ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Υλοποίηση και Πειραματική Σύγκριση Δύο Γενικών Αλγορίθμων Συνέπειας Τόξου (Arc Consistency)

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΔΑΜΙΑΝΟΥ Α. ΜΕΤΙΚΑΡΙΔΗ

Επιβλέπων: Κωνσταντίνος Στεργίου

Επίκουρος Καθηγητής Π.Δ.Μ.

Κοζάνη, Μάρτιος 2014

Πανεπιστήμιο Δυτικής Μακεδονίας
Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Υλοποίηση και πειραματική σύγκριση δύο γενικών αλγορίθμων συνέπειας τόξου (arc consistency)"

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΔΑΜΙΑΝΟΥ Α. ΜΕΤΙΚΑΡΙΔΗ

Επιβλέπων: Κωνσταντίνος Στεργίου
Επίκουρος Καθηγητής Π.Δ.Μ.

Εγκρίθηκε από την εξεταστική επιτροπή την 19η Μαρτίου 2014.

(Υπογραφή)	(Υπογραφή) .
.....	.....
Κ. Στεργίου	Π. Αγγελίδης
Επ. Καθηγητής Π.Δ.Μ.	Επ. Καθηγητής Π.Δ.Μ.

Κοζάνη, Μάρτιος 2014

Πανεπιστήμιο Δυτικής Μακεδονίας
Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

(Υπογραφή)

.....

Δαμιανός Α. Μετικαρίδης
Διπλωματούχος Μηχανικός Πληροφορικής και Τηλεπικοινωνιών ΠΔΜ

Ευχαριστίες

Από την πρώτη στιγμή της ένταξης μου στη Σχολή ,ο αγώνας για μόρφωση και γνώση ήταν σκληρός και ανελλιπής. Άπειρες ώρες δαπανήθηκαν στο διάβασμα για τις εξεταστικές περιόδους και για την ολοκλήρωση των απαραίτητων εργασιών εξαμήνου. Η εκπόνηση αυτής της διπλωματικής εργασίας είναι η τελευταία και τελική υποχρέωση που οφείλω να φέρω εις πέρας για να κερδίσω τον τίτλο του "Μηχανικού Πληροφορικής & Τηλεπικοινωνιών". Είναι πλέον βέβαιο ότι δεν θα κατάφερνα ποτέ να φτάσω ως εδώ χωρίς την καθοριστική συμβολή συγκεκριμένων παραγόντων ,προσώπων και γεγονότων καθ'όλην αυτήν την μακρά πορεία μου.

Θα ήθελα λοιπόν καταρχήν να ευχαριστήσω Τον Θεό που με εξόπλισε με μεγάλα αποθέματα ψυχικής αντοχής, θέλησης και αφοσίωσης ,τα οποία αποδείχθηκαν ιδιαίτερα χρήσιμα όλες τις στιγμές που μελετούσα για τις εξεταστικές καθώς και για την διεκπεραίωση των εργασιών που οι καθηγητές μου ανέθεταν .Κυρίως όμως Τον ευχαριστώ γιατί με εξόπλισε με αγάπη για την γνώση και το αντικείμενο το οποίο σπούδασα.

Θα ήθελα επίσης να ευχαριστήσω όλους τους καθηγητές που τα τελευταία 5 χρόνια μου δίδαξαν το αντικείμενο του Μηχανικού Πληροφορικής και Τηλεπικοινωνιών .Με κάποιους από αυτούς είχαμε άριστη συνεργασία και σχέσεις και πραγματικά είμαι ευτυχής που τους συνάντησα στην φοιτητική μου καριέρα. Για κάποιους άλλους όμως έχω πολλά παράπονα να εκφράσω. Εν πάσει περιπτώσει όμως τους ευχαριστώ, γιατί όλοι τους προσπάθησαν να είναι κάνουν σωστά την δουλειά τους.

Όσον αφορά την παρούσα διπλωματική εργασία θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου στον κύριο Κ. Στεργίου για την επίβλεψη του και για την ευκαιρία που μου έδωσε να ασχοληθώ περαιτέρω με το αντικείμενο της Τεχνητής Νοημοσύνης .Ήταν πάντα διαθέσιμος να μου προσφέρει τις γνώσεις και την εμπειρία του για τη βαθύτερη κατανόηση της περιοχής αυτού του αντικειμένου και ήταν πολύ συνεργάσιμος και βοηθητικός σε κάθε μου πρόταση και απορία.

Επίσης θέλω να ευχαριστήσω τους φίλους και συναδέλφους μου, τους οποίους γνώρισα κατά την φοιτητική μου καριέρα , επειδή με την συντροφικότητα και την παρέα τους

συνέβαλλαν αποτελεσματικά ώστε τα φοιτητικά μου χρόνια να γίνουν ίσως, οι καλύτερες αναμνήσεις που είχα στην μέχρι τώρα ζωή μου.

Σε αυτό το σημείο θέλω να αναφέρω και να ευχαριστήσω ανθρώπους, εκτός του στενού ακαδημαϊκού περιβάλλοντος, που υπήρξαν σημαντικοί παράγοντες στην ανάπτυξη του ατόμου μου, δηλαδή τους συγγενείς ,την οικογένεια και τους γονείς μου . Οι τελευταίοι, με γαλούχησαν με αγάπη και αφοσίωση ώστε να γίνω ο άνθρωπος που είμαι σήμερα. Τους ευχαριστώ για την οικονομική στήριξη που μου παρείχαν για να σπουδάσω, στερούμενοι κατ'αυτόν τον τρόπο τις ανέσεις τους και ιδιαίτερα την μητέρα μου Δέσποινα, στην οποία επιθυμώ να αφιερώσω την παρούσα εργασία.

Περίληψη

Το αντικείμενο αυτής της διπλωματικής εργασίας είναι η συνέπεια τόξου σε δυαδικά προβλήματα ικανοποίησης περιορισμών με την χρήση των αλγορίθμων AC-3 και AC-4. Τα προβλήματα ικανοποίησης περιορισμών είναι προβλήματα αναζήτησης με ιδιαίτερα χαρακτηριστικά που τα διαφοροποιούν από τα τυπικά προβλήματα αναζήτησης. Ανήκουν σε μια σημαντική υποκατηγορία της Τεχνητής Νοημοσύνης.

Ο στόχος της παρούσας διπλωματικής εργασίας είναι η υλοποίηση και έπειτα η πειραματική σύγκριση δύο αλγορίθμων επίτευξης συνέπειας τόξου ,των AC-3 (Arc Consistency 3 Alan Mackworth,1977) και AC-4 (Arc Consistency 4 Mohr & Henderson ,1986). Η υλοποίηση των δύο αυτών αλγορίθμων έγινε προγραμματιστικά ,με την χρήση της γλώσσας προγραμματισμού Java. Η πειραματική σύγκριση έγινε με την βοήθεια κάποιων δεδομένων προβλημάτων με περιορισμούς τα οποία χρησιμοποιήθηκαν ως δεδομένα εισόδου για τους παραπάνω αλγορίθμους. Τα αποτελέσματα που παρήγαγε η χρήση αυτών προβλημάτων σε συνδυασμό με τους αλγορίθμους είναι ικανά και αρκετά για να έχουμε μια έγκυρη πειραματική σύγκριση.

Τα κύρια συμπεράσματα της εργασίας αφορούν την αποτελεσματικότητα, και την αποδοτικότητα των αλγορίθμων. Η τελευταία θα αξιολογηθεί με κριτήρια την χωρική αλλά και την χρονική πολυπλοκότητα.

Όσον αφορά λοιπόν την αποτελεσματικότητα των αλγορίθμων, τα αποτελέσματα τις εργασίας επιβεβαιώνουν ότι και οι δύο αλγόριθμοι διαγράφουν ακριβώς τις ίδιες τιμές από τα πεδία τιμών των μεταβλητών που συμμετέχουν στους περιορισμούς.

Όσον αφορά την αποδοτικότητα της χωρικής πολυπλοκότητας , με βάση τα αποτελέσματα τις εκτέλεσης των αλγορίθμων μπορούμε να πούμε ότι ο αλγόριθμος AC-3 πετυχαίνει πάντα καλύτερα αποτελέσματα επιβεβαιώνοντας τον ισχυρισμό της χωρικής ανάλυσης που θέλει τον αλγόριθμο AC-3 να έχει O(e) χωρική πολυπλοκότητα έναντι O(ed^2) για τον AC-4.

Όσον αφορά την αποδοτικότητα της χρονικής πολυπλοκότητας , παρά το γεγονός ότι με βάση την χρονική ανάλυση ο αλγόριθμος AC-3 έχει χρόνο χειρότερης εκτέλεσης O(ed^3) ενώ ο AC-4 έχει O(ed^2) (όπου e ο αριθμός των περιορισμών και d ο αριθμός των τιμών του

μέγιστου πεδίου ορισμού), αποδείχθηκε ότι ο αλγόριθμος AC-3 καταναλώνει λιγότερο χρόνο εκτέλεσης για τα προβλήματα που έχουν λίγες μεταβλητής και μικρό μέγιστο πεδίο ορισμού σε σύγκριση με τον AC-4. Επίσης ο αλγόριθμος AC-3 πετυχαίνει καλύτερα αποτελέσματα όταν χρησιμοποιείται σε προβλήματα που παρουσιάζουν εξ αρχής συνέπεια τόξου, ή είναι εξ αρχής κοντά στην συνέπεια τόξου, οπότε και οι τελικές διαγραφές τιμών είναι λίγες. Ωστόσο για τα προβλήματα με μεγάλο αριθμό μεταβλητών και μέγιστου πεδίου ορισμού που βρίσκονται εξ αρχής μακριά από την συνέπεια τόξου, οπότε και ο αλγόριθμος AC-4 πραγματοποιεί πολλές διαγραφές τιμών, τα αποτελέσματα τις χρονικής ανάλυσης επιβεβαιώνονται περίτρανα και γίνεται φανερό ότι ο αλγόριθμος AC-4 πετυχαίνει καλύτερα χρονικά αποτελέσματα από τον AC-3.

Abstract

The topic of this diploma thesis is achieving arc consistency in binary constraint satisfaction problems by using the AC-3 and AC-4 algorithms. Constraint satisfaction problems are search problems with particular characteristics ,which differentiate them from typical search problems. They belong to an important subclass of Artificial Intelligence.

The purpose of this diploma thesis is to implement and then to experimentally compare two arc consistency algorithms; AC-3, (Arc Consistency 3, credited to Alan Mackworth, 1977) and AC-4 (Arc Consistency 4, credited to Mohr & Henderson ,1986). The implementation of these algorithms was done using the Java programming language. The experimental comparison was done using some constraint problems as data input to the algorithms named above. The results produced by the use of these problems in combination with the algorithms are capable and sufficient to achieve an accurate experimental comparison.

The main conclusions from this diploma thesis concern the effectiveness and the efficiency of these algorithms. The latter will be evaluated using space and time complexity as our main criteria.

Concerning the effectiveness of the algorithms, all execution results confirm that both algorithms are capable of achieving the exact same local consistency by deleting the same values from the domain of the variables which participate in the constraints.

Concerning space complexity efficiency ,based on the execution results, we can claim that the AC-3 algorithm can achieve better results, confirming the assertion of space complexity that AC-3 has a O(e) space complexity versus O(ed^2) for AC-4.

With respect to time complexity efficiency, despite the fact that time analysis gives to AC-3 a time complexity of O(ed^3) whereas AC-4 has a time complexity of O(ed^2) (where “e” stands for the number of constraints and “d” is the number of values that the biggest domain has), it is proved that the AC-3 algorithm uses less time to execute when problems with a small number of variables and a small maximum domain are given as input ,compared to AC-4. Furthermore the AC-3 algorithm achieves better time results when it is used on problems which are arc consistent or which are close to arc consistency from the beginning,

thus only a few value deletions are finally made by the algorithm. Never the less, the results of time complexity are to be soundly confirmed for problems with a large number of variables and a large maximum domain, especially when these problems are from the beginning far from arc consistency, thus the AC-4 algorithm will finally delete many values. In such cases it becomes obvious that the AC-4 algorithm can achieve better time results than the AC-3 algorithm.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Το αντικείμενο αυτής της διπλωματικής εργασίας είναι η συνέπεια τόξου σε δυαδικά προβλήματα ικανοποίησης περιορισμών με την χρήση των αλγορίθμων AC-3 και AC-4. Τα προβλήματα ικανοποίησης περιορισμών είναι σημαντικά για την Τεχνητή Νοημοσύνη επειδή πολλά προβλήματα αναζήτησης ,όπως για παράδειγμα ο χρωματισμός γράφων, πολλά λογικά παζλ ,η σχεδίαση και ο προγραμματισμός εργασιών ,μπορούν να αναπαρασταθούν ως προβλήματα ικανοποίησης περιορισμών όπου η λύση σε αυτά τα προβλήματα πρέπει να ικανοποιεί συγκεκριμένους περιορισμούς.

Παραδοσιακά τέτοιου είδους προβλήματα επιλύνονται με την μέθοδο της οπισθοδρομικής αναζήτησης “backtrack search” ,ωστόσο αυτή η μέθοδος από μόνη της δεν είναι καθόλου αποδοτική. Γι’αυτό έχουν εισαχθεί πολλές μέθοδοι ελέγχου συνέπειας ,οι οποίες συνδυάζονται με την μέθοδο της οπισθοδρομικής αναζήτησης. Έτσι η αποδοτικότητα βελτιώνεται σημαντικά.

Αυτές οι τεχνικές αποσκοπούν στο να μειώσουν όσο το δυνατόν περισσότερο τον χώρο αναζήτησης ενός προβλήματος ικανοποίησης περιορισμών. Για να επιτύχουν κάτι τέτοιο εξετάζουν την ικανοποίηση κάποιων συνδυασμών περιορισμών έτσι ώστε να διαγραφούν όλες οι τιμές των μεταβλητών που συμμετέχουν σε έναν περιορισμό και δεν ικανοποιούν τους συνδυασμούς περιορισμών που εξετάζει η συγκεκριμένη τεχνική. Για παράδειγμα κάποιες από αυτές τις τεχνικές είναι η συνέπεια διαδρομής (path consistency), η συνέπια κόμβου (node consistency) ,η συνέπεια τόξου (arc consistency) .Στην παρούσα διπλωματική εργασία θα ασχοληθούμε μόνο με την τελευταία που είναι και η πιο σημαντική. Έτσι λοιπόν οι αλγόριθμοι AC-3 και AC-4 με τους οποίους θα ασχοληθούμε αποκλειστικά, είναι αλγόριθμοι που επιτυγχάνουν τέτοιου είδους συνέπεια.

Στόχος μας σε αυτήν την διπλωματική εργασία θα είναι να υλοποιήσουμε προγραμματιστικά αυτούς τους δύο αλγορίθμους, χρησιμοποιώντας την γλώσσα προγραμματισμού Java και στην συνέχεια να συγκρίνουμε τις επιδώσεις τους χρησιμοποιώντας ως είσοδο

κάποια προβλήματα με περιορισμούς τα οποία θα τρέξουμε και έπειτα θα συγκρίνουμε τα αποτελέσματα μελετώντας την αποτελεσματικότητα και την χωρική-χρονική αποδοτικότητα των αλγορίθμων.

1.1 Συμβολή Διπλωματικής Εργασίας

Το ευρύτερο πλαίσιο στο οποίο εντάσσεται η εν λόγω εργασία είναι εκείνο των προβλημάτων ικανοποίησης περιορισμών και στην χρήση των αλγορίθμων AC-3 & AC-4 οι οποίοι είναι τεχνικές επίτευξης συνέπειας τόξου.
Η συμβολή της παρούσας διπλωματικής εργασίας μπορεί να συνοψισθεί στα εξής δύο σημεία:

Στο πρώτο μέρος της εργασίας, μέχρι και το κεφάλαιο 6, παρουσιάζεται μια θεωρητική αναφορά η οποία καλύπτει τις βασικές έννοιες που θα πρέπει κανείς να γνωρίζει για να μελετήσει κατά βάθος τους αλγορίθμους και να κατανοήσει τα συμπεράσματα που προέκυψαν από την εκτέλεση αυτών. Συγκεκριμένα δίνεται ο εκτενής ορισμός των προβλημάτων ικανοποίησης περιορισμών ,με όσες λεπτομέρειες κρίνονται απαραίτητες. Επίσης δίνεται ο εκτενής ορισμός της έννοιας «συνέπεια τόξου» καθώς και το πώς αυτή σχετίζεται με τα προβλήματα ικανοποίησης περιορισμών. Στη συνέχεια παρατίθεται ο αλγόριθμος AC-3 και ο αλγόριθμος AC-4 οι οποίοι συνοδεύονται από τον ψευδοκώδικα, τον τρόπο λειτουργίας ,τις καινοτομίες ,την ανάλυση πολυπλοκότητας και τα πλεονεκτήματα-μειονεκτήματα αυτών.

Στο δεύτερο μέρος της εργασίας, κεφάλαια 7-8, παρουσιάζεται η υλοποίηση και τα αποτελέσματα των αλγορίθμων. Πιο συγκεκριμένα , παρουσιάζεται η εκτέλεση των αλγορίθμων σε μη γραφικό περιβάλλον καθώς επίσης και σε γραφικό περιβάλλον. Στη συνέχεια δίνεται η περιγραφή των δομών δεδομένων που χρησιμοποιήθηκαν κατά την προγραμματιστική υλοποίηση των αλγόριθμων AC-3 και AC-4. Έπειτα ακολουθούν τα στατιστικά στοιχεία που προέκυψαν από την εκτέλεση των αλγορίθμων και τέλος τα συμπεράσματα της εκτέλεσης τα οποία είναι και τα τελικά συμπεράσματα για τους αλγορίθμους.

1.2 Οργάνωση Τόμου

Η εργασία αυτή είναι οργανωμένη σε επτά κεφάλαια:

- 1.Στο πρώτο κεφάλαιο παρατίθεται η εισαγωγή για την διπλωματική εργασία.
- 2. Στο δεύτερο κεφάλαιο αναφέρονται και αναλύονται ορισμένα από τα βασικά.
- χαρακτηριστικά των προβλημάτων ικανοποίησης περιορισμών.
- 3. Στο τρίτο κεφάλαιο ορίζεται διεξοδικά η έννοια της συνέπειας τόξου**.**
- 4. Στο τέταρτο κεφάλαιο ορίζεται και αναλύεται ο αλγόριθμος AC-3.
- 5. Στο πέμπτο κεφάλαιο ορίζεται και αναλύεται ο αλγόριθμος AC-4.
- 6. Στο έκτο κεφάλαιο παρουσιάζεται αναλυτικά η εκτέλεση των αλγορίθμων σε γραφικό και μη γραφικό περιβάλλον, οι δομές δεδομένων που χρησιμοποιούν οι δυο παραπάνω αλγόριθμοι και τα στατιστικά στοιχεία που εξήχθησαν από την εκτέλεση των αλγορίθμων.
- 7. Στο έβδομο και τελευταίο κεφάλαιο παρουσιάζονται τα τελικά συμπεράσματα που προέκυψαν από την εκτέλεση των αλγορίθμων.

Κεφάλαιο 2

Προβλήματα Ικανοποίησης Περιορισμών (CSP)

Στο τρέχον κεφάλαιο της εργασίας παρουσιάζονται συνοπτικά ο ορισμός, οι κατηγορίες και τα χαρακτηριστικά των προβλημάτων ικανοποίησης περιορισμών (Constraint Satisfaction Problems-CSP). Έπειτα γίνεται λόγος για τον ο τρόπο επίλυσης τους. Στη συνέχεια παραδίδεται ένα παράδειγμα τέτοιου προβλήματος. Τέλος δίνονται οι κατηγορίες των προβλημάτων τα οποία μπορούν να μοντελοποιηθούν ως προβλήματα ικανοποίησης περιορισμών.

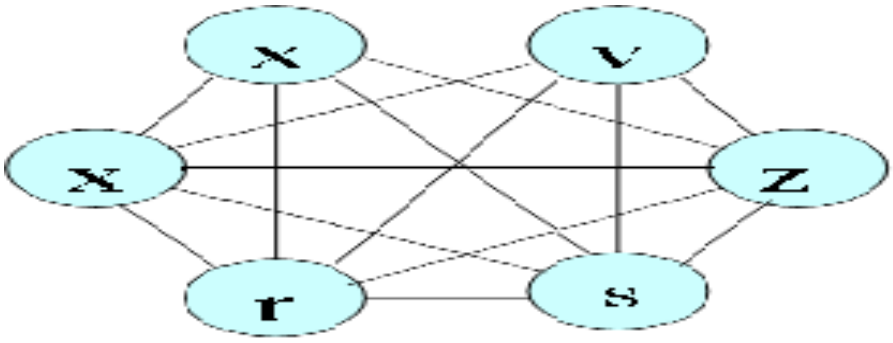
2.1 Ορισμός , Κατηγορίες και Χαρακτηριστικά των CSP.

Ένα πρόβλημα ικανοποίησης περιορισμών ορίζεται από τα εξής τρία χαρακτηριστικά: Ένα σύνολο μεταβλητών (variables) *V*1,…,*V*n. Ένα σύνολο πεδίων ορισμού (domains) *D1,…,Dk* ,ένα για κάθε μεταβλητή με τις πιθανές της (values).Ένα σύνολο περιορισμών (constraints) *C*1,…,*C*m, όπου κάθε περιορισμός περιλαμβάνει ένα υποσύνολο των μεταβλητών και προσδιορίζει τους επιτρεπτούς συνδυασμούς τιμών για αυτό το υποσύνολο. Ένας *n-αδικός* (n-ary) περιορισμός Ci σε ένα σύνολο μεταβλητών *V*1,…,*V*k είναι ένα υποσύνολο του καρτεσιανού γινομένου *D*1 x…x *D*k

Τα προβλήματα ικανοποίησης περιορισμών χωρίζονται στις εξής κατηγορίες: αυτά που αποτελούνται από περιορισμούς οι οποίοι περιλαμβάνουν μόνο μια μεταβλητή(unary problems), για παράδειγμα X > 5 .Αυτά που αποτελούνται από περιορισμούς οι οποίοι περιλαμβάνουν δύο μεταβλητές (binary problems), για παράδειγμα X = Y + 5. Και αυτά που αποτελούνται από περιορισμούς οι οποίοι περιλαμβάνουν περισσότερες των δύο μεταβλητών (n-ary problems), για παράδειγμα X = Y –Z + 5.

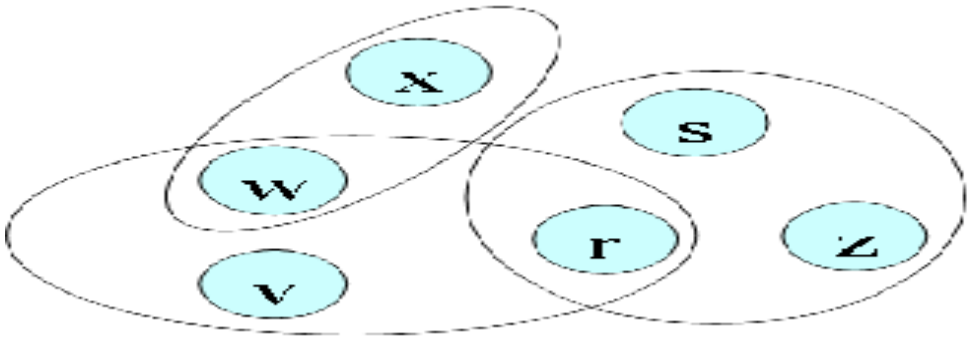
Κάθε δυαδικό πρόβλημα ικανοποίησης περιορισμών μπορεί να αναπαρασταθεί ως ένα γράφημα όπου οι μεταβλητές του προβλήματος αντιστοιχούν στους κόμβους του δικτύου ,ενώ

οι δυαδικοί περιορισμοί μεταξύ των μεταβλητών είναι οι ακμές που ενώνουν τους κόμβους του δικτύου.



Σχήμα 2.1.α: Δίκτυο περιορισμών όπου οι μεταβλητές είναι οι κόμβοι του δικτύου ,ενώ οι δυαδικοί περιορισμοί είναι οι ακμές του δικτύου.

Κάθε ν-δικό πρόβλημα ικανοποίησης περιορισμών μπορεί να αναπαρασταθεί ως ένα γράφημα όπου οι μεταβλητές του προβλήματος αντιστοιχούν στους κόμβους του δικτύου ,ενώ οι ν-αδικοί περιορισμοί μεταξύ των μεταβλητών είναι οι υπερακμές που περιλαμβάνουν τους κόμβους του δικτύου.



Σχήμα 2.1.β: Δίκτυο περιορισμών όπου οι μεταβλητές είναι οι κόμβοι του δικτύου ,ενώ οι ν-αδικοί περιορισμοί είναι οι υπερακμές του δικτύου.

Ένα CSP είναι ένα πρόβλημα αναζήτησης με ιδιαίτερα χαρακτηριστικά που το διαφοροποιούν από τα γενικά προβλήματα αναζήτησης. Συγκεκριμένα ο χώρος και το δέντρο

αναζήτησης είναι πεπερασμένα. Το ζητούμενο είναι να αναθέσουμε τιμές σε ένα σύνολο μεταβλητών ώστε να ικανοποιούνται κάποιοι περιορισμοί. Δε μας ενδιαφέρει το μονοπάτι προς τη λύση.

2.2 Στόχος και Επίλυση CSP.

Το ζητούμενο στα προβλήματα ικανοποίησης περιορισμών (constraint satisfaction problem – CSP) είναι να αναθέσουμε τιμές σε ένα σύνολο μεταβλητών έτσι ώστε να ικανοποιούνται όλοι οι περιορισμοί του προβλήματος. Οι περιορισμοί συμπεριλαμβάνουν συνδυασμούς μεταβλητών οι οποίες λαμβάνουν τιμές από επιτρεπτά πεδία ορισμού. Σε αυτήν την διπλωματική αυτά τα πεδία ορισμού είναι πάντα πεπερασμένα.

Ο πιθανός στόχος σε ένα τέτοιο πρόβλημα θα μπορούσε να είναι ένας από τους εξής: Η εύρεση μιας και μόνο λύσης στο πρόβλημα. Θα μπορούσε όμως να είναι και η εύρεση όλων των πιθανών λύσεων του προβλήματος. Επίσης θα μπορούσε να είναι η εύρεση μιας λύσης η οποία μεγιστοποιεί ή ελαχιστοποιεί κάποια ποσότητα (πρόβλημα βελτιστοποίησης). Τέλος θα μπορούσε να είναι η εύρεση μιας προσεγγιστικής λύσης στο πρόβλημα.

Η λύση ενός προβλήματος ικανοποίησης περιορισμών επιφέρεται με την ανάθεση μιας τιμής σε κάθε μεταβλητή έτσι ώστε να μην παραβιάζεται κανένας περιορισμός. Τα προβλήματα τα οποία έχουν πεπερασμένα πεδία ορισμού για τις μεταβλητές τους, όπως για παράδειγμα αυτά που θα εξετάσουμε στην παρούσα εργασία μπορούν να επιλυθούν με μια πληθώρα αλγορίθμων. Αυτοί οι αλγόριθμοι αν και σίγουρα θα βρουν την λύση, δεν εγγυώνται όλοι την ίδια αποδοτικότητα. Για παράδειγμα τέτοιοι αλγόριθμοι θα μπορούσαν να είναι οι παρακάτω:

1.**Generate and test**

Βρίσκει όλες τις τιμές οι οποίες θα μπορούσαν να είναι πιθανές λύσεις και τις δοκιμάζει κατά σειρά. Αν και αυτή είναι μια πολύ απλή brute force τεχνική ,είναι πολύ χρονοβόρα και συνεπώς μη αποδοτική.

2.Depth-first search

Η αναζήτηση ξεκινά από την ρίζα του δέντρου και συνεχίζεται όσο πιο

μακριά είναι δυνατόν, κατεβαίνοντας πρώτα προς τα κάτω (κάθετη αναζήτηση στο δέντρο).

3. **Οπισθοδρομική αναζήτηση- Backtrack search**

Ένας γενικός αλγόριθμος για την εύρεση όλων ή κάποιων λύσεων ενός προβλήματος CSP, ο οποίος σταδιακά χτίζει μερικές λύσεις και τελικά εγκαταλείπει κάθε μερική λύση η οποία δεν μπορεί να είναι ουσιαστικά η λύση του προβλήματος (οπισθοδρόμηση). Ο χώρος αναζήτησης είναι μια δομή δέντρου όπου οι μεταβλητές του προβλήματος αντιπροσωπεύονται ως κόμβοι του δέντρου. Η αναζήτηση ξεκινάει από την ρίζα του δέντρου και επεκτείνεται προς τα κάτω με ένα βήμα κάθε φορά, αναθέτοντας τιμές στις μεταβλητές και ελέγχοντας έπειτα τους περιορισμούς μέχρι να φτάσει σε πλήρη λύση. Σε κάθε κόμβο ,ο αλγόριθμος ελέγχει αν η συγκεκριμένος τιμή που δόθηκε στον κόμβο μπορεί να συμπεριληφθεί στην λύση. Αν δεν μπορεί τότε ο αλγόριθμος οπισθοδρομεί στην αμέσως προηγούμενη επιλογή του και δοκιμάζει μια άλλη τιμή για τη συγκεκριμένη μεταβλητή.

4.Arc Consistency

Η συνέπεια τόξου δεν εξασφαλίζει από μόνη της την λύση σε ένα πρόβλημα ικανοποίησης περιορισμών επειδή επιτυγχάνει τοπική συνέπεια και όχι ολική global consistency (global solution). Ωστόσο χρησιμοποιείται σε τέτοιου είδους προβλήματα ως προπαρασκευαστικό μέτρο με σκοπό να αφαιρεθούν όσο το δυνατόν περισσότερες τιμές από τα πεδία ορισμού των μεταβλητών, εφ'όσον αυτές οι τιμές δεν ικανοποιούν τους περιορισμούς του προβλήματος.

5.Depth-first search combined with arc consistency

Αυτή είναι η τεχνική που χρησιμοποιείται κατά κόρον στα πραγματικά προβλήματα CSP, και είναι συνδυασμός της τεχνικής συνέπειας τόξου με τον αλγόριθμο οπισθοδρομικής αναζήτησης.

2.3 Παράδειγμα ενός CSP.

Έχουμε το CSP με τις επακόλουθες μεταβλητές ,τα πεδία ορισμού γι'αυτές και τους περιορισμούς:

X με πεδίο ορισμού (1 2 4 5 6 7 9 10 11)

Y με πεδίο ορισμού (3 5 7 8 9)

Z με πεδίο ορισμού (0 5 6 3 8 10 11)

W με πεδίο ορισμού (10 11 4 5 6 7)

Οι περιορισμοί είναι οι παρακάτω:

$X < 10$

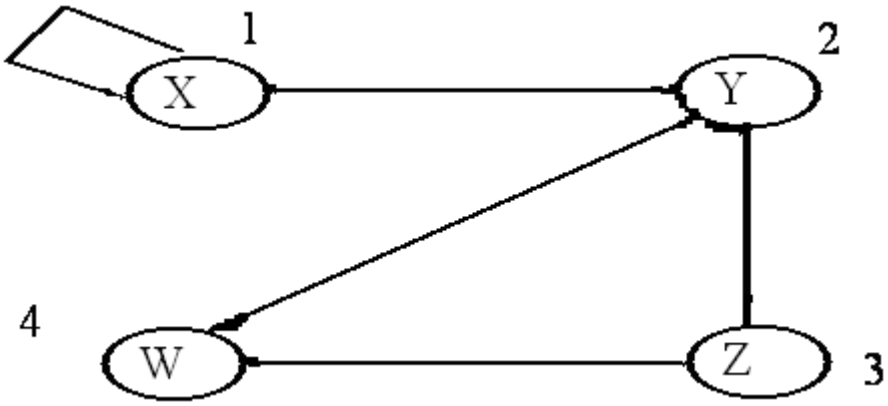
$Y = X + 1$

$Z \geq Y + 1$

$W \neq Y$

$W \neq Z$

Αυτό το CSP μπορεί να αναπαρασταθεί ως ένας γράφος G στον οποίο ο κάθε κόμβος αναπαριστά μια μεταβλητή και κάθε ακμή μεταξύ δύο μεταβλητών αναπαριστά έναν περιορισμό.



Σχήμα 2.3: Αναπαράσταση ενός CSP ως γράφο. Κάθε κόμβος αναπαριστά μια μεταβλητή και κάθε ακμή μεταξύ δύο μεταβλητών αναπαριστά έναν περιορισμό.

Το ζητούμενο είναι να βρεθούν όλα τα δυνατά συμβατά πεδία τιμών για τις μεταβλητές X, Y, W, και Z.

2.4 Εφαρμογές Μοντέλου CSP.

Τα προβλήματα ικανοποίησης περιορισμών είναι μια απλή μοντελοποίηση προβλημάτων. Παρ'όλα αυτά ,η συγκεκριμένη προσέγγιση μπορεί να χρησιμοποιηθεί για να μοντελοποιήσει πολλά πρακτικά προβλήματα όπως τα παρακάτω:

Χρονική Συλλογιστική (Temporal Reasoning)
Χωρική Συλλογιστική (Spatial Reasoning)
Δρομολόγηση Οχημάτων (Vehicle Routing)
Κατανομή Πόρων (Resource allocation)
Σύνθεση Ανθρώπινης Ομάδας (Manpower rostering**)**
Σχεδιασμός Ενεργειών (Planning)
Κατανομή Συχνοτήτων (Frequency Assignment)
Ακέραιος, γραμμικός και μη-γραμμικός προγραμματισμός (integer, linear and non-linear programming) - χρωματισμός γράφων, χρονοπρογραμματισμός, n-queens
Timetabling
Scheduling, sequencing - aircrew, job-shop
Και πολλά άλλα προβλήματα βελτιστοποίησης.

Κεφάλαιο 3

Συνέπεια Τόξου

Σε αυτό το κεφάλαιο της εργασίας παρουσιάζεται η έννοια της συνέπειας τόξου. Αρχικά δίνεται ο μαθηματικός ορισμός αυτής της έννοιας. Έπειτα δίνονται απαντήσεις στα ερωτήματα τι είναι συνέπεια τόξου και πότε εφαρμόζεται. Τέλος παρουσιάζεται ένα πολύ απλό παράδειγμα ελέγχου συνέπειας τόξου.

3.1 Ορισμός Συνέπειας Τόξου.

Έστω N ένα δίκτυο δυαδικών περιορισμών ,με V το σύνολο των μεταβλητών, D το σύνολο των πεδίων ορισμού των μεταβλητών και C το σύνολο των περιορισμών του δικτύου.

Η μεταβλητή Vi παρουσιάζει συνέπεια τόξου σχετικά με τη μεταβλητή Vj ,αν για κάθε τιμή ai που ανήκει στο πεδίου ορισμού Di, υπάρχει τουλάχιστον μια τιμή bj που ανήκει στο πεδίο ορισμού Dj της μεταβλητής Vj έτσι ώστε ο συνδυασμός των τιμών (ai,bj) να ικανοποιούν τον περιορισμό Ci,j που περιλαμβάνει τις μεταβλητές Vi και Vj.

Ο περιορισμός Cij που περιλαμβάνει τις μεταβλητές Vi και Vj παρουσιάζει συνέπεια τόξου όταν η μεταβλητή Vi παρουσιάζει συνέπεια τόξου ως προς την μεταβλητή Vj και το αντίστροφο.

Το δίκτυο N παρουσιάζει συνέπεια τόξου αν κάθε περιορισμός στο σύνολο C παρουσιάζει συνέπεια τόξου.

3.2 Τι είναι Συνέπεια Τόξου ,Που και Πότε Εφαρμόζεται.

Η συνέπεια τόξου (Arc Consistency) είναι ο παλαιότερος και ο πιο γνωστός τρόπος διάδοσης περιορισμών (Constraints Propagation). Είναι μια σχετικά απλή και ευκόλως κατανοητή τεχνική ,η οποία εξασφαλίζει ότι κάθε τιμή σε ένα πεδίο ορισμού είναι συνεπής με κάθε ζευγάρι περιορισμών. Είναι επίσης αρκετά αποδοτική τεχνική η οποία καταφέρνει να

εξαλείψει τις μη συνεπείς τιμές στα πεδία ορισμού των μεταβλητών, με σχετικά μικρό χρονικό και χωρικό κόστος.

Μπορεί να εφαρμοστεί ως προπαρασκευαστικό μέτρο στα προβλήματα ικανοποίησης περιορισμών, πριν την εκτέλεση κάποιου αλγόριθμου οπισθοδρομικής αναδόμησης backtracking search. Η συνέπεια τόξου είναι μια από τις πιο σημαντικές τεχνικές για τον έλεγχο ασυνεπειών και οποίες χρησιμοποιείται σε unary και binary προβλήματα ικανοποίησης περιορισμών.

Φυσικά υπάρχουν κι άλλες τέτοιες τεχνικές ελέγχου συνέπειας. Ωστόσο στην παρούσα εργασία θα ασχοληθούμε μόνο με την συνέπεια τόξου, στην οποία αναφέρονται και οι αλγόριθμοι AC-3 και AC-4 που θα αναπτύξουμε. Η συνέπεια τόξου είναι μάλιστα η πιο φθηνή και αποδοτική από όλες τις γνωστές τεχνικές.

3.3 Παράδειγμα Ελέγχου Συνέπειας Τόξου.

Η έννοια του ελέγχου συνέπειας τόξου είναι σχετικά απλή. Για παράδειγμα έστω δύο μεταβλητές , X και Y οι οποίες συμμετέχουν στον περιορισμό $X = Y$ και παίρνουν τιμές από τα παρακάτω πεδία αντίστοιχα :

Dx=(2, 3, 4, 5,7)

Dy=(4, 5, 6)

Ο περιορισμός X = Y δείχνει ότι μια τιμή του πεδίου Dx πρέπει να υπάρχει και στο πεδίο Dy και αντίστροφα. Για να ικανοποιηθεί ο περιορισμός πρέπει να αφαιρεθούν όλες οι μη συνεπείς από τα πεδία Dx και Dy. Οπότε αρχικά θα αφαιρεθούν οι τιμές 2, 3 ,7 από το πεδίο ορισμού Dx και έπειτα η τιμή 6 από το Dy. Συνεπώς το τελικό πεδίο ορισμού Dx = Dy είναι το (4 ,5).

Αυτή η διαδικασία ονομάζεται έλεγχος συνέπειας τόξου και μπορεί να μειώσει κατά πολύ τον χώρο αναζήτησης των λύσεων όταν εφαρμοστεί σε ένα CSP.

3.3.1 Ένα Διαισθητικό Παράδειγμα.

Έστω ότι έχουμε το παρακάτω CSP,

Μεταβλητές: X, Y, Z

Πεδία Ορισμού: Dx = (4 5 6 7),

Dy = (4 5 6 8 9)

Dz = (3 5 6 7 9)

Περιορισμοί: X = Y, Y = Z.

** Πρώτα ελέγχουμε τον περιορισμό: X = Y

Παίρνουμε: Dx = (4 5 6),

Dy = (4 5 6),

Dz καμία αλλαγή

*** Έπειτα ελέγχουμε και τον δεύτερο περιορισμό: Y = Z

Παίρνουμε: Dy = (5 6)

Dz = (5 6)

Dx = (4 5 6) (καμία αλλαγή)

Τώρα προκύπτει ένα πρόβλημα επειδή ο περιορισμός

X = Y, δεν ικανοποιείται πλέον

Άρα πρέπει να ελέγξουμε τον περιορισμό **X = Y** ξανά, ώστε το πεδίο ορισμού Dx, να γίνει επίσης Dx = (5 6).

3.4 Συνέπεια Τόξου και Τοπική Συνέπεια.

Οι αλγόριθμοι συνέπειας τόξου επιτυγχάνουν μόνο τοπική συνέπεια (***local consistency***), δηλαδή συνέπεια για κάθε τόξο. Όμως δεν επιτυγχάνουν ολική συνέπεια (***global consistency***) δηλαδή συνέπεια για ολόκληρο το πρόβλημα ικανοποίησης περιορισμών (CSP). Για παράδειγμα ,έχουμε ένα CSP με τις παρακάτω μεταβλητές:

X, Y , Z.

Η X έχει πεδίο ορισμού το (1 2 3),

Ενώ οι Y και Z έχουν και οι δύο το (1 2).

Οι περιορισμοί είναι οι:

$X \neq Y$, $Y \neq Z$, $X \neq Z$.

Χρησιμοποιώντας μόνο συνέπεια τόξου , καμία τιμή από τα πεδία ορισμού των μεταβλητών δεν μπορεί να διαγραφεί. Αυτό σημαίνει ότι οι περιορισμοί είναι συνεπείς.

Ωστόσο οι μεταβλητές X, Y και Z δεν μπορούν να πάρουν οποιαδήποτε τιμή από τα πεδία ορισμού τους επειδή για παράδειγμα όταν το $X = 2$ ή $X = 1$, το πρόβλημα είναι αδύνατο. Αυτό σημαίνει ότι οι μεταβλητή X δεν παρουσιάζει ολική συνέπεια ως προς το

πρόβλημα, επειδή παίρνοντας αυτές τις δύο τιμές δεν υπάρχει λύση για το πρόβλημα.

Η συνέπεια τόξου δεν μπορεί να εντοπίσει τέτοιου είδους ασυνέπειες. Άλλες

τεχνικές μπορούν να κάνουν κάτι τέτοιο , ωστόσο χρησιμοποιούνται σπάνια και δεν θα τις εξετάσουμε σε αυτή την διπλωματική εργασία.

Κεφάλαιο 4

Ο Αλγόριθμος AC-3

Στο κεφάλαιο 4 της παρούσας εργασίας παρουσιάζεται ο τρόπος λειτουργίας του αλγορίθμου AC-3. Στη συνέχεια παρατίθεται ο ψευδοκώδικας του συγκεκριμένου αλγορίθμου. Μετά τον ψευδοκώδικα ακολουθεί μια σύντομη ανάλυση του αλγορίθμου όσον αφορά την χρονική και την χωρική πολυπλοκότητα. Έπειτα δίνεται ένα παράδειγμα εκτέλεσης του αλγορίθμου. Και τέλος σχολιάζεται μια παραλλαγή του αλγορίθμου AC-3.

4.1 Ο Τρόπος Λειτουργίας.

Ο πλέον γνωστός αλγόριθμος συνέπειας τόξου είναι αυτός που προτάθηκε από τον Mackworth [4] , υπό το όνομα AC-3. Είναι ένας γενικός αλγόριθμος ο οποίος μπορεί να χρησιμοποιηθεί για να ελέγξει την συνέπεια τόξου οποιουδήποτε δυαδικού περιορισμού. Επειδή είναι γενικός αλγόριθμος όμως είναι και μη αποδοτικός για την επίλυση πραγματικών προβλημάτων ικανοποίησης περιορισμών ,μιας και δεν λαμβάνει υπ'όψη τα συγκεκριμένα χαρακτηριστικά του προβλήματος .

Ο αλγόριθμος εφαρμόζεται σε δυαδικά δίκτυα και για την ακρίβεια πετυχαίνει συνέπεια τόξου για τόξα 2 κορυφών .Είναι σχετικά απλώς στην κατανόησή του, αν και εισάγει μια καινοτόμα ιδέα σε σύγκριση με τους προκατόχους του, τους AC-1 και AC-2. Αυτή η ιδέα είναι ότι δεν χρειάζεται να επεξεργαστούμε από την αρχή όλους τους περιορισμούς αν μόνο λίγα πεδία ορισμού έχουν αλλάξει. Μπορούμε να κρατάμε τα τόξα των οποίων οι μεταβλητές υφίστανται αλλαγή από την διαγραφή τιμών σε μια ουρά ,έτσι ώστε να επεξεργαστούμε μόνο αυτά.

Ο αλγόριθμος αρχικά αρχικοποιεί την ουρά Q έτσι ώστε αυτή να περιέχει όλα τα τόξα (Vi,Vj) ,με i διάφορο του j ,που ανήκουν στο σύνολο των περιορισμών του προβλήματος.

Έπειτα χρησιμοποιεί έναν απλό βρόγχο ο οποίος σε κάθε επανάληψή αφαιρεί ένα τόξο (Vi,Vj) από την ουρά Q, με σκοπό να το αναθεωρήσει μέσω της ρουτίνας REVISE(Vi,Vj). Ο βρόγχος επαναλαμβάνεται μέχρις ότου η ουρά Q μείνει άδεια. Κάθε φορά που η ρουτίνα REVISE(Vi,Vj) πραγματοποιεί αλλαγές στα πεδία ορισμού της εκάστοτε μεταβλητής Vi, ο αλγόριθμος προσθέτει στην ουρά Q όλα τα τόξα τα οποία δεν βρίσκονται ήδη στην ουρά και περιλαμβάνουν την μεταβλητή Vi ως μια εκ των δύο κορυφών τους . Συνεπώς η ουρά Q αρχίζει να αποθηκεύει τα τόξα του δικτύου για τα οποία δεν είναι βέβαιο ότι υπάρχει συνέπεια τόξου. Μόλις η ουρά αδειάσει τελείως είναι πλέον επιβεβαιωμένο ότι όλα τα πεδία τιμών παρουσιάζουν συνέπεια για όλους τους περιορισμούς του προβλήματος. Με την χρήση της ουράς Q ο AC-3 αποφεύγει τις άσκοπες κλήσεις της ρουτίνας REVISE(Vi,Vj) , αντιθέτως με τους προκατόχους του , οι οποίοι δεν χρησιμοποιούν καμία ουρά.

Η ρουτίνα REVISE(Vi,Vj) είναι το κύριο συστατικό του αλγορίθμου AC-3. Δεδομένου ενός τόξου δύο κορυφών (Vi, Vj) , η ρουτίνα REVISE(Vi,Vj) παίρνει κάθε τιμή X που ανήκει στο πεδίο ορισμού Di και ελέγχει αν υπάρχει έστω και μια τιμή Y που να ανήκει στο πεδίου ορισμού Dj ,έτσι ώστε η τιμή X να παρουσιάζει συνέπεια τόξου ως προς την τιμή Y. Αν δεν βρεθεί καμία τέτοια τιμή Y, τότε η τιμή X αφαιρείται από το πεδίο Di . Η ρουτίνα επιστρέφει την τιμή true αν έστω και μια τιμή X έχει διαγραφεί από το πεδίο ορισμού Di .Αν καμία τιμή δεν έχει διαγραφεί ,αυτό σημαίνει ότι η μεταβλητή Vi παρουσιάζει συνέπεια τόξου ως προς την Vj και η ρουτίνα επιστρέφει false.

4.2 Ο Ψευδοκώδικας.

Στη συνέχεια ακολουθεί ο ψευδοκώδικας του αλγορίθμου συνέπειας τόξου AC-3 για τα προβλήματα ικανοποίησης περιορισμών [1] . Όπου Vi, Vj είναι τα σύνολα κορυφών – μεταβλητών. X,Y είναι συγκεκριμένες κορυφές –μεταβλητές. (Vi,Vj) είναι το τόξο μεταξύ των κορυφών -περιορισμός. Di και Dj είναι τα πεδία ορισμού της μεταβλητής Vi και Vj αντίστοιχα. Q είναι μια ουρά που κρατάει τα τόξα που πρέπει να ελεγχθούν για ασυνέπειες. Arcs(G) είναι το σύνολο των τόξων-περιορισμών του γραφήματος G.

Algorithm AC-3

```
procedure AC-3

  Q <- { (Vi,Vj) in arcs(G),i<>j };

  while not Q empty
      select and delete any arc (Vk,Vm) from Q;

      if REVISE(Vk,Vm) then          //if not arc consistent then…
            Q <- Q union {(Vi,Vk) such that (Vi,Vk) in arcs(G),i<>k,i<>m}
      endif

  endwhile

end AC-3
```

Algorithm REVISE

```
procedure REVISE(Vi,Vj)

  DELETE <- false;

  for each X in Di do

    if there is no such Y in Dj such that (X,Y) is consistent, then
         delete X from Di;
         DELETE <- true;
    endif;

  endfor;

return DELETE;   //I have deleted unsupported X from Di or          ---> true
                 //I haven't deleted X because it is supported     ---> false
end REVISE
```

4.3 Ανάλυση του AC-3.

Λήμμα

Ο AC-3 επιτυγχάνει συνέπεια τόξου σε δυαδικά δίκτυα σε χρόνο O(e*d^3) και χώρο O(e), όπου e το σύνολο των περιορισμών και d ο αριθμός των τιμών του πεδίου ορισμού με τις περισσότερες τιμές.

4.3.1 Απόδειξη Χρονικής Πολυπλοκότητας.

Έστω ένας περιορισμός Ci. Κάθε φορά που ο περιορισμός Ci ξαναμπαίνει στην ουρά Q το πεδίο ορισμού μιας από τις δύο μεταβλητές που ανήκουν στον περιορισμό έχει αλλάξει. Μιας και υπάρχουν το πολύ 2*d τιμές, δηλαδή d τιμές για την μία εκ των δύο μεταβλητών που συμμετέχουν στον δυαδικό περιορισμό και άλλες d τιμές για την άλλη μεταβλητή, ο αλγόριθμος AC3 θα επεξεργαστεί τον περιορισμό το πολύ 2*d φορές. Επειδή έχουμε e περιορισμούς και η επεξεργασία του καθενός παίρνει χρόνο ίσο με O(d^2), τελικά η χρονική πολυπλοκότητα του AC-3 είναι O(e * d^3).

Σημείωση: Αν το δίκτυο παρουσιάζει συνέπεια τόξου εξ αρχής ,τότε ο αλγόριθμος AC-3 θα χρειαστεί χρόνο ίσο με O(e * d^2), επειδή κάθε περιορισμός θα επεξεργαστεί μόνο μία φορά.

Η χρονική πολυπλοκότητα του αλγορίθμου AC-3 δεν είναι η βέλτιστη .Η χρήση της ουράς Q σε συνδυασμό με την ρουτίνα Revise έχει ως αποτέλεσμα να μην ελέγχονται ξανά οι περιορισμοί για τους οποίους γνωρίζουμε ότι δεν έχει γίνει καμία αλλαγή στα πεδία τιμών των μεταβλητών τους και συνεπώς παρουσιάζουν ήδη συνέπεια τόξου. Αυτό είναι ασφαλώς επιθυμητό. Ωστόσο το γεγονός ότι η ρουτίνα Revise δεν αποθηκεύει κανένα στοιχείο για τους υπολογισμούς που υλοποιεί για να βρει υποστήριξη για τις τιμές των μεταβλητών , αναγκάζει τον AC-3 να ξανακάνει αρκετές φορές τους ίδιους ελέγχους μεταξύ των τιμών των μεταβλητών για τις οποίες έχει ήδη κάνει έλεγχο ασυνέπειας.

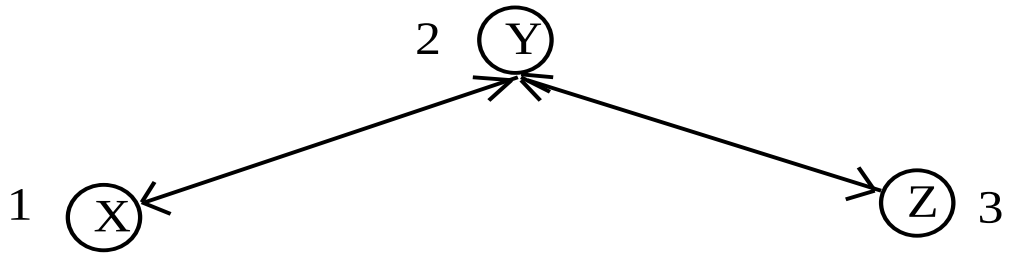
4.3.2 Απόδειξη Χωρικής Πολυπλοκότητας.

Ο αλγόριθμος AC-3 χρησιμοποιεί μόνο μια απλή δομή δεδομένων, μια ουρά. **Q.** Η ουρά **Q** κρατάει τα τόξα-περιορισμούς που πρέπει να ελεγχθούν για συνέπεια τόξου. Στην αρχή της εκτέλεση του αλγορίθμου η ουρά κρατάει όλα αυτά τα τόξα , τα οποία προσθαφαιρούνται κατά την χρήση εκτέλεση του αλγορίθμου, Με την προϋπόθεση ότι τα τόξα που υπάρχουν ήδη στην ουρά δεν ξαναπροστίθενται, είναι βέβαιο ότι τα τόξα που είναι αποθηκευμένα στην ουρά δεν θα ξεπεράσουν ποτέ σε αριθμό τον συνολικό αριθμό των περιορισμών του προβλήματος, **e**. Άρα το μέγεθος της ουράς είναι σε χώρο **O(e).**

4.4. Παράδειγμα Εκτέλεσης .

Έστω ότι έχουμε το παρακάτω CSP,

Μεταβλητές:	X, Y, Z
Πεδία ορισμού:	Dx = (4 5 6 7), Dy = (4 5 6 8 9)
	Dz = (3 5 6 7 9)
Περιορισμοί:	X= Y, Y = Z.



Σχήμα 4.1: Παράδειγμα εκτέλεσης αλγορίθμου AC-3

Αρχικά, Q = {(1, 2), (2, 1), (2, 3), (3, 2)} ή αλλιώς

Q = {(X, Y), (Y, X), (Y, Z), (Z, Y)}

*** Αφού βγει το τόξο (X, Y) από την ουρά ώστε να ελεγχθεί ,έχουμε

Dx = (4 5 6), Dy = (4 5 6 8 9), Dz καμία αλλαγή

Καμία προσθήκη στην ουρά Q.

*** Αφού βγει το τόξο (Y, X) από την ουρά ώστε να ελεγχθεί ,έχουμε

Dy = (4 5 6), Dx = (4 5 6), Dz no καμία αλλαγή

Το τόξο (Z, Y) πρέπει να εισαχθεί στην ουρά Q,

αλλά βρίσκεται ήδη εκεί, οπότε μην κάνεις τίποτα.

*** Αφού βγει το τόξο (Y, Z) από την ουρά ώστε να ελεγχθεί ,έχουμε

Dy = (5 6), Dz καμία αλλαγή, Dx = (4 5 6)

Το τόξο (X, Y) πρέπει να εισαχθεί στην ουρά Q για επανέλεγχο.

*** Αφού βγει το τόξο (Z,Y) από την ουρά ώστε να ελεγχθεί ,έχουμε

Dy καμία αλλαγή, Dz= (5 6) , Dx καμία αλλαγή

Το τόξο (X, Z) πρέπει να εισαχθεί στην ουρά Q για επανέλεγχο,

Αλλά δεν υπάρχει τόξο μεταξύ των X και Z.

*** Αφού βγει το τόξο (X,Y) από την ουρά ώστε να ελεγχθεί ,έχουμε

Dy καμία αλλαγή, Dz καμία αλλαγή , Dx = (5 6)

Το τόξο (Z, X) πρέπει να εισαχθεί στην ουρά Q για επανέλεγχο,

Αλλά δεν υπάρχει τόξο μεταξύ των Z και X.

*** Η ουρά Q τώρα είναι πλέον άδεια, οπότε η εκτέλεση του αλγόριθμου ολοκληρώνεται εδώ.

4.5. Παραλλαγή του AC-3.

Ο McGregor πρότεινε έναν διαφορετικό τρόπο διάδοσης περιορισμών (constraints propagation) για τον AC3, ο οποίος αργότερα ονομάστηκε μεταβλητο-στραφής (variable-oriented), αντίθετα με την αρχική τακτική του AC-3 η οποία είναι τοξο-στραφής (arc-oriented) [5] . Αντί να τοποθετηθούν στην ουρά Q τα τόξα τα οποία πρέπει να αναθεωρηθούν από την ρουτίνα REVISE κάθε φορά που γίνεται μια αλλαγή στο πεδίο ορισμού Di ,τοποθετούμε απλά την μεταβλητή Vi . Η ουρά Q περιέχει τις μεταβλητές οι οποίες πρέπει να εξεταστούν για ασυνέπειες. Όταν η μεταβλητή Vi βγαίνει από την ουρά , ο αλγόριθμος αναθεωρεί με την ρουτίνα REVISE όλα τα τόξα που περιέχουν την μεταβλητή Vi , τα οποία θα μπορούσαν να οδηγήσουν σε περαιτέρω διαγραφές τιμών από το πεδίο ορισμού των μεταβλητών που συμμετέχουν με την Vi .

Η υλοποίηση αυτής της έκδοσης του AC-3 είναι πιο απλή επειδή τα στοιχεία στην ουρά Q είναι απλά μεταβλητές . Ωστόσο αυτή η λιγότερο ακριβής πληροφορία παρουσιάζει

ένα μειονέκτημα. Ένα συγκεκριμένο τόξο μπορεί να χρειαστεί να αναθεωρηθεί μέσω της REVISE αρκετές φορές ενώ ο κλασικός AC-3 θα πραγματοποιούσε μόνο μία αναθεώρηση γι'αυτό το τοξο.

Κεφάλαιο 5

Ο Αλγόριθμος AC-4

Σε αυτό το κεφάλαιο της εργασίας παρουσιάζεται ο τρόπος λειτουργίας του αλγορίθμου AC-4. Στη συνέχεια παρατίθεται ο ψευδοκώδικας του συγκεκριμένου αλγορίθμου. Μετά τον ψευδοκώδικα ακολουθεί μια σύντομη ανάλυση της πολυπλοκότητας του αλγορίθμου. Έπειτα δίνεται ένα παράδειγμα εκτέλεσης του αλγορίθμου. Μετά το παράδειγμα εκτέλεσης ακολουθεί ο σχολιασμός των ελαττωμάτων του αλγορίθμου. Τέλος συγκρίνονται οι αλγόριθμοι fine-grained με τους αλγορίθμους coarse-grained.

5.1 Ο Τρόπος Λειτουργίας.

Καθώς ο AC3 δεν είναι βέλτιστος χρονικά ,οι Mohr και Henderson πρότειναν τον AC4 έτσι ώστε να βελτιωθεί η χρονική πολυπλοκότητα της οικογένειας αλγορίθμων AC [6, 7]. Η ιδέα του AC4, αντιθέτως με τον AC3, είναι η αποθήκευση όσο το δυνατόν περισσότερης πληροφορίας. Ο AC3 υλοποιεί την ελάχιστη δυνατή προσπάθεια κατά την κλήση της ρουτίνας Revise, επιβεβαιώνοντας ότι όλες οι εναπομείναντες τιμές της μεταβλητής Vi παρουσιάζουν υποστήριξη για τον περιορισμό Cij. Η ρουτίνα Revise δεν κρατάει τίποτα στην μνήμη. Το κόστος αυτής της επιλογής είναι ότι αν κληθεί η ρουτίνα Revise για το ίδιο τόξο ,τότε ο αλγόριθμος θα χρειαστεί να ξανακάνει αρκετή δουλειά την οποία την έχει κάνει ήδη. Ο AC4 ωστόσο αποθηκεύει την μέγιστη δυνατή ποσότητα πληροφορίας σε ένα προεπεξεργαστικό βήμα ,έτσι ώστε να αποφύγει να ελέγξει ξανά τις τιμές των μεταβλητών που έχουν ήδη ελεγχθεί και δεν παρουσιάζουν αλλαγές κατά την μετάδοση των διαγραφών.

Ο ψευδοκώδικας του AC-4 παρουσιάζεται στην επόμενη ενότητα . Ο αλγόριθμος υπολογίζει έναν μετρητή counter[Vi , ai , Vj] για κάθε τριπλέτα (Vi , ai , Vj) ,όπου ο

περιορισμός Cij που αποτελείται από τις μεταβλητές Vi και Vj ανήκει στο σύνολο των περιορισμών C και οι τιμές ai ανήκουν στο πεδίο ορισμού Di. Αυτός ο μετρητής τελικά μετράει πόσες τιμές της μεταβλητής Vj υποστηρίζουν την τιμή ai της μεταβλητής Vi για τον περιορισμό Cij . Ο AC4 φτιάχνει μια δομή του τύπου S[Vj , bj] η οποία περιέχει όλες τις τιμές ai της μεταβλητής Vi σε ζευγάρια μεταβλητών-τιμών (Vi , ai) που συμμετέχουν στον περιορισμό Cij και υποστηρίζουν την εκάστοτε τιμή bj της εκάστοτε μεταβλητής Vj.

Κατά την φάση αρχικοποίησης ,ο AC4 πραγματοποιεί όλους τους δυνατούς ελέγχους περιορισμών για όλους τους περιορισμούς .Κάθε φορά που ανακαλύπτει μια τιμή bj που ανήκει στο πεδίο ορισμού Dj και υποστηρίζει την τιμή ai της μεταβλητής Vi για τον περιορισμό Cij , ο μετρητής counter[Vi, ai, Vj] αυξάνεται κατά ένα , και το ζευγάρι (Vi, ai) προστίθεται στη λίστα S[Vj , bj]. Μόλις ολοκληρωθεί η εξέταση του περιορισμού Cij , αν ο μετρητής counter[Vi, ai, Vj] είναι ίσος με μηδέν, τότε αυτό σημαίνει ότι η τιμή ai της μεταβλητής Vi δεν υποστηρίζεται από καμία τιμή της μεταβλητής Vj . Άρα η τιμή ai θα πρέπει να αφαιρεθεί από το πεδίο ορισμού Di και να τοποθετηθεί στην ουρά Q με την μορφή (Vi,ai) ,έτσι ώστε να εξεταστεί ξανά για ασυνέπειες.

Μόλις ολοκληρωθεί η φάση αρχικοποίησης , ο αλγόριθμος εκτελεί την λούπα ελέγχου ασυνεπειών. Για κάθε ζευγάρι (Vj , aj) που βγαίνει από την ουρά Q , πρέπει να μειωθεί ο μετρητής counter[Vi, ai, Vj] για κάθε ζευγάρι (Vi , ai) που ανήκει στη λίστα S[Vj , bj] ,έτσι ώστε οι μετρητές να είναι πάντα ενημερωμένοι. Αν ο μετρητής counter[Vi, ai, Vj] φτάσει την τιμή μηδέν ,αυτό σημαίνει ότι η τιμή bj της μεταβλητής Vj ήταν η τελευταία που υποστήριζε την τιμή ai της μεταβλητής Vi για τον περιορισμό Cij . Τότε η τιμή ai αφαιρείται από το πεδίο ορισμού Di και τοποθετείται στην ουρά Q . Μόλις η ουρά Q αδειάσει , τότε γνωρίζουμε ότι όλες οι εναπομείναντες τιμές των πεδίων ορισμού των μεταβλητών έχουν μη-μηδενικό μετρητή για κάθε περιορισμό στον οποίο συμμετέχουν οι μεταβλητές και έτσι οι μεταβλητές είναι συνεπείς.

5.2 Ο Ψευδοκώδικας.

Η βασική ιδέα του AC-4 είναι η συσχέτιση κάθε τιμής ai του πεδίου ορισμού της μεταβλητής Vi με την υποστήριξη που παρέχει κάθε τιμή της μεταβλητής Vj . Δηλαδή τον αριθμό των τιμών στο πεδίο ορισμού Dj που υποστηρίζουν την τιμή ai. Η τιμή ai

απομακρύνεται αν χάσει την υποστήριξή της από οποιαδήποτε άλλη μεταβλητή. Ο αλγόριθμος χρησιμοποιεί μια ουρά Q ,η οποία κρατάει τις τιμές των μεταβλητών που δεν υποστηρίζονται .Έναν μετρητή counter[Vi , ai , Vj] ,ο οποίος μετράει την ποσότητα υποστήριξης για τις τιμές ai της μεταβλητής Vi από τις τιμές της μεταβλητής Vj. Και μια λίστα S[Vj , bj] , η οποία περιέχει ζευγάρια μεταβλητών-τιμών (Vi , ai) που υποστηρίζονται από τα ζευγάρια μεταβλητών-τιμών (Vj , bj). Στη συνέχεια ακολουθεί ο ψευδοκώδικας του αλγορίθμου AC-4 [2].

Algorithm AC-4

```
procedure AC-4

  Q <- INITIALIZE;
  while not Q empty                    //while there are still unsupported values

     select and delete any pair <Vj,bj> from Q;
     for each <Vi,ai> from S[Vj,bj]  do

        counter[(Vi,ai,Vj)] <- counter[(Vi,ai,Vj)]- 1;      //because bj is deleted

        if counter[(Vi,ai,Vj)]=0 & ai is still in Di then
            delete ai from Di;
            Q <- Q union {<Vi,ai>};
        endif

     endfor

  endwhile

end AC-4
```

```
Algorithm INITIALIZE

procedure INITIALIZE

 Q<- {};                              //Q holds unsupported values "Variable,Value"
 S[Vj,bj]<- {};                       //S[Vj,bj] holds supported values "Variable,Value"

 for each (Vi,Vj) in arcs(G) do                    //for each arc

    for each ai in Di do
       total <- 0;

       for each bj in Dj do
          if (ai,bj) is consistent according to the constraint (Vi,Vj) then
             total <- total+1;                     //count amount of support
             S[Vj,bj]<-S[Vj,bj] union {(Vi,ai)};   //add supported value in S as
          endif                                    //a pair in form of (Vi,ai)
       endfor;

       counter[(Vi,ai,Vj)] <- total;               //amount of support to Vi,ai from Vj
       if counter[(Vi,ai,Vj)]=0 then      //ai not supported by any value of variable Vj
          delete ai from Di;
          Q <- Q union {(Vi,ai)};                  //add unsupported value ai to Q
       endif;

    endfor;

 endfor;
 return Q;

end INITIALIZE
```

5.3 Ανάλυση του AC-4.

Λήμμα

Ο AC-4 επιτυγχάνει συνέπεια τόξου σε δυαδικά δίκτυα σε χρόνο O(e*d^2) και χώρο O(e*d^2), όπου e το σύνολο των περιορισμών και d ο αριθμός των τιμών του πεδίου ορισμού με τις περισσότερες τιμές. Η χρονική πολυπλοκότητα του αλγορίθμου είναι η βέλτιστη.

5.4. Παράδειγμα Εκτέλεσης.

Έστω ένα δίκτυο με τους περιορισμούς [c]

c1: x<=y

c2: y <> z, και τα πεδία ορισμού των μεταβλητών

D(x) = D(y) = {1, 2, 3, 4}

D(z) = {3}.

Κατά την φάση αρχικοποίησης ,ο AC4 αρχικά μετράει τον αριθμό υποστηρίξεων για κάθε τιμή σε κάθε περιορισμό και φτιάχνει τις λίστες των τιμών που υποστηρίζονται. Οπότε ,κατά την αρχικοποίηση ,ο AC4 πραγματοποιεί όλους τους πιθανούς ελέγχους περιορισμών για κάθε τιμή σε κάθε πεδίο ορισμού, στην συγκεκριμένη περίπτωση , 4 * 4 = 16 έλεγχοι περιορισμών για τον περιορισμό c1 and 4 * 1 = 4 για τον περιορισμό c2. Στο τέλος αυτής της φάσης ,οι δομές δεδομένων έχουν ως εξής:

counter[x, 1, y] = 4	counter[y, 1, x] = 1	counter[y, 1, z] = 1	counter[z, 3, y] = 3
counter[x, 2, y] = 3	counter[y, 2, x] = 2	counter[y, 2, z] = 1	
counter[x, 3,y] = 2	counter[y, 3, x] = 3	counter[y, 3, z] = 0	
counter[x, 4, y] = 1	counter[y, 4, x] = 4	counter[y, 4, z] = 1	

S[x, 1] = {(y, 1), (y, 2), (y, 3), (y, 4)}	S[y, 1] = {(x, 1), (z, 3)}	S[z, 3] = {(y, 1), (y, 2), (y,4)}
S[x, 2] = {(y, 2), (y, 3), (y, 4)}	S[y, 2] = {(x, 1), (x, 2), (z, 3)}	
S[x, 3] = {(y, 3), (y, 4)}	S[y, 3] = {(x, 1), (x, 2), (x, 3)}	
S[x, 4] = {(y,4)}	S[y, 4] = {(x, 1), (x, 2), (x, 3), (x, 4), (z, 3)}	

Ο μόνος μετρητής που ισούται με μηδέν είναι ο counter[**y, 3, z**] . Οπότε η τιμή 3 αφαιρείται από το πεδίο ορισμού της μεταβλητής y και ο AC4 μπαίνει στην κύρια λούπα με το (**y, 3**) στην ουρά **Q**. Όταν το (y, 3) εξέρχεται από την ουρά **Q**, η λίστα S[**y, 3**] διασχίζεται και οι μετρητές counter[x, 1,y], counter[x, 2, y],counter[x, 3, y] μειώνονται (επειδή τα (x, 1), (x, 2), (x, 3) βρίσκονται στην λίστα S[y, 3]). Κανένας από αυτούς τους μετρητές δεν ισούται με μηδέν και καμία τιμή δεν αφαιρείται. Παρατηρούμε ότι η διάδοση της διαγραφής της τιμής (y, 3) δεν

χρειάστηκε κανέναν έλεγχο περιορισμού .Απλά χρειάστηκε την διάσχιση των λιστών S[..] και τις αναβαθμίσεις των μετρητών .

5.5. Το Ελάττωμα του AC-4.

Αν και ο AC-4 είναι βέλτιστος όσον αφορά την χρονική πολυπλοκότητα , πάσχει από την ακριβή χωρική του πολυπλοκότητα. Επίσης η φάση αρχικοποίησής του είναι χρονικά ακριβή, έτσι ώστε σε αρκετά προβλήματα περιορισμών με μόλις μερικές εκατοντάδες περιορισμούς ,η χρήση του αλγορίθμου AC-4 θα ήταν λιγότερο αποτελεσματική από αυτήν του AC-3. Στην ουσία , μπορούμε να πούμε ανεπίσημα ότι αν και ο AC4 έχει βέλτιστη χρονική πολυπλοκότητα χειρότερης περίπτωσης **O**, σχεδόν πάντα ο αλγόριθμος εκτελείται στην χειρότερη περίπτωση **O**. Επιπλέον , ακόμα κι όταν η φάση αρχικοποίησης τελειώσει , ο AC4 διατηρεί τόσο ακριβή στοιχεία της επεξεργασίας , που τον αναγκάζουν να πραγματοποιεί μεγάλη προσπάθεια αναβαθμίζοντας τους μετρητές και διατρέχοντας της λίστες του.

5.6. Fine-Grained VS Coarse-Grained Algorithms.

Ο AC4 είναι ο πρώτος αλγόριθμος που ανήκει στην κατηγορία “αλγόριθμων λεπτής άλεσης” (fine-grained' algorithms) [8], άρα και ο ιδρυτής αυτής της κατηγορίας. Η κατηγορία ονομάστηκε έτσι επειδή οι αλγόριθμοι που ανήκουν σε αυτήν, επιτυγχάνουν συνέπεια τόξου πραγματοποιώντας ελέγχους σε επίπεδο τιμών .Αντιθέτως οι “αλγόριθμοι χοντρής άλεσης” (coarse-grained' algorithms), όπως για παράδειγμα ο AC3, πραγματοποιούν ελέγχους σε επίπεδο τόξων ή μεταβλητών. Αυτή η τακτική των ελέγχων σε επίπεδο τόξων ή μεταβλητών θεωρείται λιγότερο ακριβής και ίσως μερικές φορές προβεί σε μη αναγκαίες ενέργειες ,οι οποίες θα μπορούσαν να αποφευχθούν. Παρόλα ταύτα οι “αλγόριθμοι χοντρής άλεσης” παρουσιάζουν πλεονεκτήματα, όπως για παράδειγμα την έλλειψη επιπλέον δομών δεδομένων ,όπως οι λίστες S[**Vj , bj**] , οι οποίες έχουν σημαντικό κόστος αρχικοποίησης και συντήρησης.

Κεφάλαιο 6

Προγραμματιστική Υλοποίηση

Στο κεφάλαιο 6 παρουσιάζονται στοιχεία που αφορούν την προγραμματιστική υλοποίηση των αλγορίθμων. Αρχικά παρατίθεται η εκτέλεση των αλγορίθμων σε μη γραφικό και σε γραφικό περιβάλλον. Έπειτα ακολουθεί η περιγραφή των δομών δεδομένων που χρησιμοποιούν οι δύο αλγόριθμοι. Τέλος παρουσιάζονται τα στατιστικά στοιχεία εκτέλεσης των αλγορίθμων για κάποια δεδομένα προβλήματα ικανοποίησης περιορισμών.

6.1. Υλοποίηση Αλγορίθμων

Στην παρούσα εργασία οι αλγόριθμοι AC-3 και AC-4 υλοποιήθηκαν προγραμματιστικά με τη χρήση της γλώσσας προγραμματισμού Java και είναι ενσωματωμένοι στο πρόγραμμα CSP. Οι αλγόριθμοι αυτοί αποτελούν βασικό συστατικό του προγράμματος "CSP" που τους χρησιμοποιεί με σκοπό να πραγματοποιήσει ελέγχους συνέπειας τόξου.

6.1.1 Γενική Περιγραφή Λειτουργίας Προγράμματος

Αρχικά το πρόγραμμα λαμβάνει ως είσοδο κάποια προβλήματα ικανοποίησης περιορισμών, αποθηκευμένα σε αρχεία κειμένου .txt ειδικής μορφολογίας. Κάθε τέτοιο πρόβλημα αποτελείται από τέσσερα αρχεία κειμένου, τα con, dom ,rel ,var. Τα αρχεία con περιέχουν το σύνολο των περιορισμών του προβλήματος και τις μεταβλητές που συμμετέχουν σε κάθε περιορισμό. Τα αρχεία dom περιέχουν το σύνολο των πεδίων ορισμού, το σύνολο των τιμών για κάθε πεδίο ορισμού και τις επιτρεπτές τιμές αυτών. Τα αρχεία rel περιέχουν όλους τους δυνατούς συνδυασμούς τιμών για κάθε ζευγάρι μεταβλητών που συμμετέχουν σε

κάποιον περιορισμό του προβλήματος. Τα αρχεία var περιέχουν τον συνολικό αριθμό των μεταβλητών του προβλήματος, καθώς και από ποιο πεδίο ορισμού παίρνουν τιμές αυτές οι μεταβλητές. Αυτά τα αρχεία πρέπει να βρίσκονται ήδη μέσα στον φάκελο που περιέχει τον κώδικα του προγράμματος, πριν την έναρξη της εκτέλεσης. Στη συνέχεια το πρόγραμμα διαβάζει τα δεδομένα του προβλήματος από τα αρχεία αυτά και τα αποθηκεύει σε εσωτερικές δομές δεδομένων.

Έπειτα εκτελεί τους αλγορίθμους AC-3 και AC-4 για κάθε τέτοιο πρόβλημα , με σκοπό να πετύχει συνέπεια τόξου. Αφού ολοκληρωθεί η εκτέλεση παρουσιάζει τα αποτελέσματα της εκτέλεσης για κάθε πρόβλημα. Τα αποτελέσματα αυτά αφορούν τον χρόνο εκτέλεσης, την μνήμη που καταναλώθηκε ,τον αριθμό των τιμών που διαγράφθηκαν από τα πεδία ορισμού των μεταβλητών και κάποια άλλα στοιχεία σχετικά με κάθε αλγόριθμο ξεχωριστά. Τέλος αφού τερματιστεί η διαδικασία εκτέλεσης για όλα τα προβλήματα παρουσιάζονται στατιστικά στοιχεία που περιλαμβάνουν τους μέσους όρους των αποτελεσμάτων εκτέλεσης.

6.1.2. Οι Κλάσεις του Προγράμματος

Για να πετύχει τα παραπάνω, το πρόγραμμα "CSP" χρησιμοποιεί επτά κλάσεις ,καθεμία εκ των οποίων έχει ξεχωριστό και διακριτό ρόλο. Αυτές οι κλάσεις είναι οι csp, problemReaderClass, fileToArrayClass, ac3Class, ac4Class, statisticsClass και guiClass. Οι κλάσεις πρόκειται να αναλυθούν αμέσως ,διευκρινίζοντας τα καθήκοντα που πρέπει να επιτελέσει καθεμία από αυτές ώστε να επιτευχθεί η ορθή λειτουργία του προγράμματος "CSP".

Η κλάση csp είναι η κύρια κλάση ,η οποία ελέγχει την ροή ολόκληρου του προγράμματος. Χρησιμοποιείται για να αρχικοποιήσει τους πίνακες που κρατάνε στοιχεία σχετικά με τα πεδία ορισμού των μεταβλητών και τους συνδυασμούς τιμών που ικανοποιούν κάθε περιορισμό του προβλήματος. Στη συνέχεια συλλέγει όλα τα απαραίτητα δεδομένα για κάθε πρόβλημα, εκτελεί τους αλγορίθμους AC-3 και AC-4 για κάθε πρόβλημα παρουσιάζοντας τα αποτελέσματα της εκτέλεσης .Τέλος παρουσιάζει τις στατιστικές πληροφορίες που εξήχθησαν από την εκτέλεση των αλγορίθμων.

Η κλάση problemReaderClass χρησιμοποιείται για να διαβάσει τα αρχεία των προβλημάτων. Αρχικά διαβάζει και αποθηκεύει τα ονόματα των προβλημάτων σε μια λίστα ,η οποία στη συνέχεια ελέγχεται με σκοπό να εξαλειφθούν όσα προβλήματα δεν είναι ολοκληρωμένα, δηλαδή δεν περιέχουν και τα τέσσερα απαραίτητα αρχεία. Στη συνέχεια τα ονόματα των προβλημάτων που είναι ολοκληρωμένα και έγκυρα περνάνε στην κλάση csp ,αποθηκευμένα σε μια λίστα.

Η κλάση fileToArrayClass συλλέγει όλες τις απαραίτητες πληροφορίες από τα αρχεία των προβλημάτων και τα αποθηκεύει στις εσωτερικές δομές δεδομένων του προβλήματος ,έτσι ώστε να είναι πιο εύκολη και άμεση η επεξεργασία τους. Τέτοιου είδους πληροφορίες συμπεριλαμβάνουν τον αριθμό των προβλημάτων ,των περιορισμών των μεταβλητών , των πεδίων ορισμού, των τιμών. Επίσης τους περιορισμούς του προβλήματος λαμβάνοντας υπ'όψη από ποιες μεταβλητές αποτελείται ο καθένας καθώς επίσης και τους επιτρεπτούς συνδυασμούς τιμών για κάθε ζευγάρι μεταβλητών που συμμετέχουν σε κάθε περιορισμό.

Η κλάση ac3Class υλοποιεί τον αλγόριθμο AC-3 και την διαδικασία revise που χρησιμοποιείται από αυτόν τον αλγόριθμο. Χρησιμοποιεί μια βοηθητική μέθοδο με σκοπό να εισάγει στην ουρά ελέγχου όλα τα τόξα των μεταβλητών που πρέπει να ελεγχθούν για συνέπεια.

Η κλάση ac4Class υλοποιεί τον αλγόριθμο AC-4 και την διαδικασία initialize που χρησιμοποιείται από αυτόν τον αλγόριθμο. . Χρησιμοποιεί δύο βοηθητικές μεθόδους με σκοπό να αρχικοποιηθούν η λίστα και ο μετρητής που χρησιμοποιούνται από τον αλγόριθμο. Η λίστα κρατάει όλες τις τιμές που παρουσιάζουν υποστήριξη, ενώ ο μετρητής μετράει τον αριθμό των τιμών ai κάθε μεταβλητής Vi που υποστηρίζονται από τις τιμές κάποιας άλλης μεταβλητής Vj.

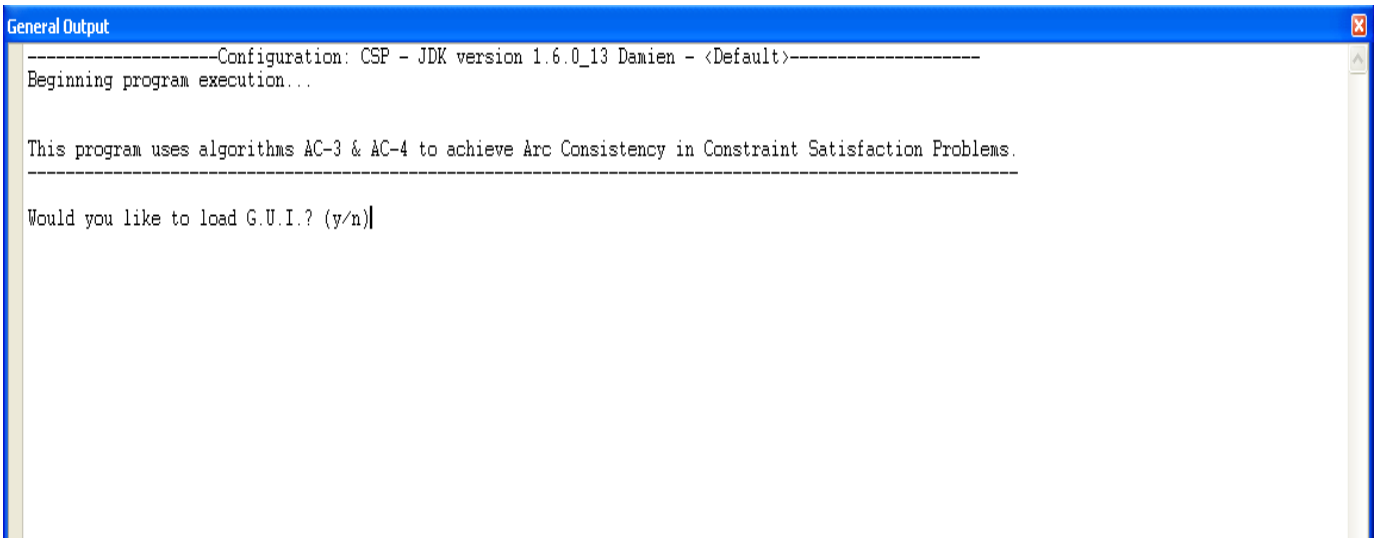
Η κλάση statisticsClass πραγματοποιεί διάφορες στατιστικές ενέργειες. Αυτές οι ενέργειες αφορούν τον χρόνο εκτέλεσης, την χρήση μνήμης ,τον αριθμό των τιμών που διαγράφηκαν και τους ελέγχους και μετρήσεις που έγιναν από τους αλγορίθμους AC-3 και AC-4 αντίστοιχα. Αρχικά θέτει αυτές τις τιμές στο μηδέν και στην συνέχεια τις αυξάνει καθώς πρέπει, και τελικά παρουσιάζει τα στατιστικά αποτελέσματα.

Η κλάση guiClass είναι η κλάση του γραφικού περιβάλλοντος χρήστη ,η οποία παρέχει την δυνατότητα χρήσης ενός γραφικού περιβάλλοντος ,αν ο χρήστης το επιθυμεί. Αυτή η κλάση κατασκευάζει την διεπαφή και έπειτα προσθέτει αισθητήρες σε κάθε στοιχείο

της διεπαφής, όπως τα κουμπιά κτλ, για να ανιχνεύσει τις κινήσεις του χρήση. Ο χρήστης μπορεί να προβεί σε κάποια συγκεκριμένη ενέργεια, όπως για παράδειγμα η ανίχνευση των προβλημάτων ,όσες φορές επιθυμεί .Επίσης μπορεί να διακόψει την εκτέλεση κάποιας διαδικασίας ,όποτε το επιθυμεί ,αντιθέτως με την εκτέλεση του προγράμματος σε μη γραφικό περιβάλλον. Ο χρήστης μπορεί να τρέξει το πρόγραμμα και χωρίς την χρήση γραφικού περιβάλλοντος.

6.2. Εκτέλεση Αλγορίθμων

Όπως λέχθηκε προηγουμένως, με το πρόγραμμα CSP ο χρήστης έχει την δυνατότητα εκτέλεσης των αλγορίθμων με την χρήση ενός γραφικού περιβάλλοντος. Αν ο χρήστης δεν επιθυμεί να χρησιμοποιήσει το περιβάλλον τότε μπορεί να τρέξει τους αλγορίθμους και χωρίς αυτό. Κατά την έναρξη της εκτέλεσης του προγράμματος ο χρήστης ερωτάται αν επιθυμεί να χρησιμοποιήσει το γραφικό περιβάλλον. Κατά την ολοκλήρωση της εκτέλεσης του προγράμματος ο χρήστης ερωτάται αν επιθυμεί να επαναχρησιμοποιήσει το πρόγραμμα. Αυτές οι δύο καταστάσεις φαίνονται στις παρακάτω εικόνες.

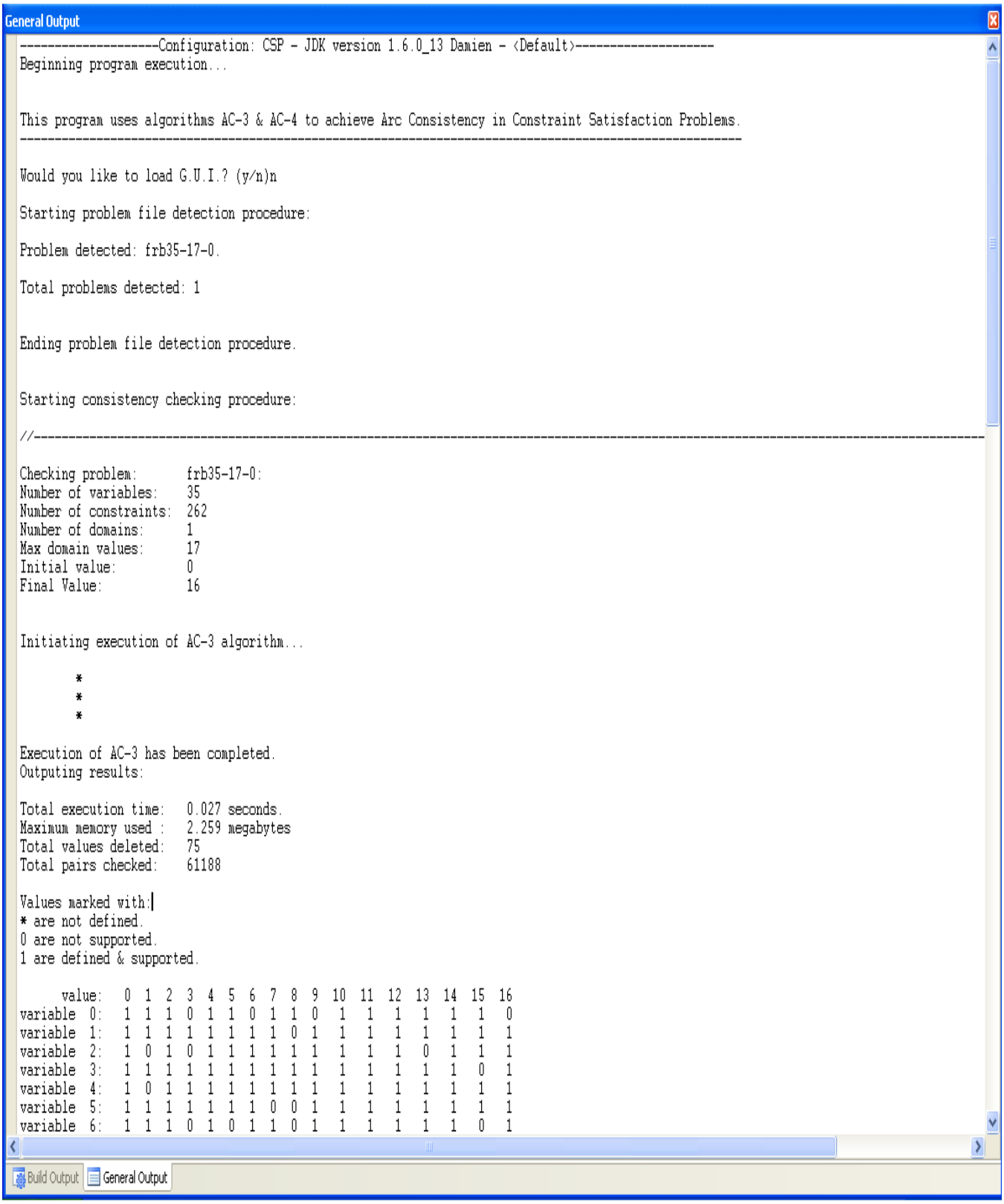


Σχήμα 6.2.α: Έναρξη εκτέλεσης προγράμματος

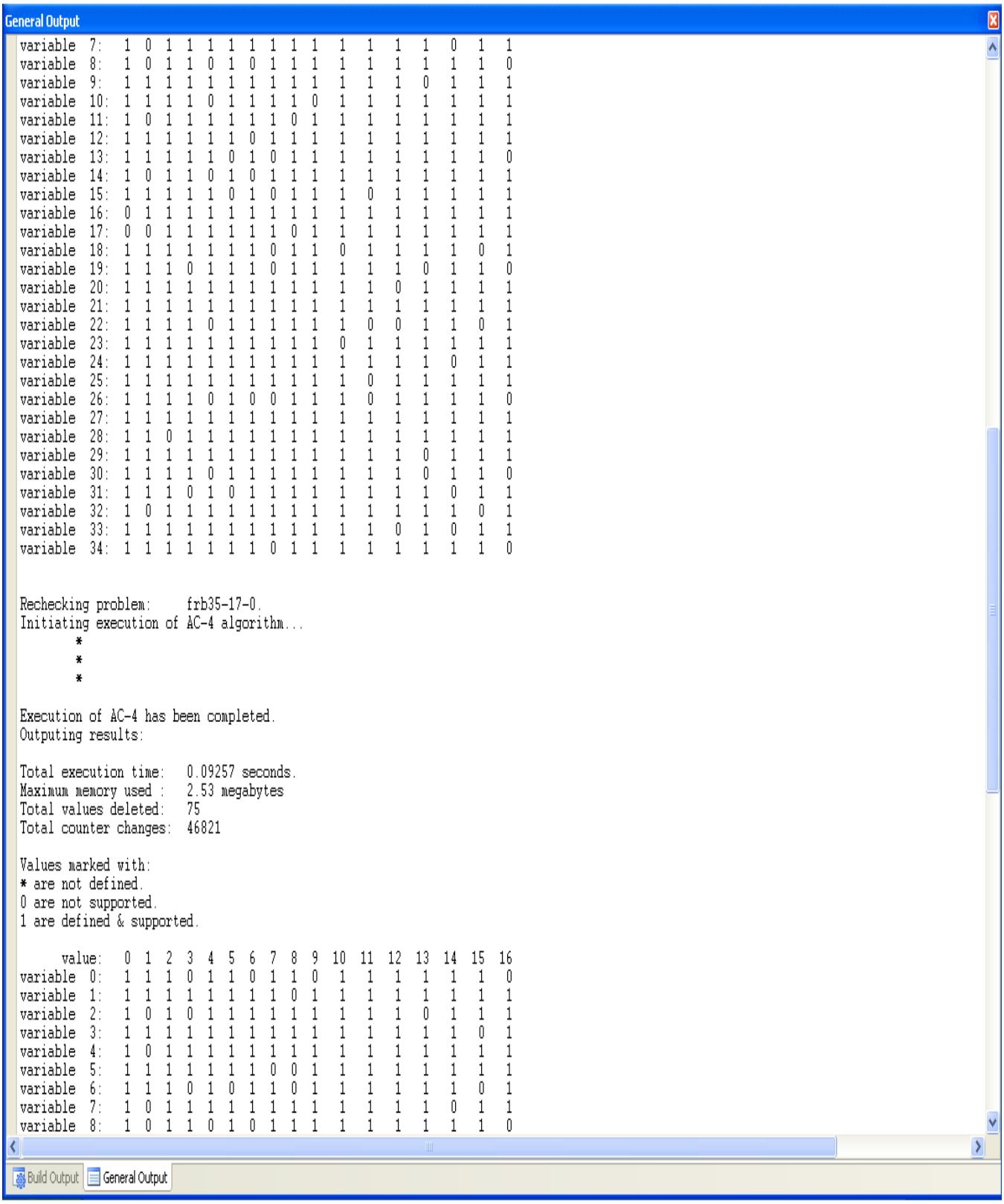
Σχήμα 6.2.β: Ολοκλήρωση εκτέλεσης προγράμματος

6.2.1. Εκτέλεση σε μη Γραφικό Περιβάλλον

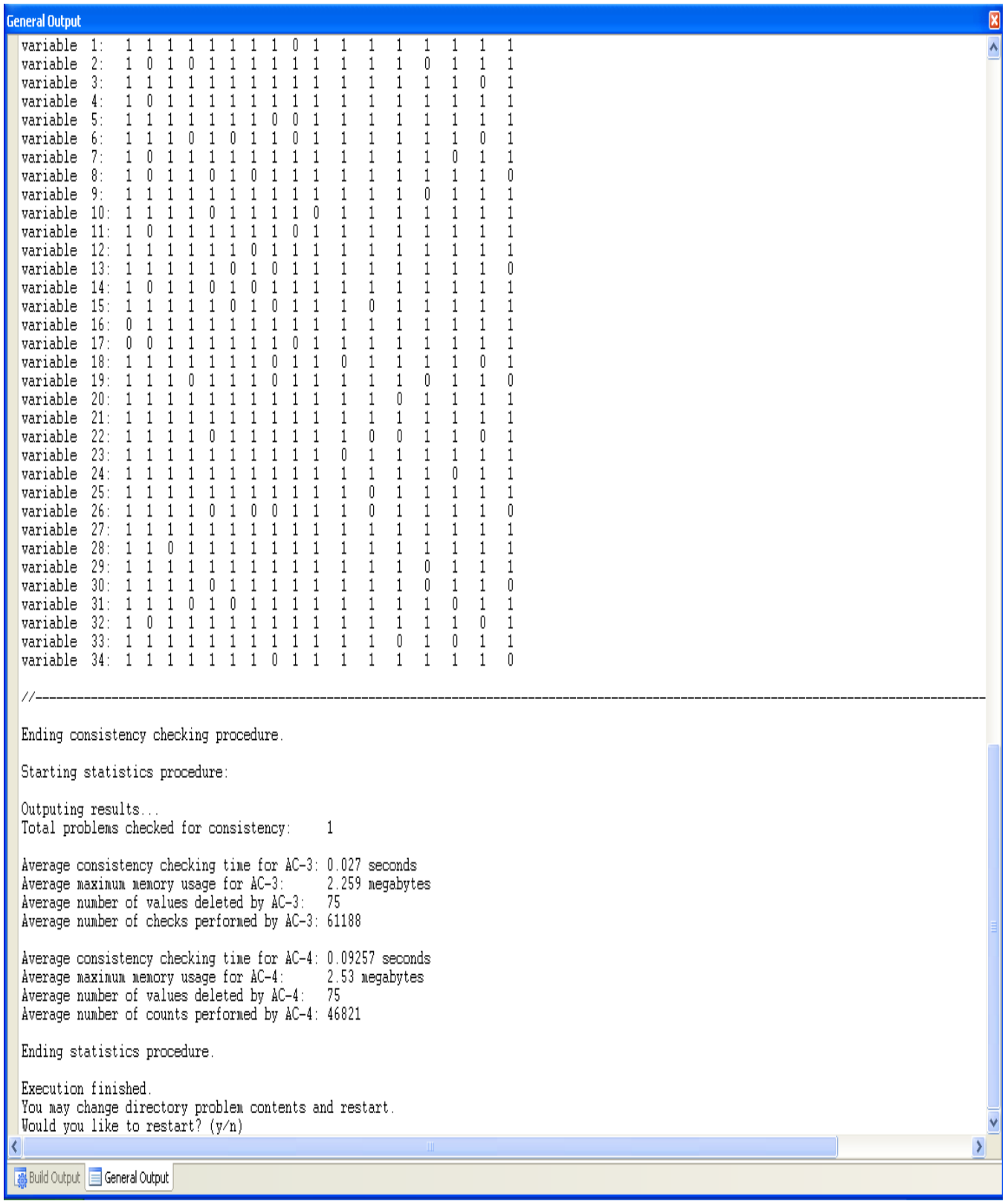
Αν ο χρήστης επιλέξει να τρέξει το πρόγραμμα σε μη γραφικό περιβάλλον τότε το πρόγραμμα θα ακολουθήσει μια σειριακή εκτέλεση. Ο χρήστης δεν θα έχει την δυνατότητα αλληλεπίδρασης με το πρόγραμμα μέχρι να τελειώσει αυτή η εκτέλεση. Αρχικά λοιπόν θα εντοπιστούν τα έγκυρα αρχεία με τα προβλήματα εισόδου. Έπειτα τα δεδομένα των αρχείων θα αποθηκευτούν στις εσωτερικές δομές δεδομένων του προγράμματος για επεξεργασία. Θα εκτελεστούν οι αλγόριθμοι AC-3 & AC-4 για κάθε πρόβλημα ξεχωριστά και θα παρουσιαστούν τα αποτελέσματα της εκτέλεσης και τα στατιστικά στοιχεία. Ένα οπτικό παράδειγμα αυτής της διαδικασίας ακολουθεί στις παρακάτω εικόνες.



Σχήμα 6.2.1.α: Παράδειγμα εκτέλεσης σε μη γραφικό περιβάλλον. Εντοπισμός αρχείων και εκτέλεση αλγορίθμου AC-3.



Σχήμα 6.2.1.β: Παράδειγμα εκτέλεσης σε μη γραφικό περιβάλλον. Εκτέλεση αλγορίθμων AC-3 ,AC-4



Σχήμα 6.2.1.γ: Παράδειγμα εκτέλεσης σε μη γραφικό περιβάλλον. Ολοκλήρωση εκτέλεσης και προβολή στατιστικών στοιχείων.

6.2.2 Εκτέλεση σε Γραφικό Περιβάλλον

Αν ο χρήστης επιλέξει να τρέξει το πρόγραμμα σε γραφικό περιβάλλον τότε το πρόγραμμα θα φορτώσει πρώτα την γραφική διεπαφή χρήστη. Ο χρήστης θα έχει την δυνατότητα αλληλεπίδρασης με το πρόγραμμα μέχρι να κλείσει το γραφικό περιβάλλον.

Η διεπαφή είναι σχεδιασμένη έτσι ώστε στο πάνω μέρος της να περιλαμβάνει ένα status bar ,το οποίο περιγράφει την τρέχουσα ενέργεια που εκτελείται. Όταν δεν εκτελείται καμία ενέργεια η περιγραφή της μπάρας είναι η "Waiting for your next action…".Αυτή η περιγραφή αλλάζει κάθε φορά που ολοκληρώνεται κάποια ενέργεια είτε εκτελείται κάποια καινούρια ενέργεια.

Στο αριστερό μέρος της διεπαφής περιλαμβάνονται πέντε κουμπιά (buttons), τα detect, check, statistics, background color, font color, τα οποία μόλις πατηθούν εκτελούν ένα ξεχωριστό κομμάτι κώδικα το καθένα προβαίνοντας σε συγκεκριμένες ενέργειες. Κάποια από αυτά τα κουμπιά είναι στην αρχή "κλειδωμένα", έτσι ώστε ο χρήστης να μην μπορεί να τα πατήσει μέχρι να επιτευχθεί κάποια ορισμένη νοητή κατάσταση, οπότε και κάποια κουμπιά θα "ξεκλειδωθούν" ενώ κάποια άλλα πιθανός να "κλειδωθούν". Για παράδειγμα σε καμία περίπτωση ο χρήστης δεν μπορεί να πατήσει το κουμπί check πριν πατήσει το κουμπί detect, ούτε το κουμπί statistics πριν το check . Με αυτόν τον τρόπο ο χρήστης κατευθύνεται έμμεσα σε κάποιο προκαθορισμένο μονοπάτι.

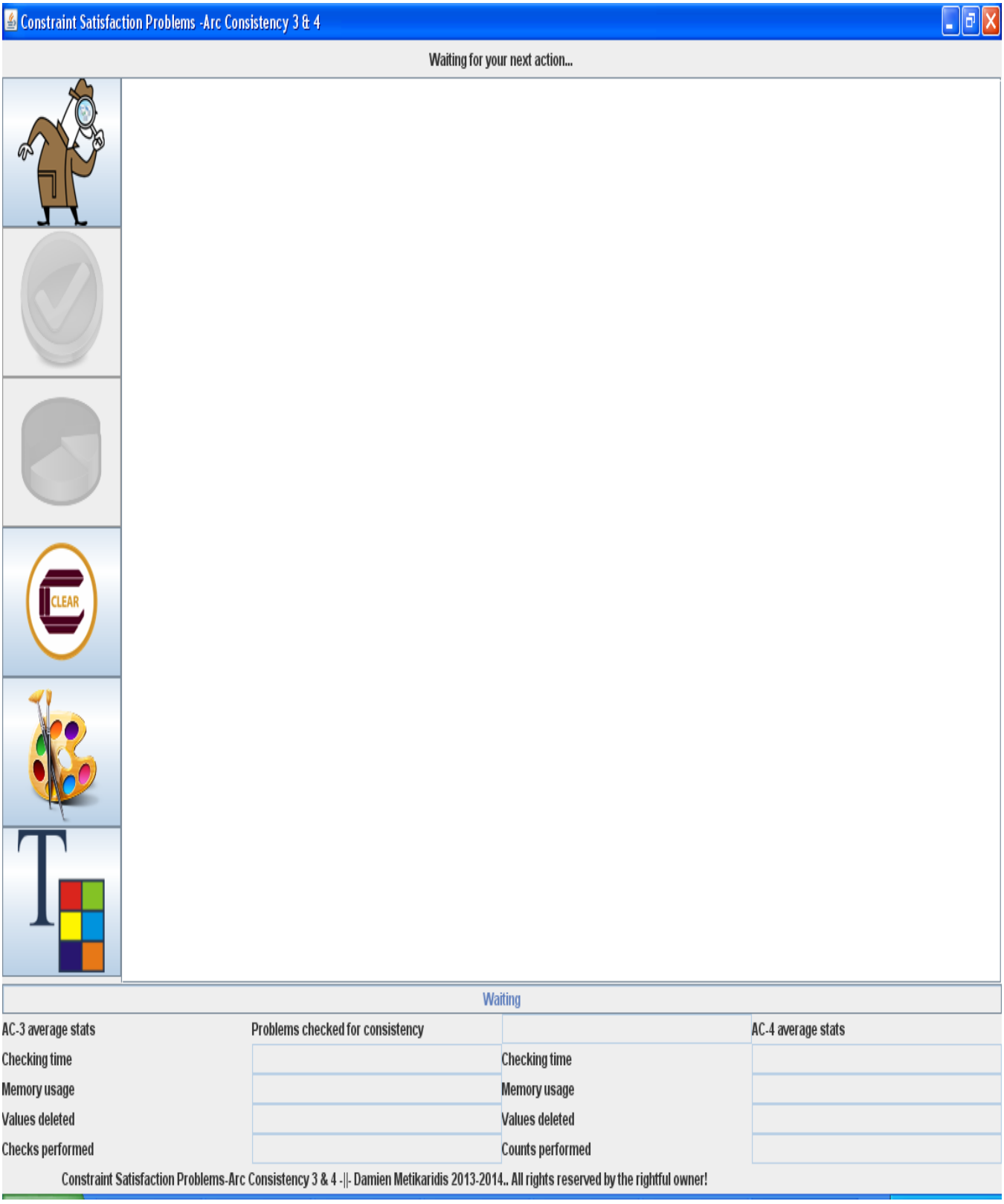
Πατώντας το κουμπί detect το πρόγραμμα εντοπίζει τα έγκυρα αρχεία εισόδου που έχουν αποθηκευμένα τα δεδομένα των προβλημάτων. Έπειτα πατώντας το κουμπί check ,τα δεδομένα των αρχείων αποθηκεύονται στις εσωτερικές δομές δεδομένων του προγράμματος για επεξεργασία και εκτελούνται οι αλγόριθμοι AC-3 & AC-4 για κάθε πρόβλημα ξεχωριστά. Αν ο χρήστης ξαναπατήσει οποιοδήποτε από αυτά τα κουμπιά ,πριν προλάβει να ολοκληρωθεί η ενέργεια που επιτελείται ,τότε η ενέργεια θα ακυρωθεί. Τέλος πατώντας το κουμπί statistics παρουσιάζονται τα στατιστικά στοιχεία της εκτέλεσης των αλγορίθμων. Επίσης ο χρήστης έχει την δυνατότητα να αλλάξει το χρώμα του background και της γραμματοσειράς οποιαδήποτε στιγμή επιθυμεί ,πατώντας τα κουμπιά background color και font color αντίστοιχα.

Στο κέντρο της διεπαφής εμφανίζονται όλα τα αποτελέσματα των ενεργειών που εκτελούνται αν πατηθεί κάποιο από τα κουμπιά που βρίσκονται στα αριστερά. Αυτά τα

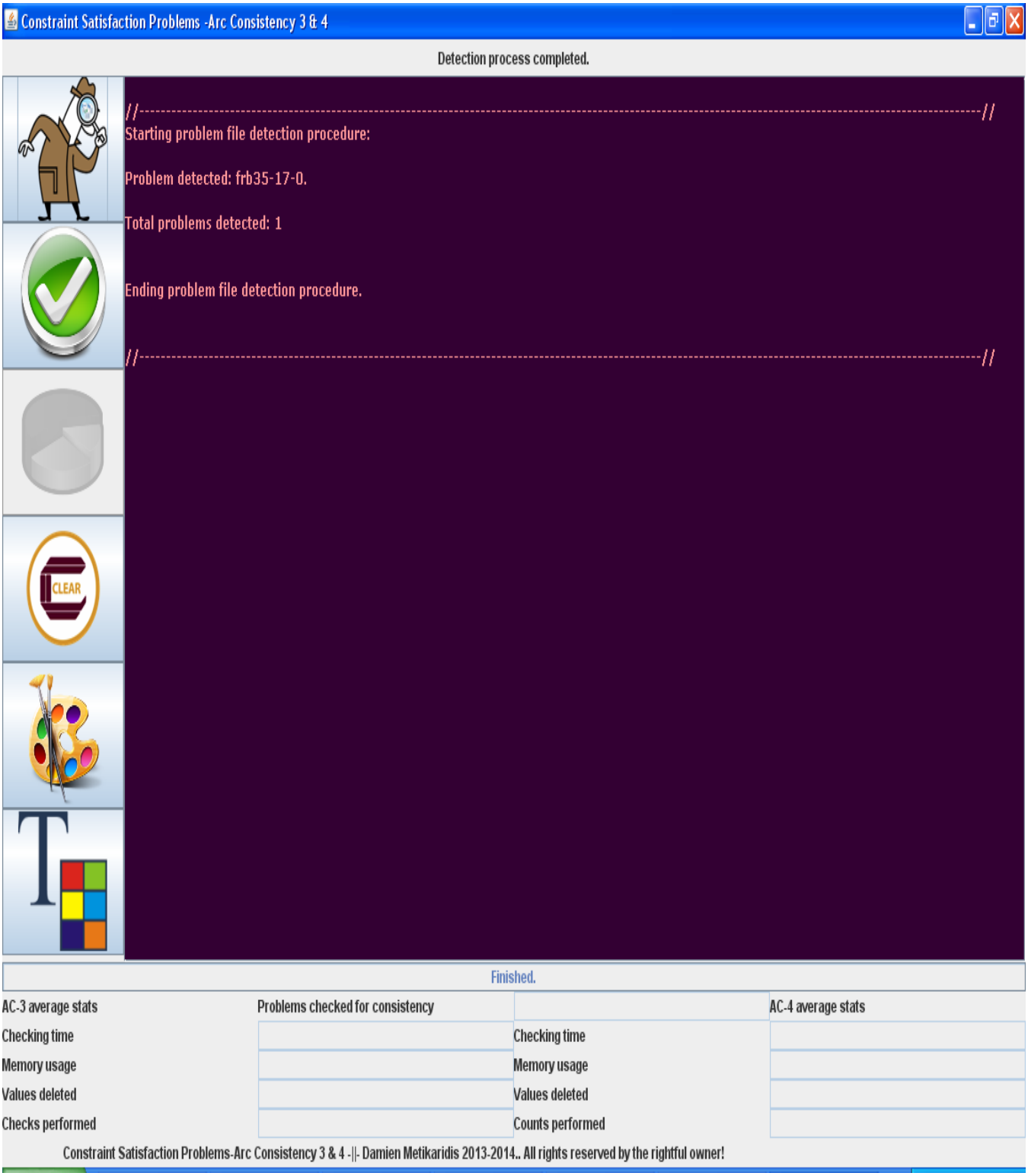
αποτελέσματα περιλαμβάνουν αποτελέσματα αναζήτησης και αποτελέσματα ελέγχου. Χρησιμοποιώντας το scroll bar στα αριστερά ,ο χρήστης μπορεί να δει όλα αυτά τα αποτελέσματα.

Στο κάτω μέρος της διεπαφής υπάρχει μια μπάρα φόρτωσης η οποία δείχνει το επί τοις εκατό ποσοστό ολοκλήρωσης κάθε διαδικασίας που εκτελείται. Έπειτα ακολουθούν τα στατιστικά στοιχεία των αλγόριθμων που θα εξαχθούν μετά την ολοκλήρωση του ελέγχου συνέπειας για κάθε πρόβλημα. Αυτά τα στοιχεία αναφέρονται στον μέσο όρο του χρόνου εκτέλεσης, της χρήσης μνήμης, των τιμών που διαγράφηκαν από τα πεδία ορισμού και των ελέγχων-μετρήσεων που πραγματοποίησαν οι AC-3 και AC-4 αντίστοιχα. Τέλος γίνεται αναφορά στα πνευματικά δικαιώματα της εφαρμογής.

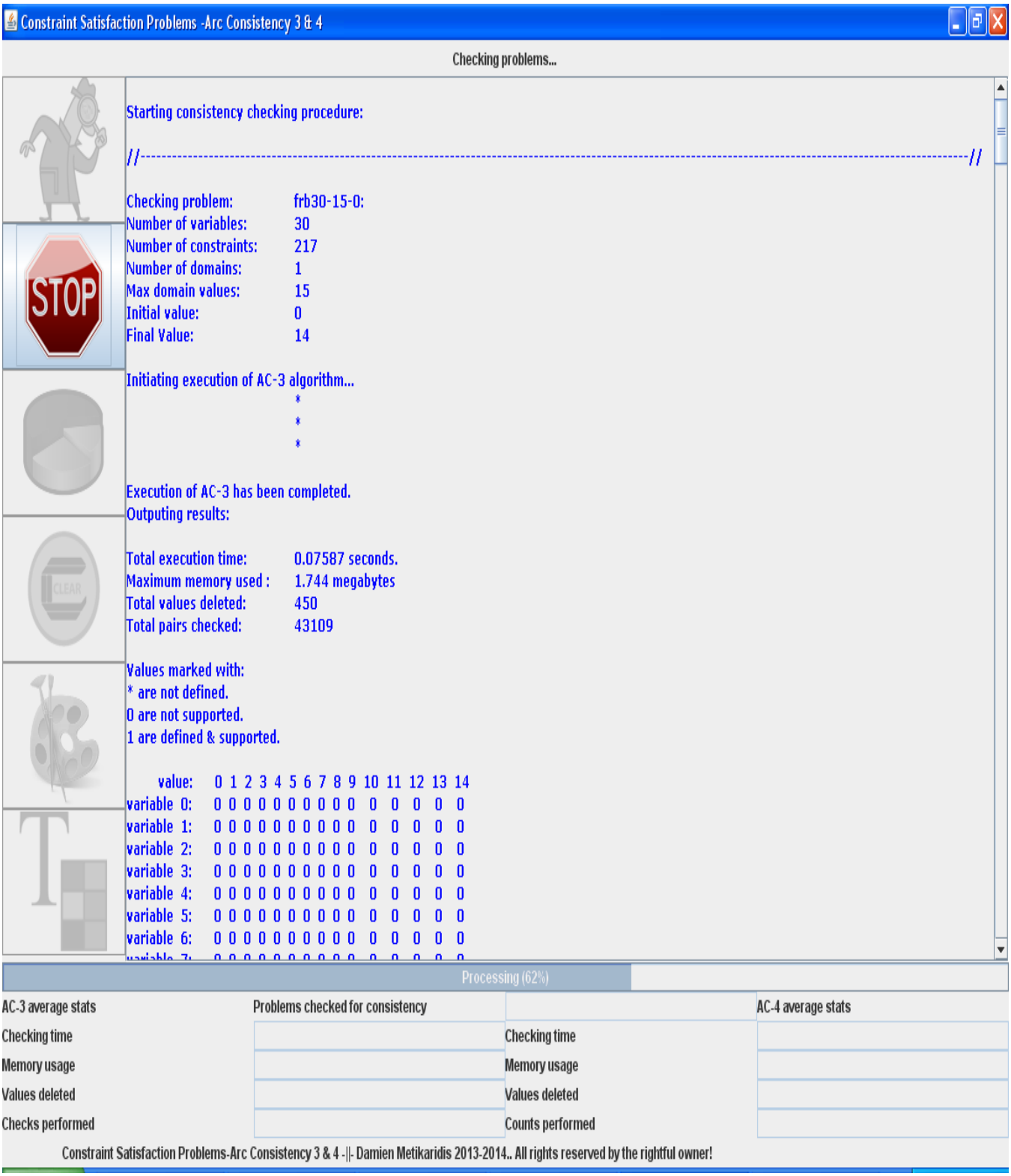
Ένα οπτικό παράδειγμα αυτής της διαδικασίας ακολουθεί στις παρακάτω εικόνες.



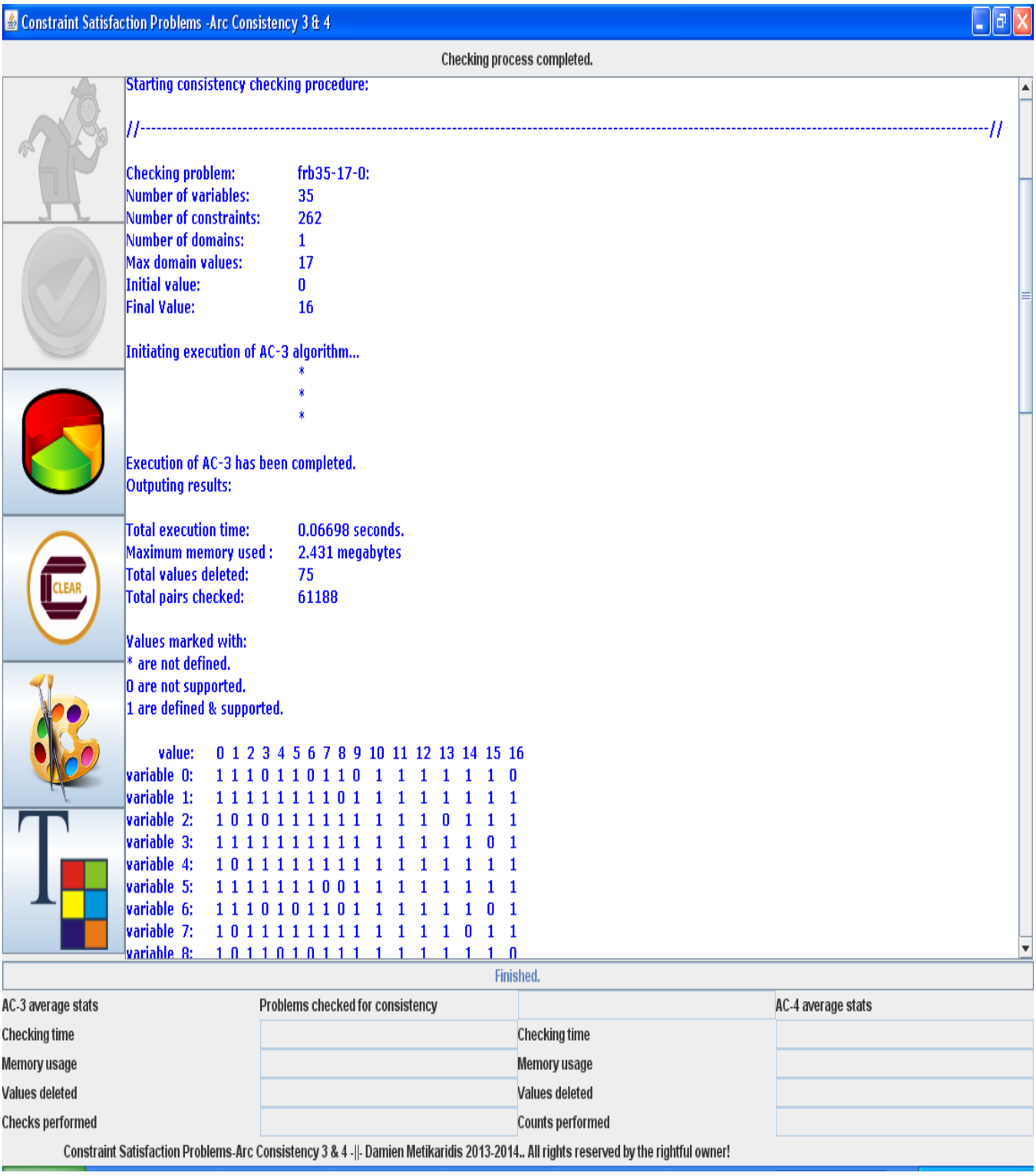
Σχήμα 6.2.2.α: Παράδειγμα εκτέλεσης σε γραφικό περιβάλλον. Αρχική κατάσταση γραφικού περιβάλλοντος.



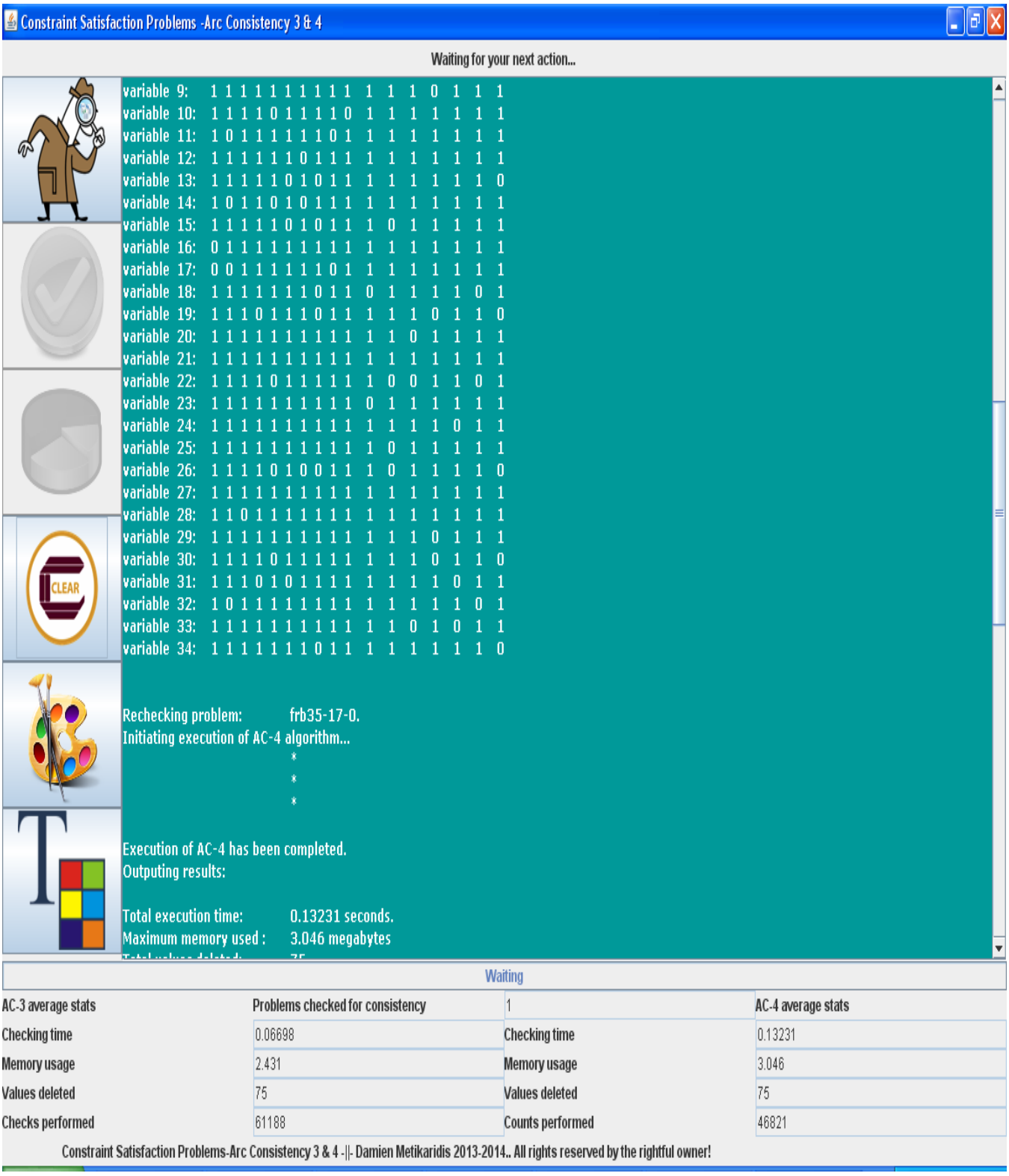
Σχήμα 6.2.2.β: Παράδειγμα εκτέλεσης σε γραφικό περιβάλλον. Ολοκλήρωση διαδικασίας εντοπισμού αρχείων ,με αλλαγμένο background και font color



Σχήμα 6.2.2.γ: Παράδειγμα εκτέλεσης σε γραφικό περιβάλλον. Ενδιάμεσο σημείο διαδικασίας ελέγχου συνεπειών ,με αλλαγμένο background και font color.



Σχήμα 6.2.2.δ: Παράδειγμα εκτέλεσης σε γραφικό περιβάλλον. Ολοκλήρωση διαδικασίας ελέγχου συνεπειών ,με αλλαγμένο background και font color.



Σχήμα 6.2.2.ε: Παράδειγμα εκτέλεσης σε γραφικό περιβάλλον .Προβολή στατιστικών στοιχείων στο κάτω μέρος της εφαρμογής ,με αλλαγμένο background και font color.

6.3 Οι Δομές Δεδομένων των Αλγορίθμων

Πέραν των όσων λέχθηκαν θεωρητικά στα προηγούμενα κεφάλαια για τις δομές δεδομένων που χρησιμοποιούν οι αλγόριθμοι AC-3 και AC-4, κρίνεται σκόπιμο να αναφερθούν σε αυτό το σημείο οι δομές δεδομένων που χρησιμοποιήθηκαν για να επιτευχθεί η συγκεκριμένη προγραμματιστική υλοποίηση των αλγορίθμων αυτών. Αυτό θα καθιστήσει πιο εύκολη την ανάγνωση του κώδικα, αλλά και την κατανόησή του. Στην συνέχεια θα γίνει αναφορά αυτών των δομών για κάθε αλγόριθμο.

6.3.1 Δομές Δεδομένων του AC-3

Κατά την προγραμματιστική υλοποίηση αυτής της εργασίας ο αλγόριθμος AC-3 χρησιμοποιεί τις δομές δεδομένων constraints, relFileData, variableDomain και queue.

Η δομή constraints είναι ένας δυσδιάστατος πίνακας ,ο οποίος κρατάει τις μεταβλητές που συμμετέχουν σε κάθε δυαδικό περιορισμό. Ο αλγόριθμος χρησιμοποιεί αυτόν τον πίνακα για να ελέγξει αν υφίσταται κάποιος περιορισμός μεταξύ δύο συγκεκριμένων μεταβλητών.

Η δομή relFileData, είναι ένας τετραδιάστατος πίνακας ,ο οποίος κρατάει τα ζευγάρια τιμών που παίρνουν τα ζευγάρια μεταβλητών ώστε να ικανοποιείται κάθε περιορισμός. Ο αλγόριθμος χρησιμοποιεί αυτόν τον πίνακα για να ελέγξει αν ισχύει η συνέπεια τόξου μεταξύ δύο μεταβλητών σε κάθε περιορισμό.

Η δομή variableDomain είναι ένας δυσδιάστατος πίνακας ,ο οποίος κρατάει τις τιμές που ανήκουν στο πεδίο ορισμού κάθε μεταβλητής. Είναι η πιο σημαντική δομή στο πρόγραμμα ,καθώς ο στόχος των αλγορίθμων είναι να θέσουν όσο περισσότερες τιμές αυτού του πίνακα μπορούν ίσες με μηδέν, δηλαδή να αφαιρέσουν αυτές τις τιμές από το πεδίο ορισμού των μεταβλητών ώστε να επιτευχθεί συνέπεια τόξου.

Η δομή queue είναι μια ουρά η οποία κρατάει τα τόξα του δικτύου, ή αλλιώς τους περιορισμούς του προβλήματος. Ο αλγόριθμος χρησιμοποιεί αυτήν την ουρά για να αποθηκεύσει όλα τα τόξα τα οποία χρειάζεται να ελεγχθούν για συνέπεια τόξου.

Για να γίνουν κατανοητές αυτές οι δομές ας υποθέσουμε ότι έχουμε για παράδειγμα τις μεταβλητές x, y ,z και τον περιορισμό x>y με Dx={1,2,3,4} ,Dy={2,3} και Dz= {2,4} .Αν

αναπαραστήσουμε την μεταβλητή x με τον αριθμό 1 ,την μεταβλητή y με τον αριθμό 2 και την μεταβλητή z με τον αριθμό 3,τότε θα ισχύουν τα επακόλουθα:

Τα στοιχεία constraints[1][2] και constraints[2][1] κρατάνε την τιμή 1 ,που σημαίνει ότι υφίσταται κάποιος περιορισμός μεταξύ των μεταβλητών 1 και 2. Ενώ για παράδειγμα το στοιχείο constraints[1][3] κρατάει την τιμή 0 ,επειδή δεν υφίσταται κανένας περιορισμός μεταξύ των μεταβλητών 1 και 3.

Επίσης το στοιχείο relFileData[1][2][3][2] κρατάει την τιμή 1, το οποίο σημαίνει ότι αν η μεταβλητή 1 πάρει την τιμή 3 και η μεταβλητή 2 πάρει την τιμή 2 ο περιορισμός μεταξύ αυτών των δύο μεταβλητών ικανοποιείται. Αντίθετα το στοιχείο relFileData[1][2][3][3] κρατάει την τιμή 0 γιατί για τις τιμές 3 και 3 ο περιορισμός μεταξύ των μεταβλητών 1 και 2 δεν ικανοποιείται.

Το στοιχείο variableDomain[1][1] κρατάει την τιμή 1 γιατί η μεταβλητή 1 μπορεί να πάρει την τιμή 1, σύμφωνα με το πεδίο ορισμού της. Ωστόσο το στοιχείο variableDomain[2][1] κρατάει την τιμή 0 γιατί η μεταβλητή 2 δεν μπορεί να πάρει την τιμή 1, σύμφωνα με το πεδίο ορισμού της.

Η ουρά Q κρατάει το τόξο 1,2 το οποίο σημαίνει ότι υφίσταται ένας περιορισμός μεταξύ των μεταβλητών 1 και 2, ο οποίος πρέπει να ελεγχθεί για συνέπεια.

6.3.2 Δομές Δεδομένων του AC-4

Κατά την προγραμματιστική υλοποίηση αυτής της εργασίας ο αλγόριθμος AC-4 χρησιμοποιεί τις δομές δεδομένων, constraints, relFileData, variableDomain ,queue , sList και counter.

Η δομή constraints είναι ένας δυσδιάστατος πίνακας ,ο οποίος κρατάει τις μεταβλητές που συμμετέχουν σε κάθε δυαδικό περιορισμό. Ο αλγόριθμος χρησιμοποιεί αυτόν τον πίνακα για να ελέγξει αν υφίσταται κάποιος περιορισμός μεταξύ δύο συγκεκριμένων μεταβλητών.

Η δομή relFileData, είναι ένας τετραδιάστατος πίνακας ,ο οποίος κρατάει τα ζευγάρια τιμών που παίρνουν τα ζευγάρια μεταβλητών ώστε να ικανοποιείται κάθε περιορισμός. Ο αλγόριθμος χρησιμοποιεί αυτόν τον πίνακα για να ελέγξει αν υπάρχει συνέπεια τόξου μεταξύ δύο μεταβλητών σε κάθε περιορισμό.

Η δομή variableDomain είναι ένας δυσδιάστατος πίνακας ,ο οποίος κρατάει τις τιμές που ανήκουν στο πεδίο ορισμού κάθε μεταβλητής. Είναι η πιο σημαντική δομή στο πρόγραμμα ,καθώς ο στόχος των αλγορίθμων είναι να θέσουν όσο περισσότερες τιμές αυτού του πίνακα μπορούν ίσες με μηδέν, δηλαδή να αφαιρέσουν αυτές τις τιμές από το πεδίο ορισμού των μεταβλητών ώστε να επιτευχθεί συνέπεια τόξου.

Η δομή queue είναι μια ουρά που κρατάει τα ζευγάρια "τιμή ,μεταβλητή" που δεν υποστηρίζονται από κάποια άλλη ,οποιαδήποτε μεταβλητή. Ο αλγόριθμος χρησιμοποιεί αυτήν την ουρά μέχρι να αδειάσουν όλα τα περιεχόμενά της, οπότε και εξασφαλίζεται η συνέπεια τόξου.

Η δομή sList είναι μια λίστα που κρατάει τις μεταβλητές Vi και τις τιμές τους ai σε ζεύγη της μορφής <Vi,ai> που υποστηρίζουν την τιμή bj της μεταβλητής Vj. Για παράδειγμα ο δείκτης της λίστα με αριθμό Vj*numberOfValues+bj ,κρατάει όλα τα ζευγάρια της μορφής Vi,ai που υποστηρίζουν την τιμή bj της μεταβλητής Vj

Η δομή counter είναι ένας δυαδικός πίνακας που λειτουργεί σαν μετρητής. Αυτός ο μετρητής τελικά μετράει πόσες τιμές της μεταβλητής Vj υποστηρίζουν την τιμή ai της μεταβλητής Vi ,αποθηκεύοντας αυτόν τον αριθμό στο στοιχείο counter [Vi*numberOfValues+ai][Vj], όπου numberOfValues είναι ο αριθμός των τιμών του πεδίου ορισμού με τις περισσότερες τιμές.

Αν επικαλεστούμε πάλι το παράδειγμα που είδαμε στην προηγούμενη ενότητα ,τότε το πεδίο ορισμού με τις περισσότερες τιμές είναι το Dx=D1 το οποίο έχει numberOfValues=4 τιμές .Αν θέλουμε να δούμε πόσες τιμές της μεταβλητής x (1) υποστηρίζουν την τιμή 3 της μεταβλητής y (2) ,αρκεί να καλέσουμε το στοιχείο counter[2*4+3][1], το οποίο πρέπει να κρατάει την τιμή 4, γιατί μόνο η τιμή 4 του πεδίου Dx=D1 ικανοποιεί τον περιορισμό $\chi>y$ όταν το y ισούται με 3.

6.4 Στατιστικά Στοιχεία Εκτέλεσης

Σε αυτήν την ενότητα θα παρουσιαστούν τα στατιστικά στοιχεία από την εκτέλεση των αλγορίθμων με είσοδο κάποια δεδομένα προβλήματα. Η εκτέλεση έγινε σε μη γραφικό περιβάλλον.

Τα προβλήματα που χρησιμοποιούνται χωρίζονται στις κατηγορίες frb, geo, qcp, qwh, Langford, composed, driver, blackhole και hanoi. Αυτά τα προβλήματα παρουσιάζουν διαφορετικό αριθμό περιορισμών , αριθμό μεταβλητών, αριθμό πεδίων ορισμού και μέγεθος μέγιστου πεδίου ορισμού έτσι ώστε να επιτευχθεί μια πολύπλευρη μελέτη της συμπεριφοράς των αλγορίθμων υπό διαφορετικές και ξεχωριστές συνθήκες κάθε φορά.

Το υπολογιστικό σύστημα στο οποίο έγινε η εκτέλεση των αλγορίθμων χρησιμοποιεί λειτουργικό σύστημα Windows xp, με επεξεργαστή CPU quad core συχνότητας 2.4Ghz , Cache size 8MB και RAM ίση με 3GB.

6.4.1 Τα Προβλήματα FRB.

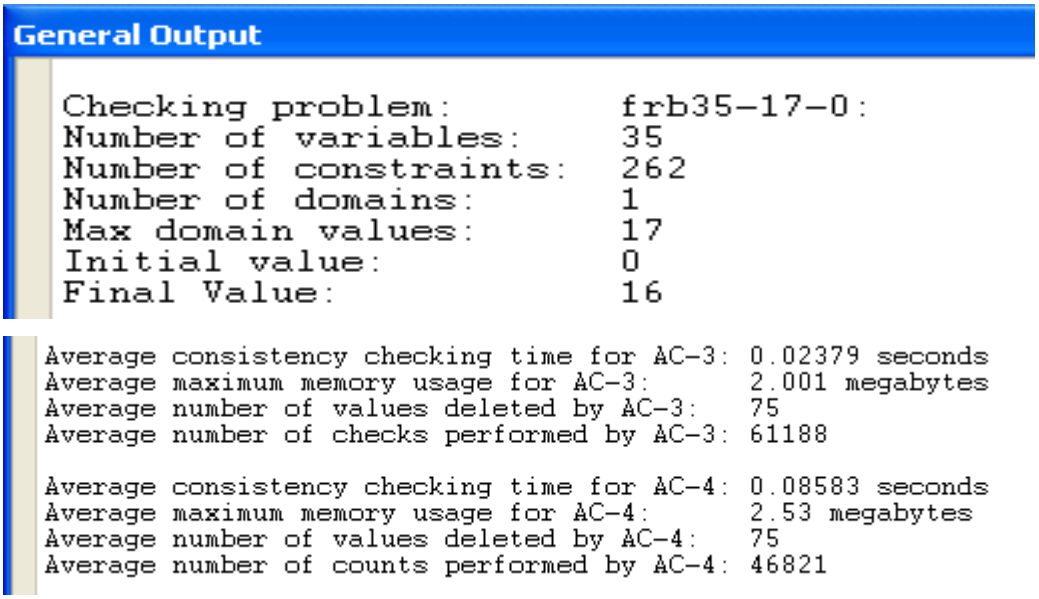
Όπως φαίνεται από τα παρακάτω στατιστικά στοιχεία για τα προβλήματα που έχουν μερικές δεκάδες μεταβλητές, και μερικές εκατοντάδες περιορισμούς ,ο αλγόριθμος AC-3 πετυχαίνει καλύτερο χρόνο εκτέλεσης απ'ότι ο AC-4. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

Όπου "cons" είναι το πλήθος των περιορισμών του προβλήματος, "vars" είναι το πλήθος των μεταβλητών του προβλήματος, "doms" είναι το πλήθος των πεδίων ορισμού του προβλήματος, "max dom size" είναι το μέγεθος του μεγαλύτερου πεδίου ορισμού του προβλήματος, "dels" είναι το πλήθος των διαγραφών που πραγματοποίησαν οι αλγόριθμοι , "time AC-3" είναι ο χρόνος που χρειάστηκε ο AC-3 για να εκτελεστεί (σε δευτερόλεπτα), "time AC-4" είναι ο χρόνος που χρειάστηκε ο AC-4 για να εκτελεστεί, "Mem AC-3" είναι η μνήμη που κατανάλωσε ο AC-3 για να εκτελεστεί (σε megabytes) και "Mem AC-4" είναι η μνήμη που κατανάλωσε ο AC-4 για να εκτελεστεί.

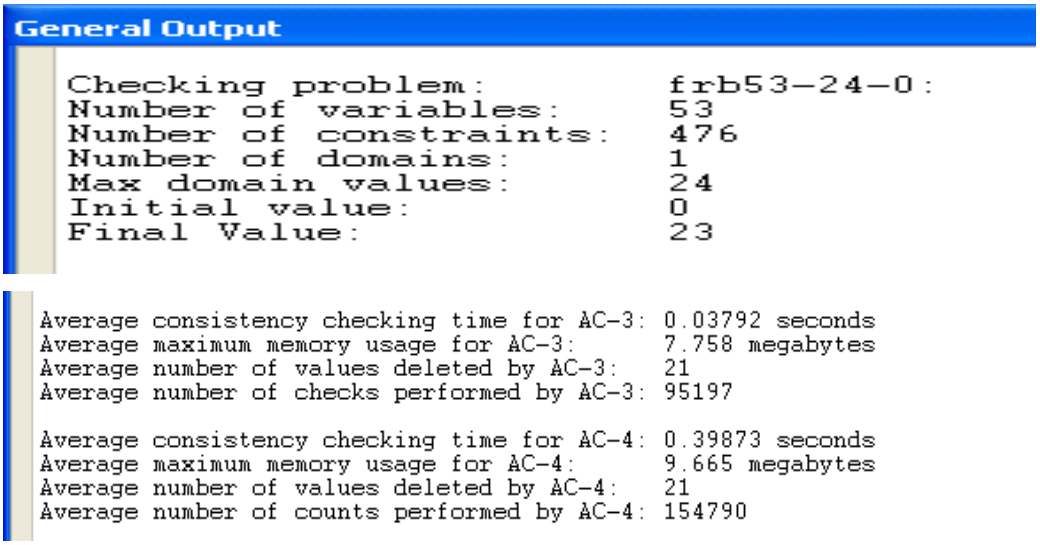
FRB	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
30-15-0	**217**	**30**	**1**	**15**	**450**	**0.034**	**0.064**	**1.562**	**1.620**
35-17-0	**262**	**35**	**1**	**17**	**75**	**0.022**	**0.087**	**2.002**	**2.531**

40-19-0	**321**	**40**	**1**	**19**	**48**	**0.027**	**0.129**	**3.020**	**3.811**
45-21-0	**378**	**45**	**1**	**21**	**49**	**0.030**	**0.203**	**4.421**	**5.571**
50-23-0	**435**	**50**	**1**	**23**	**11**	**0.029**	**0.318**	**6.299**	**8.54**
Average	**323**	**40**	**1**	**19**	**109**	**0.029**	**0.194**	**4.193**	**5.582**

Πίνακας 6.4.1.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων frb.



Σχήμα 6.4.1.α: Στατιστικά στοιχεία εκτέλεσης προβλήματος frb35-17-0



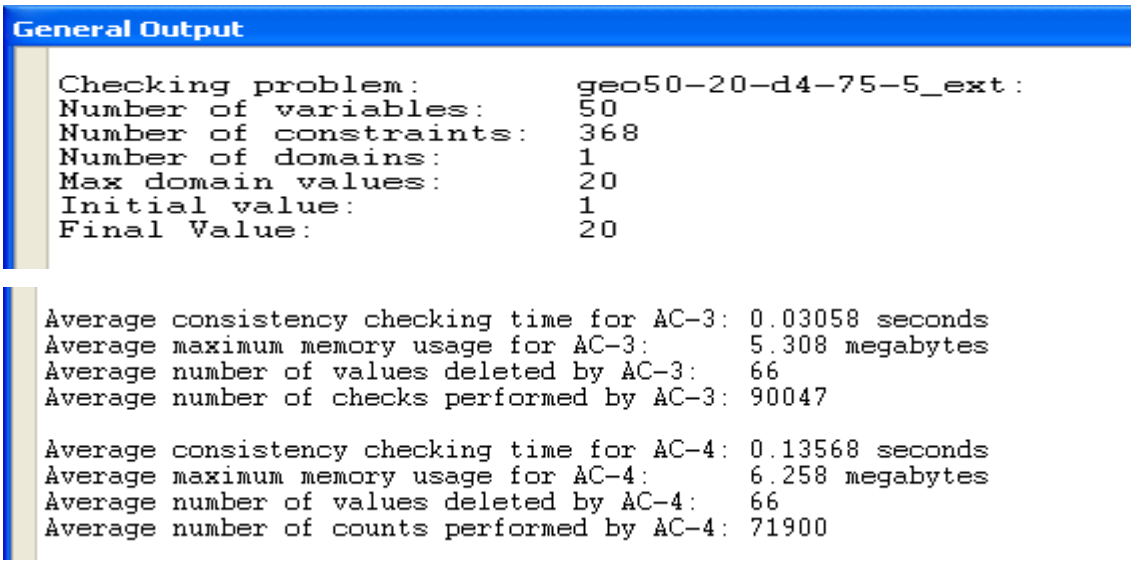
Σχήμα 6.4.1.β: Στατιστικά στοιχεία εκτέλεσης προβλήματος frb53-24-0

6.4.2 Τα Προβλήματα GEO.

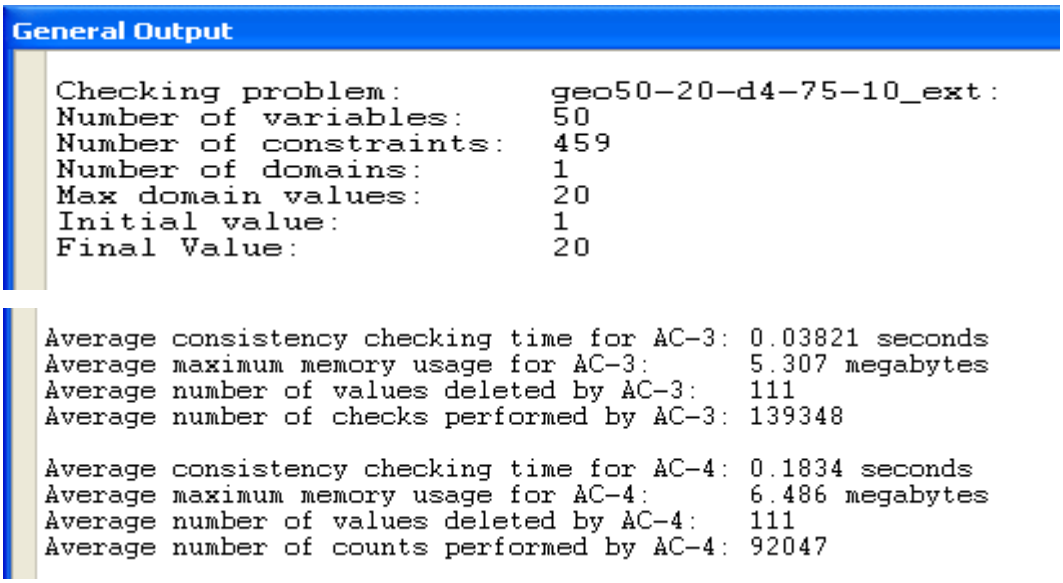
Με βάση τα επακόλουθα στατιστικά στοιχεία, το ίδιο εξακολουθεί και για τα προβλήματα geo που έχουν επίσης μερικές δεκάδες μεταβλητές, και μερικές εκατοντάδες περιορισμούς. Δηλαδή ο αλγόριθμος AC-3 πετυχαίνει καλύτερο χρόνο εκτέλεσης απ'ότι ο AC-4. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

GEO	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
50-20-d4-75-1	**472**	**50**	**1**	**20**	**1000**	**0.148**	**0.259**	**6.375**	**7.503**
50-20-d4-75-2	**466**	**50**	**1**	**20**	**1000**	**0.139**	**0.250**	**6.375**	**7.508**
50-20-d4-75-3	**394**	**50**	**1**	**20**	**72**	**0.072**	**0.192**	**6.375**	**7.385**
50-20-d4-75-4	**418**	**50**	**1**	**20**	**92**	**0.073**	**0.196**	**6.375**	**7.409**
50-20-d4-75-5	**368**	**50**	**1**	**20**	**66**	**0.069**	**0.175**	**6.375**	**7.324**
50-20-d4-75-6	**376**	**50**	**1**	**20**	**102**	**0.070**	**0.180**	**6.375**	**7.333**
50-20-d4-75-7	**471**	**50**	**1**	**20**	**1000**	**0.146**	**0.249**	**6.375**	**7.501**
50-20-d4-75-8	**477**	**50**	**1**	**20**	**93**	**0.078**	**0.237**	**6.375**	**7.532**
50-20-d4-75-9	**451**	**50**	**1**	**20**	**109**	**0.073**	**0.225**	**6.375**	**7.498**
50-20-d4-75-10	**459**	**50**	**1**	**20**	**111**	**0.086**	**0.229**	**6.697**	**7.490**
Average	**435**	**50**	**1**	**20**	**364**	**0.095**	**0.219**	**6.407**	**7.448**

Πίνακας 6.4.2.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων geo.



Σχήμα 6.4.2.α: Στατιστικά στοιχεία εκτέλεσης προβλήματος geo50-20-d4-75-5_ext



Σχήμα 6.4.2.β: Στατιστικά στοιχεία εκτέλεσης προβλήματος geo50-20-d4-75-10_ext

6.4.3 Τα Προβλήματα QCP.

Τα προβλήματα QCP χωρίζονται σε τέσσερεις υποκατηγορίες. Καθεμία από αυτές έχει διαφορετικό αριθμό περιορισμών, μεταβλητών, πεδίων ορισμού και μέγεθος μέγιστου πεδίου ορισμού. Κατά την εκτέλεση αυτών των προβλημάτων γίνεται πλέον φανερό ότι καθώς ο αριθμός των πεδίων ορισμού αυξάνεται, ο αριθμός των μεταβλητών ξεπερνά τις μερικές

δεκάδες και ο αριθμός των περιορισμών ξεπερνά τις μερικές εκατοντάδες , ο αλγόριθμος AC-3 αρχίζει να πετυχαίνει χειρότερο χρόνο εκτέλεσης απ'ότι ο AC-4. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

QCP	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
10-67-0	**900**	**100**	**11**	**10**	**364**	**0.098**	**0.094**	**6.535**	**7.074**
10-67-1	**900**	**100**	**11**	**10**	**355**	**0.088**	**0.084**	**6.274**	**7.076**
10-67-2	**900**	**100**	**11**	**10**	**371**	**0.091**	**0.081**	**6.284**	**7.075**
10-67-3	**900**	**100**	**11**	**10**	**368**	**0.091**	**0.085**	**6.284**	**7.072**
10-67-4	**900**	**100**	**11**	**10**	**358**	**0.089**	**0.086**	**6.284**	**7.088**
10-67-5	**900**	**100**	**11**	**10**	**370**	**0.090**	**0.082**	**6.284**	**7.077**
10-67-6	**900**	**100**	**11**	**10**	**345**	**0.086**	**0.087**	**6.278**	**7.087**
10-67-7	**900**	**100**	**11**	**10**	**357**	**0.088**	**0.084**	**6.284**	**7.090**
10-67-8	**900**	**100**	**11**	**10**	**352**	**0.088**	**0.086**	**6.284**	**7.093**
10-67-9	**900**	**100**	**11**	**10**	**351**	**0.087**	**0.087**	**6.274**	**7.090**
10-67-10	**900**	**100**	**11**	**10**	**362**	**0.094**	**0.089**	**6.281**	**7.070**
10-67-11	**900**	**100**	**11**	**10**	**352**	**0.087**	**0.088**	**6.284**	**7.098**
10-67-12	**900**	**100**	**11**	**10**	**359**	**0.087**	**0.082**	**6.278**	**7.066**
10-67-13	**900**	**100**	**11**	**10**	**363**	**0.090**	**0.083**	**6.284**	**7.076**
10-67-14	**900**	**100**	**11**	**10**	**364**	**0.090**	**0.083**	**6.282**	**7.071**
Average	**900**	**100**	**11**	**10**	**359**	**0.090**	**0.085**	**6.298**	**7.08**

Πίνακας 6.4.3.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qcp_1.

QCP	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
15-120-0	**3150**	**225**	**16**	**15**	**1269**	**1.236**	**0.473**	**56.611**	**61.436**
15-120-1	**3150**	**225**	**16**	**15**	**1295**	**1.274**	**0.444**	**56.364**	**61.242**
15-120-2	**3150**	**225**	**16**	**15**	**1276**	**1.259**	**0.456**	**56.366**	**61.274**
15-120-3	**3150**	**225**	**16**	**15**	**1294**	**1.260**	**0.455**	**56.356**	**61.255**
15-120-4	**3150**	**225**	**16**	**15**	**1283**	**1.245**	**0.454**	**56.361**	**61.273**
15-120-5	**3150**	**225**	**16**	**15**	**1317**	**1.295**	**0.436**	**56.355**	**61.174**
15-120-6	**3150**	**225**	**16**	**15**	**1298**	**1.266**	**0.445**	**56.366**	**61.216**
15-120-7	**3150**	**225**	**16**	**15**	**1296**	**1.230**	**0.448**	**56.355**	**61.255**
15-120-8	**3150**	**225**	**16**	**15**	**1286**	**1.238**	**0.456**	**56.360**	**61.241**
15-120-9	**3150**	**225**	**16**	**15**	**1298**	**1.269**	**0.452**	**56.361**	**61.209**
15-120-10	**3150**	**225**	**16**	**15**	**1288**	**1.262**	**0.453**	**56.366**	**61.289**
15-120-11	**3150**	**225**	**16**	**15**	**1302**	**1.264**	**0.442**	**56.356**	**61.185**
15-120-12	**3150**	**225**	**16**	**15**	**1283**	**1.247**	**0.446**	**56.355**	**61.209**
15-120-13	**3150**	**225**	**16**	**15**	**1301**	**1.282**	**0.439**	**56.366**	**61.195**
15-120-14	**3150**	**225**	**16**	**15**	**1293**	**1.266**	**0.444**	**56.368**	**61.236**
Average	**3150**	**225**	**16**	**15**	**1291**	**1.260**	**0.450**	**56.378**	**61.246**

Πίνακας 6.4.3.β: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qcp_2.

QCP	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
20-187-0	**7600**	**400**	**21**	**20**	**2912**	**8.956**	**1.840**	**309.84**	**344.86**
20-187-1	**7600**	**400**	**21**	**20**	**2894**	**8.899**	**1.892**	**309.23**	**345.184**

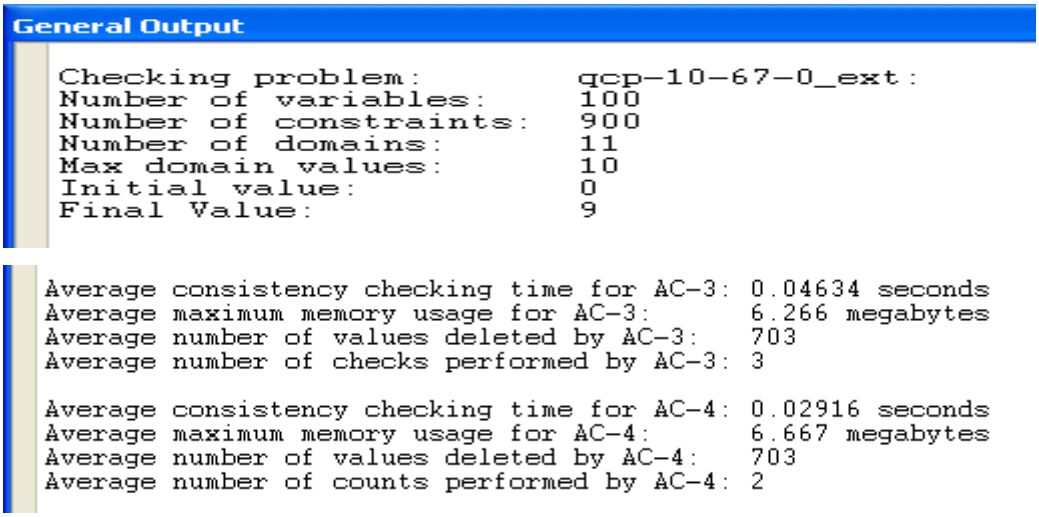
20-187-2	**7600**	**400**	**21**	**20**	**2928**	**9.222**	**1.848**	**309.23**	**345.65**
20-187-3	**7600**	**400**	**21**	**20**	**2922**	**9.431**	**1.958**	**309.23**	**345.79**
20-187-4	**7600**	**400**	**21**	**20**	**2942**	**9.733**	**1.854**	**309.23**	**345.73**
20-187-5	**7600**	**400**	**21**	**20**	**2893**	**9.512**	**2.018**	**309.23**	**345.91**
20-187-6	**7600**	**400**	**21**	**20**	**2925**	**9.162**	**1.995**	**309.23**	**345.38**
20-187-7	**7600**	**400**	**21**	**20**	**2924**	**9.648**	**1.870**	**309.23**	**345.82**
20-187-8	**7600**	**400**	**21**	**20**	**2900**	**9.312**	**1.963**	**309.23**	**345.73**
20-187-9	**7600**	**400**	**21**	**20**	**2927**	**9.210**	**1.929**	**309.23**	**345.63**
20-187-10	**7600**	**400**	**21**	**20**	**2937**	**8.549**	**1.834**	**309.23**	**345.53**
20-187-11	**7600**	**400**	**21**	**20**	**2913**	**9.110**	**1.926**	**309.23**	**345.87**
20-187-12	**7600**	**400**	**21**	**20**	**2932**	**8.985**	**2.001**	**309.23**	**345.92**
20-187-13	**7600**	**400**	**21**	**20**	**2971**	**9.143**	**1.911**	**309.23**	**345.64**
20-187-14	**7600**	**400**	**21**	**20**	**2911**	**9.445**	**1.881**	**309.23**	**345.838**
Average	**7600**	**400**	**21**	**20**	**2922**	**9.221**	**1.915**	**309.27**	**345.63**

Πίνακας 6.4.3.γ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qcp_3.

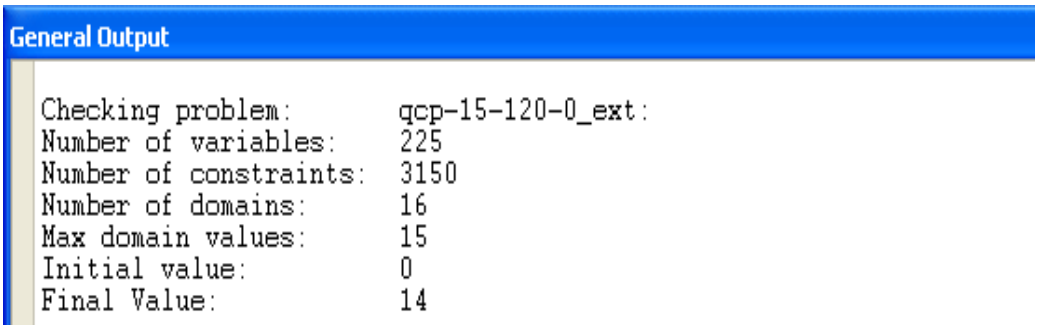
QCP	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-264-0	**15000**	**625**	**26**	**25**	**5449**	**73.27**	**7.13**	**1089.8**	**1156.9**
25-264-1	**15000**	**625**	**26**	**25**	**5419**	**71.34**	**5.80**	**1088.1**	**1155.9**
25-264-2	**15000**	**625**	**26**	**25**	**5431**	**69.69**	**6.40**	**1088.2**	**1137.2**
25-264-3	**15000**	**625**	**26**	**25**	**5437**	**71.11**	**5.83**	**1088.1**	**1155.4**
25-264-4	**15000**	**625**	**26**	**25**	**5439**	**71.92**	**6.39**	**1088.2**	**1137.1**
25-264-5	**15000**	**625**	**26**	**25**	**5439**	**70.87**	**5.82**	**1088.1**	**1156.02**
25-264-6	**15000**	**625**	**26**	**25**	**5404**	**72.26**	**6.43**	**1088.2**	**1137.2**

25-264-7	**15000**	**625**	**26**	**25**	**5399**	**71.19**	**6.07**	**1088.1**	**1155.7**
25-264-8	**15000**	**625**	**26**	**25**	**5440**	**74.86**	**6.52**	**1088.2**	**1137.0**
25-264-9	**15000**	**625**	**26**	**25**	**5433**	**76.29**	**5.98**	**1088.1**	**1156.5**
25-264-10	**15000**	**625**	**26**	**25**	**5406**	**75.19**	**7.28**	**1088.2**	**1137.3**
25-264-11	**15000**	**625**	**26**	**25**	**5381**	**76.01**	**6.15**	**1088.1**	**1156.8**
25-264-12	**15000**	**625**	**26**	**25**	**5352**	**75.22**	**6.50**	**1088.2**	**1137.4**
25-264-13	**15000**	**625**	**26**	**25**	**5392**	**70.68**	**5.83**	**1088.1**	**1156.9**
25-264-14	**15000**	**625**	**26**	**25**	**5451**	**69.12**	**6.40**	**1088.2**	**1137.2**
Average	**15000**	**625**	**26**	**25**	**5418**	**72.61**	**6.30**	**1088.3**	**1147.4**

Πίνακας 6.4.3.δ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qcp_4.



Σχήμα 6.4.3.α: Στατιστικά στοιχεία εκτέλεσης προβλήματος qcp-10-67-0_ext.



```
Average consistency checking time for AC-3: 0.37462 seconds
Average maximum memory usage for AC-3:      56.248 megabytes
Average number of values deleted by AC-3:   1905
Average number of checks performed by AC-3: 496

Average consistency checking time for AC-4: 0.15283 seconds
Average maximum memory usage for AC-4:      59.212 megabytes
Average number of values deleted by AC-4:   1905
Average number of counts performed by AC-4: 122
```

Σχήμα 6.4.3.β: Στατιστικά στοιχεία εκτέλεσης προβλήματος qcp-15-120-0_ext

6.4.4 Τα Προβλήματα QWH.

Τα προβλήματα QWH χωρίζονται σε τέσσερεις υποκατηγορίες. Καθεμία από αυτές έχει διαφορετικό αριθμό περιορισμών, μεταβλητών, πεδίων ορισμού και μέγεθος μέγιστου πεδίου ορισμού. Με την εκτέλεση των προβλημάτων που ανήκουν σε αυτές τις υποκατηγορίες ,επαληθεύτηκαν τα συμπεράσματα που προέκυψαν από την εκτέλεση και των προβλημάτων QCP. Δηλαδή ,καθώς ο αριθμός των μεταβλητών ξεπερνά τις μερικές εκατοντάδες και ο αριθμός των περιορισμών ξεπερνά τις μερικές χιλιάδες ,ο αλγόριθμος AC-3 φανερά πετυχαίνει αισθητά χειρότερο χρόνο εκτέλεσης απ'ότι ο AC-4. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

QWH	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
10-57-0	**900**	**100**	**11**	**10**	**385**	**0.099**	**0.067**	**6.535**	**6.919**
10-57-1	**900**	**100**	**11**	**10**	**377**	**0.091**	**0.062**	**6.276**	**6.925**
10-57-2	**900**	**100**	**11**	**10**	**379**	**0.089**	**0.062**	**6.276**	**6.933**
10-57-3	**900**	**100**	**11**	**10**	**380**	**0.089**	**0.058**	**6.276**	**6.921**
10-57-4	**900**	**100**	**11**	**10**	**369**	**0.088**	**0.057**	**6.276**	**6.933**
10-57-5	**900**	**100**	**11**	**10**	**374**	**0.090**	**0.056**	**6.276**	**6.923**

10-57-6	**900**	**100**	**11**	**10**	**366**	**0.091**	**0.060**	**6.276**	**6.937**
10-57-7	**900**	**100**	**11**	**10**	**382**	**0.092**	**0.055**	**6.268**	**6.919**
10-57-8	**900**	**100**	**11**	**10**	**364**	**0.087**	**0.057**	**6.277**	**6.922**
10-57-9	**900**	**100**	**11**	**10**	**345**	**0.083**	**0.061**	**6.276**	**6.943**
Average	**900**	**100**	**11**	**10**	**372**	**0.090**	**0.060**	**6.301**	**6.927**

Πίνακας 6.4.4.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qwh_1.

QWH	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
15-106-0	**3150**	**225**	**16**	**15**	**1204**	**1.147**	**0.354**	**56.60**	**63.52**
15-106-1	**3150**	**225**	**16**	**15**	**1178**	**1.118**	**0.351**	**56.35**	**63.56**
15-106-2	**3150**	**225**	**16**	**15**	**1221**	**1.133**	**0.335**	**56.35**	**63.51**
15-106-3	**3150**	**225**	**16**	**15**	**1170**	**1.087**	**0.350**	**56.35**	**63.58**
15-106-4	**3150**	**225**	**16**	**15**	**1214**	**1.159**	**0.330**	**56.34**	**63.50**
15-106-5	**3150**	**225**	**16**	**15**	**1194**	**1.123**	**0.336**	**56.35**	**63.53**
15-106-6	**3150**	**225**	**16**	**15**	**1178**	**1.111**	**0.345**	**56.35**	**63.57**
15-106-7	**3150**	**225**	**16**	**15**	**1197**	**1.133**	**0.342**	**56.35**	**63.53**
15-106-8	**3150**	**225**	**16**	**15**	**1223**	**1.160**	**0.335**	**56.35**	**63.51**
15-106-9	**3150**	**225**	**16**	**15**	**1189**	**1.114**	**0.349**	**56.34**	**63.56**
Average	**3150**	**225**	**16**	**15**	**1196**	**1.128**	**0.343**	**56.38**	**63.53**

Πίνακας 6.4.4.β: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qwh_2.

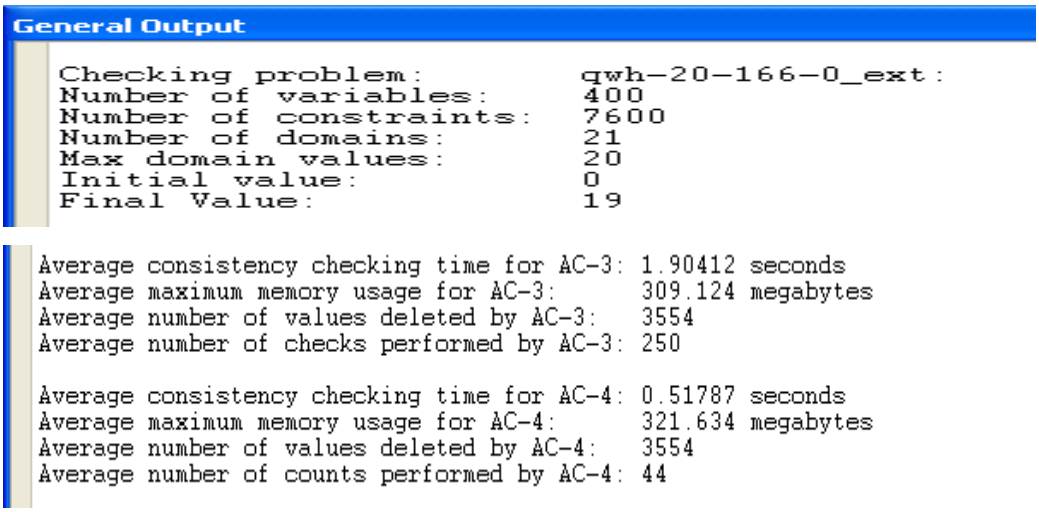
QWH	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
20-166-0	**7600**	**400**	**21**	**20**	**2674**	**8.077**	**1.550**	**309.8**	**331.2**

20-166-1	**7600**	**400**	**21**	**20**	**2690**	**8.931**	**1.426**	**309.2**	**335.2**
20-166-2	**7600**	**400**	**21**	**20**	**2647**	**8.255**	**1.454**	**309.2**	**336.1**
20-166-3	**7600**	**400**	**21**	**20**	**2691**	**8.084**	**1.428**	**309.2**	**335.4**
20-166-4	**7600**	**400**	**21**	**20**	**2694**	**8.562**	**1.419**	**309.2**	**335.1**
20-166-5	**7600**	**400**	**21**	**20**	**2686**	**8.427**	**1.448**	**309.2**	**334.9**
20-166-6	**7600**	**400**	**21**	**20**	**2681**	**8.392**	**1.419**	**309.2**	**335.9**
20-166-7	**7600**	**400**	**21**	**20**	**2681**	**7.971**	**1.454**	**309.2**	**335.4**
20-166-8	**7600**	**400**	**21**	**20**	**2654**	**8.203**	**1.493**	**309.2**	**335.1**
20-166-9	**7600**	**400**	**21**	**20**	**2715**	**8.614**	**1.393**	**309.2**	**334.8**
Average	**7600**	**400**	**21**	**20**	**2681**	**8.352**	**1.448**	**309.3**	**334.9**

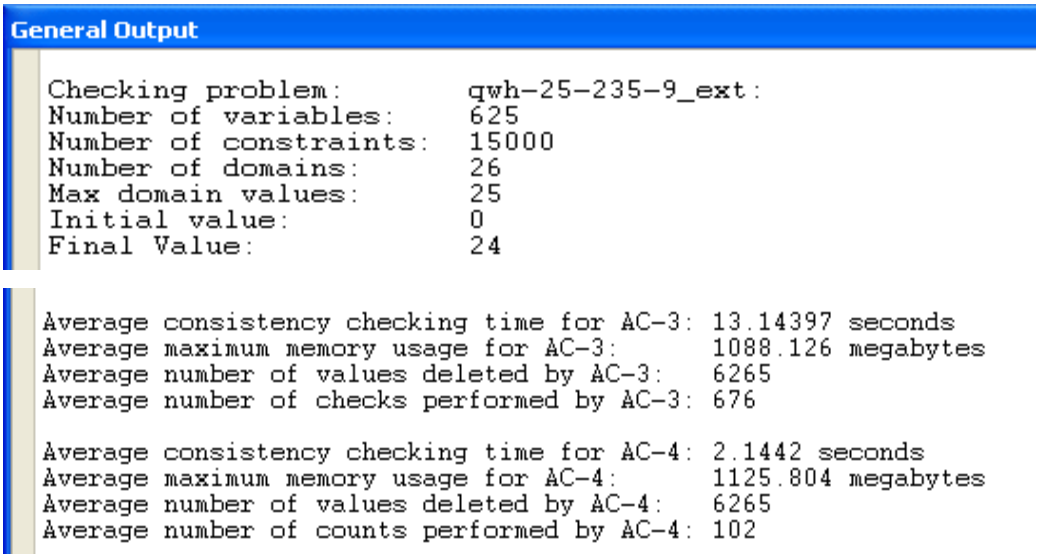
Πίνακας 6.4.4.γ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qwh_3.

QWH	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-235-0	**15000**	**625**	**26**	**25**	**4947**	**60.72**	**4.70**	**1089.8**	**1154.2**
25-235-1	**15000**	**625**	**26**	**25**	**4956**	**62.323**	**5.44**	**1088.2**	**1134.3**
25-235-2	**15000**	**625**	**26**	**25**	**4921**	**61.94**	**4.50**	**1088.1**	**1154.2**
25-235-3	**15000**	**625**	**26**	**25**	**4960**	**64.64**	**5.34**	**1088.2**	**1134.4**
25-235-4	**15000**	**625**	**26**	**25**	**4929**	**64.89**	**4.65**	**1088.1**	**1154.0**
25-235-5	**15000**	**625**	**26**	**25**	**4959**	**64.62**	**5.26**	**1088.2**	**1134.4**
25-235-6	**15000**	**625**	**26**	**25**	**4930**	**62.05**	**4.50**	**1088.1**	**1154.2**
25-235-7	**15000**	**625**	**26**	**25**	**4944**	**62.85**	**5.11**	**1088.2**	**1134.4**
25-235-8	**15000**	**625**	**26**	**25**	**4925**	**62.24**	**4.47**	**1088.1**	**1134.5**
25-235-9	**15000**	**625**	**26**	**25**	**4963**	**59.13**	**5.00**	**1088.2**	**1134.4**
Average	**15000**	**625**	**26**	**25**	**4943**	**62.54**	**4.90**	**1088.3**	**1142.3**

Πίνακας 6.4.4.δ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων qwh_4.



Σχήμα 6.4.4.α: Στατιστικά στοιχεία εκτέλεσης προβλήματος qwh-20-166-0_ext



Σχήμα 6.4.4.β: Στατιστικά στοιχεία εκτέλεσης προβλήματος qwh-25-235-9_ext

6.4.5 Τα Προβλήματα LANGFORD.

Τα προβλήματα LANGFORD που εξετάστηκαν έχουν μικρό αριθμό περιορισμών, μεταβλητών και πεδίων ορισμού. Επίσης παρουσιάζουν συνέπεια τόξου από την αρχή. Με την εκτέλεση των αλγορίθμων προκύπτει ξεκάθαρα ότι ο αλγόριθμος AC-3 πετυχαίνει πολύ

καλύτερα χρονικά αποτελέσματα σε σύγκριση με τον AC-4 ,όταν τα προβλήματα παρουσιάζουν από την αρχή συνέπεια τόξου. Αυτό οφείλεται στο γεγονός ότι ο αλγόριθμος AC-4 σπαταλάει πολύ χρόνο κατά την διαδικασία της αρχικοποίησης, γεγονός που στην συγκεκριμένη περίπτωση είναι σημαντικό χρονικό σφάλμα. Επίσης ο AC-3 χρησιμοποιεί πολύ λιγότερη μνήμη συγκριτικά με τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

LANGFORD	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
2-4	**32**	**8**	**1**	**8**	**0**	**0.014**	**0.022**	**0.33**	**0.56**
3-9	**369**	**27**	**1**	**27**	**0**	**0.023**	**4.726**	**2.67**	**7.72**
3-11	**550**	**33**	**1**	**33**	**0**	**0.028**	**16.90**	**5.42**	**17.11**
slangford-3-11	**572**	**33**	**1**	**33**	**0**	**0.028**	**16.92**	**5.42**	**17.11**
Average	**381**	**25**	**1**	**25**	**0**	**0.023**	**9.64**	**3.52**	**10.57**

Πίνακας 6.4.5.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων langford.

6.4.6 Τα Προβλήματα COMPOSED.

Τα προβλήματα COMPOSED χωρίζονται σε εννιά υποκατηγορίες. Καθεμία από αυτές έχει διαφορετικό αριθμό περιορισμών και ίδιο αριθμό μεταβλητών, πεδίων ορισμού και μεγέθους μέγιστου πεδίου ορισμού. Επίσης στα αποτελέσματα που ακολουθούν φαίνεται ότι οι διαγραφές που έγιναν είναι αρκετά λίγες. Αυτό φανερώνει ότι τα προβλήματα composed που εξετάστηκαν ήταν εξ αρχής πολύ κοντά στη συνέπεια τόξου. Αυτό το στοιχείο σε συνδυασμό με το μικρό αριθμό των παραμέτρων αυτού του προβλήματος εξηγεί γιατί ο αλγόριθμος AC-3 πετυχαίνει καλύτερο χρόνο εκτέλεσης απ’ότι ο AC-4. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-1-2-0	**224**	**33**	**1**	**10**	**8**	**0.021**	**0.100**	**1.196**	**1.310**
25-1-2-1	**224**	**33**	**1**	**10**	**14**	**0.020**	**0.096**	**0.935**	**1.309**
25-1-2-2	**224**	**33**	**1**	**10**	**3**	**0.018**	**0.090**	**0.935**	**1.311**
25-1-2-3	**224**	**33**	**1**	**10**	**9**	**0.020**	**0.090**	**0.935**	**1.309**
25-1-2-4	**224**	**33**	**1**	**10**	**1**	**0.018**	**0.091**	**0.935**	**1.310**
25-1-2-5	**224**	**33**	**1**	**10**	**10**	**0.019**	**0.091**	**0.935**	**1.309**
25-1-2-6	**224**	**33**	**1**	**10**	**9**	**0.018**	**0.093**	**0.935**	**1.309**
25-1-2-7	**224**	**33**	**1**	**10**	**3**	**0.019**	**0.091**	**0.935**	**1.310**
25-1-2-8	**224**	**33**	**1**	**10**	**7**	**0.019**	**0.091**	**0.935**	**1.310**
25-1-2-9	**224**	**33**	**1**	**10**	**3**	**0.019**	**0.091**	**0.935**	**1.313**
Average	**224**	**33**	**1**	**10**	**6**	**0.019**	**0.092**	**0.961**	**1.310**

Πίνακας 6.4.6.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_1.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-1-25-0	**247**	**33**	**1**	**10**	**8**	**0.025**	**0.116**	**1.199**	**1.351**
25-1-25-1	**247**	**33**	**1**	**10**	**14**	**0.020**	**0.112**	**0.935**	**1.349**
25-1-25-2	**247**	**33**	**1**	**10**	**3**	**0.020**	**0.106**	**0.935**	**1.351**
25-1-25-3	**247**	**33**	**1**	**10**	**9**	**0.020**	**0.103**	**0.935**	**1.349**
25-1-25-4	**247**	**33**	**1**	**10**	**1**	**0.019**	**0.103**	**0.935**	**1.349**
25-1-25-5	**247**	**33**	**1**	**10**	**10**	**0.021**	**0.105**	**0.935**	**1.349**
25-1-25-6	**247**	**33**	**1**	**10**	**9**	**0.021**	**0.105**	**0.935**	**1.349**

25-1-25-7	**247**	**33**	**1**	**10**	**3**	**0.021**	**0.105**	**0.935**	**1.350**
25-1-25-8	**247**	**33**	**1**	**10**	**7**	**0.021**	**0.104**	**0.935**	**1.350**
25-1-25-9	**247**	**33**	**1**	**10**	**3**	**0.020**	**0.108**	**0.935**	**1.352**
Average	**247**	**33**	**1**	**10**	**6**	**0.021**	**0.107**	**0.962**	**1.350**

Πίνακας 6.4.6.β: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_2.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-1-40-0	**262**	**33**	**1**	**10**	**8**	**0.020**	**0.122**	**1.202**	**1.377**
25-1-40-0	**262**	**33**	**1**	**10**	**14**	**0.017**	**0.121**	**0.935**	**1.374**
25-1-40-0	**262**	**33**	**1**	**10**	**3**	**0.017**	**0.112**	**0.935**	**1.378**
25-1-40-0	**262**	**33**	**1**	**10**	**9**	**0.016**	**0.115**	**0.935**	**1.375**
25-1-40-0	**262**	**33**	**1**	**10**	**1**	**0.015**	**0.113**	**0.935**	**1.376**
25-1-40-0	**262**	**33**	**1**	**10**	**10**	**0.015**	**0.112**	**0.935**	**1.375**
25-1-40-0	**262**	**33**	**1**	**10**	**9**	**0.016**	**0.109**	**0.935**	**1.376**
25-1-40-0	**262**	**33**	**1**	**10**	**3**	**0.017**	**0.113**	**0.935**	**1.377**
25-1-40-0	**262**	**33**	**1**	**10**	**7**	**0.015**	**0.114**	**0.935**	**1.375**
25-1-40-0	**262**	**33**	**1**	**10**	**3**	**0.016**	**0.115**	**0.935**	**1.379**
Average	**262**	**33**	**1**	**10**	**6**	**0.016**	**0.115**	**0.962**	**1.376**

Πίνακας 6.4.6.γ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_3.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-1-80-0	**302**	**33**	**1**	**10**	**8**	**0.023**	**0.155**	**1.197**	**1.446**
25-1-80-1	**302**	**33**	**1**	**10**	**14**	**0.020**	**0.147**	**0.935**	**1.444**

25-1-80-2	**302**	**33**	**1**	**10**	**3**	**0.021**	**0.138**	**0.935**	**1.447**
25-1-80-3	**302**	**33**	**1**	**10**	**9**	**0.019**	**0.139**	**0.935**	**1.445**
25-1-80-4	**302**	**33**	**1**	**10**	**1**	**0.020**	**0.140**	**0.935**	**1.445**
25-1-80-5	**302**	**33**	**1**	**10**	**10**	**0.020**	**0.139**	**0.935**	**1.444**
25-1-80-6	**302**	**33**	**1**	**10**	**9**	**0.019**	**0.142**	**0.935**	**1.446**
25-1-80-7	**302**	**33**	**1**	**10**	**3**	**0.021**	**0.140**	**0.935**	**1.447**
25-1-80-8	**302**	**33**	**1**	**10**	**7**	**0.019**	**0.138**	**0.935**	**1.445**
25-1-80-9	**302**	**33**	**1**	**10**	**3**	**0.019**	**0.139**	**0.935**	**1.448**
Average	**302**	**33**	**1**	**10**	**8**	**0.020**	**0.142**	**0.962**	**1.446**

Πίνακας 6.4.6.δ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_4.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
25-10-20-0	**620**	**105**	**1**	**10**	**1**	**0.035**	**0.254**	**7.15**	**8.230**
25-10-20-1	**620**	**105**	**1**	**10**	**2**	**0.031**	**0.255**	**6.886**	**8.232**
25-10-20-2	**620**	**105**	**1**	**10**	**0**	**0.030**	**0.247**	**6.886**	**8.236**
25-10-20-3	**620**	**105**	**1**	**10**	**3**	**0.031**	**0.244**	**6.886**	**8.235**
25-10-20-4	**620**	**105**	**1**	**10**	**2**	**0.032**	**0.243**	**6.886**	**8.231**
25-10-20-5	**620**	**105**	**1**	**10**	**4**	**0.031**	**0.244**	**6.886**	**8.232**
25-10-20-6	**620**	**105**	**1**	**10**	**3**	**0.030**	**0.240**	**6.886**	**8.237**
25-10-20-7	**620**	**105**	**1**	**10**	**2**	**0.030**	**0.246**	**6.886**	**8.229**
25-10-20-8	**620**	**105**	**1**	**10**	**2**	**0.031**	**0.243**	**6.886**	**8.229**
25-10-20-9	**620**	**105**	**1**	**10**	**3**	**0.029**	**0.245**	**6.886**	**8.231**
Average	**620**	**105**	**1**	**10**	**2**	**0.031**	**0.246**	**6.912**	**8.232**

Πίνακας 6.4.6.ε: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_5.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
75-1-2-0	**624**	**83**	**1**	**10**	**12**	**0.031**	**0.275**	**4.681**	**5.691**
75-1-2-1	**624**	**83**	**1**	**10**	**6**	**0.028**	**0.277**	**4.421**	**5.695**
75-1-2-2	**624**	**83**	**1**	**10**	**6**	**0.026**	**0.271**	**4.421**	**5.695**
75-1-2-3	**624**	**83**	**1**	**10**	**5**	**0.025**	**0.268**	**4.421**	**5.699**
75-1-2-4	**624**	**83**	**1**	**10**	**20**	**0.026**	**0.269**	**4.421**	**5.696**
75-1-2-5	**624**	**83**	**1**	**10**	**5**	**0.028**	**0.267**	**4.421**	**5.694**
75-1-2-6	**624**	**83**	**1**	**10**	**2**	**0.026**	**0.266**	**4.421**	**5.693**
75-1-2-7	**624**	**83**	**1**	**10**	**11**	**0.026**	**0.269**	**4.421**	**5.694**
75-1-2-8	**624**	**83**	**1**	**10**	**9**	**0.027**	**0.267**	**4.421**	**5.695**
75-1-2-9	**624**	**83**	**1**	**10**	**5**	**0.026**	**0.267**	**4.421**	**5.701**
Average	**624**	**83**	**1**	**10**	**8**	**0.0275**	**0.270**	**4.447**	**5.695**

Πίνακας 6.4.6.στ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_6.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
75-1-25-0	**647**	**83**	**1**	**10**	**12**	**0.029**	**0.287**	**4.686**	**5.733**
75-1-25-1	**647**	**83**	**1**	**10**	**6**	**0.026**	**0.283**	**4.421**	**5.737**
75-1-25-2	**647**	**83**	**1**	**10**	**6**	**0.024**	**0.277**	**4.421**	**5.74**
75-1-25-3	**647**	**83**	**1**	**10**	**5**	**0.022**	**0.276**	**4.421**	**5.743**
75-1-25-4	**647**	**83**	**1**	**10**	**20**	**0.023**	**0.276**	**4.421**	**5.747**
75-1-25-5	**647**	**83**	**1**	**10**	**5**	**0.025**	**0.278**	**4.421**	**5.735**
75-1-25-6	**647**	**83**	**1**	**10**	**2**	**0.023**	**0.280**	**4.421**	**5.734**

75-1-25-7	**647**	**83**	**1**	**10**	**11**	**0.022**	**0.279**	**4.421**	**5.738**
75-1-25-8	**647**	**83**	**1**	**10**	**9**	**0.022**	**0.277**	**4.421**	**5.738**
75-1-25-9	**647**	**83**	**1**	**10**	**5**	**0.023**	**0.281**	**4.421**	**5.744**
Average	**647**	**83**	**1**	**10**	**8**	**0.024**	**0.279**	**4.448**	**5.739**

Πίνακας 6.4.6.ζ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_7.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
75-1-25-0	**662**	**83**	**1**	**10**	**12**	**0.029**	**0.296**	**4.686**	**5.780**
75-1-25-0	**662**	**83**	**1**	**10**	**6**	**0.025**	**0.293**	**4.421**	**5.777**
75-1-25-0	**662**	**83**	**1**	**10**	**6**	**0.025**	**0.288**	**4.421**	**5.763**
75-1-25-0	**662**	**83**	**1**	**10**	**5**	**0.023**	**0.285**	**4.421**	**5.773**
75-1-25-0	**662**	**83**	**1**	**10**	**20**	**0.028**	**0.288**	**4.421**	**5.787**
75-1-25-0	**662**	**83**	**1**	**10**	**5**	**0.023**	**0.289**	**4.421**	**5.778**
75-1-25-0	**662**	**83**	**1**	**10**	**2**	**0.023**	**0.287**	**4.421**	**5.762**
75-1-25-0	**662**	**83**	**1**	**10**	**11**	**0.024**	**0.285**	**4.421**	**5.778**
75-1-25-0	**662**	**83**	**1**	**10**	**9**	**0.025**	**0.289**	**4.421**	**5.766**
75-1-25-0	**662**	**83**	**1**	**10**	**5**	**0.025**	**0.292**	**4.421**	**5.779**
Average	**662**	**83**	**1**	**10**	**8**	**0.025**	**0.289**	**4.448**	**5.774**

Πίνακας 6.4.6.η: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_8.

COMPOSED	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
75-1-80-0	**702**	**83**	**1**	**10**	**12**	**0.033**	**0.323**	**4.681**	**5.939**
75-1-80-1	**702**	**83**	**1**	**10**	**6**	**0.027**	**0.319**	**4.420**	**5.924**

75-1-80-2	**702**	**83**	**1**	**10**	**6**	**0.028**	**0.314**	**4.421**	**5.925**
75-1-80-3	**702**	**83**	**1**	**10**	**5**	**0.029**	**0.311**	**4.421**	**5.927**
75-1-80-4	**702**	**83**	**1**	**10**	**20**	**0.027**	**0.316**	**4.421**	**5.896**
75-1-80-5	**702**	**83**	**1**	**10**	**5**	**0.026**	**0.312**	**4.421**	**5.941**
75-1-80-6	**702**	**83**	**1**	**10**	**2**	**0.025**	**0.311**	**4.421**	**5.93**
75-1-80-7	**702**	**83**	**1**	**10**	**11**	**0.027**	**0.311**	**4.421**	**5.926**
75-1-80-8	**702**	**83**	**1**	**10**	**9**	**0.026**	**0.310**	**4.421**	**5.888**
75-1-80-9	**702**	**83**	**1**	**10**	**5**	**0.028**	**0.315**	**4.421**	**5.936**
Average	**702**	**83**	**1**	**10**	**8**	**0.028**	**0.314**	**4.447**	**5.923**

Πίνακας 6.4.6.θ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων composed_9.

6.4.7 Τα Προβλήματα DRIVER.

Στα προβλήματα DRIVER βλέπουμε ότι καθώς αυξάνεται ο αριθμός των περιορισμών ,των μεταβλητών, των πεδίων ορισμού και του μεγέθους μέγιστου πεδίου ορισμού, ο αλγόριθμος AC-3 τείνει να πετυχαίνει καλύτερα χρονικά αποτελέσματα σε σύγκριση με τον AC-4, παρά τα προγνωστικά που υποστηρίζουν το αντίθετο. Ωστόσο για μία ακόμα φορά αυτό οφείλεται περισσότερο στο γεγονός ότι τα προβλήματα αυτά ήταν εξ αρχής κοντά στην συνέπεια τόξου κι έτσι δεν χρειάστηκε να γίνουν παρά μόνο λίγες διαγραφές τιμών. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

DRIVER	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
01c	**217**	**71**	**3**	**4**	**18**	**0.025**	**0.026**	**1.34**	**1.21**
02c	**4055**	**301**	**6**	**8**	**64**	**0.125**	**0.410**	**38.35**	**42.58**
04c	**3876**	**272**	**9**	**11**	**108**	**0.139**	**0.448**	**48.32**	**52.83**

05c	**6173**	**351**	**8**	**11**	**59**	**0.202**	**0.850**	**80.24**	**93.20**
08c	**9321**	**408**	**8**	**11**	**21**	**0.263**	**1.580**	**108.3**	**129.3**
08cc	**9321**	**408**	**8**	**11**	**21**	**0.259**	**1.550**	**108.3**	**129.4**
09	**17447**	**650**	**9**	**12**	**162**	**1.210**	**6.686**	**338.8**	**366.9**
Average	**7201**	**352**	**7**	**10**	**64**	**0.318**	**1.650**	**103.3**	**116.5**

Πίνακας 6.4.7.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων driver.

6.4.8 Τα Προβλήματα BLACKHOLE.

Τα προβλήματα BLACKHOLE χωρίζονται σε τρεις υποκατηγορίες. Καθεμία από αυτές έχει διαφορετικό αριθμό περιορισμών, μεταβλητών και μέγεθος μέγιστου πεδίου ορισμού. Κατά την εκτέλεση αυτών των προβλημάτων επαληθεύεται ξανά ότι καθώς ο αριθμός των μεταβλητών ξεπερνά τις μερικές δεκάδες και ο αριθμός των περιορισμών ξεπερνά τις μερικές εκατοντάδες , ο αλγόριθμος AC-3 αρχίζει να πετυχαίνει χειρότερο χρόνο εκτέλεσης απ'ότι ο AC-4. Αυτό οφείλεται επίσης στο γεγονός ότι τα προβλήματα αυτά δεν ήταν εξ αρχής κοντά στην συνέπεια τόξου. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

BLACKHOLE	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
4-4-e-0	**432**	**64**	**4**	**16**	**666**	**0.036**	**0.032**	**5.928**	**5.945**
4-4-e-1	**432**	**64**	**4**	**16**	**666**	**0.032**	**0.028**	**5.670**	**5.945**
4-4-e-2	**432**	**64**	**4**	**16**	**666**	**0.033**	**0.027**	**5.670**	**5.945**
4-4-e-3	**432**	**64**	**4**	**16**	**666**	**0.031**	**0.030**	**5.670**	**5.945**
4-4-e-4	**432**	**64**	**4**	**16**	**666**	**0.031**	**0.027**	**5.670**	**5.945**
4-4-e-5	**432**	**64**	**4**	**16**	**666**	**0.031**	**0.028**	**5.670**	**5.945**

4-4-e-6	**432**	**64**	**4**	**16**	**666**	**0.032**	**0.027**	**5.670**	**5.945**
4-4-e-7	**432**	**64**	**4**	**16**	**666**	**0.031**	**0.027**	**5.670**	**5.945**
4-4-e-8	**432**	**64**	**4**	**16**	**666**	**0.033**	**0.027**	**5.670**	**5.945**
4-4-e-9	**432**	**64**	**4**	**16**	**666**	**0.032**	**0.027**	**5.670**	**5.945**
Average	**432**	**64**	**4**	**16**	**666**	**0.032**	**0.028**	**5.696**	**5.945**

Πίνακας 6.4.8.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων blackhole_1.

BLACKHOLE	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
4-7-e-0	**1262**	**112**	**4**	**28**	**304**	**0.214**	**27.14**	**45.10**	**62.15**
4-7-e-1	**1262**	**112**	**4**	**28**	**2088**	**0.171**	**0.137**	**44.75**	**46.16**
4-7-e-2	**1262**	**112**	**4**	**28**	**2088**	**0.172**	**0.136**	**44.75**	**46.16**
4-7-e-3	**1262**	**112**	**4**	**28**	**2088**	**0.170**	**0.134**	**44.75**	**46.16**
4-7-e-4	**1262**	**112**	**4**	**28**	**2088**	**0.173**	**0.135**	**44.75**	**46.16**
4-7-e-5	**1262**	**112**	**4**	**28**	**2088**	**0.171**	**0.135**	**44.75**	**46.16**
4-7-e-6	**1262**	**112**	**4**	**28**	**2088**	**0.171**	**0.135**	**44.75**	**46.16**
4-7-e-7	**1262**	**112**	**4**	**28**	**2088**	**0.170**	**0.138**	**44.75**	**46.16**
4-7-e-8	**1262**	**112**	**4**	**28**	**2088**	**0.172**	**0.135**	**44.75**	**46.16**
4-7-e-9	**1262**	**112**	**4**	**28**	**2088**	**0.170**	**0.135**	**44.75**	**46.16**
Average	**1262**	**112**	**4**	**28**	**1909**	**0.170**	**2.837**	**44.78**	**47.76**

Πίνακας 6.4.8.β: Στατιστικά στοιχεία εκτέλεσης προβλημάτων blackhole_2.

BLACKHOLE	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
4-7-h-0	**1262**	**112**	**4**	**28**	**2088**	**0.145**	**0.111**	**45.10**	**46.27**

4-7-h-1	**1262**	**112**	**4**	**28**	**2088**	**0.132**	**0.109**	**44.86**	**46.27**
4-7-h-2	**1262**	**112**	**4**	**28**	**2088**	**0.130**	**0.107**	**44.86**	**46.27**
4-7-h-3	**1262**	**112**	**4**	**28**	**2088**	**0.131**	**0.106**	**44.86**	**46.27**
4-7-h-4	**1262**	**112**	**4**	**28**	**2088**	**0.129**	**0.111**	**44.86**	**46.27**
4-7-h-5	**1262**	**112**	**4**	**28**	**2088**	**0.133**	**0.107**	**44.86**	**46.27**
4-7-h-6	**1262**	**112**	**4**	**28**	**2088**	**0.133**	**0.107**	**44.86**	**46.27**
4-7-h-7	**1262**	**112**	**4**	**28**	**2088**	**0.131**	**0.105**	**44.86**	**46.27**
4-7-h-8	**1262**	**112**	**4**	**28**	**2088**	**0.126**	**0.106**	**44.86**	**46.27**
4-7-h-9	**1262**	**112**	**4**	**28**	**2088**	**0.127**	**0.107**	**44.86**	**46.27**
Average	**1262**	**112**	**4**	**28**	**2088**	**0.132**	**0.108**	**44.86**	**46.27**

Πίνακας 6.4.8.γ: Στατιστικά στοιχεία εκτέλεσης προβλημάτων blackhole_3.

6.4.9 Τα Προβλήματα HANOI.

Τα προβλήματα QCP έχουν πολύ μικρό αριθμό περιορισμών ,μεταβλητών, πεδίων ορισμού αλλά σχετικά μεγάλο μέγεθος μέγιστου πεδίου ορισμού. Κατά την εκτέλεση αυτών των προβλημάτων γίνεται φανερό ότι καθώς το μέγεθος του μέγιστου πεδίου ορισμού αυξάνεται ο αλγόριθμος AC-4 τείνει να πετυχαίνει χειρότερα χρονικά αποτελέσματα σε σύγκριση με τον AC-3. Αυτό οφείλεται στο γεγονός ότι ο αλγόριθμος AC-4 λειτουργεί πραγματοποιώντας αρκετά ακριβές αρχικοποιήσεις κατά την φάση αρχικοποίησής του. Φυσικά ο AC-3 έχει πάντα λιγότερη χρήση μνήμης από τον AC-4. Και οι δύο αλγόριθμοι καταφέρνουν να διαγράψουν τον ίδιο αριθμό τιμών από τα πεδία ορισμού των μεταβλητών.

HANOI	Cons	Vars	Doms	Max Dom Size	Dels	Time AC-3	Time AC-4	Mem AC-3	Mem AC-4
3	**5**	**6**	**3**	**27**	**80**	**0.013**	**0.016**	**0.658**	**0.425**
4	**13**	**14**	**3**	**81**	**728**	**0.023**	**0.033**	**5.593**	**5.886**

5	**29**	**30**	**3**	**243**	**5838**	**0.107**	**0.290**	**208.3**	**211.3**
Average	**16**	**17**	**3**	**117**	**2215**	**0.047**	**0.113**	**71.534**	**72.55**

Πίνακας 6.4.9.α: Στατιστικά στοιχεία εκτέλεσης προβλημάτων hanoi.

Κεφάλαιο 7

Συμπεράσματα

Τα κύρια συμπεράσματα της εργασίας αφορούν την αποτελεσματικότητα, και την αποδοτικότητα των αλγορίθμων. Η τελευταία θα αξιολογηθεί με κριτήρια την χωρική αλλά και την χρονική πολυπλοκότητα.

Όσον αφορά την αποτελεσματικότητα των αλγορίθμων, τα αποτελέσματα τις εργασίας επιβεβαιώνουν ότι και οι δύο αλγόριθμοι είναι ικανοί να πετύχουν ακριβώς την ίδια τοπική συνέπια (local consistency) διαγράφοντας ακριβώς τις ίδιες τιμές από τα πεδία τιμών των μεταβλητών που συμμετέχουν στους περιορισμούς.

Όσον αφορά την αποδοτικότητα της χωρικής πολυπλοκότητας , με βάση τα αποτελέσματα τις εκτέλεσης των αλγορίθμων μπορούμε να πούμε ότι ο αλγόριθμος AC-3 πετυχαίνει πάντα καλύτερα αποτελέσματα χρησιμοποιώντας λιγότερη μνήμη. Όμως ισχυρισμός της χωρικής ανάλυσης που θέλει τον αλγόριθμο AC-3 να έχει O(e) χωρική πολυπλοκότητα ενώ τον AC-4 O(ed^2), δεν επιβεβαιώνεται φανερά στα αποτελέσματα της εργασίας.

Η χρήση μνήμης δεν φαίνεται να έχει εκθετική διαφορά μεταξύ των αλγορίθμων. Αυτό ωστόσο οφείλεται στον τρόπο προγραμματιστικής υλοποίησης των αλγορίθμων καθώς και οι δύο αλγόριθμοι χρησιμοποιούν κάποιες κοινές δομές δεδομένων οι οποίες καταναλώνουν και το μεγαλύτερο μέρος της μνήμης που χρησιμοποιούν οι αλγόριθμοι. Αυτές οι δομές είναι οι πίνακες που κρατούν τις μεταβλητές που συμμετέχουν στους περιορισμούς, οι πίνακες που κρατούν τα πεδία ορισμού για κάθε μεταβλητή και οι πίνακες που κρατούν τα ζευγάρια τιμών που ικανοποιούν τους περιορισμούς για κάθε ζευγάρι μεταβλητών.

Όσον αφορά την αποδοτικότητα της χρονικής πολυπλοκότητας , παρά το γεγονός ότι με βάση την χρονική ανάλυση ο αλγόριθμος AC-3 έχει χρόνο χειρότερης εκτέλεσης O(ed^3) ενώ ο AC-4 έχει O(ed^2) (όπου e ο αριθμός των περιορισμών και d ο αριθμός των τιμών του μέγιστου πεδίου ορισμού) ,αποδείχθηκε ότι ο αλγόριθμος AC-3 καταναλώνει λιγότερο χρόνο

εκτέλεσης για τα προβλήματα που έχουν λίγες μεταβλητές και μικρό μέγιστο πεδίο ορισμού ,σε σύγκριση με τον AC-4. Επίσης ο αλγόριθμος AC-3 πετυχαίνει καλύτερα αποτελέσματα όταν χρησιμοποιείται σε προβλήματα που παρουσιάζουν εξ αρχής συνέπεια τόξου, ή είναι εξ αρχής κοντά στην συνέπεια τόξου, οπότε και οι τελικές διαγραφές τιμών είναι λίγες. Αυτό οφείλεται στο ότι ο AC-3 δεν κάνει αρχικοποιήσεις και προχωράει κατ'ευθείαν στον έλεγχο ασυνεπειών, αποφεύγοντας έτσι να δαπανήσει πολύτιμο χρόνο για ακριβές αρχικοποιήσεις οι οποίες εν προκειμένω δεν παρουσιάζουν ιδιαίτερη χρησιμότητα για προβλήματα μικρών προδιαγραφών καθώς η περίπτωση χειρότερης χρονικής εκτέλεσης συμβαίνει αρχικά συχνά ,προσεγγίζοντας την περίπτωση μέσης χρονικής εκτέλεσης .

Ωστόσο για τα προβλήματα με μεγάλο αριθμό μεταβλητών και μέγιστου πεδίου ορισμού που βρίσκονται εξ αρχής μακριά από την συνέπεια τόξου, οπότε και ο αλγόριθμος AC-4 πραγματοποιεί πολλές διαγραφές τιμών, τα αποτελέσματα τις χρονικής ανάλυσης επιβεβαιώνονται περίτρανα και γίνεται φανερό ότι ο αλγόριθμος AC-4 πετυχαίνει καλύτερα χρονικά αποτελέσματα από τον AC-3. Αυτό οφείλεται στο ότι σε τέτοιες περιπτώσεις όντως οι αρχικοποιήσεις αξίζουν τον κόπο γιατί γλιτώνουν τον αλγόριθμου από το να ξαναελέγξει πολλαπλές φορές τις ίδιες τιμές κάποιας μεταβλητής.

Βιβλιογραφία

[1] Roman Barták, Guide to Constraint Programming, 1998.

[2] Roman Barták, Guide to Constraint Programming, 1998.

[3] Christian Bessiere ,Constraint Propagation ,Technical Report LIRMM 06020 4.2.2,CNRS/University of Montpellier, March 2006

[4] A.K. Mackworth. Consistency in networks of relations. Technical Report 75-3, Dept. of Computer Science, Univ. of B.C. Vancouver, 1975. (also in Artificial Intelligence 8, 99-118, 1977).

[5] J.J. McGregor. Relational consistency algorithms and their application in_nding subgraph and graph isomorphism. Information Science, 19:229{250, 1979.

[6] R. Mohr and T.C. Henderson. Arc and path consistency revisited. Artificial Intelligence, 28:225{233, 1986.

[7] R. Mohr and G. Masini. Good old discrete relaxation. In Proceedings ECAI'88, pages 651{656, Munchen, FRG, 1988.

[8] Y. Zhang and R.H.C. Yap. Making AC-3 an optimal algorithm. In Proceedings IJCAI'01, pages 316{321, Seattle WA, 2001.