Σχεδίαση και Ανάπτυξη Συστήματος για Αξιολόγηση και Πειραματική Μελέτη Αλγορίθμων Υδατοσήμανσης Λογισμικού

Η ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης

του Τμήματος Πληροφορικής Εξεταστική Επιτροπή

από τον

Νικόλαο Βρεττό

ως μέρος των Υποχρεώσεων για τη λήψη του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ

ΣΤΗΝ ΘΕΩΡΙΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Ιούλιος 2012

Dedication

Αφιερώνω αυτή την εργασία στους γονείς μου για την συμπαράσταση και υπομονή που επέδειξαν καθ' όλη την διάρκεια αυτής της διατριβής. Επίσης, θα ήθελα να την αφιερώσω στα αδέρφια μου και την Ελευθερία χωρίς την συμπαράσταση των οποίων δεν θα τα είχα καταφέρει.

Ευχαριστιες

Η παρούσα εργασία πραγματοποιήθηκε υπό την επίβλεψη του καθηγητή Σταύρου Νικολόπουλου, του οποίου η συμβολή υπήρξε καθοριστική. Θα ήθελα να τον ευχαριστήσω για την αμέριστη συμπαράσταση και την επιμονή που επέδειξε καθ' όλη την διάρκεια αυτής της εργασίας καθώς επίσης και για την ουσιώδη συνεργασία μας, γεγονός που αποτέλεσε σημαντικό παράγοντα για την επιτυχή ολοκλήρωση της. Επίσης, θα ήθελα να ευχαριστήσω θερμά και τα άλλα δύο μέλη της τριμελούς επιτροπής, τον αναπληρωτή καθηγητή κύριο Λεωνίδα Παληό και τον επίκουρο καθηγητή κύριο Χρήστο Νομικό για τον χρόνο που διέθεσαν για την εξέταση αυτής της εργασίας.

Περιεχομενα

Ευρετηριο Σχηματων

Ευρετηριο Πινακων

Ευρετηριο Αλγοριθμων

Περιληψη

Νικόλαος Βρεττός του Γεωργίου και της Θεοδώρας. MSc, Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούλιος, 2012. Σχεδίαση και Ανάπτυξη Συστήματος για Αξιολόγηση και Πειραματική Μελέτη Αλγορίθμων Υδατοσήμανσης Λογισμικού. Επιβλέποντας: Σταύρος Δ. Νικολόπουλος.

Τα τελευταία χρόνια, με την ανάπτυξη του διαδικτύου έχει παρατηρηθεί ραγδαία αύξηση της καταπάτησης των προσωπικών δεδομένων. Όλο και περισσότεροι άνθρωποι πέφτουν θύματα τέτοιων περιπτώσεων. Χαρακτηριστικά παραδείγματα αποτελούν τόσο το μέσο κοινωνικής δικτύωσης Facebook μέσω του οποίου οποιοσδήποτε μπορεί να υποκλέψει προσωπικές πληροφορίες, όσο και τα διαφημιστικά μηνύματα που αποστέλλονται καθημερινά στο ηλεκτρονικό ταχυδρομείο μας.

Οι παραπάνω περιπτώσεις σε συνδυασμό με πλήθος άλλων παρόμοιων συμβάντων κέντρισε το ενδιαφέρον τόσο των ερευνητών όσο και το δικό μας ώστε να ασχοληθούμε με την μελέτη τεχνικών που μπορούν να αποτρέψουν αυτές τις ενέργειες.

Αντικείμενο της παρούσας διατριβής είναι η υλοποίηση αλγορίθμων που χρησιμοποιούνται στην τεχνική της υδατοσήμανσης λογισμικού για να μετατρέψουν την τιμή ενός υδατοσήμου σε γράφημα, το οποίο με τη σειρά του ενσωματώνεται στο λογισμικό. Επίσης, στόχος της διατριβής είναι η μελέτη της ανθεκτικότητας των γραφημάτων που παράγουν οι αλγόριθμοι εφαρμόζοντας μια σειρά διάφορων τύπων επιθέσεων.

Στα πλαίσια της μελέτης μας, αναπτύξαμε ένα σύστημα το οποίο υλοποιεί τους αλγορίθμους υδατοσήμανσης λογισμικού `CKCT` και `CN` που προτάθηκαν από τους C. Collberg, S. Kobourov, E. Carter, and C. Thomborson (2003) και τους M. Chroni and S.D. Nikolopoulos (2011), αντίστοιχα. Μέσω αυτού του συστήματος μελετάμε την αποτελεσματικότητα των ανωτέρω αλγορίθμων όπως επίσης και την ανθεκτικότητα σε επιθέσεις των αναγώγιμων μεταθετικών γραφημάτων που δημιουργούνται από αυτούς.

Extended Abstract in English

Nikolaos G. Vrettos., MSc, Computer Science Department, University of Ioannina, Greece.
July 2012, Design and Implementation of a System for Evaluating Software Watermarking Algorithms. Thesis Supervisor: Stavros D. Nikolopoulos.

In recent years, the development of the Internet has seen rapid growth of the abuse of personal data. More and more people fall victims of such cases. Representative examples of such cases are both the Facebook, where anyone can steal personal information and promotional messages, which are sent daily in our e-mail.

These cases in combination with many other similar incidents intrigued both researchers and us in order to deal with the study of techniques that can prevent these actions.

So the subject of this thesis is the implementation of the algorithms that are used in software watermarking technique to convert the value of a watermark on a graph, which in turn incorporated into the software. Also aim of this thesis is to study the resilience of graphs produced by algorithms by implementing a number of different types of attacks.

As part of our study, we developed a system that implements the algorithms `CKCT` and `CN` proposed by C. Collberg, S. Kobourov, E. Carter, and C. Thomborson (2003) and M. Chroni and S.D. Nikolopoulos (2011), respectively. Through this system we are studying the effectiveness of these algorithms and the resistance of the reducible permutation graphs, that these algorithms create, to attacks.

Κεφαλαιο 1

Εισαγωγη

1.1 Θέμα και Στόχος Διατριβής

Η συνεχώς αυξανόμενη ανάπτυξη του διαδικτύου και της τεχνολογίας έχουν οδηγήσει στην δημιουργία ενός εκ των σημαντικότερων προβλημάτων της εποχής μας, εκείνο της ασφάλειας των προσωπικών δεδομένων. Μια από τις κυριότερες κατηγορίες αυτού του προβλήματος είναι η πειρατεία λογισμικού, η οποία έχει απασχολήσει αρκετά τόσο την διεθνή κοινότητα όσο και την έρευνα, λόγω των οικονομικών επιπτώσεων που έχει επιφέρει.

Σύμφωνα με την Business Software Alliance (BSA), έναν μη κερδοσκοπικό οργανισμό του κλάδου της πληροφορικής που εστιάζει το ενδιαφέρον του στην καταπολέμηση της καταπάτησης των πνευματικών δικαιωμάτων, "ως πειρατεία ορίζεται η μη εξουσιοδοτημένη αντιγραφή ή η διανομή λογισμικού, η οποία πραγματοποιείται με την λήψη, αντιγραφή, κοινή χρήση, πώληση ή εγκατάσταση πολλαπλών αντιγράφων σε προσωπικούς ή εταιρικούς υπολογιστές". Μια πρόσφατη έρευνά της έδειξε ότι 7 στους 10 χρήστες υπολογιστών στην Ελλάδα έχουν αποκτήσει πειρατικό λογισμικό. Ειδικότερα, στο διάστημα 2010 - 2011 η πειρατεία αυξήθηκε κατά 2% αγγίζοντας το 61% του πληθυσμού, γεγονός που μας κατέταξε στην τρίτη θέση μεταξύ των χωρών που πάσχουν από αυτό το φαινόμενο.

Όμως, φαινόμενα πειρατείας δεν παρατηρούνται μόνο στη σφαίρα του λογισμικού, αλλά και σε μέσα που έχουν ιδιαίτερη ζήτηση στον κόσμο της πληροφορικής, λόγου χάριν σε εικόνες, βίντεο, μουσική και έγγραφα. Προκειμένου λοιπόν να αποφευχθούν τέτοιου είδους φαινόμενα που πλήττουν την οικονομία της χώρας, έχει κάνει την εμφάνιση της μια ιδιαίτερη τεχνική, εκείνη της υδατοσήμανσης. Με τον όρο αυτό εννοούμε την εισαγωγή μιας κρυφής πληροφορίας (υδατόσημο) σε ένα από τα προαναφερθέντα μέσα από τον νόμιμο κάτοχο των

πνευματικών δικαιωμάτων, με απώτερο σκοπό την προστασία του υλικού από παράνομη χρήση του.

Στην παρούσα διατριβή το ενδιαφέρον μας θα επικεντρωθεί στην τεχνική της υδατοσήμανσης που αφορά το λογισμικό. Συγκεκριμένα, θα γίνει χρήση της τεχνικής της υδατοσήμανσης λογισμικού που είναι βασισμένη σε γραφήματα (Graph-based Software Watermarking) η οποία έχει ως γενική λειτουργία την μετατροπή της τιμής ενός υδατοσήμου σε έναν συγκεκριμένο τύπο γραφήματος. Ειδικότερα, για την υλοποίηση της παραπάνω λειτουργίας χρησιμοποιούνται δύο αλγόριθμοι, ο `CKCT` που προτάθηκε από τον C. Collberg και τους συνεργάτες του και ο `CN` που προτάθηκε από την Μ. Χρόνη και τον Σ.Δ. Νικολόπουλο. Οι παραπάνω αλγόριθμοι μετατρέπουν την τιμή του υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα.

Ο συνολικός στόχος της εν λόγω διατριβής είναι η ανάπτυξη συστήματος για την πειραματική μελέτη της ανθεκτικότητας των δομών, που δημιουργούνται από τους αλγορίθμους που προαναφέρθηκαν, σε επιθέσεις αλλαγής κατεύθυνσης ακμής, προσθήκης/διαγραφής ακμής, αλλαγής προορισμού ακμής και προσθήκης/διαγραφής κόμβου.

1.2 Δομή της Διατριβής

Η διατριβή περιέχει 6 κεφάλαια. Στο Κεφάλαιο 2, παρουσιάζουμε μια επισκόπηση της έρευνας που έχει συντελεστεί έως τώρα στον τομέα της υδατοσήμανσης λογισμικού. Στη συνέχεια, στο Κεφάλαιο 3 παραθέτουμε τους αλγορίθμους τους οποίους θα υλοποιήσουμε και θα μελετήσουμε ως προς την ανθεκτικότητα τους σε επιθέσεις, ενώ στο Κεφάλαιο 4 περιγράφουμε την σχεδίαση και την υλοποίηση του συστήματος που περιέχει αυτούς τους αλγορίθμους. Τέλος, στο Κεφάλαιο 5 αναφέρουμε τα πειραματικά αποτελέσματα που προέκυψαν από την μελέτη των αλγορίθμων. Κλείνοντας την παρούσα διατριβή, στο Κεφάλαιο 6 παρουσιάζουμε τα συμπεράσματα που εξάγονται από την παραπάνω μελέτη.

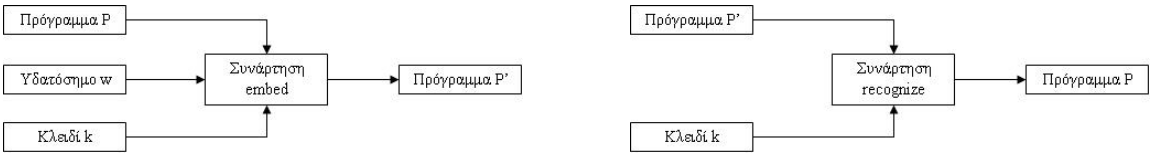
Κεφαλαιο 2

Υδατοσημανση Λογισμικου - Επισκοπηση

2.1 Εισαγωγή

Η υδατοσήμανση λογισμικού αποτελεί μια τεχνική μέσω της οποίας επιτυγχάνεται η εισαγωγή μιας κρυφής πληροφορίας (υπογραφής) στο λογισμικό, έτσι ώστε να είναι δυνατή η πιστοποίηση της επίσημης κατοχής των πνευματικών δικαιωμάτων του τελευταίου. Ειδικότερα, αυτή η κρυφή πληροφορία εισάγεται στον κώδικα του λογισμικού και εξάγεται σε κάθε περίπτωση που τίθεται ζήτημα πνευματικής ιδιοκτησίας.

Η διαδικασία της εισαγωγής και της εξαγωγής μπορεί εύκολα να περιγραφεί από τις παρακάτω συναρτήσεις: $embed(P, w, k) \rightarrow P'$ και $recognize(P', k) \rightarrow w$. Η συνάρτηση $embed$ δέχεται ως όρισμα το πρόγραμμα P που θέλουμε να προστατεύσουμε, το υδατόσημο w που θέλουμε να εισάγουμε στο P και ένα κλειδί k, το οποίο είναι γνωστό μόνο από τον κάτοχο των πνευματικών δικαιωμάτων του P. Ως αποτέλεσμα της συνάρτησης αυτής δίδεται το προστατευμένο πρόγραμμα P' (βλέπε Σχήμα 2.1(a)). Αντίθετα, η συνάρτηση $recognize$ υλοποιεί ακριβώς την αντίστροφη διαδικασία, δηλαδή δέχεται σαν όρισμα το προστατευμένο πρόγραμμα P' και το ίδιο κλειδί k με αυτό που χρησιμοποιήσαμε για να εισάγουμε το υδατόσημο. Το αποτέλεσμα αυτής της συνάρτησης είναι η εξαγωγή του υδατοσήμου w από το πρόγραμμα που υποπτευόμαστε ότι είναι παράνομο (βλέπε Σχήμα 2.1(b)).



(a) (b)

Σχήμα 2.1: (a) Εισαγωγή υδατοσήμου και (b) εξαγωγή υδατοσήμου

2.2 Κατηγοριοποίηση Αλγορίθμων Υδατοσήμανσης

Το πλήθος των υπαρκτών αλγορίθμων υδατοσήμανσης λογισμικού μπορεί να κατηγοριοποιηθεί με βάση τα ακόλουθα χαρακτηριστικά [18]:

- **Περιοχή αποθήκευσης υδατοσήμου στο πρόγραμμα:** Με βάση αυτό το χαρακτηριστικό τα υδατόσημα διακρίνονται σε δύο βασικές κατηγορίες, εκ των οποίων η πρώτη είναι τα *στατικά* και η δεύτερη τα *δυναμικά* υδατόσημα. Τα στατικά υδατόσημα τοποθετούνται είτε μέσα στο τμήμα του κώδικα του προγράμματος είτε μέσα στο τμήμα των δεδομένων και δεν αλλάζουν κατα την διάρκεια εκτέλεσης του προγράμματος. Αντίθετα, τα δυναμικά υδατόσημα δημιουργούνται κατα την διάρκεια εκτέλεσης του προγράμματος. Στην ουσία δεν εισάγεται ένα υδατόσημο αλλά κάποιοι κώδικες που εκφράζουν το υδατόσημο. Προκειμένου να ανακτηθεί ένα τέτοιο υδατόσημο αρκεί να αναλύσουμε τις δομές δεδομένων που δημιουργούνται όταν το υδατογραφημένο πρόγραμμα εκτελείται.

  Έτσι, μπορούμε τους αλγορίθμους υδατοσήμανσης να τους διακρίνουμε σε δύο βασικές κατηγορίες: τους στατικούς αλγορίθμους υδατοσήμανσης οι οποίοι εισάγουν στατικά ένα υδατόσημο και τους δυναμικούς αλγορίθμους υδατοσήμανσης που το εισάγουν δυναμικά.

- **Τύπο recognition συνάρτησης:** Η συνάρτηση recognizer έχει δύο μορφές ανάλογα με την πληροφορία που χρειάζεται για να αναγνωρίσει το υδατόσημο. Έτσι, ένα υδατόσημο και κατ' επέκταση ένας αλγόριθμος χαρακτηρίζεται ως *τυφλός (blind)* όταν η συνάρτηση recognizer απαιτεί μόνο το υδατογραφημένο πρόγραμμα P' και το κλειδί k. Αντίθετα, δίνουμε τον χαρακτηρισμό *ενημερωμένο (informed)* σε έναν αλγόριθμο όταν η συνάρτηση recognizer εκτός από το P' και το k απαιτεί και το αρχικό πρόγραμμα P και το αρχικά εισαχθέν υδατόσημο w.

- **Τεχνική εισαγωγής:** Προκειμένου να ενσωματωθεί το υδατόσημο στο πρόγραμμα αρκεί να προβούμε σε ορισμένους μετασχηματισμούς, οι οποίοι διακρίνονται σε τρείς μορφές: αναδιάταξη ή μετονομασία τμημάτων του κώδικα, εισαγωγή νέου κώδικα και τέλος χειρισμός στατιστικών ιδιοτήτων του προγράμματος, όπως συχνότητα εμφάνισης εντολών. Με βάση τους παραπάνω μετασχηματισμούς προκύπτουν και οι αντίστοιχοι αλγόριθμοι.

- **Τύπο υπογραφής:** Με βάση αυτό το χαρακτηριστικό διακρίνουμε δύο ειδών υπογραφές: υπογραφή πατρότητας (authorship mark) και υπογραφή δακτυλικού αποτυπώματος (fingerprint mark). Η υπογραφή πατρότητας χρησιμοποιείται για να γίνεται αντιληπτό ποιός είναι ο κάτοχος των πνευματικών δικαιωμάτων ενός προγράμματος. Αντίθετα, η υπογραφή δακτυλικού αποτυπώματος χρησιμοποιείται για να αναγνωρίζουμε ποιός είναι ο αγοραστής του προγράμματος. Έτσι, με βάση τα παραπάνω είναι εύκολο να διακρίνουμε τις δύο κατηγορίες αλγορίθμων που προκύπτουν ανάλογα με την υπογραφή που εισάγουν.

2.3 Σύζευξη Αλγορίθμων με το Γενικότερο Ιστορικό Πλαίσιο

Η τεχνική της υδατοσήμανσης λογισμικού, που αποτελεί το ουσιώδες αντικείμενο αυτής της διατριβής, έκανε την εμφάνιση της το 1990 με την εργασία του H. Tamada και των συνεργάτων του [23]. Λίγο αργότερα, το 1996 οι Davidson και Myhrvold [14] προέβησαν στη δημοσίευση του πρώτου αλγορίθμου υδατοσήμανσης λογισμικού. Προγενέστερα και συγκεκριμένα το 1954, μια άλλη τεχνική είχε αναπτυχθεί, αυτή των ψηφιακών υδατογραφημάτων η οποία υπήρξε και η αρχή για την ανάπτυξη της τεχνικής της υδατοσήμανσης λογισμικού. Στη συνέχεια, ορισμένες προκαταρκτικές έννοιες, σχετικές με την τεχνική αυτή, παρουσιάστηκαν από τους D. Grover [15], P. Samson [21], S. Moskowitz και M. Cooperman [16]. Όμως, δεν είχαν ακόμη παρουσιαστεί αναλυτικοί ορισμοί για την τεχνική αυτή, μέχρι τη στιγμή που ο C. Collberg και οι συνεργάτες του τούς διατύπωσαν [11, 12].

Μέχρι και σήμερα, οι στατικοί αλγόριθμοι υδατοσήμανσης λογισμικού είναι πολύ περισσότεροι σε σχέση με τους δυναμικούς, γεγονός που οφείλεται στην ευκολία των στατικών υδατοσήμων. Ένας αλγόριθμος που ανήκει σε αυτή την κατηγορία παρουσιάστηκε από τον Venkatesan και τους συνεργάτες του [24] ο οποίος μετατρέπει το υδατόσημο σε γράφημα και στη συνέχεια τό ενσωματώνει στο γράφημα ελέγχου ροής του προγράμματος. Επίσης, στην ίδια κατηγορία ανήκουν και οι αλγόριθμοι που παρουσίασε ο Stern και οι συνεργάτες του [22] καθώς επίσης και οι Qu και Potkonjak [19]. Ουσιαστικά, ο αλγόριθμος των πρώτων τροποποιούσε τη συχνότητα των εντολών έτσι ώστε να εισάγει το υδατόσημο, ενώ των τελευταίων έκανε χρήση του προβλήματος χρωματισμού των γραφημάτων για να πετύχει αυτή τη λειτουργία. Φυσικά το βασικό μειονέκτημα όλων των στατικών αλγορίθμων είναι ότι μπορεί το υδατόσημο να καταστραφεί κάνοντας βελτιστοποίηση στον κώδικα του προγράμματος. Στην κατηγορία των δυναμικών αλγορίθμων ανήκει ο αλγόριθμος του C. Collberg και του C. Thomborson [11]. Ήταν ο πρώτος δυναμικός αλγόριθμος που δημιουργήθηκε. Η λειτουργία του ήταν να εισάγει το υδατόσημο στη δομή του γραφήματος το οποίο δημιουργείται κατά την εκτέλεση του προγράμματος με συγκεκριμένη είσοδο.

Στην ενότητα που ακολουθεί θα παρουσιάσουμε μια αναλυτικότερη περιγραφή των αλγορίθμων που αναφέραμε καθώς επίσης και ένα διαχωρισμό αυτών ανάλογα με την τεχνική που ακολουθούν.

2.4 Αλγόριθμοι Υδατοσήμανσης

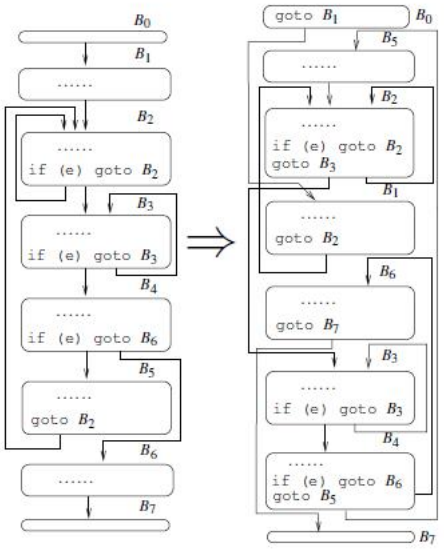
Όπως προαναφέρθηκε υπάρχει μεγάλο πλήθος αλγορίθμων για υδατοσήμανση σε λογισμικό. Όλοι όμως αυτοί οι αλγόριθμοι διαφέρουν μεταξύ τους ως προς την τεχνική στην οποία στηρίζονται. Ανάλογα με την τεχνική που χρησιμοποιούν οι αλγόριθμοι κατατάσσονται στις ακόλουθες κατηγορίες (βλέπε [8]):

- *Αλγόριθμοι αναδιάταξης βασικών block (Basic block reordering algorithms)*
- *Αλγόριθμοι κατανομής μητρώου (Register allocation algorithms)*
- *Αλγόριθμοι εκτεταμένου φάσματος (Spread-spectrum algorithms)*
- *Αδιαφανείς αλγόριθμοι κατηγορήματος (Opaque predicate algorithms)*
- *Αφηρημένος αλγόριθμος διερμηνευτή (Abstract interpretation algorithm)*
- *Αλγόριθμος δυναμικού μονοπατιού (Dynamic path algorithm)*
- *Αλγόριθμοι βασισμένοι σε γραφήματα (Graph-based algorithms)*

Στη συνέχεια, δίδεται μια λεπτομερέστερη περιγραφή αυτών των κατηγοριών τονίζοντας ιδιαίτερα τους αλγορίθμους που είναι βασισμένοι σε γραφήματα καθώς σε αυτούς θα εστιάσουμε το ενδιαφέρον μας στην παρούσα εργασία.

2.4.1 Αλγόριθμοι Αναδιάταξης Βασικών Block

Όπως προαναφέρθηκε, οι Davidson και Myhrvold παρουσίασαν τον πρώτο δημοσιευμένο στατικό αλγόριθμο υδατοσήμανσης λογισμικού. Στον συγκεκριμένο αλγόριθμο, η εισαγωγή του υδατοσήμου γίνεται αναδιατάσσοντας την σειρά των βασικών block του γραφήματος ελέγχου ροής του προγράμματος. Με λίγα λόγια, ο αλγόριθμος προχωρά σε αναδιάταξη τμημάτων εντολών του προγράμματος με συγκεκριμένη είσοδο και έξοδο. Φυσικά αυτή η αναδιάταξη συμβαίνει με τέτοιο τρόπο ώστε να μην αλλοιώνεται η λειτουργικότητα του αρχικού προγράμματος. Αυτή η αναδιάταξη είναι ουσιαστικά και το υδατόσημο που εισάγεται στο πρόγραμμα. Ένα παράδειγμα αυτής της τεχνικής παρουσιάζεται στο παρακάτω σχήμα.



Σχήμα 2.2: Παράδειγμα αναδιάταξης βασικών μπλόκ

Βασικό μειονέκτημα του εν λόγω αλγορίθμου είναι ότι δεν μπορούμε να εισάγουμε ένα δεύτερο υδατόσημο με την ίδια τεχνική καθώς η επαναδιάταξη των βασικών block θα προκαλούσε την καταστροφή του αρχικά εισαχθέντος υδατοσήμου.

2.4.2 Αλγόριθμοι Κατανομής Μητρώου

Με τον όρο κατανομή μητρώου εννοούμε την διαδικασία της ανάθεσης ενος μεγάλου αριθμού μεταβλητών του προγράμματος σε ένα μικρό πλήθος καταχωρητών. Η διαδικασία αυτή μπορεί να εφαρμοστεί πάνω σε ένα βασικό μπλόκ οπότε σε αυτή την περίπτωση έχουμε την τοπική κατανομή μητρώου (local register allocation), πάνω σε μια συνάρτηση ή διαδικασία οπότε έχουμε καθολική κατανομή μητρώου (global register allocation) ή μεταξύ συναρτήσεων λειτουργώντας ως σύμβαση κλήσης οπότε και έχουμε μεταξύ διαδικασιών κατανομή μητρώου (interprocedural register allocation).

Με αυτή την κατηγορία αλγορίθμων ασχολήθηκαν οι Qu και Potkonjak [19, 20] οι οποίοι παρουσίασαν έναν στατικό αλγόριθμο που υδατογραφεί την λύση του προβλήματος χρωματισμού ενος γραφήματος, δηλαδή εισάγει στο γράφημα ένα μήνυμα M. Ο αλγόριθμος αυτός είναι γνωστός με το όνομα QP. Το πρόβλημα του χρωματισμού απαιτεί την κατανομή όσο το δυνατόν λιγότερων χρωμάτων στους κόμβους ενος γραφήματος έτσι ώστε οι κόμβοι που συνδέονται με ακμή να μην έχουν το ίδιο χρώμα.

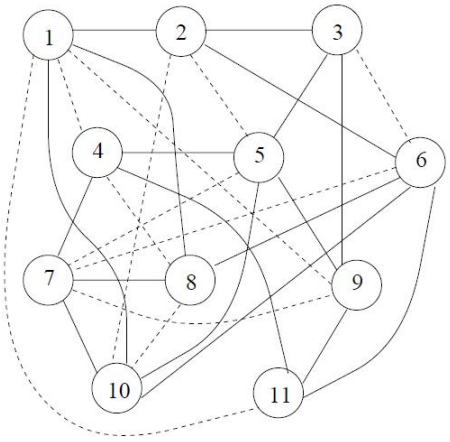
Αυτός ο αλγόριθμος εισάγει ένα υδατόσημο στην κατανομή μητρώου του προγράμματος μέσω μιας τεχνικής που ονομάζεται προσθήκη ακμών (adding-edges). Τα βήματα αυτής της τεχνικής παρουσιάζονται στον παρακάτω αλγόριθμο. Πρέπει να σημειωθεί ότι ως είσοδο σε αυτό τον αλγόριθμο δίνουμε το γράφημα G και την δυαδική αναπαράσταση του μηνύματος M.

Αλγόριθμος 1 Αλγόριθμος Προσθήκης Ακμών

1. Αντίγραψε το $G(V, E)$ στο $G'(V, E)$
2. Για κάθε bit m_i
   a. Βρες τους δύο πιο κοντινούς κόμβους u_{i_1} και u_{i_2} που δεν συνδέονται με τον κόμβο u_i
   b. Αν $m_i = 0$ πρόσθεσε την ακμή (u_i,u_{i_1}) στο E'
   c. Αλλιώς πρόσθεσε την ακμή (u_i,u_{i_2}) στο E'
3. Ανάφερε το G'

Το βήμα 2a του αλγορίθμου υπολογίζεται εφαρμόζοντας τα εξής: $i_2 > i_1 > i$, οι ακμές $(u_i, u_{i_1}), (u_i,u_{i_2}) \notin E'$ και $(u_i, u_j) \in E$ για όλα τα $i < j < i_1, i_1 < j < i_2$. Το ουσιώδες με αυτή την τεχνική είναι ότι προσθέτοντας μια ακμή μεταξύ δύο κόμβων με βάση το μήνυμα M, τότε αυτοί οι δύο κόμβοι πρέπει να χρωματιστούν με διαφορετικά χρώματα τα οποία μπορεί να μην υπάρχουν στο αρχικό γράφημα G.

Ένα παράδειγμα αυτής της τεχνικής παρουσιάζεται στο παρακάτω σχήμα. Το μήνυμα είναι το $1998_{10} = 11111001110_2$. Οι διακεκομένες ακμές είναι αυτές που προστίθενται με βάση την παραπάνω τεχνική.



Σχήμα 2.3: Παράδειγμα προσθήκης ακμών

Οι Zhu και Thomborson [25] παρουσίασαν μια πιο διευκρινισμένη έκδοση του παραπάνω αλγορίθμου. Παρατήρησαν ότι ένα βασικό ελάττωμα του Αλγορίθμου QP είναι ότι δεν είναι εξαγώγιμος καθώς είναι πιθανό να εισάγουμε δύο διαφορετικά μηνύματα και να πάρουμε το ίδιο υδατογραφημένο γράφημα.

Οι Myles και Collberg [17] πρότειναν μια νέα έκδοση του αλγορίθμου των Qu και Potkonjak. Αυτός ο αλγόριθμος είναι γνωστός με το όνομα QPS. Αποδείχτηκε όμως ότι και αυτός ο αλγόριθμος έχει μειονεκτήματα και το βασικότερο είναι ότι δεν είναι ανθεκτικός σε επιθέσεις.

2.4.3 Αλγόριθμοι Εκτεταμένου Φάσματος

Η υδατοσήμανση εκτεταμένου φάσματος αναπτύχθηκε κυρίως για την υδατοσήμανση ψηφιακών μέσων. Οι τεχνικές εκτεταμένου φάσματος είναι μέθοδοι με τις οποίες ένα σήμα που παράχθηκε με συγκεκριμένο εύρος ζώνης σκόπιμα διαχέεται στο πεδίο συχνοτήτων και αυτό έχει ως αποτέλεσμα ένα σήμα με ευρύτερο εύρος ζώνης.

Η τεχνική όμως εκτεταμένου φάσματος εφαρμόστηκε και για την υδατοσήμανση λογισμικού. Οι αλγόριθμοι που χρησιμοποιούν τέτοιες τεχνικές διαφέρουν από όλους τους άλλους καθώς βλέπουν τον κώδικα ως ένα ολόκληρο αντικείμενο και όχι ως μια ακολουθία εντολών. Τα υδατογραφημένα προγράμματα που προκύπτουν είναι πιο ανθεκτικά σε επιθέσεις καθώς το υδατόσημο τοποθετείται σε ολόκληρο το πρόγραμμα και όχι σε μια συγκεκριμένη θέση.

Ένας τέτοιος στατικός αλγόριθμος προτάθηκε από τον Stern και τους συνεργάτες του [22] ο οποίος τροποποιούσε την συχνότητα με την οποία εμφανίζονταν ομάδες εντολών έτσι ώστε να εισάγει το υδατόσημο.

2.4.4 Αδιαφανείς Αλγόριθμοι Κατηγορήματος

Ένα αδιαφανές κατηγόρημα είναι ένα κατηγόρημα, δηλαδή μια έκφραση που αξιολογείται είτε ως αληθής είτε ως ψευδής, για την οποία το αποτέλεσμα είναι γνωστό εκ των προτέρων. Είναι δύσκολο για την αυτόματη ανάλυση λογισμικού να βρεί την τιμή του κατηγορήματος.

Ένας στατικός αλγόριθμος αυτής της κατηγορίας προτάθηκε από τον Arboit [1]. Με αυτόν, τμήματα του υδατοσήμου κωδικοποιούνται ως σταθερές μέσα σε αδιαφανή κατηγορήματα. Για να εξάγουμε το υδατόσημο αρκεί να εξετάσουμε το πρόγραμμα για αδιαφανή κατηγορήματα και να τα αποκωδικοποιήσουμε έτσι ώστε να πάρουμε την τιμή τους.

2.4.5 Αφηρημένος Αλγόριθμος Διερμηνευτή

Ένας αλγόριθμος αυτής της κατηγορίας αναπτύχθηκε από τους P. Cousot και R. Cousot [13]. Γενικά, αυτός ο αλγόριθμος εισάγει το υδατόσημο σε ακέραιες τοπικά ορισμένες μεταβλητές. Συγκεκριμένα, η διαδικασία της εισαγωγής, έτσι όπως παρουσιάστηκε από τους Cousot είναι η εξής: αρχικά πρέπει να βρούμε μια συνάρτηση σε ολόκληρο το πρόγραμμα που να περιέχει επανάληψη. Εκεί μέσα θα εισάγουμε το υδατόσημο. Φυσικά, το υδατόσημο πρέπει να μην είναι μια σταθερά καθώς είναι ευάλωτο σε επιθέσεις. Οι Cousot πρότειναν το υδατόσημο να είναι ένα πολυώνυμο. Η επιλογή του πολυωνύμου πρέπει να γίνεται με τέτοιο τρόπο έτσι ώστε το υπόλοιπο της διαίρεσης του πολυωνύμου με το κλειδί να μας δίνει την τιμή του υδατοσήμου.

Ένα παράδειγμα της παραπάνω διαδικασίας είναι το ακόλουθο [10]: Έστω ότι το κλειδί είναι το 30001 και η τιμή του υδατοσήμου είναι το 21349. Επίσης, έστω ότι επιλέγουμε το δευτέρου βαθμού πολυώνυμο $P(x) = a * x^2 + b * x + c$ από το οποίο θέλουμε να βρούμε τους συντελεστές έτσι ώστε το πολυώνυμο να μας επιστρέφει την τιμή του υδατοσήμου:

$$W = a * W^2 + b * W + c \pmod{key} \tag{2.1}$$

όπου key είναι το κλειδί. Στη συνέχεια, διαλέγουμε δύο μικρές σταθερές για τα a και b και λύνουμε ως προς c την Εξίσωση 2.1. Αν επιλέξουμε a = 4 και b = 1566 τότε το c = 21494. Έτσι, στο Σχήμα 2.4 βλέπουμε πως θα γινόταν εισαγωγή το υδατόσημο μας.

```
int k = 1;
int i;
for (i = 1; i <= 20; i++){
      k = k * i;
}
```

(a)

```
int k = 1;
int i;
int w = -158657
for (i = 1; i <= 20; i++){
      k = k * i;
      w = w * (4 * w + 1566) + 21494;
}
```

(b)

Σχήμα 2.4: (a) Μη υδατογραφημένο πρόγραμμα και (b) Υδατογραφημένο πρόγραμμα

Μπορεί κάποιος εύκολα να παρατηρήσει ότι ο υπολογισμός του πολυωνύμου γίνεται μονο σε ένα σημείο στον κώδικα και αυτό δεν είναι και πολύ καλό. Έτσι, χωρίζουμε το πολυώνυμο σε μικρά τμήματα και καθένα από αυτά το τοποθετούμε μέσα στην επανάληψη. Επομένως, αν στο παραπάνω παράδειγμα εφαρμόσουμε αυτό, τότε προκύπτει ο υδατογραφημένος κώδικας του Σχήματος 2.5.

```
int k = 1;
int i;
int w = -158657
for (i = 1; i <= 20; i++){
      int g = w * 4;
      g = g + 1566;
      k = k * i;
      g = g * w;
      w = g + 21494;
}
```

Σχήμα 2.5: Τελικό υδατογραφημένο πρόγραμμα

2.4.6 Αλγόριθμος Δυναμικού Μονοπατιού

Οι αλγόριθμοι δυναμικού μονοπατιού εισάγουν το υδατόσημο στην εκτελέσιμη διακλαδωμένη δομή του προγράμματος. Οι αλγόριθμοι αυτοί στηρίχτηκαν στο ότι οι διακλαδώσεις που εκτελούνται από ένα πρόγραμμα παίζουν σημαντικό ρόλο στον υπολογισμό και επίσης στο ότι η εκτελέσιμη διακλαδωμένη δομή του προγράμματος είναι πολύ δύσκολο να αναλυθεί.

Ένας τέτοιος αλγόριθμος προτάθηκε από τον Collberg και τους συνεργάτες του. Ο αλγόριθμος αυτός καταρχάς καθορίζει, δίνοντας συγκεκριμένη είσοδο, την δυναμική συμπεριφορά του αρχικού προγράμματος παρακολουθώντας το μονοπάτι εκτέλεσης και εισάγει σε κατάλληλα σημεία το υδατόσημο αλλάζοντας την ακολουθία των διακλαδώσεων. Για την εξαγωγή του υδατοσήμου ξανατρέχουμε τον αλγόριθμο δίνοντας του σαν είσοδο αυτήν

που χρησιμοποιήσαμε για την εισαγωγή και ελέγχουμε την ακολουθία των διακλαδώσεων για να εξάγουμε το υδατόσημο. Επομένως, η είσοδος που θα δώσουμε για να εισάγουμε το υδατόσημο είναι στην ουσία και το κλειδί με το οποίο γίνεται αυτή η εισαγωγή.

2.4.7 Αλγόριθμοι Βασισμένοι σε Γραφήματα

Σε αυτή την κατηγορία αλγορίθμων ανήκουν οι αλγόριθμοι που μετατρέπουν την τιμή του υδατοσήμου σε γράφημα. Ο Collberg και οι συνεργάτες του έχουν περιγράψει αρκετούς τέτοιους αλγορίθμους κωδικοποίησης. Στην γενική περίπτωση αυτή η κωδικοποίηση γίνεται από μια συνάρτηση $encode(w) \rightarrow G$ η οποία μετατρέπει την τιμή του υδατοσήμου σε γράφημα. Αντίθετα, η αποκωδικοποίηση στην γενική περίπτωση πραγματοποιείται συνήθως από μια συνάρτηση $decode(G) \rightarrow w$ η οποία αποκωδικοποιεί το γράφημα και μας επιστρέφει την τιμή του υδατοσήμου. Πολλές φορές οι δύο αυτές συναρτήσεις μαζί αναφέρονται και ως *κωδικοποιητής*. Συνήθως χρησιμοποιούνται κλάσεις γραφημάτων και κωδικοποιητές που έχουν τις ακόλουθες ιδιότητες [9]:

- Κατάλληλους τύπους γραφημάτων: Τα γραφήματα που ανήκουν στην κλάση $\mathcal{G}$ πρέπει να είναι κατευθυνόμενα και να έχουν κόμβους με μικρό βαθμό εξερχόμενων ακμών έτσι ώστε να ταιριάζουν με τα γραφήματα των αληθινών προγραμμάτων.

- Υψηλή ανθεκτικότητα: Η συνάρτηση $decode(G)$ δεν πρέπει να είναι ευαίσθητη σε μικρές αλλαγές του γραφήματος G, δηλαδή σε περιπτώσεις προσθήκης ή διαγραφής ακμής και προσθήκης ή διαγραφής κόμβου. Επομένως, αν το $G' \approx G$, όπου G' το τροποποιημένο γράφημα και G το αρχικό γράφημα, τότε πρέπει να ισχύει το εξής: $decode(G') \rightarrow w$.

- Μικρό μέγεθος: έστω P_w το υδατογραφημένο πρόγραμμα και P το αρχικό. Τότε πρέπει να ισχύει ότι η διαφορά $\mid P_w \mid - \mid P \mid$ είναι μικρή.

- Αποτελεσματικούς κωδικοποιητές: οι δύο συναρτήσεις των κωδικοποιητών πρέπει να υλοποιούνται σε πολυωνυμικό χρόνο.

Συνήθως τα είδη γραφημάτων που χρησιμοποιούνται είναι τα προσανατολισμένα με δείκτη στον πατέρα δένδρα (oriented parent-pointer trees), με βάση k γραφήματα (radix-k graphs), τα μεταθετικά γραφήματα (permutation graphs), τα επίπεδα κυβικά δένδρα (planted planar cubic trees) και τέλος τα αναγώγιμα μεταθετικά γραφήματα (reducible permutation graphs). Αναλυτικότεροι ορισμοί γι' αυτά θα δοθούν στο Κεφάλαιο 3.

Ο πρώτος αλγόριθμος βασισμένος σε γραφήματα προτάθηκε από τον Venkatesan και τους συνεργάτες του. Οι τελευταίοι παρουσίασαν έναν στατικό αλγόριθμο που είναι γνωστός με το όνομα GTW και ο οποίος μετατρέπει την τιμή του υδατοσήμου σε γράφημα ελέγχου ροής. Στον Αλγόριθμο 2 παρουσιάζονται συνολικά τα βήματα για την εισαγωγή ενός υδατοσήμου με τον Αλγόριθμο GTW.

Για την εξαγωγή του υδατοσήμου απλά αποκωδικοποιούμε τα μαρκαρισμένα μπλόκ τους γραφήματος ελέγχου ροής του υδατογραφημένου προγράμματος.

Αλγόριθμος 2 Αλγόριθμος GTW

1. κωδικοποίησε την τιμή του υδατοσήμου σε γράφημα ελέγχου ροής
2. συνδύασε το γράφημα ελέγχου ροής του προγράμματος και το γράφημα του υδατοσήμου
3. σύνδεσε τα δύο γραφήματα προσθέτοντας ψευδής ακμές ελέγχου ροής
4. μαρκάρισε τα βασικά μπλόκ του γραφήματος ελέγχου ροής του υδατοσήμου

Για να είναι σταθερό το υδατογράφημα ο GTW Αλγόριθμος απαιτεί το γράφημα που αντιστοιχεί σε αυτό να μοιάζει όσο το δυνατόν περισσότερο στα γραφήματα ελέγχου ροής, δηλαδή πρέπει να έχει τις εξής ιδιότητες:

- να είναι αναγώγιμο,
- να έχει μικρό πλήθος εξερχόμενων ακμών,
- να είναι ισχνά.

Όπως αναφέρθηκε στην Ενότητα 2.3 ο πρώτος δυναμικός αλγόριθμος προτάθηκε από τους C. Collberg και του C. Thomborson [11]. Αυτός ο αλγόριθμος, γνωστός και με το όνομα CT, ανήκει στην κατηγορία των αλγορίθμων βασισμένων σε γραφήματα και εισάγει το υδατόσημο στό γράφημα που δημιουργείται κατά την εκτέλεση του προγράμματος. Τα βήματα της εισαγωγής αυτής είναι αυτά που παρουσιάζονται στον Αλγόριθμο 3.

Αλγόριθμος 3 Αλγόριθμος CT

1. κωδικοποίησε την τιμή του υδατοσήμου σε γράφημα G
2. χώρισε το G σε έναν αριθμό από υπογραφήματα
3. μετάτρεψε σε ακολουθία εντολών κάθενα από αυτά
4. εισάγαγε αυτές τις ακολουθίες κατα μήκος ενός ειδικού μονοπατιού εκτέλεσης του αρχικού προγράμματος

Από την άλλη για να εξάγουμε το υδατόσημο εκτελούμε το υδατογραφημένο πρόγραμμα με συγκεκριμένη είσοδο η οποία θα το αναγκάσει να ακολουθήσει το ειδικό μονοπάτι εκτέλεσης στο οποίο εισαγάγαμε το υδατόσημο. Έτσι, με αυτό τον τρόπο θα εκτελέσουμε την αποκωδικοποίηση και θα πάρουμε το υδατόσημο.

Φυσικά όπως και ο GTW έτσι και ο CT έχει μια βασική απαίτηση για να είναι σταθερό το υδατόσημο. Αυτή είναι το γράφημα του υδατοσήμου να μοιάζει με πραγματική δυναμική δομή δεδομένων. Αυτό επιτυγχάνεται αν διατηρήσουμε τις ακόλουθες ιδιότητες:

- να έχει μικρό πλήθος εξερχόμενων ακμών,
- να έχει μοναδικό κόμβο ρίζα από τον οποίο οποιοσδήποτε άλλος κόμβος είναι προσβάσιμος.

2.5 Κατηγορίες Επιθέσεων

Στις μέρες μας υπάρχει πλήθος αλγορίθμων οι οποίοι έχουν σκοπό να μειώσουν τα φαινόμενα παράνομης διακίνησης προγραμμάτων. Όμως όπως φαίνεται δεν έχουμε πετύχει το βέλτιστο αποτέλεσμα καθώς κάθε μέρα τα φαινόμενα αυτά όλο και αυξάνονται. Εκτός των ατόμων που ασχολούνται με την ανάπτυξη αλγορίθμων προστασίας τών προγραμμάτων υπάρχουν και άτομα που εξειδικεύονται στο πώς να κλέβουν προγράμματα και να τα διακινούν είτε ώς ελεύθερα, είτε ως δικά τους. Το να κλέψουν ένα πρόγραμμα το επιτυγχάνουν εφαρμόζοντας ένα σύνολο επιθέσεων στο υδατογραφημένο πρόγραμμα. Τα είδη αυτών των επιθέσεων είναι τα ακόλουθα [3]:

- Προσθετική επίθεση: Σε αυτή την περίπτωση ο εισβολέας εισάγει ένα δικό του υδατόσημο στο ήδη υδατογραφημένο γράφημα με σκοπό να μην μπορεί ο πραγματικός κάτοχος των πνευματικών δικαιωμάτων να αποδείξει ότι το πρόγραμμα του ανήκει.

- Αφαιρετική επίθεση: Με αυτή την επίθεση ο εισβολέας αφαιρεί το υδατόσημο από το πρόγραμμα χωρίς όμως να επηρεάσει την λειτουργικότητα του.

- Στρεβλωτική επίθεση: Εδώ ο εισβολέας τροποποιεί το υδατόσημο με σκοπό να μην είναι δυνατή η εξαγωγή του από τον πραγματικό κάτοχο.

- Επίθεση αναγνώρισης: Με αυτή την επίθεση τροποποιείται ή απενεργοποιείται ο ανιχνευτής του υδατοσήμου ή τα δεδομένα που παίρνει ως είσοδο με σκοπό να επιστρέφει παραπλανητικά αποτελέσματα.

Τέλος, σε αλγορίθμους βασισμένους σε γραφήματα οι επιθέσεις που συμβαίνουν είναι συνήθως αντιστροφή ακμής, προσθήκη ή διαγραφή ακμής και προσθήκη ή διαγραφή κόμβου. Τις επιθέσεις αυτές θα τις μελετήσουμε αναλυτικότερα στα κεφάλαια που ακολουθούν.

Κεφαλαιο 3

Υδατοσημανση Λογισμικου Βασισμενη Σε Γραφηματα

3.1 Εισαγωγή

Στην Ενότητα 2.4 αναφερθήκαμε στις βασικότερες κατηγορίες αλγορίθμων υδατοσήμανσης λογισμικού και παρουσιάσαμε μερικούς από αυτούς. Η παρούσα διατριβή πραγματεύεται θέματα που αφορούν την κατηγορία αλγορίθμων βασισμένη σε γραφήματα. Συγκεκριμένα, θα ασχοληθούμε με αλγορίθμους κωδικοποίησης μιας τιμής ενός υδατοσήμου σε γράφημα καθώς επίσης και με τους αντίστοιχους αλγορίθμους αποκωδικοποίησης.

Έτσι, στην Ενότητα 3.2 αυτού του κεφαλαίου θα αναφερθούν ορισμένες βασικές έννοιες που είναι απαραίτητες για την καλύτερη κατανόηση των αλγορίθμων κωδικοποίησης και αποκωδικοποίησης που θα παρουσιάσουμε. Επίσης, στην Ενότητα 3.3 θα γίνει μια αναλυτική περιγραφή των τύπων γραφημάτων που χρησιμοποιούνται στην υδατοσήμανση λογισμικού και οι οποίοι αναφέρθηκαν ονομαστικά στην Ενότητα 2.5. Τέλος, στις Ενότητες 3.4 και 3.5 θα παρουσιαστούν δύο αλγόριθμοι κωδικοποίησης και αποκωδικοποίησης οι οποίοι προτάθηκαν από τον C. Collberg και τους συνεργάτες του [9] και τους Μ. Χρόνη και Σ.Δ. Νικολόπουλο [3].

3.2 Βασικές Έννοιες

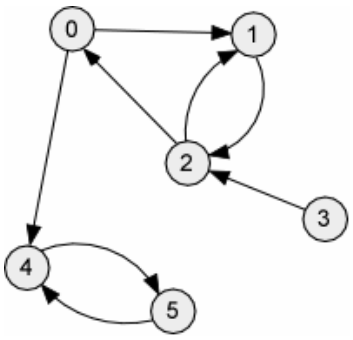
Στην παρούσα παράγραφο θα παρουσιάσουμε κάποιες βασικές έννοιες οι οποίες αποτελούν το υπόβαθρο που πρέπει να κατέχει όποιος ασχολείται με υδατοσήμανση λογισμικού που βασίζεται σε γραφήματα. Επίσης, θα παρουσιάσουμε κάποιους ορισμούς που είναι απαραίτητοι για τους αλγορίθμους που θα αναλύσουμε στις Παραγράφους 3.4 και 3.5.

Ορισμός 3.1 (Κατευθυνόμενο Γράφημα). Με τον όρο κατευθυνόμενο γράφημα εννοούμε ένα ζευγάρι $G = (V, E)$ όπου:

- το σύνολο V περιέχει το σύνολο των κόμβων (κορυφών)
- και το σύνολο E περιέχει διατεταγμένα ζεύγη κορυφών, τα οποία καλούνται τόξα ή κεταυθυνόμενες ακμές.

Διαφέρει από έναν μη κατευθυνόμενο γράφο στο ότι ο τελευταίος αποτελείται από ένα σύνολο μη διατεταγμένων ζευγών από κορυφές, οι οποίες ονομάζονται συνήθως ακμές.

Στο Σχήμα 3.1 παρουσιάζουμε ένα παράδειγμα ενός κατευθυνόμενου γραφήματος με έξι κόμβους και οκτώ ακμές.



Σχήμα 3.1: Κατευθυνόμενο γράφημα

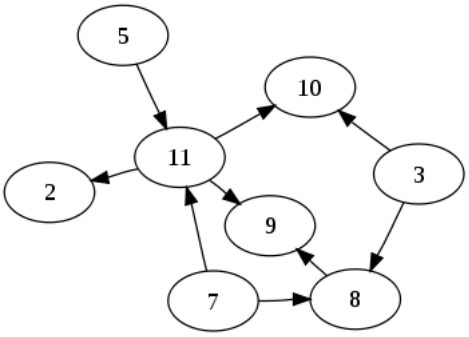
Ορισμός 3.2 (Μονοπάτι Γραφήματος). Μονοπάτι σε ένα γράφημα $G = (V, E)$ (κατευθυνόμενο και μή) ονομάζουμε μια ακολουθία από κορυφές αυτού έτσι ώστε κάθε κορυφή αυτής να έχει ακμή e στην επόμενη κορυφή στην ακολουθία ($e \in E$). Ένα μονοπάτι στο οποίο δεν επαναλαμβάνονται ακμές ονομάζεται απλό (simple). Ένα μονοπάτι (απλό μονοπάτι) αποτελεί κύκλο αν υπάρχει ακμή που να συνδέει τον αρχικό και τον τελικό κόμβο του στο E.

Ορισμός 3.3 (Κατευθυνόμενο Άκυκλο Γράφημα (DAG)). Ένα κατευθυνόμενο άκυκλο γράφημα είναι ένα κατευθυνόμενο γράφημα στο οποίο δεν υπάρχουν κατευθυνόμενοι κύκλοι.

Το γράφημα του Σχήματος 3.1 δεν είναι άκυκλο καθώς υπάρχει ο εξής κύκλος: από τον κόμβο μηδέν στον κόμβο ένα, από τον κόμβο ένα στον κόμβο δύο και τέλος από τον

κόμβο δύο πίσω ξανά στον κόμβο μηδέν. Φυσικά σε αυτό το γράφημα υπάρχουν και άλλοι κατευθυνόμενοι κύκλοι.

Ένα γράφημα που ανήκει στα κατευθυνόμενα άκυκλα γραφήματα είναι αυτό της Σχήματος 3.2.



Σχήμα 3.2: Κατευθυνόμενο άκυκλο γράφημα

Ορισμός 3.4 (Γράφημα Ροής (Flow Graph)). Με τον όρο γράφημα ροής εννοούμε ένα κατευθυνόμενο γράφημα του οποίου κάθε κόμβος είναι προσβάσιμος από έναν αρχικό κόμβο.

Πάνω στα γραφήματα ελέγχου ροής ορίζεται μια σχέση κυριαρχίας (domination relationship) η οποία στην ουσία μας λέει πότε ένας κόμβος κυριαρχεί (dominates) σε έναν άλλο κόμβο. Ο ορισμός της σχέσης κυριαρχίας είναι ο ακόλουθος.

Ορισμός 3.5 (Σχέση Κυριαρχίας). Σε ένα γράφημα ελέγχου ροής ένας κόμβος a κυριαρχεί σε έναν κόμβο b αν κάθε μονοπάτι από τον κόμβο αφετηρία προς τον κόμβο b περνάει από τον κόμβο a. Συνήθως αυτό γράφεται και ως εξής: a dom b.

Ορισμός 3.6 (Αναζήτηση Κατά Βάθος (DFS)). Η αναζήτηση κατά βάθος είναι ένας αλγόριθμος που χρησιμοποιείται για την διάσχιση ή την αναζήτηση σε ένα γράφημα. Αυτός ο αλγόριθμος ξεκινά από έναν αρχικό κόμβο και διερευνά κόμβους που βρίσκονται σε όσο γίνεται μεγαλύτερη απόσταση από τον αρχικό. Η διαδικασία σταματά όταν διερευνηθούν όλοι οι κόμβοι.

Μέχρι στιγμής έχουμε αναφέρει αναλυτικούς ορισμούς που αφορούν γενικά τα γραφήματα. Από εδώ και πέρα θα γίνει αναφορά σε ορισμούς που αφορούν του αλγορίθμους που θα παρουσιάσουμε.

Και οι δύο αλγόριθμοι χρησιμοποιούν αυτο-αντιστρέψιμες μεταθέσεις. Υποθέτουμε ότι π είναι μια μετάθεση πάνω στο σύνολο 1, 2, ..., n. Μπορούμε να θεωρήσουμε το π ως μια ακολουθία $[\pi_1, \pi_2, ..., \pi_n]$. Συμβολίζουμε με π_i^{-1} την θέση του αριθμού i στην ακολουθία αυτή. Η αντίστροφη της μετάθεσης $[\pi_1, \pi_2, ..., \pi_n]$ είναι η μετάθεση $[q_1, q_2, ..., q_n]$ με $q_{p_i} = \pi_{q_i} = i$ [3]. Με βάση αυτό προκύπτει ο εξής βασικός ορισμός:

Ορισμός 3.7 (Αυτο-αντιστρέψιμη Μετάθεση (Self-inverting Permutation)). Μια αυτο-αντιστρέψιμη μετάθεση είναι μια μετάθεση που είναι από μόνη της αντιστρέψιμη, δηλαδή $\pi_{\pi_i} = i$. Άρα κάθε μετάθεση έχει μοναδική αντιστροφή, και αν πάρουμε την αντίστροφη

μιας αντίστροφης μετάθεσης τότε θα προκύψει η αρχική μετάθεση. Επομένως, αυτομάτως προκύπτει ότι μια μετάθεση είναι αυτο-αντιστρέψιμη αν και μόνο αν έχει κύκλους μεγέθους 1 ή 2.

Για παράδειγμα, αν πάρουμε την μετάθεση [3, 4, 1, 2, 5, 7, 6] προκύπτουν οι κύκλοι (1,3), (2,4), (5) και (6,7) όπως φαίνεται και στο Σχήμα 3.3. Άρα, εφόσον όλοι οι κύκλοι που υπάρχουν είναι μεγέθους 1 ή 2 συμπεραίνουμε ότι αυτή η μετάθεση είναι αυτο-αντιστρέψιμη.

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 4 & 1 & 2 & 5 & 7 & 6 \end{pmatrix} \Longrightarrow (1,3), (2,4), (5), (6,7)$$

Σχήμα 3.3: Κύκλοι μετάθεσης [3, 4, 1, 2, 5, 7, 6]

Ορισμός 3.8 (Αύξουσα Αναπαράσταση Κύκλων (Increasing Cycle Representation)). Υποθέτουμε ότι συμβολίζουμε με $C_{1,2}$ το σύνολο όλων των κύκλων μιας αυτο-αντιστρέψιμης μετάθεσης, δηλαδή $C_{1,2} = \{c_1 = (x_1, y_1), c_2 = (x_2, y_2), ..., c_k = (x_k, y_k)\}$. Έστω $\prec$ να είναι η γραμμική διάταξη πάνω σε αυτό το σύνολο έτσι ώστε να ισχύει $c_i \prec c_j$ αν $x_i < x_j$ για κάθε $1 \leq i, j \leq k$. Ονομάζουμε αυτό το σύνολο αύξουσα αναπαράσταση των κύκλων της αυτο-αντιστρέψιμης μετάθεσης αν ισχύει $c_1 \prec c_2 \prec ... \prec c_k$ [3].

3.3 Κλάσεις Γραφημάτων Υδατοσήμανσης

Όπως αναφέραμε στην Παράγραφο 2.4.7 οι αλγόριθμοι υδατοσήμανσης βασισμένοι σε γραφήματα μετατρέπουν την τιμή ενός υδατοσήμου σε γράφημα και στη συνέχεια εισάγουν αυτό το γράφημα μέσα στο πρόγραμμα. Σε αυτή την παράγραφο θα αναλύσουμε τις βασικές κλάσεις γραφημάτων που χρησιμοποιούνται για την κωδικοποίηση της τιμής ενός υδατοσήμου σε γράφημα και θα αναφέρουμε τα πλεονεκτήματα και τα μειονεκτήματα αυτών των κλάσεων. Οι κλάσεις γραφημάτων που θα αναλύσουμε είναι τα προσανατολισμένα με δείκτη στον πατέρα δένδρα (oriented parent-pointer trees), τα γραφήματα με βάση k (radix-k graphs), τα μεταθετικά γραφήματα (permutation graphs), τα επίπεδα κυβικά δένδρα (planted planar cubic trees) και τέλος τα αναγώγιμα μεταθετικά γραφήματα (reducible permutation graphs).

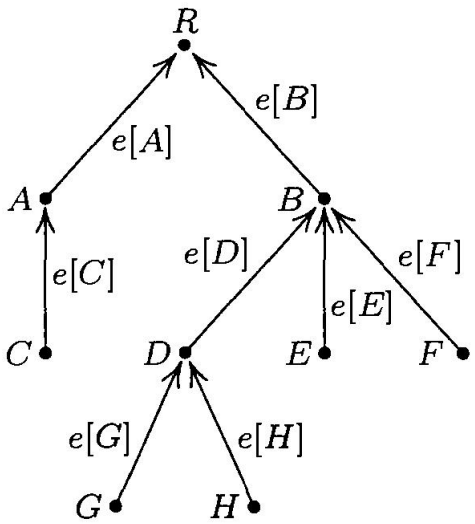
3.3.1 Δένδρα Προσανατολισμένα με Δείκτη στον Πατέρα

Ορισμός 3.9. Ένα προσανατολισμένο (oriented) δένδρο, μερικές φορές καλείται και ριζομένο (rooted) δένδρο, είναι ένα κατευθυνόμενο γράφημα με έναν συγκεκριμένο κόμβο R έτσι ώστε:

- κάθε κόμβος $V \neq R$ είναι ο αρχικός κόμβος από ακριβώς μια ακμή, ορίζεται ως $e[V]$
- ο R είναι ο αρχικός κόμβος από καμία ακμή

- ο R είναι μια ρίζα καθώς για κάθε κόμβο $V \neq R$ υπάρχει ένα κατευθυνόμενο μονοπάτι από τον V στον R.

Ένα προσανατολισμένο δένδρο φαίνεται στο ακόλουθο σχήμα:



Σχήμα 3.4: Προσανατολισμένο Δένδρο

Εξ ορισμού προκύπτει ότι για κάθε κόμβο $V \neq R$ υπάρχει ένα μοναδικό κατευθυνόμενο μονοπάτι από τον V στον R, και ως εκ τούτου δεν υπάρχουν κατευθυνόμενοι κύκλοι.
Για να απαριθμήσουμε αυτά τα δένδρα πρέπει να γνωρίζουμε τον αριθμό των δένδρων για ένα συγκεκριμένο πλήθος κόμβων καθώς επίσης και τον τρόπο δημιουργίας τους. Ο Knuth παρουσίασε μια φόρμουλα που μας δίνει τον αριθμό των διαφορετικών δένδρων. Αυτή η φόρμουλα είναι η εξής:

$$A(z) = z+z^2+2*z^3+4*z^4+9*z^5+20*z^6+48*z^7+115*z^8+286*z^9+719*z^{10}+1842*z^{11}+...$$

Σε αυτή τη φόρμουλα οι εκθέτες των z είναι το πλήθος των κόμβων και ο συντελεστής κάθε z είναι ο αριθμός των διαφορετικών προσανατολισμένων με δείκτη στον πατέρα δένδρων, δηλαδή με 5 κόμβους έχουμε 9 τέτοια δένδρα, με 6 κόμβους έχουμε 20 τέτοια δένδρα κ.ο.κ.
Τέλος, μπορούμε να κατασκευάζουμε αυτά τα δένδρα δημιουργώντας κάθε φορά αυτό με το μεγαλύτερο υποδένδρο.
Σύμφωνα με τον ορισμό αυτών των γραφημάτων προκύπτουν τα εξής πλεονεκτήματα:

- μια επίθεση προσθήκης ή διαγραφής ακμής γίνεται εύκολα αντιληπτή καθώς είτε κάποιος κόμβος θα έχει δύο εξερχόμενες ακμές, το οποίο εξ ορισμού δεν μπορεί να ισχύει, είτε θα πάψει να είναι το γράφημα συνεκτικό,

- ο χρόνος δημιουργίας τέτοιων γραφημάτων είναι πολυωνυμικός.

Φυσικά μπορούμε να παρατηρήσουμε ότι με ελάχιστες επιθέσεις (ακόμα και με μια επίθεση) μπορεί αντί του αρχικού γραφήματος, το οποίο αντιστοιχεί σε ένα συγκεκριμένο υδατόσημο,

να προκύψει ένα άλλο γράφημα το οποίο θα είναι προσανατολισμένο με δείκτη στον πατέρα δένδρο και το οποίο θα αντιστοιχεί σε άλλο υδατόσημο διαφορετικό του αρχικού. Αυτό που μόλις αναφέρθηκε αποτελεί και το βασικό μειονέκτημα αυτής της κλάσης γραφημάτων.

3.3.2 Γραφήματα με Βάση k

Τα γραφήματα αυτής της κατηγορίας αποτελούνται από μια κυκλικά συνδεδεμένη λίστα μεγέθους k, έναν αρχικό κόμβο που δείχνει στο πρώτο στοιχείο της λίστας και κάποιες ακμές που προκύπτουν από την μετατροπή της τιμής του υδατοσήμου στον αντίστοιχο αριθμό με βάση το k. Η διαδικασία με την οποία κατασκευάζουμε τέτοιου είδους γραφήματα παρουσιάζεται στον παρακάτω αλγόριθμο.

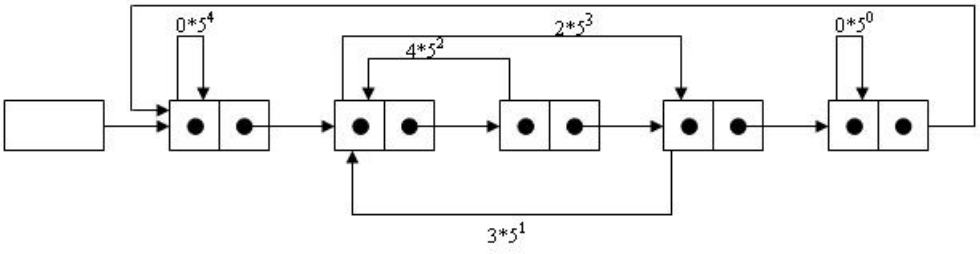
Αλγόριθμος 4 Αλγόριθμος Δημιουργίας Γραφημάτων με Βάση k

1. Φτιάξε την αναπαράσταση της τιμής του υδατοσήμου με βάση το k
2. Φτιάξε μια κυκλικά συνδεδεμένη λίστα με μήκος k. Κάθε κόμβος της λίστας έχει δύο πεδία, ένα που δείχνει στον επόμενο κόμβο στην κυκλική λίστα και άλλο ένα που χρησιμοποιείται για την αναπαράσταση του με βάση k ψηφίου κάθε κόμβου
3. Στην αναπαράσταση που προέκυψε από το βήμα 1 ο κάθε εκθέτης είναι ο αντίστοιχος κόμβος στην κυκλικά συνδεδεμένη λίστα και ο κάθε συντελεστής είναι το πλήθος των κόμβων που πρέπει να προσπελαστούν έτσι ώστε να βρεθεί ο κόμβος στον οποίο θα πρέπει να δείχνει το δεύτερο πεδίο κάθε κόμβου. Εισάγαγε έναν επιπλέον κόμβο ο οποίος να δείχνει στον τελευταίο κόμβο (αυτόν με τον μεγαλύτερο εκθέτη στην αναπαράσταση)

Για την καλύτερη κατανόηση του παραπάνω αλγορίθμου παραθέτουμε το παράδειγμα δημιουργίας του γραφήματος με βάση 5 του αριθμού 365. Η ανάλυση του αριθμού αυτού με βάση το 5 είναι η εξής:

$$365 = 0 * 5^4 + 2 * 5^3 + 4 * 5^2 + 3 * 5^1 + 0 * 5^0$$

Το γράφημα που προκύπτει από τα βήματα 2 και 3 του αλγορίθμου για την τιμή 365 είναι το εξής:



Σχήμα 3.5: Radix-5 του Αριθμού 365

Από τον τρόπο κατασκευής αυτών των γραφημάτων παρατηρούμε τα παρακάτω πλεονεκτήματα:

- επιθέσεις αλλαγής κατεύθυνσης και προσθήκης/διαγραφής ακμής γίνονται εύκολα αντιληπτές καθώς σε κάθε κόμβο το πλήθος των εξερχόμενων ακμών είναι δύο,
- επιθέσεις εισαγωγής και διαγραφής κόμβων γίνονται εύκολα αντιληπτές καθώς το πλήθος τους είναι ίσο με τον αριθμό k με τον οποίο γίνεται η κωδικοποίηση.

Παρ' όλα τα πλεονεκτήματα αυτών των γραφημάτων μπορεί εύκολα κάποιος να παρατηρήσει ότι δεν γίνονται εύκολα αντιληπτές περιπτώσεις επιθέσεων αλλαγής του προορισμού των ακμών. Επομένως, βασικό μειονέκτημα αυτών των γραφημάτων είναι ότι με ελάχιστες αλλαγές προκύπτει πάλι ένα γράφημα με βάση k το οποίο όμως αντιστοιχεί σε διαφορετική τιμή υδατοσήμου από εκείνη που είχαμε αρχικά.

3.3.3 Μεταθετικά Γραφήματα

Ορισμός 3.10. Ένα μη κατευθυνόμενο γράφημα G με κόμβους που αριθμούνται από το 1 στο n, δηλαδή $V(G) = \{1, 2, ..., n\}$, ονομάζεται μεταθετικό γράφημα αν υπάρχει μια μετάθεση $\pi = (\pi_1, \pi_2, ..., \pi_n)$ στο N τέτοια ώστε $(i, j) \in E$ αν και μόνο αν $(i - j)(\pi_i^{-}1 - \pi_j^{-}1) < 0$.

Όπως και στα γραφήματα με βάση k έτσι και στα μεταθετικά γραφήματα γίνεται χρήση μιας κυκλικά συνδεδεμένης λιστας. Η δημιουργία μεταθετικών γραφημάτων χωρίζεται σε δύο φάσεις. Στην πρώτη μετατρέπεται η τιμή του υδατοσήμου σε μετάθεση ενω στη δεύτερη μετατρέπεται η μετάθεση σε μεταθετικό γράφημα. Οι δύο αλγόριθμοι που ακολουθούν χρησιμοποιούνται για την μετατροπή της τιμής ενός υδατοσήμου σε μετάθεση και το αντίστροφο [10].

Αλγόριθμος 5 Αλγόριθμος Κωδικοποίησης Ακεραίου σε Μετάθεση

1. $perm =< 0, 1, 2, ..., len - 1 >$
2. Για$(r = 2; r <= len; r++)$
   αντιμετάθεσε τα $perm[r - 1]$ και $perm[V\%r]$
   $V = V/r$
3. Επέστρεψε $perm$

Αλγόριθμος 6 Αλγόριθμος Αποκωδικοποίησης Μετάθεσης σε Ακέραιο

1. $V = 0$
2. $f = 0$
3. Για$(r = len; r >= 2; r--)$
   Για$(s = 0; s < r; s++)$
   Αν $(perm[s] == r - 1)$
   $f = s$
   break;
   Αντιμετάθεσε τα $perm[r-1]$ και $perm[f]$
   $V = f + r * V$
4. Επέστρεψε V

Στον αλγόριθμο κωδικοποίησης ενός ακεραίου σε μετάθεση δίνεται ως είσοδο ο ακέραιος V καθώς επίσης και η τιμή του len το οποίο πρέπει να είναι τουλάχιστον n, όπου $V < n!$. Αντίθετα, ο αλγόριθμος αποκωδικοποίησης μιας μετάθεσης δέχεται ως είσοδο τη μετάθεση $perm$ και το μήκος len της μετάθεσης. Για την καλύτερη κατανόηση αυτών των αλγορίθμων παραθέτουμε ένα παράδειγμα μετατροπής του αριθμού 1024 σε μετάθεση καθώς επίσης και την αποκωδικοποίηση αυτής της μετάθεσης στην τιμή που αντιστοιχεί.

Αλγόριθμος Κωδικοποίησης Ακεραίου σε Μετάθεση

Για $n = 7$ ισχύει ότι $V < n!$. Άρα ο αλγόριθμος εργάζεται ως εξής:
$V = 1024$
$perm =< 0, 1, 2, 3, 4, 5, 6 >$
$r = 2$:
αντιμετάθεσε τα $perm[1]$ και $perm[0] \rightarrow < 1, 0, 2, 3, 4, 5, 6 >$.
$V = 512$.
$r = 3$:
αντιμετάθεσε τα $perm[2]$ και $perm[2] \rightarrow < 1, 0, 2, 3, 4, 5, 6 >$.
$V = 170$.
$r = 4$:
αντιμετάθεσε τα $perm[3]$ και $perm[2] \rightarrow < 1, 0, 3, 2, 4, 5, 6 >$.
$V = 42$.
$r = 5$:
αντιμετάθεσε τα $perm[4]$ και $perm[2] \rightarrow < 1, 0, 4, 2, 3, 5, 6 >$.
$V = 8$.
$r = 6$:
αντιμετάθεσε τα $perm[5]$ και $perm[2] \rightarrow < 1, 0, 5, 2, 3, 4, 6 >$.
$V = 1$.

$r = 7$:

αντιμετάθεσε τα $perm[6]$ και $perm[1] \rightarrow < 1, 6, 5, 2, 3, 4, 0 >$.

$V = 0$.

Επέστρεψε $perm =< 1, 6, 5, 2, 3, 4, 0 >$

Αλγόριθμος Αποκωδικοποίησης Μετάθεσης σε Ακέραιο

Δίνουμε ως είσοδο την μετάθεση $perm =< 1, 6, 5, 2, 3, 4, 0 >$ και το $len = 7$.
Ο αλγόριθμος υπολογίζει τα εξής:
$V = 0$
$f = 0$
$r = 7$:

$s = 0$:

$perm[0] == 6$ ΨΕΥΔΗΣ

$s = 1$:

$perm[1] == 6$ ΑΛΗΘΗΣ

$f = 1$

αντιμετάθεσε τα $perm[6]$ και $perm[1] \rightarrow < 1, 0, 5, 2, 3, 4, 6 >$

$V = 1 + 7 * 0 = 1$

$r = 6$:

$s = 0$:

$perm[0] == 5$ ΨΕΥΔΗΣ

$s = 1$:

$perm[1] == 5$ ΨΕΥΔΗΣ

$s = 2$:

$perm[2] == 5$ ΑΛΗΘΗΣ

$f = 2$

αντιμετάθεσε τα $perm[5]$ και $perm[2] \rightarrow < 1, 0, 4, 2, 3, 5, 6 >$

$V = 2 + 6 * 1 = 8$

$r = 5$:

$s = 0$:

$perm[0] == 4$ ΨΕΥΔΗΣ

$s = 1$:

$perm[1] == 4$ ΨΕΥΔΗΣ

$s = 2$:

$perm[2] == 4$ ΑΛΗΘΗΣ

$f = 2$

αντιμετάθεσε τα $perm[4]$ και $perm[2] \rightarrow < 1, 0, 3, 2, 4, 5, 6 >$

$V = 2 + 5 * 8 = 42$

$r = 4$:

$s = 0$:

$perm[0] == 3$ ΨΕΥΔΗΣ

$s = 1$:

$perm[1] == 3$ ΨΕΥΔΗΣ

$s = 2$:

$perm[2] == 3$ ΑΛΗΘΗΣ

$f = 2$

αντιμετάθεσε τα $perm[3]$ και $perm[2] \rightarrow < 1, 0, 2, 3, 4, 5, 6 >$

$V = 2 + 4 * 42 = 170$

$r = 3$:

$s = 0$:

$perm[0] == 2$ ΨΕΥΔΗΣ

$s = 1$:

$perm[1] == 2$ ΨΕΥΔΗΣ

$s = 2$:

$perm[2] == 2$ ΑΛΗΘΗΣ

$f = 2$

αντιμετάθεσε τα $perm[2]$ και $perm[2] \rightarrow < 1, 0, 2, 3, 4, 5, 6 >$

$V = 2 + 3 * 170 = 512$

$r = 2$:

$s = 0$:

$perm[0] == 1$ ΑΛΗΘΗΣ

$f = 0$

αντιμετάθεσε τα $perm[1]$ και $perm[0] \rightarrow < 0, 1, 2, 3, 4, 5, 6 >$

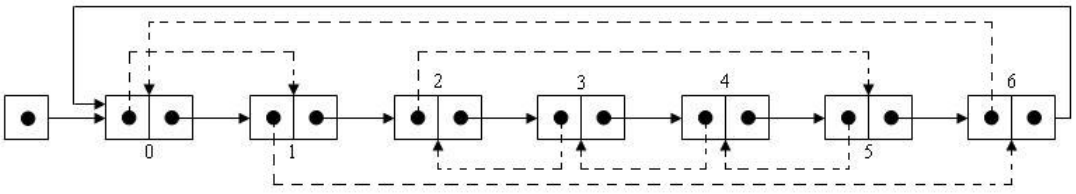
$V = 0 + 2 * 512 = 1024$

Επέστρεψε $V = 1024$

Τέλος, έχοντας υπολογίσει την μετάθεση στην οποία αντιστοιχεί η τιμή ενός υδατοσήμου ολοκληρώνουμε την διαδικασία κατασκευής ενός μεταθετικού γραφήματος μετατρέποντας την μετάθεση σε γράφημα. Η διαδικασία κατασκευής έχει τα ακόλουθα βήματα:

- κατασκευάζουμε την κυκλικά συνδεδεμένη λίστα η οποία αποτελείται από τους κόμβους που ορίζονται από την μετάθεση
- για κάθε $perm[i]$ προσθέτουμε μια ακμή από τον κόμβο i στον κόμβο $perm[i]$

Έτσι, αν πάρουμε την μετάθεση $< 1, 6, 5, 2, 3, 4, 0 >$ του αριθμού 1024 τότε το μεταθετικό γράφημα στο οποίο αυτή αντιστοιχεί είναι το εξής:



Σχήμα 3.6: Μεταθετικό γράφημα του αριθμου 1024

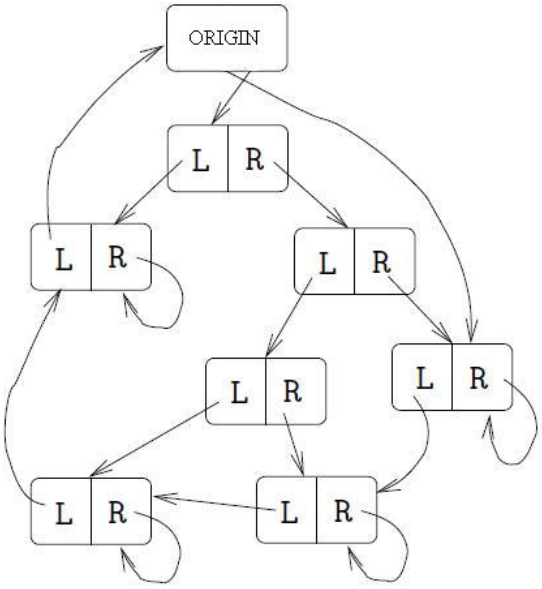
Από τον τρόπο κατασκευής αυτών των γραφημάτων προκύπτουν δύο βασικές ιδιότητες:

- Κάθε κόμβος έχει πλήθος εισερχόμενων και εξερχόμενων ακμών 2.
- Υπάρχει σίγουρα μονοπάτι χάμιλτον.

Σύμφωνα με τις δύο αυτές ιδιότητες κάθε επίθεση προσθήκης ή διαγραφής ακμής γίνεται εύκολα αντιληπτή. Το ίδιο ισχύει και στην περίπτωση επίθεσης αλλαγής κατεύθυνσης. Όμως αν έχουμε περιπτώσεις όπου γίνονται περισσότερες από μια επίθεσεις αυτού του είδους, τότε υπάρχει δυσκολία στο να τις καταλάβουμε. Έτσι, με λίγες επιθέσεις μπορούμε να καταλήξουμε σε διαφορετικό μεταθετικό γράφημα το οποίο καθιστά αδύνατη την αποκωδικοποίηση του γραφήματος στην αρχικά επιλεχθείσα τιμή του υδατοσήμου.

3.3.4 Επίπεδα Κυβικά Δένδρα

Τα γραφήματα αυτής της κλάσης είναι δυαδικά δένδρα με έναν επιπλέον κόμβο $ORIGIN$ που αποτελεί την ρίζα. Κάθε κόμβος του γραφήματος έχει δύο πεδία, ένα που δείχνει στο αριστερό παιδί και ένα που δείχνει στο δεξιό παιδί. Οι κόμβοι που έχουν παιδιά είναι μόνο οι εσωτερικοί κόμβοι ενώ από αυτούς εξαιρείται η ρίζα $ORIGIN$. Οι κόμβοι που αποτελούν τα φύλλα του δένδρου έχουν μια αυτοαναφορά στο δεξί πεδίο ενώ το αριστερό πεδίο χρησιμοποιείται για τον σχηματισμό μιας μοναδικά συνδεδεμένης λίστας μεταξύ αυτών των κόμβων και της ρίζας $ORIGIN$. Τους κόμβους σε αυτή τη λίστα τους συνδέουμε σύμφωνα με την αντίστροφη διάταξη που προκύπτει από την αναζήτηση κατά βάθος. Μπορούμε εύκολα να παρατηρήσουμε ότι η ρίζα $ORIGIN$ είναι προσπελάσιμη από κάθε άλλο κόμβο του γραφήματος. Ένα τέτοιο γράφημα είναι αυτό του Σχήματος 3.7.



Σχήμα 3.7: Επίπεδο Κυβικό Δένδρο

Σε τέτοιου είδους γραφήματα είναι εύκολο να διακρίνουμε αν έχει πραγματοποιηθεί κάποια επίθεση εισαγωγής ή διαγραφής ακμής καθώς είτε το δένδρο θα πάψει να είναι δυαδικό, είτε θα πάψει να υφίσταται η συνδεδεμένη λίστα. Επίσης, εύκολα διακρίνουμε επιθέσεις αντιστροφής ακμής ή προσθήκης/διαγραφής κόμβου. Σε αυτού του είδους τα γραφήματα αδυνατούμε να εντοπίσουμε πολλαπλές επιθέσεις γεγονός που αποτελεί και την βασική τους αδυναμία.

3.3.5 Αναγώγιμα Μεταθετικά Γραφήματα

Η τελευταία κλάση γραφημάτων την οποία θα αναλύσουμε και η οποία χρησιμοποιείται από τους αλγορίθμους της παρούσας διατριβής είναι αυτή των αναγώγιμων μεταθετικών γραφημάτων.
Θεώρημα Α: Έστω F ένα γράφημα ροής. Οι ακόλουθες προτάσεις για το F είναι ισοδύναμες:

1. είναι αναγώγιμο,

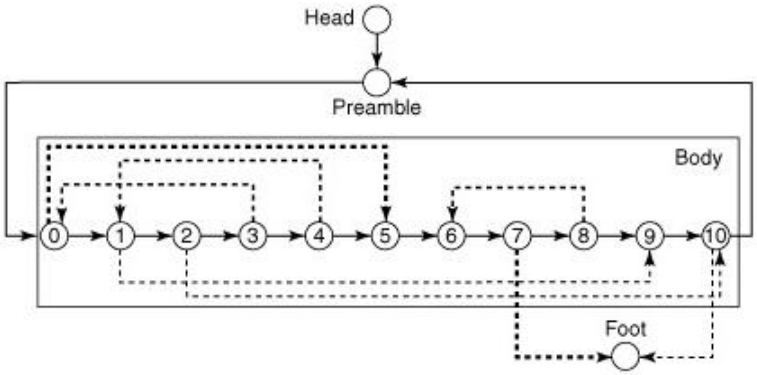
2. έχει μοναδικό DFS κατευθυνόμενο άκυκλο γράφημα,

3. μπορεί να μετασχηματιστεί σε έναν κόμβο επαναλαμβάνοντας τους μετασχηματισμούς T_1 και T_2, όπου ο T_1 αφαιρεί μια ακμή που σχηματίζει self-loop σε έναν κόμβο και ο T_2 επιλέγει έναν μη αρχικό κόμβο y που έχει μόνο μια εισερχόμενη ακμή και συνενώνει τους κόμβους x και y σε έναν κόμβο.

Τα αναγώγιμα μεταθετικά γραφήματα συνήθως αποτελούνται από τα εξής τέσσερα μέρη:

- **Αρχικός Κόμβος (Header Node)**: Αυτός είναι ο κόμβος ρίζα στο γράφημα και έχει πλήθος εξερχόμενων ακμών ένα. Οποιοσδήποτε άλλος κόμβος στο γράφημα είναι προσπελάσιμος από αυτον τον κόμβο.
- **Κόμβοι Προοίμιο (Preamble Nodes)**: Αυτοί οι κόμβοι είναι προαιρετικοί, όμως σε περίπτωση που υπάρχουν βρίσκονται αμέσως μετά τον αρχικό κόμβο. Το πλήθος των εξερχόμενων ακμών τους είναι ένα καθώς υπάρχουν μόνο οι ακμές που χρησιμοποιούνται από το μονοπάτι χάμιλτον.
- **Σώμα (Body)**: Αποτελείται από τους κόμβους οι οποίοι κωδικοποιούν την αυτοαντιστρέψιμη μετάθεση.
- **Τελικός Κόμβος (Footer Node)**: Αυτός ο κόμβος έχει πλήθος εξερχόμενων ακμών μηδέν και είναι προσπελάσιμος από κάθε άλλο κόμβο του γραφήματος.

Ένα αναγώγιμο μεταθετικό γράφημα φαίνεται στο Σχήμα 3.8.



Σχήμα 3.8: Αναγώγιμο Μεταθετικό Γράφημα

Το βασικό πλεονέκτημα αυτής της κατηγορίας γραφημάτων σε σχέση με τις άλλες είναι ότι αν γίνει κάποια επίθεση από αυτές που αναφέρθηκαν στην Παράγραφο 2.5 υπάρχει μεγάλη πιθανότητα το γράφημα να πάψει να είναι αναγώγιμο. Επομένως, έχουμε μεγαλύτερη πιθανότητα να μπορέσουμε να αντιληφθούμε κάποια πιθανή επίθεση που έχει γίνει στο γράφημα.

Αυτή η κλάση γραφημάτων θα μας απασχολήσει περισσότερο καθώς στις παραγράφους που ακολουθούν θα παρουσιάσουμε δύο αλγορίθμους, οι οποίοι μετατρέπουν την τιμή ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα καθώς επίσης και τους αντίστοιχους αλγορίθμους αποκωδικοποίησης.

3.4 Αλγόριθμος Κωδικοποίησης - Αποκωδικοποίησης CKCT

Έχοντας αναφέρει όλες τις βασικές έννοιες οι οποίες είναι απαραίτητες για τους αλγορίθμους που θα παρουσιάσουμε καθώς επίσης και τις κλάσεις γραφημάτων που χρησιμοποιούνται συνήθως σε αλγορίθμους υδατοσήμανσης βασισμένους σε γραφήματα μπορούμε να προχωρήσουμε στην περιγραφή του Αλγορίθμου CKCT που προτάθηκε από τον C. Collberg και τους συνεργάτες του [9] και ο οποίος μετατρέπει μια τιμή ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα. Συγκεκριμένα, παραθέτουμε τον τρόπο με τον οποίο μετατρέπεται η τιμή του υδατοσήμου σε αυτο-αντιστρέψιμη μετάθεση καθώς επίσης και τα βήματα της κωδικοποίησης αυτής της μετάθεσης σε αναγώγιμο μεταθετικό γράφημα. Επιπλέον, παρουσιάζουμε την αντίστροφη διαδικασία, δηλαδή την αποκωδικοποίηση του αναγώγιμου μεταθετικού γραφήματος στην αυτο-αντιστρέψιμη μετάθεση που αυτό αντιστοιχεί και την μετατροπή της μετάθεσης αυτής στον ακέραιο τον οποίο αναπαριστά.

Αρχίζοντας, αναλύουμε την διαδικασία κωδικοποίσης ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα. Αυτή η διαδικασία απαιτεί αρχικά την μετατροπή της τιμής ενος υδατοσήμου, έστω n, σε αυτο-αντιστρέψιμη μετάθεση. Η γενικότερη ιδέα αυτής της μετατροπής είναι η παραγωγή όλων των δυνατών μεταθέσεων αρχίζοντας από μια αρχική μετάθεση και η επιλογή της $n-$οστής αυτο-αντιστρέψιμης μετάθεσης. Όμως από αυτή τη διαδικασία προκύπτουν δύο βασικά ζητήματα. Το πρώτο είναι η επιλογή του μεγέθους της μετάθεσης που θα επιλέξουμε αρχικά έτσι ώστε να παράξουμε τουλάχιστον n αυτο-αντιστρέψιμες μεταθέσεις και το δεύτερο είναι ο τρόπος με τον οποίο παράγουμε όλες τις μεταθέσεις από την αρχική και η εύρεση των αυτο-αντιστρέψιμων μεταθέσεων από αυτές.
Απάντηση στο πρώτο ζήτημα δίδεται από τον ακόλουθο τύπο:

$$I(n) = n! \sum_{k=0}^{\lfloor \frac{n}{2} \rfloor} \frac{1}{2^k k!(n-2k)!} \quad (1)$$

Αυτή η συνάρτηση μάς απαντάει στο πόσες είναι οι αυτο-αντιστρέψιμες μέταθέσεις που υπάρχουν αν πάρουμε μια μετάθεση με μήκος n και παράξουμε όλες τις δυνατές μεταθέσεις της. Για παράδειγμα, αν δώσουμε το $n = 4$ τότε το $I(n) = 10$, δηλαδή αν πάρουμε μια αρχική μετάθεση με τέσσερα στοιχεία και παράξουμε όλες τις δυνατές μεταθέσεις, οι οποίες είναι $n!$ δηλαδή 24 στο παράδειγμα μας, τότε μόνο οι 10 από αυτές είναι αυτο-αντιστρέψιμες. Οπότε αν δώσουμε ως τιμή υδατοσήμου έναν αριθμό στο διάστημα 1 έως 10 τότε χρειαζόμαστε μια μετάθεση με τέσσερα στοιχεία για να την κωδικοποιήσουμε.

Το δεύτερο ζήτημα είναι καθαρά υποκειμενικό καθώς αφορά τον τρόπο με τον οποίο θα παράξουμε όλες τις δυνατές μεταθέσεις και ο οποίος διαφέρει από προγραμματιστή σε προγραμματιστή. Στην παρούσα διατριβή, ο τρόπος με τον οποίο παράγουμε όλες τις δυνατές μεταθέσεις παρουσιάζεται στον παρακάτω αλγόριθμο.

Αλγόριθμος 7 Αλγόριθμος Permutations

1. Θέσε έναν μετρητή $z =$ με την τιμή του υδατοσήμου
2. Ανάλυσε την μετάθεση σε επίπεδα μέχρι να φτάσεις στο τελευταίο επίπεδο
   Το τελευταίο επίπεδο ορίζεται να είναι η επιλογή όλων των στοιχείων της μετάθεσης εκτός του τελευταίου. Αν είσαι ήδη στο τελευταίο επίπεδο πήγαινε κατευθείαν στο βήμα 3
3. Υπολόγισε καινούργια μετάθεση στο τρέχον επίπεδο που είσαι
4. Έλεγξε την καινούργια μετάθεση αν είναι αυτο-αντιστρέψιμη. Αν είναι τότε μείωσε το z κατά 1
5. Αν το $z = 0$ τότε τερμάτισε και επέστρεψε την τελευταία αυτο-αντιστρέψιμη μετάθεση που υπολόγισες
6. Έλεγξε αν υπάρχει καινούργια μετάθεση στο τρέχον επίπεδο και κάνε το εξής:
   Αν υπάρχει υπολόγισε τη και επέστρεψε στο βήμα 2
   Αλλιώς επέστρεψε ένα επίπεδο πίσω και επανέλαβε το βήμα 6

Για να κατανοήσουμε καλυτερα το πώς εργάζεται ο παραπάνω αλγόριθμος, αλλά και πού μας χρησιμεύει η συνάρτηση (1) παραθέτουμε το παράδειγμα μετατροπής του αριθμού 8 σε αυτο-αντιστρέψιμη ακολουθία. Η όλη διαδικασία αυτής της μετατροπής είναι η εξής:
Αρχικά: υπολογίζεται το κατάλληλο n για το οποίο θα έχουμε τουλάχιστον 8 αυτο-αντιστρέψιμες μεταθέσεις:

- για $n = 1$ το $I(n) = 1$
- για $n = 2$ το $I(n) = 2$
- για $n = 3$ το $I(n) = 4$
- για $n = 4$ το $I(n) = 10$

Άρα για $n = 4$ έχουμε τουλάχιστον 8 αυτο-αντιστρέψιμες μεταθέσεις. Επομένως κατασκευάζουμε την μετάθεση [1 2 3 4]. Έτσι καλούμε τον Αλγόριθμο Permutations δίδοντας ως είσοδο αυτή τη μετάθεση και την τιμή του υδατοσήμου. Ο αλγόριθμος θα παράγει μεταθέσεις μέχρις ότου να βρεί την όγδοη αυτο-αντιστρέψιμη μετάθεση. Όλες οι δυνατές μεταθέσεις που μπορούν να παραχθούν με τον αλγόριθμο που περιγράψαμε προηγουμένως παρουσιάζονται στο παρακάτω σχήμα.

Επίπεδο 0	Επίπεδο 1	Επίπεδο 2	Μεταθέσεις
1 2 3 4			
	1 2 (3 4)		
		1 (2 3 4)	
			1 2 3 4
			2 3 4 1
			3 4 1 2
			4 1 2 3
		1 (3 4 2)	
			1 3 4 2
			3 4 2 1
			4 2 1 3
			2 1 3 4
		1 (4 2 3)	
			1 4 2 3
			4 2 3 1
			2 3 1 4
			3 1 4 2
	1 2 (4 3)		
		1 (2 4 3)	
			1 2 4 3
			2 4 3 1
			4 3 1 2
			3 1 2 4
		1 (4 3 2)	
			1 4 3 2
			4 3 2 1
			3 2 1 4
			2 1 4 3
		1 (3 2 4)	
			1 3 2 4
			3 2 4 1
			2 4 1 3
			4 1 3 2

Σχήμα 3.9: Όλες οι δυνατές μεταθέσεις αρχίζοντας από την μετάθεση [1 2 3 4]

Ο Αλγόριθμος Permutations θα σταματήσει στην μετάθεση [3 2 1 4] καθώς αυτή είναι η όγδοη αυτο-αντιστρέψιμη μετάθεση. Έτσι, το 8 μετατρέπεται στην μετάθεση [3 2 1 4].

Τέλος για να ολοκληρωθεί η διαδικασία της κωδικοποίησης αρκεί να παραθέσουμε τον αλγόριθμο που μετατρέπει μια αυτο-αντιστρέψιμη μετάθεση σε αναγώγιμο μεταθετικό γράφημα. Προτού όμως παρουσιάσουμε τα βήματα αυτού πρέπει να τονιστεί ότι ως είσοδο δέχεται μια αυτο-αντιστρέψιμη μετάθεση σ για την οποία ισχύει ότι $\sigma(0) = 0$ και $\sigma = \sigma^{-1}$. Άρα κάθε φορά αυξάνουμε το μήκος της αυτο-αντιστρέψιμης μετάθεσης που παράγεται από τον Αλγόριθμο 7 κατά 1 προσθέτοντας το στοιχείο 0 στην πρώτη θέση. Έτσι, η μετάθεση έχει μήκος $n + 1$ όμως παράγονται τόσες αυτο-αντιστρέψιμες ακολουθίες όσες θα παράγονταν και με τα n στοιχεία. Επομένως, τα βήματα δημιουργίας του αναγώγιμου μεταθετικού γραφήματος $G = (V, E)$ παρουσιάζονται στον ακόλουθο αλγόριθμο:

Αλγόριθμος 8 Αλγόριθμος RPG_Encoding

1. Αρχικοποίηση: $V = \{h, u_0, u_1, ..., u_k\}$ και $E = \{(u_i, u_{i+1}) : 0 \leq i < k\}$, όπου το h είναι ο αρχικός κόμβος (header) και u_k ο τελικός κόμβος (footer). Οι υπόλοιποι κόμβοι αποτελούν το σώμα του γραφήματος
2. Για κάθε $i \in Z_k \setminus \{0\}$ έτσι ώστε $\sigma(i-1) < \sigma(i)$, θέσε $E = E \cup \{(u_{\sigma(i-1)}, u_{\sigma(i)})\}$.
   Αν η ακμή $(u_{\sigma(i-1)}, u_{\sigma(i)})$ έχει προστεθεί από το βήμα 1 τότε μην προσθέσεις αυτή στο σύνολο E αλλά την $(u_{\sigma(i-1)}, u_k)$.
   Μέχρι αυτό το σημείο το G είναι άκυκλο και αναγώγιμο
3. Υπολόγισε τους κυρίαρχους κόμβους του $G - \{h\}$, χρησιμοποιώντας το u_0 ως ρίζα
4. Αρχικοποίησε $B = \{j \in \{0, 1, ..., k-1\} : \sigma(j) > \sigma(j+1 mod k)\}$. Αρχικοποίησε $L = \emptyset$
5. Για κάθε $j \in B$, αν το $u_{\sigma(j+1) mod k}$ κυριαρχεί στο $u_{\sigma(j)}$, θέσε $E = E \cup \{(u_{\sigma(j)}, u_{\sigma(j+1)})\}$, αλλιώς θέσε $L = L \cup \{(u_{\sigma(j)}, u_{\sigma(j+1)})\}$. Στην περίπτωση όπου μια ακμή προστίθεται στο E, είναι πίσω ακμή και γι' αυτό το λόγο δεν επηρεάζει την ιεραρχία των κυρίαρχων κόμβων
6. Έστω $\tau : L \to Z$ να ορίζεται ως $\tau((u_i, u_j)) = |\{(u_{i'}, u_{j'}) \in L : i' < i, j' > j\}|$.
   Έστω $m = max\tau(L)$
7. Αν $m = 0$, θέσε $E = E \cup \{(h, u_0)\}$ και επέστρεψε $G = (V, E)$
8. Αλλιώς, θέσε $V = V \cup \{p_1, p_2, ..., p_m\}$. Για κάθε $(u_i, u_j) \in L$, αν $\tau((u_i, u_j)) > 0$, θέσε $E = E \cup \{(u_i, p_{\tau((u_i, u_j))})\}$, όπου $\{p_1, p_2, ..., p_m\}$ είναι οι κορυφές προοίμιο.
   Θέσε $E = E \cup \{(h, p_m), (p_m, p_{m-1}), ..., (p_2, p_1), (p_1, u_0)\}$ και επέστρεψε το $G = (V, E)$

Στο βήμα 3 του αλγορίθμου παρατηρούμε ότι απαιτείται να γίνει υπολογισμός των κυρίαρχων κόμβων στο γράφημα που έχει δημιουργηθεί από τα βήματα 1 και 2. Τους κυρίαρχους κόμβους σε ένα γράφημα μπορούμε να τους υπολογίσουμε με τον ακόλουθο αλγόριθμο:

Αλγόριθμος 9 Αλγόριθμος Εύρεσης Κυρίαρχων Κόμβων

1. $Dom(n_0) = \{n_0\}$
2. Για κάθε $n \in N - \{n_0\}$:
   θέσε $Dom(n) = N$
3. Όσο υπάρχουν αλλαγές σε κάποιο $Dom(n)$ επανάλαβε
   Για κάθε $n \in N - \{n_0\}$:
   $Dom(n) = \{n\}$ ένωση με την τομή, για όλα τα p που ανήκουν στους προκατόχους του n, των $Dom(p)$

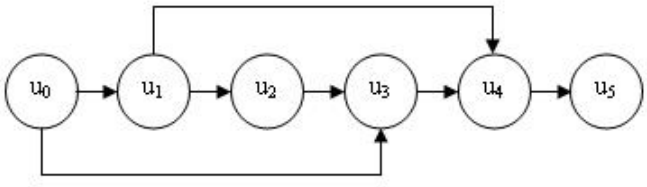
Για την καλύτερη κατανόηση του Αλγορίθμου RPG_Encoding παραθέτουμε το παράδειγμα μετατροπής της αυτο-αντιστρέψιμης μετάθεσης [0 3 2 1 4] της τιμής του υδατοσήμου 8, που υπολογίσαμε με τον Αλγόριθμο Permutations, σε αναγώγιμο μεταθετικό γράφημα. Αναλυτικά ο αλγόριθμος υπολογίζει τα εξής:

Βήμα 1: αρχικοποιούνται τα σύνολα $V = \{h, u_0, u_1, u_2, u_3, u_4, u_5\}$ και $E = \{(u_0, u_1), (u_1, u_2), (u_2, u_3), (u_3, u_4), (u_4, u_5)\}$

Βήμα 2:

- για $i = 1$ το $\sigma(0) < \sigma(1)$, άρα $E = E \cup \{(u_0, u_3)\}$
- για $i = 2$ το $\sigma(1) > \sigma(2)$
- για $i = 3$ το $\sigma(2) > \sigma(3)$
- για $i = 4$ το $\sigma(3) < \sigma(4)$, άρα $E = E \cup \{(u_1, u_4)\}$

Άρα το σύνολο $E = \{(u_0, u_1), (u_1, u_2), (u_2, u_3), (u_3, u_4), (u_4, u_5), (u_0, u_3), (u_1, u_4)\}$ και το κατευθυνόμενο άκυκλο γράφημα είναι το εξής:



Σχήμα 3.10: Κατευθυνόμενο άκυκλο γράφημα μέχρι το βήμα 2 του Αλγορίθμου 8

Βήμα 3:

- $Dom(u_0) = \{u_0\}$
- $Dom(u_1) = \{u_0, u_1\}$
- $Dom(u_2) = \{u_0, u_1, u_2\}$
- $Dom(u_3) = \{u_0, u_3\}$
- $Dom(u_4) = \{u_0, u_4\}$
- $Dom(u_5) = \{u_0, u_4, u_5\}$

Βήμα 4: αρχικοποιείται το $B = \{1, 2, 4\}$

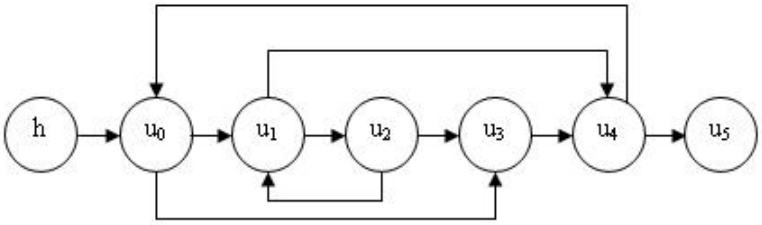
Βήμα 5: προστίθενται στο σύνολο E οι ακμές (u_2, u_1) και (u_4, u_0) και στο σύνολο L η ακμή (u_3, u_2)

Βήμα 6: υπολογίζεται το $\tau(u_3, u_2)$ και ισούται με 0. Άρα και το $m = 0$

Βήμα 7: εφόσον το $m = 0$ προστίθεται στο σύνολο E η ακμή (h, u_0)

Βήμα 8: αυτό το βήμα δεν εκτελείται εφόσον ισχύει ότι το $m = 0$

Έτσι, το αναγώγιμο μεταθετικό γράφημα της τιμής 8 είναι το εξής:



Σχήμα 3.11: Αναγώγιμο μεταθετικό γράφημα του αριθμού 8 από τον Αλγόριθμο 8

Με τους αλγορίθμους που έχουμε παρουσιάσει μέχρι στιγμής σε αυτή την ενότητα ολοκληρώνουμε τη διαδικασία κωδικοποίησης μιας τιμής ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα έτσι όπως αυτή προτάθηκε από τον C. Collberg και τους συνεργάτες του [9]. Στη συνέχεια, παραθέτουμε τους αλγορίθμους που υλοποιούν την αντίστροφη διαδικασία, δηλαδή την αποκωδικοποίηση ενός αναγώγιμου μεταθετικού γραφήματος στην τιμή του υδατοσήμου που αυτό αντιστοιχεί.

Έχοντας ένα αναγώγιμο μεταθετικό γράφημα, η αποκωδικοποίηση του στην αυτο-αντιστρέψιμη μετάθεση που αυτό αντιστοιχεί γίνεται με τον παρακάτω αλγόριθμο:

Αλγόριθμος 10 Αλγόριθμος RPG_Decoding

---

1. Υπολόγισε ένα μονοπάτι χάμιλτον P του G
2. Έστω u_0 ο πρώτος κόμβος με προς τα εμπρός ακμή που δεν ανήκει στο P. Έστω $u_1, u_2, ..., u_{k-1}$ οι υπο-ακόλουθοι κόμβοι στο P, χωρίς τον f. Έστω $p_m, p_{m-1}, ..., p_1$ οι κόμβοι μεταξύ του h και του u_0
3. Έστω $B = \{u_0, u_1, ..., u_{k-1}\}$
4. Δημιούργησε μια άδεια ταξινομημένη λίστα κόμβων L. Για κάθε $i \in \{0, 1, ..., k-1\}$, αν το u_i έχει μια εξερχόμενη ακμή στο $B \cup \{f\}$, εισήγαγε το u_i στην L. Αν το u_i δεν έχει καμία εξερχόμενη ακμή στα $\{p_1, p_2, ..., p_m\}$, εισάγαγε το στο τέλος της λίστας. Αλλιώς, αν έχει μια ακμή στο p_j, εισάγαγε το πρίν το τελευταίο j στοιχείο της λίστας
5. Για κάθε $i \in \{0, 1, ..., k-1\}$, αν το u_i έχει μια μόνο εισερχόμενη ακμή από το $B \cup \{h, p_1\}$, πρόσθεσε μια ακμή από το πρώτο στοιχείο της L στο u_i και αφαίρεσε αυτό το πρώτο στοιχείο από την L
6. Αρχικοποίησε έναν πίνακα ακεραίων $A[]$ με k στοιχεία. Θέσε $A[0] = 0$
7. Για κάθε $i \in \{1, 2, ..., k-1\}$, θα υπάρχει τώρα ακριβώς μια ακμή από το $u_{A[i-1]}$ στο $B \cup \{f\}$ η οποία δεν ανήκει στο μονοπάτι P. Αν αυτή η ακμή πηγαίνει σε έναν κόμβο u_j, θέσε $A[i] = j$. Αν αυτή η ακμή πηγαίνει στον κόμβο f, θέσε $A[i] = A[i-1] + 1$
8. Επέστρεψε τον πίνακα A που τώρα περιέχει την αυτο-αντιστρέψιμη μετάθεση

Αν στον παραπάνω αλγόριθμο δώσουμε ως είσοδο το αναγώγιμο μεταθετικό γράφημα του Σχήματος 3.11 τότε οι υπολογισμοί που θα γίνουν είναι οι ακόλουθοι:

Βήμα 1: υπολογίζεται το μονοπάτι χάμιλτον το οποίο είναι από τον h στον u_0, από τον u_0 στον u_1, από τον u_1 στον u_2, από τον u_2 στον u_3, από τον u_3 στον u_4 και από τον u_4 στον f

Βήμα 2: αυτό το βήμα είναι τετριμένο

Βήμα 3: έστω $B = \{u_0, u_1, u_2, u_3, u_4\}$

Βήμα 4: δημιουργείται η λίστα $L = \{u_3\}$

Βήμα 5: προστίθεται η ακμή (u_3, u_2)

Βήμα 6: αρχικοποιείται ο πίνακας A που έχει πέντε στοιχεία

Βήμα 7: υπολογίζεται ο πίνακας $A = [0, 3, 2, 1, 4]$

Βήμα 8: επιστρέφεται ο πίνακας A

Τέλος, για να ολοκληρωθεί η διαδικασία της αποκωδικοποίησης αρκεί να εφαρμόσουμε ξανά τον Αλγόριθμο 7 που παράγει όλες τις διαφορετικές μεταθέσεις και αντί να ελέγχουμε πότε το z θα γίνει ίσο με μηδέν πρέπει να ελέγχουμε ποιές από αυτές τις μεταθέσεις που παράγονται είναι αυτο-αντιστρέψιμες και να αυξάνουμε έναν μετρητή κάθε φορά που βρίσκουμε μια. Ο μετρητής αρχικοποιείται στο μηδέν. Η όλη διαδικασία θα σταματά και θα επιστρέφεται ο μετρητής, ο οποίος είναι η τιμή του υδατοσήμου, μόλις παραχθεί μια μετάθεση η οποία θα είναι ίδια με αυτή που προέκυψε από τον Αλγόριθμο 10.

Έτσι, στον Αλγόριθμο CKCT η συνάρτηση κωδικοποίησης *encode* αποτελείται από τους Αλγορίθμους Permutations και RPG_Encoding, ενώ η συνάρτηση αποκωδικοποίησης *decode* απο τον Αλγορίθμο RPG_Decoding και τον Permutations έχοντας όμως εφαρμόσει την αλλαγή που αναφέραμε παραπάνω. Μπορούμε να υποστηρίξουμε ότι αυτός ο κωδικοποιητής είναι "καλός" καθώς διέπεται και από τις τέσσερις ιδιότητες που αναφέραμε στην Ενότητα 2.4.7. Συγκεκριμένα, γίνεται χρήση κατάλληλων τύπων γραφημάτων γιατί προκύπτουν γραφήματα που έχουν πλήθος εξερχόμενων ακμών το πολύ δύο και επομένως ομοιάζουν με τα γραφήματα των προγραμμάτων. Επίσης, τα γραφήματα που προκύπτουν από αυτόν τον κωδικοποιητή έχουν μεγάλο βαθμό ανθεκτικότητας καθώς οποιαδήποτε επίθεση σε ακμή γίνεται εύκολα αντιληπτή. Αυτό συμβαίνει γιατί είτε θα είναι ακμή από αυτές που διατηρούν την λίστα των κόμβων, είτε ακμή από αυτές που προστέθηκαν σύμφωνα με τους κυρίαρχους κόμβους του γραφήματος. Επιπλέον, το μέγεθος του υδατογραφημένου προγράμματος δεν διαφέρει πολύ ως προς το μέγεθος του αρχικού προγράμματος και αυτό προκύπτει από το ότι τα γραφήματα που δημιουργούνται και που θα εισαχθούν στο πρόγραμμα είναι σχετικά μικρά σε μέγεθος. Τέλος, οι Αλγόριθμοι RPG_Encoding και RPG_Decoding μπορούν να υλοποιηθούν σε πολυωνυμικό χρόνο, άρα ο κωδικοποιητής είναι αποτελεσματικός. Έτσι, σύμφωνα με όλα όσα αναφέραμε μπορούμε να πούμε ότι η κλάση γραφημάτων που χρησιμοποιείται από αυτόν τον κωδικοποιητή είναι κατάλληλη όμως και ο ίδιος ο κωδικοποιητής είναι "καλός".

3.5 Αλγόριθμος Κωδικοποίησης - Αποκωδικοποίησης CN

Στην προηγούμενη παράγραφο έγινε αναφορά στον Αλγόριθμο CKCT ο οποίος μετατρέπει μια ακέραια τιμή ενος υδατοσήμου σε αυτο-αντιστρέψιμη μετάθεση και με τη σειρά της αυτή μετατρέπεται σε αναγώγιμο μεταθετικό γράφημα. Στην παρούσα ενότητα θα παρουσιάσουμε έναν αλγόριθμο, τον CN, που προτάθηκε από τους Μ. Χρόνη και Σ.Δ. Νικολόπουλο [3] και ο οποίος ανήκει στην ίδια κατηγορία με τον CKCT, δηλαδή και αυτός μετατρέπει την τιμή του υδατοσήμου σε αυτο-αντιστρέψιμη μετάθεση και μετά σε αναγώγιμο μεταθετικό γράφημα.

Η κωδικοποίηση σε αυτό τον αλγόριθμο έχει δύο φάσεις. Η πρώτη φάση κωδικοποιεί μια ακέραια τιμή ενός υδατοσήμου w σε αυτο-αντιστρέψιμη μετάθεση (self-inverting permutation). Αυτό πραγματοποιείται με τον Αλγόριθμο Encode_W_to_SIP ο οποίος υπολογίζει την δυαδική αναπαράσταση του w και κατασκευάζει μια διτονική μετάθεση (bitonic permutation) μήκους 2n+1. Τελικά, από αυτή την διτονική μετάθεση προκύπτει η αυτοαντιστρέψιμη μετάθεση η οποία έχει και αυτή το ίδιο μήκος με την πρώτη. Στην δεύτερη

φάση γίνεται η μετατροπή της αυτο-αντιστρέψιμης μετάθεσης σε κατευθυνόμενο άκυκλο γράφημα και στη συνέχεια μετατρέπεται αυτό το γράφημα σε αναγώγιμο μεταθετικό γράφημα. Αυτή η δεύτερη φάση υλοποιείται με τον Αλγόριθμο Encode_SIP_to_RPG.

Ορισμός 3.11 (Διτονική Μετάθεση (Bitonic Permutation)). Μια μετάθεση π ονομάζεται διτονική είτε αν μονότονα αυξάνεται και μετά μονότονα μειώνεται, είτε αν συμβαίνει το αντίστροφο, δηλαδή μονότονα μειώνεται και έπειτα μονότονα αυξάνεται.

Στον Αλγόριθμο 11, στον οποίο περιγράφονται τα βήματα μετατροπής ενός ακεραίου αριθμού σε αυτο-αντιστρέψιμη μετάθεση, γίνεται χρήση διτονικών μεταθέσεων που μονότονα αυξάνουν και μετά μονότονα μειώνουν.

Αλγόριθμος 11 Αλγόριθμος Encode_W_to_SIP

1. Υπολόγισε την δυαδική αναπαράσταση του w, έστω $B = b_1 b_2 ... b_n$
2. Κατασκεύασε τον δυαδικό αριθμό $B' = 00...0||B||1$ με μήκος 2n+1, και μετά την δυαδική ακολουθία $B^* = (b_1, b_2, ..., b_{n'})$ αντιστρέφοντας την B'
3. Βρες την ακολουθία $X = (x_1, x_2, ..., x_k)$ των θέσεων των 0 και την ακολουθία $Y = (y_1, y_2, ..., y_k)$ των θέσεων των 1 στην B^* από αριστερά προς τα δεξιά
4. Κατασκεύασε την διτονική μετάθεση $\pi^b = X||Y^R$ με τους 2n+1 αριθμούς. Έστω $\pi^b = (x_1, x_2, ..., x_k, y_m, y_{m-1}, ..., y_1)$
5. Θέσε $(\pi_1, \pi_2, ..., \pi_\varkappa, \pi_{\varkappa+1}, ..., \pi_{n'}) = (x_1, x_2, ..., x_k, y_m, y_{m-1}, ..., y_1)$
   $i = 1$
   $j = n$
   Όσο $i < j$ επανάλαβε
   κατασκεύασε τον κύκλο $c_i = (\pi_i, \pi_j)$ μεγέθους 2
   $i = i + 1$
   $j = j - 1$
   Τέλος_επανάληψης
   Αν $i = j$ τότε κατασκεύασε τον κύκλο $c_i = (\pi_i)$ μεγέθους 1
6. Κατασκεύασε την μετάθεση $\pi^* = (\pi_1, \pi_2, ..., \pi_{n'})$ από τους n'=2n+1 αριθμούς έτσι ώστε $\pi_i = i, 1 \leq i \leq n'$
7. Έστω C το σύνολο όλων των κύκλων που υπολογίσθηκαν στο βήμα 5
   Για κάθε κύκλο $(\pi_i, \pi_j) \in C$ μεγέθους 2 θέσε $\pi_{\pi_i} = \pi_j$ και $\pi_{\pi_j} = \pi_i$
8. Επέστρεψε την αυτο-αντιστρέψιμη μετάθεση π^*

Για την καλύτερη κατανόηση του παραπάνω αλγορίθμου παραθέτουμε ένα παράδειγμα μετατροπής του αριθμού $w = 8$ σε αυτο-αντιστρέψιμη μετάθεση.

Βήμα 1: μετατρέπεται το w στην δυαδική του μορφή. Οπότε το $B = 1000$.

Βήμα 2: κατασκευάζεται ο δυαδικός αριθμός $B' = 000010001$ και έπειτα η δυαδική ακολουθία $B^* = (1, 1, 1, 1, 0, 1, 1, 1, 0)$.

Βήμα 3: υπολογίζονται οι ακολουθίες $X = (5, 9)$ και $Y = (1, 2, 3, 4, 6, 7, 8)$.

Βήμα 4: κατασκευάζεται η διτονική μετάθεση $\pi^b = (5, 9, 8, 7, 6, 4, 3, 2, 1)$.

Βήμα 5: κατασκευάζεται το σύνολο $C = \{(5,1),(9,2),(8,3),(7,4),(6)\}$.

Βήμα 6: κατασκευάζεται η μετάθεση $\pi^* = (1,2,3,4,5,6,7,8,9)$.

Βήμα 7: ανακατασκευάζεται η π^* σύμφωνα με το σύνολο των κύκλων που υπολογίστηκαν στο βήμα 5, οπότε $\pi^* = (5,9,8,7,1,6,4,3,2)$.

Βήμα 8: επιστρέφεται η αυτο-αντιστρέψιμη ακολουθία π^* η οποία αντιστοιχεί στον αριθμό 8.

Με τον Αλγόριθμο 4 ολοκληρώνεται η πρώτη φάση της κωδικοποίησης. Πρωτού ακολουθήσει η παρουσίαση των βημάτων της φάσης της μετατροπής μιας αυτο-αντιστρέψιμης μετάθεσης σε αναγώγιμο μεταθετικό γράφημα πρέπει να οριστεί το πώς υπολογίζονται τα κυρίαρχα στοιχεία σε μια μετάθεση. Σε μια μετάθεση π ένα στοιχείο j κυριαρχείται από ένα στοιχείο i αν $i > j$ και $\pi_i^{-1} < \pi_j^{-1}$, δηλαδή αν το στοιχείο i είναι μεγαλύτερο του j και η θέση του i είναι μικρότερη από εκείνη του j. Ένα στοιχείο j κυριαρχείται άμεσα (directly) από ένα στοιχείο i αν το i κυριαρχεί του j και δεν παρεμβάλλεται μεταξύ αυτών άλλο στοιχείο k στην π τέτοιο ώστε το i να κυριαρχεί στο k και το k με την σειρά του να κυριαρχεί στο j. Με βάση αυτό ορίζουμε το σύνολο κυριαρχίας $dom(i)$ ενός στοιχείου i το οποίο είναι το σύνολο όλων εκείνων των στοιχείων της μετάθεσης π τα οποία κυριαρχούνται από το i. Αντίστοιχα, ορίζουμε και το σύνολο άμεσης κυριαρχίας $didom(i)$ το οποίο περιέχει όλα τα στοιχεία της μετάθεσης π που κυριαρχούνται άμεσα από το i [3].

Έχοντας ορίσει τον τρόπο υπολογισμού των κυρίαρχων στοιχείων σε μια μετάθεση, παραθέτουμε τα βήματα μετατροπής μιας αυτο-αντιστρέψιμης ακολουθίας σε αναγώγιμο μεταθετικό γράφημα.

Αλγόριθμος 12 Αλγόριθμος Encode_SIP_to_RPG

1. Κατασκεύασε ένα κατευθυνόμενο άκυκλο γράφημα $D[\pi^*]$ με n κόμβους ως εξής:
   a. $V(D[\pi^*]) = \{v_1, v_2, ..., v_n\}$
   b. Υπολόγισε το σύνολο $didom(i)$ για κάθε στοιχείο i της π, $1 \leq i \leq n-1$
   c. Για κάθε $j \in didom(i)$, πρόσθεσε την ακμή (v_i, v_j) στο σύνολο $E(D[\pi^*])$
   d. Πρόσθεσε δύο κόμβους $s = v_{n+1}$ και $t = v_0$ στο σύνολο $V(D[\pi^*])$
   e. Πρόσθεσε $(s, v_i) \in E(D[\pi^*])$, για κάθε v_i που έχει πλήθος εισερχόμενων ακμών μηδέν
   f. Πρόσθεσε $(v_i, t) \in E(D[\pi^*])$, για κάθε v_i που έχει πλήθος εξερχόμενων ακμών μηδέν
2. Για κάθε κόμβο $v_i \in V(D[\pi^*])$, $1 \leq i \leq n$, κάνε
   a. Υπολόγισε το σύνολο $P(v_i) = \{v_j \in V(D[\pi^*]) | (v_j, v_i) \in E(D[\pi^*])\}$
   b. Επίλεξε τον κόμβο v_m με την μεγαλύτερη ετικέτα στο $P(v_i)$
   c. Θέσε $p(v_i) = v_m$
3. Κατασκεύασε το κατευθυνόμενο γράφημα $F[\pi^*]$ με n+2 κόμβους, ως εξής:
   a. $V(F[\pi^*]) = \{t = u_0, u_1, ..., u_n, u_{n+1} = s\}$
   b. Για i από n μέχρι 0 κάνε
      Πρόσθεσε την ακμή (u_{i+1}, u_i) στο $E(F[\pi^*])$
4. Για κάθε κόμβο $u_i \in V(F[\pi^*])$,$1 \leq i \leq n$, κάνε
   Πρόσθεσε την ακμή (u_i, u_m) στο σύνολο $E(F[\pi^*])$ αν $v_m = p(v_i)$
5. Επέστρεψε το γράφημα $F[\pi^*]$

Στον παραπάνω αλγόριθμο παρατηρούμε ότι στο βήμα 1 δημιουργείται το κατευθυνόμενο άκυκλο γραφήμα και στη συνέχεια στα βήματα 2, 3 και 4 αυτό μετατρέπεται σε αναγώγιμο μεταθετικό γράφημα. Παραθέτουμε αναλυτικά τα βήματα μετατροπής της μετάθεσης $\pi^* = (5, 9, 8, 7, 1, 6, 4, 3, 2)$, που είδαμε στο προηγούμενο παράδειγμα, σε κατευθυνόμενο άκυκλο γράφημα και στη συνέχεια σε αναγώγιμο μεταθετικό γράφημα σύμφωνα με τον παραπάνω αλγόριθμο.

Βήμα 1: κατασκευάζεται το κατευθυνόμενο άκυκλο γράφημα ως εξής:

- $V(D[\pi^*]) = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$
- $didom(v_5) = \{v_1, v_4\}$
  $didom(v_9) = \{v_8\}$
  $didom(v_8) = \{v_7\}$
  $didom(v_7) = \{v_1, v_6\}$
  $didom(v_1) = \{\}$
  $didom(v_6) = \{v_4\}$
  $didom(v_4) = \{v_3\}$
  $didom(v_3) = \{v_2\}$
- προστίθενται οι ακμές: $(v_5, v_1), (v_5, v_4), (v_9, v_8), (v_8, v_7), (v_7, v_1), (v_7, v_6), (v_6, v_4), (v_4, v_3), (v_3, v_2)$

Βήμα 2: για κάθε κόμβο $v_i \in V(D[\pi])$ υπολογίζονται τα εξής:

- $P(v_1) = \{v_5, v_7\}$
  $P(v_2) = \{v_3\}$
  $P(v_3) = \{v_4\}$
  $P(v_4) = \{v_5, v_6\}$
  $P(v_5) = \{s\}$
  $P(v_6) = \{v_7\}$
  $P(v_7) = \{v_8\}$
  $P(v_8) = \{v_9\}$
  $P(v_9) = \{s\}$
- $p(v_1) = v_7$
  $p(v_2) = v_3$
  $p(v_3) = v_4$
  $p(v_4) = v_6$
  $p(v_5) = s$
  $p(v_6) = v_7$
  $p(v_7) = v_8$
  $p(v_8) = v_9$
  $p(v_9) = s$

Βήμα 3: κατασκευάζεται το κατευθυνόμενο γράφημα $F[\pi^*]$ που έχει n+2 κόμβους:

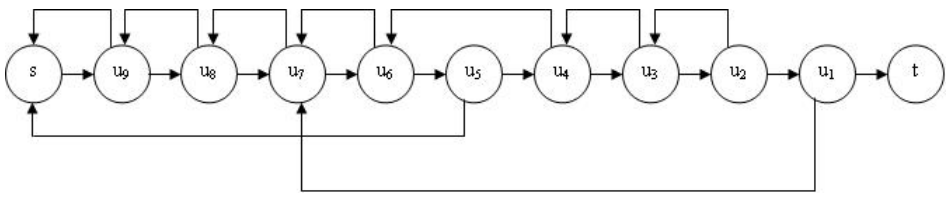
- $V(F[\pi^*]) = \{t = u_0, u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10} = s\}$
- προστίθενται οι ακμές: $(s, u_9), (u_9, u_8), (u_8, u_7), (u_7, u_6), (u_6, u_5), (u_5, u_4), (u_4, u_3), (u_3, u_2), (u_2, u_1), (u_1, t)$.

  Επομένως, το γράφημα μέχρι στιγμής είναι το εξής:

Σχήμα 3.14: Κατευθυνόμενο γράφημα από το βήμα 3 του Αλγορίθμου Encode_SIP_to_RPG

Βήμα 4: προστίθενται οι ακόλουθες ακμές: $(u_1, u_7), (u_2, u_3), (u_3, u_4), (u_4, u_6), (u_5, s), (u_6, u_7), (u_7, u_8), (u_8, u_9), (u_9, s)$

Άρα, το αναγώγιμο μεταθετικό γράφημα που προκύπτει είναι το εξής:



Σχήμα 3.15: Αναγώγιμο μεταθετικό γράφημα του αριθμού 8

Παρατηρούμε ότι η αυτο-αντιστρέψιμη μετάθεση που δημιουργείται από τον Αλγόριθμο 4 έχει πάντα περιττό μήκος, άρα και το γράφημα που δημιουργείται έχει πάντοτε περιττό μήκος. Κάτι τέτοιο σημαίνει ότι οποιαδήποτε προσθήκη ή διαγραφή κόμβου στο γράφημα γίνεται εύκολα αντιληπτή.

Οι Αλγόριθμοι 4 και 5 αποτελούν τον συνολικό αλγόριθμο κωδικοποίησης ενός ακέραιου αριθμού σε αναγώγιμο μεταθετικό γράφημα. Στη συνέχεια αυτής της παραγράφου θα δοθούν τα βήματα της αντίστροφης διαδικασίας, δηλαδή το πώς αποκωδικοποιούμε ένα αναγώγιμο μεταθετικό γράφημα στον ακέραιο αριθμό που αυτό αντιστοιχεί.

Η αποκωδικοποίηση στον Αλγόριθμο `CN` έτσι όπως προτάθηκε από τους Μ. Χρόνη και Σ.Δ. Νικολόπουλο μπορούμε να πούμε ότι χωρίζεται και αυτή σε δύο φάσεις. Σε πρώτη φάση το αναγώγιμο μεταθετικό γράφημα αποκωδικοποιείται στην αυτο-αντιστρέψιμη ακολουθία που αυτό αντιστοιχεί κάνοντας ορισμένες μετατροπές στις ακμές του γραφήματος και αναζήτηση κατά βάθος σε δένδρα. Στην δεύτερη φάση της αποκωδικοποίησης η αυτο-αντιστρέψιμη μετάθεση που πήραμε από την πρώτη φάση μετατρέπεται σε έναν ακέραιο κάνοντας χρήση της αύξουσας αναπαράστασης κύκλων.

Αρχίζουμε, παραθέτοντας τα βήματα της πρώτης φάσης καθώς επίσης και ένα παράδειγμα για να τα κατανοήσουμε καλύτερα.

Αλγόριθμος 13 Αλγόριθμος Decode_RPG_to_SIP

---

1. Διάγραψε τις κατευθυνόμενες ακμές (u_{i+1}, u_i) από το σύνολο ακμών $E(F[\pi^*])$, $1 \leq i \leq n$, και τον κόμβο $t = u_0$ από το $V(F[\pi^*])$
2. Άλλαξε την κατεύθυνση των ακμών που έμειναν στο γράφημα $F[\pi^*]$. Το γράφημα που προκύπτει είναι ένα δένδρο $T[\pi^*]$
3. Εφάρμοσε αναζήτηση κατά βάθος στο δένδρο $T[\pi^*]$ ξεκινώντας από τον κόμβο s και πηγαίνοντας πάντα προς το παιδί με την μικρότερη ετικέτα
4. Ταξινόμησε τους κόμβους $s, u_1, u_2, ..., u_n$ του δένδρου $T[\pi^*]$ σύμφωνα με τον χρόνο ανακάλυψης τους $d[]$ στην αναζήτηση κατά βάθος και θέσε $\pi = (u_{d[0]}, u_{d[1]}, ...u_{d[n]})$, όπου $u_{d[0]} = s$ και $d[0] < d[1] < ... < d[n]$
5. Διάγραψε τον κόμβο s από την διάταξη π
6. Επέστρεψε $\pi^* = \pi$

Την αναζήτηση κατά βάθος του τρίτου βήματος του αλγορίθμου την πραγματοποιούμε εφαρμόζοντας τον ακόλουθο αλγόριθμο:

Αλγόριθμος 14 Αλγόριθμος Αναζήτησης Κατά Βάθος (DFS)

---

1. Για κάθε $u \in V$ θέσε $color[u] = 0$
2. Θέσε $time = 0$
3. Κάλεσε τον αλγόριθμο DFS_visit με όρισμα των αρχικό κόμβο s

---

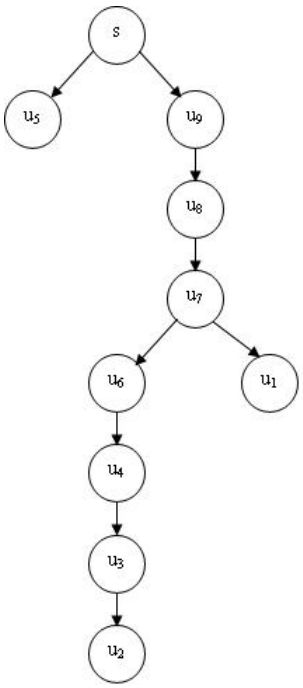
Αλγόριθμος 15 Αλγόριθμος DFS_visit(node)

1. Θέσε $color[node] = 1$
2. Θέσε $d[node] = time$
3. Θέσε time++
4. Αποθήκευσε τους γείτονες του κόμβου node σε μια λιστα
5. Ταξινόμησε την λίστα κατά αύξουσα σειρά σύμφωνα με τις ετικέτες των κόμβων
6. Θέσε $size$ =μέγεθος της λίστας
7. Για $i = 0$ μέχρι $i < size$ επανάλαβε
   Πάρε τον κόμβο που βρίσκεται στην θέση i της λιστας, έστω k
   Αν $color[k] = 0$ τότε κάλεσε DFS_visit(k)

Έτσι, αν πάρουμε το γράφημα του Σχήματος 3.15 ο Αλγόριθμος 13 θα εφαρμοστεί ως εξής:

Βήμα 1: διαγράφονται οι ακμές $(s, u_9), (u_9, u_8), (u_8, u_7), (u_7, u_6), (u_6, u_5), (u_5, u_4), (u_4, u_3), (u_3, u_2), (u_2, u_1), (u_1, t)$ και ο κόμβος t

Βήμα 2: αντιστρέφονται όσες ακμές έχουν απομείνει στο γράφημα $F[\pi^*]$:



Σχήμα 3.16: Δένδρο που προκύπτει από το βήμα 2 του Αλγορίθμου Decode_RPG_to_SIP για $w = 8$

Το γράφημα που προκύπτει είναι ένα δένδρο με ρίζα το s

Βήμα 3: γίνεται DFS αναζήτηση ξεκινόντας από τον s

Βήμα 4: υπολογίζονται οι χρόνοι όπου ανακαλύφθηκαν οι κόμβοι στην DFS αναζήτηση και αυτοί είναι:

- για τον κόμβο s είναι d[0]=0
- για τον κόμβο u_1 είναι d[5]=5
- για τον κόμβο u_2 είναι d[9]=9
- για τον κόμβο u_3 είναι d[8]=8
- για τον κόμβο u_4 είναι d[7]=7
- για τον κόμβο u_5 είναι d[1]=1
- για τον κόμβο u_6 είναι d[6]=6
- για τον κόμβο u_7 είναι d[4]=4
- για τον κόμβο u_8 είναι d[3]=3
- για τον κόμβο u_9 είναι d[2]=2

Άρα η μετάθεση $\pi = (s, u_5, u_9, u_8, u_7, u_1, u_6, u_4, u_3, u_2)$

Βήμα 5: διαγράφεται ο κόμβος s από την π

Βήμα 6: επιστρέφεται η $\pi^* = (u_5, u_9, u_8, u_7, u_1, u_6, u_4, u_3, u_2)$, δηλαδή $\pi^* = (5, 9, 8, 7, 1, 6, 4, 3, 2)$.

Τέλος, για να ολοκληρωθεί η αποκωδικοποίηση πρέπει να αναφερθούν τα βήματα μετατροπής μιας αυτο-αντιστρέψιμης μετάθεσης σε έναν ακέραιο, δηλαδή τα βήματα που υλοποιούν την δεύτερη φάση της αποκωδικοποίησης. Αυτά παρουσιάζονται στον παρακάτω αλγόριθμο.

Αλγόριθμος 16 Αλγόριθμος Decode_SIP_to_W

1. Υπολόγισε την αύξουσα αναπαράσταση κύκλων $C = (c_1, c_2, ..., c_k)$ της αυτο-αντιστρέψιμης μετάθεσης $\pi^* = (\pi_1, \pi_2, ..., \pi_{n'})$, όπου $n' = 2n + 1$. Δηλαδή θέλουμε να υπολογίσουμε το εξής: $c_1 \prec c_2 \prec ... \prec c_k$
2. Θέσε $i = 1$ και $j = n'$
3. Κατασκεύασε την μετάθεση π^b μήκους n' ως εξής:
   Όσο το σύνολο C δεν είναι άδειο κάνε το εξής:
   Επίλεξε το ελάχιστο στοιχείο c από την ακολουθία C
   Αν ο επιλεγμένος κύκλος είναι μεγέθους 2 και έστω ότι είναι $c = (a, b)$ τότε:
   $\pi_i = b$ και $\pi_j = a$
   $i = i + 1$ και $j = j - 1$
   Αλλιώς αν ο επιλεγμένος κύκλος είναι μεγέθους 1 και έστω ότι είναι $c = (a)$ τότε:
   $\pi_i = a$ και $i = i + 1$
   Αφαίρεσε τον κύκλο c από το σύνολο C
4. Βρες την πρώτη αύξουσα υποακολουθία $X = (\pi_1, \pi_2, ...\pi_k)$ και μετά την φθίνουσα υποακολουθία $Y = (\pi_{k+1}, \pi_{k+2}, ..., \pi_{k'})$ από το π
5. Κατασκεύασε την δυαδική ακολουθία $B^* = (b_1, b_2, ..., b_{n'})$ ως εξής:
   Θέσε τα μηδενικά στις θέσεις $\pi_1, \pi_2, ...\pi_k$ και τους άσσους στις θέσεις $\pi_{k+1}, \pi_{k+2}, ..., \pi_{k'}$
6. Υπολόγισε το αντίστροφο του B^* το οποίο είναι το $B' = (b_1, b_2, ..., b_n, b_{n+1}, ..., b_{n'-1}, b_{n'})$
7. Μετάτρεψε τον $B' = (b_{n+1}, ..., b_{n'-1})$ σε δεκαδικό και αυτός είναι ο ακέραιος

Ο παραπάνω αλγόριθμος με είσοδο την αυτο-αντιστρέψιμη ακολουθία $\pi^* = (5, 9, 8, 7, 1, 6, 4, 3, 2)$, θα εκτελούνταν ως εξής:

Βήμα 1: υπολογίζεται η αύξουσα αναπαράσταση κύκλων $C = \{c_1 = (1, 5), c_2 = (2, 9), c_3 = (3, 8), c_4 = (4, 7), c_5 = (6)\}$

Βήμα 2: τίθεται $i = 1$ και $j = 9$

Βήμα 3: σε αυτό το βήμα επιλέγονται οι κύκλοι και κατασκευάζεται η ακολουθία π^b. Στην αρχή επιλέγεται ο κύκλος c_1 και τίθεται $\pi_1 = 5$ και $\pi_9 = 1$. Αυξάνεται το i κατά ένα και μειώνεται το j κατά ένα. Έπειτα επιλέγεται ο κύκλος c_2 και τίθεται $\pi_2 = 9$ και $\pi_8 = 2$. Η διαδικασία συνεχίζεται με τον ίδιο τρόπο. Οπότε στο τέλος το $\pi^b = (5, 9, 8, 7, 6, 4, 3, 2, 1)$

Βήμα 4: υπολογίζεται η πρώτη αύξουσα υποακολουθία $X = (5, 9)$ και φθίνουσα υποακολουθία $Y = (8, 7, 6, 4, 3, 2, 1)$

Βήμα 5: κατασκευάζεται η δυαδική ακολουθία $B^* = (1, 1, 1, 1, 0, 1, 1, 1, 0)$

Βήμα 6: υπολογίζεται η ανάστροφη της B^* η οποία είναι $B' = (0, 0, 0, 0, 1, 0, 0, 0, 1)$

Βήμα 7: μετατρέπεται ο δυαδικός αριθμός $B = 1000$ σε δεκαδικό. Επομένως αυτή η αυτο-αντιστρέψιμη μετάθεση αντιστοιχεί στον αριθμό 8.

Χάριν του παραπάνω τρόπου κατασκευής των αναγώγιμων μεταθετικών γραφημάτων γίνονται εύκολα αντιληπτές δύο βασικές ιδιότητες. Πρώτον παρατηρούμε ότι μπορεί να γίνει αντιληπτή κάποια επίθεση που θα γίνει στο γράφημα καθώς υπάρχει πιθανότητα κατά την αποκωδικοποίηση της αυτο-αντιστρέψιμης μετάθεσης στον ακέραιο που αυτή αντιστοιχεί να πάψει να υφίσταται η διτονική ιδιότητα και δεύτερον, όπως αναφέρθηκε στον Αλγόριθμο 11 παίρνουμε την τιμή του υδατοσήμου στη δυαδική του μορφή. Αν το μήκος αυτού του δυαδικού αριθμού υποθέσουμε ότι είναι n τότε ο αλγόριθμος το αλλάζει και το κάνει ίσο με $2n+1$ βάζοντας μπροστά από τον αριθμό αυτό n μηδενικά και προσθέτοντας έναν άσσο στα δεξιά. Αυτό όμως έχει ως αποτέλεσμα τη διευκόλυνση του να αξιολογήσουμε αν κάποιος έχει κάνει κάποια επίθεση καθώς υπάρχει μεγάλη πιθανότητα κατά την αποκωδικοποίηση να εμφανιστεί κάποιος άσσος στα πρώτα n ψηφία από αριστερά ή ακόμα και να έχει αλλάξει τον άσσο που βρισκόταν στα δεξιά σε μηδέν. Έτσι, αυτές οι δύο ιδιότητες σε συνδυασμό με την ιδιότητα που αναφέρθηκε πριν, ότι δηλαδή τα γραφήματα έχουν περιττό αριθμό κόμβων, αποτελούν τις τρεις βασικότερες ιδιότητες στις οποίες στηριζόμαστε για να διακρίνουμε τυχόν επιθέσεις.

Έχοντας αναλύσει τα βήματα του Αλγορίθμου `CN` και έχοντας αναφέρει τις βασικές ιδιότητες των γραφημάτων που δημιουργούνται από αυτόν κρίνεται αναγκαίο να εξετάσουμε αν είναι "καλή" η όλη κωδικοποίηση καθώς και ο τύπος γραφημάτων που χρησιμοποιείται από αυτόν. Όπως αναφέρθηκε η μετατροπή ενός ακέραιου σε γράφημα και το αντίστροφο περιγράφεται από δύο συναρτήσεις, την *encode* και την *decode* αντίστοιχα, οι οποίες αποτελούν τον *κωδικοποιητή* ενός γραφήματος. Η κλάση γραφημάτων που θα επιλέξουμε καθώς και ο *κωδικοποιητής* πρέπει να έχει τέσσερις βασικές ιδιότητες. Αυτές οι ιδιότητες αναφέρθηκαν στην Ενότητα 2.4.7. Στον Αλγόριθμο `CN` που περιγράψαμε σε αυτή την ενότητα η συνάρτηση *encode*, η οποία μετατρέπει έναν ακέραιο σε αναγώγιμο μεταθετικό γράφημα, είναι ο συνδυασμός των Αλγορίθμων Encode_W_to_SIP και Encode_SIP_to_RPG, ενώ η συνάρτηση *decode*, η οποία αποκωδικοποιεί αυτό το γράφημα στον ακέραιο που αυτό αντιστοιχεί, είναι ο συνδυασμός των Αλγορίθμων Decode_RPG_to_SIP και Decode_SIP_to_W. Επομένως, ο *κωδικοποιητής* σε αυτόν τον αλγόριθμο είναι αυτές οι τέσσερις συναρτήσεις και η κλάση των γραφημάτων που χρησιμοποιείται είναι τα αναγώγιμα μεταθετικά γραφήματα.

Στα γραφήματα που δημιουργούνται με τον Αλγόριθμο `CN` κάθε κόμβος έχει πλήθος εξερχόμενων ακμών ακριβώς δύο, το οποίο σημαίνει ότι έχουν μικρό πλήθος εξερχόμενων ακμών και μοιάζουν πάρα πολύ με τα γραφήματα των προγραμμάτων. Έτσι, μπορούμε να συμπεράνουμε ότι ο τύπος γραφημάτων που χρησιμοποιείται είναι κατάλληλος, το οποίο συνεπάγεται ότι ισχύει η πρώτη ιδιότητα που αναφέρθηκε στην Παράγραφο 2.4.7. Επιπλέον γίνεται εύκολα αντιληπτή οποιαδήποτε επίθεση στις ακμές του γραφήματος καθώς θα παραβιάζεται είτε κάποια ακμή από αυτές που διατηρούν την λίστα των κόμβων, είτε κάποια από αυτές που προστέθηκαν από την επιλογή των μέγιστων άμεσων κόμβων. Άρα, τα γραφήματα που δημιουργούνται έχουν μεγάλο βαθμό ανθεκτικότητας σε επιθέσεις γεγονός που αποτελεί τη δεύτερη ιδιότητα που θέλουμε να ικανοποιείται. Η τρίτη ιδιότητα ικανοποιεί-

ται καθώς το μέγεθος των γραφημάτων που δημιουργείται από αυτόν τον αλγόριθμο είναι μικρό, άρα και το υδατογραφημένο πρόγραμμα δεν θα μεγαλώσει ριζικά σε σχέση με το αρχικό. Τέλος, ο *κωδικοποιητής* που χρησιμοποιείται σε αυτόν τον αλγόριθμο είναι αποτελεσματικός καθώς και οι δύο συναρτήσεις $Encode\_SIP\_to\_RPG$ και $Decode\_RPG\_to\_SIP$ απαιτούν $O(n)$ χρόνο και χώρο η κάθε μία. Άρα, εφόσον ισχύουν και οι τέσσερις ιδιότητες μπορούμε να ισχυριστούμε ότι η επιλογή αυτής της κλάσης γραφημάτων καθώς και του κωδικοποιητή που παρουσιάστηκε παραπάνω είναι"καλή".

Κεφαλαιο 4

Συστημα Υδατοσημανσης Λογισμικου

4.1 Εισαγωγή

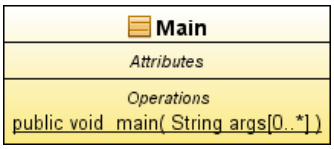
Στο Κεφάλαιο 3 περιγράψαμε αναλυτικά τα βήματα δύο αλγορίθμων, οι οποίοι μετατρέπουν την τιμή ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα. Στο παρόν κεφάλαιο θα παρουσιάσουμε το Σύστημα *WaterGraph* που αναπτύξαμε το οποίο υλοποιεί αυτούς τους δύο αλγορίθμους κάνοντας χρήση της γλώσσας προγραμματισμού *Java*. Με αυτό το σύστημα δίνεται η δυνατότητα στον χρήστη να κωδικοποιήσει μια τιμή ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα χρησιμοποιώντας οποιονδήποτε αλγόριθμο απο τους δύο. Επιπλέον, προσφέρει δύο βασικές λειτουργίες, πρώτον την γραφική αναπαράσταση του γραφήματος κάνοντας χρήση της βιβλιοθήκης *JUNG* και δεύτερον τη δυνατότητα διεξαγωγής επιθέσεων στο εκάστοτε γράφημα που δημιουργείται. Επίσης, δίνεται η δυνατότητα της αποκωδικοποίησης ενός γραφήματος που δημιουργήσαμε στην αυτο-αντιστρέψιμη μετάθεση που αυτό αντιστοιχεί και με τη σειρά της αυτή στην αντίστοιχη τιμή του υδατοσήμου. Τέλος, ο χρήστης είναι ικανός να εξετάσει πειραματικά την ανθεκτικότητα των δομών που δημιουργούνται από τον κάθε αλγόριθμο ως προς μια συγκεκριμένη επίθεση καθώς επίσης και η δυνατότητα εξαγωγής στατιστικών αποτελεσμάτων από αυτά τα πειράματα.

Έτσι, στην Ενότητα 4.2 θα δώσουμε μια αναλυτική περιγραφή του σχεδιασμού του Συστήματος *WaterGraph*, ενώ στην Ενότητα 4.3 θα παρουσιάσουμε αυτό το σύστημα και θα περιγράψουμε τον τρόπο χρήσης του.

4.2 Σχεδιασμός Συστήματος

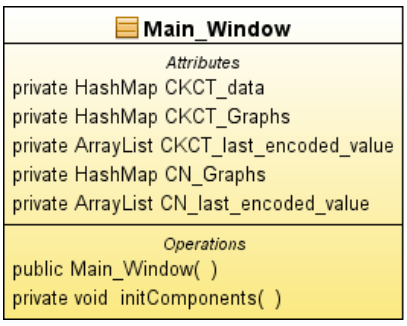
Στην παρούσα ενότητα θα αναφέρουμε τον τρόπο με τον οποίο σχεδιάσαμε το Σύστημα *WaterGraph*, δηλαδή θα περιγράψουμε το σύνολο των κλάσεων καθώς επίσης και την λειτουργία που επιτελεί η κάθε κλάση. Για να επιτευχθεί αυτό γίνεται χρήση της γλώσσας UML, η οποία είναι μια γλώσσα μοντελοποίησης με την οποία θα προσδιορίσουμε, θα οπτικοποιήσουμε, θα αναπτύξουμε και θα τεκμηριώσουμε το σύστημα μας. Για την καλύτερη κατανόηση έχουμε διασπάσει τον σχεδιασμό σε τέσσερα μέρη. Το πρώτο μέρος αναφέρεται στις λειτουργίες κωδικοποίησης, αποκωδικοποίησης και εφαρμογής επιθέσεων του Αλγορίθμου `CKCT`, ενώ το δεύτερο μέρος πάλι στις ίδιες λειτουργίες, του Αλγορίθμου όμως `CN`. Τέλος, τα μέρη τρία και τέσσερα αναφέρονται στο πειραματικό στάδιο όπου ελέγχεται η ανθεκτικότητα που έχουν σε επιθέσεις οι δύο Αλγόριθμοι `CKCT` και `CN` αντίστοιχα. Αξίζει να σημειωθεί ότι και στα τέσσερα αυτά UML διαγράμματα που θα παρουσιάσουμε υπάρχει κάτι κοινό και αυτό είναι οι κλάσεις Main, Main_Window και Mode. Αυτές οι κλάσεις αποτελούν τον κύριο άξονα του συστήματος μας καθώς πρέπει να προϋπάρχουν για να δημιουργηθούν αντικείμενα και των άλλων κλάσεων που θα αναφέρουμε στη συνέχεια.

Η κλάση Main, η οποία φαίνεται στο Σχήμα 4.1 είναι η κλάση εκκίνησης του συστήματος μας. Δημιουργεί ένα αντικείμενο της κλάσης Main_Window και την καλεί.



Σχήμα 4.1: Κλάση Main

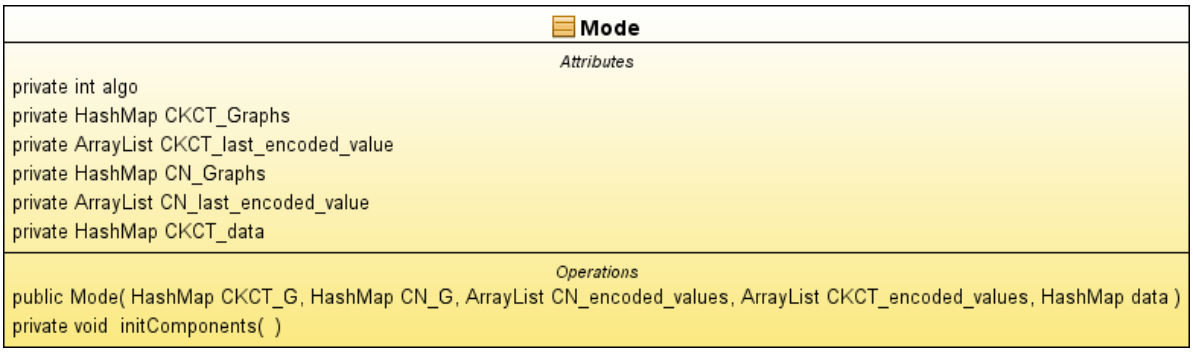
Η κλάση Main_Window του Σχήματος 4.2 είναι ένα στοιχείο γραφικής διεπαφής με το οποίο ο χρήστης έχει την δυνατότητα είτε να εισέλθει στο πρόγραμμα, καλώντας την κλάση Mode, είτε να το τερματίσει.



Σχήμα 4.2: Κλάση Main_Window

Τέλος, η κλάση Mode είναι και αυτή ένα στοιχείο γραφικής διεπαφής όπου ο χρήστης έχει την δυνατότητα να επιλέξει έναν από τους δύο αλγορίθμους που αναφέρθηκαν στο Κεφάλαιο 3 και είτε να εκτελέσει τις λειτουργίες κωδικοποίησης, αποκωδικοποίησης και επιθέσεων, είτε να εξετάσει πειραματικά την ανθεκτικότητα των δομών που δημιουργούν

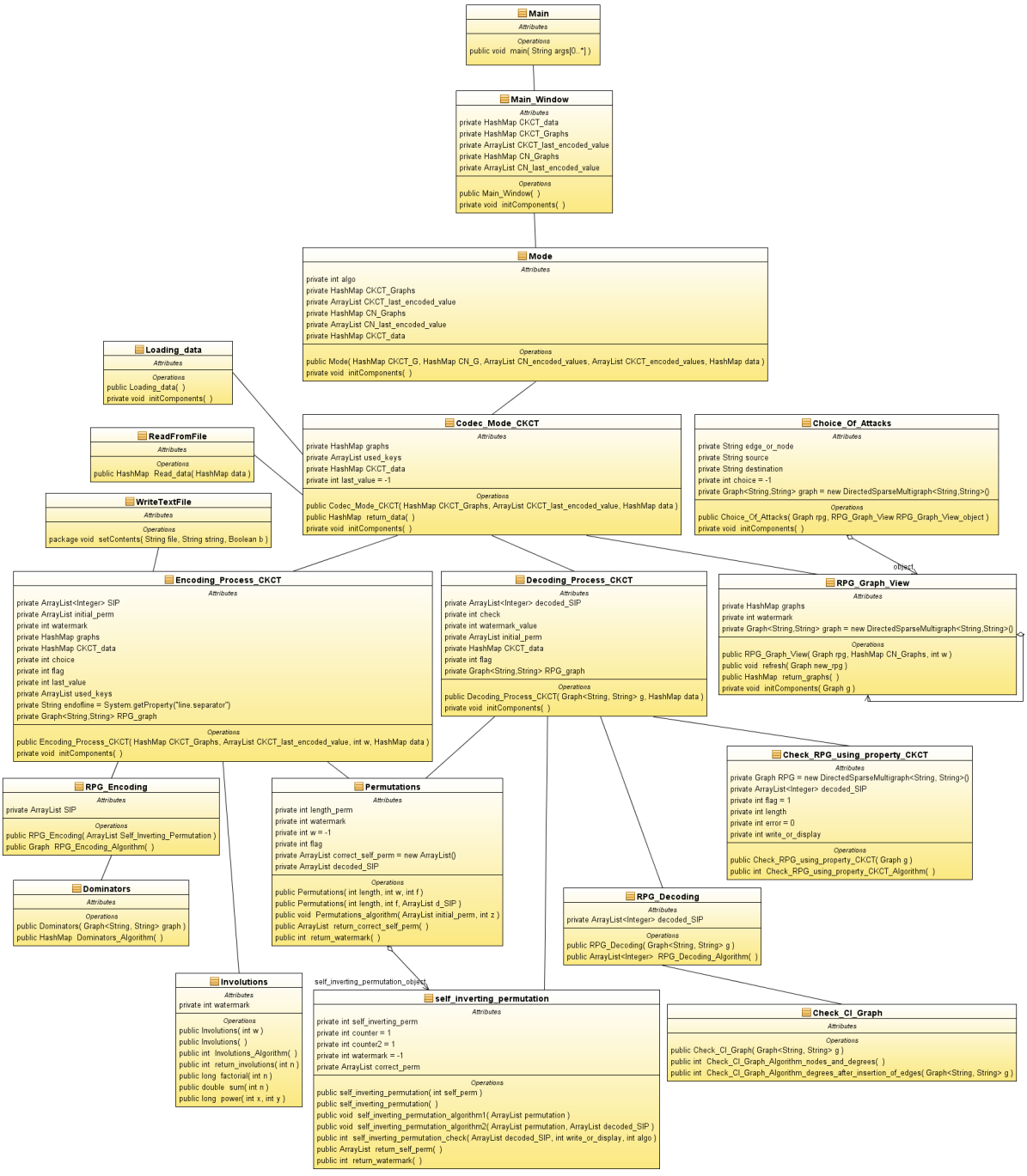
αυτοί οι αλγόριθμοι σε επιθέσεις και να εξάγει αυτά τα αποτελέσματα. Η κλάση Mode παρουσιάζεται στο Σχήμα 4.3.



Σχήμα 4.3: Κλάση Mode

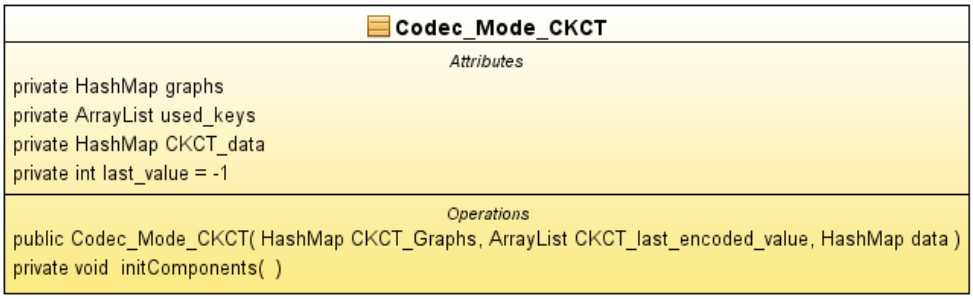
4.2.1 Πρώτο Μέρος Σχεδίασης - Κωδικοποιητής Αλγορίθμου CKCT

Έχοντας αναλύσει τις τρεις βασικές κλάσεις, οι οποίες θα παρουσιάζονται και στα τέσσερα μέρη της σχεδίασης μπορούμε να προχωρήσουμε στην παρουσίαση και στην ανάλυση του πρώτου μέρους. Το σχέδιο αυτού του μέρους φαίνεται στο παρακάτω σχήμα.



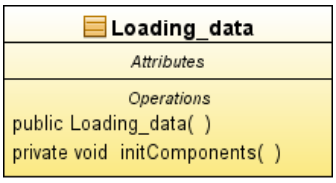
Σχήμα 4.4: Πρώτο μέρος της σχεδίασης

Όπως αναφέραμε και προηγουμένως αυτό το μέρος αποτελεί την κωδικοποίηση, αποκωδικοποίηση και εφαρμογή επιθέσεων του Αλγορίθμου CKCT. Ο χρήστης έχει την δυνατότητα να εφαρμόσει κάποια από τις προαναφερθείσες λειτουργίες μέσω της κλάσης Codec_Mode_CKCT η οποία παρουσιάζεται στο Σχήμα 4.5.



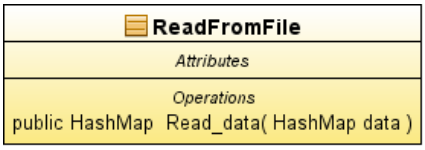
Σχήμα 4.5: Κλάση Codec_Mode_CKCT

Από την στιγμή που δημιουργείται ένα αντικείμενο της κλάσης Codec_Mode_CKCT αυτομάτως δημιουργούνται και δύο αντικείμενα των κλάσεων Loading_data (βλέπε Σχήμα 4.6) και ReadFromFile (βλέπε Σχήμα 4.7).



Σχήμα 4.6: Κλάση Loading_data

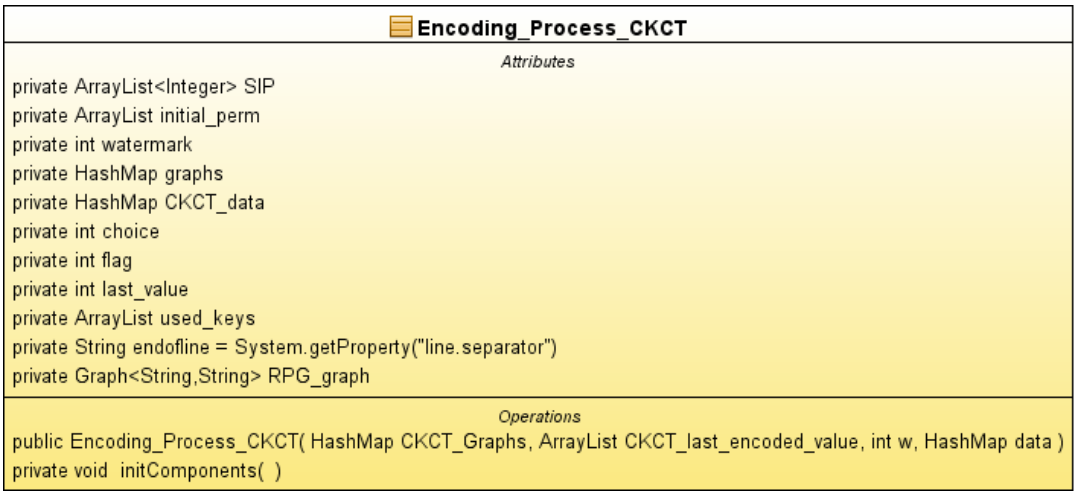
Η κλάση Loading_data υλοποιεί μια μπάρα προόδου και εμφανίζεται επειδή φορτώνονται από ένα αρχείο στην μνήμη του υπολογιστή όλες οι αυτο-αντιστρέψιμες ακολουθίες για τιμές του υδατοσήμου από 1 έως 140152 καθώς το να δημιουργούμε κάθε φορά τις αυτο-αντιστρέψιμες ακολουθίες σύμφωνα με τον Αλγόριθμο CKCT είναι αρκετά χρονοβόρο σύμφωνα με την υλοποίηση μας. Η κλάση ReadFromFile επιλέγει το αρχείο από το οποίο θα διαβάσει τα δεδομένα και τα φορτώνει στην μνήμη του υπολογιστή.



Σχήμα 4.7: Κλάση ReadFromFile

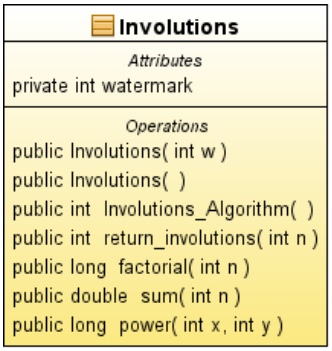
Εφόσον ολοκληρωθεί η διαδικασία φόρτωσης των δεδομένων ο χρήστης έχει τη δυνατότητα πλέον να επιλέξει μια από τις τρεις λειτουργίες που αναφέρθηκαν στην αρχή του δεύτερου μέρους της σχεδίασης του συστήματος μας. Αν επιλεγεί η λειτουργία της κωδικοποίησης τότε

δημιουργείται ένα αντικείμενο της κλάσης Encoding_Process_CKCT (βλέπε Σχήμα 4.8), η οποία αποτελεί μια γραφική διεπαφή με τον χρήστη.



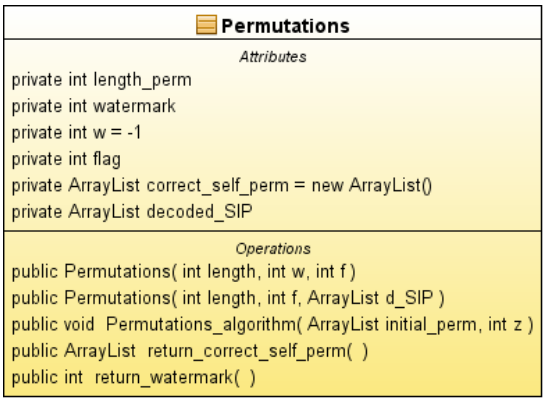
Σχήμα 4.8: Κλάση Encoding_Process_CKCT

Για την κωδικοποίηση της τιμής του υδατοσήμου σε αυτο-αντιστρέψιμη ακολουθία η κλάση Encoding_Process_CKCT αρχικά ελέγχει αν υπάρχει φορτωμένη στην μνήμη η τιμή αυτή. Αν υπάρχει, επιστρέφει την αυτο-αντιστρέψιμη μετάθεση, αλλιώς δημιουργεί αντικείμενα των κλάσεων Involutions, Permutations και self_inverting_permutation έτσι ώστε να υπολογίσει την αυτο-αντιστρέψιμη μετάθεση. Η κλάση Involutions υλοποιεί την Εξίσωση 1 που παρουσιάστηκε στην Παράγραφο 3.4 και παρουσιάζεται στο Σχήμα 4.9.

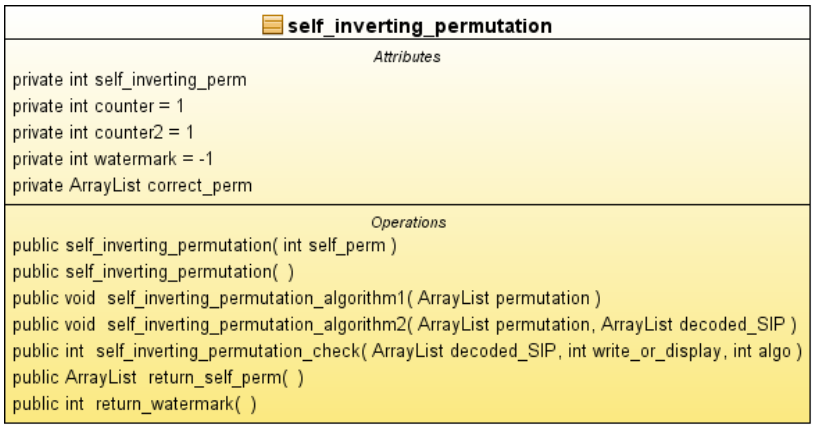


Σχήμα 4.9: Κλάση Encoding_Process_CKCT

Οι κλάσεις Permutations και self_inverting_permutation υλοποιούν τον Αλγόριθμο 7 και παρουσιάζονται στο Σχήμα 4.10.



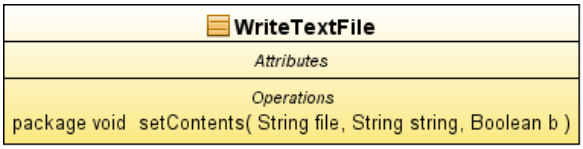
(α) Permutations



(β) self_inverting_permutation

Σχήμα 4.10: Κλάσεις υλοποίησης Αλγορίθμου 7

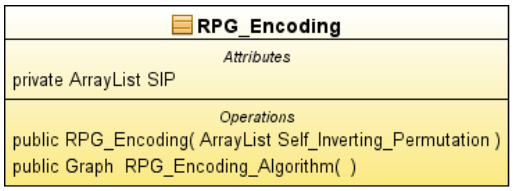
Πρέπει να σημειωθεί ότι κάθε αυτο-αντιστρέψιμη μετάθεση που υπολογίζεται μέσω των παραπάνω κλάσεων φορτώνεται στην μνήμη και επιπλέον γράφεται στο αρχείο μέσω της κλάσης WriteTextFile που φαίνεται στο παρακάτω σχήμα.



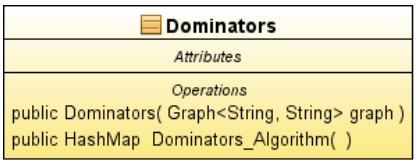
Σχήμα 4.11: Κλάση WriteTextFile

Η εγγραφή γίνεται για μελλοντική χρήση της τιμής του υδατοσήμου χωρίς να σπαταλήσουμε χρόνο για τον υπολογισμό της αυτο-αντιστρέψιμης μετάθεσης που αυτό αντιστοιχεί. Για την ολοκλήρωση της διαδικασίας της κωδικοποίησης του Αλγορίθμου CKCT δημιουργούμε ένα αντικείμενο της κλάσης RPG_Encoding και την καλούμε καθώς αυτή υλοποιεί τον Αλγόριθμο 8. Όπως αναφέρθηκε σε αυτόν τον αλγόριθμο στο Κεφάλαιο 3 πρέπει να γίνει υπολογισμός των κυρίαρχων κόμβων σύμφωνα με τον Αλγόριθμο 9. Η υλοποίηση του τελευταίου γίνεται από την κλάση Dominators η οποία συμπληρώνει την κλάση RPG_Encoding. Και οι

δύο αυτές κλάσεις, οι οποίες παρουσιάζονται στο Σχήμα 4.12, αποτελούν την κωδικοποίηση της αυτο-αντιστρέψιμης μετάθεσης σε αναγώγιμο μεταθετικό γράφημα.



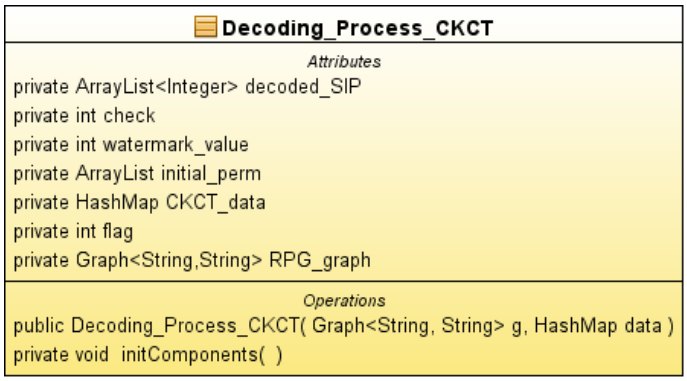
(α) RPG_Encoding



(β) Dominators

Σχήμα 4.12: Κλάσεις υλοποίησης κωδικοποίησης αναγώγιμου μεταθετικού γραφήματος Αλγορίθμου CKCT

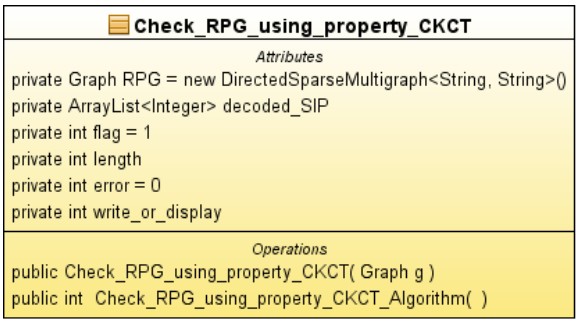
Με αυτές τις κλάσεις ολοκληρώνεται η διαδικασία της κωδικοποίησης της τιμής ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα σύμφωνα με τα βήματα του Αλγορίθμου CKCT. Αν τώρα ο χρήστης επιλέξει να εκτελέσει την αντίστροφη διαδικασία, δηλαδή την αποκωδικοποίηση ενός αναγώγιμου μεταθετικού γραφήματος στην αντίστοιχη τιμή του υδατοσήμου τότε δημιουργείται ένα αντικείμενο της κλάσης Decoding_Process_CKCT (βλέπε Σχήμα 4.13) και καλείται αυτή η κλάση.



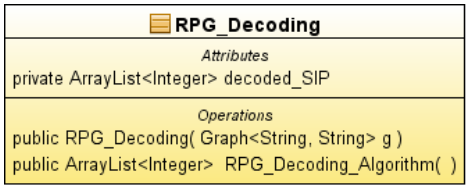
Σχήμα 4.13: Κλάση Decoding_Process_CKCT

Αυτή η κλάση αποτελεί μια γραφική διεπαφή και δίνει στον χρήστη την δυνατότητα να ελέγξει αν το γράφημα είναι αναγώγιμο, να αποκωδικοποιήσει το γράφημα στην αυτο-αντιστρέψιμη μετάθεση του και επίσης να αποκωδικοποιήσει την μετάθεση αυτή στην αντίστοιχη τιμή του υδατοσήμου. Οι πρώτες δύο δυνατότητες υλοποιούνται αντίστοιχα από τις κλάσεις Check_RPG_using_property_CKCT και RPG_Decoding (βλέπε Σχήμα 4.14). Η αποκωδικοποίηση της αυτο-αντιστρέψιμης μετάθεσης στην αντίστοιχη τιμή του υδατοσή-

μου επιτυγχάνεται πάλι μέσω των κλάσεων Permutations και self_inverting_permutation που παρουσιάσαμε προηγουμένως στο Σχήμα 4.10.



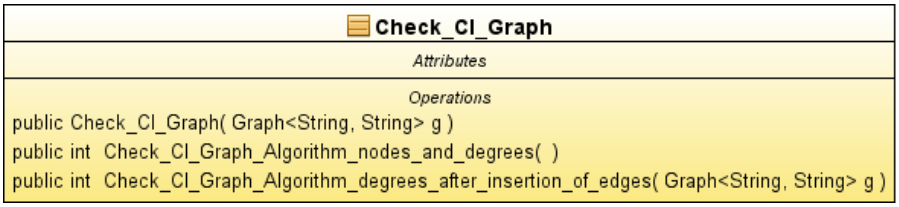
(α) Check_RPG_using_property_CKCT



(β) RPG_Decoding

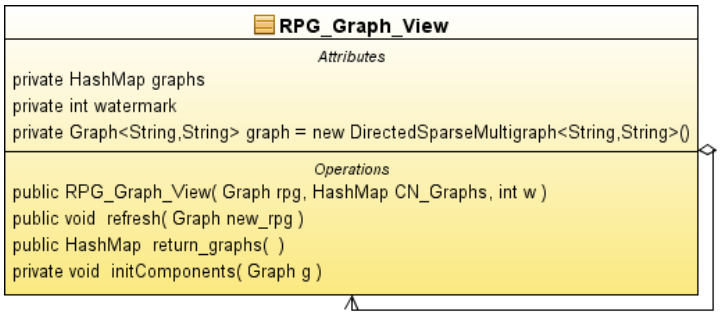
Σχήμα 4.14: Κλάσεις αποκωδικοποίησης αναγώγιμου μεταθετικού γραφήματος Αλγορίθμου CKCT

Πρέπει να τονίσουμε ότι η κλάση Decoding_Process_CKCT δημιουργεί ένα αντικείμενο της κλάσης self_inverting_permutation έτσι ώστε να ελέγξει αν ειναι αυτο-αντιστρέψιμη η μετάθεση μετά την αποκωδικοποίηση και επίσης η RPG_Decoding δημιουργεί αντικείμενα της κλάσης Check_Cl_Graph (βλέπε Σχήμα 4.15) έτσι ώστε να ελέγχει τους βαθμούς των κόμβων κατά την αποκωδικοποίηση.

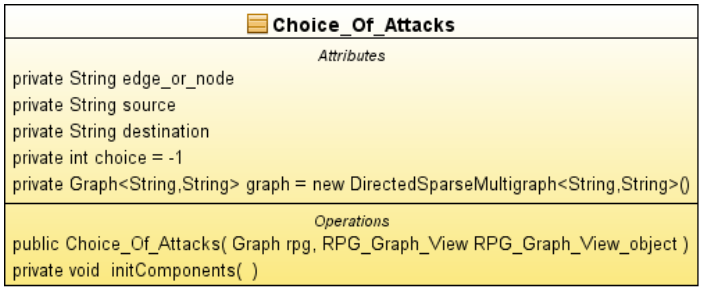


Σχήμα 4.15: Κλάση Check_Cl_Graph

Τέλος, ο χρήστης έχει την δυνατότητα να επιλέξει μια ήδη κωδικοποιημένη τιμή υδατοσήμου και να εφαρμόσει κάποια επίθεση σε αυτό. Αυτή η λειτουργία υλοποιείται από την κλάση RPG_Graph_View η οποία εμφανίζει γραφικά το γράφημα του υδατοσήμου που επέλεξε και δίνει την δυνατότητα να γίνουν επιθέσεις σε αυτό. Για την πραγματοποίηση αυτών των επιθέσεων η κλάση RPG_Graph_View συνδέεται με σχέση συνάθροισης με την κλάση Choice_Of_Attacks. Και οι δύο αυτές κλάσεις φαίνονται στο Σχήμα 4.16.



(α) RPG_Graph_View

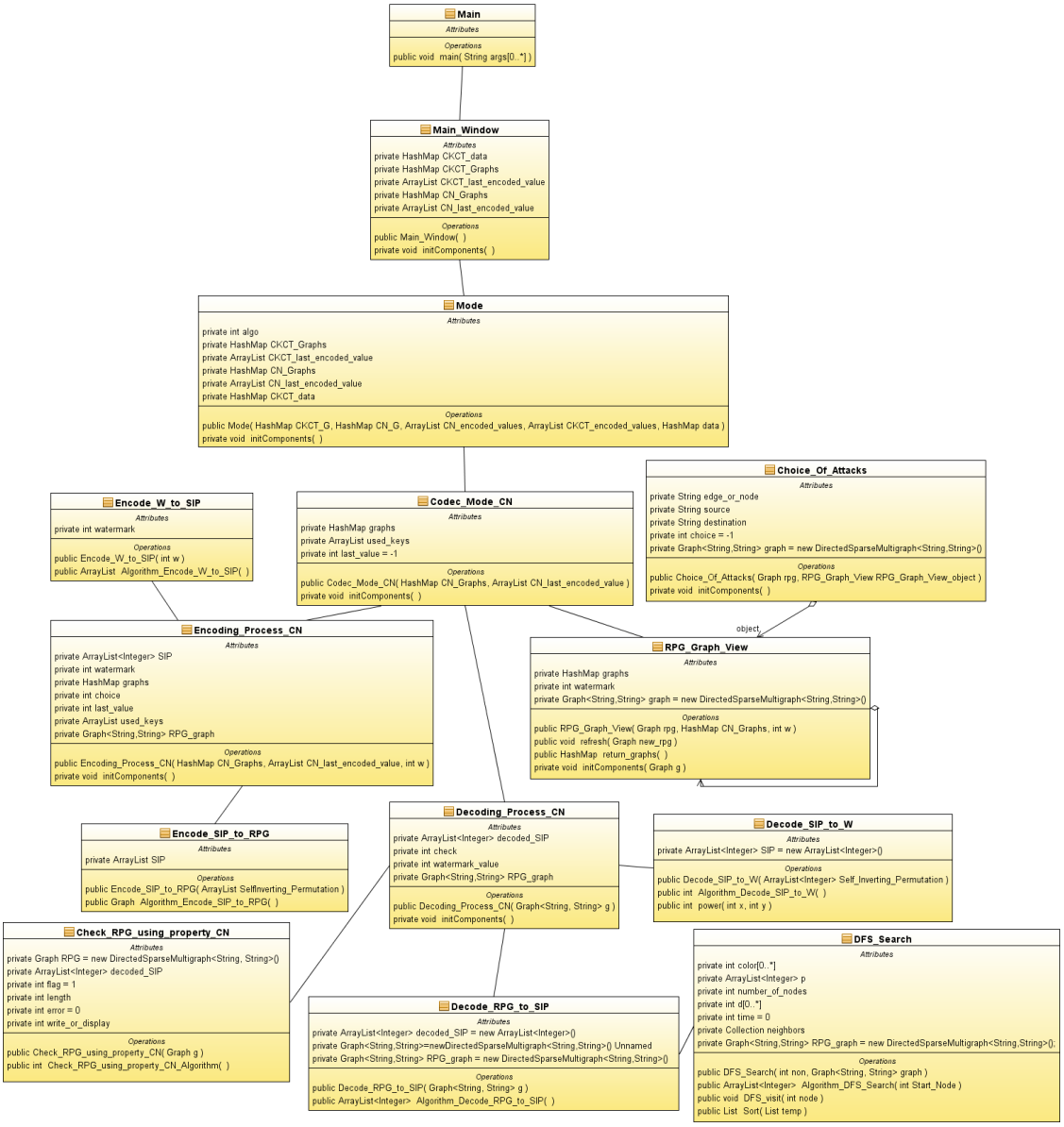


(β) Choice_Of_Attacks

Σχήμα 4.16: Κλάσεις υλοποίησης επιθέσεων σε αναγώγιμο μεταθετικό γράφημα

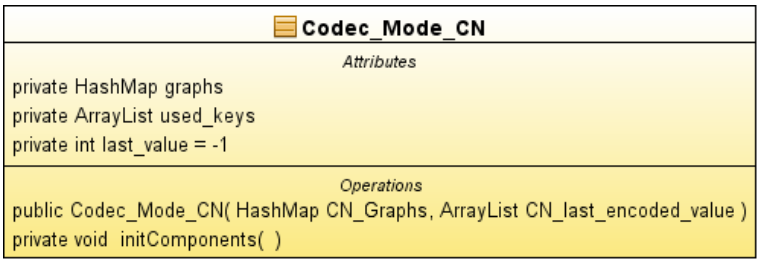
4.2.2 Δεύτερο Μέρος Σχεδίασης - Κωδικοποιητής Αλγορίθμου CN

Οι κλάσεις που έχουμε παρουσιάσει μέχρι και σε αυτό το σημείο ολοκληρώνουν το πρώτο μέρος του σχεδιασμού μας. Έτσι, μπορούμε στη συνέχεια να προχωρήσουμε στην παρουσίαση του δεύτερου μέρους της σχεδίασης, το οποίο φαίνεται στο Σχήμα 4.17.



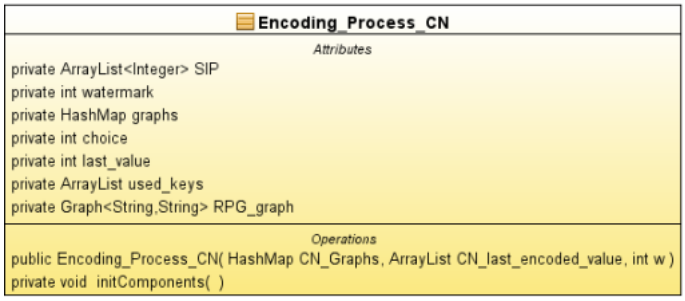
Σχήμα 4.17: Δεύτερο μέρος της σχεδίασης

Το δεύτερο μέρος της σχεδίασης αναφέρεται στις λειτουργίες κωδικοποίησης και αποκωδικοποίησης σύμφωνα με τον Αλγόριθμο CN και εφαρμογής επιθέσεων σε γραφήματα που δημιουργούνται από αυτόν. Όπως και στο πρώτο μέρος της σχεδίασης, έτσι και εδώ ο χρήστης έχει την δυνατότητα να εφαρμόσει κάποια από τις προαναφερθείσες λειτουργίες μέσω της κλάσης Codec_Mode_CN η οποία παρουσιάζεται στο Σχήμα 4.18.

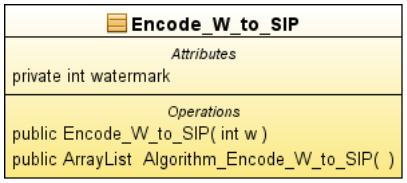


Σχήμα 4.18: Κλάση Codec_Mode_CN

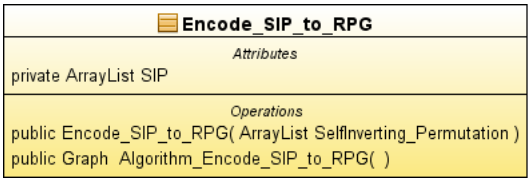
Στην περίπτωση που ο χρήστης επιλέξει να κωδικοποιήσει μια τιμή ενός υδατοσήμου καλείται η κλάση Encoding_Process_CN η οποία αποτελεί ένα στοιχείο γραφικής διεπαφής με τον χρήστη και η οποία αρχικά καλεί την κλάση Encode_W_to_SIP για να μετατρέψει την τιμή του υδατοσήμου σε αυτο-αντιστρέψιμη μετάθεση και στη συνέχεια την κλάση Encode_SIP_to_RPG με την οποία μετατρέπει αυτή την μετάθεση σε αναγώγιμο μεταθετικό γράφημα. Και οι τρεις αυτές κλάσεις φαίνονται στο σχήμα που ακολουθεί.



(α) Encoding_Process_CN



(β) Encode_W_to_SIP

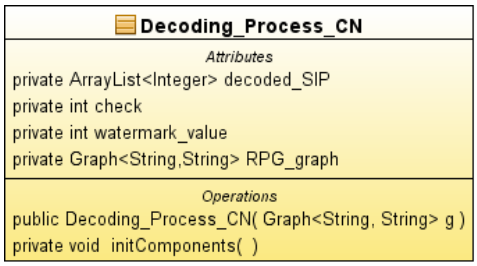


(γ) Encode_SIP_to_RPG

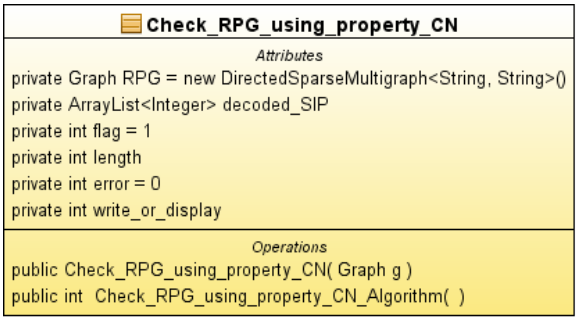
Σχήμα 4.19: Κλάσεις κωδικοποίησης Αλγορίθμου CN

Αν πάλι ο χρήστης επιλέξει να αποκωδικοποιήσει ένα αναγώγιμο μεταθετικό γράφημα καλείται η κλάση Decoding_Process_CN (βλέπε Σχήμα 4.20(α)) η οποία δίνει στον χρήστη την δυνατότητα να ελέγξει το γράφημα αν είναι αναγώγιμο, να αποκωδικοποιήσει το αναγώγιμο μεταθετικό γράφημα σε αυτο-αντιστρέψιμη μετάθεση και τέλος να αποκωδικοποιήσει την αυτο-αντιστρέψιμη μετάθεση στην τιμή του υδατοσήμου. Και οι τρεις αυτές λειτουργίες

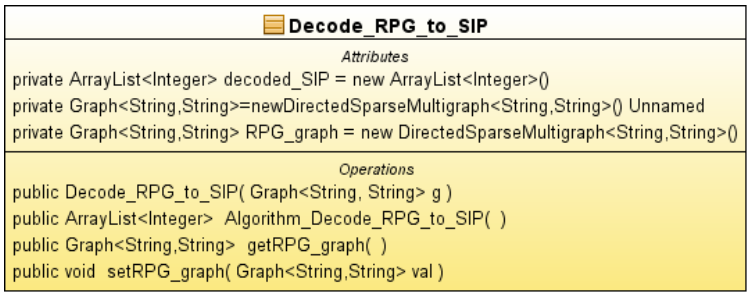
υλοποιούνται από τις κλάσεις Check_RPG_using_property_CN, Decode_RPG_to_SIP και Decode_SIP_to_W αντίστοιχα και οι οποίες φαίνονται στο παρακάτω σχήμα.



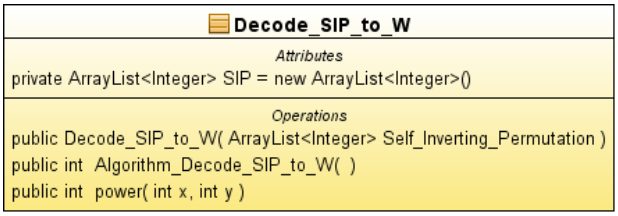
(α) Decoding_Process_CN



(β) Check_RPG_using_property_CN



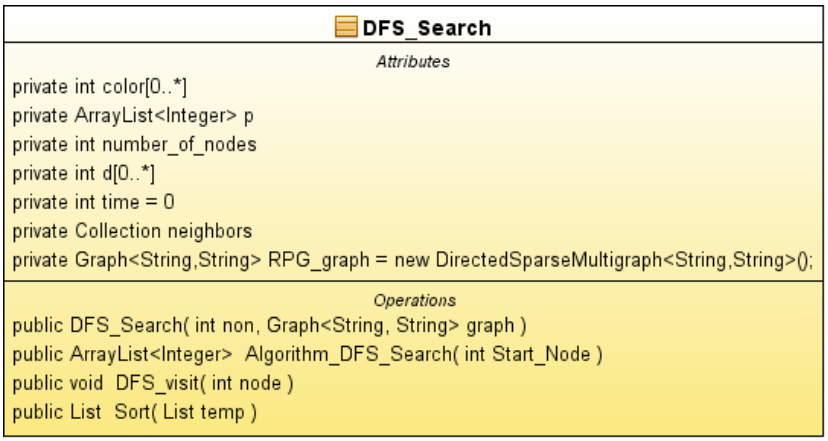
(γ) Decode_RPG_to_SIP



(δ) Decode_SIP_to_W

Σχήμα 4.20: Κλάσεις αποκωδικοποίησης Αλγορίθμου CN

Πρέπει να σημειωθεί ότι σύμφωνα με τον Αλγόριθμο 13 απαιτείται να γίνει αναζήτηση κατά βάθος. Η αναζήτηση κατά βάθος υλοποιείται από την κλάση DFS_Search και παρουσιάζεται στο Σχήμα 4.21.



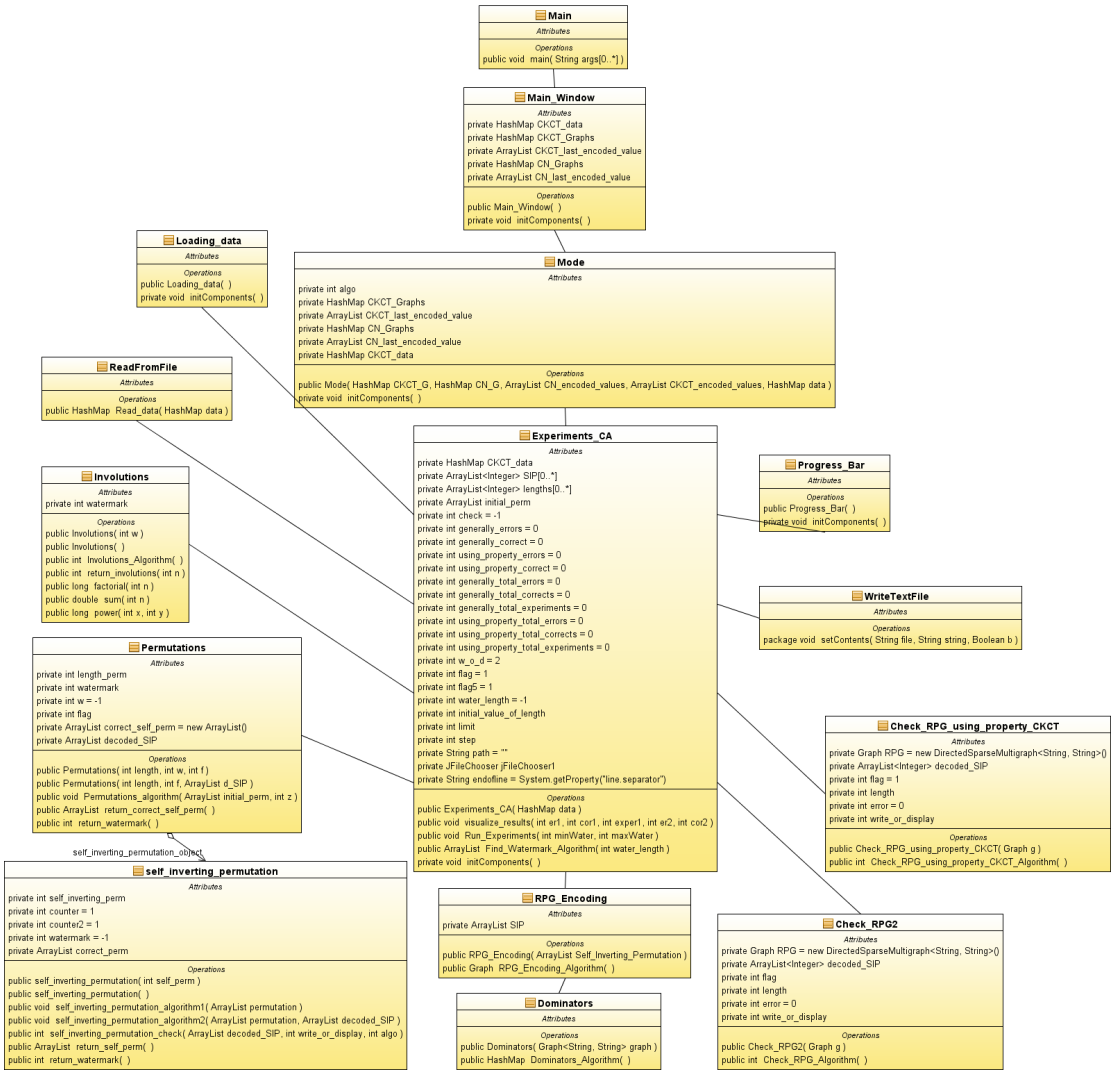
Σχήμα 4.21: Κλάση DFS_Search

Τέλος, αν ο χρήστης επιλέξει να εφαρμόσει κάποια επίθεση επάνω σε κάποιο κωδικοποιημένο γράφημα τότε καλούνται οι κλάσεις του Σχήματος 4.16 καθώς είναι κοινές και για τους δύο αλγορίθμους.

Έτσι, με αυτές τις κλάσεις ολοκληρώνεται και το δεύτερο μέρος της σχεδίασης του Συστήματος *WaterGraph*.

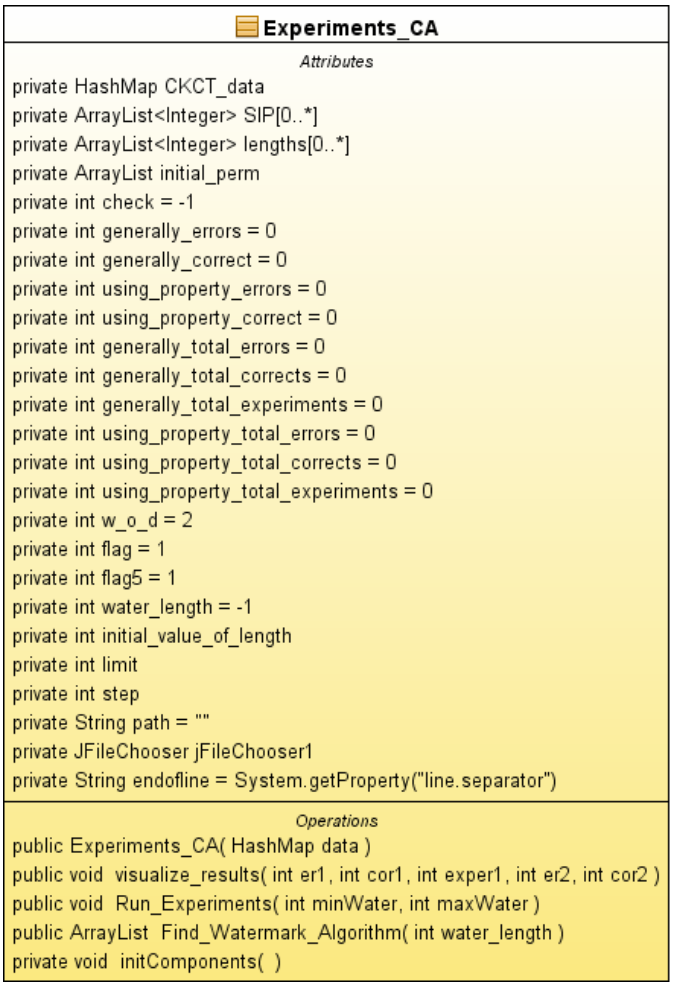
4.2.3 Τρίτο Μέρος Σχεδίασης - Πειραματική Λειτουργία Αλγορίθμου CKCT

Σε αυτή την υποενότητα θα παρουσιάσουμε το τρίτο μέρος της σχεδίασης το οποίο αποτελεί την πειραματική έρευνα για ανθεκτικότητα σε επιθέσεις στα γραφήματα που δημιουργούνται από τον Αλγόριθμο CKCT. Σχεδιαστικά το τρίτο μέρος περιγράφεται από το Σχήμα 4.22.



Σχήμα 4.22: Τρίτο μέρος της σχεδίασης

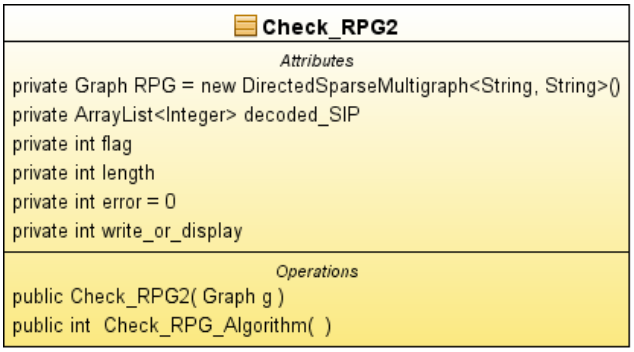
Μέσω της κλάσης Experiments_CA (βλέπε Σχήμα 4.23) δίνεται η δυνατότητα στο χρήστη να επιλέξει μια κατηγορία επίθεσης και να τρέξει ένα πλήθος πειραμάτων για να εξάγει κάποια ερευνητικά αποτελέσματα τα οποία θα δείχνουν πόσο ανθεκτικά είναι τα γραφήματα που δημιουργούνται από τον Αλγόριθμο CKCT.



Σχήμα 4.23: Κλάση Experiments_CA

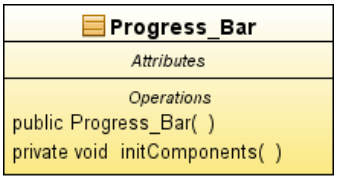
Όπως αναφέραμε και προηγουμένως ο Αλγόριθμος CKCT είναι αρκετά αργός στην παραγωγή των αυτο-αντιστρέψιμων ακολουθιών, γεγονός που οφείλεται στην υλοποίηση του. Γι' αυτό το λόγο γίνεται χρήση των κλάσεων Loading_data και ReadFromFile (βλέπε Σχήματα 4.6 και 4.7 αντίστοιχα) με τις οποίες φορτώνει στην μνήμη του υπολογιστή τις αυτο-αντιστρέψιμες ακολουθίες για τις τιμές υδατοσήμων από 1 έως 140152, οι οποίες είναι αποθηκευμένες σε ένα αρχείο. Υστερα από την ολοκλήρωση της διαδικασίας φόρτωσης των δεδομένων η κλάση Experiments_CA κωδικοποιεί τις τιμές των υδατοσήμων μια προς μια σε αυτο-αντιστρέψιμες μεταθέσεις μέσω των κλάσεων Involutions, Permutations και self_inverting_permutation (βλέπε Σχήματα 4.9 και 4.10 αντίστοιχα) και στη συνέχεια χρησιμοποιεί τις κλάσεις RPG_Encoding και Dominators (βλέπε Σχήμα 4.12) έτσι ώστε να παράξει τα αναγώγιμα μεταθετικά γραφήματα αυτών των μεταθέσεων. Όλες αυτές οι κλάσεις περιγράφηκαν εκτενέστερα προγενέστερα επομένως δεν θα τις αναλύσουμε περισσότερο. Τέλος, η κλάση Experiments_CA για κάθε γράφημα που δημιουργεί εφαρμόζει τυχαία σε αυτό την αντίστοιχη επίθεση που είχε επιλέξει εξ αρχής ο χρήστης και ελέγχει αν τα γραφήματα παραμένουν αναγώγιμα. Για τον έλεγχο των γραφημάτων χρησιμοποιεί τις κλάσεις Check_RPG2 (βλέπε Σχήμα 4.24) και Check_RPG_using_property_CKCT (βλέπε Σχήμα 4.14(α)), όπου η πρώτη χρησιμοποιεί το Θεώρημα Α της Παραγράφου 3.3.5 για να

ελέγξει αν το γράφημα είναι αναγώγιμο ενώ η δεύτερη χρησιμοποιεί και το θεώρημα Α αλλά και τις ιδιότητες των βαθμών των εξερχόμενων ακμών που ισχύουν για τους κόμβους των γραφημάτων που δημιουργήθηκαν από τον Αλγόριθμο CKCT.



Σχήμα 4.24: Κλάση Check_RPG2

Επειδή το πλήθος των πειραμάτων μπορεί να είναι αρκετά μεγάλο, και ως εκ τούτου να χρειάζεται αρκετός χρόνος για να εκτελεστούν όλα αυτά τα πειράματα, η κλάση Experiments_CA καλεί την κλάση Progress_Bar η οποία αντιστοιχεί σε μια μπάρα προόδου και η οποία εμφανίζεται στην οθόνη καθ' όλη τη διάρκεια εκτέλεσης των πειραμάτων. Η κλάση αυτή παρουσιάζεται σχηματικά παρακάτω.

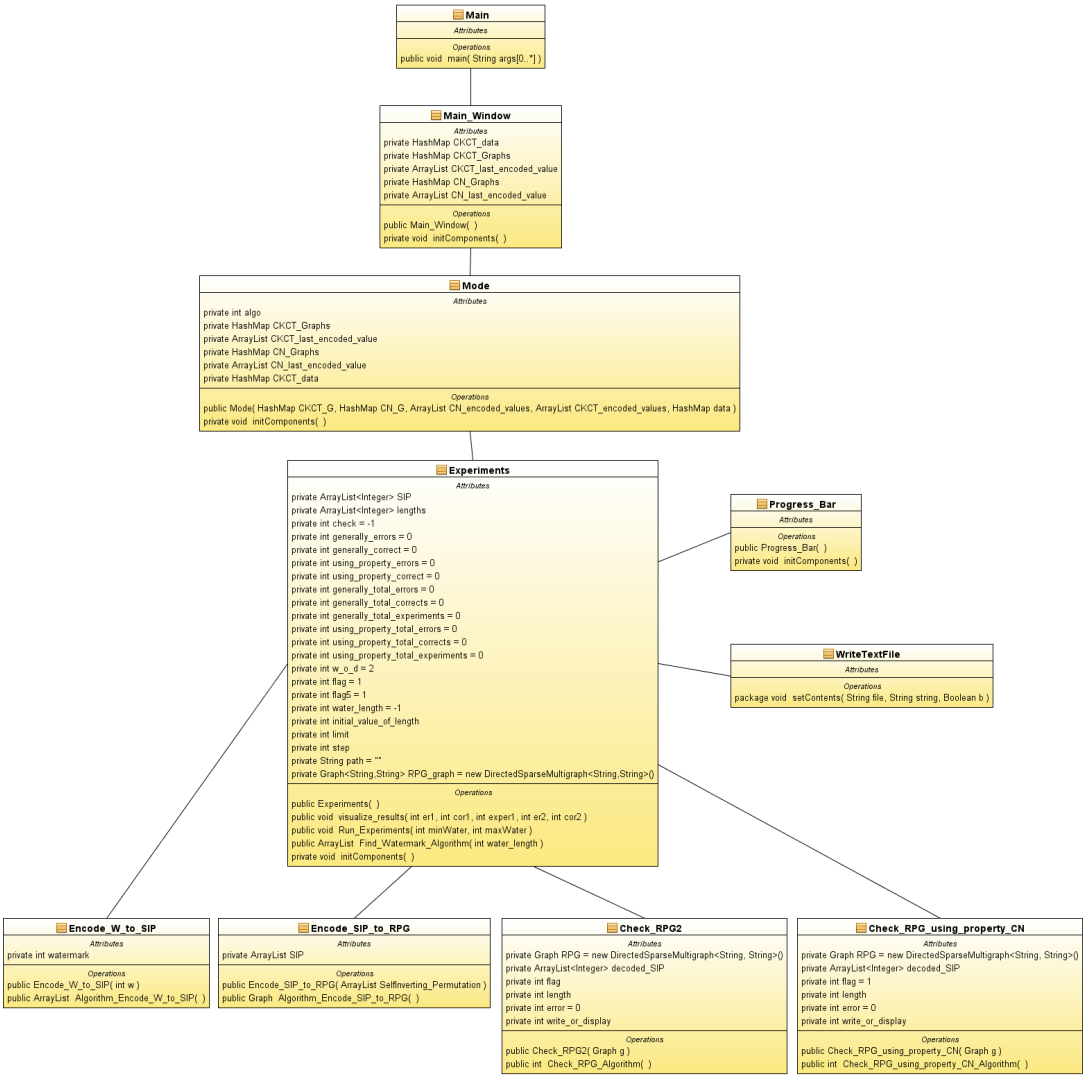


Σχήμα 4.25: Κλάση Progress_Bar

Τελός, αποθηκεύονται τα πειραματικά αποτελέσματα σε ένα αρχείο το οποίο επιλέγει ο χρήστης για μελλοντική τους χρήση. Η εγγραφή σε αρχείο πραγματοποιείται μέσω της κλάσης WriteTextFile την οποία την παρουσιάσαμε προηγουμένως στο Σχήμα 4.11.

4.2.4 Τέταρτο Μέρος Σχεδίασης - Πειραματική Λειτουργία Αλγορίθμου CN

Το τέταρτο και τελευταίο μέρος αποτελεί την πειραματική έρευνα για ανθεκτικότητα των δομών που δημιουργεί ο Αλγόριθμος CN. Ο σχεδιασμός αυτού του μέρους ομοιάζει πάρα πολύ με τον σχεδιασμό του τρίτου μέρους όπως θα μπορούσε κάποιος να παρατηρήσει στο Σχήμα 4.26.



Σχήμα 4.26: Τέταρτο μέρος της σχεδίασης

Η λειτουργία αυτού του μέρους είναι ίδια με αυτή του τρίτου μέρους, δηλαδή ως σκοπό έχει την κωδικοποίηση κάποιων τιμών που έχει επιλέξει ο χρήστης σε αναγώγιμα μεταθετικά γραφήματα και την εφαρμογή επιθέσεων σε αυτά έτσι ώστε να μπορέσουν να εξαχθούν κάποια αποτελέσματα για την ανθεκτικότητα αυτών των δομών που δημιουργούνται από τον Αλγόριθμο CN. Όπως και στο τρίτο μέρος, έτσι και εδώ υπάρχει μια κλάση Experiments, αντίστοιχη της Experiments_CA, η οποία παρουσιάζεται στο Σχήμα 4.27 και η οποία αποτελεί τον βασικό κορμό αυτού του μέρους.

Σχήμα 4.27: Κλάση Experiments

Η βασική διαφορά αυτού του μέρους από το τρίτο έγκειται στις κλάσεις που χρησιμοποιεί για την κωδικοποίηση των γραφημάτων και για τον έλεγχο τους. Συγκεκριμένα, για την κωδικοποίηση των τιμών των υδατοσήμων σε αυτο-αντιστρέψιμες μεταθέσεις χρησιμοποιείται η κλάση Encode_W_to_SIP την οποία είδαμε στο Σχήμα 4.19(β). Για την κωδικοποίηση αυτών των μεταθέσεων σε αναγώγιμα μεταθετικά γραφήματα καλείται η κλάση Encode_SIP_to_RPG που παρουσιάσαμε στο δεύτερο μέρος της σχεδίασης στο Σχήμα 4.19(γ). Αφού ολοκληρωθεί η διαδικασία της κωδικοποίησης και εφαρμοστεί μια επίθεση στο κάθε γράφημα που δημιουργείται κάθε φορά γίνεται έλεγχος για το αν τα γραφήματα παραμένουν αναγώγιμα. Ο έλεγχος αυτός, όπως και στο τρίτο μέρος του σχεδιασμού, γίνεται μέσω των κλάσεων Check_RPG2 (βλέπε Σχήμα 4.24) και Check_RPG_using_property_CN (βλέπε Σχήμα 4.20(β)), όπου η πρώτη όπως αναφέραμε ελέγχει το γράφημα μέσω του Θεωρήματος Α της Παραγράφου 3.3.5 και η δεύτερη κάνει χρήση του Θεωρήματος αλλά και τις ιδιότητας των βαθμών των εξερχόμενων ακμών των κόμβων για τον Αλγόριθμο `CN`. Αξίζει να σημειωθεί ότι όπως και στο τρίτο μέρος έτσι και εδώ, καθ' όλη τη διάρκεια εκτέλεσης των πειραμάτων καλείται η κλάση Progress_bar που αντιστοιχεί σε μια μπάρα προόδου και ότι τα αποτελέσματα των πειραμάτων γράφονται σε ένα αρχείο μέσω της κλάσης WriteToFile.

Έτσι, ολοκληρώσαμε την παρουσίαση και την ανάλυση και των τεσσάρων μερών της σχεδίασης του Συστήματος *WaterGraph*. Για να δείτε την υλοποίηση των κλάσεων ακολουθήστε τον σύνδεσμο $http://www.cs.uoi.gr/\sim stavros/mypage-students.html$.

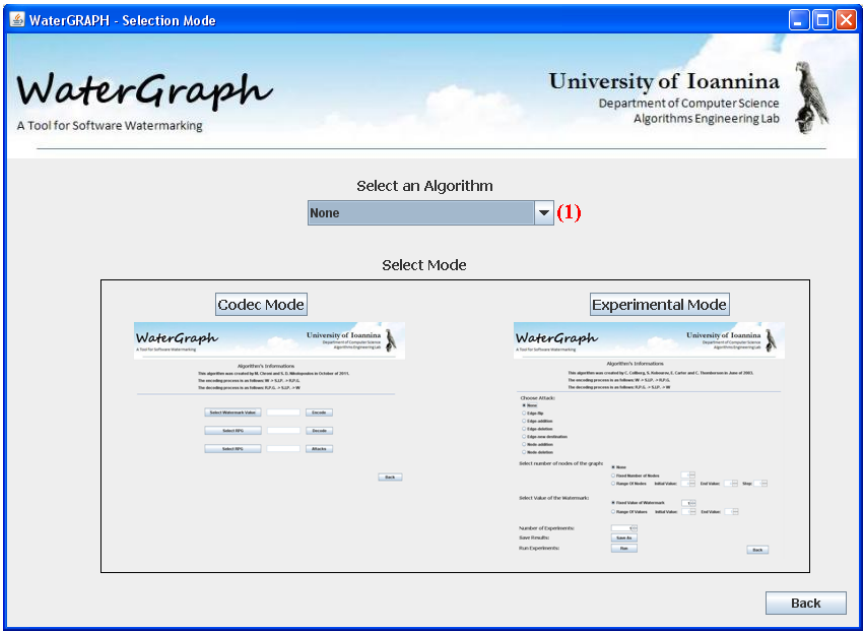
4.3 Παρουσίαση Συστήματος

Στην προηγούμενη παράγραφο παρουσιάσαμε τον σχεδιασμό του Συστήματος *WaterGraph* και αναλύσαμε την λειτουργία που επιτελεί η κάθε κλάση. Στην παρούσα παράγραφο θα παρουσιάσουμε τις φόρμες που δημιουργούνται από τις παραπάνω κλάσεις καθώς επίσης και την λειτουργία της κάθε φόρμας.

Κατα την εκκίνηση του συστήματος μας η πρώτη φόρμα που εμφανίζεται είναι αυτή του Σχήματος 4.28.

Σχήμα 4.28: Φόρμα εκκίνησης του συστήματος

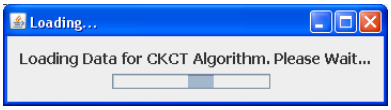
Στην παραπάνω φόρμα ο χρήστης μπορεί είτε να εισαχθεί στο πρόγραμμα επιλέγοντας το κουμπί Enter, είτε να το τερματίσει επιλέγοντας το κουμπί Exit. Στην περίπτωση που επιλεγεί το κουμπί Enter εμφανίζεται η φόρμα του Σχήματος 4.29.



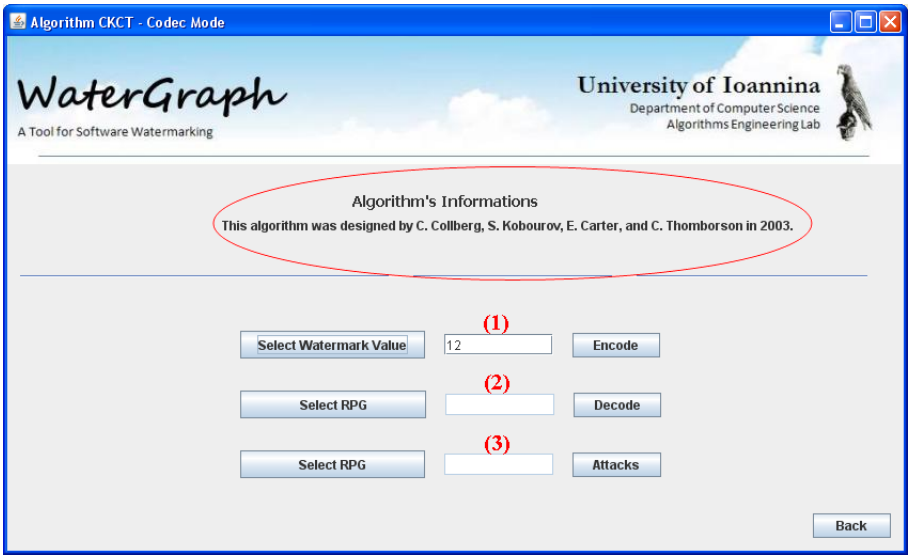
Σχήμα 4.29: Φόρμα επιλογής αλγορίθμου και εξέταση λειτουργίας και αποτελεσματικότητάς του

Αυτή η φόρμα περιέχει τη λίστα με τον αριθμό (1) στην οποία ο χρήστης πρέπει να επιλέξει τον αλγόριθμο με τον οποίο επιθυμεί να ασχοληθεί. Οι αλγόριθμοι που περιέχονται σε αυτή την λίστα είναι αυτοί που μελετήσαμε και στο Κεφάλαιο 3, δηλαδή ο `CN` και ο `CKCT`. Αν έχει επιλεγεί αλγόριθμος τότε ο χρήστης μπορεί είτε να επιλέξει το κουμπί Codec Mode, με το οποίο θα μεταβεί στην αντίστοιχη φορμα του επιλεγμένου αλγορίθμου και με την οποία θα μπορέσει να ασχοληθεί με τον *κωδικοποιητή* αυτού, είτε το κουμπί Experimental Mode, με το οποίο θα μεταβεί στην αντίστοιχη φόρμα του αλγορίθμου που έχει επιλέξει και με την οποία θα μπορέσει να εκτελέσει κάποια πειράματα. Στην περίπτωση που δεν έχει επιλέξει κάποιον αλγόριθμο τότε θα εμφανιστεί προειδοποιητικό μήνυμα το οποίο θα προτρέπει τον χρήστη να επιλέξει έναν. Τέλος, σε αυτή τη φόρμα υπάρχει και το κουμπί Back με το οποίο ο χρήστης κλείνει στην ουσία αυτή τη φόρμα και η μόνη φόρμα που είναι ενεργή είναι εκείνη της εκκίνησης του συστήματος.

Εαν ο χρήστης στη φόρμα του Σχήματος 4.29 επιλέξει τον Αλγόριθμο `CKCT` και το κουμπί Codec Mode τότε θα εμφανιστεί το παράθυρο του Σχήματος 4.30 και η φόρμα του Σχήματος 4.31.



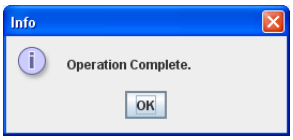
Σχήμα 4.30: Παράθυρο εμφάνισης προόδου δεδομένων



Σχήμα 4.31: Κωδικοποιητής Αλγορίθμου `CKCT`

Το παράθυρο του Σχήματος 4.30 περιέχει μια μπάρα προόδου η οποία προειδοποιεί τον χρήστη ότι φορτώνονται δεδομένα για τον Αλγόριθμο `CKCT`, δηλαδή φορτώνονται στην μνήμη του υπολογιστή οι αυτο-αντιστρέψιμες μεταθέσεις για τιμές υδατοσήμου από 1 έως 140152 οι οποίες είναι εγγεγραμμένες σε ένα αρχείο. Αυτό το παράθυρο θα εμφανιστεί επάνω στη φόρμα του Σχήματος 4.31, επικαλύπτοντάς την εν μέρει, και τα κουμπιά της

τελευταίας θα είναι ανενεργά καθ' όλη τη διάρκεια εκτέλεσης της τρίτης. Όταν ολοκληρωθεί η διαδικασία φόρτωσης, το παράθυρο του Σχήματος 4.30 εξαφανίζεται και εμφανίζεται το μήνυμα που φαίνεται στο Σχήμα 4.32.



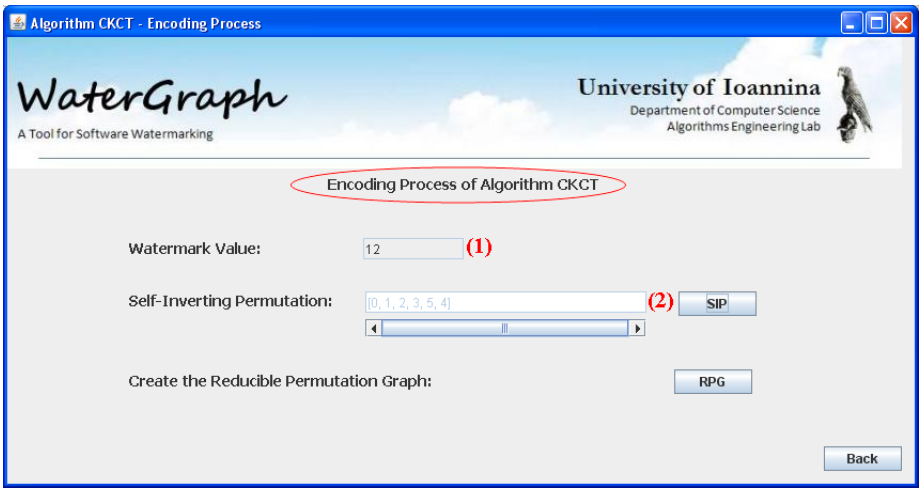
Σχήμα 4.32: Μήνυμα ολοκλήρωσης φόρτωσης δεδομένων Αλγορίθμου CKCT

Όταν ο χρήστης πατήσει το κουμπί OK σε αυτό το παράθυρο τότε αυτό εξαφανίζεται και ενεργοποιούνται τα κουμπιά της φόρμας του Σχήματος 4.31 που προηγουμένως ήταν ανενεργά.

Στην ίδια φόρμα υπάρχουν κάποιες πληροφορίες που ενημερώνουν τον χρήστη από ποιούς έχει δημιουργηθεί ο αλγόριθμος και πότε. Όλες οι παραπάνω πληροφορίες περιέχονται στο χώρο που περικλείεται μέσα στην κόκκινη έλλειψη. Μέσω αυτής της φόρμας ο χρήστης έχει τις εξής δυνατότητες:

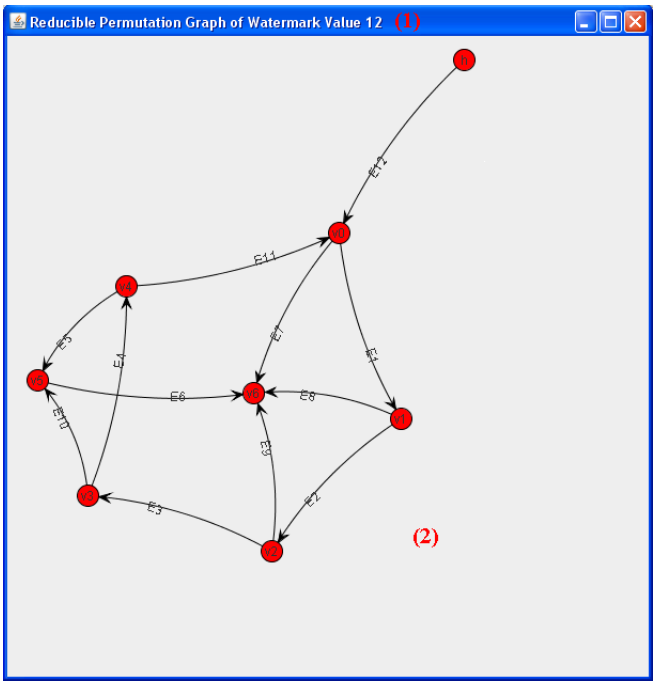
- κωδικοποίηση της τιμής ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα,
- αποκωδικοποίηση ενός αναγώγιμου μεταθετικού γραφήματος που έχει δημιουργηθεί από την διαδικασία κωδικοποίησης,
- εφαρμογή επιθέσεων επάνω σε ένα ήδη δημιουργημένο αναγώγιμο μεταθετικό γράφημα.

Για την κωδικοποίηση της τιμής ενός υδατοσήμου σε αναγώγιμο μεταθετικό γράφημα ο χρήστης πρέπει να επιλέξει το κουμπί Encode. Η τιμή του υδατοσήμου συμπληρώνεται από τον χρήστη στο πεδίο (1) της φόρμας πατώντας το κουμπί Select Watermark Value, ούτως ώστε να ενεργοποιηθεί το πεδίο (1) και στη συνέχεια συμπληρώνει την τιμή που επιθυμεί σε αυτό το πεδίο. Αν ο χρήστης είχε κωδικοποιήσει και άλλη τιμή υδατοσήμου νωρίτερα τότε στα ανενεργά πεδία (1), (2) και (3) θα εμφανιζόταν η τελευταία τιμή που έχει κωδικοποιήσει. Φυσικά αυτό δεν τον εμποδίζει να κωδικοποιήσει κάποια άλλη τιμή καθώς θα μπορούσε να αλλάξει την τιμή του υδατοσήμου πατώντας όπως προαναφέραμε το κουμπί Select Watermark Value και αλλάζοντας την τιμή του πεδίου. Εαν πάλι πατήσει το κουμπί Encode χωρίς να υπάρχει τιμή στο πεδίο (1) τότε θα εμφανιστεί προειδοποιητικό μήνυμα που θα προτρέπει τον χρήστη να συμπληρώσει την τιμή που θέλει να κωδικοποιήσει. Πρέπει να σημειώσουμε ότι οι τιμές που κωδικοποιούνται από τον χρήστη αποθηκεύονται στην μνήμη του υπολογιστή έτσι ώστε να μην χρειάζεται κάθε φορά που χρησιμοποιεί την ίδια τιμή να την κωδικοποιεί ξανά και ξανά. Για τη συνέχεια της παρουσίασης συμπληρώσαμε στο πεδίο (1) την τιμή 12 την οποία και θα κωδικοποιήσουμε. Αφού συμπληρωθεί το πεδίο (1) και πατηθεί το κουμπί Encode τότε εμφανίζεται η φόρμα του Σχήματος 4.33.



Σχήμα 4.33: Φόρμα κωδικοποίησης Αλγορίθμου CKCT

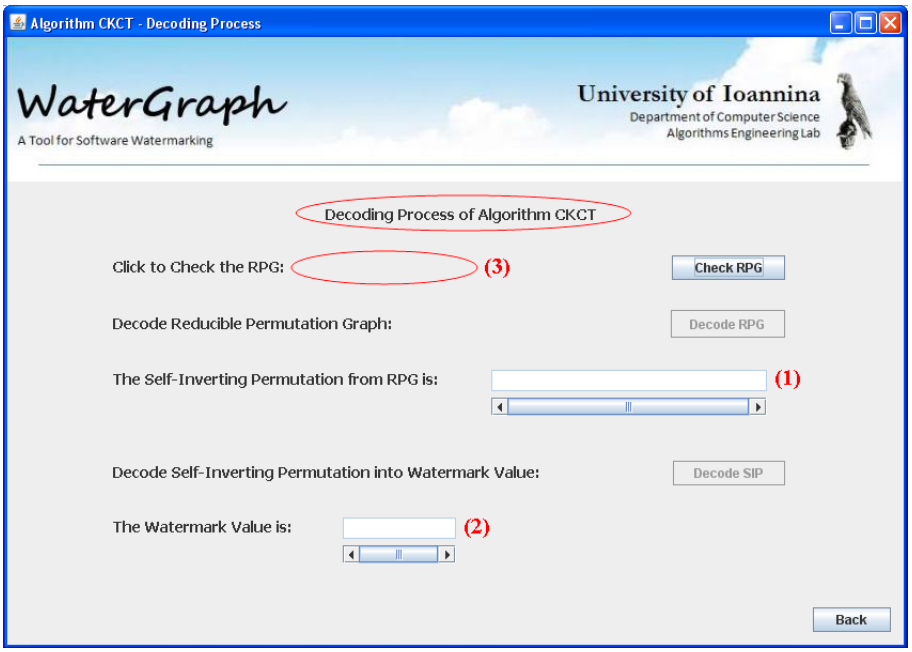
Η παραπάνω φόρμα αποτελεί την φόρμα κωδικοποίησης του Αλγορίθμου CKCT, όπως αναφέρεται και στην περιοχή που είναι μέσα στην κόκκινη έλλειψη. Στο πεδίο (1) αυτής της φόρμας τοποθετείται η τιμή του υδατοσήμου που δόθηκε στη φόρμα του Σχήματος 4.31 στο πεδίο (1) και είναι αυτή η τιμή που θέλουμε να κωδικοποιήσουμε. Σε αυτό το πεδίο εμφανίζεται η τιμή 12 που είχαμε συμπληρώσει στην προηγούμενη φόρμα. Για την κωδικοποίηση αυτής της τιμής σε αναγώγιμο μεταθετικό γράφημα ο χρήστης πρέπει καταρχάς να δημιουργήσει την αυτο-αντιστρέψιμη μετάθεση επιλέγοντας το κουμπί SIP. Στην περίπτωση που δεν έχει κωδικοποιηθεί ξανά η συγκεκριμένη τιμή υδατοσήμου η μετάθεση εμφανίζεται απευθείας στο ανενεργό πεδίο (2) αυτής της φόρμας. Σε αντίθετη περίπτωση, ο χρήστης ενημερώνεται με προειδοποιητικό μήνυμα ότι έχει κωδικοποιήσει και νωρίτερα αυτή την τιμή και προτρέπεται να απαντήσει με ένα Yes ή No για το αν επιθυμεί να επαναλάβει τη διαδικασία με την ίδια τιμή. Αν επιλεγεί το Yes τότε εμφανίζεται και πάλι η αυτο-αντιστρέψιμη μετάθεση αυτής της τιμής, ειδάλλως ο χρήστης προτρέπεται από προειδοποιητικό μήνυμα να επιλέξει καινούργια τιμή και το πεδίο (2) παραμένει κενό. Πρέπει να τονίσουμε ότι αυτό το πεδίο είναι κενό μέχρι να επιλεγεί το κουμπί SIP. Για την τιμή 12 του υδατοσήμου εμφανίζεται στο πεδίο (2), μετά την επιλογή του κουμπιού SIP, η αυτο-αντιστρέψιμη μετάθεση [0,1,2,3,5,4]. Για να ολοκληρώσει την κωδικοποίηση ο χρήστης αρκεί να επιλέξει το κουμπί RPG με το οποίο θα δημιουργήσει το αναγώγιμο μεταθετικό γράφημα της τιμής που έχει επιλέξει και το οποίο θα εμφανιστεί στο παράθυρο του ακόλουθου σχήματος.



Σχήμα 4.34: Παράθυρο εμφάνισης αναγώγιμου μεταθετικού γραφήματος

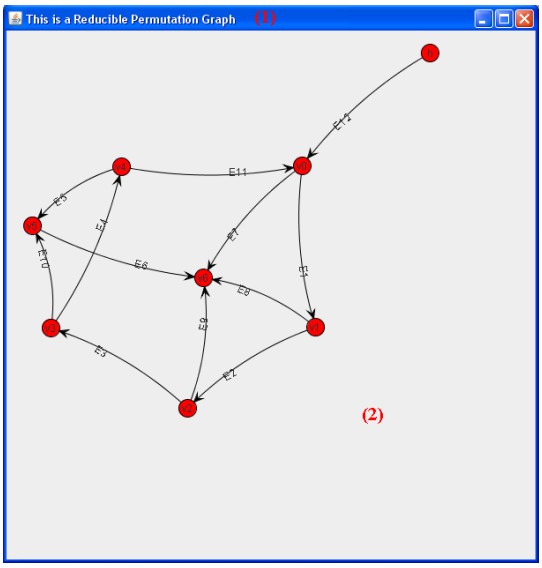
Σε αυτό το παράθυρο στην περιοχή (1) εγγράφεται η τιμή του υδατοσήμου που κωδικοποιήσαμε και στην περιοχή (2) εμφανίζεται το αναγώγιμο μεταθετικό γράφημα που αυτή αντιστοιχεί. Στο παράδειγμα μας, στην περιοχή (1) εμφανίζεται η τιμή 12 και στην περιοχή (2) το αναγώγιμο μεταθετικό γράφημα αυτής της τιμής. Στη φόρμα του Σχήματος 4.33 υπάρχει επίσης το κουμπί Back το οποίο τήν κλείνει.

Για την αποκωδικοποίηση της τιμής ενός αναγώγιμου μεταθετικού γραφήματος που έχει δημιουργηθεί από τον Αλγόριθμο CKCT αρκεί ο χρήστης να συμπληρώσει στο πεδίο (2) του Σχήματος 4.31 την τιμή του υδατοσήμου του οποίου θέλει να αποκωδικοποιήσει το γράφημα. Στο πεδίο (2) αυτής της φόρμας τοποθετείται πάντα η τελευταία τιμή που έχει κωδικοποιήσει ο χρήστης, όμως αυτή η τιμή μπορεί να τροποποιηθεί επιλέγοντας το μεσαίο κουμπί Select RPG έτσι ώστε να ενεργοποιήσει το ανενεργό πεδίο (2) και να εισάγει την τιμή που επιθυμεί. Αν η τιμή που θα συμπληρώσει δεν αντιστοιχεί σε κάποιο κωδικοποιημένο αναγώγιμο μεταθετικό γράφημα τότε άμεσα μετά την επιλογή του κουμπιού Decode ο χρήστης προτρέπεται μέσω προειδοποιητικού μηνύματος να κωδικοποιήσει πρώτα αυτή την τιμή. Στην περίπτωση όμως που υπάρχει η εν λόγω τιμή τότε εμφανίζεται η ακόλουθη φόρμα. Επειδή έχουμε κωδικοποιήσει ήδη την τιμή 12 θα εμφανιστεί στο πεδίο (2) αυτή η τιμή και δεν θα εμφανιστεί κανένα προειδοποιητικό μήνυμα με την επιλογή του κουμπιού Decode.



Σχήμα 4.35: Φόρμα αποκωδικοποίησης Αλγορίθμου CKCT

Σε αυτή τη φόρμα ο χρήστης πληροφορείται ότι εκτελεί τη διαδικασία αποκωδικοποίησης του Αλγορίθμου CKCT μέσω του μηνύματος που βρίσκεται μέσα στην κόκκινη έλλειψη στην κορυφή της φόρμας. Πριν ξεκινήσει τη διαδικασία αποκωδικοποίησης του γραφήματος που επέλεξε, ο χρήστης πρέπει να ελέγξει αν το γράφημα είναι αναγώγιμο και αν ισχύουν οι ιδιότητες για τους βαθμούς των εξερχόμενων ακμών. Αυτός ο έλεγχος πραγματοποιείται μέσω του κουμπιού Check RPG. Με την επιλογή αυτού του κουμπιού, στη φόρμα στην περιοχή (3) εγγράφεται είτε το μήνυμα "The graph is RPG", αν έχει περάσει τον έλεγχο, είτε το μήνυμα "The graph is not RPG" σε αντίθετη περίπτωση. Επίσης, εμφανίζεται το παράθυρο του Σχήματος 4.36. Στο παράδειγμα μας το γράφημα της τιμής 12 που έχουμε χρησιμοποιήσει δεν έχει δεχτεί κάποια επίθεση άρα είναι αναγώγιμο. Επομένως, στην περιοχή (3) θα εμφανιστεί το πρώτο μήνυμα που αναφέραμε.

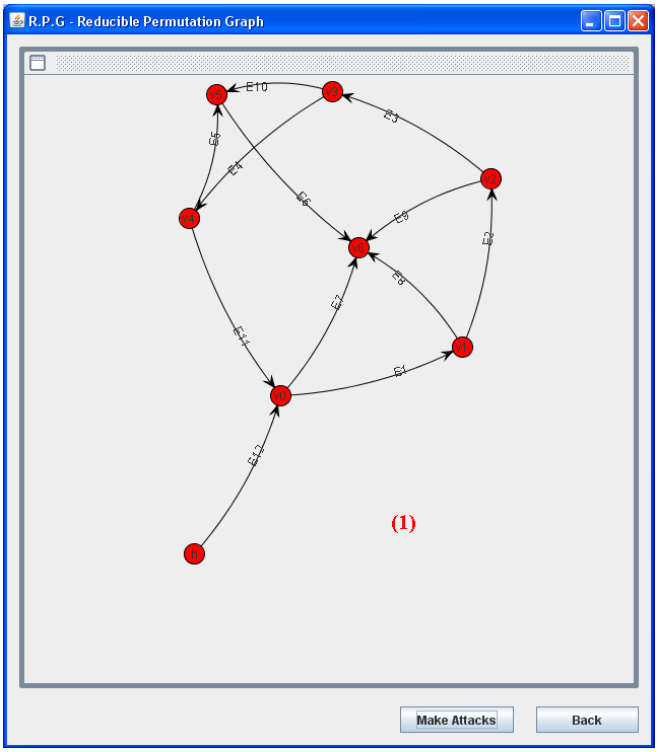


Σχήμα 4.36: Παράθυρο εμφάνισης γραφήματος αναγώγιμου ή μη

Σε αυτό το παράθυρο στην περιοχή (1) αναγράφεται πάλι αντίστοιχο μήνυμα αν το γράφημα έχει περάσει ή όχι τον παραπάνω έλεγχο και στην περιοχή (2) εμφανίζεται το γράφημα αυτό. Για το παράδειγμα μας θα εμφανιστεί στην περιοχή (1) το μήνυμα "The graph is RPG" και στην περιοχή (2) το αναγώγιμο μεταθετικό γράφημα που αντιστοιχεί στην τιμή 12. Μετά το πέρας του ελέγχου αυτού, στην περίπτωση που το γράφημα τον έχει περάσει με απόλυτη επιτυχία, ενεργοποιείται στο Σχήμα 4.35 το κουμπί Decode RPG το οποίο ήταν ανενεργό. Πλέον ο χρήστης μπορεί να ξεκινήσει τη διαδικασία της αποκωδικοποίησης του γραφήματος στην αυτο-αντιστρέψιμη μετάθεση που αυτό αντιστοιχεί. Αυτό πραγματοποιείται αν ο χρήστης επιλέξει το κουμπί Decode RPG μέσω του οποίου θα εγγραφεί στο πεδίο (1) η αυτο-αντιστρέψιμη μετάθεση. Για την τιμή 12 που έχουμε επιλέξει θα εμφανιστεί στο πεδίο (1) η μετάθεση [0,1,2,3,5,4]. Με την ολοκλήρωση του υπολογισμού της αυτο-αντιστρέψιμης μετάθεσης ενεργοποιείται το κουμπί Decode SIP το οποίο ήταν ανενεργό μέχρι αυτό το σημείο. Μέσω αυτού του κουμπιού ο χρήστης αποκωδικοποιεί την αυτο-αντιστρέψιμη μετάθεση στην αντίστοιχη τιμή υδατοσήμου. Έτσι, στο πεδίο (2) εγγράφεται η τιμή του υδατοσήμου και στην ουσία η διαδικασία της αποκωδικοποίησης έχει ολοκληρωθεί. Για την μετάθεση [0,1,2,3,5,4] που παράχθηκε από την προηγούμενη διαδικασία θα εγγραφεί στο πεδίο (2) η τιμή 12. Για να κλείσει ο χρήστης την φόρμα αυτή υπάρχει το κουμπί Back με το οποίο επιτελείται αυτή η λειτουργία.

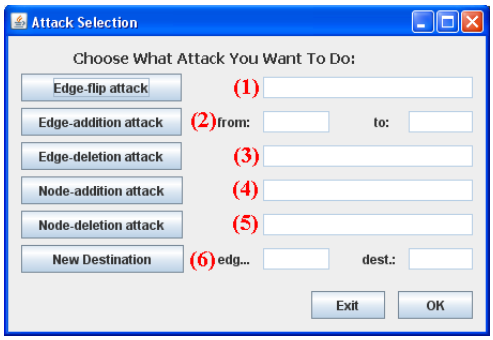
Τέλος, για να εφαρμόσει ο χρήστης επιθέσεις πάνω στα γραφήματα που έχει ήδη δημιουργήσει αρκεί στη φόρμα του Σχήματος 4.31 να επιλέξει το κουμπί Attacks μέσω του οποίου θα μεταβεί στη φόρμα του Σχήματος 4.37. Όπως και προηγουμένως, έτσι και εδώ στο πεδίο (3) του Σχήματος 4.31 υπάρχει η τελευταία τιμή που κωδικοποίησε ο χρήστης, στην περίπτωση μας θα περιέχει την τιμή 12 καθώς αυτή κωδικοποιήθηκε τελευταία, η οποία όμως μπορεί να αλλάξει μέσω του κουμπιού Select RPG. Σε περίπτωση που δεν υπάρχει τιμή σε αυτό το πεδίο και ο χρήστης επιλέξει το κουμπί Attacks εμφανίζεται προειδοποιητικό μήνυμα που αναφέρει στον χρήστη ότι δεν υπάρχει το συγκεκριμένο γράφημα στο οποίο

επιθυμεί να εφαρμόσει επιθέσεις.



Σχήμα 4.37: Φόρμα εφαρμογής επιθέσεων

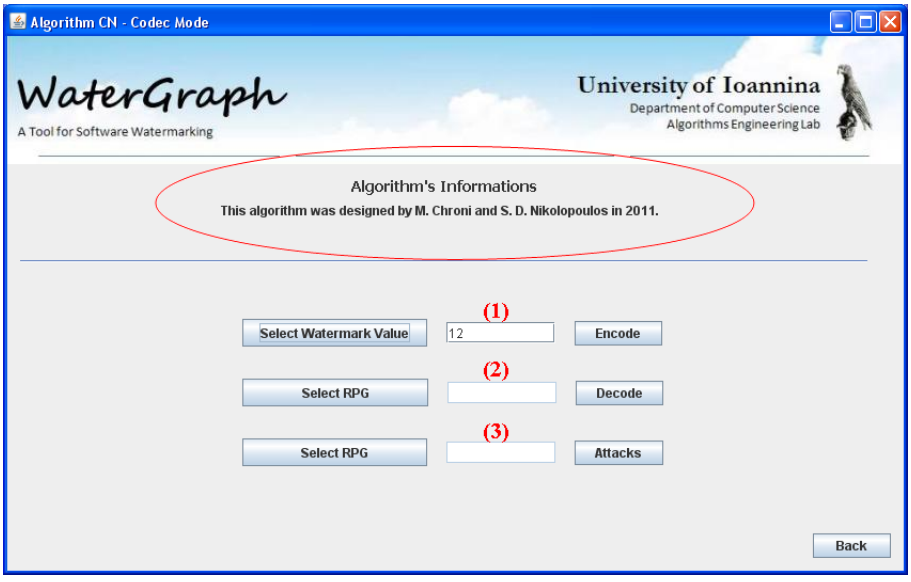
Στην περιοχή (1) της παραπάνω φόρμας εμφανίζεται το αναγώγιμο μεταθετικό γράφημα, το οποίο είχε επιλέξει ο χρήστης από την φόρμα που την κάλεσε. Επειδή στο πεδίο (3) της τελευταίας φόρμας υπήρχε η τιμή 12 όταν επιλέχθηκε το κουμπί Attacks, στην περιοχή (1) του Σχήματος 4.37 θα εμφανιστεί το αναγώγιμο μεταθετικό γράφημα της τιμής αυτής. Σε αυτό το σημείο δίδονται στον χρήστη δύο επιλογές, είτε να εφαρμόσει κάποια επίθεση στο γράφημα που εμφανίζεται στην περιοχή (1) επιλέγοντας το κουμπί Make Attacks, είτε να κλείσει αυτή τη φόρμα μέσω του κουμπιού Back. Σε περίπτωση που ο χρήστης επιλέξει το κουμπί Make Attacks εμφανίζεται η φόρμα του Σχήματος 4.38. Σε αυτή τη φόρμα ο χρήστης έχει την δυνατότητα να εφαρμόσει μια εκ των έξι επιθέσεων, οι οποίες είναι οι ακόλουθες: αντιστροφή ακμής, προσθήκη ακμής, διαγραφή ακμής, προσθήκη κόμβου, διαγραφή κόμβου και αλλαγή προορισμού ακμής.



Σχήμα 4.38: Φορμα επιλογής επίθεσης

Για να επιλέξει ο χρήστης να κάνει μια απο τις παραπάνω επιθέσεις πρέπει πρώτα να πατήσει το αντίστοιχο κουμπί, έτσι ώστε να ενεργοποιηθεί το αντίστοιχο πεδίο (ένα εκ των (1), (2), (3), (4), (5), (6)) και στη συνέχεια να εγγράψει την ακμή ή τον κόμβο που επιθυμεί σε αυτό. Πρέπει να τονιστεί ότι για να είναι διακριτές οι ακμές απο τον χρήστη, οι τελευταίες έχουν μοναδικό όνομα το οποίο παρουσιάζεται πάνω στο γράφημα. Για να ολοκληρώσει την επίθεση αρκεί ο χρήστης να πατήσει το κουμπί OK, ειδάλλως μπορεί να μην εφαρμόσει κάποια επίθεση πατώντας το κουμπί Exit με το οποίο κλείνει αυτή η φόρμα.

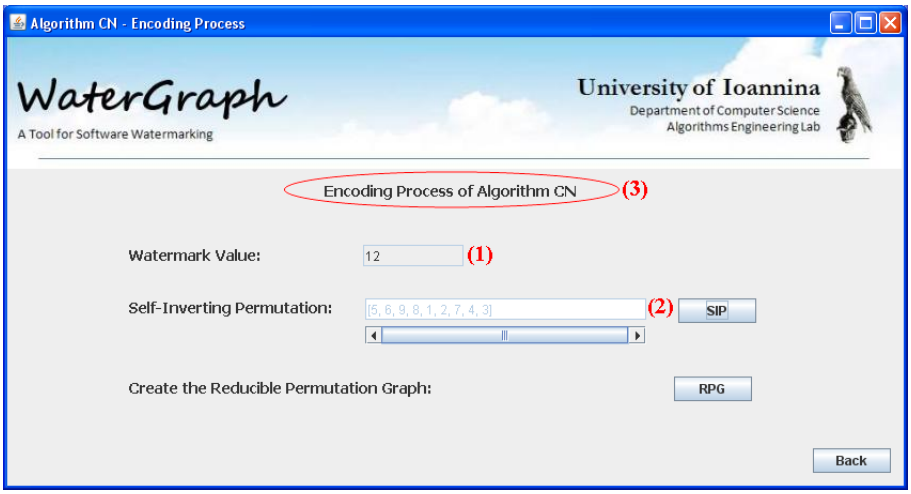
Μέχρι στιγμής έχουμε παρουσιάσει όλες τις φόρμες που θα παρουσιάζονταν αν ο χρήστης επέλεγε τον Αλγόριθμο CKCT στη φόρμα που παρουσιάζεται μετά την εκκίνηση του συστήματος και πατούσε το κουμπί Codec Mode. Αν όμως επέλεγε τον Αλγόριθμο CN και πατούσε το ίδιο κουμπί τότε η φόρμα που θα εμφανιζόταν θα ήταν η ακόλουθη.



Σχήμα 4.39: Κωδικοποιητής Αλγορίθμου CN

Παρατηρείται ότι η φόρμα αυτή ομοιάζει σε μεγάλο βαθμό με εκείνη που είδαμε στο Σχήμα 4.31 για τον Αλγόριθμο CKCT. Δύο βασικές διαφορές αυτών των φορμών είναι οι πληροφορίες που περιέχουν (βλέπε την κόκκινη έλλειψη στο Σχήμα 4.39) καθώς επίσης και ότι για τον Αλγόριθμο CN δεν φορτώνονται πληροφορίες σε αντίθεση με τον CKCT.

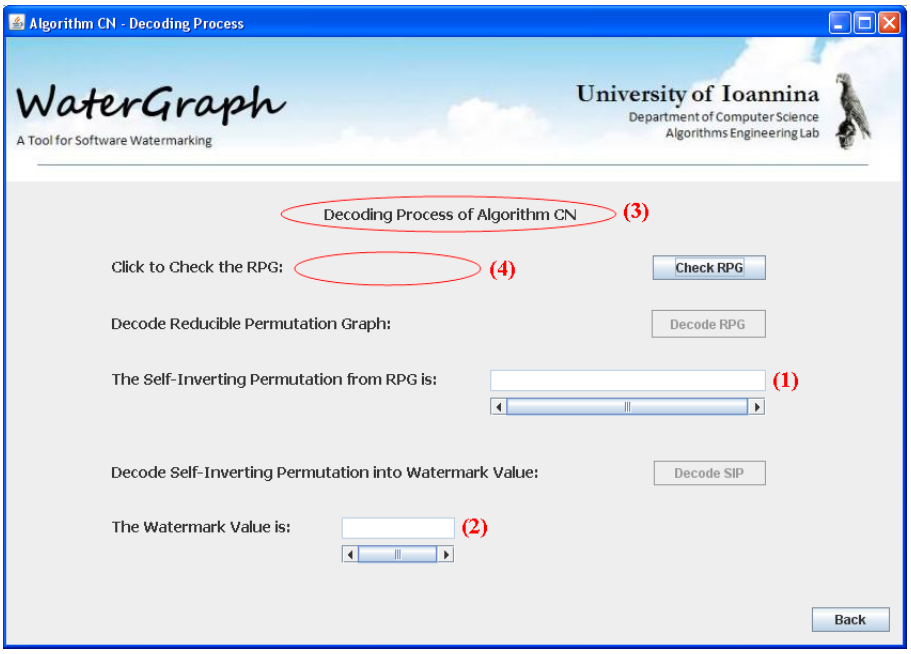
Οι λειτουργίες που προσφέρει η παραπάνω φόρμα είναι ακριβώς οι ίδιες με αυτές που πρόσφερε και εκείνη του Σχήματος 4.31 και για τις οποίες έγινε λόγος προγενέστερα. Αν ο χρήστης θελήσει να κωδικοποιήσει την τιμή ενός υδατοσήμου τότε εμφανίζεται η φόρμα του Σχήματος 4.40.



Σχήμα 4.40: Φόρμα κωδικοποίησης Αλγορίθμου CN

Μέσω αυτής της φόρμας ο χρήστης μπορεί να μετατρέψει την τιμή του υδατοσήμου σε αυτο-αντιστρέψιμη μετάθεση και στη συνέχεια αυτή σε αναγώγιμο μεταθετικό γράφημα όπως ακριβώς έκανε και στη φόρμα του Σχήματος 4.33. Μεταξύ αυτών των δύο φορμών η μόνη οπτική διαφορά είναι η πληροφορία που δίνει στον χρήστη και η οποία βρίσκεται μέσα στην κόκκινη έλλειψη. Θα πρέπει να τονιστεί ότι σε αυτές τις δύο φόρμες, παρα το ότι είναι ίδιες, διαφέρουν στον τρόπο με τον οποίο υπολογίζουν τις αυτο-αντιστρέψιμες μεταθέσεις και τα αναγώγιμα μεταθετικά γραφήματα. Γι' αυτό το λόγο προκύπτει και διαφορετική αυτο-αντιστρέψιμη μετάθεση γι' αυτόν τον αλγόριθμο σε σχέση με τον CKCT παρ' όλο που χρησιμοποιήσαμε την ίδια τιμή υδατοσήμου. Αφού ο χρήστης ολοκληρώσει την κωδικοποίηση της τιμής ενός υδατοσήμου σε γράφημα τότε αυτό εμφανίζεται μέσα στο παράθυρο του Σχήματος 4.34 όπως δηλαδή και στον Αλγόριθμο CKCT. Πρέπει να τονίσουμε ότι τα γραφήματα που προκύπτουν από τον Αλγόριθμο CN είναι εντελώς διαφορετικά από αυτά του CKCT γι' αυτό το λόγο το γράφημα του Σχήματος 4.34 δεν αντιστοιχεί σε αυτό που θα παρήγαγε ο CN.

Επίσης, για να αποκωδικοποιήσει ο χρήστης ένα ήδη κωδικοποιημένο γράφημα επιλέγει την τιμή του υδατοσήμου στο πεδίο (2) και το κουμπί Decode στο Σχήμα 4.39. Η φόρμα που εμφανίζεται είναι η εξής:



Σχήμα 4.41: Φόρμα αποκωδικοποίησης Αλγορίθμου CN

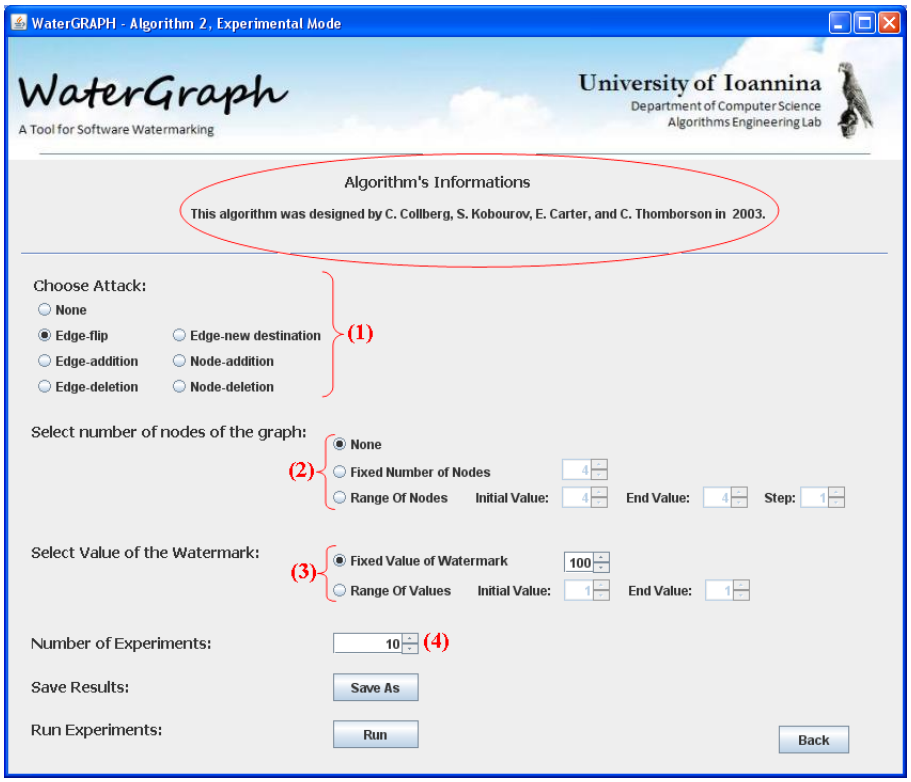
Όπως και στη φόρμα αποκωδικοποίησης του Αλγορίθμου CKCT έτσι και στην παραπάνω ο χρήστης πρέπει να ελέγξει το γράφημα που θέλει να αποκωδικοποιήσει αν είναι αναγώγιμο και αν ισχύει η ιδιότητα για το πλήθος των εξερχόμενων ακμών από τους κόμβους. Στο πεδίο (4) θα εμφανιστούν αντίστοιχα μηνύματα όπως και στη φόρμα του Σχήματος 4.35 και επίσης θα εμφανιστεί και εδώ το παράθυρο του Σχήματος 4.36 το οποίο θα επιτελεί τις ίδιες λειτουργίες που περιγράφηκαν και πρωτύτερα. Όμοια με την αντίστοιχη φόρμα του Αλγορίθμου CKCT έτσι και εδώ το κουμπί Decode RPG ενεργοποιείται εφόσον ολοκληρωθεί ο έλεγχος του γραφήματος με επιτυχία. Αν ο χρήστης επιλέξει το κουμπί Decode RPG εγγράφεται στο πεδίο (1) η αυτο-αντιστρέψιμη μετάθεση και στη συνέχεια ενεργοποιείται το κουμπί Decode SIP μέσω του οποίου εγγράφεται η τιμή του υδατοσήμου στο πεδίο (2). Για να κλείσει ο χρήστης αυτή τη φόρμα αρκεί να επιλέξει το κουμπί Back. Οι δύο φόρμες διαφέρουν στον τρόπο με τον οποίο υλοποιούν τη διαδικασία της αποκωδικοποίησης.

Τέλος, αν ο χρήστης θελήσει να εφαρμόσει κάποιες επιθέσεις επάνω στα γραφήματα που έχει δημιουργήσει με τον Αλγόριθμο CN αρκεί μέσω της φόρμας του Σχήματος 4.39 να συμπληρώσει το πεδίο (3) και στη συνέχεια να επιλέξει το κουμπί Attacks. Μετα την επιλογή αυτή εμφανίζεται η φόρμα του Σχήματος 4.37 και ο χρήστης επιτελεί ακριβώς τις ίδιες λειτουργίες με εκείνες που έκανε και προηγουμένως για τον Αλγόριθμο CKCT. Πρέπει να τονίσουμε ότι στην περιοχή (1) αυτής της φόρμας θα εμφανιστεί το γράφημα που κωδικοποιείται με βάση τον Αλγόριθμο CN και όχι αυτό που παρουσιάζεται.

Μέχρι τώρα έχουμε παρουσιάσει το γραφικό περιβάλλον για τους κωδικοποιητές των Αλγορίθμων CKCT και CN. Θα ολοκληρώσουμε αυτό το κεφάλαιο παρουσιάζοντας τις φόρμες που χρησιμοποιούνται απο τους αλγορίθμους για την πειραματική μελέτη τις ανθεκτικότητας των δομών που δημιουργούν.

Για να εξετάσει ο χρήστης πειραματικά την ανθεκτικότητα των δομών που δημιουργεί ο

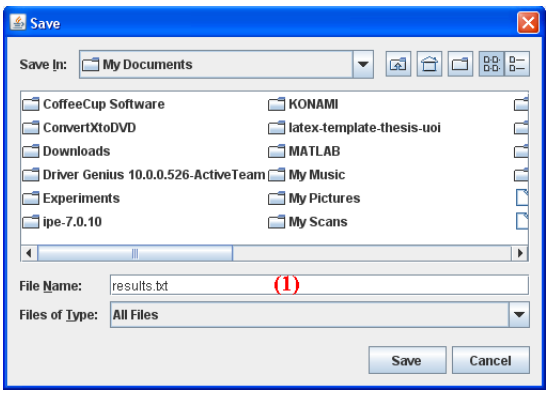
Αλγόριθμος CKCT αρκεί στη φόρμα που εμφανίζεται μετά την εισαγωγή στο σύστημα στο πεδίο (1) να επιλέξει τον συγκεκριμένο αλγόριθμο και να πατήσει το κουμπί Experimental Mode. Η φόρμα που εμφανίζεται είναι η εξής:



Σχήμα 4.42: Φόρμα πειραμάτων Αλγορίθμου CKCT

Αν χρησιμοποιείται για πρώτη φορά ο Αλγόριθμος CKCT τότε φορτώνονται από ένα αρχείο οι αυτο-αντιστρέψιμες μεταθέσεις για τις τιμές 1 έως και 140152 ακριβώς με τον ίδιο τρόπο που φορτώθηκαν και στη φόρμα του Σχήματος 4.31. Στην κορυφή της παραπάνω φόρμας παρουσιάζονται κάποιες γενικές πληροφορίες για τον αλγόριθμο που έχει επιλεγεί από τον χρήστη και οι οποίες βρίσκονται μέσα στην κόκκινη έλλειψη. Στην περιοχή (1) ο χρήστης επιλέγει την επίθεση που επιθυμεί να εφαρμόσει σε όλα του τα πειράματα, έχοντας φυσικά σαν προεπιλογή το none. Έχοντας επιλέξει την επίθεση που επιθυμεί, ο χρήστης πρέπει να επιλέξει τα γραφήματα που θα δημιουργεί είτε με βάση το μέγεθος τους, είτε με βάση την τιμή στην οποία αντιστοιχούν. Αυτές οι δύο επιλογές αντιστοιχούν στις περιοχές (2) και (3). Όπως παρατηρείται στην περιοχή (2) το none είναι προεπιλεγμένο, δηλαδή δεν δημιουργούνται γραφήματα με βάση το μέγεθος τους, αλλά με βάση την τιμή τους, γι' αυτό και είναι ενεργοποιημένη η περιοχή (3) και όχι η (2). Αν ο χρήστης επιθυμεί την δημιουργία γραφημάτων με βάση το μέγεθος τους, αρκεί να επιλέξει μια διαφορετική επιλογή από την περιοχή (2), οπότε θα απενεργοποιηθεί και η περιοχή (3). Οι επιλογές που δίδονται στην περιοχή (2) είναι είτε γραφήματα με συγκεκριμένο μέγεθος, είτε γραφήματα σε ένα συγκεκριμένο εύρος μεγέθους με συγκεκριμένο βήμα. Αντίθετα, στην περιοχή (3) δίδονται οι επιλογές είτε συγκεκριμένης τιμής υδατοσήμου, είτε εύρους τιμών. Έχοντας επιλέξει ο χρήστης μια από τις παραπάνω ενέργειες πρέπει να ορίσει τον αριθμό των πειραμάτων που

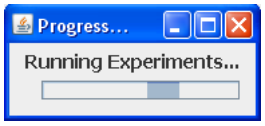
επιθυμεί να διεξάγει. Αυτό γίνεται συμπληρώνοντας τιμή στο πεδίο (4) της φόρμας. Για τη συνέχεια της παρουσίασης επιλέγουμε στην περιοχή (1) την επίθεση Edge-flip, στην περιοχή (3) επιλέγουμε να δημιουργήσουμε το γράφημα που αντιστοιχεί στην τιμή 100 και στην περιοχή (4) συμπληρώνουμε τον αριθμό 10, ώστε να εφαρμοστούν 10 πειράματα. Τέλος, πρέπει να οριστεί και το αρχείο στο οποίο επιθυμεί ο χρήστης να εγγράψει τα αποτελέσματα επιλέγοντας το κουμπί Save As. Η φόρμα που δημιουργείται είναι η ακόλουθη.



Σχήμα 4.43: Φόρμα αποθήκευσης αποτελεσμάτων

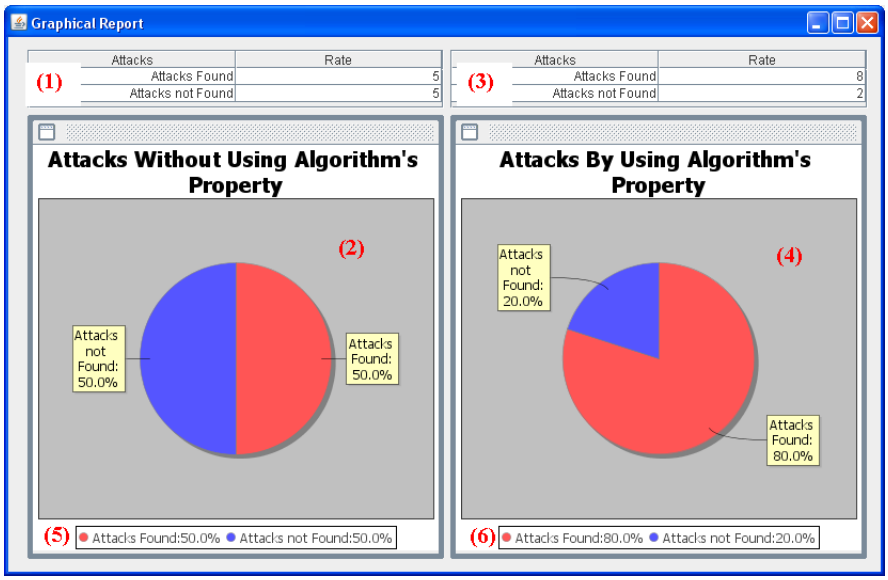
Σε αυτή τη φόρμα ο χρήστης επιλέγει το μονοπάτι στο οποίο θα αποθηκεύσει τα δεδομένα καθώς επίσης και το αρχείο που επιθυμεί να τα αποθηκεύσει. Το όνομα του αρχείου που επιθυμεί να τα αποθηκεύσει, το συμπληρώνει στο πεδίο (1) αν δεν υπάρχει, αλλιώς αν είναι ήδη δημιουργημένο απλά το επιλέγει. Αφού ολοκληρώσει αυτή την επιλογή πατάει το κουμπί Save και η φόρμα εξαφανίζεται. Στο παράδειγμα μας, επιθυμούμε να εγγράψουμε τα αποτελέσματα μας στο αρχείο results.txt.

Έχοντας ορίσει όλες τις παραμέτρους που είναι απαραίτητες για τη διεξαγωγή των πειραμάτων ο χρήστης επιλέγει το κουμπί Run για να εκτελέσει τα πειράματα του. Καθ' όλη τη διάρκεια διεξαγωγής των πειραμάτων απενεργοποιούνται όλα τα κουμπιά και εμφανίζεται η ακόλουθη μπάρα προόδου για να δηλώσει στον χρήστη ότι γίνεται κάποια ενέργεια.



Σχήμα 4.44: Μπάρα προόδου διεξαγωγής πειραμάτων

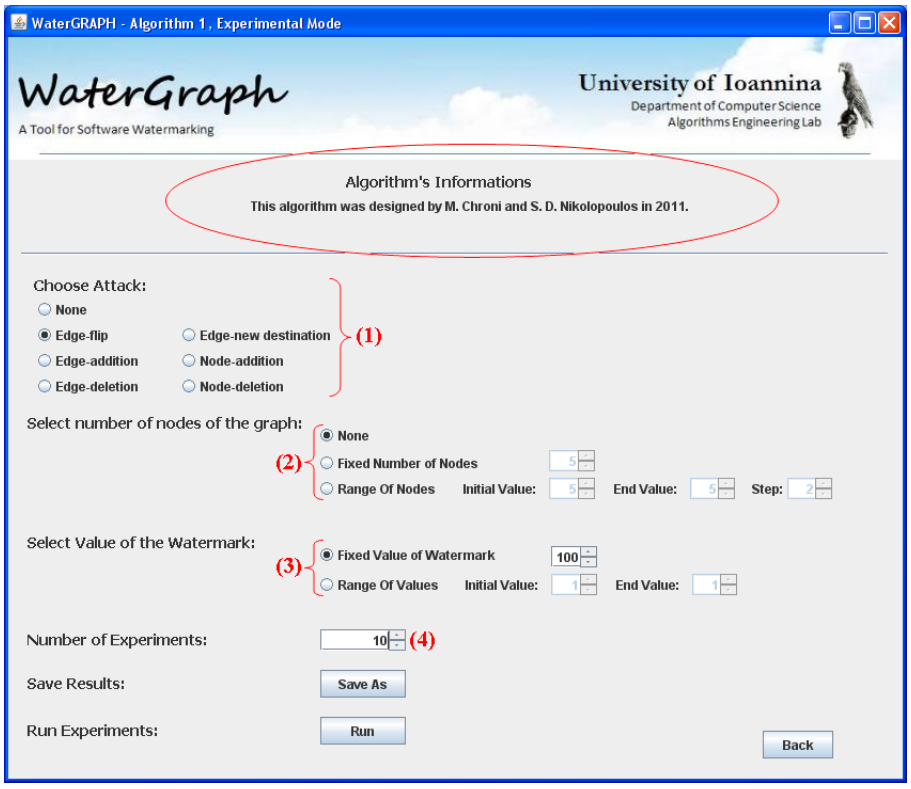
Η μπάρα προόδου εξαφανίζεται μετά το πέρας των πειραμάτων, ενεργοποιούνται τα κουμπιά της φόρμας του Σχήματος 4.42 που είχαν πριν απενεργοποιηθεί και εμφανίζεται το παράθυρο του Σχήματος 4.45 που μας αναφέρει τα αποτελέσματα και μας δείχνει μια γραφική απεικόνιση αυτών.



Σχήμα 4.45: Παράθυρο πειραματικών αποτελεσμάτων

Στην περιοχή (1) αυτού του παραθύρου εμφανίζεται ο αριθμός των επιθέσεων που εντοπίστηκαν από αυτές που έγιναν ελέγχοντας μόνο αν το γράφημα είναι αναγώγιμο, ενώ στην περιοχή (2) εμφανίζονται με μορφή διαγράμματος πίτας τα ίδια αποτελέσματα. Στην περιοχή (5) εμφανίζονται τα ποσοστά των επιθέσεων που βρέθηκαν και που δεν βρέθηκαν. Αντίθετα, στις περιοχές (3), (4) και (6) εμφανίζονται πάλι οι επιθέσεις που εντοπίστηκαν και που δεν εντοπίστηκαν ελέγχοντας όμως αυτή τη φορά εκτός του αν το γράφημα είναι αναγώγιμο και το πλήθος των εξερχόμενων ακμών από κάθε κόμβο. Για το παράδειγμα μας, στην περιοχή (1) εμφανίζεται ότι εντοπίστηκαν 5 επιθέσεις από τις 10, στην περιοχή (5) ότι εντοπίστηκε το 50% των επιθέσεων, στην περιοχή (3) ότι εντοπίστηκαν 8 επιθέσεις σύμφωνα με τον δεύτερο έλεγχο και στην περιοχή (6) ότι εντοπίστηκε το 80% των επιθέσεων. Αν ο χρήστης επιθυμεί να κλείσει τη φόρμα του Σχήματος 4.42, αρκεί να επιλέξει το κουμπί Back αυτής, οπότε και θα εξαφανιστεί.

Για να ολοκληρώσουμε την παρουσίαση του Συστήματος *WaterGraph* αρκεί να παρουσιάσουμε και τις φόρμες που υλοποιούν την πειραματική έρευνα για τον Αλγόριθμο `CN`. Για να διεξάγει ο χρήστης πειράματα για τον Αλγόριθμο `CN` αρκεί στο πεδίο (1) του Σχήματος 4.29 να επιλέξει τον συγκεκριμένο αλγόριθμο και να πατήσει το κουμπί Experimental Mode. Η φόρμα που θα εμφανιστεί είναι αυτή του Σχήματος 4.46.



Σχήμα 4.46: Φόρμα πειραμάτων Αλγορίθμου CN

Παρατηρείται ότι όπως και στη φόρμα πειραμάτων του Αλγορίθμου CKCT έτσι και εδώ στην κορύφη αυτής υπάρχουν πληροφορίες για τον αλγόριθμο που χρησιμοποιείται από τον χρήστη. Οι περιοχές (1), (2), (3) και (4) επιτελούν ακριβώς τον ίδιο σκοπό όπως και στην προαναφερθείσα φόρμα, επομένως δεν θα τις περιγράψουμε εκτενέστερα. Επίσης, τα κουμπιά Save As και Run εκτελούν ακριβώς τις ίδιες λειτουργίες. Κατα την διεξαγωγή πειραμάτων απενεργοποιούνται τα κουμπιά της φόρμας και ενεργοποιούνται μετα την ολοκλήρωση τους. Επίσης, εμφανίζεται και εδώ το παράθυρο του Σχήματος 4.45 που μας αναφέρει τα αποτελέσματα των πειραμάτων καθώς επίσης και τα αντίστοιχα διαγράμματα πίτας. Πρέπει να τονίσουμε ότι αυτά τα διαγράμματα πίτας δεν αντιστοιχούν στα αποτελέσματα που επιστρέφει ο Αλγόριθμος CN αλλά σε αυτά του Αλγορίθμου CKCT. Εάν κάποιος επιθυμεί να δει τα αποτελέσματα πρέπει να εκτελέσει το σύστημα μας και να εισάγει αυτά τα δεδομένα. Η σημαντικότερη διαφορά μεταξύ των συγκρινόμενων φορμών είναι ο τρόπος με τον οποίο δημιουργούν τα γραφήματα καθώς καθένας από τους δύο αλγορίθμους έχει διαφορετική κωδικοποίηση.

Εδώ ολοκληρώνεται η παρουσίαση του Συστήματος *WaterGraph* και των λειτουργιών του. Στο επόμενο κεφάλαιο θα παρουσιάσουμε τα πειραματικά αποτελέσματα που εξήχθησαν από το παραπάνω σύστημα και για τους δύο αλγορίθμους έτσι ώστε να μπορέσουμε να συγκρίνουμε την ανθεκτικότητα αυτών.

Κεφαλαιο 5

Πειραματικα Αποτελεσματα

5.1 Εισαγωγή

Στην τεχνική της υδατοσήμανσης λογισμικού που είναι βασισμένη σε γραφήματα ιδιαίτερη σημασία παρουσιάζει η ανθεκτικότητα των δομών που χρησιμοποιούνται από τον εκάστοτε αλγόριθμο σε επιθέσεις. Στόχος αυτού του κεφαλαίου είναι η παρουσίαση των πειραματικών αποτελεσμάτων που εξήχθησαν από την μελέτη των δύο αλγορίθμων που παρουσιάστηκαν στο Κεφάλαιο 3 μέσω του Συστήματος *WaterGraph*.

Συγκεκριμένα, στις Ενότητες 5.2 και 5.3 θα παρουσιαστούν τα πειραματικά αποτελέσματα που εξήχθησαν από τους Αλγορίθμους CKCT και CN αντίστοιχα μέσω του συστήματος που παρουσιάσαμε στο Κεφάλαιο 4 για όλων των ειδών τις επιθέσεις δίδοντας συγκεκριμένη είσοδο.

5.2 Πειραματικά Αποτελέσματα Αλγορίθμου CKCT

Στην παρούσα ενότητα θα παρουσιάσουμε αναλυτικά και θα σχολιάσουμε τα πειραματικά αποτελέσματα που προκύπτουν για τον Αλγορίθμο CKCT για κάθε μια από τις ακόλουθες επιθέσεις: αλλαγή κατεύθυνσης ακμής, προσθήκη ακμής, διαγραφή ακμής, αλλαγή προορισμού ακμής, προσθήκη κόμβου και τέλος διαγραφή κόμβου. Αξίζει να σημειωθεί ότι τα πειραματικά αποτελέσματα προκύπτουν δημιουργώντας κάθε φορά ένα γράφημα για μια συγκεκριμένη τιμή υδατοσήμου και εφαρμόζοντας μια μοναδική επίθεση την φόρα. Τα πειράματα που θα παρουσιάσουμε για όλες τις παραπάνω επιθέσεις είναι τα εξής: στο Πείραμα

1 θα εφαρμόσουμε 1000, 2000, 3000, 4000 και 5000 πειράματα για τα μεγέθη γραφημάτων 5, 7, 9, 11 και 13, στο Πείραμα 2 θα εφαρμόσουμε 10, 20, 30, 40 και 50 πειράματα για τις τιμές υδατοσήμου 10, 100, 1000, 10000 και 100000 και τέλος στο Πείραμα 3 θα εφαρμόσουμε ξανά το ίδιο πλήθος πειραμάτων όπως και στο πρώτο, όμως αυτή τη φορά για τυχαίες τιμές υδατοσήμων στα διαστήματα 1 έως 99 (small values), 100 έως 999 (medium values) και 1000 έως 10000 (large values). Μετά το πέρας μιας επίθεσης σε ένα γράφημα γίνονται δύο διαφορετικοί έλεγχοι, από τους οποίους ο πρώτος ελέγχει γενικά αν το γράφημα παραμένει αναγώγιμο μετά την επίθεση και ο δεύτερος αν το γράφημα παραμένει αναγώγιμο σε συνδυασμό με το αν οι κόμβοι έχουν πλήθος εξερχόμενων ακμών το πολύ δύο. Σε κάθε πείραμα θα εμφανίζονται με την μορφή ραβδογραμμάτων τα αποτελέσματα που επιστράφηκαν όμως στο Πείραμα 1 θα εμφανίζονται επιπλέον και τα ποσοστά των επιθέσεων που εντοπίστηκαν για όλα τα μεγέθη γραφημάτων για κάθε πλήθος πειραμάτων με μορφή διαγραμμάτων πίτας.

5.2.1 Επίθεση Αλλαγής Κατεύθυνσης Ακμής

■ **Πείραμα 1:**

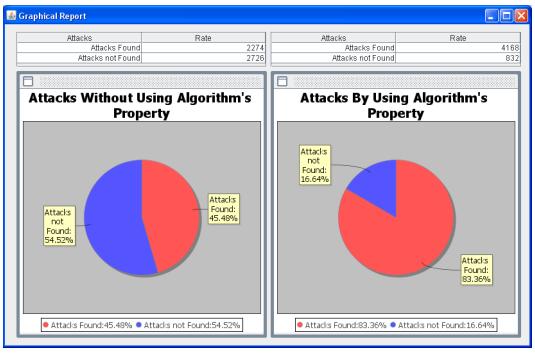
Στους παρακάτω πίνακες παρουσιάζονται τα αποτελέσματα που προκύπτουν εφαρμόζοντας στον Αλγόριθμο CKCT το πρώτο πείραμα που περιγράφηκε προηγουμένως.

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	337	433	447	515	542
2000	660	855	980	993	1075
3000	1036	1293	1441	1499	1615
4000	1305	1692	1910	2028	2163
5000	1724	2153	2392	2577	2666

Πίνακας 5.1: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	677	837	872	876	906
2000	1339	1657	1731	1776	1811
3000	2014	2504	2604	2678	2708
4000	2644	3313	3459	3538	3638
5000	3345	4157	4319	4472	4506

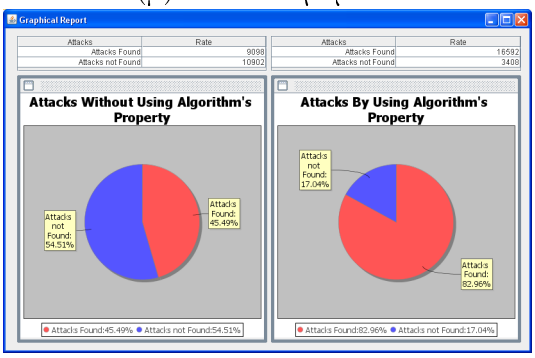
Πίνακας 5.2: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο



(α) 1000 Πειράματα

(β) 2000 Πειράματα

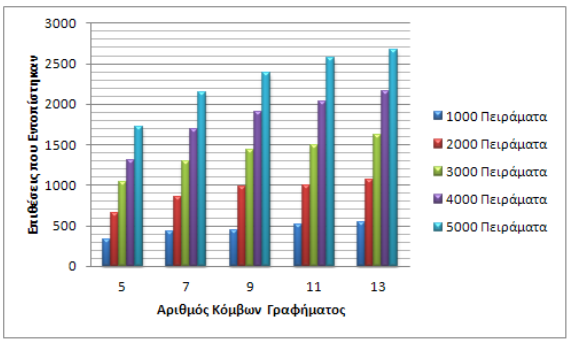
(γ) 3000 Πειράματα



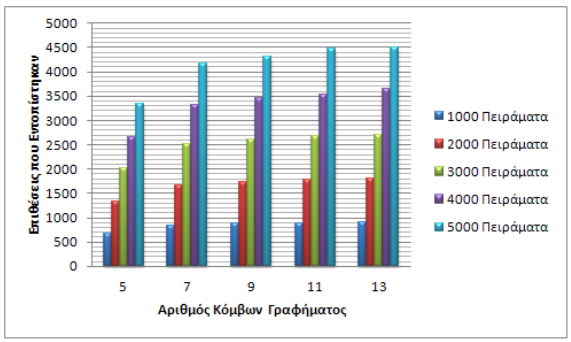
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.1: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.2: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Στον Πίνακα 5.1 εμφανίζονται τα αποτελέσματα που εξάγονται από τον πρώτο έλεγχο που προαναφέρθηκε, ενώ στον 5.2 τα αποτελέσματα από τον δεύτερο έλεγχο. Μπορούμε να παρατηρήσουμε ότι ο αριθμός των επιθέσεων που γίνεται αντιληπτός αυξάνεται ελέγχοντας εκτός του αν το γράφημα είναι αναγώγιμο και το πλήθος των εξερχόμενων ακμών των κόμβων. Αυτό φυσικά είναι αναμενόμενο καθώς υπάρχει μεγάλη πιθανότητα με μια αλλαγή κατεύθυνσης το γράφημα να παραμένει αναγώγιμο, όμως το πλήθος των εξερχόμενων ακμών να είναι μεγαλύτερο από δύο. Όμοια αποτελέσματα παρατηρούνται και στα άλλα δύο πειράματα που ακολουθούν.

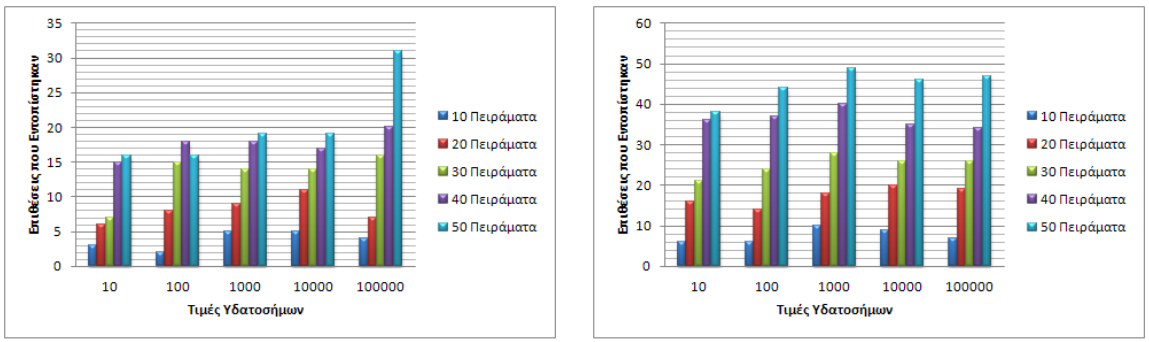
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	3	2	5	5	4
20	6	8	9	11	7
30	7	15	14	14	16
40	15	18	18	17	20
50	16	16	19	19	31

Πίνακας 5.3: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	6	6	10	9	7
20	16	14	18	20	19
30	21	24	28	26	26
40	36	37	40	35	34
50	38	44	49	46	47

Πίνακας 5.4: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.3: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

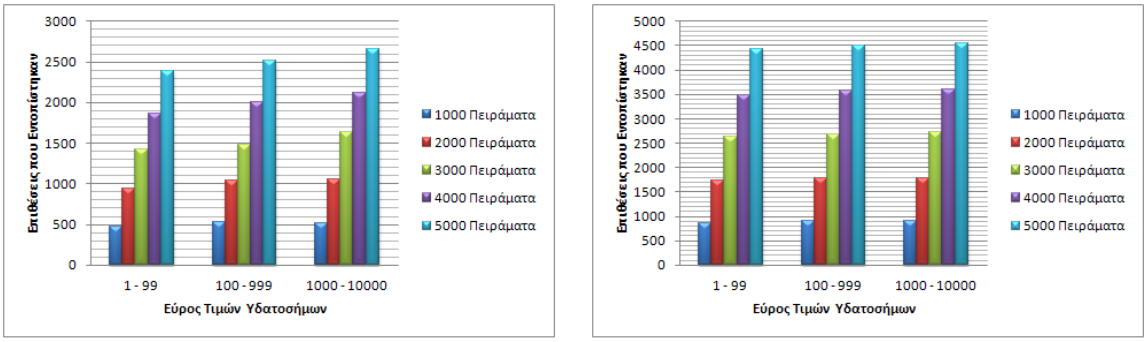
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	474	523	519
2000	944	1033	1053
3000	1428	1478	1635
4000	1867	2002	2116
5000	2391	2509	2660

Πίνακας 5.5: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	854	912	906
2000	1736	1789	1783
3000	2622	2674	2718
4000	3488	3577	3611
5000	4424	4491	4541

Πίνακας 5.6: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.4: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.2.2 Επίθεση Προσθήκης Ακμής

■ **Πείραμα 1:**

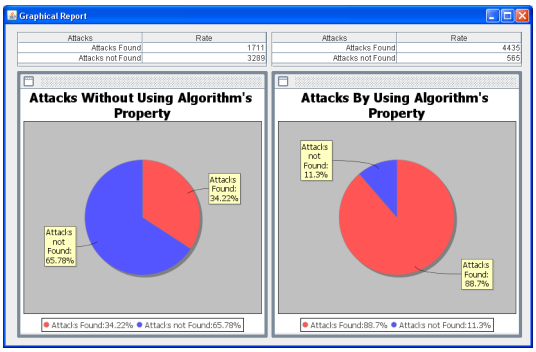
Εφαρμόζοντας το πρώτο πείραμα προκύπτουν τα ακόλουθα πειραματικά αποτελέσματα:

Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	247	327	351	397	389
2000	514	661	718	768	778
3000	692	988	1087	1099	1166
4000	1000	1334	1455	1504	1558
5000	1274	1686	1767	1789	1939

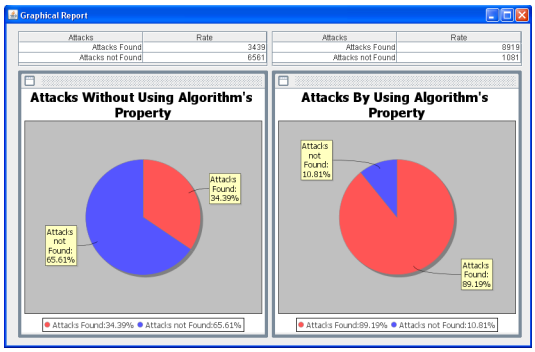
Πίνακας 5.7: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	843	879	909	907	897
2000	1717	1783	1792	1814	1813
3000	2526	2717	2725	2720	2715
4000	3397	3535	3629	3644	3647
5000	4235	4439	4512	4537	4556

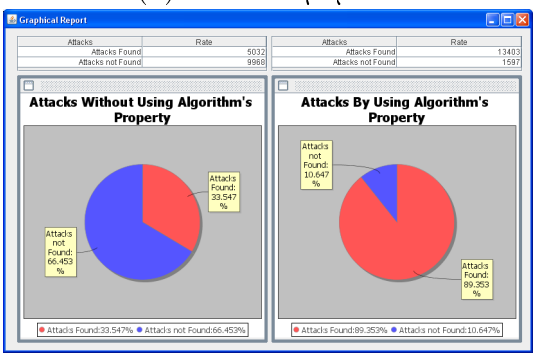
Πίνακας 5.8: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



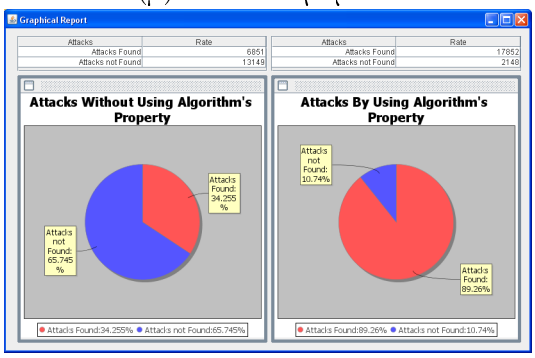
(α) 1000 Πειράματα



(β) 2000 Πειράματα



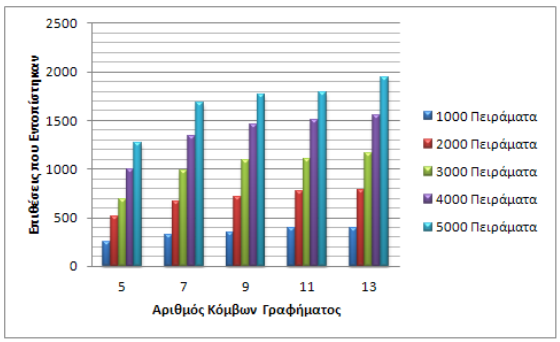
(γ) 3000 Πειράματα



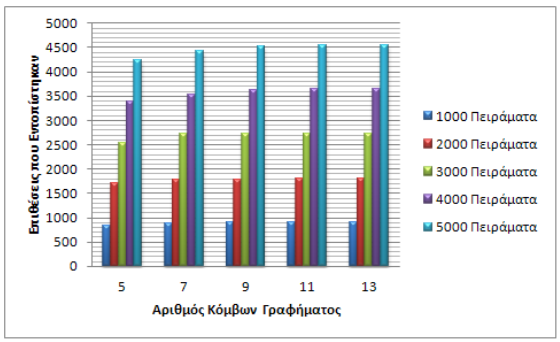
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.5: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.6: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Μπορούμε να παρατηρήσουμε ότι τα αποτελέσματα που παίρνουμε σύμφωνα με τον δεύτερο έλεγχο είναι πολύ καλύτερα από αυτά του πρώτου, γεγονός που οφείλεται στην χρήση της ιδιότητας που έχουν οι κόμβοι αυτών των γραφημάτων που δημιουργούνται από τον Αλγόριθμο CKCT, δηλαδή ότι το πλήθος των εξερχόμενων ακμών είναι το πολύ δύο. Αυτό το φαινόμενο παρατηρείται και στα Πειράματα 2 και 3 και ο λόγος είναι ο ίδιος καθώς με την προσθήκη μιας ακμής υπάρχει μεγάλη πιθανότητα ένας κόμβος να έχει βαθμό εξερχόμενων ακμών μεγαλύτερο από δύο. Τα παραπάνω αποτελέσματα φαίνονται και στα ραβδογράμματα του Σχήματος 5.6.

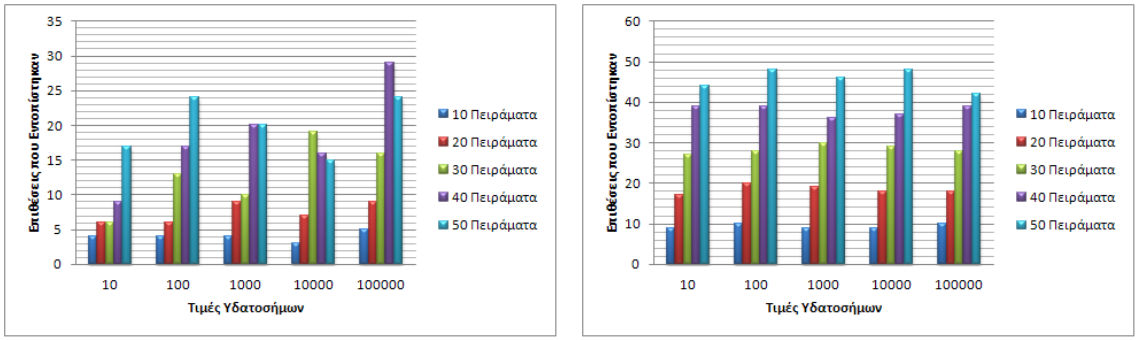
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	4	4	4	3	5
20	6	6	9	7	9
30	6	13	10	19	16
40	9	17	20	16	29
50	17	24	20	15	24

Πίνακας 5.9: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	9	10	9	9	10
20	17	20	19	18	18
30	27	28	30	29	28
40	39	39	36	37	39
50	44	48	46	48	42

Πίνακας 5.10: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.7: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

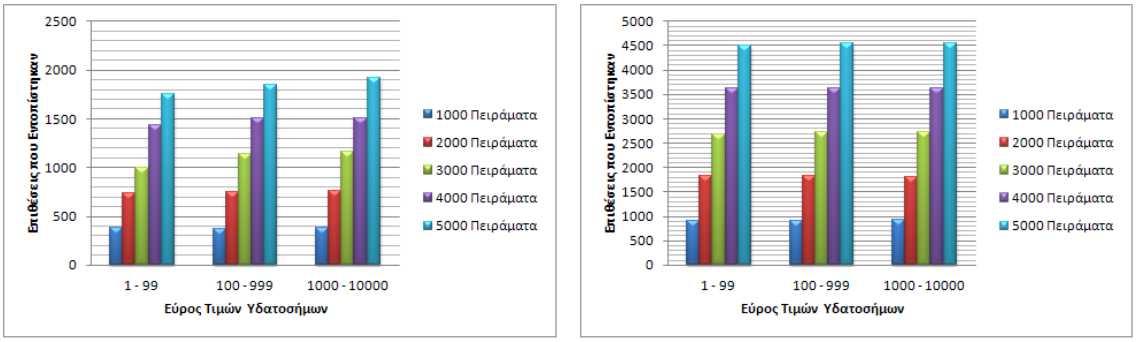
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	387	367	379
2000	732	744	764
3000	997	1140	1161
4000	1432	1503	1501
5000	1751	1851	1915

Πίνακας 5.11: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	907	903	918
2000	1819	1832	1807
3000	2690	2726	2728
4000	3631	3634	3622
5000	4495	4552	4544

Πίνακας 5.12: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.8: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.2.3 Επίθεση Διαγραφής Ακμής

Σε αυτή την υποενότητα θα εξετάσουμε την ανθεκτικότητα των γραφημάτων που δημιουργούνται από τον Αλγόριθμο CKCT αν εφαρμόσουμε επιθέσεις διαγραφής ακμής. Τα αποτελέσματα από το Πείραμα 1 φαίνονται στους Πίνακες 5.13 και 5.14.

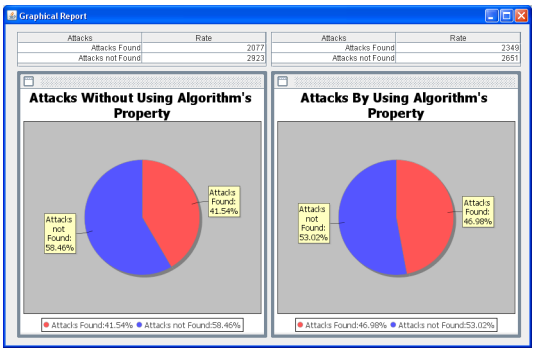
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	524	379	417	360	397
2000	994	793	805	802	762
3000	1500	1217	1153	1209	1190
4000	1983	1621	1579	1524	1497
5000	2487	2028	1972	1919	1901

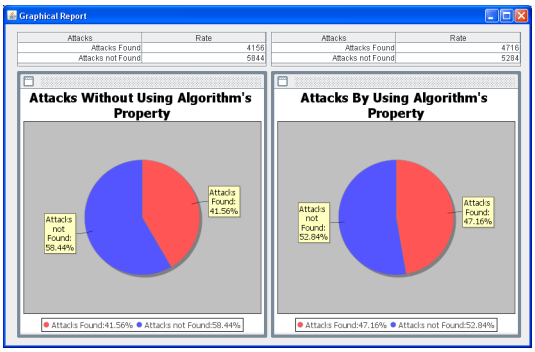
Πίνακας 5.13: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	524	441	473	443	468
2000	994	926	946	955	895
3000	1500	1388	1337	1415	1416
4000	1983	1884	1857	1832	1775
5000	2487	2339	2353	2293	2260

Πίνακας 5.14: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



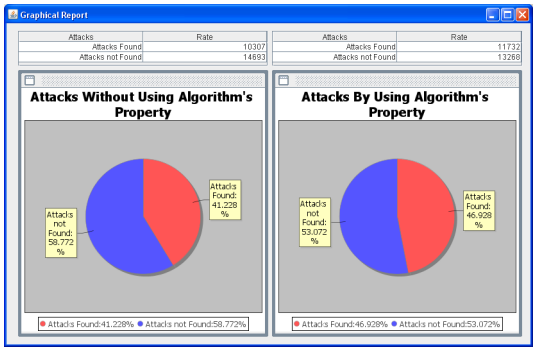
(α) 1000 Πειράματα



(β) 2000 Πειράματα

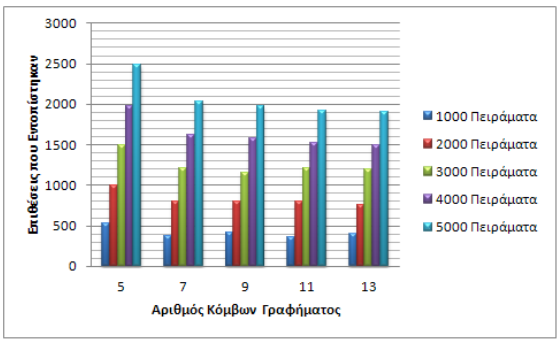
(γ) 3000 Πειράματα

(δ) 4000 Πειράματα

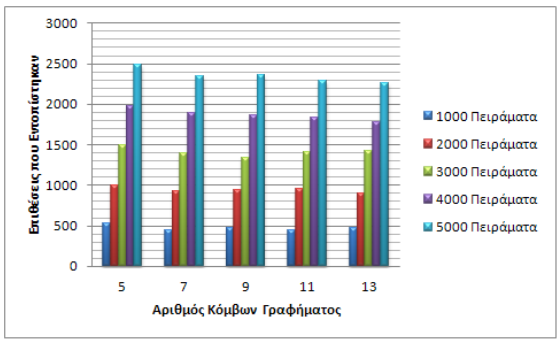


(ε) 5000 Πειράματα

Σχήμα 5.9: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.10: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Από τα παραπάνω αποτελέσματα συμπεραίνουμε ότι και με τους δύο ελέγχους οι επιθέσεις που εντοπίζονται δεν είναι αρκετές. Αυτό συμβαίνει διότι όταν διαγράφουμε μια ακμή δεν γίνεται το πλήθος των εξερχόμενων ακμών να ξεπεράσει την τιμή 2. Ο δεύτερος έλεγχος έχει λίγο καλύτερα ποσοστά από τον πρώτο γιατι ελέγχει αν οι κόμβοι έχουν τουλάχιστον μια εξερχόμενη ακμή. Όμοια αποτελέσματα προκύπτουν για τα Πειράματα 2 και 3.

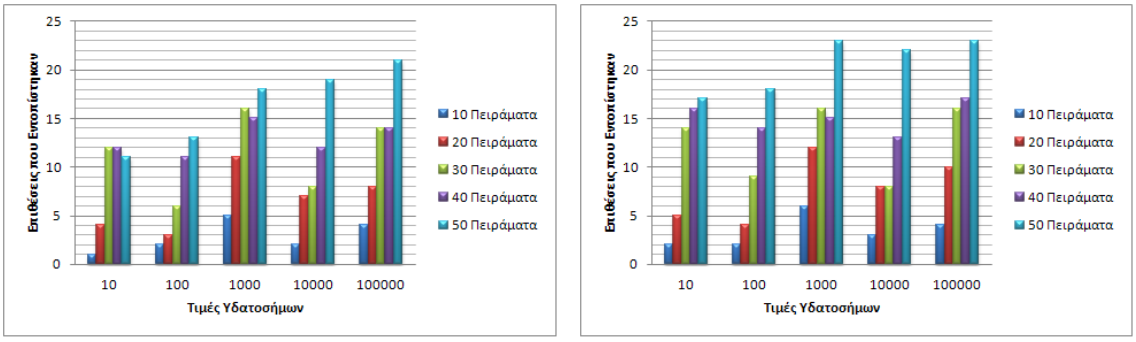
■ **Πείραμα 2:**

Τιμές Υδατοσήμων / Αριθμός Πειραμάτων	10	100	1000	10000	100000
10	1	2	5	2	4
20	4	3	11	7	8
30	12	6	16	8	14
40	12	11	15	12	14
50	11	13	18	19	21

Πίνακας 5.15: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Τιμές Υδατοσήμων / Αριθμός Πειραμάτων	10	100	1000	10000	100000
10	2	2	6	3	4
20	5	4	12	8	10
30	14	9	16	8	16
40	16	14	15	13	17
50	17	18	23	22	23

Πίνακας 5.16: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.11: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

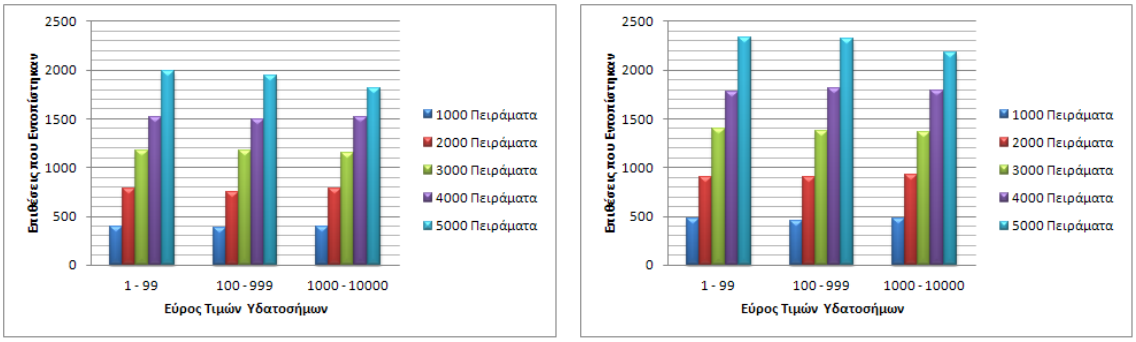
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	399	387	394
2000	780	752	782
3000	1174	1177	1147
4000	1516	1497	1522
5000	1992	1942	1816

Πίνακας 5.17: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	473	453	482
2000	907	897	922
3000	1393	1379	1358
4000	1778	1807	1793
5000	2331	2322	2180

Πίνακας 5.18: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.12: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.2.4 Επίθεση Αλλαγης Προορισμού Ακμής

Η τελευταία κατηγορία επίθεσης η οποία εφαρμόζεται σε ακμές των γραφημάτων που δημιουργούνται από τον Αλγόριθμο CKCT είναι αυτή της αλλαγής προορισμού μιας ακμής. Τα αποτελέσματα που προκύπτουν για κάθε πείραμα είναι τα εξής:

■ **Πείραμα 1:**

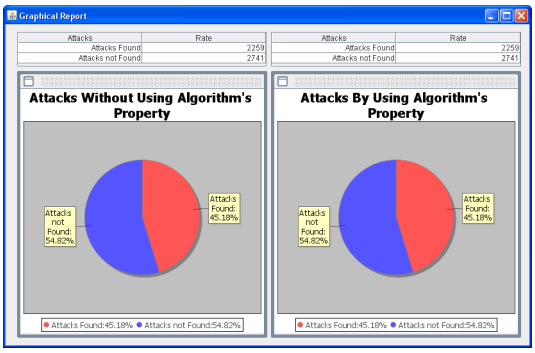
Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	430	398	430	491	510
2000	871	788	860	909	950
3000	1345	1194	1275	1375	1440
4000	1709	1594	1692	1847	1940
5000	2184	1898	2182	2301	2353

Πίνακας 5.19: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	430	398	430	491	510
2000	871	788	860	909	950
3000	1345	1194	1275	1375	1440
4000	1709	1594	1692	1847	1940
5000	2184	1898	2182	2301	2353

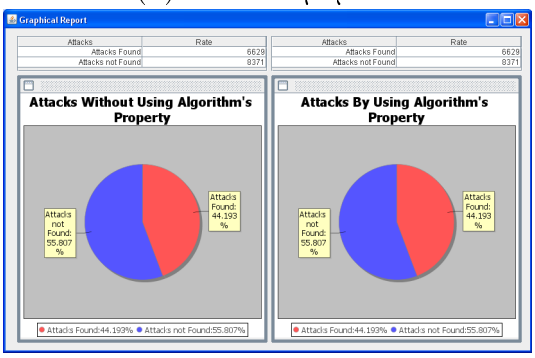
Πίνακας 5.20: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο

Οι παραπάνω πίνακες αποτελούν τις επιθέσεις που γίνονται αντιληπτές από τον πρώτο και τον δεύτερο έλεγχο αντίστοιχα, εάν εφαρμόσουμε το πρώτο πείραμα. Μπορούμε εύκολα να παρατηρήσουμε ότι και με τους δύο ελέγχους επιστρέφονται ακριβώς τα ίδια αποτελέσματα γεγονός που είναι απόλυτα φυσιολογικό καθώς με το να αλλάξουμε τον προορισμό μιας ακμής δεν επηρεάζεται ο βαθμός των εξερχόμενων ακμών κάποιου κόμβου, επομένως η ιδιότητα για τους βαθμούς των κόμβων δεν χρησιμεύει σε τέτοιου είδους επιθέσεις. Στο Σχήμα 5.13 παρουσιάζονται γραφικά κατά μέσο όρο πόσες επιθέσεις γίνονται αντιληπτές αν εφαρμόσουμε x πειράματα για 5, 7, 9, 11 και 13 κόμβους. Επίσης, δίδουμε τα ραβδογράμματα που αντιστοιχούν στα αποτελέσματα που παρουσιάστηκαν στους παραπάνω πίνακες.

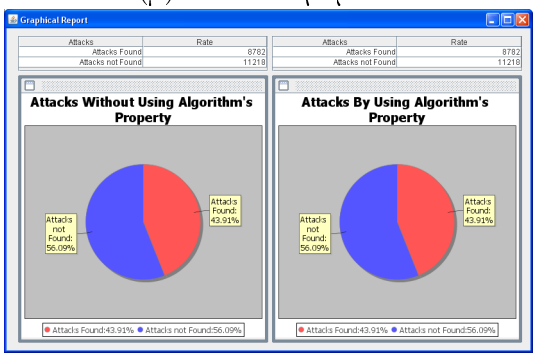


(α) 1000 Πειράματα

(β) 2000 Πειράματα



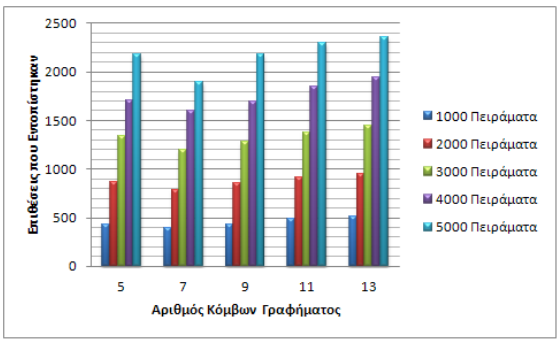
(γ) 3000 Πειράματα



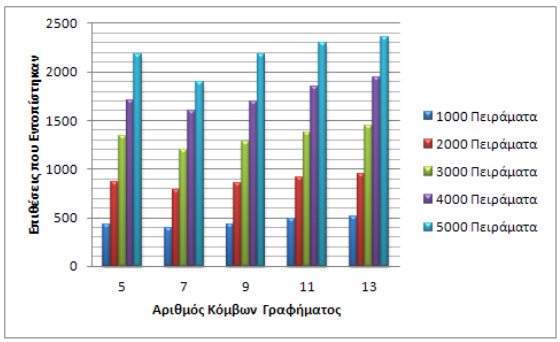
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.13: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.14: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Μπορούμε να παρατηρήσουμε ότι κατα μέσο όρο το ποσοστό των επιθέσεων που γίνονται αντιληπτές είναι σχεδόν ίδιο για 1000, 2000, 3000, 4000 και 5000 πειράματα. Στη συνέχεια παραθέτουμε και τα αποτελέσματα που προκύπτουν αν εφαρμόσουμε τα Πειράματα 2 και 3.

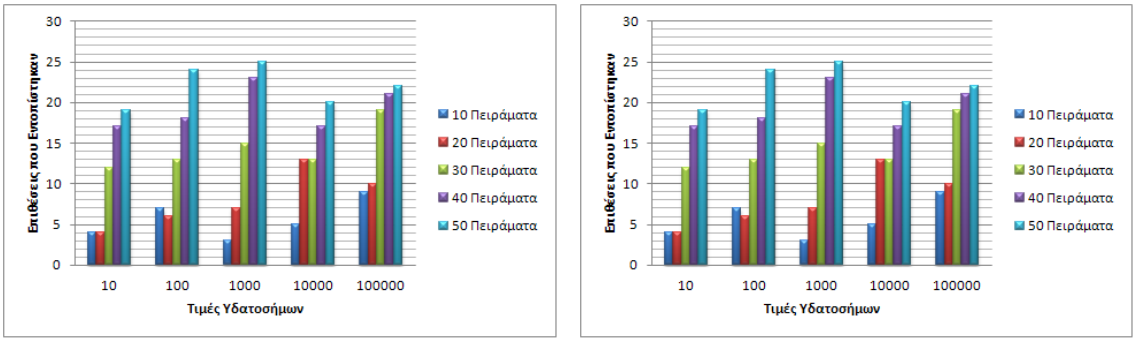
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	4	7	3	5	9
20	4	6	7	13	10
30	12	13	15	13	19
40	17	18	23	17	21
50	19	24	25	20	22

Πίνακας 5.21: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	4	7	3	5	9
20	4	6	7	13	10
30	12	13	15	13	19
40	17	18	23	17	21
50	19	24	25	20	22

Πίνακας 5.22: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.15: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

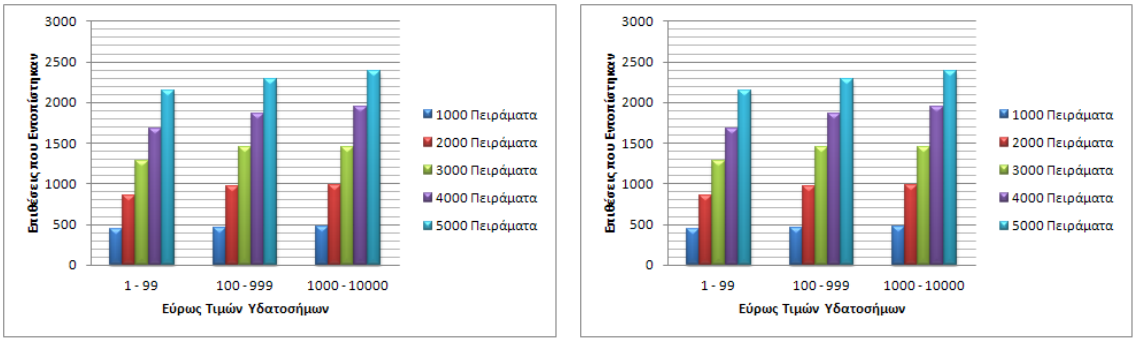
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	438	455	476
2000	853	963	989
3000	1284	1448	1450
4000	1675	1861	1949
5000	2153	2293	2393

Πίνακας 5.23: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	438	455	476
2000	853	963	989
3000	1284	1448	1450
4000	1675	1861	1949
5000	2153	2293	2393

Πίνακας 5.24: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.16: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Αξίζει να λεχθεί ότι όπως και στο Πείραμα 1 έτσι και στα 2 και 3 επιστρέφονται ίδια αποτελέσματα τόσο για τον πρώτο όσο και για τον δεύτερο έλεγχο. Ο λόγος που συμβαίνει αυτό είναι ο ίδιος με αυτόν του πρώτου πειράματος.

5.2.5 Επίθεση Προσθήκης Κόμβου

Έχοντας αναφέρει όλες τις επιθέσεις που αφορούν ακμές μπορούμε πλέον να ασχοληθούμε με αυτές που προσθέτουν ή διαγράφουν έναν κόμβο. Στην παρούσα υποενότητα θα παραθέσουμε και θα σχολιάσουμε τα αποτελέσματα που εξάγονται από την προσθήκη ενός κόμβου σε ένα αναγώγιμο μεταθετικό γράφημα κάνοντας χρήση και των δύο προαναφερθέντων ελέγχων. Αρχικά, παραθέτουμε στους παρακάτω πίνακες τα αποτελέσματα του πρώτου πειράματος καθώς επίσης και την γραφική αναπαράσταση του μέσου όρου αυτών έτσι όπως εμφανίζεται από το Σύστημα *WaterGraph*. Επίσης, παρουσιάζουμε με την μορφή ραβδογραμμάτων τα αποτελέσματα που είναι εγγεγραμμένα στους δύο αυτούς πίνακες.

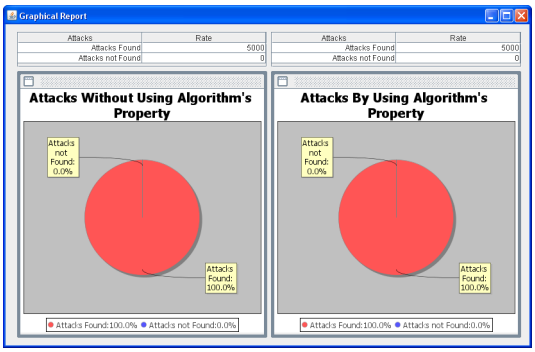
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

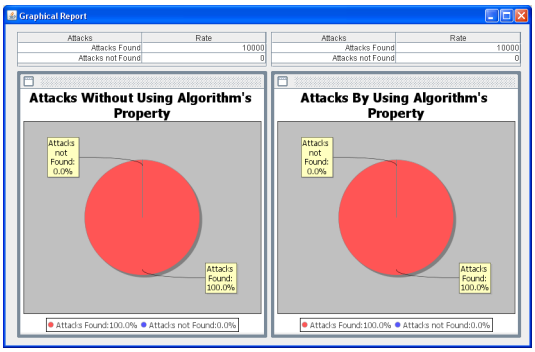
Πίνακας 5.25: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

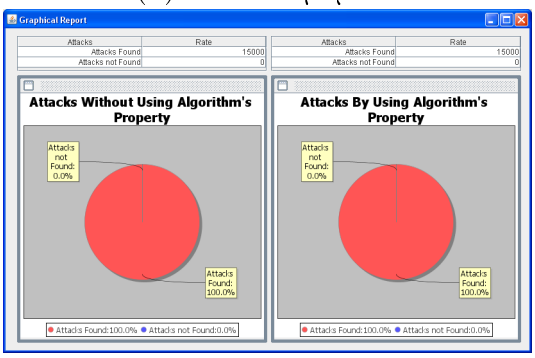
Πίνακας 5.26: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



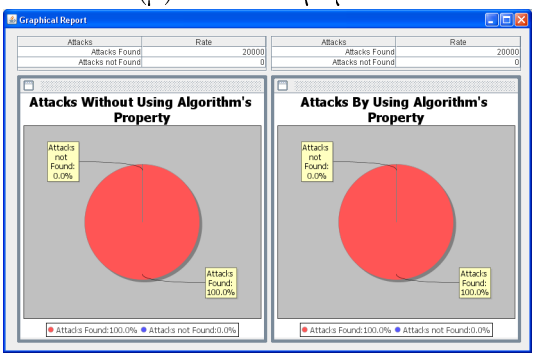
(α) 1000 Πειράματα



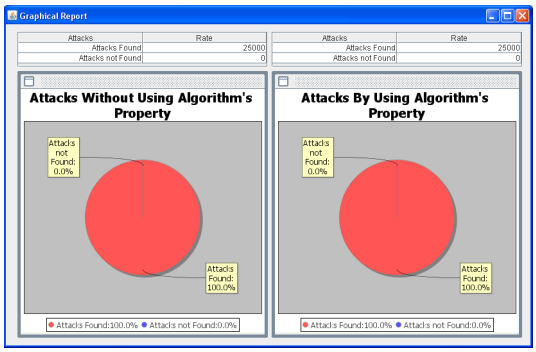
(β) 2000 Πειράματα



(γ) 3000 Πειράματα

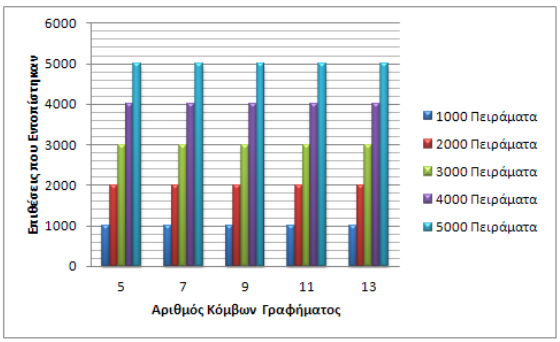


(δ) 4000 Πειράματα

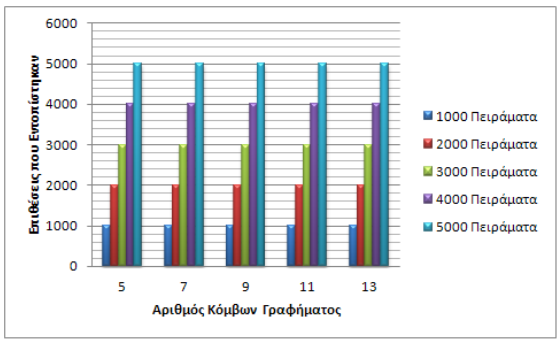


(ε) 5000 Πειράματα

Σχήμα 5.17: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.18: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Παρατηρούμε ότι και με τους δύο ελέγχους εντοπίζονται όλες οι επιθέσεις προσθήκης κόμβου. Αυτό είναι απόλυτα σωστό καθώς όπως τονίσαμε στην αρχή της Ενότητας 5.2 κάθε φορά εφαρμόζεται μια μοναδική επίθεση. Μια μοναδική επίθεση όμως προσθήκης κόμβου μπορεί να εντοπιστεί άμεσα κάνοντας χρήση μόνο του πρώτου ελέγχου. Παρόμοια αποτελέσματα θα παρατηρήσουμε και στα Πειράματα 2 και 3.

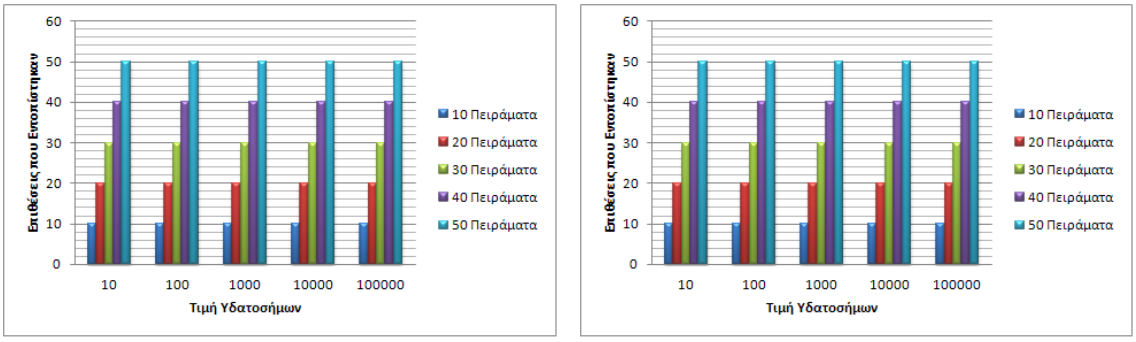
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.27: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.28: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.19: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

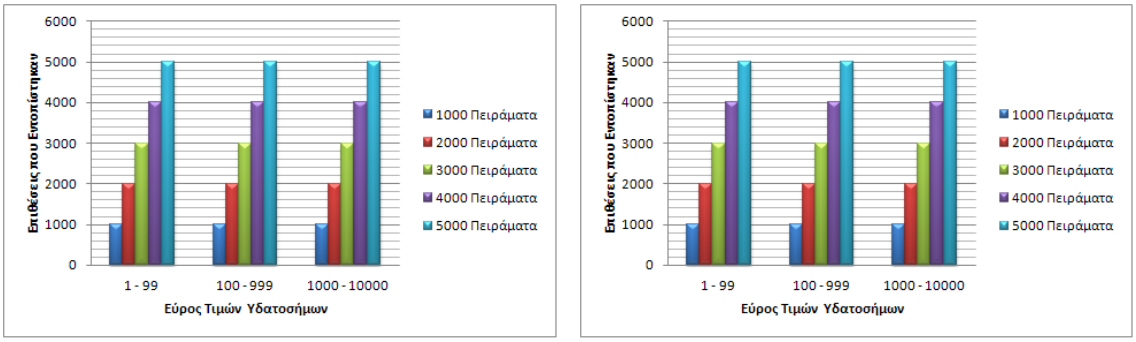
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.29: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.30: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.20: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.2.6 Επίθεση Διαγραφής Κόμβου

Η τελευταία επίθεση την οποία θα εφαρμόσουμε για να διαπιστώσουμε πόσο ανθεκτικός είναι ο Αλγόριθμος CKCT είναι αυτή της διαγραφής ενός κόμβου από το γράφημα.
■ **Πείραμα 1:**

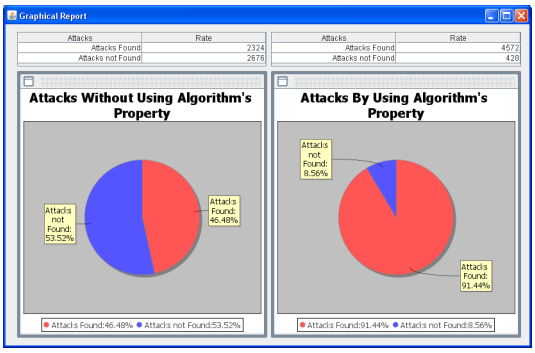
Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	392	409	476	534	513
2000	826	889	944	997	1080
3000	1209	1272	1429	1517	1538
4000	1544	1710	1859	2004	2081
5000	1955	2191	2448	2491	2607

Πίνακας 5.31: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

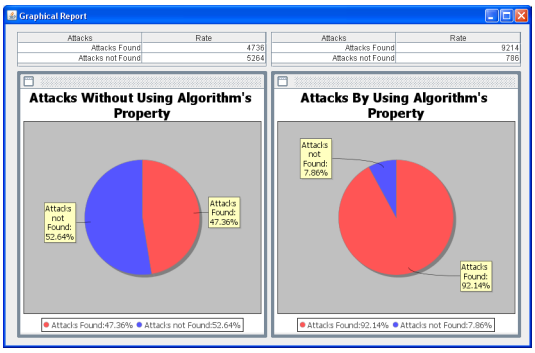
Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	791	885	950	970	976
2000	1604	1828	1907	1922	1953
3000	2386	2713	2850	2886	2913
4000	3175	3611	3788	3858	3891
5000	3958	4498	4743	4817	4871

Πίνακας 5.32: Αλγόριθμος CKCT, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο

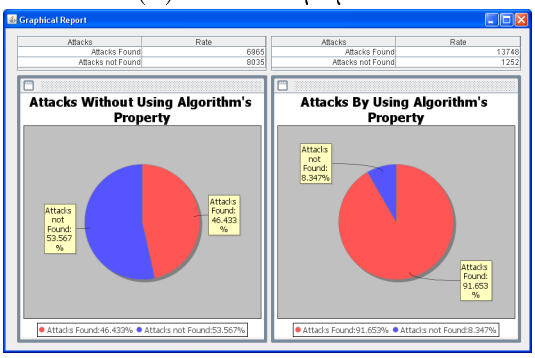
Αν κοιτάξουμε προσεκτικά τα αποτελέσματα που είναι εγγεγραμμένα στους δύο πίνακες θα διαπιστώσουμε ότι προκύπτουν καλύτερα αποτελέσματα χρησιμοποιώντας τον δεύτερο έλεγχο. Αυτό οφείλεται στο γεγονός ότι με τη διαγραφή ενός κόμβου διαγράφονται και οι ακμές που εισέρχονται ή εξέρχονται από αυτόν, επομένως υπάρχει μεγάλη πιθανότητα ο βαθμός των εξερχόμενων ακμών ενός άλλου κόμβου να γίνει μηδενικός. Άρα, έτσι όπως δημιουργούνται τα γραφήματα από αυτόν τον αλγόριθμο δίνουν την δυνατότητα να εντοπίσουμε επιθέσεις χάριν της ιδιότητας που διαθέτουν οι κόμβοι, δηλαδή ότι ο βαθμός των εξερχόμενων ακμών των κόμβων του σώματος του γραφήματος (όλοι εκτός του κόμβου h και του f) δεν είναι μικρότερος του 1 και μεγαλύτερος του 2. Οι επιθέσεις που διαφεύγουν του εντοπισμού είναι περιπτώσεις διαγραφής του κόμβου f ή όπως αλλιώς τον ονομάζουμε v_{k+1}, όπου k το πλήθος των κόμβων του σώματος.



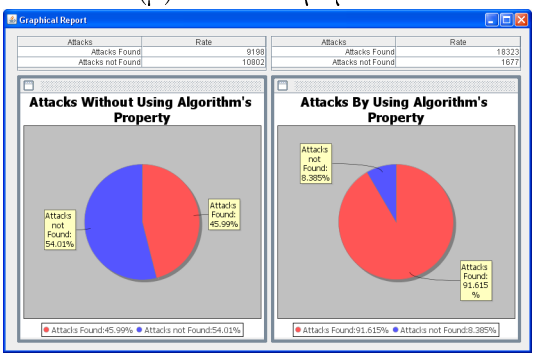
(α) 1000 Πειράματα



(β) 2000 Πειράματα



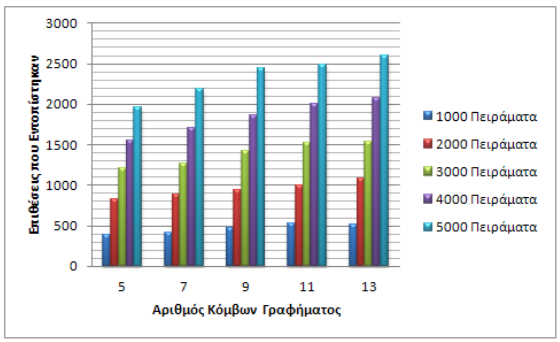
(γ) 3000 Πειράματα



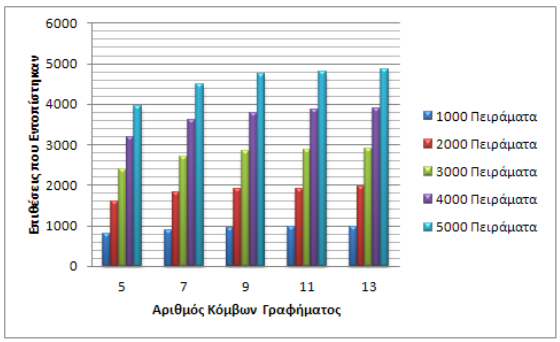
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.21: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.22: Αλγόριθμος CKCT, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Παρόμοια αποτελέσματα θα διαπιστώσουμε και στα ακόλουθα δύο πειράματα λόγω της ίδιας ιδιότητας.

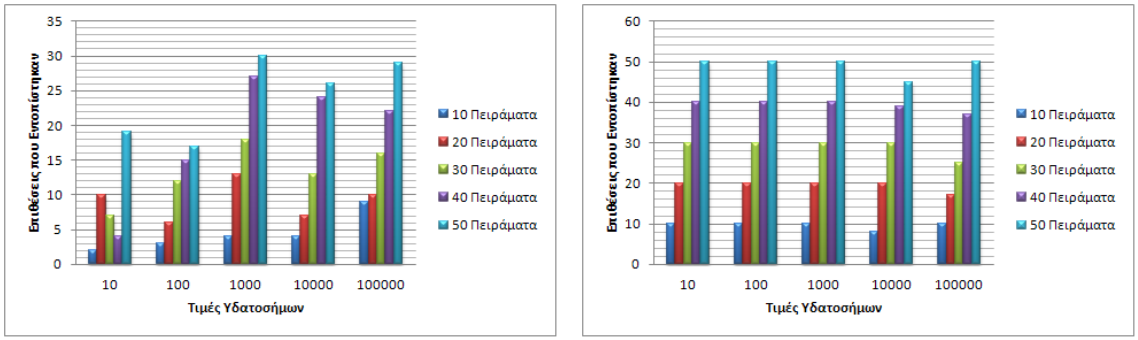
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	2	3	4	4	9
20	10	6	13	7	10
30	7	12	18	13	16
40	4	15	27	24	22
50	19	17	30	26	29

Πίνακας 5.33: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	8	10
20	20	20	20	20	17
30	30	30	30	30	25
40	40	40	40	39	37
50	50	50	50	45	50

Πίνακας 5.34: Αλγόριθμος CKCT, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.23: Αλγόριθμος CKCT, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

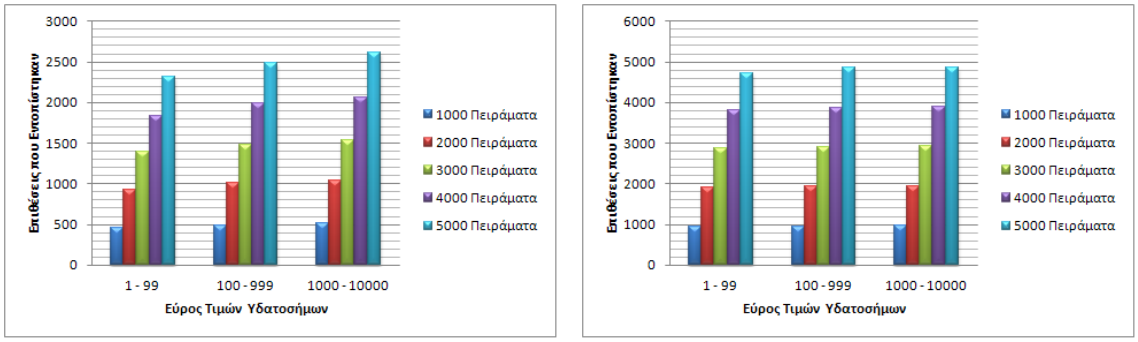
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	465	485	515
2000	926	1013	1036
3000	1398	1485	1536
4000	1829	1993	2066
5000	2315	2482	2612

Πίνακας 5.35: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	946	959	978
2000	1901	1933	1941
3000	2867	2900	2918
4000	3805	3865	3894
5000	4717	4848	4846

Πίνακας 5.36: Αλγόριθμος CKCT, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.24: Αλγόριθμος CKCT, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.3 Πειραματικά Αποτελέσματα Αλγορίθμου CN

Στην Ενότητα 5.2 παρουσιάσαμε τα πειραματικά αποτελέσματα που εξήχθησαν εφαρμόζοντας τρία διαφορετικά πειράματα για κάθε κατηγορία επίθεσης πάνω σε αναγώγιμα μεταθετικά γραφήματα που δημιουργούνται από τον Αλγόριθμο CKCT. Σε αυτή την ενότητα θα παρουσιάσουμε τα αποτελέσματα που προκύπτουν εκτελώντας τα ίδια πειράματα και επιθέσεις σε αναγώγιμα μεταθετικά γραφήματα που δημιουργούνται από τον Αλγόριθμο CN.

Στα γραφήματα που δημιουργούνται από τον Αλγόριθμο CN κάθε κόμβος του σώματος έχει πλήθος εξερχόμενων ακμών ακριβώς 2. Πάνω σε αυτή την ιδιότητα θα στηριχθεί ο δεύτερος έλεγχος καθώς επίσης και στο Θεώρημα Α της υποενότητας 3.3.5. Ο πρώτος έλεγχος είναι ίδιος με αυτόν της προηγούμενης ενότητας. Πρέπει να τονίσουμε ότι τα αποτελέσματα κάθε πειράματος θα παρουσιάζονται ακριβώς με τον ίδιο τρόπο που παρουσιάζονταν και για τον Αλγόριθμο CKCT.

5.3.1 Επίθεση Αλλαγής Κατεύθυνσης Ακμής

Σε αυτή την υποενότητα παρουσιάζουμε τα αποτελέσματα που προκύπτουν εκτελώντας τα τρία πειράματα (βλέπε Ενότητα 5.2) για επιθέσεις αλλαγής κατεύθυνσης ακμής.

■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	414	503	578	598	627
2000	851	979	1145	1232	1270
3000	1328	1540	1713	1795	1934
4000	1746	2018	2260	2413	2462
5000	2125	2517	2770	3029	3130

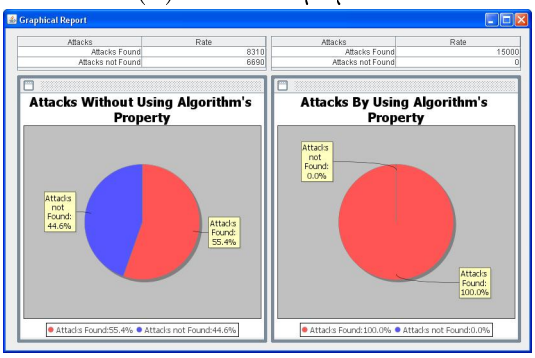
Πίνακας 5.37: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

Πίνακας 5.38: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο

(α) 1000 Πειράματα

(β) 2000 Πειράματα

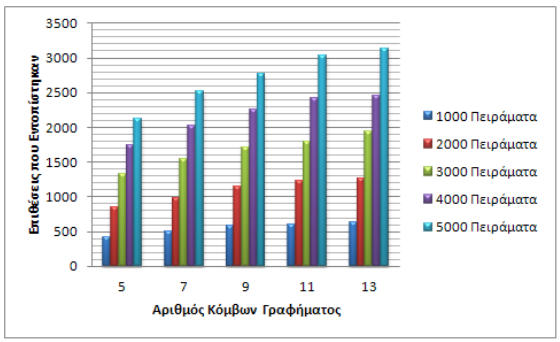


(γ) 3000 Πειράματα

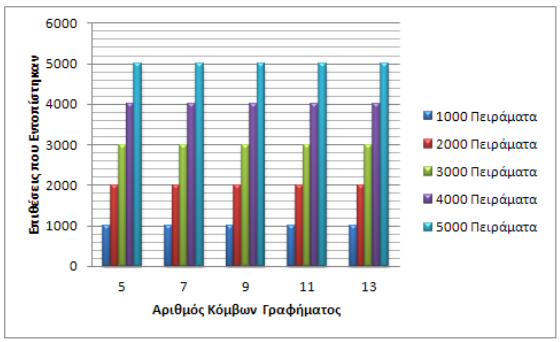
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.25: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.26: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

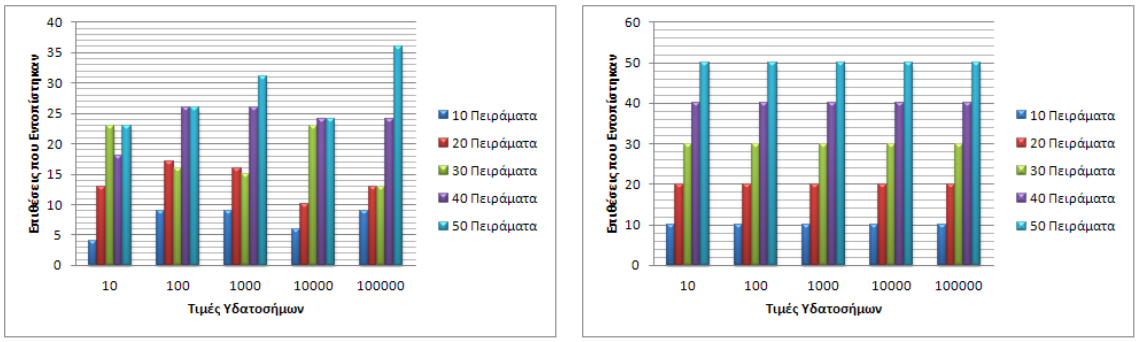
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	4	9	9	6	9
20	13	17	16	10	13
30	23	16	15	23	13
40	18	26	26	24	24
50	23	26	31	24	36

Πίνακας 5.39: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.40: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.27: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

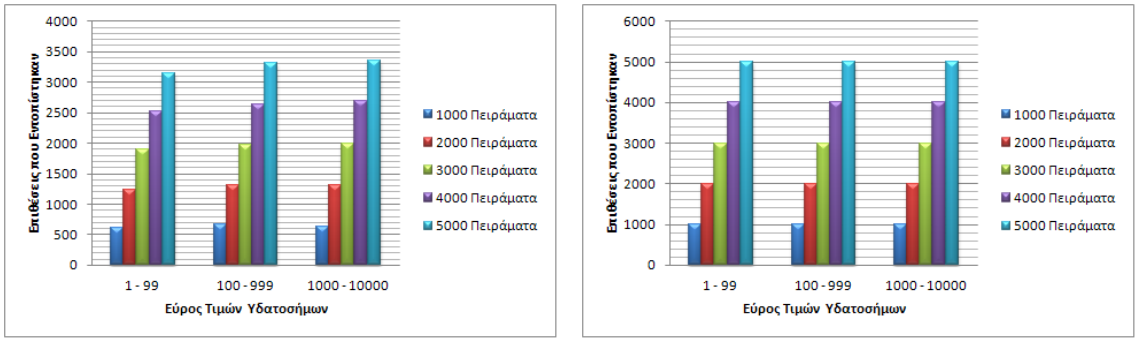
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	619	670	637
2000	1243	1305	1306
3000	1893	1964	1992
4000	2517	2628	2699
5000	3149	3313	3359

Πίνακας 5.41: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.42: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής κατεύθυνσης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.28: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Όπως παρατηρούμε, τα αποτελέσματα και των τριών αυτών πειραμάτων είναι καλύτερα στην περίπτωση που χρησιμοποιούμε τον δεύτερο έλεγχο παρά τον πρώτο. Συγκεκριμένα, με τον δεύτερο έλεγχο έχουμε 100% επιτυχία εύρεσης μιας επίθεσης αλλαγής κατεύθυνσης ακμής καθώς αυτό συνεπάγεται και την αύξηση του βαθμού των εξερχόμενων ακμών ενός κόμβου κατά ένα, γεγονός που συμβάλλει στο να γίνει ο βαθμός του μεγαλύτερος από

δύο (υπενθυμίζουμε ότι όλοι οι κόμβοι του σώματος του γραφήματος έχουν βαθμό ακριβώς δύο). Επίσης, αν ένας εκ των δύο κόμβων που συνδέει η ακμή είναι ο s τότε αυτός θα αποκτήσει μηδενικό βαθμό που πάλι θα οδηγήσει στο συμπέρασμα ότι έχει πραγματοποιηθεί κάποια επίθεση. Τέλος, αν ο κόμβος που συνδέει η ακμή είναι ο t τότε μετά την επίθεση ο τελευταίος θα αποκτήσει βαθμό ένα, οπότε θα αντιληφθούμε και πάλι την επίθεση (ο t έχει βαθμό εξερχόμενων ακμών μηδέν). Άρα, τα γραφήματα είναι ανθεκτικά σε τέτοιου είδους επιθέσεις.

5.3.2 Επίθεση Προσθήκης Ακμής

Σε αυτή την υποενότητα θα παραθέσουμε τα αποτελέσματα που παρήχθησαν από την έρευνα που διεξήχθη μέσω της εφαρμογής επιθέσεων προσθήκης ακμής και στα τρία προαναφερθέντα πειράματα. Οι ακόλουθοι πίνακες δίδουν αυτά τα αποτελέσματα και για τα τρία αυτά πειράματα με αντίστοιχο τρόπο.

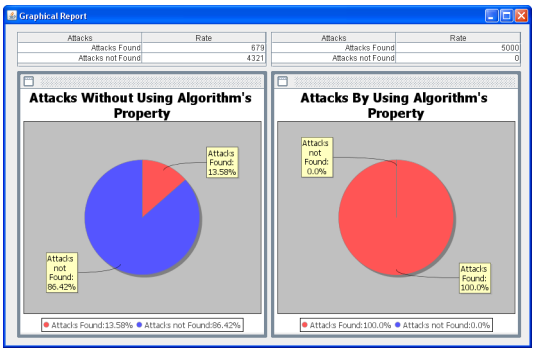
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	0	100	143	198	238
2000	0	232	346	407	470
3000	0	346	534	622	652
4000	0	468	644	811	947
5000	0	575	834	1002	1222

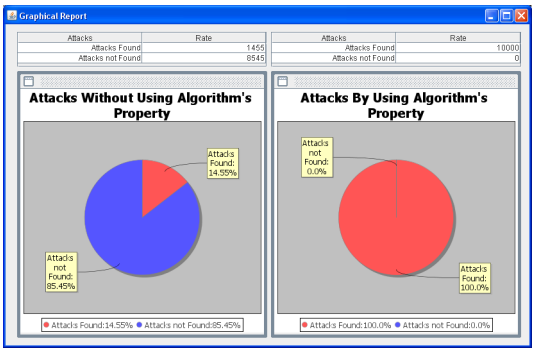
Πίνακας 5.43: Αλγόριθμος `CN`, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

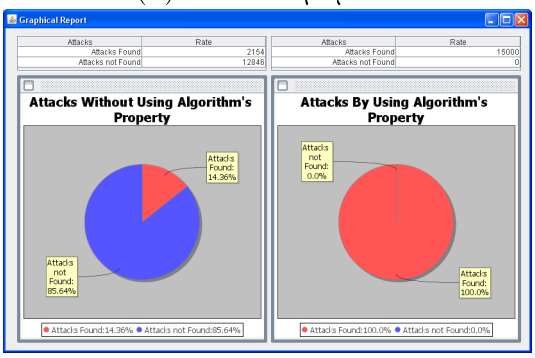
Πίνακας 5.44: Αλγόριθμος `CN`, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



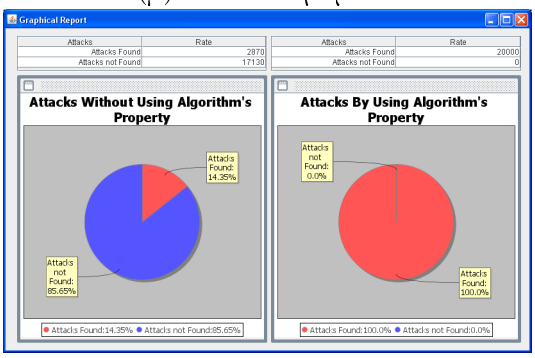
(α) 1000 Πειράματα



(β) 2000 Πειράματα



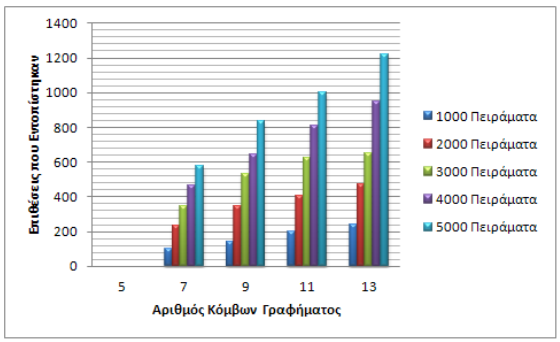
(γ) 3000 Πειράματα



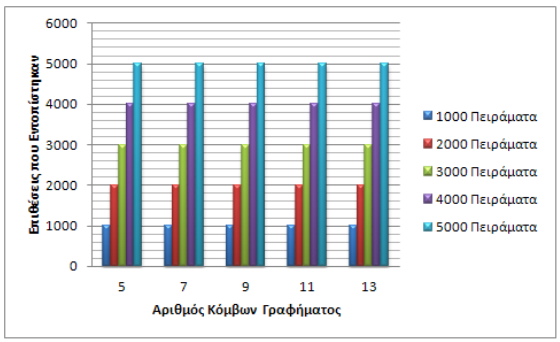
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.29: Γραφική αναπαράσταση αποτελεσμάτων συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.30: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

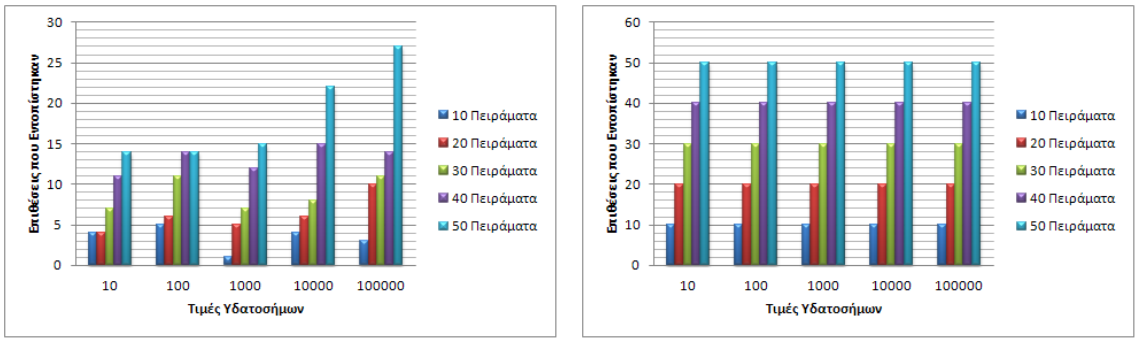
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	4	5	1	4	3
20	4	6	5	6	10
30	7	11	7	8	11
40	11	14	12	15	14
50	14	14	15	22	27

Πίνακας 5.45: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.46: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.31: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

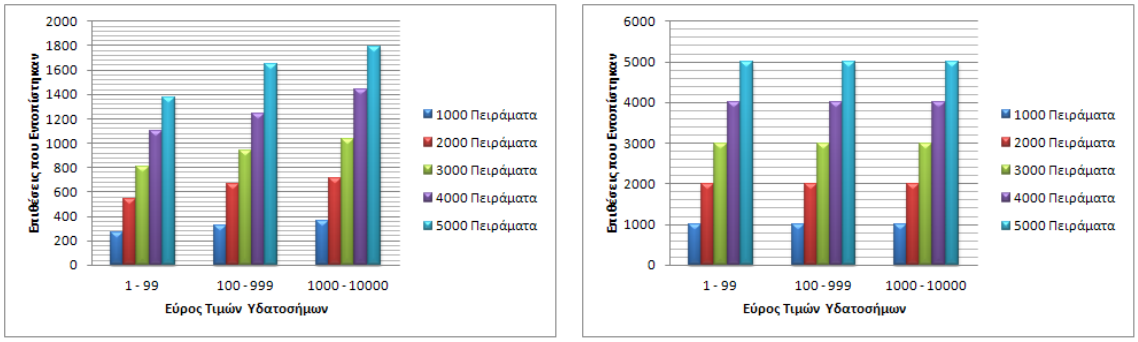
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	267	326	364
2000	545	661	708
3000	806	940	1035
4000	1104	1237	1438
5000	1378	1648	1789

Πίνακας 5.47: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.48: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.32: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Εάν μελετήσουμε τα παραπάνω αποτελέσματα και των τριών πειραμάτων θα διαπιστώσουμε ότι και πάλι όπως στην επίθεση αλλαγής κατεύθυνσης ακμής, έτσι και εδώ ο δεύτερος έλεγχος επιφέρει 100% επιτυχία στο να αντιληφθούμε ότι έχει γίνει μια επίθεση στο γράφημα. Αυτό οφείλεται στην ιδιότητα που αναφέραμε στην αρχή της Ενότητας 5.3 για τον βαθμό των εξερχόμενων ακμών των κόμβων του σώματος και των κόμβων s και t. Χωρίς

αυτή την ιδιότητα παρατηρούμε ότι αντιλαμβανόμαστε χαμηλό ποσοστό επιθέσεων προσθήκης ακμής καθώς το γράφημα σε πολλες περιπτώσεις, παρά την εισαγωγή μιας ακμής, παραμένει αναγώγιμο. Έτσι, μπορούμε να ισχυριστούμε ότι οι δομές που δημιουργούνται από τον Αλγόριθμο CN είναι πολύ ανθεκτικές σε μονές επιθέσεις προσθήκης ακμών.

5.3.3 Επίθεση Διαγραφής Ακμής

Όμοια με τις υποενότητες 5.3.1 και 5.3.2, έτσι και σε αυτό το σημείο θα παρατηρήσουμε ότι με βάση τον δεύτερο έλεγχο γίνονται αντιληπτές όλες οι επιθέσεις διαγραφής ακμής. Όπως εύκολα αντιλαμβάνεται κάποιος αυτό οφείλεται για άλλη μια φορά στο ότι ο βαθμός των εξερχόμενων ακμών που έχει ένας κόμβος είναι αυστηρά καθορισμένος. Ως απόδειξη των όσων αναφέρθηκαν σε αυτή την υποενότητα ακολουθούν τα εξής πειράματα:

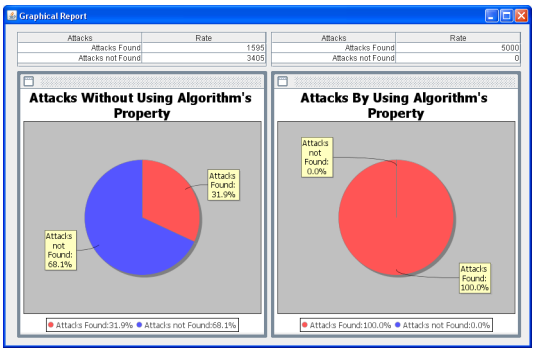
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	155	267	345	415	413
2000	264	494	740	800	864
3000	423	803	1080	1196	1291
4000	589	1121	1364	1565	1763
5000	753	1373	1766	2096	2160

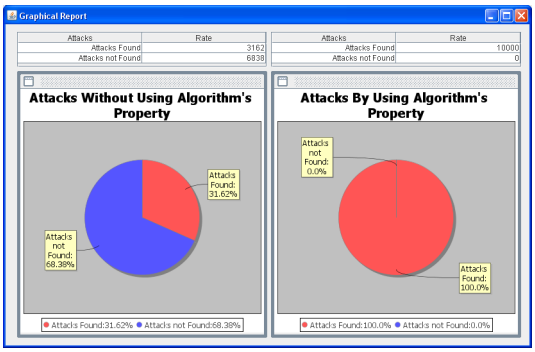
Πίνακας 5.49: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

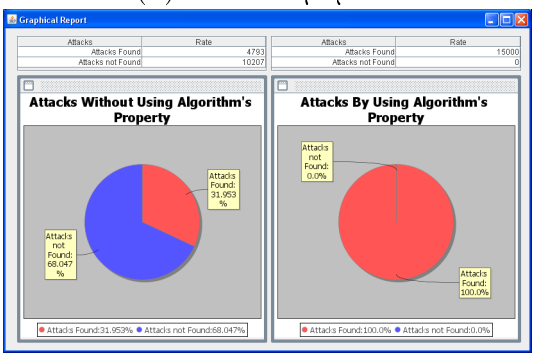
Πίνακας 5.50: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



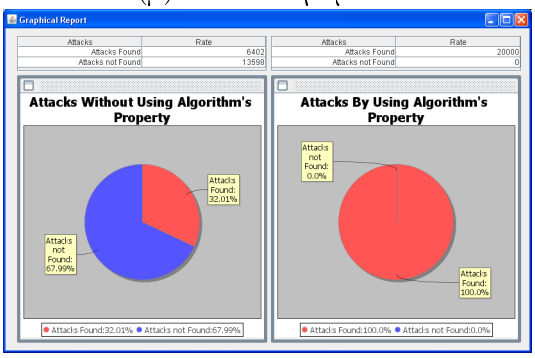
(α) 1000 Πειράματα



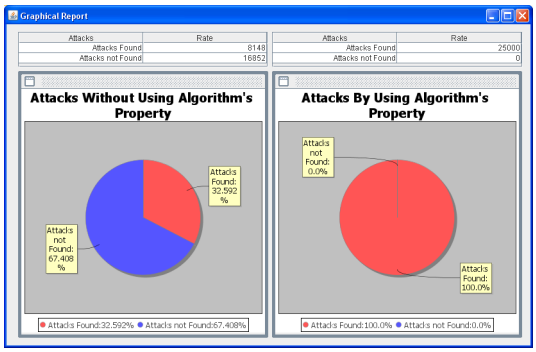
(β) 2000 Πειράματα



(γ) 3000 Πειράματα

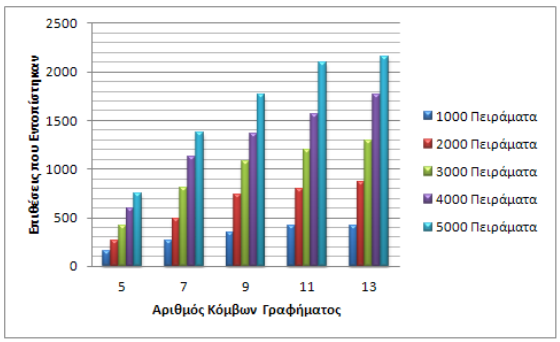


(δ) 4000 Πειράματα

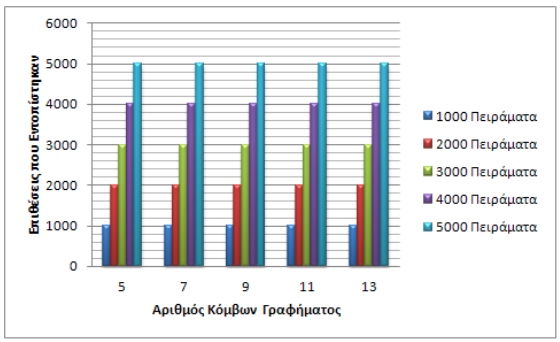


(ε) 5000 Πειράματα

Σχήμα 5.33: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.34: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

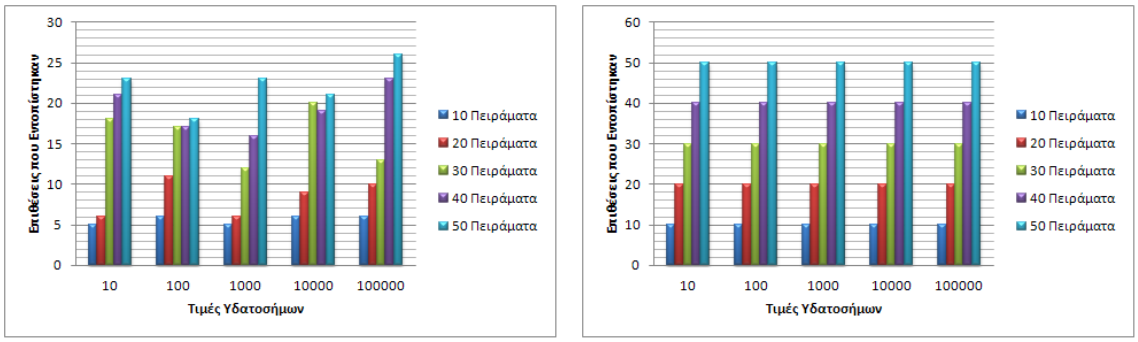
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	5	6	5	6	6
20	6	11	6	9	10
30	18	17	12	20	13
40	21	17	16	19	23
50	23	18	23	21	26

Πίνακας 5.51: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.52: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.35: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

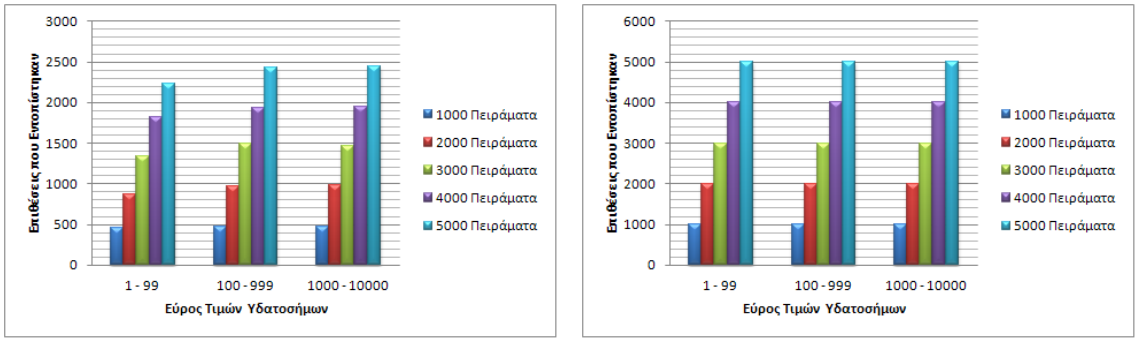
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	459	472	470
2000	864	964	980
3000	1340	1487	1461
4000	1817	1939	1953
5000	2231	2433	2444

Πίνακας 5.53: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.54: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.36: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.3.4 Επίθεση Αλλαγής Προορισμού Ακμής

Στην παρούσα υποενότητα παραθέτουμε τα αποτελέσματα και των τριών πειραμάτων σε περιπτώσεις αλλαγής του προορισμού μιας ακμής.

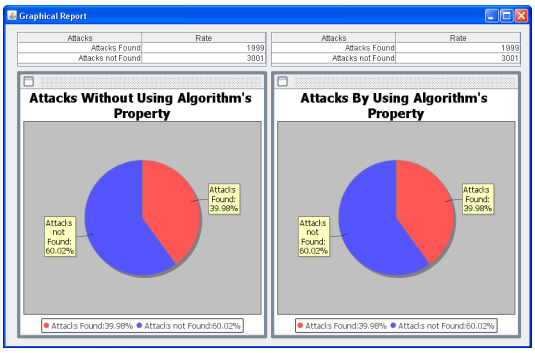
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	308	353	389	459	490
2000	625	643	773	910	943
3000	973	1047	1182	1328	1477
4000	1253	1413	1575	1762	1911
5000	1564	1709	1940	2179	2466

Πίνακας 5.55: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

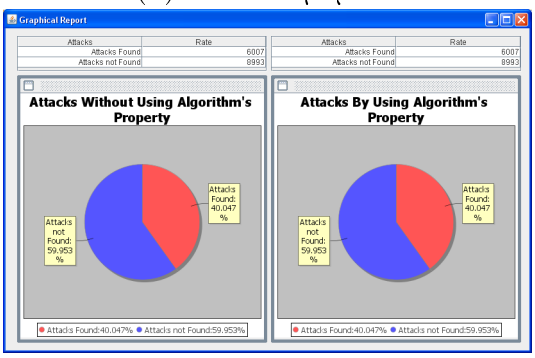
Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	308	353	389	459	490
2000	625	643	773	910	943
3000	973	1047	1182	1328	1477
4000	1253	1413	1575	1762	1911
5000	1564	1709	1940	2179	2466

Πίνακας 5.56: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο

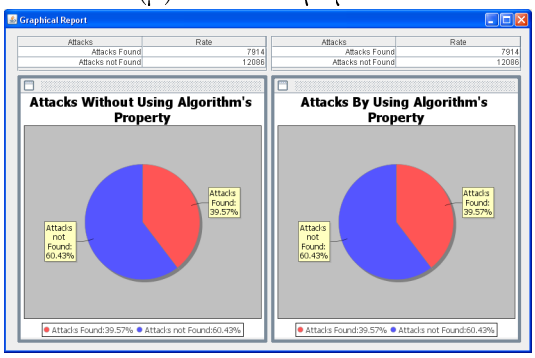


(α) 1000 Πειράματα

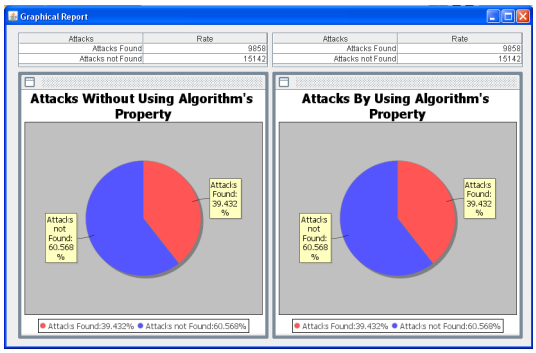
(β) 2000 Πειράματα



(γ) 3000 Πειράματα

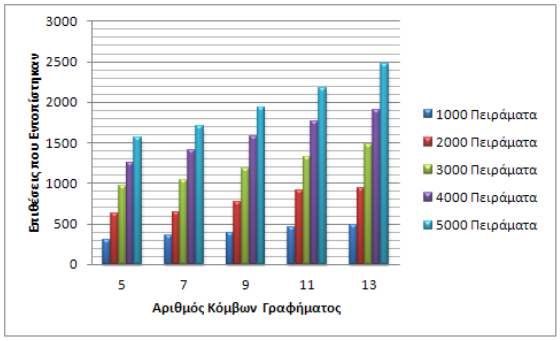


(δ) 4000 Πειράματα

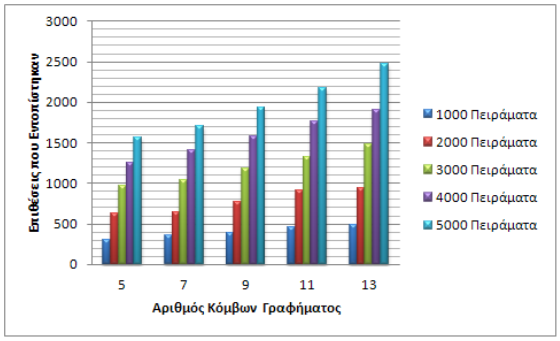


(ε) 5000 Πειράματα

Σχήμα 5.37: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.38: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

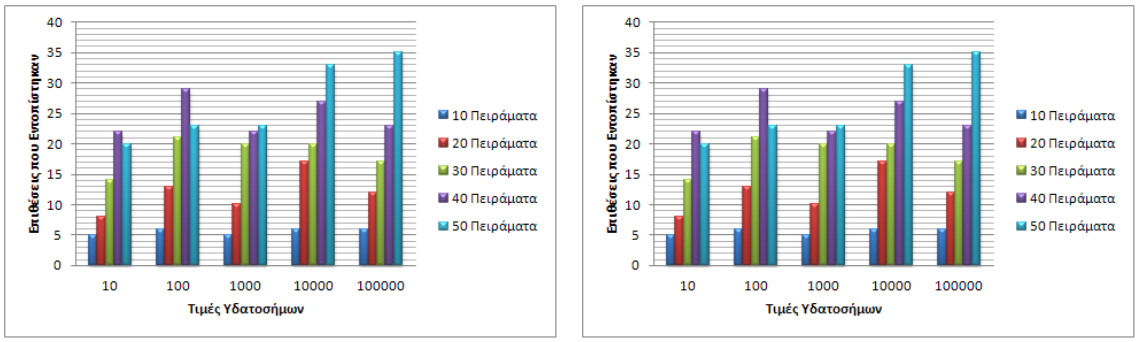
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	5	6	5	6	6
20	8	13	10	17	12
30	14	21	20	20	17
40	22	29	22	27	23
50	20	23	23	33	35

Πίνακας 5.57: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	5	6	5	6	6
20	8	13	10	17	12
30	14	21	20	20	17
40	22	29	22	27	23
50	20	23	23	33	35

Πίνακας 5.58: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.39: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

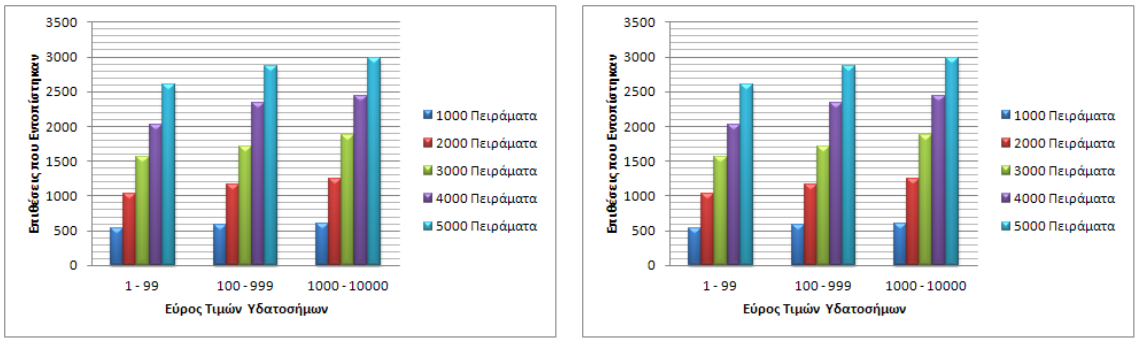
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	527	580	599
2000	1034	1160	1247
3000	1561	1703	1872
4000	2022	2341	2436
5000	2599	2866	2984

Πίνακας 5.59: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	527	580	599
2000	1034	1160	1247
3000	1561	1703	1872
4000	2022	2341	2436
5000	2599	2866	2984

Πίνακας 5.60: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης αλλαγής προορισμού ακμής σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.40: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Όπως παρατηρούμε στα παραπάνω πειράματα τόσο ο πρώτος έλεγχος, όσο και ο δεύτερος προσφέρουν ακριβώς το ίδιο στον εντοπισμό μιας επίθεσης αλλαγής προορισμού μιας ακμής καθώς δεν αλλάζει κανένας βαθμός εξερχόμενων ακμών κάποιου κόμβου. Επομένως, και οι δύο έλεγχοι πραγματοποιούν ακριβώς την ίδια διαδικασία και επιστρέφουν ακριβώς τα ίδια αποτελέσματα, χωρίς φυσικά κανένας να μην εντοπίζει το 100% των επιθέσεων αλλά πολύ χαμηλά, σε σχέση με τις επιθέσεις που έχουν εξεταστεί μέχρι στιγμής, ποσοστά.

5.3.5 Επίθεση Προσθήκης Κόμβου

Μέχρι αυτό το σημείο έχουμε παραθέσει τα πειραματικά αποτελέσματα που αφορούν επιθέσεις που σχετίζονται με ακμές του γραφήματος. Από αυτό το σημείο και μέχρι το τέλος αυτού του κεφαλαίου θα παρουσιάσουμε τα αποτελέσματα των πειραμάτων για επιθέσεις που αφορούν τους κόμβους του γραφήματος. Συγκεκριμένα, σε αυτή την υποενότητα θα προβάλλουμε τα αποτελέσματα που προκύπτουν από την προσθήκη ενός κόμβου στο αρχικό γράφημα. Τα αποτελέσματα και των τριών πειραμάτων είναι τα ακόλουθα:

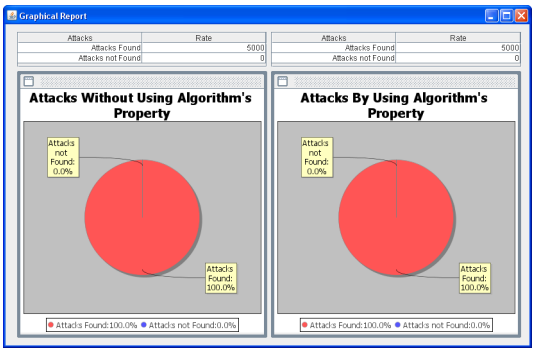
■ **Πείραμα 1:**

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

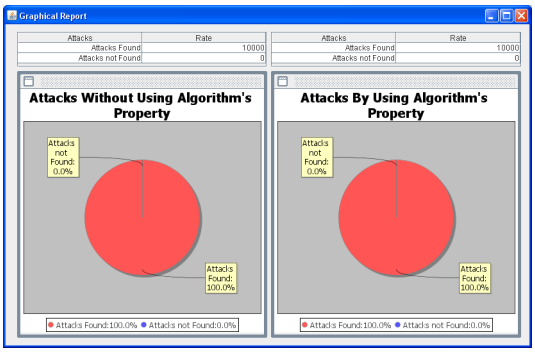
Πίνακας 5.61: Αλγόριθμος `CN`, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Πλήθος Κόμβων / Αριθμός Πειραμάτων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

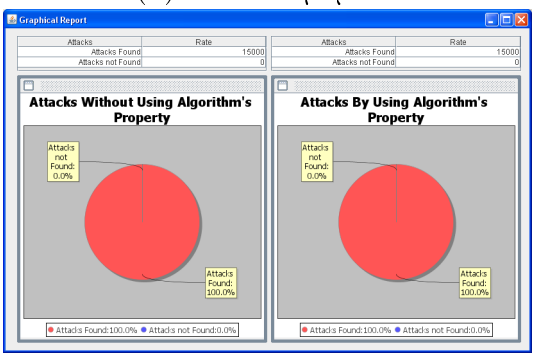
Πίνακας 5.62: Αλγόριθμος `CN`, Πείραμα 1 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



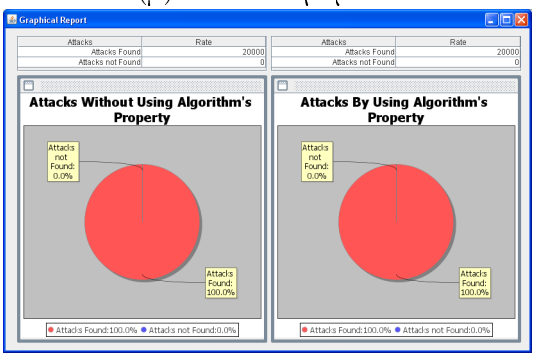
(α) 1000 Πειράματα



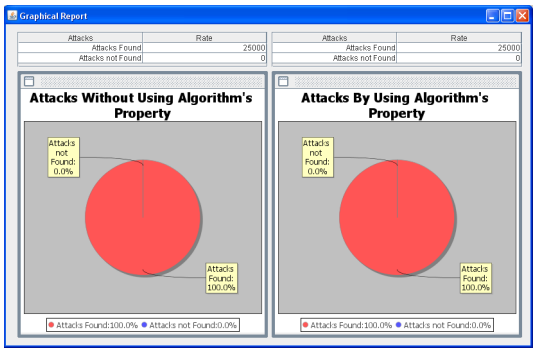
(β) 2000 Πειράματα



(γ) 3000 Πειράματα

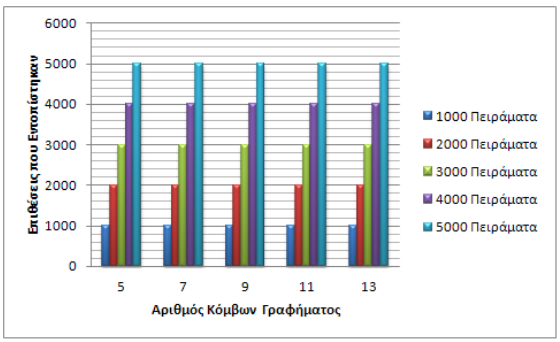


(δ) 4000 Πειράματα

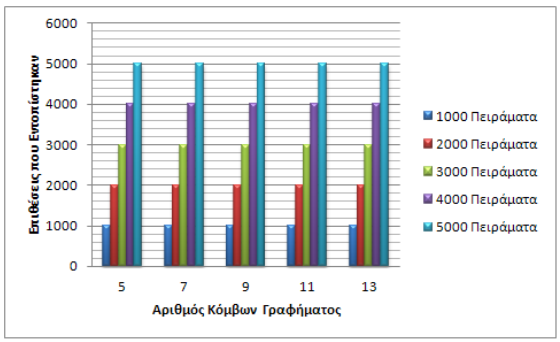


(ε) 5000 Πειράματα

Σχήμα 5.41: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.42: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Εάν παρατηρήσουμε τα παραπάνω αποτελέσματα και αυτά που είχαν προκύψει στην αντίστοιχη επίθεση στον Αλγόριθμο CKCT θα αντιληφθούμε ότι υπάρχει 100% επιτυχία στον εντοπισμό επιθέσεων καθώς όπως έχουμε αναφέρει κάθε φορά εκτελούμε μια μοναδική επίθεση. Επομένως, μια προσθήκη κόμβου θα γίνεται αντιληπτή μόνο και μόνο από τον πρώτο έλεγχο. Τα αποτελέσματα των δύο άλλων πειραμάτων είναι τα εξής:

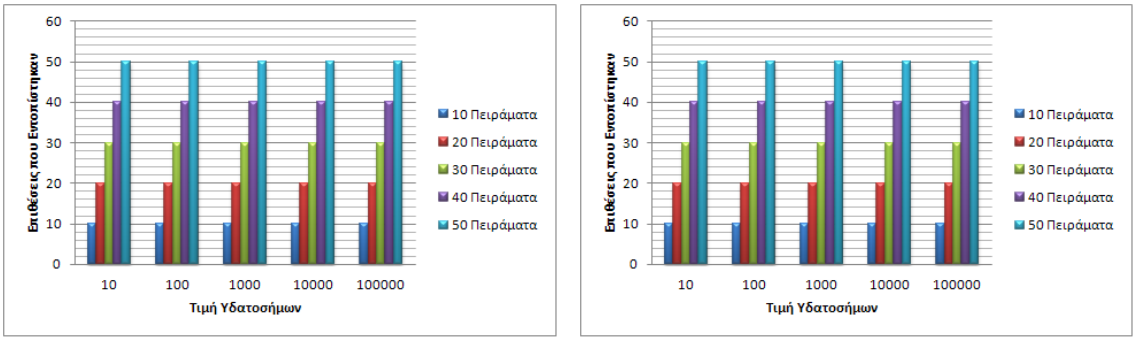
■ **Πείραμα 2:**

Τιμές Υδατοσήμων / Αριθμός Πειραμάτων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.63: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Τιμές Υδατοσήμων / Αριθμός Πειραμάτων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.64: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.43: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

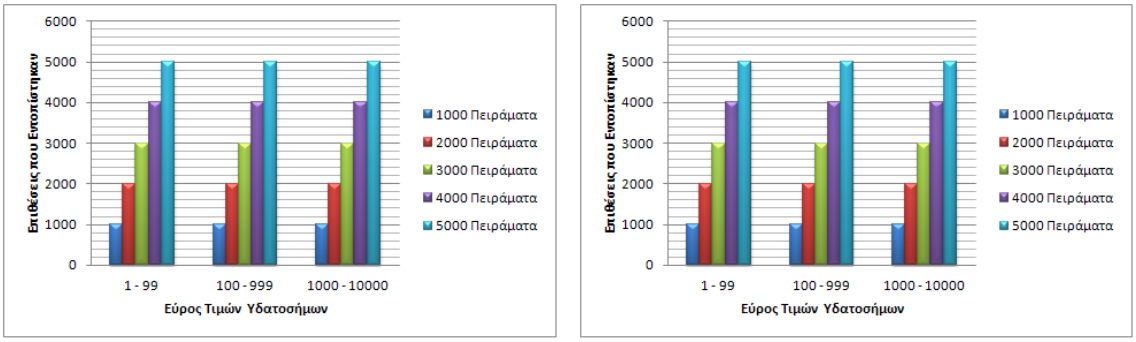
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.65: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.66: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης προσθήκης κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.44: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

5.3.6 Επίθεση Διαγραφής Κόμβου

Η τελευταία επίθεση με την οποία θα ασχοληθούμε σε αυτό το κεφάλαιο είναι αυτή της διαγραφής ενός κόμβου του αρχικού γραφήματος. Στη συνέχεια προβάλλουμε τα αποτελέσματα των πειραμάτων καθώς επίσης και μια γραφική απεικόνιση αυτών.

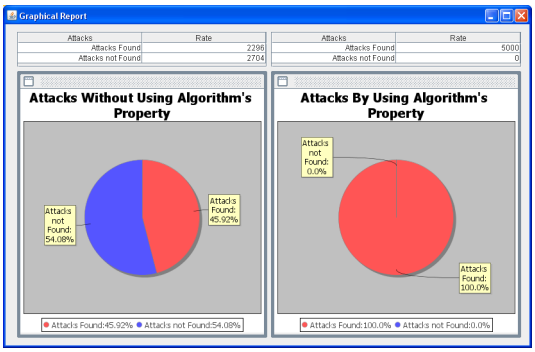
■ **Πείραμα 1:**

Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	400	301	433	549	613
2000	807	572	850	1094	1219
3000	1158	871	1277	1608	1878
4000	1597	1173	1735	2161	2443
5000	2021	1412	2238	2782	3001

Πίνακας 5.67: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

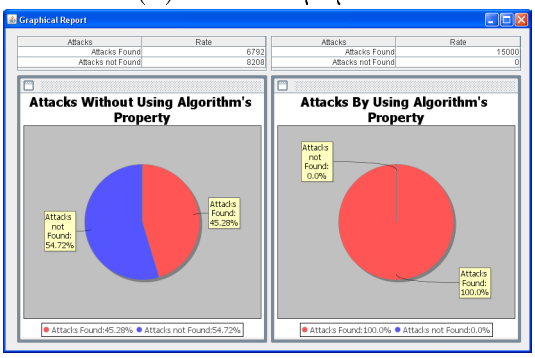
Αριθμός Πειραμάτων \ Πλήθος Κόμβων	5	7	9	11	13
1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000
3000	3000	3000	3000	3000	3000
4000	4000	4000	4000	4000	4000
5000	5000	5000	5000	5000	5000

Πίνακας 5.68: Αλγόριθμος CN, Πείραμα 1 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο

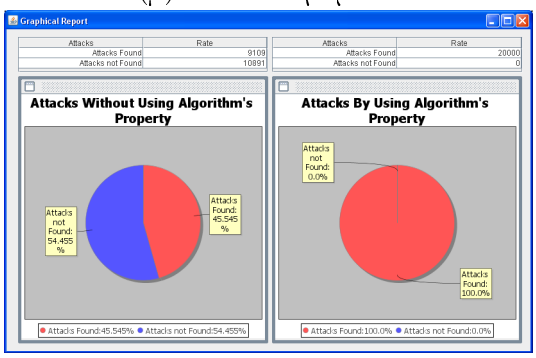


(α) 1000 Πειράματα

(β) 2000 Πειράματα



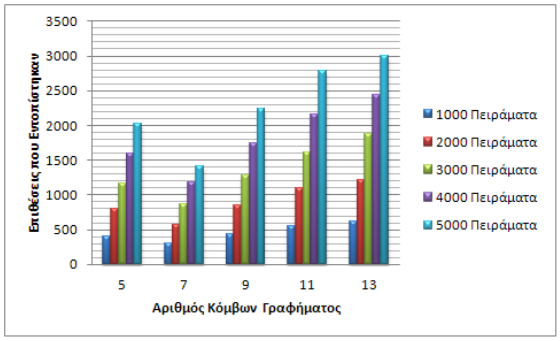
(γ) 3000 Πειράματα



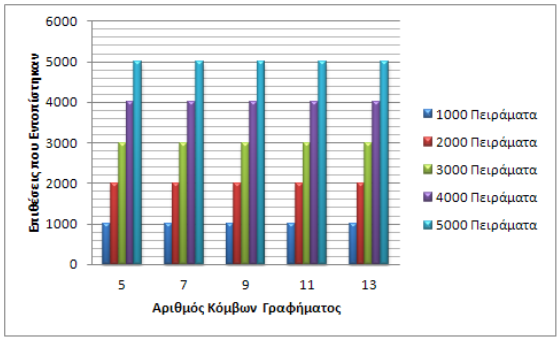
(δ) 4000 Πειράματα

(ε) 5000 Πειράματα

Σχήμα 5.45: Γραφική αναπαράσταση αποτελεσμάτων Συστήματος *WaterGraph*



(a)



(b)

Σχήμα 5.46: Αλγόριθμος `CN`, Πείραμα 1 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

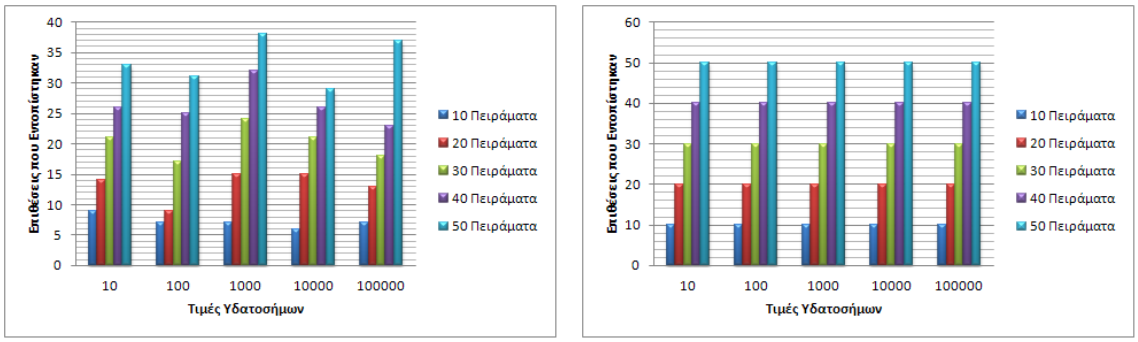
■ **Πείραμα 2:**

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	9	7	7	6	7
20	14	9	15	15	13
30	21	17	24	21	18
40	26	25	32	26	23
50	33	31	38	29	37

Πίνακας 5.69: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

Αριθμός Πειραμάτων \ Τιμές Υδατοσήμων	10	100	1000	10000	100000
10	10	10	10	10	10
20	20	20	20	20	20
30	30	30	30	30	30
40	40	40	40	40	40
50	50	50	50	50	50

Πίνακας 5.70: Αλγόριθμος CN, Πείραμα 2 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.47: Αλγόριθμος CN, Πείραμα 2 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

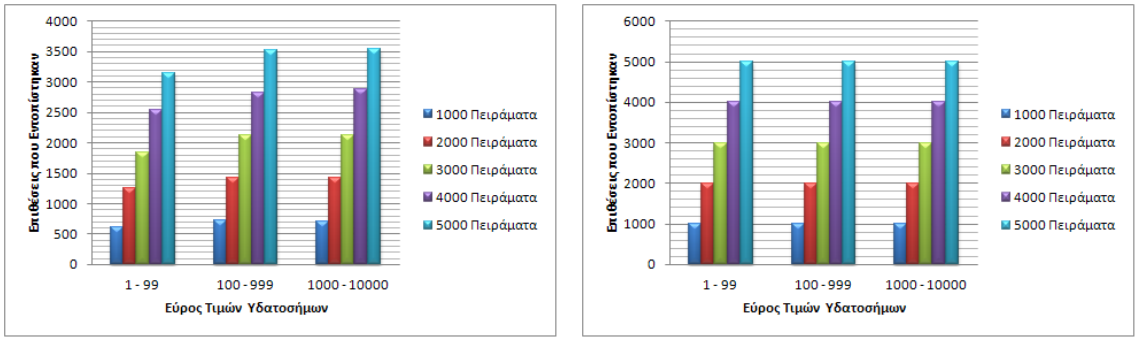
■ **Πείραμα 3:**

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	619	717	707
2000	1259	1422	1433
3000	1848	2120	2124
4000	2533	2825	2876
5000	3136	3525	3546

Πίνακας 5.71: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον πρώτο έλεγχο

Εύρος Τιμών Υδατοσήμων / Αριθμός Πειραμάτων	1 - 99	100 - 999	1000 - 10000
1000	1000	1000	1000
2000	2000	2000	2000
3000	3000	3000	3000
4000	4000	4000	4000
5000	5000	5000	5000

Πίνακας 5.72: Αλγόριθμος CN, Πείραμα 3 - Αποτελέσματα επίθεσης διαγραφής κόμβου σύμφωνα με τον δεύτερο έλεγχο



(a) (b)

Σχήμα 5.48: Αλγόριθμος CN, Πείραμα 3 - επιθέσεις που βρέθηκαν με βάση (a) τον πρώτο έλεγχο και (b) τον δεύτερο έλεγχο

Εάν εξετάσουμε τα παραπάνω πειραματικά αποτελέσματα είναι δυνατόν να εξαχθεί το συμπέρασμα ότι ο δεύτερος έλεγχος έχει απόλυτη επιτυχία στο να αντιλαμβάνεται επιθέσεις διαγραφής κόμβων του αναγώγιμου μεταθετικού γραφήματος σε σύγκριση με τον πρώτο. Αυτό λαμβάνει χώρα εξαιτίας του ότι μέσω της διαγραφής ενός κόμβου αυτομάτως διαγράφονται και οι ακμές που εισέρχονται σε αυτόν με αποτέλεσμα να μειώνεται το πλήθος των

εξερχόμενων ακμών κάποιων κόμβων. Έτσι, χάριν της ιδιότητας των βαθμών των κόμβων που αναφέραμε στην αρχή της Ενότητας 5.3 γίνεται πολύ εύκολος ο εντοπισμός τέτοιου είδους επιθέσεων.

Κεφαλαιο 6

Συμπερασματα

Στόχος της παρούσας διατριβής ήταν η μελέτη της ανθεκτικότητας των δομών που δημιουργούνται από τους αλγορίθμους που αναφέραμε σε επιθέσεις αλλαγής κατεύθυνσης ακμής, προσθήκης/διαγραφής ακμής, αλλαγής προορισμού ακμής και προσθήκης/διαγραφής κόμβου. Για την μελέτη αυτών των δομών αναπτύξαμε το Σύστημα *WaterGraph* μέσω του οποίου διεξήγαμε τρία διαφορετικά πειράματα για κάθε επίθεση. Το πρώτο δημιουργούσε γραφήματα με βάση το μέγεθος τους, το δεύτερο με βάση συγκεκριμένες τιμές υδατοσήμων και το τρίτο επέλεγε τιμές τυχαία σε ένα εύρος τιμών. Πρέπει να υπενθυμήσουμε ότι σε κάθε γράφημα διεξάγεται μια μοναδική επίθεση. Σε αυτή την ενότητα παρουσιάζουμε τα γενικά συμπεράσματα που εξήχθησαν από τα αποτελέσματα που παρήχθησαν από αυτή τη μελέτη.

Στην περίπτωση επιθέσεων αλλαγής κατεύθυνσης ακμής ο Αλγόριθμος `CN`, κάνοντας χρήση της ιδιότητας των βαθμών των εξερχόμενων ακμών που έχει, εντοπίζει το 100% των επιθέσεων και στα τρία πειράματα σε αντίθεση με τον `CKCT` ο οποίος φθάνει σε αρκετά υψηλό επίπεδο όμως δεν τις εντοπίζει όλες. Επίσης, αυτο που παρατηρούμε είναι ότι ακόμα και χωρίς τη χρήση της ιδιότητας που έχουν οι κόμβοι, ο αλγόριθμος `CN` είναι πιο ανθεκτικός από τον `CKCT` ο οποίος από την άλλη έχει πολύ χαμηλά ποσοστά εντοπισμού επιθέσεων. Επιπλέον, σε επιθέσεις προσθήκης ή διαγραφής ακμής ο Αλγόριθμος `CN` εντοπίζει και πάλι το 100% εξαιτίας της ιδιότητας των κόμβων, ενώ ο `CKCT` είναι αρκετά πιο χαμηλά σε ποσοστά. Όμως αυτό που παρατηρούμε είναι ότι στην γενική περίπτωση όπου δεν χρησιμοποιείται η ιδιότητα των κόμβων ο τελευταίος αλγόριθμος είναι αισθητά καλύτερος από τον αρχικό. Στην περίπτωση που εξετάζουμε την επίθεση αλλαγής προορισμού μιας ακμής και οι δύο αλγόριθμοι έχουν σχεδόν τα ίδια ποσοστά εντοπισμού επιθέσεων είτε κάνοντας χρήση της ιδιότητας των κόμβων, είτε όχι. Άριστα αποτελέσματα προκύπτουν επίσης και για τους δύο αλγορίθμους στην περίπτωση προσθήκης κόμβου καθώς γίνεται πολύ εύκολα αντιληπτή εφόσον δημιουργούνται δύο συνεκτικές συνιστώσες. Τέλος, σε περιπτώσεις διαγραφης κόμβων ο `CN` εντοπίζει το 100% των επιθέσεων χάριν της ιδιότητας των εξερχόμενων βαθμών των κόμβων, αφού διαγραφή ενός κόμβου σημαίνει αυτομάτως και διαγραφή των ακμών που φεύγουν και που πηγαίνουν σε αυτόν, ενώ ο `CKCT` δεν φτάνει αυτό το ποσοστό καθώς

ο βαθμός των εξερχόμενων ακμών δεν είναι αυστηρά ορισμένος.

Έτσι, από τα παραπάνω μπορούμε να εξάγουμε το συμπέρασμα ότι ο Αλγόριθμος CN είναι πιο ανθεκτικός σε επιθέσεις σε σχέση με τον CKCT καθώς έχει αυστηρά ορισμένο τον βαθμό των εξερχόμενων ακμών των κόμβων, σε αντίθεση με τον τελευταίο.

Σε μελλοντική έρευνα, θα μπορούσε να ελεγχθεί πειραματικά το κατά πόσο ανθεκτικές είναι οι δομές που δημιουργούν αυτοί οι δύο αλγόριθμοι σε περιπτώσεις που δεν εφαρμόζεται μια μοναδική επίθεση στο εκάστοτε γράφημα, αλλά περισσότερες, καθώς επίσης και η σύγκριση των αναγώγιμων μεταθετικών γραφημάτων που δημιουργούνται από αυτούς με άλλες δόμές.

Βιβλιογραφια

[1] G. Arboit, "A Method for Watermarking Java Programs via Opaque Predicates", International Conference on Electronic Commerce Research (ICECR-5), 2002.

[2] M. Chroni and S.D. Nikolopoulos, "Encoding watermark integers as self-inverting permutations", *11th Int'l Conference on Computer Systems and Technologies (CompSysTech'10)*, ACM ICPS 471, pp. 125–130, 2010.

[3] M. Chroni and S.D. Nikolopoulos, "Efficient Encoding Watermark Numbers as Reducible Permutation Graphs", Department of Computer Science, University of Ioannina, Greece, Technical report TR-2011-3, 2011.

[4] M. Chroni and S.D. Nikolopoulos, "Encoding watermark numbers as cographs using self-inverting permutations", *12th Int'l Conference on Computer Systems and Technologies (CompSysTech'11)*, ACM ICPS 578 (Best Paper Award), pp. 142–148, 2011.

[5] M. Chroni and S.D. Nikolopoulos, "Multiple Encoding of a Watermark Number into Reducible Permutation Graphs using Cotrees", *13th Int'l Conference on Computer Systems and Technologies (CompSysTech'12)*, ACM ICPS (to appear), 2012.

[6] M. Chroni and S.D. Nikolopoulos, "An Efficient Graph Codec System for Software Watermarking", *36th Int'l Conference on Computers, Software, and Applications (COMPSAC'12); Workshop STPSA'12*, IEEE Proceedings, 2012.

[7] M. Chroni and S.D. Nikolopoulos, "An embedding graph-based model for software watermarking", *8th Int'l Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP'12)*, IEEE Proceedings, 2012.

[8] C. Collberg, E. Carter, S. Debray, A. Huntwork, J. Kececioglu, C. Linn, and M. Stepp, "Dynamic path-based software watermarking", 2004.

[9] C. Collberg, S. Kobourov, E. Carter, and C. Thomborson, "Error-Correcting Graphs for Software Watermarking", Proceedings of the 29th Workshop on Graph Theoretic Concepts in Computer Science, pages 156-167, 2003.

[10] C. Collberg and J. Nagra, "Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection", Addison-Wesley Professional, 1st ed., 2009.

[11] C. Collberg and C. Thomborson, "Software watermarking: Models and dynamic embeddings", Proceedings of Symposium on Principles of Programming Languages, pp. 311-324, 1999.

[12] C. Collberg, C. Thomborson, and D. Low, "On the limits of software watermarking", Technical Report #164, Department of Computer Science, The University of Auckland, 1998.

[13] P. Cousot and R. Cousot, "An abstract interpretation-based framework for software watermarking", Principles of Programming Languages 2003, POPL'03, pp. 311-324, 2003.

[14] R. Davidson and N. Myhrvold,"Method and system for generating and auditing a signature for a computer program", US Patent, vol. 5.559.884, 1996.

[15] D. Grover, "The Protection of Computer Software - Its Technology and Applications", 2nd ed., Cambridge University Press, 1997.

[16] S. Moskowitz and M. Cooperman, "Method for stega-cipher protection of computer code", US Patent, vol. 5.745.569, 1994.

[17] G. Myles and C. Collberg, "Software Watermarking Through Register Allocation: Implementation, Analysis, and Attacks", ICISC 2003, LNCS 2971, pp. 274ð293, 2004.

[18] G. Myles and H. Jin, "Self-validating Branch-Based Software Wtermarking", IH 2005, LNCS3727, pp. 342-356, 2005.

[19] G. Qu and M. Potkonjak, "Hiding signatures in graph coloring solutions", Information Hiding Workshop, 1999.

[20] G. Qu, J. Wong, and M. Potkonjak, "Optimization-intensive watermarking techniques for decision problems", Design Automation Conference, pp. 33-36, 1999.

[21] P. Samson, "Apparatus and method for serializing and validating copies of computer software", US Patent, vol. 5.287.408, 1994.

[22] J.P. Stern, G. Hachez, F. Koeune, and J.J. Quisquater, "Robust object watermarking: Application to code", Information Hiding Workshop, 1999.

[23] H. Tamada, M. Nakamura, A. Monden, and K. Matsumoto, "Design and evaluation of birthmarks for detecting theft of java programs", Proc. IASTED International Conference on Software Engineering (IASTED SE2004), pp. 569-575, Feb 2004.

[24] R. Venkatesan, V. Vazirani, and S. Sinha, "A graph theoretic approach to software watermarking", Information Hiding Workshop, 2001.

[25] W. Zhu and C. Thomborson, "Extraction in software watermarking", MM-Sec, pp. 175-181, 2006.

Βιογραφικο

Ο Νικόλαος Γ. Βρεττός γεννήθηκε το έτος 1988 και μεγάλωσε στο Μεγανήσι Λευκάδος. Το έτος 2002 εγκαταστάθηκε στα Ιωάννινα, όπου και αποφοίτησε από το 2ο Ενιαίο Λύκειο και το 2006 εισήχθη στο Τμήμα Πληροφορικής των Θετικών Επιστημών του Πανεπιστημίου Ιωαννίνων. Το έτος 2010 ολοκλήρωσε την εν λόγω σχολή και την ίδια χρονιά εισήχθη στο Πρόγραμμα Μεταπτυχιακών Σπουδών του ίδιου τμήματος.