος αναπαράστασης των αποτελεσμάτων σχετικά με την κατάταξη των αλγορίθμων, ακολουθεί την ίδια δομή με αυτή των αποτελεσμάτων αποδοτικότητας. Αυτή την φορά, τα αποτελέσματά μας βρίσκονται στην δεύτερη γραμμή των κελιών των πινάκων, και η κατάταξη γίνεται σε αύξουσα σειρά, μιας και εδώ κατατάσσεται πρώτος ο αλγόριθμος με τον μικρότερο μέσο χρόνο επίλυσης.

Επίσης, δίνεται παρακάτω ένας συγκεντρωτικός πίνακας για όλους του αλγορίθμους. Για αυτόν έγινε καταμέτρηση των εμφανίσεων κάθε αλγορίθμου στις θέσεις κατάταξης, δηλαδή σε πόσα προβλήματα π.χ. η RS βγήκε πρώτη στην κατάταξη (ήταν δηλαδή γρηγορότερη) σε πόσα δεύτερη κλπ.

Alorithm Rank	Tabu	Tabu (με Path Relinking)	Iterated Gradient Descent	Random Search	Variable Neighborhood Search
1	5	6	29	139	33
2	38	25	66	18	25
3	58	70	33	1	6
4	64	29	39	1	15
5	9	40	2	0	80

Όπως και στον αντίστοιχο πίνακα για την αποδοτικότητα των αλγορίθμων, έτσι και σε αυτόν τον πίνακα κάθε στήλη περιλαμβάνει τα αποτελέσματα ενός αλγορίθμου, ενώ σε κάθε γραμμή αναφέρεται ο αριθμός των προβλημάτων, στα οποία ο αντίστοιχος αλγόριθμος βρέθηκε σε αυτή την θέση κατάταξης.

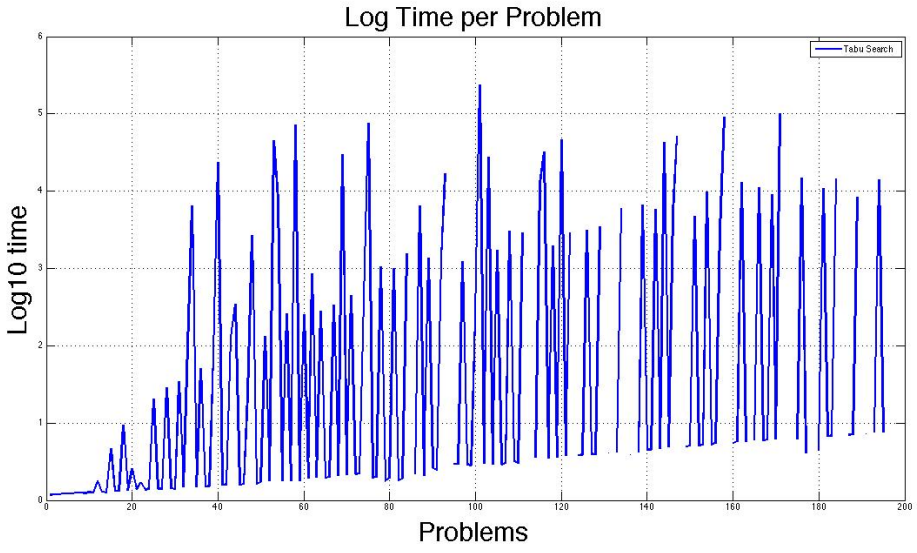
Εξετάζοντας τον παραπάνω πίνακα, παρατηρούμε ότι σε ορισμένες περιπτώσεις το πλήθος των προβλημάτων σε κάθε γραμμή υπερβαίνει τον συνολικό αριθμό των προβλημάτων που ερευνήσαμε. Αυτό συμβαίνει διότι πολλά προβλήματα βρέθηκαν να έχουν ίδιο πολύ μικρό χρόνο επίλυσης. Έτσι, στην κατάταξη για ένα τέτοιο πρόβλημα, οι αλγόριθμοι κατέλαβαν την ίδια θέση.

Από τα δεδομένα που προκύπτουν από τον παραπάνω πίνακα, φαίνεται ότι ο αλγόριθμος που ήταν γρηγορότερος στην εύρεση λύσης στα περισσότερα προβλήματα ήταν η RS. Δεύτερη στην κατάταξη φαίνεται να είναι η IGD, ενώ ακολουθούν οι TS-PR, TS, και VNS. Σύμφωνα με την κατάταξη που προκύπτει παρατηρούμε ότι στις δύο πρώτες θέσεις βρίσκονται οι RS και IGD. Ένα κοινό χαρακτηριστικό που έχουν οι δύο αυτοί αλγόριθμοι είναι το γεγονός ότι, αφενός στην RS, σε κάθε επανάληψη παίρνουμε ένα τυχαίο διάνυσμα, που δεν έχει κάποια σχέση με την προηγούμενη επανάληψη του αλγορίθμου, ενώ στην IGD όταν το διάνυσμα που βρίσκει ο αλγόριθμος από την προηγούμενη επανάληψη αλλάζει την πορεία του μονοπατιού που ακολουθούσε μέχρι εκείνη τη στιγμή προς τιμές χειρότερες, τότε παίρνουμε και εδώ ένα τυχαίο διάνυσμα. Δηλαδή, φαίνεται ότι η ιδιότητα των συγκεκριμένων αλγορίθμων να μην επικεντρώνονται σε μικρές περιοχές για μεγάλο αριθμό επαναλήψεων, αλλά να παίρνουν τυχαίες τιμές πιο συχνά, τους βοηθάει να βρούν γρηγορότερα λύση στο πρόβλημα. Το ίδιο συμπέρασμα μπορεί να βγει και για τους επόμενους αλγορίθμους στην κατάταξη, καθώς φαίνεται ότι όσο

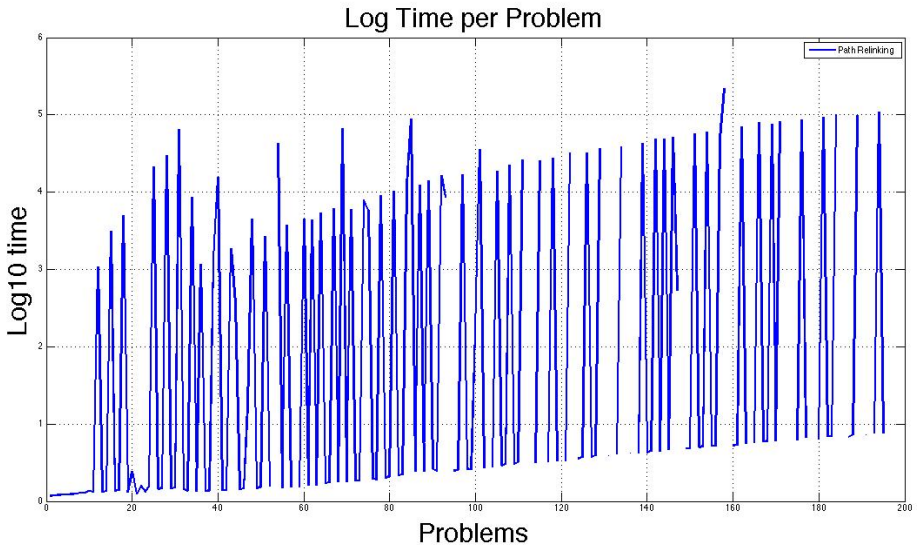
επικεντρωνόμαστε σε πιο μικρές περιοχές του χώρου αναζήτησης, τόσο αυτό μας οδηγεί στο να κάνουμε περισσότερο χρόνο να βρούμε λύση στο πρόβλημά μας.

Είναι χαρακτηριστικό το γεγονός ότι στην κατάταξη η TS-PR βρέθηκε σε καλύτερη θέση από την TS. Αν ανατρέξουμε στην ανάλυση των δύο μεθόδων, θα δούμε ότι η βασική διαφορά των δύο αλγορίθμων έγκειται στο γεγονός ότι η TS-PR κάνει τοπική αναζήτηση με χρήση της TS αλλά σε ένα νέο διάνυσμα που προκύπτει κάθε φορά από δύο προηγούμενα διανύσματα, ενώ η TS τροποποιεί κάθε φορά το ήδη υπάρχον διάνυσμα με τον τρόπο που ήδη αναλύσαμε. Οπότε και εδώ, σύμφωνα με τις τροποποιήσεις των αλγορίθμων που έχουν αναλυθεί στα προηγούμενα κεφάλαια, φαίνεται ότι ο αλγόριθμος που πιο συχνά ψάχνει για λύσεις σε νέες περιοχές αναζήτησης βελτιώνει την ταχύτητα εύρεσης λύσης. Από την ανάλυση αυτή γίνεται κατανοητός ο λόγος για τον οποίο η VNS βρίσκεται στην τελευταία θέση, μιας και είναι ο αλγόριθμος που από τη φύση του επικεντρώνεται σε αναζήτηση σε γειτονιές χωρίς να κρατάει ιστορικό, κάτι που θα μπορούσε να οδηγήσει σε επισκεψιμότητα σε ίδιες πιθανές λύσεις κάνοντας την διαδικασία πολύ πιο χρονοβόρα.

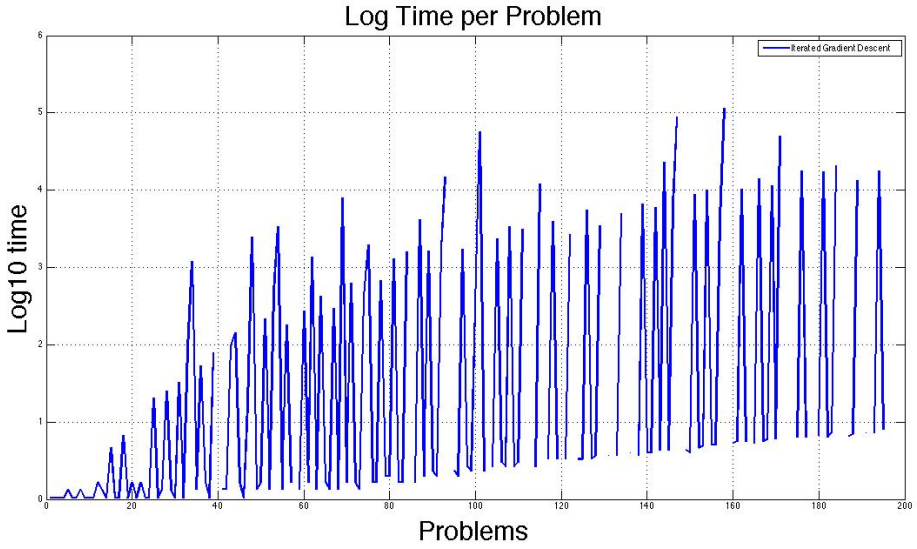
Τέλος δίνεται και εδώ ο μέσος απαιτούμενος χρόνος επίλυσης όσο τα προβλήματα γίνονται δυσκολότερα. Από τα αποτελέσματα που συλλέξαμε από τα πειράματα στο σύνολο των αλγορίθμων δημιουργήσαμε τα γραφήματα που ακολουθούν. Σε αυτά αναπαρίστανται οι διακιμάνσεις του μέσου χρόνου που χρειάστηκε ο κάθε αλγόριθμος σε συνάρτηση με την δυσκολία των προβλημάτων CW(n,k).



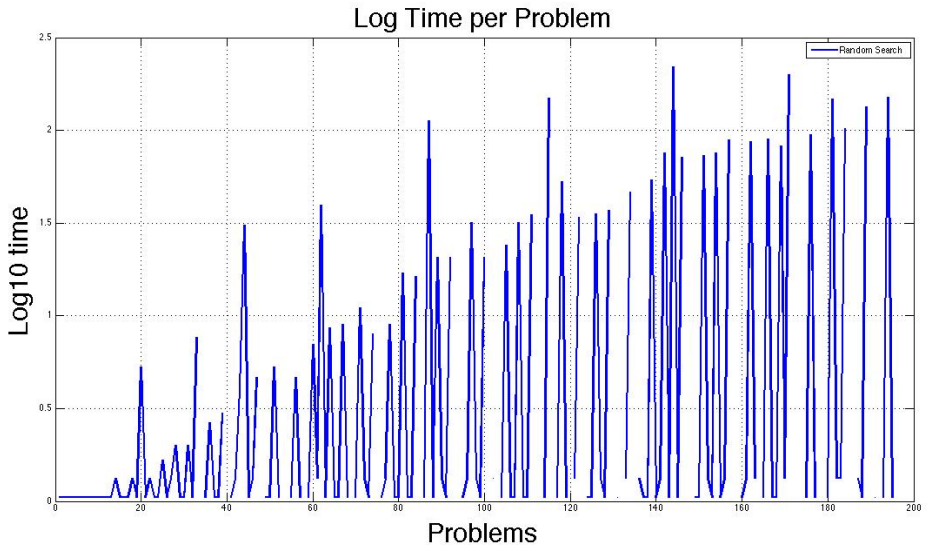
TS


Log Time per Problem
Path Relinking
Log10 time
Problems
0
1
2
3
4
5
6
0
20
40
60
80
100
120
140
160
180
200

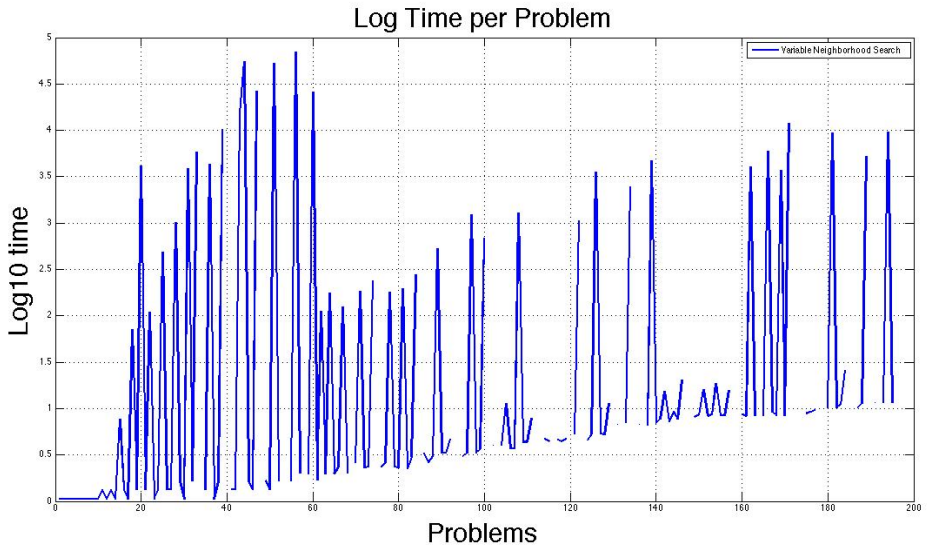
TS-PR


Log Time per Problem
Iterated Gradient Descent
Log10 time
Problems
0
1
2
3
4
5
6
0
20
40
60
80
100
120
140
160
180
200

IGD



RS



VNS

Στον κατακόρυφο άξονα των γραφημάτων αναπαριστούμε τον λογάριθμο του μέσου χρόνου επίλυσης των προβλημάτων. Ο λόγος που επιλέξαμε να δείχνουμε τον λογάριθμο είναι διότι οι χρόνοι είχαν πολύ μεγάλες διαφορές για διάφορα προβλήματα και έτσι οι μικρότερες τιμές δεν θα φαίνονταν στο γράφημα. Στον οριζόντιο άξονα αναπαριστούμε τα προβλήματα που ερευνήσαμε. Η σειρά διάταξης των προβλημάτων ακολουθεί την αντίστοιχη των γραφημάτων αποδοτικότητας.

Τα παραπάνω γραφήματα υποστηρίζουν πλήρως τις παρατηρήσεις που αναφέραμε παραπάνω.

4.4. Γενικά συμπεράσματα

Όπως φαίνεται από την ανάλυση που προηγήθηκε, τα γενικά συμπεράσματα που μπορούμε να εξάγουμε είναι ότι οι αλγόριθμοι που χρησιμοποιούν μνήμη σε συνδιασμό με τεχνικές παραγωγής νέων λύσεων από προηγούμενες, φαίνεται να έχουν μεγαλύτερες πιθανότητες να βρουν την λύση του προβλήματος όσο πιο δύσκολο γίνεται το πρόβλημα, αλλά στα πιο εύκολα προβλήματα τείνουν να είναι βραδύτεροι στην εύρεση της λύσης. Απεναντίας, καλύτερο χρόνο επίλυσης των προβλημάτων φαίνεται να παρουσιάζουν οι αλγόριθμοι που παίρνουν τυχαία διανύσματα πιο συχνά, ερευνώντας νέες περιοχές του χώρου αναζήτησης. Αυτό βέβαια οδηγεί σε μείωση των ποσοστών επιτυχίας στην εύρεση λύσης στα δυσκολότερα προβλήματα, ειδικά στην περίπτωση αλγορίθμων που δεν χρησιμοποιούν μνήμη.

Παράρτημα Α : Πίνακες Αποτελεσμάτων

Αποτελέσματα για την μέθοδο TS.

Πίνακας Α.1

N	k	1	2	3	4	5	6	7	8	9	10
1	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.177733 0.049843	-	-	-	-	-	-	-	-	-
2	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.184467 0.022242	-	-	-	-	-	-	-	-	-
3	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.189067 0.021844	-	-	-	-	-	-	-	-	-
4	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.219033 0.026119	0.215467 0.025970	-	-	-	-	-	-	-	-
5	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.219067 0.042916	-	-	-	-	-	-	-	-	-
6	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.2477 0.038190	0.248767 0.026093	-	-	-	-	-	-	-	-
7	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.231933 0.058179	0.274467 0.051734	-	-	-	-	-	-	-	-
8	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.2532 0.057810	0.7697 1.540237	-	-	-	-	-	-	-	-
9	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.2819 0.059623	-	-	-	-	-	-	-	-	-
10	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.2628 0.066279	3.616567 4.014182	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.2

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.3273 0.046885	-	-	-	-	-	-	-	-	-
12	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.321 0.045744	8.5193 6.680457	-	-	-	-	-	-	-	-
13	% suc	100%	-	100%	-	-	-	-	-	-	-
	Time avg/std	0.364833 0.084534	-	1.595767 1.100412	-	-	-	-	-	-	-
14	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3865 0.056812	0.683833 0.716674	-	-	-	-	-	-	-	-
15	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.374967 0.086685	-	-	-	-	-	-	-	-	-
16	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.405567 0.079418	19.50246 23.29294	-	-	-	-	-	-	-	-
17	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.409667 0.075416	-	-	-	-	-	-	-	-	-
18	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.4147 0.049314	27.438 31.79068	-	-	-	-	-	-	-	-
19	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.4242 0.067711	-	-	-	-	-	-	-	-	-
20	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3925 0.112182	33.1348 27.76687	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.3

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
21	% suc	100%	100%	-	90%	-	-	-	-	-	-
	Time avg/std	0.491 0.044981	100.7546 95.11283	-	6544.899 4915.827	-	-	-	-	-	-
22	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.497067 0.057448	49.69493 45.06583	-	-	-	-	-	-	-	-
23	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.503467 0.055063	-	-	-	-	-	-	-	-	-
24	% suc	100%	100%	30%	-	-	-	-	-	-	-
	Time avg/std	0.5216 0.101940	75.19039 83.95614	23586.46 15385.38	-	-	-	-	-	-	-
25	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.580267 0.099338	-	-	-	-	-	-	-	-	-
26	% suc	100%	100%	100%	-	-	-	-	-	-	-
	Time avg/std	0.605867 0.120887	129.6810 106.4900	347.7162 314.3701	-	-	-	-	-	-	-
27	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.584533 0.086900	-	-	-	-	-	-	-	-	-
28	% suc	100%	100%	-	100%	-	-	-	-	-	-
	Time avg/std	0.635733 0.132269	23.21066 15.89804	-	2667.724 2667.555	-	-	-	-	-	-
29	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.6528 0.070108	-	-	-	-	-	-	-	-	-
30	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.7296 0.089883	131.7845 177.2547	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.4

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
31	% suc	100%	-	-	93.3%	100%	-	-	-	-	-
	Time avg/std	0.785067 0.115142	-	-	44739.53 33276.96	7767.284 9131.430	-	-	-	-	-
32	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.776533 0.157426	255.5477 309.3289	-	-	C	-	-	-	-	-
33	% suc	100%	-	-	-	20%	-	-	-	-	-
	Time avg/std	0.785067 0.133332	-	-	-	72376.29 45155.60	-	-	-	-	-
34	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.810667 0.151575	249.2842 246.6899	-	-	-	-	-	-	-	-
35	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.913067 0.145490	849.2542 618.2367	-	-	-	-	-	-	-	-
36	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.964267 0.160271	277.7941 256.0004	-	-	-	-	-	-	-	-
37	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.955733 0.177021	-	-	-	-	-	-	-	-	-
38	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.9984 0.169411	332.2623 349.7115	-	-	-	-	-	-	-	-
39	% suc	100%	-	33.3%	-	-	-	-	-	-	-
	Time avg/std	1.092267 0.163759	-	30192.79 28714.07	-	-	-	-	-	-	-
40	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	1.160533 0.230285	444.9620 420.0531	-	-	C	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.5

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
41	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.1776 0.238638	-	-	-	-	-	-	-	-	-
42	% suc	100%	100%	-	26.6%	-	-	-	-	-	-
	Time avg/std	1.245866 0.290980	405.8282 550.1453	-	75903.23 40944.42	-	-	-	-	-	-
43	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.955733 0.533330	-	-	-	-	-	-	-	-	-
44	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.989867 0.569398	1064.482 1229.102	-	-	-	-	-	-	-	-
45	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.785067 0.516103	-	-	-	-	-	-	-	-	-
46	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	0.955733 0.533330	1004.714 1201.904	-	-	-	N	-	-	-	-
47	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.8192 0.564082	-	-	-	-	N	-	-	-	-
48	% suc	100%	100%	0%	-	C	C	-	-	-	-
	Time avg/std	0.9216 0.492196	1552.281 1305.329	- -	-	C	C	-	-	-	-
49	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.2288 1.153519	6533.598 7592.019	-	-	-	-	-	-	-	-
50	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.092267 1.039188	1354.683 1601.969	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.6

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
51	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.636233 0.216768	-	-	-	-	-	-	-	-	-
52	% suc	100%	100%	100%	-	-	0%	-	-	-	-
	Time avg/std	1.493133 0.369416	1354.469 1435.141	16681.22 18933.66	-	-	- -	-	-	-	-
53	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.911467 0.747823	-	-	-	-	-	-	-	-	-
54	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	1.911467 0.747823	1238.835 1381.934	-	-	-	-	C	-	-	-
55	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	1.911467 0.747823	-	-	-	-	N	-	-	-	-
56	% suc	100%	100%	-	50%	-	-	-	-	-	-
	Time avg/std	1.8432 0.624903	240.2304 223.5160	-	239641.3 182159.2	-	-	-	-	-	-
57	% suc	100%	-	-	-	-	-	3.3%	-	-	-
	Time avg/std	2.048 2.083011	-	-	-	-	-	27470.08 0	-	-	-
58	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.911467 2.078377	1728.648 1581.544	-	-	-	-	-	-	-	-
59	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.911467 2.078377	-	-	-	-	-	-	-	-	-
60	% suc	100%	100%	-	-	-	?	-	-	-	-
	Time avg/std	2.048 2.083011	3086.472 3276.714	-	-	-	?	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.7

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
61	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.184533 2.078377	-	-	-	-	-	-	-	-	-
62	% suc	100%	100%	-	0%	0%	-	-	-	-	-
	Time avg/std	2.048 2.083011	2872.934 2560.846	-	- -	- -	-	-	-	-	-
63	% suc	100%	100%	-	3.3%	-	-	?	-	-	-
	Time avg/std	2.563433 0.118023	12571.91 13850.64	-	32465.40 0	-	-	?	-	-	-
64	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	2.5504 0.165777	1959.769 1959.993	-	-	C	-	C	-	-	-
65	% suc	100%	-	3.3%	-	-	-	-	-	-	-
	Time avg/std	2.608 0.316532	-	46306.30 0	-	-	-	-	-	-	-
66	% suc	100%	100%	-	-	0%	-	-	-	-	-
	Time avg/std	2.788267 0.181940	2908.928 3094.021	-	-	- -	-	-	-	-	-
67	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.805334 0.373590	-	-	-	-	-	-	-	-	-
68	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	2.891733 0.170260	3114.850 3411.387	-	-	-	-	-	-	-	-
69	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	2.9632 0.106497	-	-	-	-	N	-	-	-	-
70	% suc	100%	100%	-	0%	-	C	-	N	-	-
	Time avg/std	2.939733 0.412353	3447.424 4802.963	-	-	-	C	-	N	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.8

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
71	% suc	100%	-	-	-	0%	N	-	-	-	-
	Time avg/std	3.114667 0.071964	-	-	-	-	N	-	-	-	-
72	% suc	100%	100%	0%	-	-	-	C	-	-	-
	Time avg/std	3.2448 0.157800	5913.815 4398.251	-	-	-	-	C	-	-	-
73	% suc	100%	-	-	-	-	-	-	0%	-	-
	Time avg/std	2.894933 0.697992	-	-	-	-	-	-	-	-	-
74	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	3.246933 0.544080	6651.911 5426.303	-	-	-	-	-	-	-	-
75	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.524267 0.124570	-	-	-	-	-	-	-	-	-
76	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	3.554133 0.376846	5795.263 4221.321	-	-	-	-	C	-	-	-
77	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	3.690666 0.487216	43057.68 48535.86	-	-	-	N	-	-	-	-
78	% suc	100%	100%	20%	-	-	0%	-	-	-	-
	Time avg/std	3.916799 0.214184	5456.896 4878.787	51037.69 31509.91	-	-	-	-	-	-	-
79	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.933866 0.142349	-	-	-	-	-	-	-	-	-
80	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	4.053333 0.308692	4725.290 5536.330	-	-	C	-	C	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.9

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
81	% suc	100%	-	-	-	-	-	N	-	-	-
	Time avg/std	4.096 0.164676	-	-	-	-	-	N	-	-	-
82	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4.309333 0.301282	9901.073 9788.451	-	-	-	-	-	-	-	-
83	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4.309333 0.594066	-	-	-	-	-	-	-	-	-
84	% suc	100%	100%	-	3.3%	-	-	-	0%	-	-
	Time avg/std	4.522667 0.295603	3204.727 2742.049	-	90634.24 0	-	-	-	-	-	-
85	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	4.48 0.174665	-	-	-	-	N	-	-	-	-
86	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4.753067 0.401595	12979.53 11552.63	-	-	-	-	-	-	-	-
87	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	4.787199 0.152566	-	-	-	-	-	-	-	-	-
88	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	4.991999 0.334459	11123.31 15597.68	-	-	C	-	-	-	-	-
89	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4.932266 0.115142	-	-	-	-	-	-	-	-	-
90	% suc	100%	100%	-	-	-	-	C	C	-	-
	Time avg/std	5.0688 0.123963	9102.174 7642.304	-	-	-	-	C	C	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS (συνέχεια).

Πίνακας Α.10

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
91	% suc	100%	100%	0%	-	-	0%	-	-	0%	-
	Time avg/std	5.1456 0.462611	99945.28 98374.30	-	-	-	-	-	-	- -	-
92	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	5.290668 0.452652	14825.71 9059.309	-	-	-	N	-	N	-	-
93	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	3.089067 0.678756	-	-	-	-	-	-	-	-	-
94	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	3.4304 0.738294	10791.08 8959.495	-	-	-	N	-	N	-	-
95	% suc	100%	-	-	-	-	N	C	-	-	-
	Time avg/std	5.785601 0.238638	-	-	-	-	N	C	-	-	-
96	% suc	100%	100%	0%	-	C	0%	C	-	-	-
	Time avg/std	5.870934 0.655036	14513.20 11255.24	-	-	C	-	C	-	-	-
97	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	6.024533 0.681414	-	-	-	-	-	-	-	-	-
98	% suc	100%	100%	-	0%	-	-	-	N	-	-
	Time avg/std	6.143999 0.232887	8356.897 9758.273	-	-	-	-	-	N	-	-
99	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	6.280532 0.354043	-	-	-	-	-	-	-	-	-
100	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	6.655999 0.698663	14162.29 14387.26	-	-	-	-	-	-	-	-
101	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	6.553599 0.690856	-	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k.)$

Αποτελέσματα για την μέθοδο TS-PR.

Πίνακας Α.11

N	K	1	2	3	4	5	6	7	8	9	10
1	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.169933 0.043229	-	-	-	-	-	-	-	-	-
2	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.1854 0.069324	-	-	-	-	-	-	-	-	-
3	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.2003 0.043350	-	-	-	-	-	-	-	-	-
4	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.2231 0.051110	0.215667 0.036837	-	-	-	-	-	-	-	-
5	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.233667 0.042033	-	-	-	-	-	-	-	-	-
6	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.251067 0.054777	0.289467 0.046837	-	-	-	-	-	-	-	-
7	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.290133 0.041964	0.3562 0.064129	-	-	-	-	-	-	-	-
8	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3248 0.070823	1089.492 1782.458	-	-	-	-	-	-	-	-
9	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.3218 0.091525	-	-	-	-	-	-	-	-	-
10	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3546 0.067259	3136.848 4713.854	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k.)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.12

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.376267 0.046099	-	-	-	-	-	-	-	-	-
12	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.394933 0.057974	4997.662 5799.555	-	-	-	-	-	-	-	-
13	% suc	100%	-	100%	-	-	-	-	-	-	-
	Time avg/std	0.296533 0.099053	-	1.424 1.058613	-	-	-	-	-	-	-
14	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.237867 0.058029	0.590933 0.444814	-	-	-	-	-	-	-	-
15	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.317867 0.058181	-	-	-	-	-	-	-	-	-
16	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.5504 0.080341	21173.80 22397.16	-	-	-	-	-	-	-	-
17	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.448 0.060599	-	-	-	-	-	-	-	-	-
18	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.4864 0.076283	30212.04 32575.75	-	-	-	-	-	-	-	-
19	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.469333 0.091028	-	-	-	-	-	-	-	-	-
20	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.4992 0.061524	65313.27 76287.69	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.13

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
21	% suc	100%	100%	-	100%	-	-	-	-	-	-
	Time avg/std	0.4352 0.152566	0.379733 0.061524	-	8600.305 4089.150	-	-	-	-	-	-
22	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3446 0.017578	1161.789 451.4081	-	-	-	-	-	-	-	-
23	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.358133 0.021902	-	-	-	-	-	-	-	-	-
24	% suc	100%	100%	30%	-	-	-	-	-	-	-
	Time avg/std	0.379733 0.017731	1375.206 612.4468	15498.06 10296.41	-	-	-	-	-	-	-
25	% suc	100%	-	-	- -	-	-	-	-	-	-
	Time avg/std	0.398933 0.091569	-	-	-	-	-	-	-	-	-
26	% suc	100%	100%	100%	-	-	-	-	-	-	-
	Time avg/std	0.388266 0.076898	1872.219 634.9259	361.1797 365.3936	-	-	-	-	-	-	-
27	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.433066 0.070676	-	-	-	-	-	-	-	-	-
28	% suc	100%	100%	-	86.6%	-	-	-	-	-	-
	Time avg/std	0.458367 0.054307	16.49253 11.33507	-	4537.620 8999.702	-	-	-	-	-	-
29	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.477867 0.057571	-	-	-	-	-	-	-	-	-
30	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.522667 0.085425	2690.457 1196.168	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.14

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
31	% suc	100%	-	-	0%	90%	-	-	-	-	-
	Time avg/std	0.5696 0.095594	-	-	-	42642.09 42203.97	-	-	-	-	-
32	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.477866 0.129898	3714.303 1023.614	-	-	C	-	-	-	-	-
33	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	0.546133 0.129898	-	-	-	-	-	-	-	-	-
34	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.537967 0.116831	4453.623 844.6324	-	-	-	-	-	-	-	-
35	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.631467 0.021002	4343.457 1121.948	-	-	-	-	-	-	-	-
36	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.629333 0.159521	5329.121 1011.961	-	-	-	-	-	-	-	-
37	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.706134 0.016666	-	-	-	-	-	-	-	-	-
38	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.754133 0.088010	6080.659 1571.586	-	-	-	-	-	-	-	-
39	% suc	100%	-	40%	-	-	-	-	-	-	-
	Time avg/std	0.7872 0.038141	-	65979.44 42826.39	-	-	-	-	-	-	-
40	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.776533 0.207031	5936.196 2959.349	-	-	C	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.15

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
41	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.853333 0.279896	-	-	-	-	-	-	-	-	-
42	% suc	100%	100%	-	3.3%	-	-	-	-	-	-
	Time avg/std	0.853333 0.245485	7896.763 2147.610	-	5816.768 0	-	-	-	-	-	-
43	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.937167 0.138957	-	-	-	-	-	-	-	-	-
44	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.900633 0.215328	9040.228 3607.725	-	-	-	-	-	-	-	-
45	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.035733 0.040723	-	-	-	-	-	-	-	-	-
46	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	1.0816 0.091054	10380.62 2827.636	-	-	-	N	-	-	-	-
47	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	1.147733 0.028785	-	-	-	-	N	-	-	-	-
48	% suc	100%	100%	23.3%	-	C	C	-	-	-	-
	Time avg/std	1.205333 0.033962	12330.44 3197.890	88041.32 54049.59	-	C	C	-	-	-	-
49	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.450666 1.188133	12422.02 3402.979	-	-	-	-	-	-	-	-
50	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.4336 0.843983	14149.35 2708.067	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.16

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
51	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.621333 1.067931	-	-	-	-	-	-	-	-	-
52	% suc	100%	100%	100%	-	-	0%	-	-	-	-
	Time avg/std	1.467733 0.259797	16570.88 337.6644	8521.745 12444.32	-	-	-	-	-	-	-
53	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.4885 0.028588	-	-	-	-	-	-	-	-	-
54	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	1.543033 0.170438	16905.06 4617.941	-	-	-	-	C	-	-	-
55	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	1.624533 0.026150	-	-	-	-	N	-	-	-	-
56	% suc	100%	100%	-	26.6%	-	-	-	-	-	-
	Time avg/std	1.5872 0.441377	158.3370 118.9111	-	35832.5 30355.46	-	-	-	-	-	-
57	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	1.739733 0.130589	-	-	-	-	-	-	-	-	-
58	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.778133 0.264504	18631.16 8299.552	-	-	-	-	-	-	-	-
59	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.926399 0.030762	-	-	-	-	-	-	-	-	-
60	% suc	100%	100%	-	-	-	?	-	-	-	-
	Time avg/std	2.011734 0.160051	22302.63 5802.869	-	-	-	?	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.17

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
61	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.069334 0.059025	-	-	-	-	-	-	-	-	-
62	% suc	100%	100%	-	0%	0%	-	-	-	-	-
	Time avg/std	2.244267 0.064949	26230.62 774.7128	-	-	-	-	-	-	-	-
63	% suc	100%	100%	-	0%	-	-	?	-	-	-
	Time avg/std	2.210134 0.304268	25831.68 7202.201	-	-	-	-	?	-	-	-
64	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	2.218667 0.357853	27388.01 5224.555	-	-	C	-	C	-	-	-
65	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	2.346667 0.252295	-	-	-	-	-	-	-	-	-
66	% suc	100%	100%	-	-	0%	-	-	-	-	-
	Time avg/std	2.304 0.455968	32470.35 691.8334	-	-	-	-	-	-	-	-
67	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.577067 0.093477	-	-	-	-	-	-	-	-	-
68	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	2.7136 0.238638	31950.20 8278.179	-	-	-	-	-	-	-	-
69	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	2.7136 0.273911	-	-	-	-	N	-	-	-	-
70	% suc	100%	100%	-	0%	-	C	-	N	-	-
	Time avg/std	2.901333 0.279896	36712.82 509.1959	-	-	-	C	-	N	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.18

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
71	% suc	100%	-	-	-	0%	N	-	-	-	-
	Time avg/std	2.884266 0.778037	-	-	-	-	N	-	-	-	-
72	% suc	100%	100%	0%	-	-	-	C	-	-	-
	Time avg/std	3.106133 0.186955	38997.53 12829.86	-	-	-	-	C	-	-	-
73	% suc	100%	-	-	-	-	-	-	0%	-	-
	Time avg/std	3.191466 0.372701	-	-	-	-	-	-	-	-	-
74	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	3.259733 0.284699	42398.58 11651.01	-	-	-	-	-	-	-	-
75	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.310933 0.516103	-	-	-	-	-	-	-	-	-
76	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	3.447466 0.501895	49131.93 2199.725	-	-	-	-	C	-	-	-
77	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	3.4816 0.636370	49002.52 9566.193	-	-	-	N	-	-	-	-
78	% suc	100%	100%	3.3%	-	-	0%	-	-	-	-
	Time avg/std	3.720534 0.501895	51198.43 9895.544	539.6480 0	-	-	-	-	-	-	-
79	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.857067 0.516103	-	-	-	-	-	-	-	-	-
80	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	3.857067 0.440507	57334.38 1755.669	-	-	C	-	C	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k.)$

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.19

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
81	% suc	100%	-	-	-	-	-	N	-	-	-
	Time avg/std	4.027734 0.460571	-	-	-	-	-	N	-	-	-
82	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4.164267 0.373911	59438.32 11435.09	-	-	-	-	-	-	-	-
83	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4.164267 0.373911	-	-	-	-	-	-	-	-	-
84	% suc	100%	100%	-	3.3%	-	-	-	0%	-	-
	Time avg/std	4.164267 1.003791	50714.96 27610.42	-	221509.6 0	-	-	-	-	-	-
85	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	4.300799 0.984392	-	-	-	-	N	-	-	-	-
86	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4.369067 1.287803	71016.85 3382.370	-	-	-	-	-	-	-	-
87	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	4.642133 0.921142	-	-	-	-	-	-	-	-	-
88	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	4.778666 0.981940	80270.19 15482.17	-	-	C	-	-	-	-	-
89	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4.915199 1.020462	-	-	-	-	-	-	-	-	-
90	% suc	100%	100%	-	-	-	-	C	C	-	-
	Time avg/std	5.051733 1.039188	76295.71 20200.78	-	-	-	-	C	C	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο TS-PR (συνέχεια).

Πίνακας Α.20

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
91	% suc	100%	100%	0%	-	-	0%	-	-	0%	-
	Time avg/std	5.051733 1.395600	81615.79 4405.716	-	-	-	-	-	-	-	-
92	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	5.188266 1.495647	86095.72 16596.55	-	-	-	N	-	N	-	-
93	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	5.666132 1.163920	-	-	-	-	-	-	-	-	-
94	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	5.393065 1.138797	93695.92 5769.030	-	-	-	N	-	N	-	-
95	% suc	100%	-	-	-	-	N	C	-	-	-
	Time avg/std	5.870932 1.039188	-	-	-	-	N	C	-	-	-
96	% suc	100%	100%	0%	-	C	0%	C	-	-	-
	Time avg/std	5.870932 0.889185	101844.7 4631.606	-	-	C	-	C	-	-	-
97	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	5.870933 2.064412	-	-	-	-	-	-	-	-	-
98	% suc	100%	100%	-	0%	-	-	-	N	-	-
	Time avg/std	6.280533 2.078377	97422.12 26675.97	-	-	-	-	-	N	-	-
99	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	6.280533 2.078377	-	-	-	-	-	-	-	-	-
100	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	6.553599 2.040925	110331.4 21266.59	-	-	-	-	-	-	-	-
101	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	6.553599 2.040925	-	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k.)$

Αποτελέσματα για την μέθοδο VNS.

Πίνακας Α.21

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
2	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
3	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
4	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
5	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
6	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
7	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
8	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	0.05 0	-	-	-	-	-	-	-	-
9	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
10	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	6.666667 24.11657	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.22

N	k	1	2	3	4	5	6	7	8	9	10
11	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
12	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	71 130.9474	-	-	-	-	-	-	-	-
13	% suc	100%	-	100%	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	4099.666 9064.026	-	-	-	-	-	-	-
14	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	108.3333 276.0070	-	-	-	-	-	-	-	-
15	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
16	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	490.6666 822.4938	-	-	-	-	-	-	-	-
17	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
18	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	1012 1702.015	-	-	-	-	-	-	-	-
19	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
20	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	3849.666 3653.169	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.23

N	K	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
21	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	5799.333 4809.900	-	-	-	-	-	-	-	-
22	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.3328 1.822820	4309.333 6188.026	-	-	-	-	-	-	-	-
23	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
24	% suc	100%	100%	0%	-	-	-	-	-	-	-
	Time avg/std	0.667733 2.541140	10216.33 11096.56	-	-	-	-	-	-	-	-
25	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.3328 1.822820	-	-	-	-	-	-	-	-	-
26	% suc	100%	100%	3.3%	-	-	-	-	-	-	-
	Time avg/std	0.3328 1.822820	15535.66 18063.70	55130.11 0	-	-	-	-	-	-	-
27	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.6656 2.533021	-	-	-	-	-	-	-	-	-
28	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.337067 1.846190	26761.33 29006.51	-	-	-	-	-	-	-	-
29	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.6656 2.533021	-	-	-	-	-	-	-	-	-
30	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.337067 1.846190	52369 43525.96	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.24

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
31	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	0.6656 2.533021	-	-	-	-	-	-	-	-	-
32	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.6656 2.533021	69296.67 65905.91	-	-	C	-	-	-	-	-
33	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	1.006933 3.073421	-	-	-	-	-	-	-	-	-
34	% suc	100%	46.6%	-	-	-	-	-	-	-	-
	Time avg/std	0.989867 3.021363	25667.87 65043.92	-	-	-	-	-	-	-	-
35	% suc	100%	16.6%	-	-	-	-	-	-	-	-
	Time avg/std	0.682667 2.597971	109.9776 245.9173	-	-	-	-	-	-	-	-
36	% suc	100%	36.6%	-	-	-	-	-	-	-	-
	Time avg/std	0.989867 3.024353	175.4763 301.0797	-	-	-	-	-	-	-	-
37	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.989867 3.024353	-	-	-	-	-	-	-	-	-
38	% suc	100%	40%	-	-	-	-	-	-	-	-
	Time avg/std	1.3312 3.455852	123.9040 285.5039	-	-	-	-	-	-	-	-
39	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	1.024 3.124516	-	-	-	-	-	-	-	-	-
40	% suc	100%	33.3%	-	-	C	-	-	-	-	-
	Time avg/std	1.6384 3.741697	182.0672 381.1511	-	-	C	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k.)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.25

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
41	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.297067 3.379503	-	-	-	-	-	-	-	-	-
42	% suc	100%	33.3%	-	0%	-	-	-	-	-	-
	Time avg/std	1.365333 3.540438	239.0016 501.3535	-	-	-	-	-	-	-	-
43	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.365333 3.540438	-	-	-	-	-	-	-	-	-
44	% suc	100%	26.6%	-	-	-	-	-	-	-	-
	Time avg/std	1.6384 4.607803	177.4079 501.7855	-	-	-	-	-	-	-	-
45	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.365333 3.540438	-	-	-	-	-	-	-	-	-
46	% suc	100%	26.6%	-	-	-	N	-	-	-	-
	Time avg/std	1.297067 3.379503	194.0479 532.3536	-	-	-	N	-	-	-	-
47	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	1.2288 3.253767	-	-	-	-	N	-	-	-	-
48	% suc	100%	26.6%	0%	-	C	C	-	-	-	-
	Time avg/std	2.048 4.268902	277.5039 771.6981	- -	-	C	C	-	-	-	-
49	% suc	100%	16.6%	-	-	-	-	-	-	-	-
	Time avg/std	2.321067 4.393568	1.6384 3.663573	-	-	-	-	-	-	-	-
50	% suc	100%	30%	-	-	-	-	-	-	-	-
	Time avg/std	2.048 4.268902	527.9288 1039.640	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.26

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
51	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.321066 4.393568	-	-	-	-	-	-	-	-	-
52	% suc	100%	30%	0%	-	-	0%	-	-	-	-
	Time avg/std	2.321066 4.393568	3.640888 5.587867	-	-	-	- -	-	-	-	-
53	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.048 4.268902	-	-	-	-	-	-	-	-	-
54	% suc	100%	16.6%	-	-	-	-	C	-	-	-
	Time avg/std	2.333333 4.393568	1233.996 1676.421	-	-	-	-	C	-	-	-
55	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	2.338133 4.312477	-	-	-	-	N	-	-	-	-
56	% suc	100%	23.3%	-	0%	-	-	-	-	-	-
	Time avg/std	2.6624 5.201274	682.8617 1793.368	-	-	-	-	-	-	-	-
57	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	3.003733 4.672738	-	-	-	-	-	-	-	-	-
58	% suc	100%	20%	-	-	-	-	-	-	-	-
	Time avg/std	3.003733 4.672738	10.24 6.476344	-	-	-	-	-	-	-	-
59	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.6624 4.496603	-	-	-	-	-	-	-	-	-
60	% suc	100%	26.6%	-	-	-	?	-	-	-	-
	Time avg/std	2.6624 4.496603	1299.968 2403.987	-	-	-	?	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.27

N	k	1	2	3	4	5	6	7	8	9	10
61	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.345067 4.817518	-	-	-	-	-	-	-	-	-
62	% suc	100%	20%	-	0%	0%	-	-	-	-	-
	Time avg/std	3.345066 4.817518	6.826666 5.287913	-	- -	- -	-	-	-	-	-
63	% suc	100%	10%	-	0%	-	-	?	-	-	-
	Time avg/std	5.287913 5.625415	3.413333 5.912066	-	-	-	-	?	-	-	-
64	% suc	100%	10%	-	-	C	-	C	-	-	-
	Time avg/std	3.618133 4.860855	3.413333 5.912066	-	-	C	-	C	-	-	-
65	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	3.6864 4.941026	-	-	-	-	-	-	-	-	-
66	% suc	100%	23.3%	-	-	0%	-	-	-	-	-
	Time avg/std	4.300799 5.026635	1055.305 2775.829	-	-	- -	-	-	-	-	-
67	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.549867 4.897906	-	-	-	-	-	-	-	-	-
68	% suc	100%	23.3%	-	-	-	-	-	-	-	-
	Time avg/std	4.369067 5.885095	3569.956 4490.142	-	-	-	-	-	-	-	-
69	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	4.369067 5.262323	-	-	-	-	N	-	-	-	-
70	% suc	100%	13.3%	-	0%	-	C	-	N	-	-
	Time avg/std	4.232534 5.100424	10.24 2.364826	-	-	-	C	-	N	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.28

N	k	1	2	3	4	5	6	7	8	9	10
71	% suc	100%	-	-	-	0%	N	-	-	-	-
	Time avg/std	5.7344 5.950269	-	-	-	-	N	-	-	-	-
72	% suc	100%	20%	0%	-	-	-	C	-	-	-
	Time avg/std	6.007467 6.430025	2469.887 3812.242	-	-	-	-	C	-	-	-
73	% suc	100%	-	-	-	-	-	-	0%	-	-
	Time avg/std	5.7344 5.950269	-	-	-	-	-	-	-	-	-
74	% suc	100%	20%	-	-	-	-	-	-	-	-
	Time avg/std	5.597867 5.212617	4694.698 5329.635	-	-	-	-	-	-	-	-
75	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	6.007467 5.664700	-	-	-	-	-	-	-	-	-
76	% suc	100%	6.6%	-	-	-	-	C	-	-	-
	Time avg/std	6.690133 6.130646	14.336 8.688928	-	-	-	-	C	-	-	-
77	% suc	100%	6.6%	-	-	-	N	-	-	-	-
	Time avg/std	6.280533 6.865164	8.192 0	-	-	-	N	-	-	-	-
78	% suc	100%	13.3%	0%	-	-	0%	-	-	-	-
	Time avg/std	6.690133 6.316559	19.45600 15.46206	-	-	-	-	-	-	-	-
79	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	7.099733 6.265982	-	-	-	-	-	-	-	-	-
80	% suc	100%	20%	-	-	C	-	C	-	-	-
	Time avg/std	7.645866 7.031683	15.01866 6.688740	-	-	C	-	C	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.29

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
81	% suc	100%	-	-	-	-	-	N	-	-	-
	Time avg/std	7.372801 6.216546	-	-	-	-	-	N	-	-	-
82	% suc	100%	23.3%	-	-	-	-	-	-	-	-
	Time avg/std	7.645867 7.431677	17.55428 7.370632	-	-	-	-	-	-	-	-
83	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	7.372801 6.921116	-	-	-	-	-	-	-	-	-
84	% suc	100%	16.6%	-	0%	-	-	-	0%	-	-
	Time avg/std	7.372801 6.921116	14.74560 3.663574	-	-	-	-	-	-	-	-
85	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	7.645867 7.431677	-	-	-	-	N	-	-	-	-
86	% suc	100%	13.3%	-	-	-	-	-	-	-	-
	Time avg/std	7.372801 7.860435	4048.895 8081.407	-	-	-	-	-	-	-	-
87	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	7.372801 6.578271	-	-	-	-	-	-	-	-	-
88	% suc	100%	10%	-	-	C	-	-	-	-	-
	Time avg/std	7.372801 6.578271	5977.429 10331.92	-	-	C	-	-	-	-	-
89	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	8.192 7.135136	-	-	-	-	-	-	-	-	-
90	% suc	100%	20%	-	-	-	-	C	C	-	-
	Time avg/std	7.372801 6.216546	3713.707 9040.503	-	-	-	-	C	C	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο VNS (συνέχεια).

Πίνακας Α.30

N	k	1	2	3	4	5	6	7	8	9	10
91	% suc	100%	6.6%	0%	-	-	0%	-	-	0%	-
	Time avg/std	7.372801 7.560307	11862.01 16752.25	-	-	-	-	-	-	- -	-
92	% suc	100%	6.6%	-	-	-	N	-	N	-	-
	Time avg/std	7.918934 5.477787	8.192 0	-	-	-	N	-	N	-	-
93	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	8.738133 6.780371	-	-	-	-	-	-	-	-	-
94	% suc	100%	10%	-	-	-	N	-	N	-	-
	Time avg/std	9.284267 7.676743	9404.416 16253.45	-	-	-	N	-	N	-	-
95	% suc	100%	-	-	-	-	N	C	-	-	-
	Time avg/std	9.284266 5.982591	-	-	-	-	N	C	-	-	-
96	% suc	100%	6.6%	0%	-	C	0%	C	-	-	-
	Time avg/std	10.10346 8.519716	24.576 0	-	-	C	-	C	-	-	-
97	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	9.284266 7.972489	-	-	-	-	-	-	-	-	-
98	% suc	100%	23.3%	-	0%	-	-	-	N	-	-
	Time avg/std	10.37653 8.852729	5285.010 13925.02	-	-	-	-	-	N	-	-
99	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	10.64959 8.092527	-	-	-	-	-	-	-	-	-
100	% suc	100%	23.3%	-	-	-	-	-	-	-	-
	Time avg/std	10.6496 6.507534	9567.085 16389.74	-	-	-	-	-	-	-	-
101	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	10.64959 6.141645	-	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD.

Πίνακας Α.31

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
2	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
3	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
4	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.333333 1.825741	-	-	-	-	-	-	-	-
5	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
6	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.333333 1.825741	-	-	-	-	-	-	-	-
7	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
8	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.666667 2.537081	-	-	-	-	-	-	-	-
9	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
10	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	3.666667 6.686751	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.32

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
12	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	5.666667 8.976341	-	-	-	-	-	-	-	-
13	% suc	100%	-	100%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	0.666667 2.537081	-	-	-	-	-	-	-
14	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.666667 2.537081	-	-	-	-	-	-	-	-
15	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
16	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	19.66666 28.82567	-	-	-	-	-	-	-	-
17	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
18	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	24.33333 33.28905	-	-	-	-	-	-	-	-
19	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
20	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	31.66666 51.66759	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.33

N	k	1	2	3	4	5	6	7	8	9	10
21	% suc	100%	100%	-	33.3%	-	-	-	-	-	-
	Time avg/std	0.05 0	92 115.1131	-	1191 726.7347	-	-	-	-	-	-
22	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	52.33333 59.17148	-	-	-	-	-	-	-	-
23	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
24	% suc	100%	100%	0%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	77 86.62841	-	-	-	-	-	-	-	-
25	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
26	% suc	100%	100%	100%	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	96.66666 109.7436	142.6666 139.7518	-	-	-	-	-	-	-
27	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
28	% suc	100%	100%	-	93.3%	-	-	-	-	-	-
	Time avg/std	0.05 0	16 12.48447	-	2495.357 1711.718	-	-	-	-	-	-
29	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
30	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	218 198.9524	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.34

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
31	% suc	100%	-	-	3.3%	70%	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	480 0	3390.952 2668.488	-	-	-	-	-
32	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.333333 1.825741	179 200.6601	-	-	C	-	-	-	-	-
33	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
34	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	273.6666 247.2257	-	-	-	-	-	-	-	-
35	% suc	100%	100%	-		-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	1379.333 1067.129	-	-	-	-	-	-	-	-
36	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	419.3333 471.7387	-	-	-	-	-	-	-	-
37	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
38	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	297.3333 278.0961	-	-	-	-	-	-	-	-
39	% suc	100%	-	10%	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	7903.316 4762.160	-	-	-	-	-	-	-
40	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.666667 2.537081	625.3333 516.5918	-	-	C	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.35											
N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
41	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
42	% suc	100%	100%	-	3.3%	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	426 563.8329	-	1979.967 0	-	-	-	-	-	-
43	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	-	-	-	-	-
44	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	680 946.0115	-	-	-	-	-	-	-	-
45	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1 3.051285	-	-	-	-	-	-	-	-	-
46	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	1 3.051285	1283 1191.672	-	-	-	N	-	-	-	-
47	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.666667 2.537081	-	-	-	-	N	-	-	-	-
48	% suc	100%	100%	0%	-	C	C	-	-	-	-
	Time avg/std	0.666667 2.537081	1596.333 1361.549	-	-	C	C	-	-	-	-
49	% suc	100%	90%	-	-	-	-	-	-	-	-
	Time avg/std	0.666667 2.537081	4180.740 3138.243	-	-	-	-	-	-	-	-
50	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.999467 3.049662	1619.666 2042.875	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.36

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
51	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.333333 3.457463	-	-	-	-	-	-	-	-	-
52	% suc	100%	100%	83.3%	-	-	0%	-	-	-	-
	Time avg/std	1.000533 3.052916	1732.667 1496.727	14978.79 11559.87	-	-	-	-	-	-	-
53	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.333333 3.457463	-	-	-	-	-	-	-	-	-
54	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	0.999467 3.049662	1700.333 1784.285	-	-	-	-	C	-	-	-
55	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	1.666133 3.789291	-	-	-	-	N	-	-	-	-
56	% suc	100%	100%	-	3.3%	-	-	-	-	-	-
	Time avg/std	1.333333 3.457474	306.6666 334.4492	-	56649.98 0	-	-	-	-	-	-
57	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	1.333333 3.457474	-	-	-	-	-	-	-	-	-
58	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	1.666133 3.789291	2330.000 2555.866	-	-	-	-	-	-	-	-
59	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.998933 4.066225	-	-	-	-	-	-	-	-	-
60	% suc	100%	100%	-	-	-	?	-	-	-	-
	Time avg/std	1.668267 3.794151	3365.999 3980.411	-	-	-	?	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.37

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
61	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	1.666133 3.789291	-	-	-	-	-	-	-	-	-
62	% suc	100%	100%	-	0%	0%	-	-	-	-	-
	Time avg/std	2.001067 4.070573	3122.666 3518.160	-	-	-	-	-	-	-	-
63	% suc	100%	96.6%	-	0%	-	-	?	-	-	-
	Time avg/std	1.666133 3.789291	12034.82 9189.811	-	-	-	-	?	-	-	-
64	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	2.3296 4.294947	3957.666 5237.684	-	-	C	-	C	-	-	-
65	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	2.333867 4.302869	-	-	-	-	-	-	-	-	-
66	% suc	100%	100%	-	-	0%	-	-	-	-	-
	Time avg/std	2.333867 4.302869	2646.335 2348.576	-	-	-	-	-	-	-	-
67	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.333867 4.302869	-	-	-	-	-	-	-	-	-
68	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	2.3296 4.294947	5575.666 6117.585	-	-	-	-	-	-	-	-
69	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	2.333867 4.302869	-	-	-	-	N	-	-	-	-
70	% suc	100%	100%	-	0%	-	C	-	N	-	-
	Time avg/std	2.666667 4.497819	3466.000 3107.002	-	-	-	C	-	N	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.38

N	k	1	2	3	4	5	6	7	8	9	10
71	% suc	100%	-	-	-	0%	N	-	-	-	-
	Time avg/std	2.666667 4.497819	-	-	-	-	N	-	-	-	-
72	% suc	100%	100%	0%	-	-	-	C	-	-	-
	Time avg/std	2.666667 4.497819	5040.998 5019.262	-	-	-	-	C	-	-	-
73	% suc	100%	-	-	-	-	-	-	0%	-	-
	Time avg/std	2.999467 4.660140	-	-	-	-	-	-	-	-	-
74	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	2.670933 4.505054	6560.333 6143.021	-	-	-	-	-	-	-	-
75	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	2.999467 4.660140	-	-	-	-	-	-	-	-	-
76	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	2.999467 4.660140	6032.665 5525.190	-	-	-	-	C	-	-	-
77	% suc	100%	70%	-	-	-	N	-	-	-	-
	Time avg/std	3.336533 4.799447	23073.32 18051.61	-	-	-	N	-	-	-	-
78	% suc	100%	100%	16.6%	-	-	0%	-	-	-	-
	Time avg/std	3.328 4.786961	6584.337 6911.773	88804.14 38287.45	-	-	-	-	-	-	-
79	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	3.336533 4.799447	-	-	-	-	-	-	-	-	-
80	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	3 4.893482	8893.333 7289.604	-	-	C	-	C	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.39

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
81	% suc	100%	-	-	-	-	-	N	-	-	-
	Time avg/std	3.666667 4.893482	-	-	-	-	-	N	-	-	-
82	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4 4.905098	9992.332 5860.392	-	-	-	-	-	-	-	-
83	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4 4.985593	-	-	-	-	-	-	-	-	-
84	% suc	100%	100%	-	6%	-	-	-	0%	-	-
	Time avg/std	4 4.985593	4372.667 5336.404	-	115524.9 150988.4	-	-	-	-	-	-
85	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	4.3328 5.042136	-	-	-	-	N	-	-	-	-
86	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	4.6656 4.985593	10302.66 19151.49	-	-	-	-	-	-	-	-
87	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	4.6656 5.069165	-	-	-	-	-	-	-	-	-
88	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	4.3328 5.685089	13905.00 13037.67	-	-	C	-	-	-	-	-
89	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	4.663466 5.070734	-	-	-	-	-	-	-	-	-
90	% suc	100%	100%	-	-	-	-	C	C	-	-
	Time avg/std	5.000533 5.086114	11314.00 10803.92	-	-	-	-	C	C	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο IGD (συνέχεια).

Πίνακας Α.40

N	k	1	2	3	4	5	6	7	8	9	10
91	% suc	100%	43.3%	0%	-	-	0%	-	-	0%	-
	Time avg/std	5.000533 5.086114	49456.16 29883.50	-	-	-	-	-	-	-	-
92	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	5.333333 5.074370	17952.00 13068.87	-	-	-	N	-	N	-	-
93	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	5.3248 5.707095	-	-	-	-	-	-	-	-	-
94	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	5.666133 5.043391	17165.34 21608.06	-	-	-	N	-	N	-	-
95	% suc	100%	-	-	-	-	N	C	-	-	-
	Time avg/std	5.3248 5.069611	-	-	-	-	N	C	-	-	-
96	% suc	100%	100%	0%	-	C	0%	C	-	-	-
	Time avg/std	6.331733 4.904454	20689.33 19715.67	-	-	C	-	C	-	-	-
97	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	5.666133 5.043391	-	-	-	-	-	-	-	-	-
98	% suc	100%	100%	-	0%	-	-	-	N	-	-
	Time avg/std	5.9904 4.978843	13278.63 11655.03	-	-	-	-	-	N	-	-
99	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	6.331733 4.904454	-	-	-	-	-	-	-	-	-
100	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	6.331733 5.560903	17604.67 14965.11	-	-	-	-	-	-	-	-
101	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	6.997333 7.016755	-	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS.

Πίνακας Α.41

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
2	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
3	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
4	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
5	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
6	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
7	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
8	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.05 0	-	-	-	-	-	-	-	-
9	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
10	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	0.05 0	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.42

N	k	1	2	3	4	5	6	7	8	9	10
11	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
12	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.333333 1.825741	-	-	-	-	-	-	-	-
13	% suc	100%	-	100%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	4.333333 6.260623	-	-	-	-	-	-	-
14	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.333333 1.825741	-	-	-	-	-	-	-	-
15	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
16	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	0.666667 2.537081	-	-	-	-	-	-	-	-
17	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
18	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	1 3.051285	-	-	-	-	-	-	-	-
19	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
20	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	1 3.051285	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.43

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
21	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.05 0	6.666667 8.441822	-	-	-	-	-	-	-	-
22	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	1.666667 3.790490	-	-	-	-	-	-	-	-
23	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
24	% suc	100%	100%	0%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	2 4.068381	-	-	-	-	-	-	-	-
25	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
26	% suc	100%	100%	3.3%	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	3 5.349830	30 0	-	-	-	-	-	-	-
27	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
28	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	3.666667 6.149478	-	-	-	-	-	-	-	-
29	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
30	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	4.333333 5.683207	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.44

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
31	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
32	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.05 0	3.666667 4.901325	-	-	C	-	-	-	-	-
33	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
34	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	6 6.214554	-	-	-	-	-	-	-	-
35	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	38.66666 35.49971	-	-	-	-	-	-	-	-
36	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	7.666667 6.789105	-	-	-	-	-	-	-	-
37	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
38	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	8 8.051557	-	-	-	-	-	-	-	-
39	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
40	% suc	100%	100%	-	-	C	-	-	-	-	-
	Time avg/std	0.05 0	10 10.17095	-	-	C	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.45

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
41	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
42	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.05 0	7 8.769067	-	-	-	-	-	-	-	-
43	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
44	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	8 8.866830	-	-	-	-	-	-	-	-
45	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
46	% suc	100%	100%	-	-	-	N	-	-	-	-
	Time avg/std	0.05 0	16 17.14039	-	-	-	N	-	-	-	-
47	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	N	-	-	-	-
48	% suc	100%	100%	0%	-	C	C	-	-	-	-
	Time avg/std	0.05 0	15.33333 13.06042	-	-	C	C	-	-	-	-
49	% suc	100%	53.5%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	111.875 47.07706	-	-	-	-	-	-	-	-
50	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	19.66666 18.84296	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.46

N	k	1	2	3	4	5	6	7	8	9	10
51	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
52	% suc	100%	100%	0%	-	-	0%	-	-	-	-
	Time avg/std	0.05 0	19.66666 15.19603	-	-	-	-	-	-	-	-
53	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
54	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	0.333333 1.825741	31 28.44838	-	-	-	-	C	-	-	-
55	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	N	-	-	-	-
56	% suc	100%	100%	-	0%	-	-	-	-	-	-
	Time avg/std	0.05 0	19.66666 12.72611	-	-	-	-	-	-	-	-
57	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
58	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	23 19.14554	-	-	-	-	-	-	-	-
59	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
60	% suc	100%	100%	-	-	-	?	-	-	-	-
	Time avg/std	0.05 0	31 30.32581	-	-	-	?	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.47

N	k	1	2	3	4	5	6	7	8	9	10
61	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
62	% suc	100%	100%	-	0%	0%	-	-	-	-	-
	Time avg/std	0.05 0	34 29.43139	-	-	-	-	-	-	-	-
63	% suc	100%	23.3%	-	0%	-	-	?	-	-	-
	Time avg/std	0.05 0	148.5714 80.08923	-	-	-	-	?	-	-	-
64	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	0.05 0	52 43.97491	-	-	C	-	C	-	-	-
65	% suc	100%	-	0%	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
66	% suc	100%	100%	-	-	0%	-	-	-	-	-
	Time avg/std	0.333333 1.825741	33 25.20946	-	-	-	-	-	-	-	-
67	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
68	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	34.33333 29.67447	-	-	-	-	-	-	-	-
69	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	N	-	-	-	-
70	% suc	100%	100%	-	0%	-	C	-	N	-	-
	Time avg/std	0.05 0	36 44.30303	-	-	-	C	-	N	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.48

N	k	1	2	3	4	5	6	7	8	9	10
71	% suc	100%	-	-	-	0%	N	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	N	-	-	-	-
72	% suc	100%	96.6%	0%	-	-	-	C	-	-	-
	Time avg/std	0.333333 1.825741	45.51724 34.59832	-	-	-	-	C	-	-	-
73	% suc	100%	-	-	-	-	-	-	0%	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
74	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	53 66.28829	-	-	-	-	-	-	-	-
75	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
76	% suc	100%	100%	-	-	-	-	C	-	-	-
	Time avg/std	0.333333 1.825741	75 67.91221	-	-	-	-	C	-	-	-
77	% suc	100%	20%	-	-	-	N	-	-	-	-
	Time avg/std	0.05 0	218.3333 119.0658	-	-	-	N	-	-	-	-
78	% suc	100%	100%	0%	-	-	0%	-	-	-	-
	Time avg/std	0.05 0	70.33333 82.98247	-	-	-	-	-	-	-	-
79	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
80	% suc	100%	100%	-	-	C	-	C	-	-	-
	Time avg/std	0.05 0	72.33333 72.38228	-	-	C	-	C	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Αποτελέσματα για την μέθοδο RS (συνέχεια).

Πίνακας Α.49

N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
81	% suc	100%	-	-	-	-	-	N	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	N	-	-	-
82	% suc	100%	93.3%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	75 64.31980	-	-	-	-	-	-	-	-
83	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
84	% suc	100%	100%	-	0%	-	-	-	0%	-	-
	Time avg/std	0.333333 1.825741	88 92.63945	-	-	-	-	-	-	-	-
85	% suc	100%	-	-	-	-	N	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	N	-	-	-	-
86	% suc	100%	100%	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	86.33333 81.97911	-	-	-	-	-	-	-	-
87	% suc	100%	-	-	-	-	-	0%	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
88	% suc	100%	93.3%	-	-	C	-	-	-	-	-
	Time avg/std	0.05 0	88.92857 69.83256	-	-	C	-	-	-	-	-
89	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
90	% suc	100%	100%	-	-	-	-	C	C	-	-
	Time avg/std	0.05 0	81.33333 72.52744	-	-	-	-	C	C	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για $CW(n, k,)$

Αποτελέσματα για την μέθοδο RS (συνέχεια).

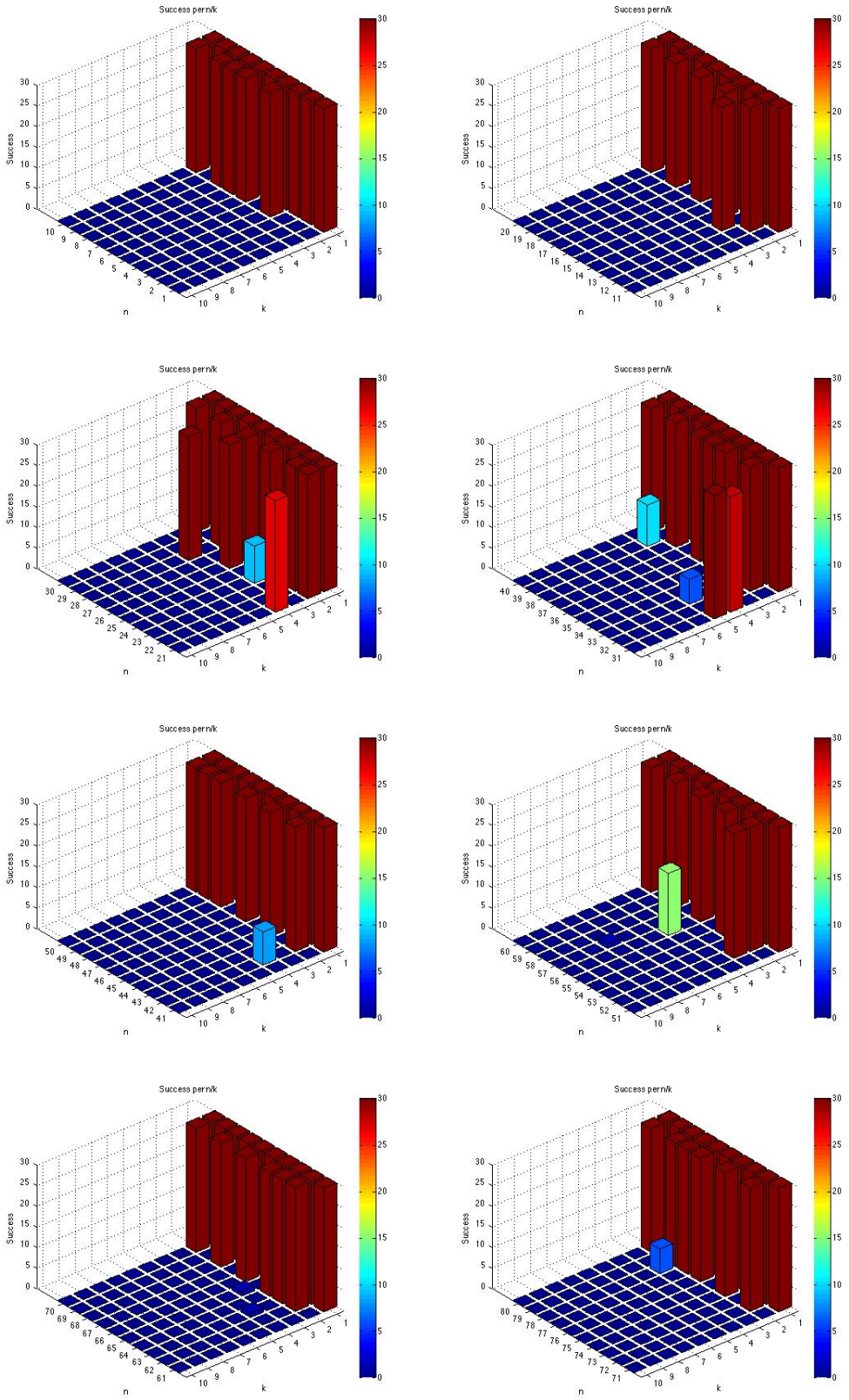
Πίνακας Α.50

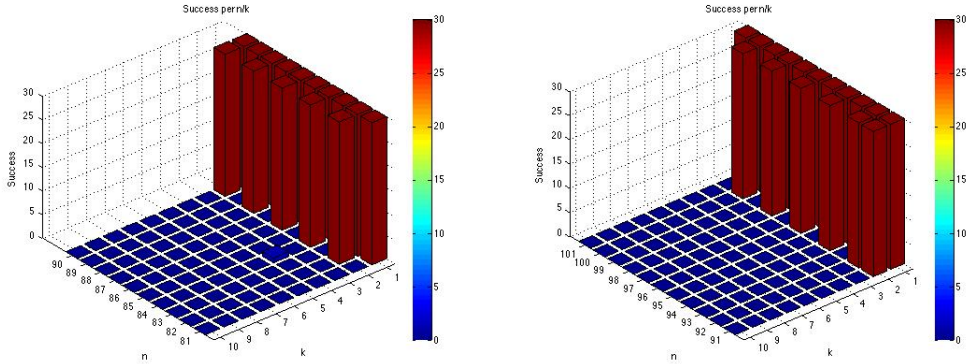
N	k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
91	% suc	100%	26.6%	0%	-	-	0%	-	-	0%	-
	Time avg/std	0.333333 1.825741	198.75 107.2297	-	-	-	-	-	-	- -	-
92	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	0.05 0	94 101.3563	-	-	-	N	-	N	-	-
93	% suc	100%	-	-	0%	0%	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
94	% suc	100%	100%	-	-	-	N	-	N	-	-
	Time avg/std	0.05 0	146.6666 108.4795	-	-	-	N	-	N	-	-
95	% suc	100%	-	-	-	-	N	C	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	N	C	-	-	-
96	% suc	100%	96.6%	0%	-	C	0%	C	-	-	-
	Time avg/std	0.333333 1.825741	101.0344 80.46049	-	-	C	-	C	-	-	-
97	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.333333 1.825741	-	-	-	-	-	-	-	-	-
98	% suc	100%	96.6%	-	0%	-	-	-	N	-	-
	Time avg/std	0.05 0	132.4137 102.7707	-	-	-	-	-	N	-	-
99	% suc	100%	-	-	-	0%	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-
100	% suc	100%	96.6%	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	149.3103 119.7297	-	-	-	-	-	-	-	-
101	% suc	100%	-	-	-	-	-	-	-	-	-
	Time avg/std	0.05 0	-	-	-	-	-	-	-	-	-

Ύπαρξη λύσης (*Y*), μη-ύπαρξη (.), η μη-ύπαρξη αποδείχτηκε με υπολογιστή (*C*), η μη-ύπαρξη αποδείχτηκε στο paper [1] της βιβλιογραφίας (*N*), και ανοιχτά προβλήματα (?) για *CW*(*n*, *k*,)

Παράρτημα Β : Γραφήματα Επιτυχιών

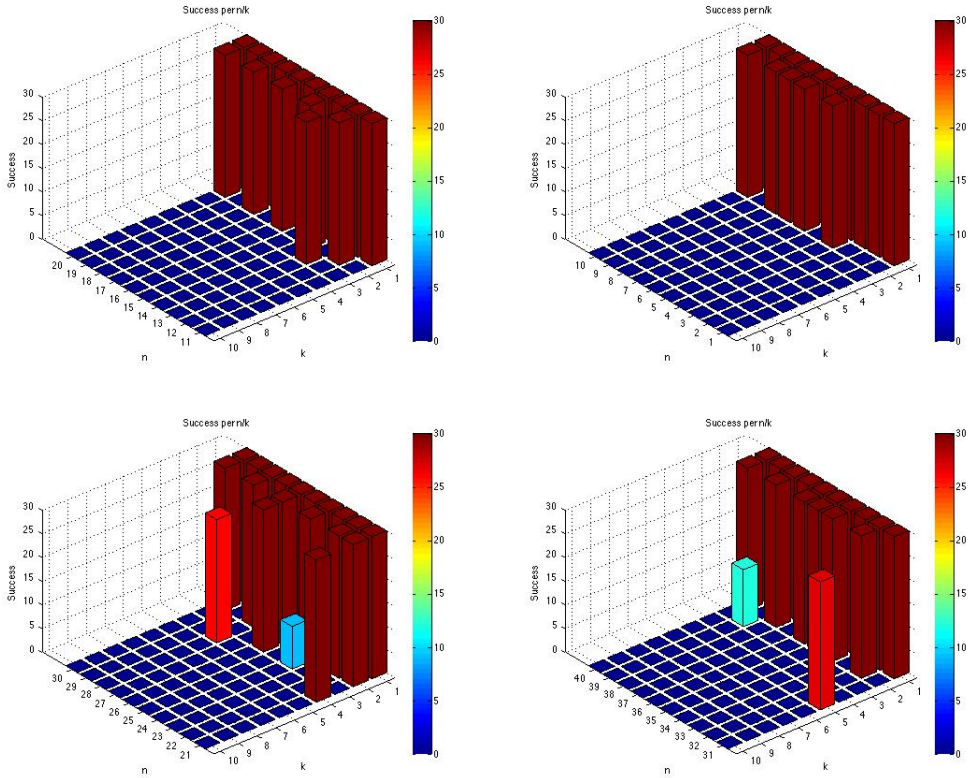
Β.1 Γραφήματα για την TS



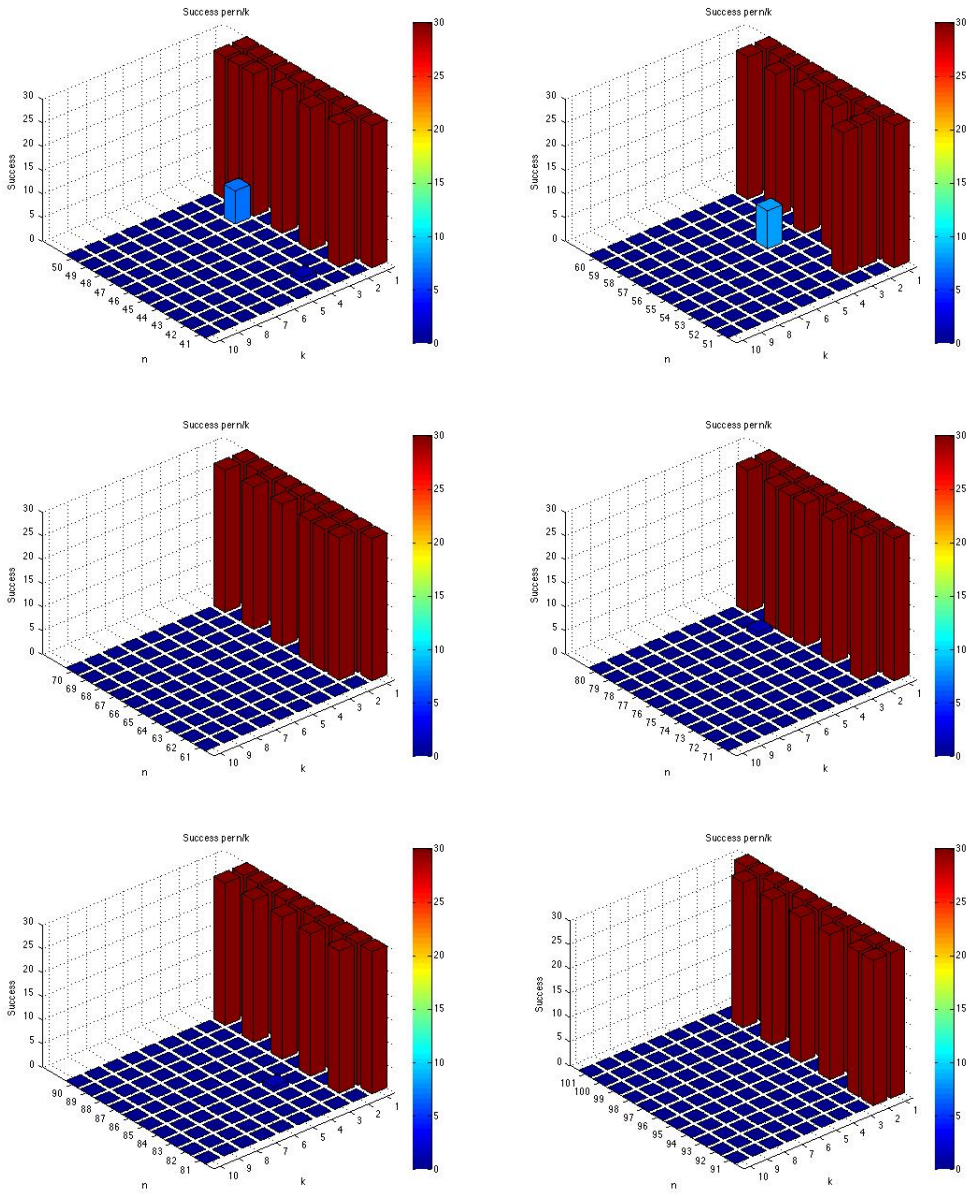


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την TS.

Β.2 Γραφήματα για την TS-PR

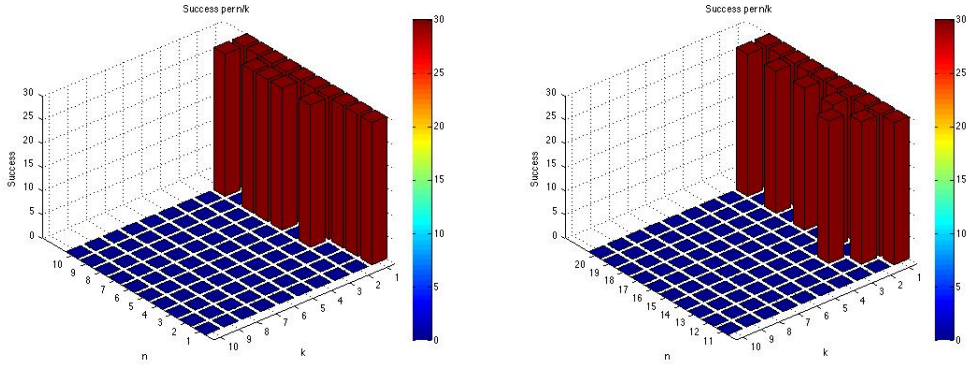


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την TS-PR.

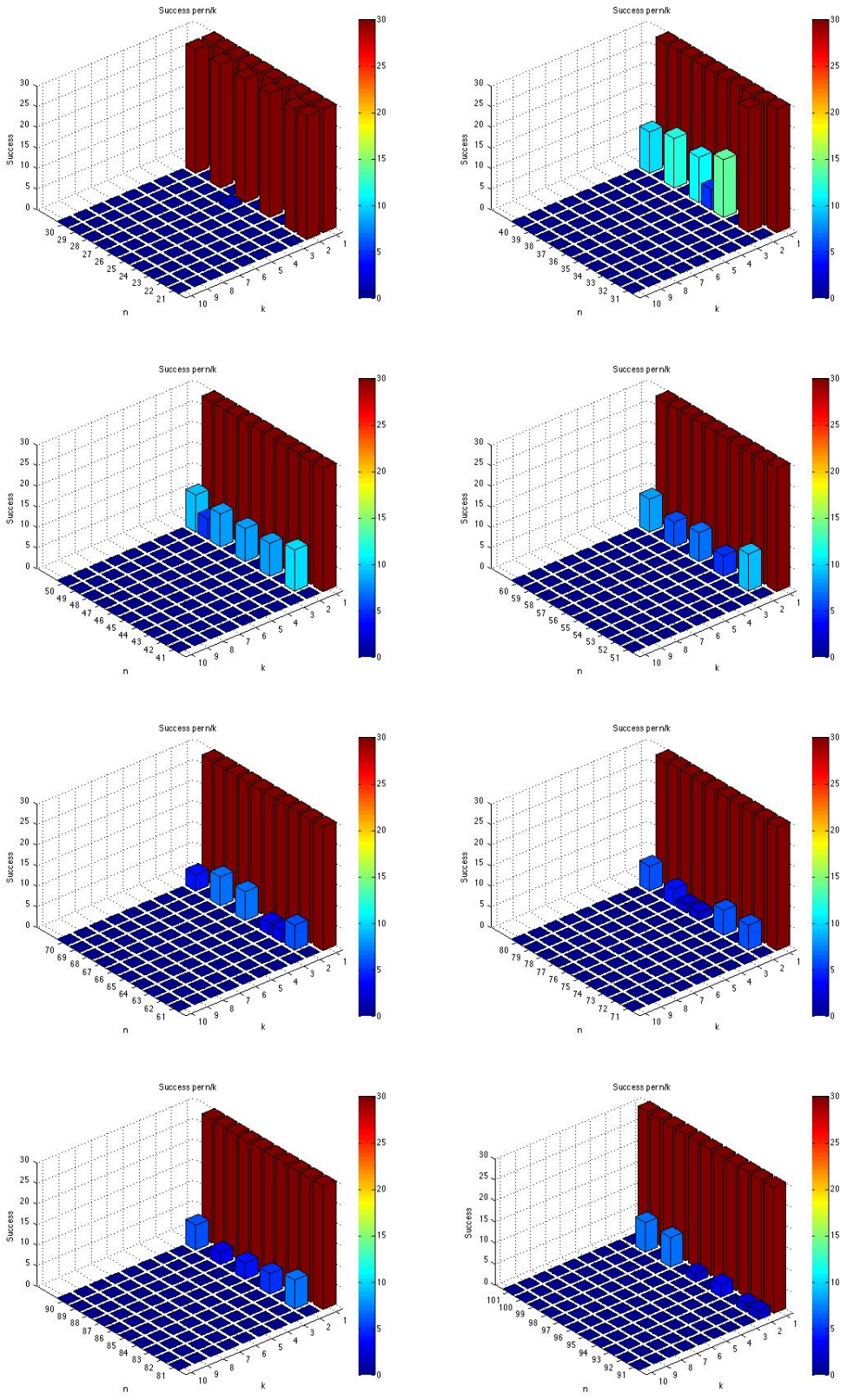


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την TS-PR.

B.3 Γραφήματα για την VNS

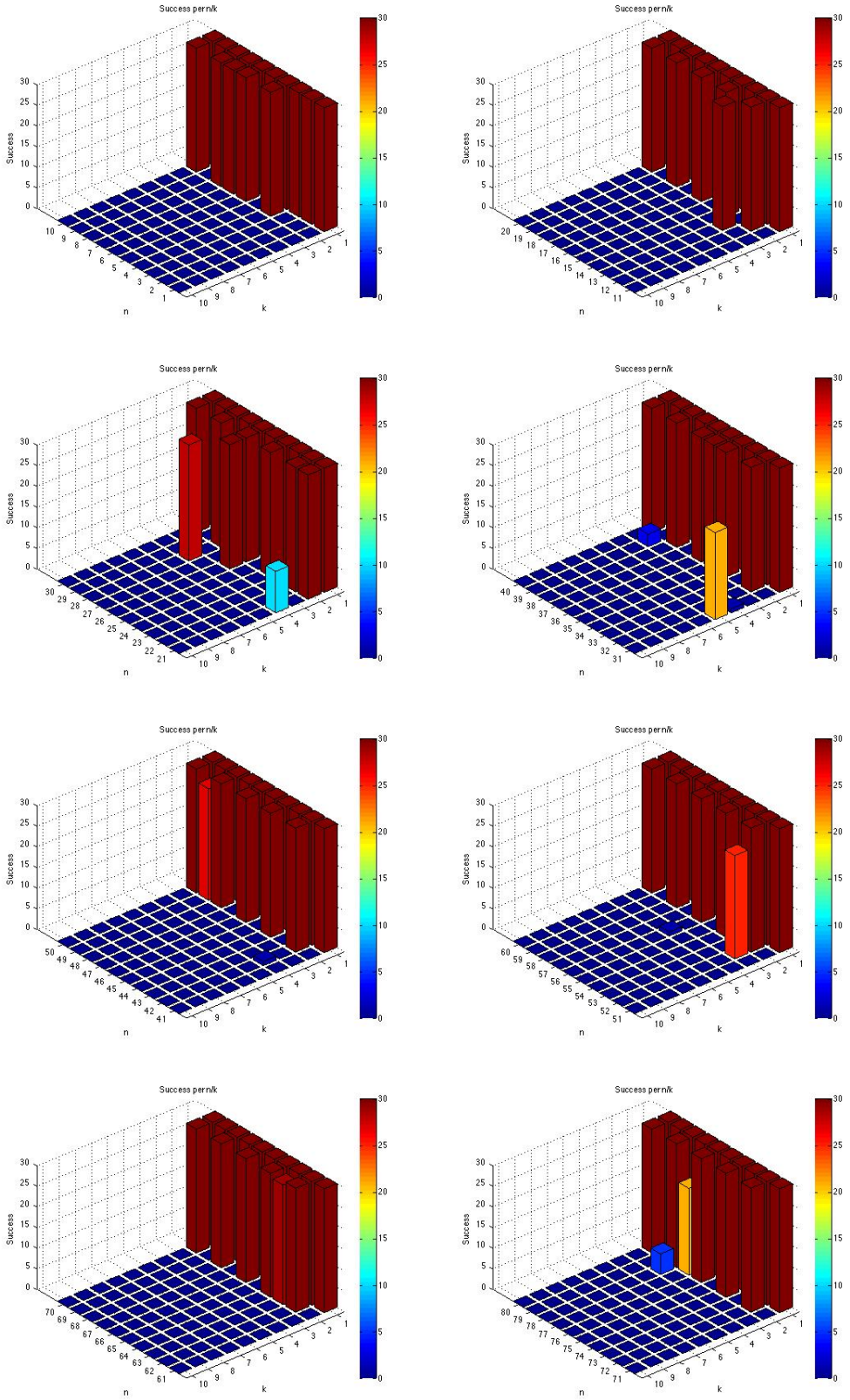


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την VNS.

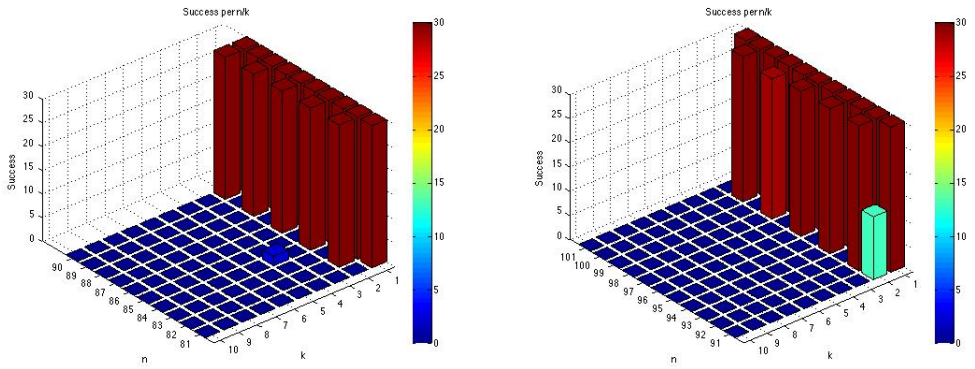


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την VNS.

B.4 Γραφήματα για την IGD

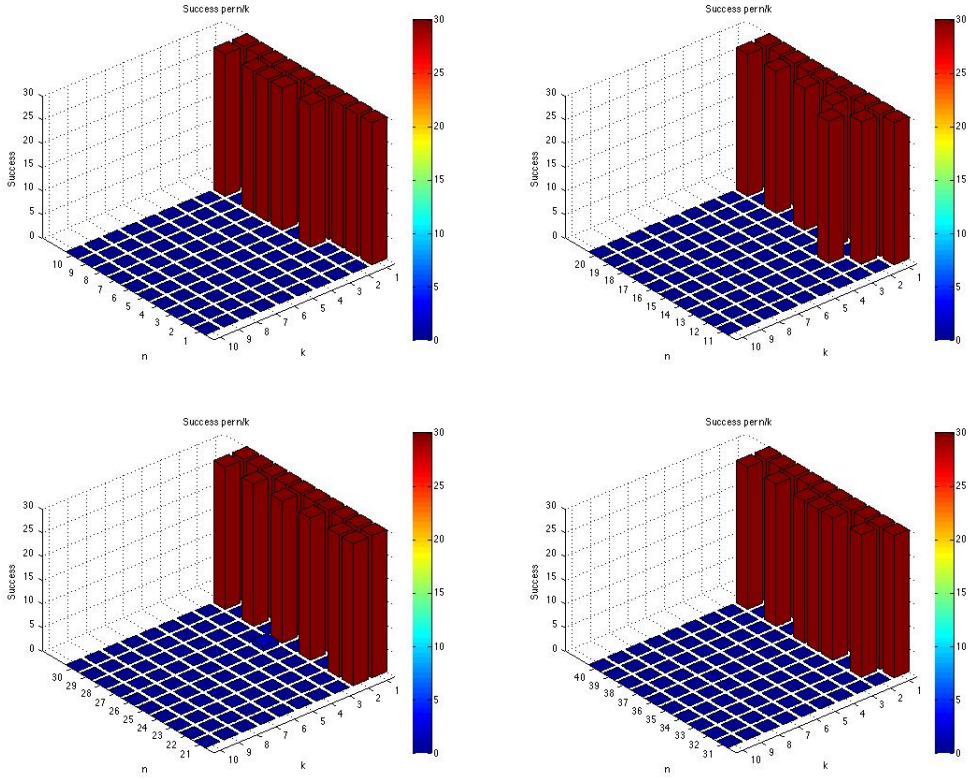


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την IGD.

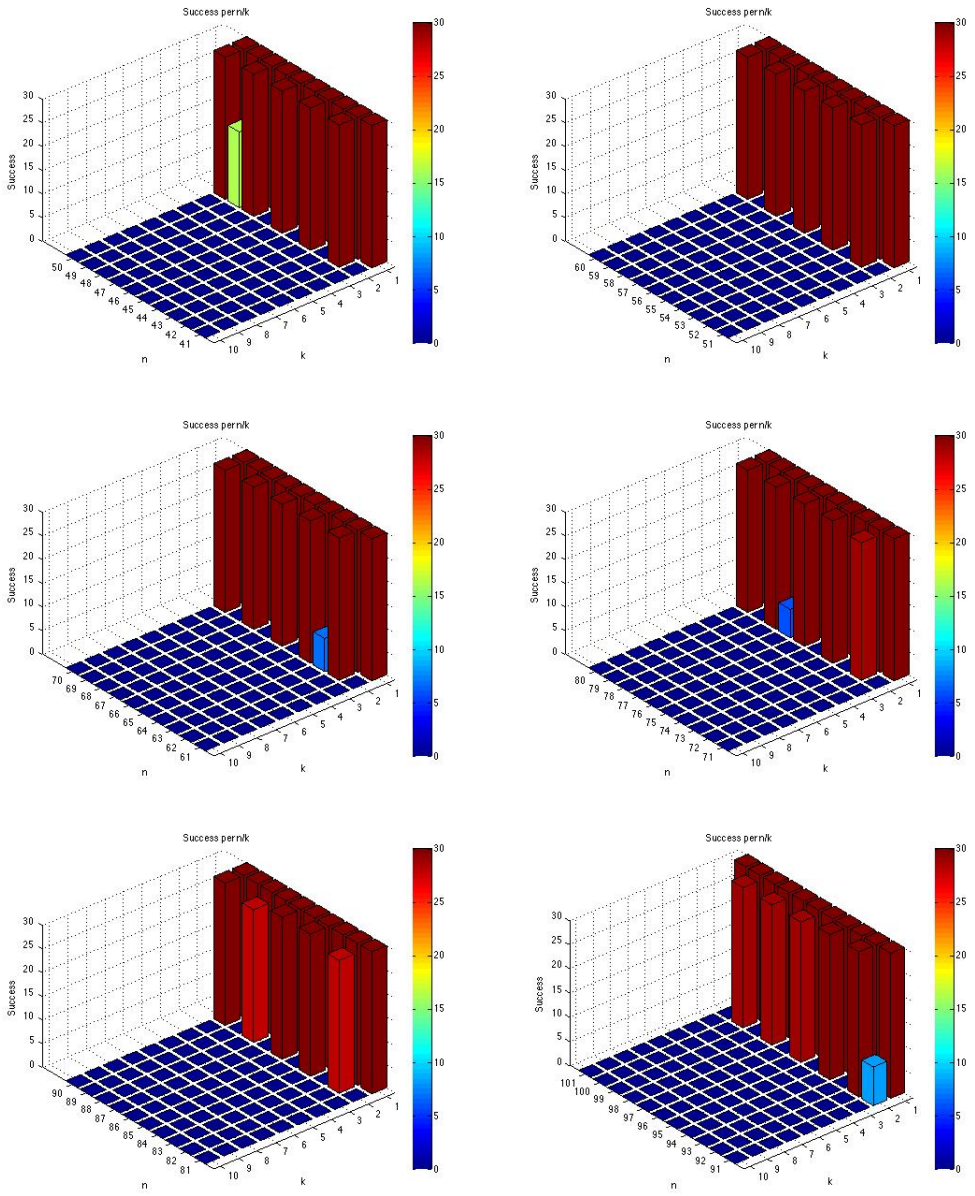


Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την IGD.

B.5 Γραφήματα για την RS



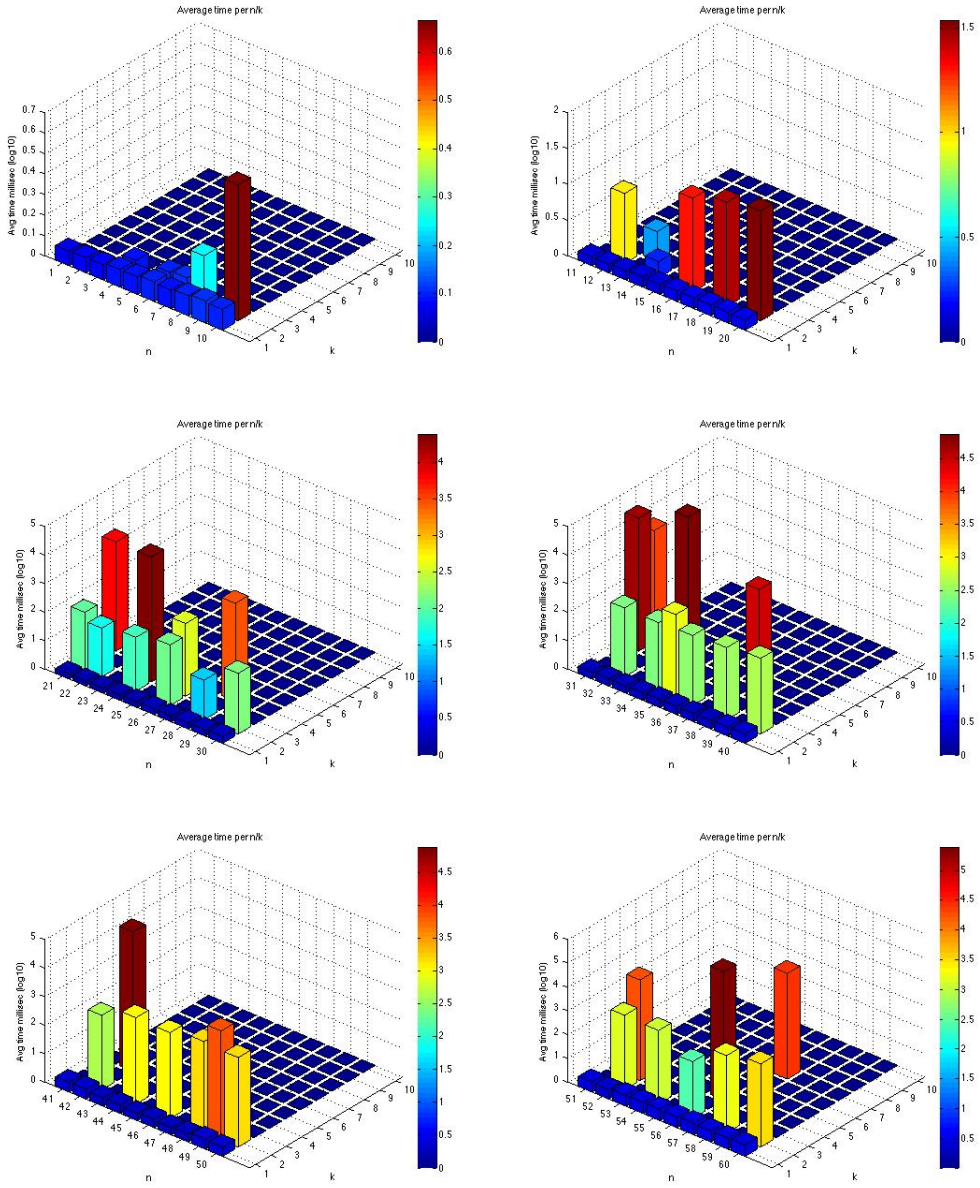
Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την RS.



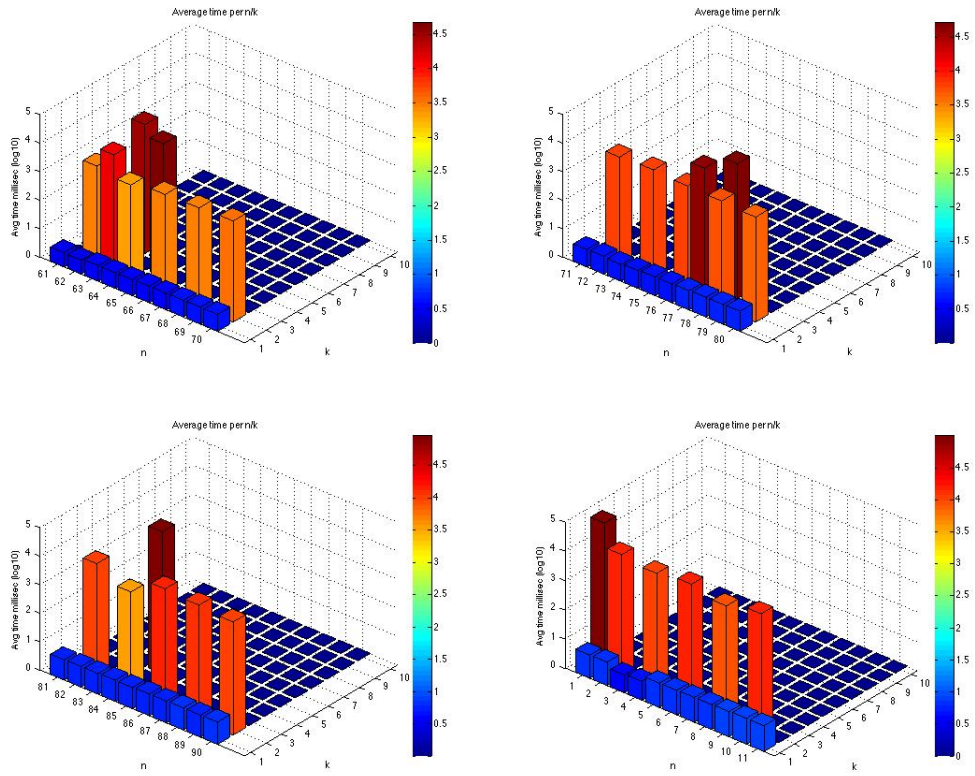
Σύνολο των επιτυχημένων πειραμάτων στα 30 πειράματα ανά πρόβλημα για την RS.

Παράρτημα Γ : Γραφήματα Απαιτούμενου Χρόνου

Γ.1 Γραφήματα για την TS

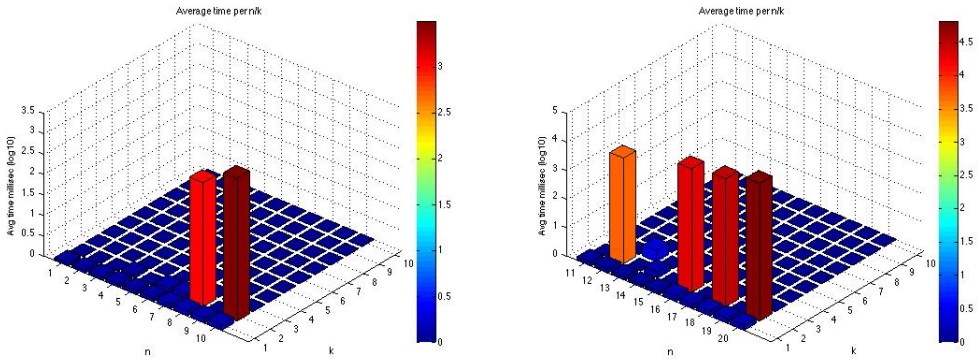


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την TS.

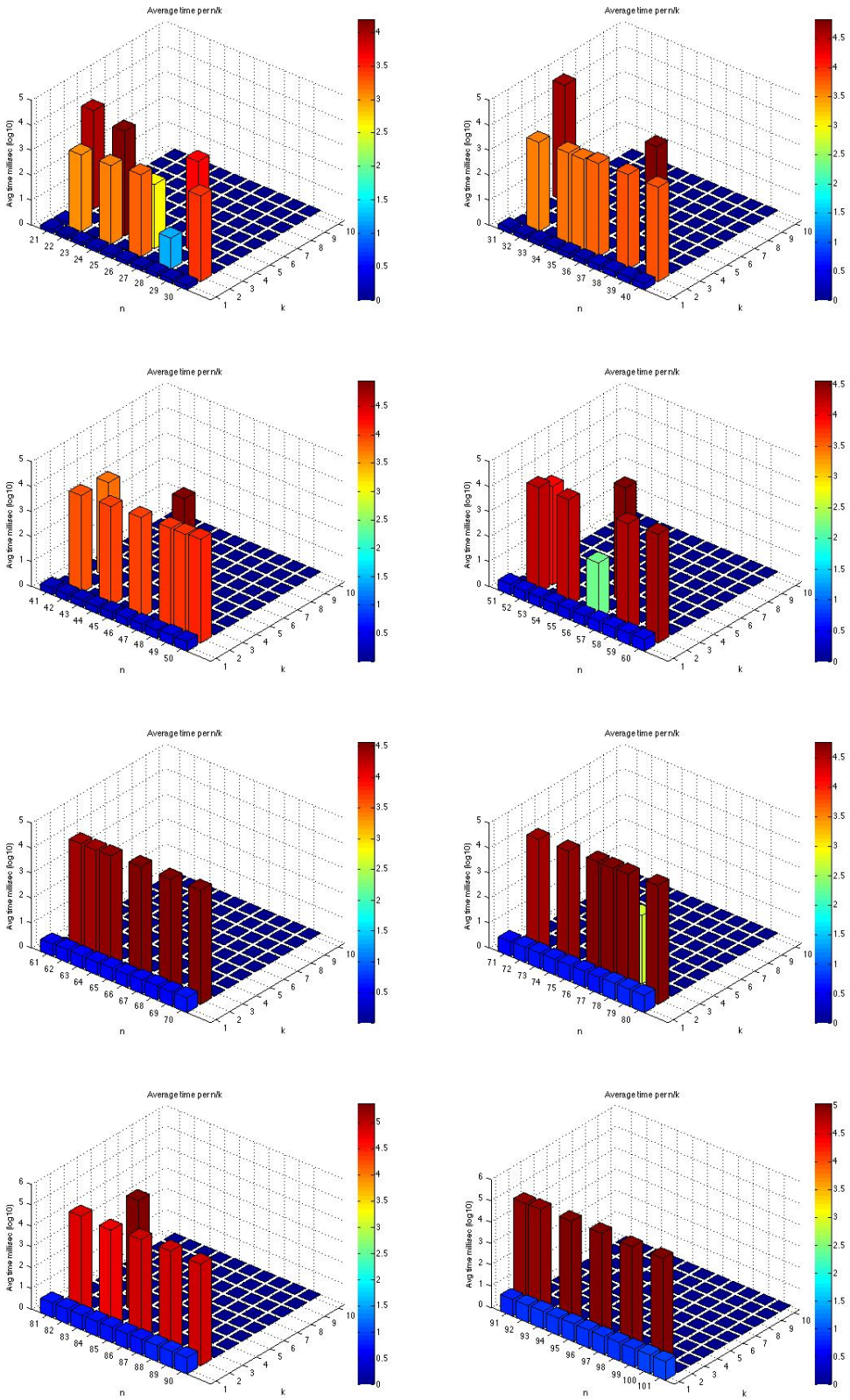


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την TS.

Γ.2 Γραφήματα για την TS-PR

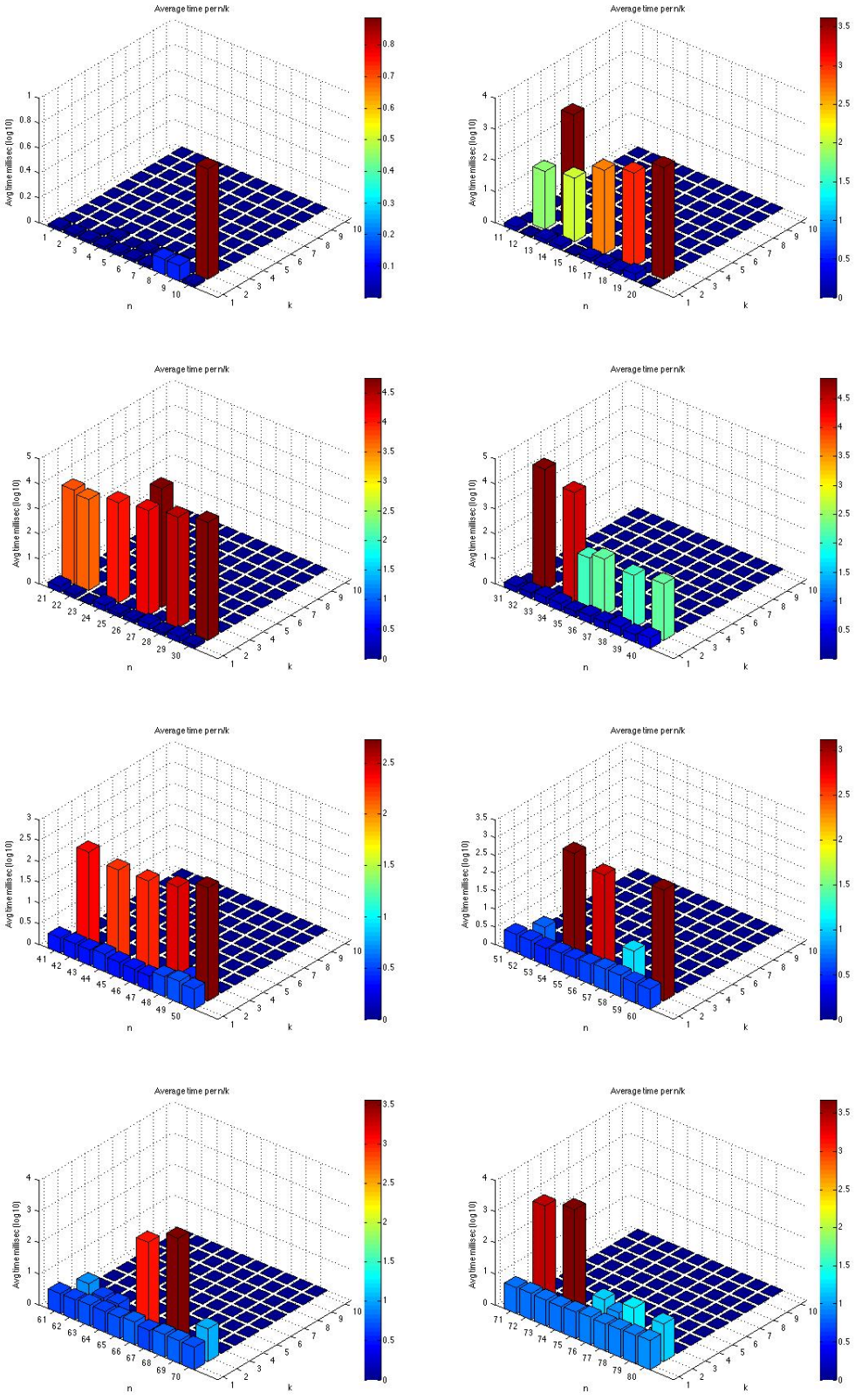


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την TS-PR.

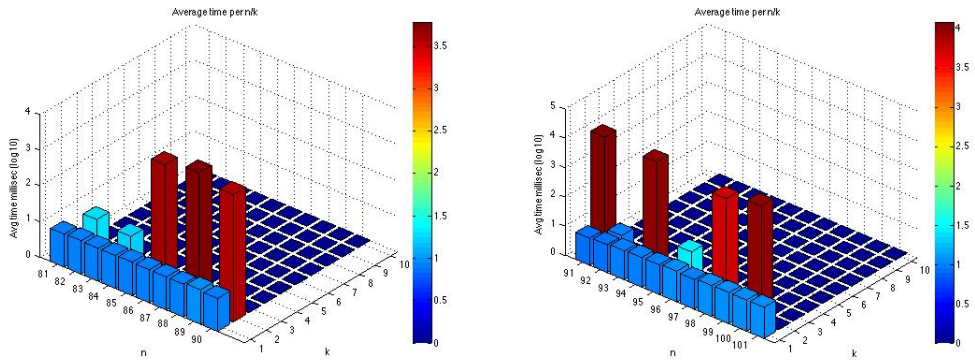


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την TS-PR.

Γ.3 Γραφήματα για την VNS

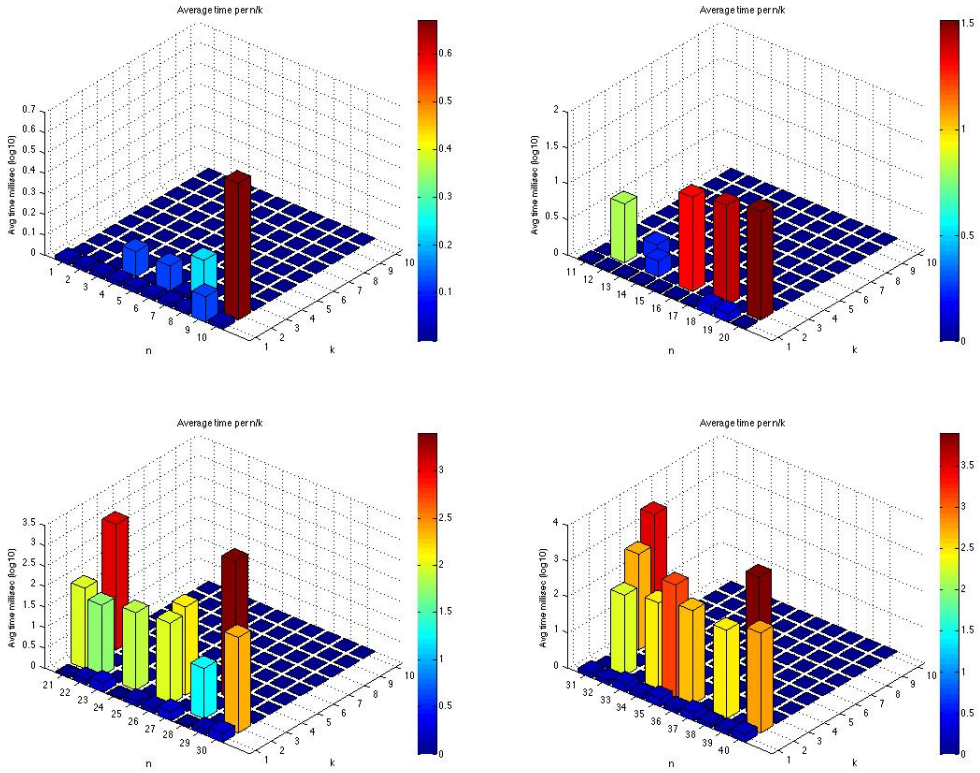


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την VNS.

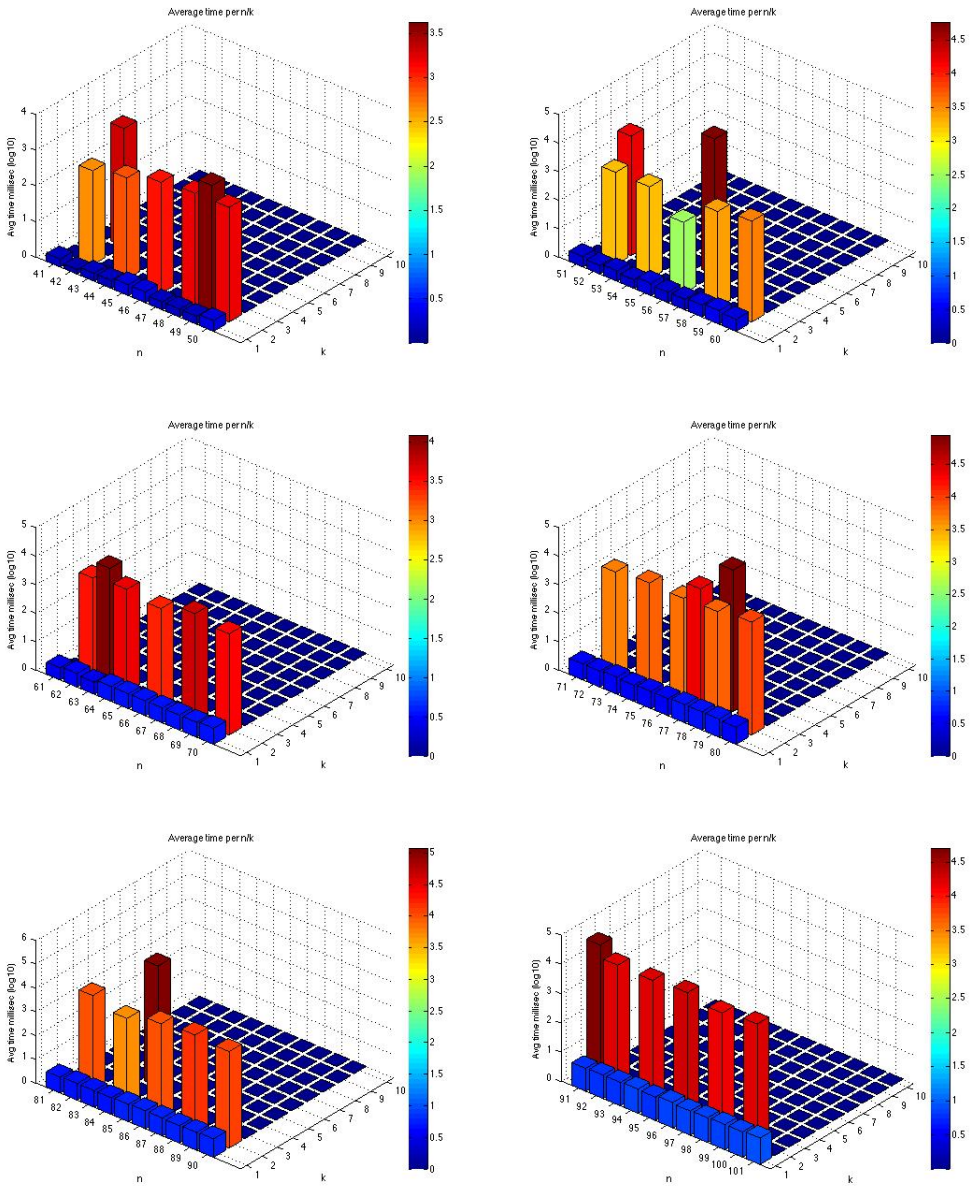


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την VNS.

Γ.4 Γραφήματα για την IGD

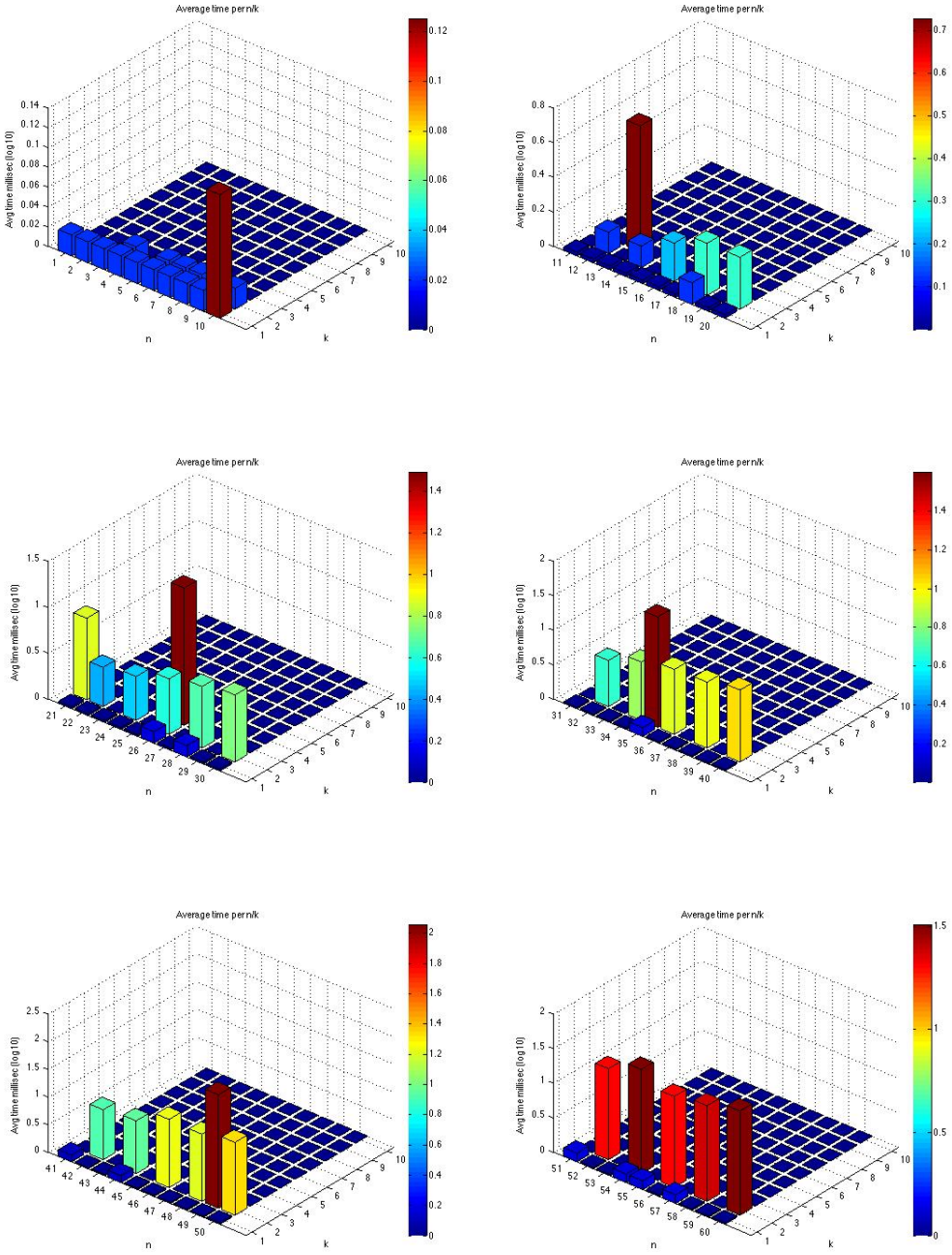


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την IGD.

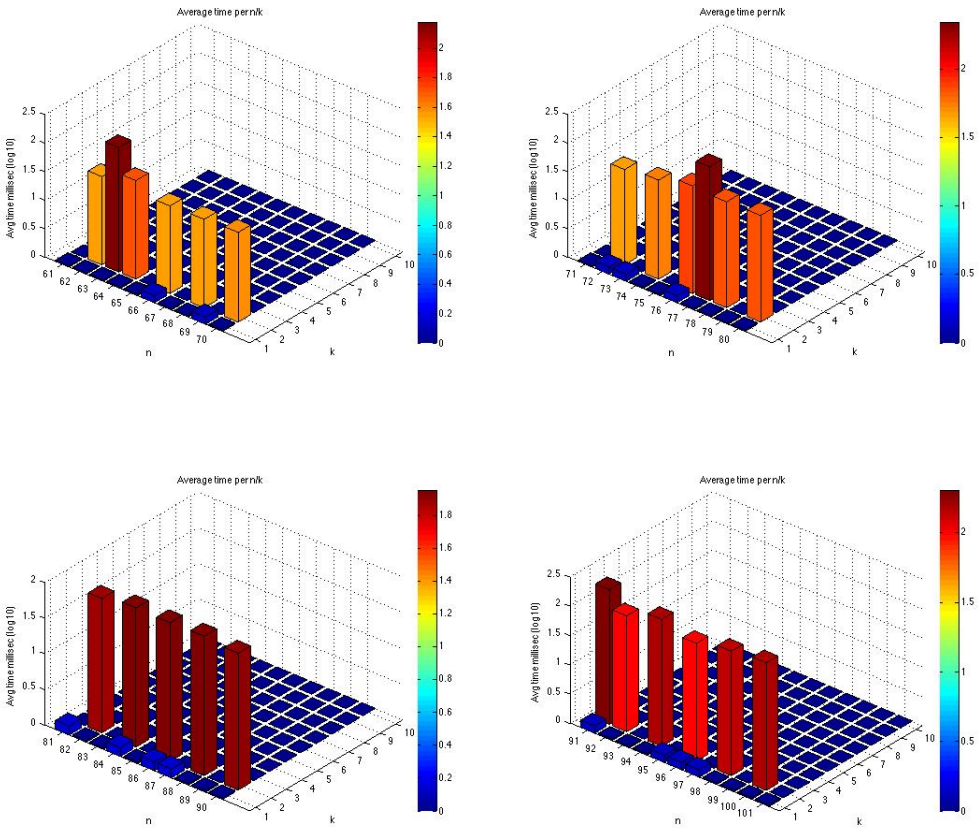


Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την IGD.

Γ.5 Γραφήματα για την RS



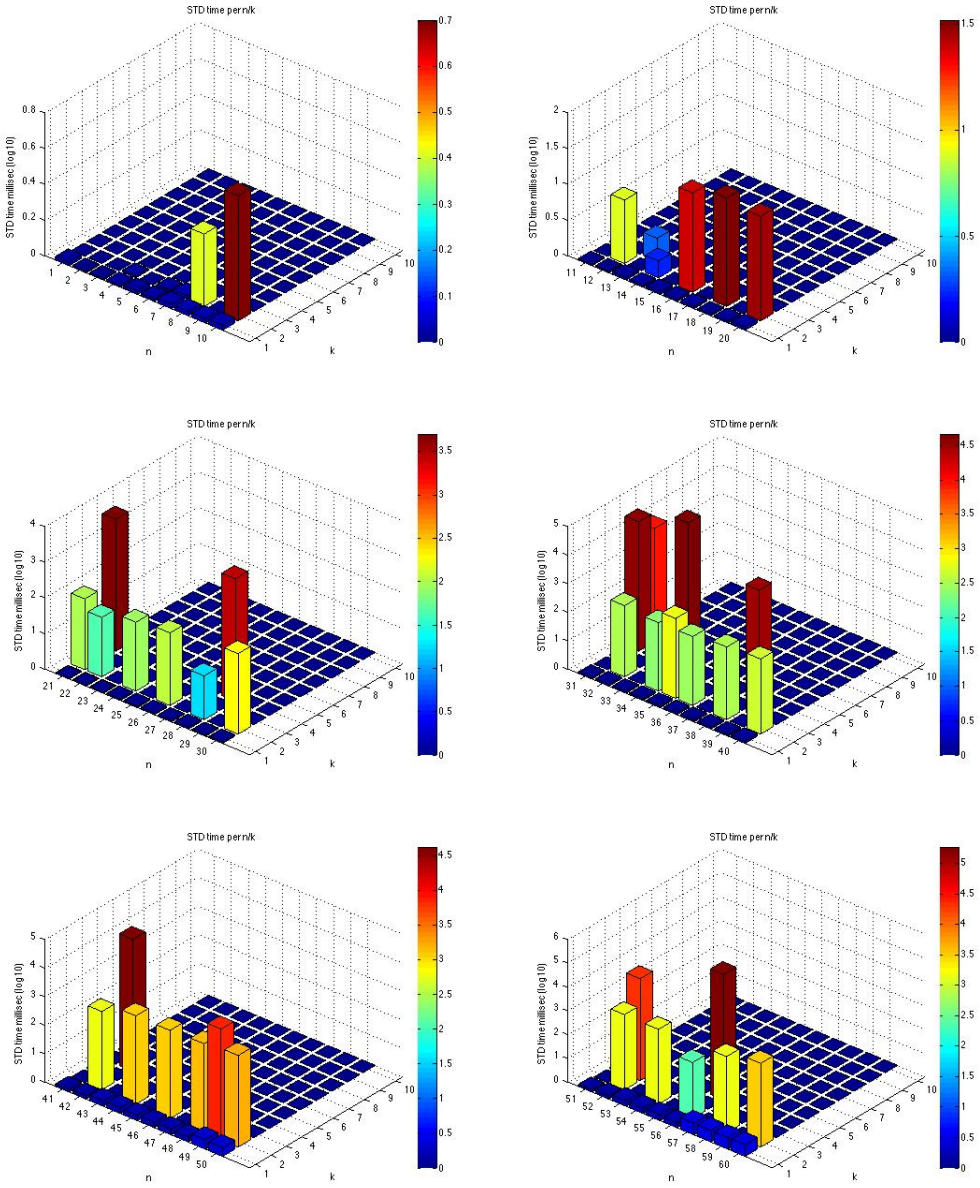
Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την RS.



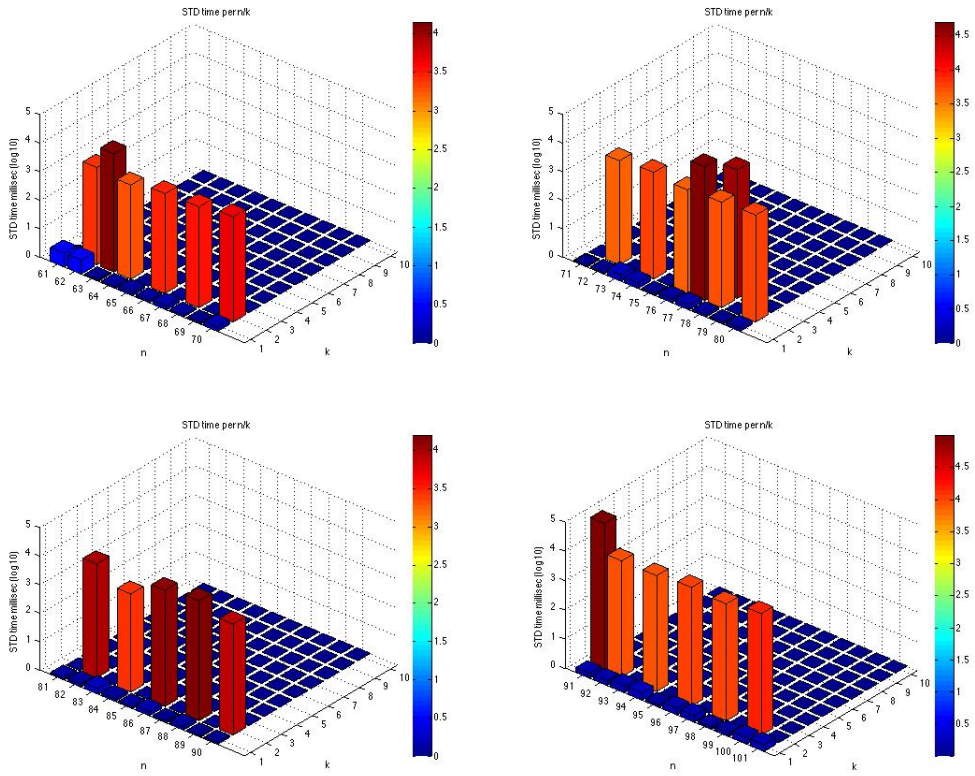
Απαιτούμενος χρόνος επίλυσης ανά πρόβλημα για την RS.

Παράρτημα Δ : Γραφήματα Τυπικής Απόκλισης

Δ.1 Γραφήματα για την TS

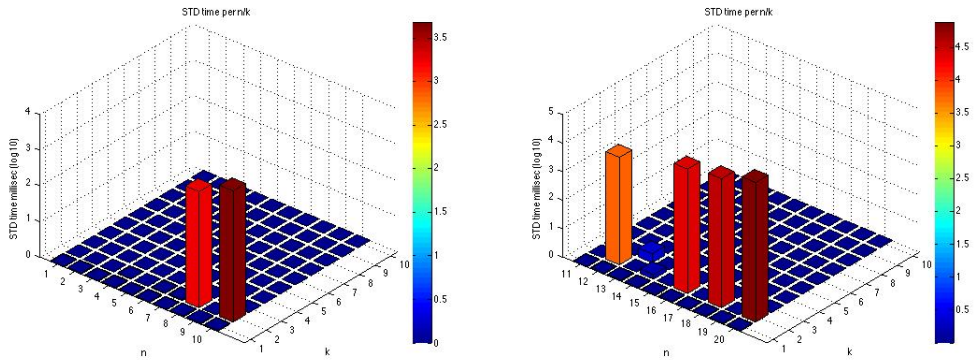


Τυπική απόκλιση ανά πρόβλημα για την TS.

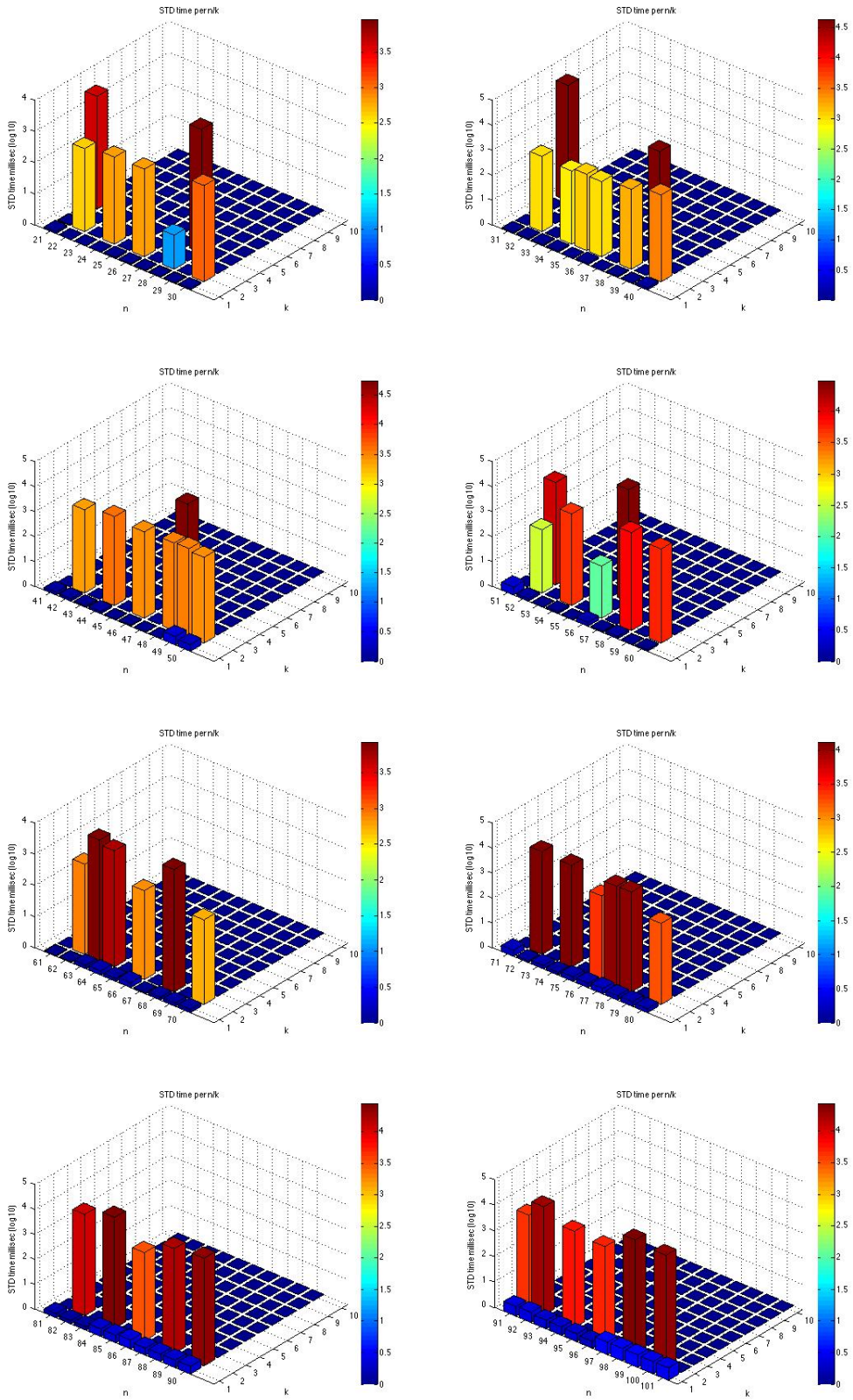


Τυπική απόκλιση ανά πρόβλημα για την TS.

Δ.2 Γραφήματα για την TS-PR

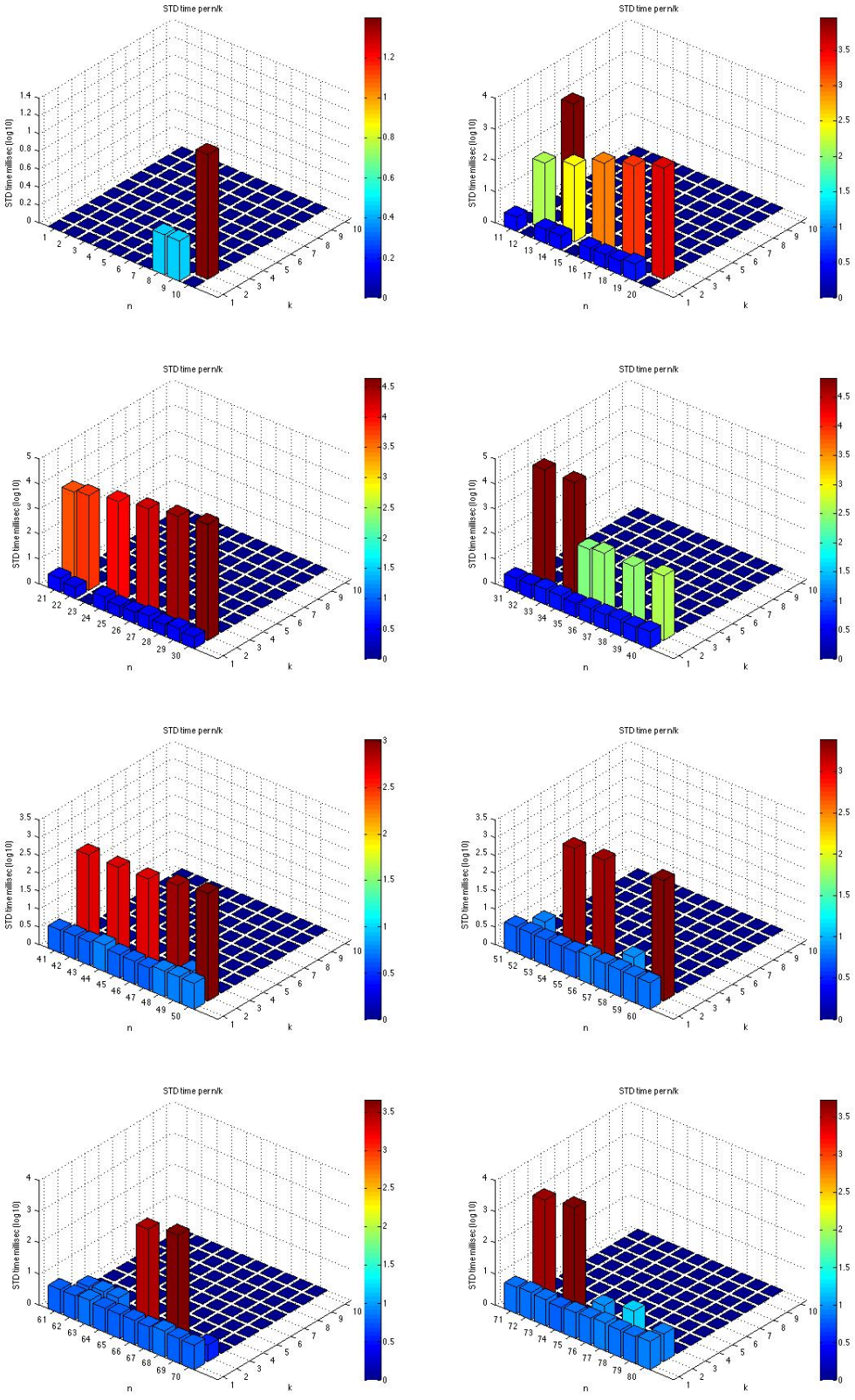


Τυπική απόκλιση ανά πρόβλημα για την TS-PR.

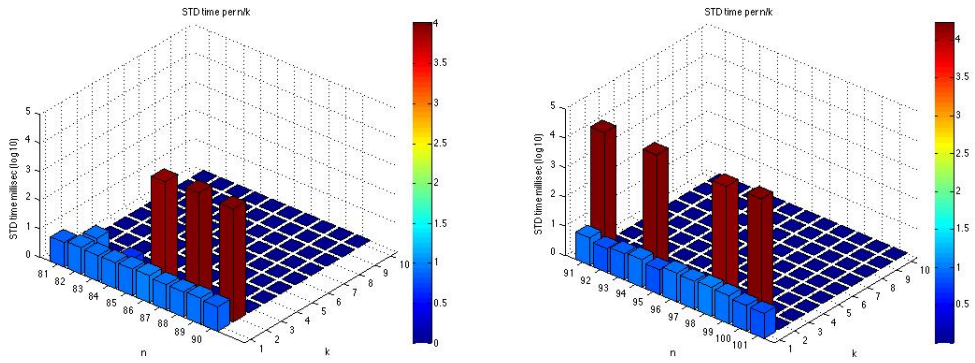


Τυπική απόκλιση ανά πρόβλημα για την TS-PR.

Δ.3 Γραφήματα για την VNS

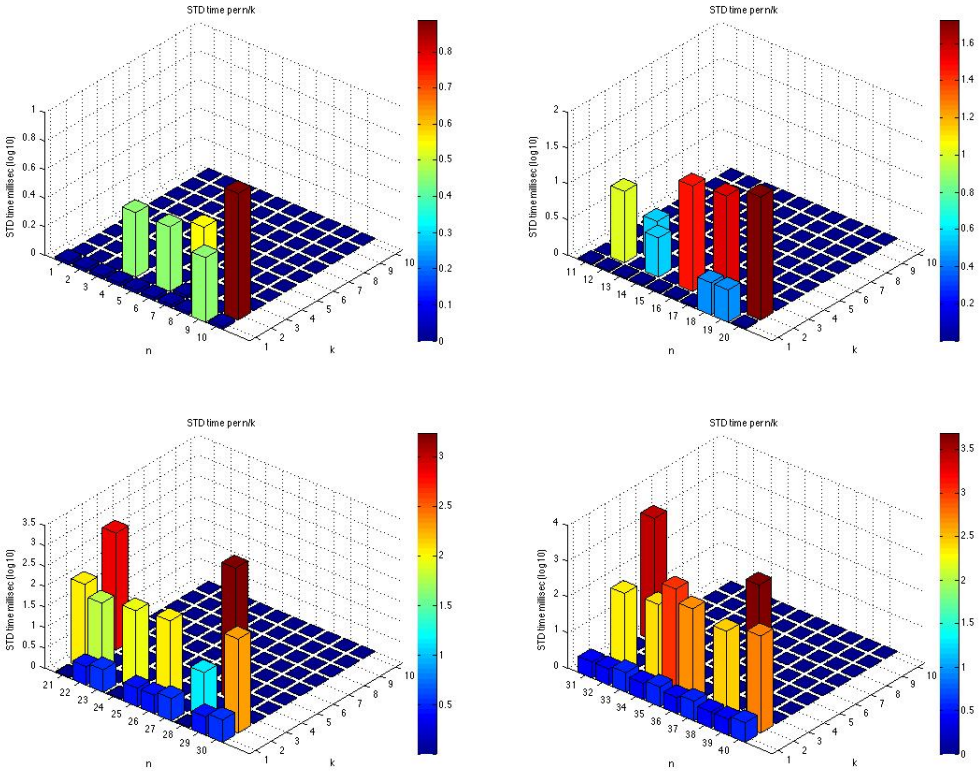


Τυπική απόκλιση ανά πρόβλημα για την VNS.

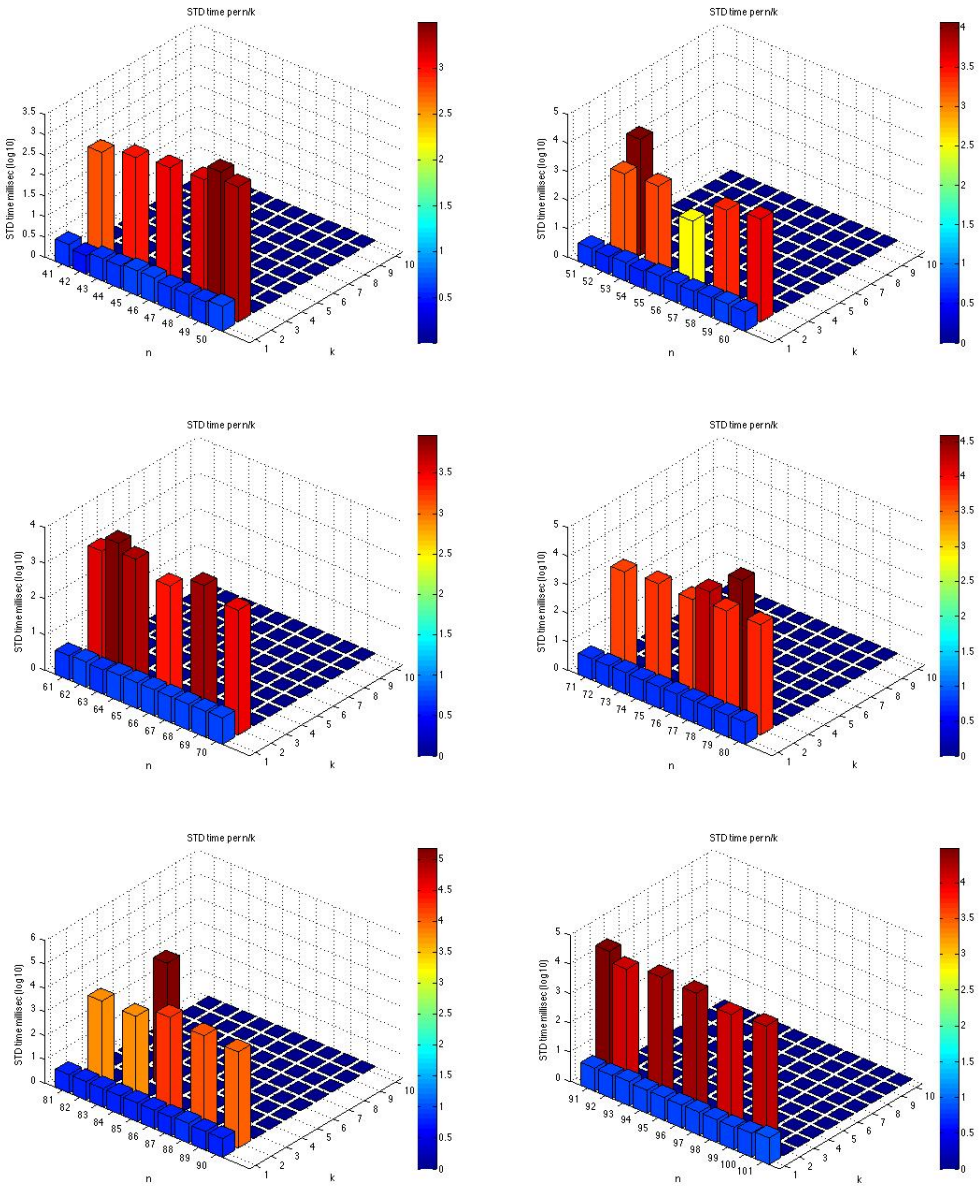


Τυπική απόκλιση ανά πρόβλημα για την VNS.

Δ.4 Γραφήματα για την IGD

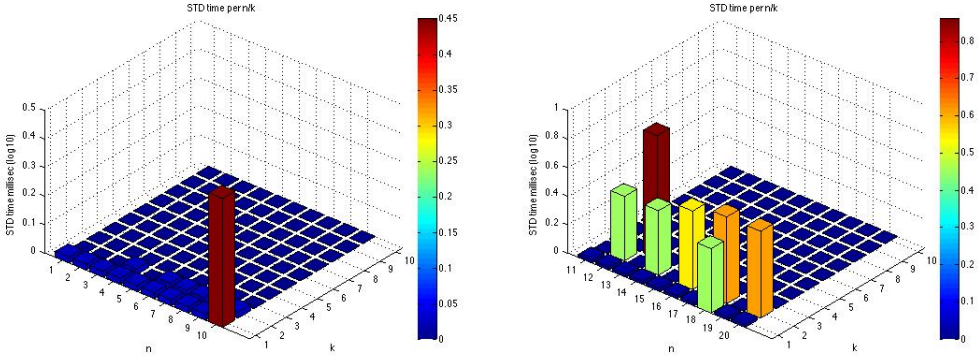


Τυπική απόκλιση ανά πρόβλημα για την IGD.

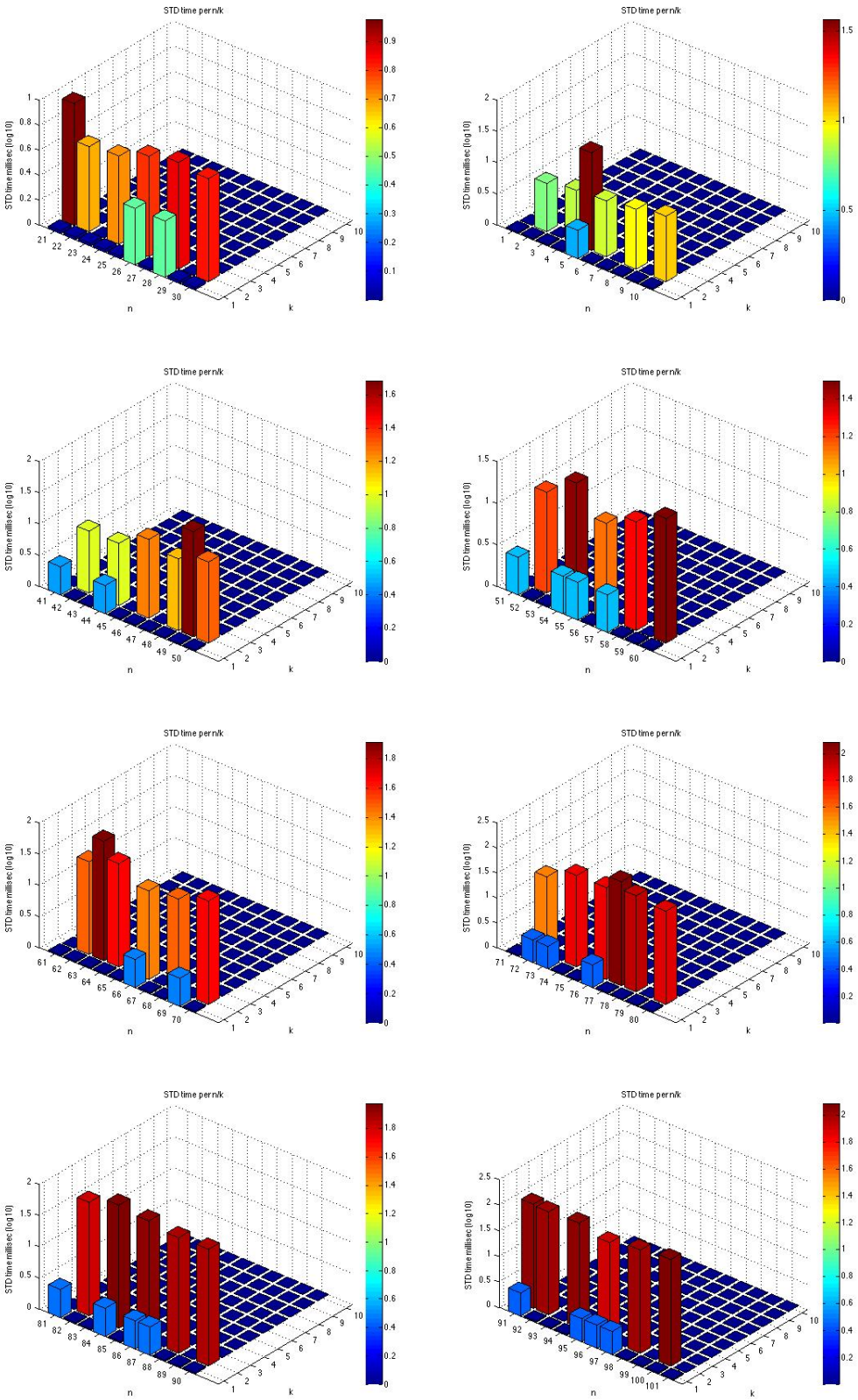


Τυπική απόκλιση ανά πρόβλημα για την IGD.

Δ.5 Γραφήματα για την RS



Τυπική απόκλιση ανά πρόβλημα για την RS.



Τυπική απόκλιση ανά πρόβλημα για την RS.

Παράρτημα Ε : Πίνακες κατάταξης αλγορίθμων

Πίνακας Ε.1

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
1	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS,VNS),TS-PR,TS	-	-	-	-	-	-	-	-	-
2	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	-	-	-	-	-	-	-	-	-
3	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	-	-	-	-	-	-	-	-	-
4	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	(RS,VNS),TS,TS-PR,IGD	-	-	-	-	-	-	-	-
5	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	-	-	-	-	-	-	-	-	-
6	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	(RS,VNS),TS,TS-PR,IGD	-	-	-	-	-	-	-	-
7	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	(IGD,RS,VNS),TS,TS-PR	-	-	-	-	-	-	-	-
8	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS),TS,TS-PR,VNS	(RS,VNS),IGD,TS,TS-PR	-	-	-	-	-	-	-	-
9	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,TS-PR,(IGD,VNS)	-	-	-	-	-	-	-	-	-
10	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,VNS),TS,RS,TS-PR	RS,TS,IGD,VNS,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.2

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
11	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS),TS,VNS,TS-PR	-	-	-	-	-	-	-	-	-
12	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	RS,IGD,TS,VNS,TS-PR	-	-	-	-	-	-	-	-
13	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)		(TS,TS-PR,IGD,RS,VNS)							
	Χρόνος	(IGD,RS),TS-PR,VNS,TS	-	IGD,TS-PR,TS,RS,VNS	-	-	-	-	-	-	-
14	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS),TS-PR,VNS,TS	RS,TS-PR,IGD,TS,VNS	-	-	-	-	-	-	-	-
15	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS,VNS),TS-PR,TS	-	-	-	-	-	-	-	-	-
16	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS),VNS,TS,TS-PR	RS,TS,IGD,VNS,TS-PR	-	-	-	-	-	-	-	-
17	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(IGD,RS),VNS,TS,TS-PR	-	-	-	-	-	-	-	-	-
18	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	RS,IGD,TS,VNS,TS-PR	-	-	-	-	-	-	-	-
19	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS,TS-PR,VNS	-	-	-	-	-	-	-	-	-
20	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	(IGD,RS,VNS),TS,TS-PR	RS,IGD,TS,VNS,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.3

CW(*n*,*k*)	Κριτήριο	1	2	3	4	5	6	7	8	9	10
21	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)		TS-PR,TS,IGD,--						
	Χρόνος	(IGD,RS),TS-PR,TS,VNS	TS-PR,RS,IGD,TS,VNS	-	IGD,TS,TS-PR,--	-	-	-	-	-	-
22	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	RS,VNS,IGD,TS-PR,TS	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-
23	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	(RS,VNS),TS-PR,TS,IGD	-	-	-	-	-	-	-	-	-
24	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR),---							
	Χρόνος	(IGD,RS),TS-PR,TS,VNS	RS,TS,IGD,TS-PR,VNS	TS-PR,TS,---	-	-	-	-	-	-	-
25	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,VNS,IGD,TS-PR,TS	-	-	-	-	-	-	-	-	-
26	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),(RS,VNS)							
	Χρόνος	VNS,(IGD,RS),TS-PR,TS	RS,IGD,TS,TS-PR,VNS	RS,IGD,TS,TS-PR,VNS	-	-	-	-	-	-	-
27	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS-PR,TS,VNS,IGD	-	-	-	-	-	-	-	-	-
28	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)		TS,IGD,TS-PR,--						
	Χρόνος	IGD,RS,VNS,TS-PR,TS	RS,IGD,TS-PR,TS,VNS	-	IGD,TS,TS-PR,--	-	-	-	-	-	-
29	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
30	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	RS,VNS,TS-PR,IGD,TS	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.4

CW(*n*,*k*)	Κριτήριο	1	2	3	4	5	6	7	8	9	10
31	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)			TS,IGD,---	TS,TS-PR,IGD,--					
	Χρόνος	RS,IGD,TS-PR,VNS,TS	-	-	IGD,TS,---	IGD,TS,TS-PR,--	-	-	-	-	-
32	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS,VNS)								
	Χρόνος	RS,IGD,TS-PR,VNS,TS	RS,IGD,TS,TS-PR,VNS	-	-	-	-	-	-	-	-
33	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)				TS,----					
	Χρόνος	RS,TS-PR,IGD,TS,VNS	-	-	-	TS,----	-	-	-	-	-
34	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,TS,IGD,VNS	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-
35	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,IGD,VNS,TS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
36	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
37	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
38	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
39	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)		TS-PR,TS,IGD,--							
	Χρόνος	RS,IGD,TS-PR,VNS,TS	-	IGD,TS,TS-PR,--	-	-	-	-	-	-	-
40	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.5

CW(*n*,*k*)	Κριτήριο	1	2	3	4	5	6	7	8	9	10
41	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
42	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS		TS,(TS-PR,IGD),--						
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	IGD,TS-PR,TS,--	-	-	-	-	-	-
43	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
44	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
45	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-	-
46	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS,IGD,TS-PR,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
47	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS,TS-PR,VNS	-	-	-	-	-	-	-	-	-
48	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS	TS-PR,----							
	Χρόνος	RS,IGD,TS,TS-PR,VNS	RS,VNS,TS,IGD,TS-PR	TS-PR,----	-	-	-	-	-	-	-
49	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR),IGD,RS,VNS								
	Χρόνος	RS,IGD,TS,TS-PR,VNS	VNS,RS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
50	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS,TS-PR,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.6

CW(*n*,*k*)	Κριτήριο	1	2	3	4	5	6	7	8	9	10
51	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
52	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS	(TS,TS-PR),IGD,--							
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	TS-PR,IGD,TS,--	-	-	-	-	-	-	-
53	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
54	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
55	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS-PR,IGD,TS,VNS	-	-	-	-	-	-	-	-	-
56	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS		TS,TS-PR,IGD,--						
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,TS-PR,TS,IGD,VNS	-	TS-PR,IGD,TS,--	-	-	-	-	-	-
57	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)						TS,----			
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	TS,----	-	-	-
58	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
59	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,TS-PR,IGD,VNS	-	-	-	-	-	-	-	-	-
60	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.7

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
61	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
62	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS,TS-PR,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
63	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR),IGD,RS,VNS		TS,----						
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,IGD,TS,TS-PR	-	TS,----	-	-	-	-	-	-
64	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,IGD,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
65	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)		TS,----							
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	TS,----	-	-	-	-	-	-	-
66	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,IGD,TS,VNS	RS,VNS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
67	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
68	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,TS,VNS,IGD,TS-PR	-	-	-	-	-	-	-	-
69	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
70	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.8

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
71	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
72	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
73	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-	-
74	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS,TS-PR,VNS	RS,VNS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
75	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
76	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
77	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR),IGD,RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,IGD,TS,TS-PR	-	-	-	-	-	-	-	-
78	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS	TS,IGD,TS-PR,--							
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	TS-PR,TS,IGD,--	-	-	-	-	-	-	-
79	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
80	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.9

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
81	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
82	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
83	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
84	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS		IGD,(TS,TS-PR),--						
	Χρόνος	RS,IGD,TS-PR,TS,VNS	VNS,RS,TS,IGD,TS-PR	-	TS,IGD,TS-PR,--	-	-	-	-	-	-
85	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS-PR,IGD,TS,VNS	-	-	-	-	-	-	-	-	-
86	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,IGD,TS,VNS	RS,TS-PR,VNS,TS,IGD	-	-	-	-	-	-	-	-
87	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS-PR,IGD,TS,VNS	-	-	-	-	-	-	-	-	-
88	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
89	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
90	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Πίνακας Ε.10

$CW(n,k)$	Κριτήριο	1	2	3	4	5	6	7	8	9	10
91	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR),IGD,RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,IGD,TS-PR,TS	-	-	-	-	-	-	-	-
92	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS-PR,TS,IGD,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
93	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,IGD,TS-PR,VNS	-	-	-	-	-	-	-	-	-
94	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD,RS),VNS								
	Χρόνος	RS,TS,TS-PR,IGD,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
95	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS,TS-PR,VNS	-	-	-	-	-	-	-	-	-
96	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,TS-PR,TS,IGD,VNS	VNS,RS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
97	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,IGD,TS-PR,TS,VNS	-	-	-	-	-	-	-	-	-
98	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,IGD,TS,TS-PR,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
99	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,TS,TS-PR,IGD,VNS	-	-	-	-	-	-	-	-	-
100	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)	(TS,TS-PR,IGD),RS,VNS								
	Χρόνος	RS,IGD,TS-PR,TS,VNS	RS,VNS,TS,IGD,TS-PR	-	-	-	-	-	-	-	-
101	Επιτυχίες	(TS,TS-PR,IGD,RS,VNS)									
	Χρόνος	RS,(TS,TS-PR),IGD,VNS	-	-	-	-	-	-	-	-	-

Η αναλυτική περιγραφή των πινάκων βρίσκεται στα υποκεφάλαια 8.2 και 8.3.

Βιβλιογραφία

[1] Krishnasamy Thiru Arasu, Alex J. Gutman. “Circulant weighing matrices” , in *Cryptography and Communications*. Vol.2. Issue 2. pp. 155-171. Springer Science & Business Media, LLC, 2010.

[2] Alex James Gutman, “Preliminaries” in, *Circulant Weighing Matrices*. pp. 3-6. Wright State University, 2009.

[3] Ilias S. Kotsireas “Algorithms and Metaheuristics for Combinatorial Matrices”, in *Handbook of Combinatorial Optimization*, pp. 283-309. Springer New York. 2013.

[4] Michel Gendreau · Jean-Yves Potvin, “Handbook of Metaheuristics-Second Edition”, DOI 10.1007/978-1-4419-1665-5 Springer New York Dordrecht Heidelberg London, 2010

[5] Singiresu S. Rao, “Engineering Optimization: Theory and Practice”**,** John Wiley & Sons, Jul 20, 2009

[6] Thomas Weise, “Global Optimization Algorithms – Theory and Application 2nd Ed.”, http://blog.it-weise.de, 2009-06-26

[7] Panos M. Pardalos, H. Edwin Romeijn, “Handbook of global optimization Volume 2”, Kluwer Academic Publishers Boston/Dordrecht/London. 2002.

[8] Ilias S. Kotsireas, Christos Koukouvinos, Jennifer Seberry, “Hadamard ideals and Hadamard matrices with two circulant cores” , European Journal of Combinatorics 27 (2006) 658–668 , 10 May 2005

[9] M. Chiarandini, I.S. Kotsireas, C. Koukouvinos, L. Paquete, “Heuristic algorithms for Hadamard matrices with two circulant cores”, Theoretical Computer Science 407 (2008) 274–277, 2008

Σύντομο βιογραφικό

Ο Κωνσταντίνος Καλτσάς γεννήθηκε στη Θεσσαλονίκη στις 20 Απριλίου 1984. Αποφοίτησε από το τμήμα Πληροφορικής του Πανεπιστημίου Πειραιά με κατεύθυνση «Τεχνολογία λογισμικού και ευφυή συστήματα» και θέμα πτυχιακής εργασίας «Σχεδιασμός Διαδραστικής Διαδικτυακής Εφαρμογής με κινούμενη εικόνα και ήχο». Συνέχισε τις μεταπτυχιακές του σπουδές στο Τμήμα Μηχανικών Η/Υ & Πληροφορικής του Πανεπιστημίου Ιωαννίνων με εξειδίκευση «Επιστημονικοί Υπολογισμοί» και θέμα μεταπτυχιακής εργασίας «Συγκριτική Μελέτη Μεταευρετικών Αλγορίθμων Βελτιστοποίησης για Υπολογισμό Κυκλικών Πινάκων Στάθμισης». Στον επαγγελματικό τομέα έχει 12 χρόνια εμπειρίας στον χώρο του Web Development, Mobile development και server back-end development.