ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Επιτάχυνση της εύρεσης της κυκλικής αναπαράστασης Συνόλων Μερικής Διάταξης «POSET» με χρήση GPU.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Νεοκλής Γ. Χατζηγεωργίου

Επιβλέπων : Νεκτάριος Κοζύρης
Αναπληρωτής Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2011

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Επιτάχυνση της εύρεσης της κυκλικής αναπαράστασης Συνόλων Μερικής Διάταξης «POSET» με χρήση GPU.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Νεοκλής Γ. Χατζηγεωργίου

Επιβλέπων : Νεκτάριος Κοζύρης
Αναπληρωτής Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 28η Ιουλίου 2011

....

Αθήνα, Ιούλιος 2011

.....

Νεοκλής Γ. Χατζηγεωργίου

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Νεοκλής Γ. Χατζηγεωργίου, 2011
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Στη διπλωματική μας μελετούμε την εικασία Sydney, Sydney και Urrutia περί κυκλικής αναπαράστασης των συνόλων μερικής διάταξης. Πιο συγκεκριμένα, ακολουθούμε την μεθοδολογία Bayon, Lygeros και Sereni, οι οποίοι απέδειξαν ότι όλα τα σύνολα μερικής διάταξης τάξης 10, έχουν κυκλική αναπαράσταση. Το αποτέλεσμα αυτό επιτεύχθηκε με γενετικό αλγόριθμο. Στην εργασία μας παραλληλίσαμε τον αλγόριθμό τους μέσω χρήσης καρτών γραφικών GPGPU με το προγραμματιστικό μοντέλο CUDA με στόχο να λειτουργεί με τις κάρτες TESLA με αρχιτεκτονική FERMI. Η υλοποίησή μας εφαρμόστηκε στα σύνολα μερικής διάταξης τάξης 11 και βρίσκεται σε εξέλιξη όσον αφορά στην υπολογιστική επίλυση των κυκλικών αναπαραστάσεων.

Λέξεις Κλειδιά

Εικασία SSU, Κυκλική Διάταξη, Σύνολα Μερικής Διάταξης, Poset, Εύρεση Poset διάστασης 1 και 2, Εύρεση μη συνεκτικών Poset, Κατηγοριοποίηση Poset, Επιταχυντές γραφικών, GPU, CUDA.

Abstract

In our work, we study Sidney, Sidney and Urrutia Conjecture on circle representation of partially ordered set. More specifically, we follow the methodology of Bayon, Lygeros and Sereni, who proved that all the posets of order 10, have circle representation. This result was achieved by genetic algorithm. In our work, we parallelized the algorithm through the use of graphics cards with GPGPU CUDA programming model designed to work with the cards TESLA architecture FERMI. The implementation was applied to posets of order 11 and is ongoing in terms of computational resolution of circle representations.

Keywords

Conjecture SSU, Circle Order, Partially Ordered Set, Poset, Check Poset dimension 1 or 2, Classification of Posets, Graphic Accelerator, GPU, CUDA.

Ευχαριστίες

Κατ' αρχάς, ευχαριστώ πολύ τον επιβλέποντα καθηγητή κ. Νεκτάριο Κοζύρη για τη συμπαράστασή του και τη βοήθειά του κατά τη διάρκεια της διπλωματικής μου εργασίας. Επίσης, οφείλω ένα μεγάλο ευχαριστώ στον κ. Βασίλειο Καρακάση για την πολύτιμη βοήθεια και τη μεγάλη συνεισφορά του στην εργασία, καθώς, και σε όλους τους ανθρώπους από το Computer Science Laboratory (cslab).

Επίσης, θα ήθελα να ευχαριστήσω ιδιαίτερα τους γονείς μου Ελένη Ηλιάδου Χατζηγεωργίου και Γεώργιο Χατζηαντρέα Χατζηγεωργίου που τόσα χρόνια μου διάβαζαν και ηχογραφούσαν τα κείμενα των μαθημάτων.

Θα ήθελα να ευχαριστήσω τον θείο μου Μιχάλη Πατατάκο που με ενέπνευσε με τις τεχνικές του ικανότητες.

Θα ήθελα να ευχαριστήσω τους τεχνικούς του Ραδιοφωνικού Ιδρύματος Κύπρου, τον Άγγελο Δικωμίτη, Μιχαλάκη Αντρέου, Ρούλη Ζαχαριάδη, για την πρώτη καθοδήγηση στον τομέα της ηλεκτρονικής επιστήμης.

Θα ήθελα να ευχαριστήσω τους τεχνικούς της Μετεωρολογικής Υπηρεσίας Κύπρου, τον Αντρέα Στυλιανίδη, Γεώργιο Χριστοδούλου, Γιαννάκη Ορφανίδη.

Θα ήθελα να ευχαριστήσω τον Παύλο Παυλίδη γιατί με καθοδήγησε στον τομέα των ηλεκτρονικών υπολογιστών και του προγραμματισμού.

Θα ήθελα να ευχαριστήσω το Νίκο Λυγερό για την καθοδήγηση του στα άδυτα των μαθηματικών και στον προγραμματισμό Maple.

Θα ήθελα να ευχαριστήσω τους φίλους μου Πόλυ Γεωργίου, Χαρίδημο Σπύρου, Κυριάκο Τούμπα και Παναγιώτη Χατζηχριστοδούλου για την πρακτική τους συμμετοχή στην αποσφαλμάτωση του προγράμματος στα Windows.

Τέλος θα ήθελα να ευχαριστήσω ειδικά τους ανθρώπους που με βοήθησαν στην υπολογιστική εφαρμογή του προγράμματος μέσω των υπολογιστών τους και του χρόνου που μου παραχώρησαν. Αυτοί είναι οι Άρτεμις Σκουρίδου Brannigan, Ελένη Ηλιάδου Χατζηγεωργίου, Kenneth Brannigan, Πάολα Βαγιωνή, Πόλυς Γεωργίου, Λητώ Καλογιάννη, Χρυσάνθη Καπράγκου, Βασίλειος Καρακάσης, Δημήτρης Κάτσιος, Κώστας Κάτσιος, Νεκτάριος Κοζύρης, Πέτρος Κουτσόλαμπρος, Κορνήλιος Κούρτης, Κυριάκος Κυπριανού, Χάρης Κυπριανού, Χρίστος Κωνσταντινίδης, Δέσποινα Κωνσταντίνου, Νίκος Λυγερός, Βαγγέλης Μπενάτος, Θεόδουλος Παπαθεοδούλου, Μιχάλης Πατατάκος, Όλγα Ραπτοπούλου, Χαρά Ρεμούνδου, Κείρα Σταμπολίδου, Μαρία Σταμπουλίδου, Αντρέας Στυλιανίδης, Μαρία Τζιβελέκη, Βασιλική Τσατσαμπά, Εύη Χαριτίδου, Κωνσταντίνα Χαρμπή, Χριστόδουλος Χατζηγεωργίου, Γιώργος Χατζηγεωργίου, Αντρέας Χατζηγεωργίου, Παναγιώτης Χατζηχριστοδούλου, Σπύρος Χατζηχριστοδούλου, Χριστόφορος Χριστοφή.

Θα ήθελα να ευχαριστήσω την NVidia διότι ονόμασε κάρτες της με το όνομα του Никола Тесла (Nikola Tesla).

Αυτή η εργασία αφιερώνεται στην οικογένειά μου και στην μνήμη του Νικολα Τεσλα (Nikola Tesla)

Περιεχόμενα

Κατάλογοι

Κατάλογος Σχημάτων

Κατάλογος Διαγραμμάτων

Κατάλογος Πινάκων

Κατάλογος Διαγραμμάτων Ροής

Κεφάλαιο 1

Εισαγωγή

Στην εργασία αυτή μελετούνται τα σύνολα μερικής διάταξης, γνωστά και ως Posets. Παρουσιάζεται η εικασία Sidney – Sidney – Urrutia, γνωστή ως SSU. Στην εικασία αυτή τίθεται το ερώτημα, σε ποιες περιπτώσεις ένα Poset έχει κυκλική αναπαράσταση. Η κυκλική αναπαράσταση αναφέρεται στη δυνατότητα σχεδίασης ενός πίνακα μεταβάσεων Poset σε κύκλους. Μελετώνται, επίσης, μερικές ειδικές κατηγορίες των Posets. Επιπρόσθετα, παρουσιάζεται η εκθετική αύξηση του πληθυσμού των Posets σε σχέση με τον αριθμό των κόμβων που χρησιμοποιούνται για να δημιουργήσουν τα Posets. Γίνεται για πρώτη φορά προσπάθεια παραλληλισμού μέσω OPENMP και Cuda. Γενικότερα, στο κεφάλαιο αυτό παρουσιάζεται η συνεισφορά των Posets στην επιστήμη. Στη συνέχεια, γίνεται περιγραφή της δομής του κειμένου.

Συνεισφορά

1. Ιστορικό πλαίσιο των εφαρμογών των συνόλων μερικής διάταξης στην πληροφορική.

Πολλά προβλήματα αποφάσεων είναι πολύπλοκα και μεγάλης τάξης. Η πολυπλοκότητα, το κόστος και οι φυσικές πιέσεις οδηγούν στην αποκέντρωση. Μόνο που ένας έλεγχος εντελώς επιμερισμένος και παράλληλος είναι εξαιρετικά δύσκολος. Ένα σημαντικό υπόβαθρο όλου αυτού του θέματος είναι η ροή πληροφορίας. Έτσι τίθεται το εξής πρόβλημα: Ποιά είναι τα κατάλληλα εργαλεία και η κατάλληλη γλώσσα για τη διαχείρηση αυτής της ροής πληροφοριών. Η θεωρία συνόλων μερικής διάταξης είναι η απάντηση και για αυτό το λόγο αυτές οι μαθηματικές οντότητες είναι χρήσιμες στα δίκτυα υπολογιστών. Τα σύνολα μερικής διάταξης είναι υποκατηγορία των προσανατολισμένων γράφων. Αυτό που διαφοροποιεί ένα Poset από ένα προσανατολισμένο γράφο είναι ότι το Poset δεν έχει κύκλωμα, δηλαδή κύκλο με ίδια κατεύθυνση στα βέλη, αλλά μπορεί να έχει κύκλο. Τα Posets έχουν ένα ευρύ φάσμα εφαρμογών. Χρησιμοποιούνται όπου έχουμε αντικείμενα ή άτομα που έχουν μια σχετική διάταξη μεταξύ τους, είτε αυτή είναι επικοινωνία είτε ιεραρχία. Πέραν του 80% των δημοσιεύσεων σε Posets γίνεται μέσω κονδυλίων από τον αμερικανικό στρατό. Εφαρμόζονται τόσο σε διοικητικά μοντέλα όσο, σε στρατηγικές πολέμου και τακτικές μάχης. Τα τελευταία χρόνια χρησιμοποιήθηκαν τα Posets για την επαλήθευση της απόδειξης του Καραθεοδωρή περί ανεξαρτησίας των αξιωμάτων της θεωρίας των **Somas.**

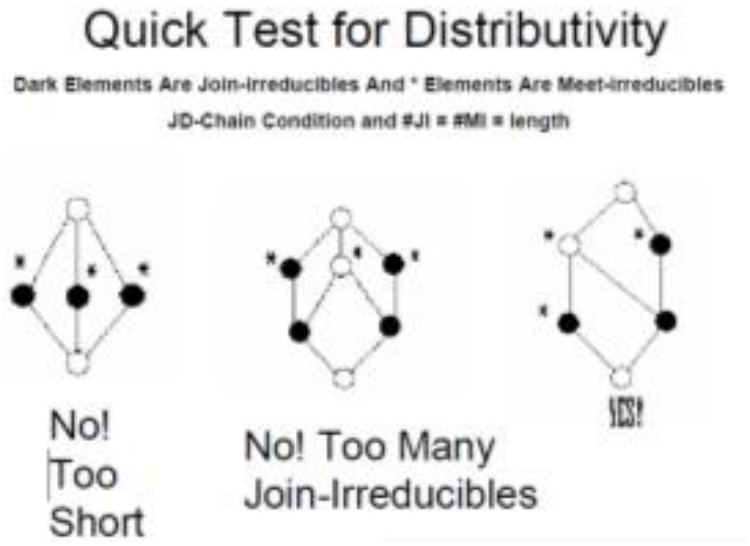
2. Ιστορικό πλαίσιο των κυκλικών αναπαραστάσεων.

Ο προβληματισμός των κυκλικών αναπαραστάσεων των συνόλων μερικής διάταξης προέρχεται από το χώρο της φυσικής. Στη θεωρία του Newton ο χώρος και ο χρόνος είναι ανεξάρτητες οντότητες. Ο χώρος είναι ευκλείδιος και η υπογραφή του είναι (+,+,+,+). Σε αυτή τη θεωρία η ταχύτητα του φωτός είναι άπειρη. Άρα, το αίτιο δεν υπάρχει ή μεταδίδεται άμεσα. Κατά συνέπεια το Poset (| .) δεν μπορεί να έχει κυκλική αναπαράσταση. Στη θεωρία της ειδικής σχετικότητας του Einstein ο χρόνος και ο χώρος αποτελούν το χωροχρόνο, ο οποίος έχει υπογραφή (+,+,+,-). Καθώς η ταχύτητα του φωτός είναι πεπερασμένη αυτός ο αιτιατός χωροχρόνος είναι πιο πολύπλοκος από τον προηγούμενο και η εύρεση του μικρότερου Poset που δεν έχει κυκλική αναπαράσταση σε αυτό το χωροχρόνο αποτελεί ένα δύσκολο πρόβλημα. Το 1989 οι Brightwell και Winkler απέδειξαν ότι υπάρχει σύνολο μερικής διάταξης τάξης 30 το οποίο δεν έχει σφαιρική αναπαράσταση. Στην ίδια μελέτη απέδειξαν ότι υπάρχει σύνολο μερικής διάταξης τάξης 14 το οποίο δεν έχει κυκλική αναπαράσταση. Με αυτή την καινοτομία άνοιξαν τον δρόμο της μελέτης των συνόλων μερικής διάταξης με κυκλική αναπαράσταση.

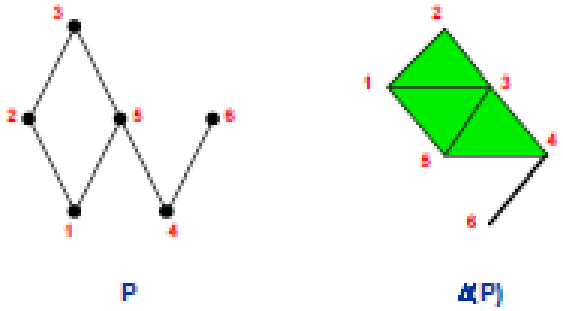
Η επαλήθευση της εικασίας είναι σημαντική γιατί αυτό αφορά κάθε σύστημα επικοινωνίας. Η εύρεση κυκλικά μη αναπαραστάσιμων Posets θα χρησιμεύσει στο να είναι στη γνώση των μηχανικών ότι τέτοιες διατάξεις μπορεί να εμφανίσουν

προβλήματα στην επικοινωνία, τόσο μη συγχρονισμού όσο και αδυναμίας λήψης της πληροφορίας. Η κυκλική αναπαράσταση έχει πρώτα από όλα μεγάλη εφαρμογή στα αυτοοργανούμενα δίκτυα (ad hoc). Παραδείγματος χάρη, ένας λόχος στρατιωτών επικοινωνεί με ασυρμάτους μεταξύ τους. Όταν βρεθούν σε μία μη κυκλικά αναπαραστάσιμη διάταξη θα εμφανιστεί αδυναμία σωστής επικοινωνίας, μιας και αποδεδειγμένα ξέρουμε ότι η ηλεκτρομαγνητική ακτινοβολία έχει πεπερασμένη ταχύτητα. Αντίστοιχα με τις ασύρματες επικοινωνίες γίνεται το ίδιο και στα ενσύρματα δίκτυα τόσο ηλετρονικά όσο και φωτονικά.

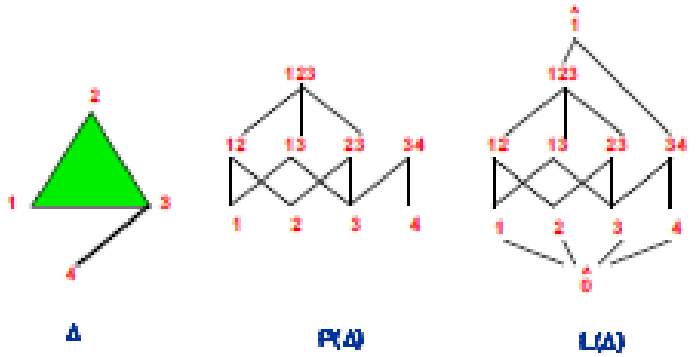
3. Παραδείγματα εφαρμογσής της θεωρίας των Posets.



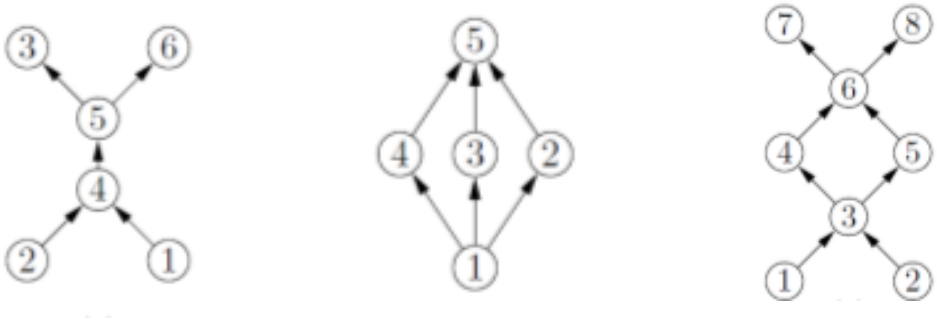
Σχήμα 1.1 : Επίλυση του γρήγορου ελέγχου του επιμερισμού.



Διάγραμμα 1.1 : Επίλυση του γρήγορου ελέγχου του επιμερισμού

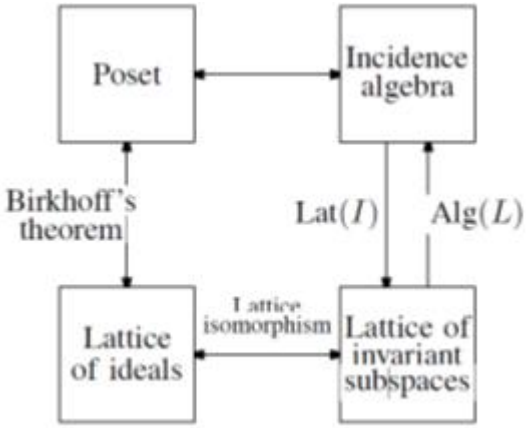


Διάγραμμα 1.2 : Poset έδρας και πλέγμα έδρας ενός συμπλέγματος.

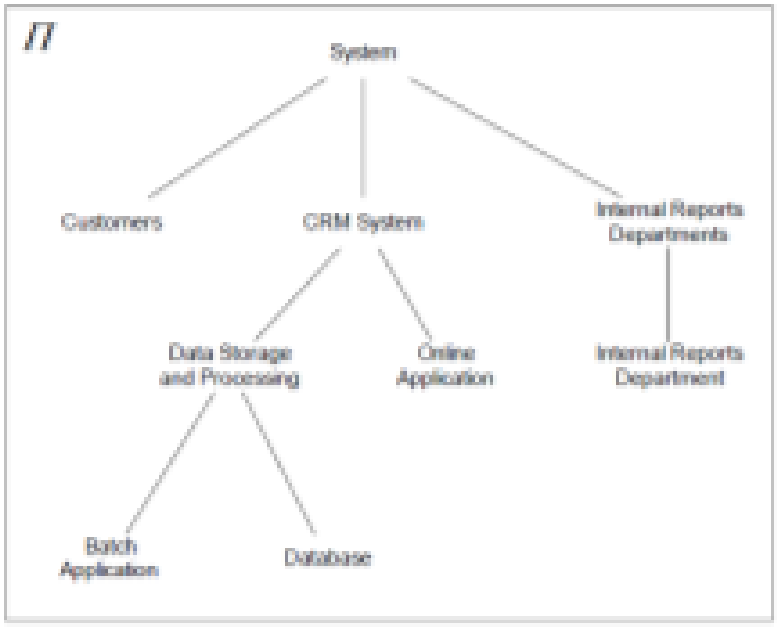


Πολλαπλή αλυσίδα Πλέγμα Poset

Διάγραμμα 1.3 : Παραδείγματα Επικοινωνίας



Σχήμα 1.2 : Αντιμεταθετικό διάγραμμα μεταξύ συνόλων μερικής διάταξης, πλεγμάτων και άλγεβρας.



Σχήμα 1.3 : Δυϊκό διάγραμμα Hasse για τους τύπους δικτύου.

Δομή του κειμένου

Η παρούσα διπλωματική εργασία χωρίζεται σε έξι κεφάλαια. Στο πρώτο κεφάλαιο γίνεται μια σύντομη εισαγωγή στα σύνολα μερικής διάταξης.

Στο δεύτερο κεφάλαιο καταγράφονται διάφορες έννοιες. Περιγράφονται εν συντομία οι γράφοι, οι ομάδες, τα δέντρα και τα Poset. Επίσης, γίνεται αναφορά στην πολυπλοκότητα και την αποκρισιμότητα. Περαιτέρω, δίνεται περιγραφικά η εικασία SSU, μερικές ειδικές κατηγορίες Posets, η απαρίθμηση και μια ιστορική αναδρομή.

Το τρίτο κεφάλαιο αφιερώνεται στον γενετικό αλγόριθμο και την περιγραφή του. Επιπλέον παρουσιάζονται τα διαγράμματα ροής.

Στο τέταρτο κεφάλαιο γίνεται μνεία στην GPU. Αναφέρεται ανάμεσα σε άλλα, η αρχιτεκτονική τους και η υλοποίηση σε αυτές. Ακόμη, περιγράφεται το προγραμματιστικό μοντέλο CUDA και η διαδικασία εύρεσης Poset σε GPU.

Το πέμπτο κεφάλαιο περιέχει τα πειραματικά αποτελέσματα της μελέτης της επιτάχυνσης σε πολυπύρηνους επεξεργαστές, σε κάρτες γραφικών (GPUs) και σε συνδυασμό πολυπύρηνου επεξεργαστή με GPU.

Στο έκτο κεφάλαιο παρουσιάζονται τα συμπεράσματα αυτής της εργασίας. Ακόμη, δίνονται κάποιες μελλοντικές κατευθύνσεις τόσο για την εξέλιξη του γενετικού αλγορίθμου, όσο και για την αύξηση της επιτάχυνσης με διάφορες τεχνικές.

Κεφάλαιο 2

Poset και εικασία SSU

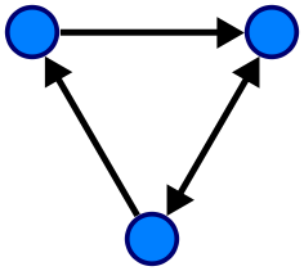
(Sidney – Sidney – Urrutia Conjecture)

Στο κεφάλαιο αυτό θα γίνει περιγραφή της εικασίας SSU (Sidney – Sidney – Urrutia Conjecture) . Η εικασία αυτή αναφέρεται στο αν όλα τα σύνολα μερικής διάταξης (Posets) είναι κυκλικά αναπαραστάσιμα, δηλαδή μπορούν να αναπαρασταθούν με κύκλους, μέχρι το P_{14}. Στην αρχή γίνεται μια εισαγωγή στις κυριότερες έννοιες των γράφων και στις ιδιότητές τους. Ακολούθως αναφέρονται οι ιδιότητες των ομάδων. Έπειτα εξετάζουμε τί είναι τα προσανατολισμένα δέντρα και πια σχέση έχουν με τα Posets. Αναφέρεται η σχέση που έχουν με την εικασία SSU και αν είναι άμεσα επιλύσιμα σύμφωνα με τα θεωρήματα που παρατίθενται. Γίνεται ο ορισμός των συνόλων μερικής διάταξης (Posets). Εξετάζουμε γενικότερα τα θεωρήματα για τα Posets και τα χαρακτηριστικά τους και πώς μπορούν να αναπαρασταθούν διαγραμματικά μέσω διαγραμμάτων Hasse. Αφού αναφέρουμε θέματα πολυπλοκότητας που δείχνουν ότι η εύρεση της διάστασης ενός Poset είναι δυσεπίλυτο πρόβλημα, προσπαθούμε να πλησιάσουμε την SSU από άλλη σκοπιά. Παρουσιάζουμε τα μαθηματικά που μπορούν να χρησιμοποιηθούν για την απόδειξή της με αριθμητικές μεθόδους και δείχνουμε ότι είναι αποκρίσιμα. Η εξέταση ειδικών κατηγοριών κρίθηκε χρήσιμη ώστε να δείξουμε τη μερική επίλυση της εικασίας. Η μόνη μέθοδος που παραμένει είναι η εξετάση που γίνεται για κάθε Poset, το οποίο εξετάζουμε ξεχωριστά για να δούμε αν ικανοποιεί την εικασία ή όχι. Γίνεται επιπλέον μία ιστορική αναφορά για τις προσπάθειες που έγιναν στο παρελθόν.

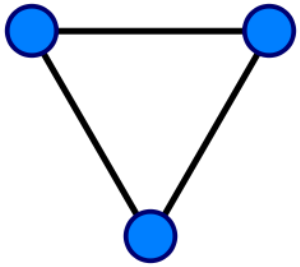
Γράφος - Graph

Στα μαθηματικά, ο γράφος είναι μία αφηρημένη αναπαράσταση ενός συνόλου αντικειμένων τα οποία συνδέονται μεταξύ τους. Τα διασυνδεδεμένα αντικείμενα τα οποία αναπαριστούνται από τα αφηρημένα μαθηματικά καλούνται κόμβοι ή κορυφές {V: vertex, vertices.} και οι σύνδεσμοι που ενώνουν τους κόμβους καλούνται ακμές {E: edge, edges}. Οι ακμές μπορεί να είναι κατευθυνόμενες «μονοδρομικές» (ασυμμετρικός) ή μη κατευθυνόμενες «αμφιδρομικές» (συμμετρικός).

Ορισμός : Ένας γράφος G = G(V,E) είναι ένα διατεταγμένο ζεύγος δύο ξένων μεταξύ τους συνόλων $V \neq \varnothing$ και E, όπου E είναι υποσύνολο του συνόλου των μη διατεταγμένων ζευγών του V.



(α) Κατευθυνόμενος Γράφος



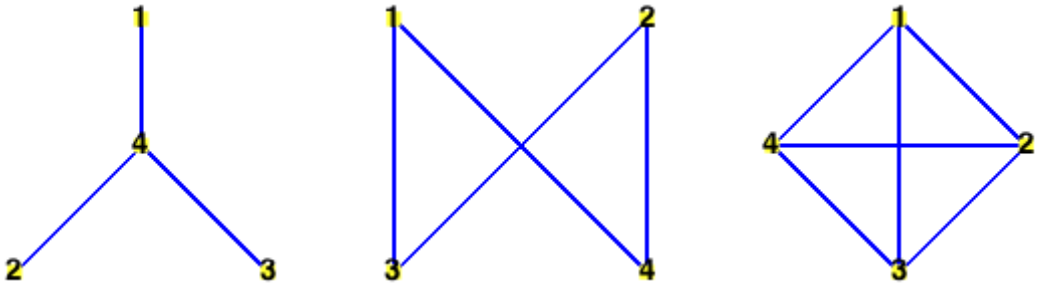
(β) Μη κατευθυνόμενος Γράφος

Σχήμα 2.1: Γράφοι

Οι γράφοι αποτελούν ένα ιδιαίτερα διαδεδομένο τύπο δομής δεδομένων στην επιστήμη υπολογιστών, και οι αλγόριθμοι για το χειρισμό τους είναι θεμελιώδους σημασίας για τον κλάδο. Υπάρχουν εκατοντάδες ενδιαφέροντα υπολογιστικά προβλήματα που ορίζονται μέσω γράφων. Ένα τέτοιο παράδειγμα είναι ότι ο χρωματισμός του γράφου είναι ίσος με την πολυπλοκότητα διάστασης 3 (dimension 3) που αποδείχτηκε από τον Μιχάλη Γιαννακάκη. Κύριος συμβολισμός ενός γράφου είναι ο εξής: G = (V,E). Υπάρχουν δύο καθιερωμένοι τρόποι αναπαράστασης ενός γράφου G = (V,E): α) υπό τη μορφή μιας συλλογής από καταλόγους γειτνίασης και β) υπό τη μορφή ενός πίνακα γειτνίασης. Αμφότερες οι αναπαραστάσεις μπορούν να εφαρμοστούν τόσο σε κατευθυνόμενους όσο και σε μη κατευθυνόμενους γράφους. Συνήθως προτιμάται η αναπαράσταση μέσω καταλόγων γειτνίασης, διότι προσφέρεται για πυκνή αναπαράσταση αραιών γράφων, δηλαδή, γράφων στους οποίους |E| είναι πολύ μικρότερο του $|V|^2$. Όταν όμως ο γράφος είναι πυκνός, δηλαδή όταν το |E| είναι συγκρίσιμο με το $|V|^2$ ή όταν είναι απαραίτητο να μπορούμε να αποφανθούμε γρήγορα εάν δύο κόμβοι συνδέονται μέσω ακμής, τότε πιθανόν η αναπαράσταση μέσω πίνακα γειτνίασης είναι προτιμότερη.

Η αναπαράσταση του γράφου μέσω πινάκων χωρίζεται σε δύο κατηγορίες: Τον πίνακα Γειτνίασης (Adjacency matrix) και τον πίνακα Πρόσπτωσης (Incidence matrix). Μπορεί φαινομενικά να ακούγονται το ίδιο, παρόλα αυτά έχουν μερικές

ριζικές διαφορές. Ο πίνακας γειτνίασης είναι συμμετρικός $A = \begin{matrix} 0ή1 & B \\ B^T & 0ή1 \end{matrix}$ και απευθύνεται σε μη προσανατολισμένους γράφους. Από την άλλη, ο πίνακας πρόσπτωσης αναφέρεται και σε γράφους οι οποίοι είναι κατευθυνόμενοι και συνήθως μη συμμετρικοί.

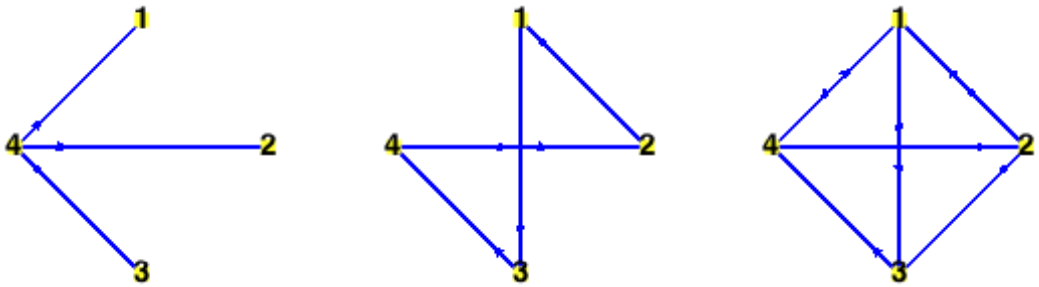


	1	2	3	4
1	0	0	0	1
2	0	0	0	1
3	0	0	0	1
4	1	1	1	0

	1	2	3	4
1	0	0	1	1
	0	0	1	1
3	1	1	0	0
4	1	1	0	0

	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

Σχήμα 2.2 *Παραδείγματα με πίνακες Γειτνίασης*



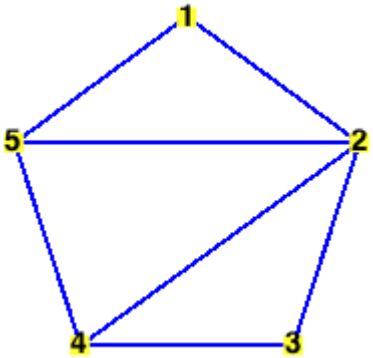
	1	2	3	4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	1	1	1	0

	1	2	3	4
1	1	1	0	0
2	0	0	0	1
3	1	0	1	1
4	0	1	0	1

	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	0	0	1
4	1	0	0	0

Σχήμα 2.3 : *Παραδείγματα με πίνακες Πρόσπτωσης*

Η αναπαράσταση μέσω καταλόγων γειτνίασης ενός γράφου G = (V,E) συνίσταται σε μία συστοιχία Adj από |V| καταλόγους, ένα για κάθε κόμβο του V. Για κάθε $u \in V$, ο κατάλογος γειτνίασης Adj[u] περιέχει όλους τους κόμβους υ για τους οποίους υπάρχει ακμή $u,v \in E$. Αναλυτικότερα, ο Adj[u] αποτελείται από όλους τους κόμβους που γειτνιάζουν με τον u στο G. Εάν ο γράφος G είναι κατευθυνόμενος, το άθροισμα των μηκών όλων των καταλόγων γειτνίασης ισούται με |E|, αφού οποιαδήποτε ακμή (u,υ) αναπαρίσταται μέσω της παρουσίας του υ στον κατάλογο Adj[u]. Εάν ο G είναι μη κατευθυνόμενος, το αντίστοιχο άθροισμα των μηκών είναι 2|E|, αφού εάν η (u,υ) είναι μια μη κατευθυνόμενη ακμή, τότε το u εμφανίζεται στον κατάλογο γειτνίασης του υ και αντιστρόφως. Τόσο για τους κατευθυνόμενους όσο και για τους μη κατευθυνόμενους γράφους, η αναπαράσταση μέσω καταλόγων γειτνίασης έχει την ευπρόσδεκτη ιδιότητα ότι η ποσότητα της απαιτούμενης μνήμης είναι Θ(V+E).



(α)
Σχηματική αναπαράσταση γράφου G

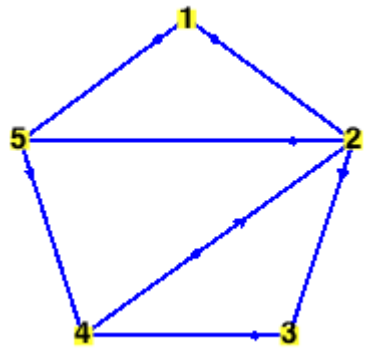
1 2 5 /
2 1 3 4 5 /
3 2 4 /
4 2 3 5 /
5 1 2 4 5 /

(β)
Αναπαράσταση του G μέσω καταλόγων γειτνίασης

	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	1

(γ)
Αναπαράσταση του G μέσω πίνακα γειτνίασης

Σχήμα 2.4 : Μη κατευθυνόμενος Γράφος



(α)
Σχηματική αναπαράσταση γράφου G

1 2 5 /
2 3 4 5 /
3 4 /
4 2 /
5 4 /

(β)
Αναπαράσταση του G μέσω καταλόγων γειτνίασης

	1	2	3	4	5
1	0	1	0	0	1
2	0	0	1	1	1
3	0	0	0	1	0
4	0	1	0	0	0
5	0	0	0	1	0

(γ)
Αναπαράσταση του G μέσω πίνακα γειτνίασης

Σχήμα 2.5 : Κατευθυνόμενος Γράφος

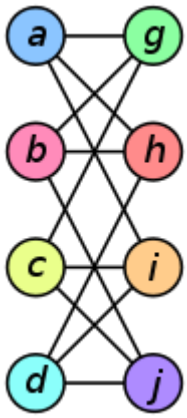
Στην αναπαράσταση μέσω πίνακα γειτνίασης ενός γράφου G = (V,E), υποθέτουμε ότι οι κόμβοι είναι αριθμημένοι από το 1 έως το |V| κατά αυθαίρετο τρόπο. Στην περίπτωση αυτή, η αναπαράσταση ενός γράφου G μέσω πίνακα γειτνίασης συνίσταται σε έναν $V \times V$ πίνακα A = (α_{ij}) με στοιχεία $\alpha_{ij} = \begin{matrix} 1, & εάν (i,j) \in E \\ 0, & διαφορετικά \end{matrix}$. Ο πίνακας γειτνίασης ενός γράφου απαιτεί μνήμη $\Theta(V^2)$, ανεξάρτητα από το πλήθος των ακμών του γράφου.

Αν και η αναπαράσταση μέσω καταλόγων γειτνίασης είναι ασυμπτωτικά τουλάχιστον εξίσου ταχεία με την αναπαράσταση μέσω πίνακα γειτνίασης, όταν οι γράφοι είναι αρκετά μικροί πιθανόν ο πίνακας γειτνίασης να είναι προτιμότερος λόγω της απλότητάς του. Επιπλέον, εάν ο γράφος είναι αβαρής, η αναπαράσταση μέσω πίνακα γειτνίασης προσφέρει ένα επιπρόσθετο πλεονέκτημα όσον αφορά τον αποθηκευτικό χώρο: αντί να χρησιμοποιείται μία λέξη μνήμης για κάθε στοιχείο του πίνακα αυτού, χρησιμοποιείται μόνο ένα δυαδικό ψηφίο (bit) ανά στοιχείο. Ακόμα μια βελτιστοποίηση που συνηθίζεται στους μη κατευθυνόμενους γράφους είναι η μετατροπή του πίνακα σε άνω ή κάτω τριγωνικό και αποθήκευση αυτού. Με αυτό επιτυγχάνεται μείωση του αποθηκευτικού χώρου.

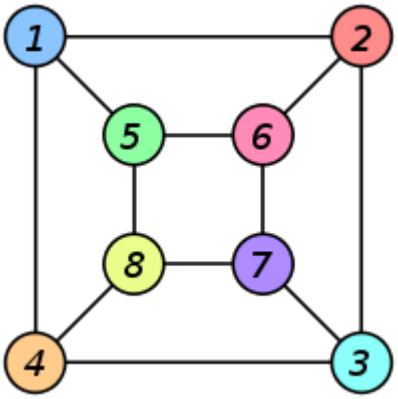
Ιδιότητες Γράφων

Ισομορφισμός:

Ορισμός: Δύο γράφοι G και H είναι ισομορφικοί εάν και μόνο εάν υπάρχει αντιστοιχία μεταξύ κορυφών και ακμών. $\boldsymbol{f: V\ G\ \rightarrow V(H)}$



Γράφος G.



Γράφος H.

$f\ a\ = 1$
$f\ b\ = 6$
$f\ c\ = 8$
$f\ d\ = 3$
$f\ g\ = 5$
$f\ h\ = 2$
$f\ i\ = 4$
$f\ j\ = 7$

Ένας ισομορφισμός μεταξύ των γράφων G και H.

Σχήμα 2.6 : Παράδειγμα ισομορφισμού Πηγή: Wikipedia

Σαν υπολογιστική πολυπλοκότητα το πρόβλημα του ισομορφισμού είναι μη πολυωνυμικού χρόνου, δηλαδή ανήκει στην κατηγορία των NP προβλημάτων.

Αυτομορφισμός:

Ορισμός: Ένας γράφος στα μαθηματικά είναι αυτομορφικός, εάν υπάρχει μορφή η οποία να είναι συμμετρική και ο γράφος να χαρτογραφείται στον εαυτό του, διατηρώντας παράλληλα τη σύνδεση κορυφή με ακμή.

Βαθμός Κόμβου:

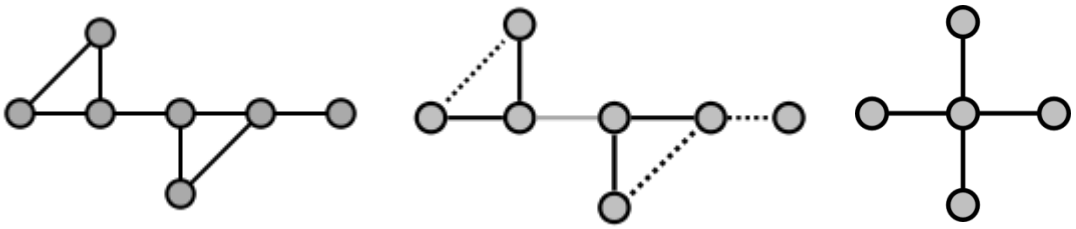
Ορισμός : Βαθμός ενός κόμβου ορίζεται ως το άθροισμα των εισερχόμενων και εξερχόμενων ακμών, όταν η τιμή των εξερχόμενων ακμών είναι ένα (+1) και των εισερχόμενων ακμών είναι μείον ένα (-1).

Ελάσσονας Γράφος - Minor Graph

Ορισμός : Ο γράφος H είναι ελάσσονας γράφος του γράφου G εάν με την εφαρμογή των επομένων κανόνων στο γράφο G καταλήγουμε στο γράφο H.

Κανόνες : Οι κανόνες για να εξετάσουμε αν ένας γράφος είναι ελάσσονας γράφος κάποιου άλλου γράφου είναι οι ακόλουθοι:

1. Μπορούμε να βγάλουμε μία κορυφή
2. Μπορούμε να βγάλουμε μία ακμή
3. Μπορούμε να συρρικνώσουμε μία ακμή



Σχήμα 2.7 : Σχηματικό διάγραμμα εφαρμογής κανόνων για έλεγχο εάν ο H είναι υπογράφος του G

Ιδιότητες Ομάδων

Έστω Α ένα σύνολο διάφορο του κενού και εφοδιασμένο με μία εσωτερική πράξη $\circ$, δηλαδή $\circ : A \times A \to A$
Τότε η δομή $(A,\circ)$ καλείται :

- ημιομάδα, αν η πράξη είναι προσεταιριστική , δηλαδή αν ισχύει
  $a \circ b \circ c = a \circ b \circ c \quad \forall a,b,c \in A$

- μονοειδές, αν είναι ημιομάδα και επιπλέον η πράξη έχει ουδέτερο στοιχείο, δηλαδή αν ισχύουν οι εξής δύο συνθήκες:
  i. $a \circ b \circ c = a \circ b \circ c \quad \forall a,b,c \in A$
  ii. υπάρχει στοιχείο του Α, το οποίο συμβολίζουμε με e και καλούμε ουδέτερο στοιχείο, τέτοιο ώστε
  $a \circ e = e \circ a = a, \quad \forall a \in A$

- ομάδα, αν είναι μονοειδές και κάθε στοιχείο έχει αντίστροφο, δηλαδή αν ισχύουν οι ακόλουθες συνθήκες:
  i. $a \circ b \circ c = a \circ b \circ c \quad \forall a,b,c \in A$
  ii. υπάρχει στοιχείο του Α, το οποίο συμβολίζουμε με e και καλούμε ουδέτερο στοιχείο, τέτοιο ώστε
  $a \circ e = e \circ a = a \quad \forall a \in A$
  iii. για κάθε $a \in A$ υπάρχει στοιχείο του Α, το οποίο συμβολίζουμε με a^{-1} και καλούμε αντίστροφο του a, τέτοιο ώστε
  $a^{-1} \circ a = a \circ a^{-1} = e \quad \forall a \in A$

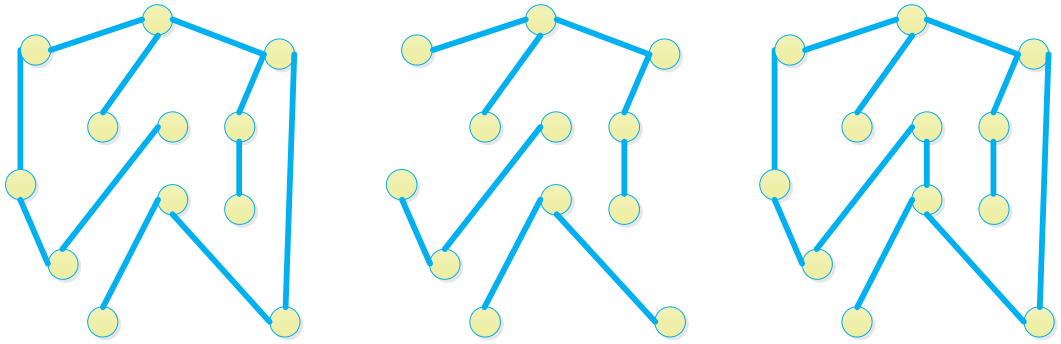
Επιπλέον μια ομάδα καλείται **αβελιανή ή αντιμεταθετική**, αν ισχύει η εξής ιδιότητα:
$a \circ b = b \circ a \quad \forall a,b \in A$

Δέντρα

Τα δέντρα ερμηνεύονται ως Poset όταν τα προσανατολίζουμε. Στη συνέχεια παραθέτω την έννοια των δέντρων στη θεωρία γράφων.

Γενικά περί Δέντρων στη θεωρία γράφων:

Γενικά στη θεωρία γράφων ένα ελεύθερο δέντρο είναι ένας συνδεδεμένος, άκυκλος, μη κατευθυνόμενος γράφος. Συχνά, όταν λέμε ότι ένας γράφος αποτελεί δέντρο, παραλείπουμε τον προσδιορισμό «ελεύθερο». Εάν ένας μη κατευθυνόμενος γράφος είναι άκυκλος, αλλά ενδεχομένως μη συνδεδεμένος, τότε συνιστά δάσος. Οι πλείστοι αλγόριθμοι που αφορούν δέντρα μπορούν να εφαρμοστούν και σε δάση. Στο σχήμα α) βλέπουμε ένα δέντρο , στο σχήμα β) ένα δάσος και στο σχήμα γ) ένα γράφο που δεν αποτελεί ούτε δέντρο ούτε δάσος, αφού περιλαμβάνει ένα κύκλο. Ένα δάσος δεν είναι δέντρο επειδή δεν είναι συνεκτικό.



α) β) γ)

Σχήμα 2.8 : α) Δέντρο. β) Δάσος. γ) Γράφος ο οποίος περιέχει έναν κύκλο, και συνεπώς δεν είναι ούτε δέντρο ούτε δάσος. Πηγή : Εισαγωγή στους αλγορίθμους.

Στο θεώρημα που ακολουθεί συνοψίζονται πολλά σημαντικά στοιχεία σχετικά με τα δέντρα.

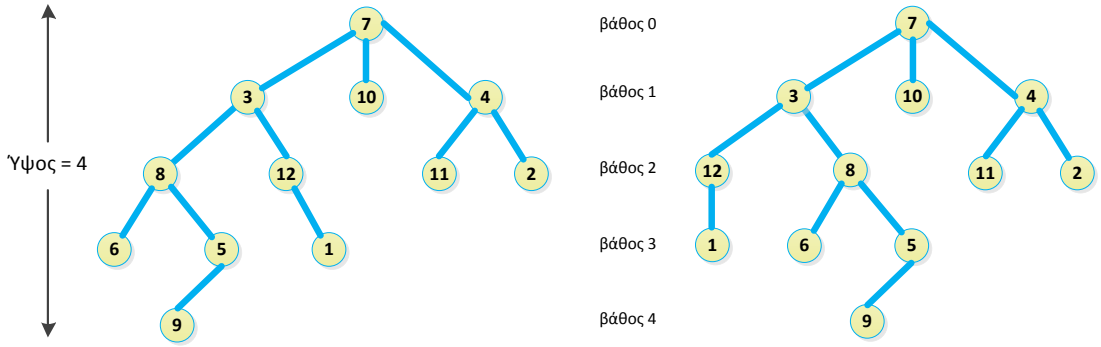
Θεώρημα : Ιδιότητες ελευθέρων δέντρων.

Έστω $G = (V, E)$ ένας μη κατευθυνόμενος γράφος. Οι ακόλουθες προτάσεις είναι ισοδύναμες μεταξύ τους.

1) Το G είναι ελεύθερο δέντρο.
2) Οποιοιδήποτε δύο κόμβοι του G συνδέονται μεταξύ τους μέσω μίας και μόνο απλής διαδρομής.
3) Το G είναι συνδεδεμένο, αλλά εάν αφαιρεθεί από το σύνολο ακμών E οποιαδήποτε ακμή, ο γράφος που προκύπτει είναι μη συνεκτικός.
4) Το G είναι συνδεδεμένο , και $E = V - 1$.
5) Το G είναι άκυκλο, και $E = V - 1$.
6) Το G είναι άκυκλο, αλλά εάν προστεθεί στο σύνολο E οποιαδήποτε ακμή, ο γράφος που προκύπτει περιέχει έναν κύκλο.

Έρριζα και διατεταγμένα δέντρα.

Έρριζο δέντρο είναι ένα ελεύθερο δέντρο στο οποίο ένας από τους κόμβους διακρίνεται από τους υπόλοιπους. Ο διακεκριμένος κόμβος ονομάζεται ρίζα ή ριζικός κόμβος του δέντρου. Στο σχήμα α) βλέπουμε ένα έρριζο δέντρο με 12 συνολικά κόμβους, και ρίζα τον κόμβο 7.



α) β)

Σχήμα 2.9 : Έρριζα και διατεταγμένα δέντρα. α) Έρριζο δέντρο με ύψος 4. β) Ένα έρριζο δέντρο. Ως έρριζο δέντρο, είναι πανομοιότυπο με το δέντρο του σχήματος (α), αλλά ως διατεταγμένο είναι διαφορετικό, αφού οι θυγατρικοί του κόμβου 3 εμφανίζονται με διαφορετική διάταξη. Πηγή : Εισαγωγή στους αλγορίθμους.

Έστω ένας κόμβος χ σε κάποιο έρριζο δέντρο Τ με ριζικό κόμβο r. Οποιοσδήποτε κόμβος y στη μοναδική διαδρομή από τον r μέχρι τον χ ονομάζεται πρόγονος του χ. Εάν ο y είναι πρόγονος του χ, ο χ είναι απόγονος του y (Κάθε κόμβος είναι πρόγονος και απόγονος του εαυτού του.). Εάν ο y είναι πρόγονος του χ και $\chi \neq y$, τότε ο y είναι γνήσιος πρόγονος του χ και ο χ γνήσιος απόγονος του y. Το υπόδεντρο που εκφύεται από τον χ, ή όπως αλλιώς λέγεται το υπόδεντρο με ρίζα το χ είναι το δέντρο που επάγεται από τους απογόνους του χ, και έχει ως ριζικό κόμβο τον χ. Παραδείγματος χάριν, το υπόδεντρο που εκφύεται από τον κόμβο 8 στο σχήμα α) περιέχει τους κόμβους 8,6,5 και 9.

Εάν η τελευταία ακμή στη διαδρομή από το ριζικό κόμβο r ενός δέντρου Τ μέχρι κάποιον κόμβο χ είναι η (y,χ), τότε ο y είναι ο πατρικός (κόμβος) του χ, και ο χ θυγατρικός (κόμβος) του y. Οι ριζικοί κόμβοι έχουν τον ίδιο πατρικό, ονομάζονται αδελφικοί (κόμβοι). Οι κόμβοι που δεν διαθέτουν θυγατρικούς ονομάζονται καταληκτικοί κόμβοι ή φύλλα. Οι μη καταληκτικοί κόμβοι ονομάζονται εσωτερικοί.

Το πλήθος των θυγατρικών ενός κόμβου χ σε ένα έρριζο δέντρο Τ ονομάζεται βαθμός του χ. Ο βαθμός ενός κόμβου σε ένα ελεύθερο δέντρο ισούται, όπως και σε ένα μη κατευθυνόμενο γράφο, με το πλήθος των προσκείμενων κόμβων. Σε ένα έρριζο δέντρο, όμως, ο βαθμός ισούται με το πλήθος των θυγατρικών – ο πατρικός ενός κόμβου δεν λαμβάνεται υπ' όψιν όσον αφορά το βαθμό του. Το μήκος της διαδρομής από το ριζικό κόμβο r μέχρι κάποιον κόμβο χ είναι το βάθος του χ στο Τ. Το ύψος ενός κόμβου σε ένα δέντρο ισούται με το πλήθος των ακμών στη μακρύτερη απλή κατωφερή διαδρομή από τον κόμβο μέχρι κάποιο φύλλο, ενώ το ύψος του δέντρου ισούται με το ύψος του ριζικού κόμβου. Το ύψος ενός δέντρου ισούται επίσης με το μεγαλύτερο από τα βάθη όλων των κόμβων του δέντρου.

Ένα διατεταγμένο δέντρο είναι ένα έρριζο δέντρο στο οποίο οι θυγατρικοί κάθε κόμβου είναι διατεταγμένοι. Δηλαδή, εάν ένας κόμβος έχει k θυγατρικούς, κάποιος από αυτούς είναι ο πρώτος, κάποιος ο δεύτερος, ..., και κάποιος είναι ο k-οστός.Τα δύο δέντρα του σχήματος είναι διαφορετικά εάν θεωρηθούν διατεταγμένα, αλλά όμοια εάν θεωρηθούν απλώς έρριζα δέντρα.

Παρόλο τον ειδικό χαρακτήρα που έχουν τα δέντρα, τα δέντρα στη θεωρία Poset είναι ακόμη πιο περιορισμένα. Τα δέντρα στα Poset είναι προσανατολισμένα. Αυτό έχει σαν συνέπεια όταν βάλουμε σε ένα έρριζο δέντρο προσανατολισμό από τον πατέρα προς τα παιδιά ή το αντίστροφο να μην είναι Poset.

Ιδιότητες:

1) Τα προσανατολισμένα δέντρα είναι άκυκλα.
2) Σε ένα κόμβο όλα τα βέλη εξέρχονται ή εισέρχονται.

Τα δέντρα P έχουν διάγραμμα Hasse που είναι δέντρο άρα $\dim P \leq 3$.

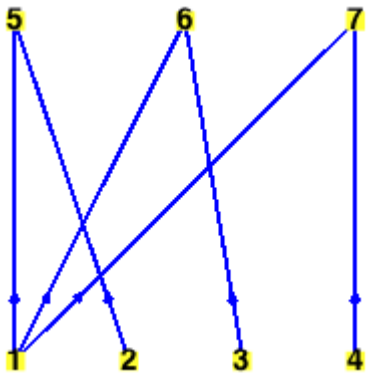
Παρατήρηση : Τα δέντρα ως Poset, σε διάγραμμα Hasse, έχουν ύψος 2.

Θεώρημα :

Τα δέντρα είναι επίπεδα επειδή δεν εμπεριέχουν τον $K_{3,3}$ και K_5.(Kuratowski)

Παρατήρηση :

Παρόλο που τα δέντρα δεν περιέχουν τον $K_{3,3}$ και K_5 εντούτοις το διάγραμμα Hasse δεν είναι επίπεδο.



Διάγραμμα 2.1 : Διάγραμμα τύπου Hasse του T.

1) $\dim T \leq width\ T = 4$ (Hiraguchi)
2) $\dim T \leq \frac{T}{2} = 3.5$ $\dim T \leq 3$ (Hiraguchi)
3) $\dim T = 3$ (Chaunier & Lygeros 1991)

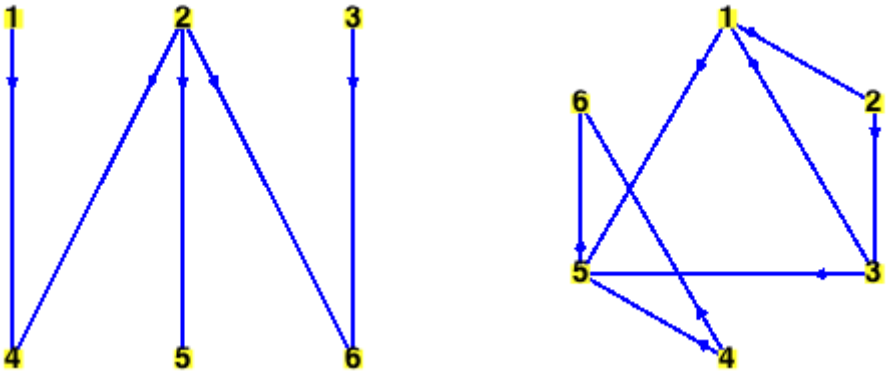
Παρατήρηση : Το T είναι κρίσιμο, δηλαδή όταν του αφαιρούμε οποιοδήποτε σημείο τότε η διάσταση κατεβαίνει μία μονάδα.

Απόδειξη:

Αν βγάλουμε το 1 ο γράφος χάνει τη συνεκτικότητά του και αποτελείται από τρία ευθύγραμμα τμήματα. Άρα είναι διάστασης 1.

Αν βγάλουμε το 5 ή το 6 ή το 7 έχουμε ως αποτέλεσμα ένα Poset διάστασης 2.

Αν βγάλουμε το 2 ή το 3 ή το 4 έχουμε ως αποτέλεσμα το εξής conjugate, άρα η διάσταση είναι μικρότερη ή ίση του 2.



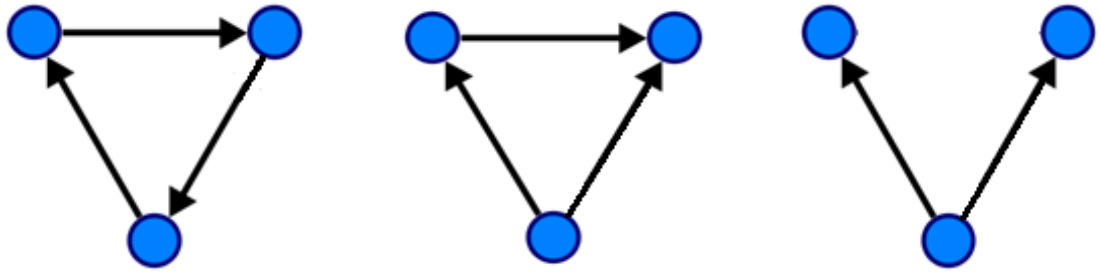
Διάγραμμα 2.2 : Αριστερά διάγραμμα t και δεξιά το conjugate του t.

Άρα αποδείξαμε ότι το Poset είναι κρίσιμο ως προς τη διάστασή του.

Είδαμε ότι τα δέντρα P δεν έχουν πάντα διάσταση μικρότερη του 3. Κατά συνέπεια δεν μπορούμε να εφαρμόσουμε το θεώρημα Dushnik & Miller.

Σύνολα Μερικής Διάταξης - POSET

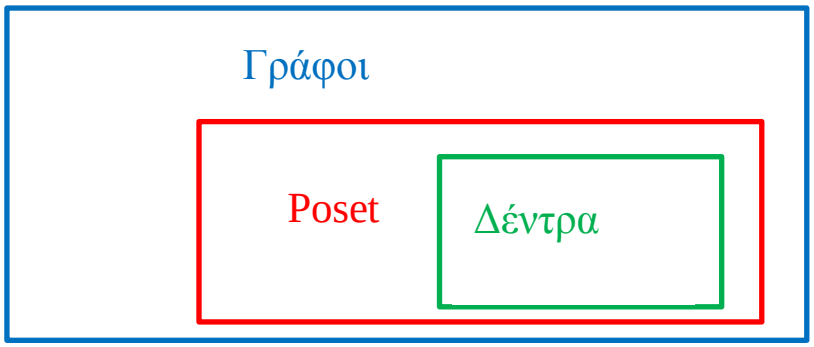
Το Poset (**P**artially **O**rdered **Set**) , δηλαδή, το Σύνολο Μερικής Διάταξης είναι μια υποκατηγορία των γράφων. Ένα Poset είναι ένα σύνολο σε συνδυασμό με μια μερική διάταξη σε αυτό. Τυπικά, ένα μερικώς διατεταγμένο σύνολο ορίζεται ως ένα διατεταγμένο δέντρο $P = X, \leq$, όπου X καλείται η αιτιολογία του P και $\leq$ είναι η μερική διάταξη του P. Με άλλα λόγια Poset είναι ένας κατευθυνόμενος γράφος ο οποίος μπορεί να έχει κύκλους, αλλά όχι κυκλώματα.



(α) Γράφος (β) Poset (γ) Δέντρο

Σχήμα 2.10 : Παράδειγμα Γράφου και Poset

Όπως καταλαβαίνουμε τα Poset είναι μία υποκατηγορία των γράφων. Τα δέντρα είναι γράφοι χωρίς κύκλους, κατά συνέπεια είναι υποκατηγορία των Poset.



Σχήμα 2.11 : Σχηματική αναπαράσταση της κατηγοριοποίησης των δομών.

Αξιώματα:

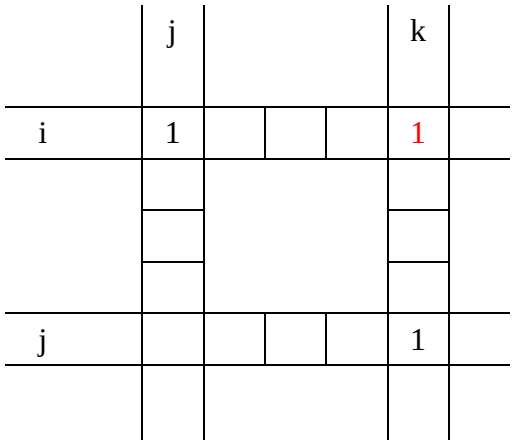
- $\forall x \in P : x \leq y$, Αντανακλαστική
- $\forall x, y \in P^2 : x \leq y \ \Lambda \ y \leq x \quad x = y$, Αντισυμμετρική
- $\forall x, y, z \in P^3 : x \leq y \ \Lambda \ y \leq z \quad x \leq z$, Μεταβατική

Έλεγχος εάν ένας πίνακας πρόσπτωσης είναι Poset:

Ελέγχουμε για κάθε τριάδα του πίνακα πρόσπτωσης εάν A[i,j] ισούται με 1 και A[j,k] εάν ισούται με 1, τότε A[i,k] πρέπει να ισούται με 1.

```
with(LinearAlgebra);

test_poset := proc ( A::Matrix  )
        local i,j,k,flag_poset,N;

        flag_poset := 1;
        N:= ColumnDimension(A);

        for i from 1 to N while (flag_poset = 1) do
            for j from 1 to N  while (flag_poset = 1) do
                for k from 1 to N  while ((flag_poset = 1) and (k <> i )) do
                        if ( ((A[i,j] = 1)  and (A[j,k]=1)) and (A[i,k]) = 0 ) then
                            flag_poset := 0;
                        end if;
                end do;
            end do;
        end do;

        RETURN(flag_poset);

end proc;
```

Κώδικας : Κώδικας Maple για έλεγχο εάν ένας πίνακας πρόσπτωσης είναι Poset.



Σχήμα 2.12 : Σχηματική Αναπαράσταση αλγορίθμου ελέγχου εάν ένας πίνακας πρόσπτωσης είναι Poset

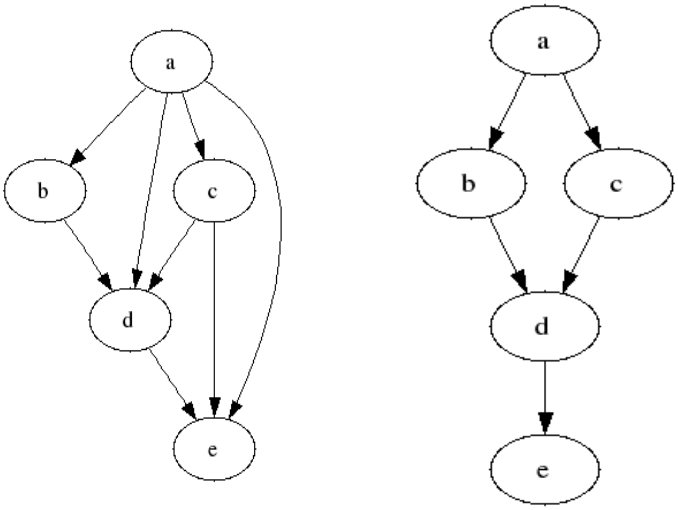
Διαγράμματα Hasse

Τα διαγράμματα Hasse ανήκουν στη θεωρία διάταξης των μαθηματικών. Είναι τύπος μαθηματικού διαγράμματος που χρησιμοποιείται για την αναπαράσταση ενός γράφου μερικής διάταξης (poset), σε μία ελαττωμένη μορφή (transitive reduction) γράφου.

Συγκεκριμένα, για ένα σύνολο μερικής διάταξης (poset) (S, ≤) κάποιος αναπαριστά κάθε στοιχείο του S σαν μια κορυφή στο επίπεδο και σύρει ένα τμήμα γραμμής ή καμπύλης που πηγαίνει προς τα πάνω από x στο y οποτεδήποτε το y καλύπτει το x (δηλαδή, οποτεδήποτε $x < y$ και δεν υπάρχει z έτσι ώστε $x \leq z \leq y$).

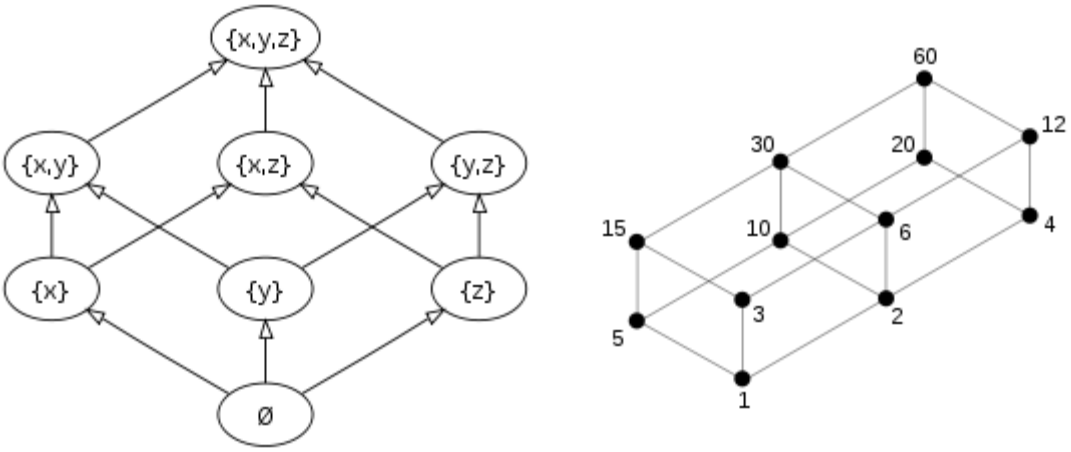
Οι καμπύλες αυτές μπορούν να διασταυρώνονται, αλλά δεν πρέπει να ακουμπούν σε καμιά άλλη κορυφή πλην του καταληκτικού τους σημείου.

Ένα τέτοιο διάγραμμα, με σήμανση στις (ονοματισμένες) κορυφές, προσδιορίζει μονοσήμαντα τη μερική διάταξή του.



Σχήμα 2.13 : Παράδειγμα ελάττωσης (transitive reduction)
Πηγή : Wikipedia

Η φράση διάγραμμα Hasse μπορεί να αναφέρεται και στην ελαττωμένη αναπαράσταση, όπως σε ένα αφηρημένο κατευθυνόμενο ακυκλικό γράφο, ανεξαρτήτως από οποιαδήποτε απεικόνιση του γράφου.



Διάγραμμα 2.3 : Διαγράμματα Hasse Πηγή : Wikipedia

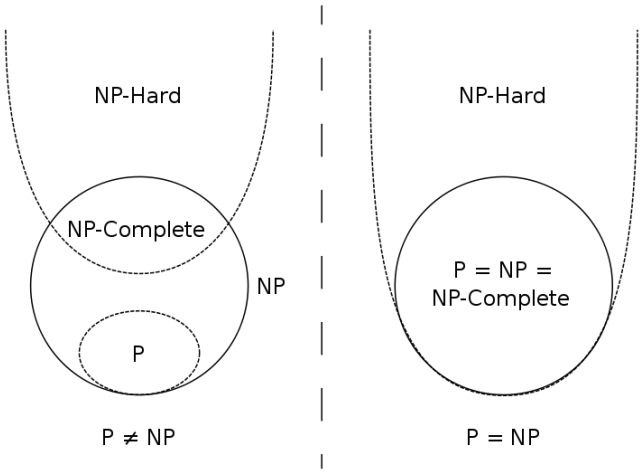
Παρά το γεγονός ότι τα διαγράμματα Hasse είναι απλά όσο και διαισθητικά εργαλεία για τη διαχείριση πεπερασμένων posets , καταλήγει να είναι σχετικά δύσκολο να σχεδιασθούν "καλά" διαγράμματα.

Ο λόγος είναι ότι υπάρχουν γενικά πολλοί δυνατοί τρόποι για να σχεδιασθεί ένα διάγραμμα Hasse για ένα δεδομένο poset.

Η απλή τεχνική που αρχίζεις με τα ελάχιστα στοιχεία μιας τάξης και στη συνέχεια προσθέτεις στοιχεία μεγαλύτερης τάξης, κλιμακωτά, συχνά παράγει αρκετά φτωχά αποτελέσματα καθώς μπορεί να χαθούν εύκολα συμμετρίες και η εσωτερική δομή της διάταξης.

Πολυπλοκότητα Αλγορίθμων

Η πολυπλοκότητα αλγορίθμων είναι ένα μεγάλο κεφάλαιο στην επιστήμη των υπολογιστών και ιδιαίτερα στη θεωρία υπολογισμού. Έτσι, τα προβλήματα και κατ' επέκταση και οι αλγόριθμοι χωρίστηκαν σε τάξεις πολυπλοκότητας, όπως φαίνεται στο παρακάτω διάγραμμα.



Σχήμα 2.14 : Διάγραμμα Euler για σχηματική αναπαράσταση P , NP , NP-complete και NP-hard κλάσεων από προβλήματα. Πηγή : Wikipedia

Θέματα Πολυπλοκότητας

Θα εξετάσουμε μερικά θέματα πολυπλοκότητας. Μια ενδιαφέρουσα κλάση προβλημάτων, τα λεγόμενα «NP- πλήρη» προβλήματα, των οποίων ο χαρακτήρας είναι άγνωστος. Για κανένα NP-πλήρες πρόβλημα δεν έχει ανακαλυφθεί ακόμη κάποιος αλγόριθμος πολυωνυμικού χρόνου για οποιοδήποτε από αυτά. Αυτό το ζήτημα, ονομάζεται «P ≠ NP ». Από την πρώτη στιγμή που τέθηκε, το 1971, έχει αναδειχθεί σε ένα από τα πιο σοβαρά και πιο δυσνόητα ανοικτά προβλήματα στην έρευνα της θεωρητικής επιστήμης υπολογιστών. Ένα ιδιαίτερα προκλητικό χαρακτηριστικό των NP-πλήρων προβλημάτων είναι ότι αρκετά από αυτά φαίνονται εκ πρώτης όψεως παρόμοια με προβλήματα που επιδέχονται αλγορίθμους πολυωνυμικού χρόνου. Σε κανένα από τα ακόλουθα ζεύγη προβλημάτων, το ένα πρόβλημα είναι επιλύσιμο σε πολυωνυμικό χρόνο και το άλλο σε NP-πλήρες, αν και τα δύο προβλήματα φαίνεται να διαφέρουν ελάχιστα μεταξύ τους.

Ελαφρύτατη έναντι βαρύτατης μεμονωμένης διαδρομής

Μπορούμε να προσδιορίσουμε μια μεμονωμένη *ελαφρύτατη* διαδρομή, ακόμη και με αρνητικά βάρη ακμών, σε ένα κατευθυνόμενο γράφο G=(V,E) σε χρόνο O(V·E). Η εύρεση μιας βαρύτατης απλής διαδρομής μεταξύ δύο κόμβων, όμως, είναι δύσκολη. Ακόμη και το να προσδιοριστεί εάν ένας γράφος περιλαμβάνει απλή διαδρομή με πλήθος ακμών μεγαλύτερο ή ίσο μίας δεδομένης ποσότητας αποτελεί NP-πλήρες πρόβλημα.

Διαδρομή Euler έναντι χαμιλτονιανού κύκλου

Μία ***διαδρομή Euler*** ενός συνδεδεμένου, κατευθυνόμενου γράφου G=(V,E) είναι ένας κύκλος ο οποίος διατρέχει κάθε ακμή του G ακριβώς μία φορά, αν και ενδέχεται να διέρχεται από ένα κόμβο περισσότερες από μία φορές. Είναι δυνατό να προσδιορίσουμε εάν κάποιο γράφημα περιλαμβάνει διαδρομή Euler σε χρόνο μόλις O(E), και μάλιστα στο χρόνο αυτό μπορούμε να βρούμε επίσης τις ακμές της διαδρομής αυτής. Ένας ***χαμιλτονιανός κύκλος*** ενός κατευθυνόμενου γράφου G = (V,E) είναι ένας απλός κύκλος που περιλαμβάνει όλους τους κόμβους του V. Το να προσδιοριστεί εάν κάποιος κατευθυνόμενος γράφος περιλαμβάνει ένα ***χαμιλτονιανό κύκλο*** αποτελεί NP-πλήρες πρόβλημα.

NP-πληρότητα και κλάσεις P και NP:

Η **κλάση P** αποτελείται από τα προβλήματα που μπορούν να επιλυθούν σε χρόνο $O(n^k)$, όπου το n είναι το μέγεθος της εισόδου του προβλήματος και k κάποια σταθερά. Τα περισσότερα από τα προβλήματα που εξετάσαμε στα προηγούμενα κεφάλαια ανήκουν στην κλάση P.

Η **κλάση NP** αποτελείται από τα προβλήματα που είναι « επαληθεύσιμα » σε πολυωνυμικό χρόνο. Η έκφραση αυτή σημαίνει ότι εάν μας δινόταν με κάποιο τρόπο κάποιο «πιστοποιητικό» μίας λύσης, θα μπορούσαμε να επαληθεύσουμε ότι το πιστοποιητικό αυτό είναι ορθό σε χρόνο πολυωνυμικό ως προς το μέγεθος της εισόδου του προβλήματος. Παραδείγματος χάριν, στο πρόβλημα του χαμιλτονιανού κύκλου, ένα τέτοιο πιστοποιητικό για κάποιο δεδομένο κατευθυνόμενο γράφο G = (V,E) θα ήταν κάποια ακολουθία $< u_1, u_2, u_3, \dots, u_V >$ από $|V|$ κόμβους. Μπορούμε να ελέγξουμε εύκολα σε πολυωνυμικό χρόνο ότι $u_i, u_{i+1} \in E$ για $i = 1,2,3, \dots, V - 1$ καθώς και ότι $u_V, u_1 \in E$.

Οποιαδήποτε προβλήματα που ανήκουν στην κλάση P ανήκουν επίσης στην NP, αφού εάν ένα πρόβλημα ανήκει στην P μπορεί να επιλυθεί σε πολυωνυμικό χρόνο χωρίς καν να δοθεί πιστοποιητικό. Η σχέση αυτή μπορεί να δειχθεί ότι $P \leq NP$. Το ανοικτό ερώτημα είναι εάν το P είναι γνήσιο υποσύνολο του NP ή όχι.

Σε περιγραφικό επίπεδο, ένα πρόβλημα ανήκει στην κλάση NPC (οπότε και χαρακτηρίζεται NP-πλήρες) εάν ανήκει στην NP και είναι εξίσου «δύσκολο» με οποιοδήποτε της NP. Οι περισσότεροι ερευνητές στο χώρο της θεωρητικής επιστήμης υπολογιστών πιστεύουν ότι τα NP-πλήρη προβλήματα είναι δυσεπίλυτα, διότι με δεδομένο το ευρύ φάσμα των NP-πλήρων προβλημάτων που έχουν μελετηθεί μέχρι σήμερα, χωρίς να έχει ανακαλυφθεί για κανένα από αυτά λύση πολυωνυμικού χρόνου, θα ήταν πραγματικά εκπληκτικό εάν όλα αυτά τα προβλήματα μπορούσαν να επιλυθούν σε πολυωνυμικό χρόνο. Εντούτοις, με δεδομένη την προσπάθεια που έχει καταβληθεί μέχρι στιγμής για να αποδειχθεί ότι τα NP-πλήρη προβλήματα είναι δυσεπίλυτα, χωρίς να έχει προκύψει κανένα οριστικό συμπέρασμα, δεν μπορούμε να

αποκλείσουμε το ενδεχόμενο τα NP-πλήρη προβλήματα να είναι στην πραγματικότητα επιλύσιμα σε πολυωνυμικό χρόνο.

Επισκόπηση τεχνικών απόδειξης της NP-πληρότητας

Οι τεχνικές με τις οποίες αποδεικνύεται ότι κάποιο συγκεκριμένο πρόβλημα είναι NP-πλήρες διαφέρουν από τις τεχνικές που χρησιμοποιούνται για το σχεδιασμό και ανάλυση αλγορίθμων. Η διαφορά αυτή έχει μία θεμελιώδη αιτία: όταν αποδεικνύουμε ότι ένα πρόβλημα είναι NP-πλήρες, διατυπώνουμε μια θέση για το πόσο δύσκολο είναι, και όχι για το πόσο εύκολο είναι. Δεν προσπαθούμε να αποδείξουμε ότι υπάρχει δραστικός αλγόριθμος, αλλά ότι κατά πάσα πιθανότητα δεν υπάρχει. Καταυτόν τον τρόπο, οι αποδείξεις της NP-πληρότητας είναι παρόμοιες με την απόδειξη ότι για οποιονδήποτε αλγόριθμο συγκριτικής ταξινόμησης υπάρχει ένα κάτω χρονικό φράγμα $\Omega(n \log n)$.

Για να δείξουμε ότι κάποιο πρόβλημα είναι NP-πλήρες βασιζόμαστε σε τρεις καίριες έννοιες:

Προβλήματα διάγνωσης έναντι προβλημάτων βελτιστοποίησης

Πολλά ενδιαφέροντα προβλήματα είναι προβλήματα βελτιστοποίησης, στα οποία σε κάθε εφικτή (δηλ. «επιτρεπτή») λύση αντιστοιχεί κάποια τιμή, και το ζητούμενο είναι να βρεθεί μια εφικτή λύση με την καλύτερη τιμή. Παραδείγματος χάριν, σε ένα πρόβλημα που ονομάζεται ΒΡΑΧΥΤΑΤΗ ΔΙΑΔΡΟΜΗ, μας δίνεται ένας μη κατευθυνόμενος γράφος G και δύο κόμβοι u και υ, και μας ζητείται να βρούμε μια διαδρομή από τον u μέχρι τον υ η οποία να αποτελείται από τις ελάχιστες δυνατές ακμές. (Με άλλα λόγια, το πρόβλημα ΒΡΑΧΥΤΑΤΗ ΔΙΑΔΡΟΜΗ είναι το πρόβλημα της μεμονωμένης ελαφρύτατης διαδρομής σε ένα αβαρή, μη κατευθυνόμενο γράφο.) Η NP-πληρότητα, όμως, αφορά άμεσα όχι τα προβλήματα βελτιστοποίησης, αλλά τα προβλήματα διάγνωσης, στα οποία η απάντηση είναι απλώς «ναι» ή «όχι» (ή, σε πιο τυπική μορφή, «1» ή «0»).

Παρ' όλο που το να αποδειχθεί ότι κάποιο πρόβλημα είναι NP-πλήρες μας περιορίζει στην επικράτεια των προβλημάτων διάγνωσης, τα προβλήματα βελτιστοποίησης συνδέονται με τα προβλήματα διάγνωσης με μια χρήσιμη σχέση. Ένα δεδομένο πρόβλημα βελτιστοποίησης συνήθως μπορεί να διατυπωθεί ως αντίστοιχο πρόβλημα διάγνωσης, εάν επιβληθεί ένα φράγμα στη βελτιστοποιητέα τιμή. Για το πρόβλημα ΒΡΑΧΥΤΑΤΗ ΔΙΑΔΡΟΜΗ, παραδείγματος χάριν, ένα αντίστοιχο πρόβλημα διάγνωσης, το οποίο θα ονομάσουμε ΔΙΑΔΡΟΜΗ, είναι εάν, όταν δοθεί κάποιος κατευθυνόμενος γράφος G, κάποιοι κόμβοι u και υ, και ένας ακέραιος k, υπάρχει διαδρομή από τον u μέχρι τον υ που να αποτελείται από k το πολύ ακμές.

Η σχέση μεταξύ ενός προβλήματος βελτιστοποίησης και του αντίστοιχου προβλήματος διάγνωσης λειτουργεί υπέρ μας, όταν προσπαθούμε να αποδείξουμε ότι το πρόβλημα βελτιστοποίησης είναι «δύσκολο». Ο λόγος είναι ότι το πρόβλημα διάγνωσης είναι κατά μια έννοια «ευκολότερο», ή τουλάχιστον «όχι δυσκολότερο». Ας δούμε ένα συγκεκριμένο παράδειγμα: για να επιλύσουμε το πρόβλημα ΔΙΑΔΡΟΜΗ μπορούμε να λύσουμε το πρόβλημα ΒΡΑΧΥΤΑΤΗ ΔΙΑΔΡΟΜΗ και κατόπιν να συγκρίνουμε το πλήθος των ακμών στη βραχύτερη διαδρομή πού βρέθηκε με την τιμή της παραμέτρου k του προβλήματος διάγνωσης. Με άλλα λόγια, εάν ένα πρόβλημα βελτιστοποίησης είναι εύκολο, το αντίστοιχο πρόβλημα διάγνωσης είναι επίσης εύκολο. Για να το διατυπώσουμε με τρόπο που να συνδέεται αμεσότερα με την NP-πληρότητα, όταν μπορούμε να καταδείξουμε ότι ένα πρόβλημα διάγνωσης είναι δύσκολο, καταδεικνύουμε ταυτόχρονα ότι το αντίστοιχό του πρόβλημα βελτιστοποίησης είναι επίσης δύσκολο. Συνεπώς, παρ' όλο πού η θεωρία της NP-πληρότητας περιορίζει την προσοχή μας στα προβλήματα διάγνωσης, συχνά έχει επιπτώσεις και στα προβλήματα βελτιστοποίησης.

Αναγωγές

Η τακτική του να αποδεικνύει κανείς ότι κάποιο πρόβλημα δεν είναι δυσκολότερο ή ευκολότερο από κάποιο άλλο εφαρμόζεται ακόμη και όταν και τα δύο προβλήματα είναι προβλήματα διάγνωσης. Η ιδέα αυτή αξιοποιείται σχεδόν σε κάθε απόδειξη NP-πληρότητας, ως εξής: Έστω ένα πρόβλημα διάγνωσης, λόγου χάριν το Α, το οποίο θα θέλαμε να επιλύσουμε σε πολυωνυμικό χρόνο. Η είσοδος σε κάποιο συγκεκριμένο πρόβλημα αποτελεί όπως λέμε ένα στιγμιότυπο του προβλήματος. Παραδείγματος χάριν, για το πρόβλημα ΔΙΑΔΡΟΜΗ, ένα στιγμιότυπο θα ήταν κάποιος συγκεκριμένος γράφος G, δύο συγκεκριμένοι κόμβοι u και υ του G, και ένας συγκεκριμένος ακέραιος k. Ας υποθέσουμε επίσης ότι υπάρχει κάποιο διαφορετικό πρόβλημα διάγνωσης, π.χ. το Β, το οποίο γνωρίζουμε ήδη πώς να επιλύσουμε σε πολυωνυμικό χρόνο. Τέλος, ας υποθέσουμε ότι έχουμε μια διαδικασία η οποία μετασχηματίζει οποιοδήποτε στιγμιότυπο α του Α σε κάποιο στιγμιότυπο β του Β με τα εξής χαρακτηριστικά:

1. Ο μετασχηματισμός απαιτεί πολυωνυμικό χρόνο.

2. Οι απαντήσεις είναι οι ίδιες. Δηλαδή η απάντηση για το στιγμιότυπο α είναι «ναι» εάν και μόνο εάν η απάντηση για το β είναι επίσης «ναι».

Μια τέτοια διαδικασία ονομάζεται αναγωγικός αλγόριθμος πολυωνυμικού χρόνου, και όπως φαίνεται στο Σχήμα 2.18 μας παρέχει έναν τρόπο να επιλύσουμε το πρόβλημα Α σε πολυωνυμικό χρόνο:

1. Για κάποιο δεδομένο στιγμιότυπο α του προβλήματος Α, μετασχηματίζουμε το α στο στιγμιότυπο β του προβλήματος Β χρησιμοποιώντας έναν αναγωγικό αλγόριθμο πολυωνυμικού χρόνου.

2. Εκτελούμε το διαγνωστικό αλγόριθμο πολυωνυμικού χρόνου για το Β, επί του στιγμιότυπου β.

3. Χρησιμοποιούμε την απάντηση για το β ως απάντηση για το α.

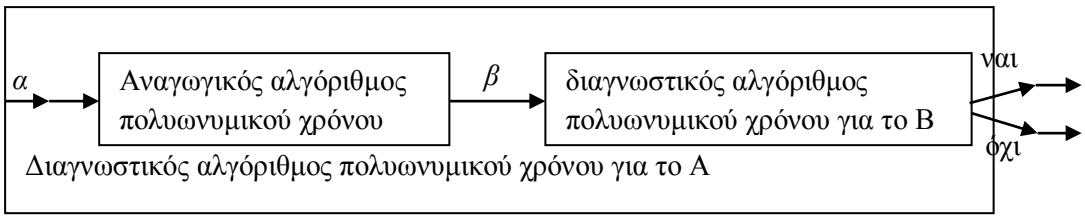
Εφ' όσον καθένα από τα τρία αυτά βήματα απαιτεί πολυωνυμικό χρόνο, το ίδιο ισχύει και για τα τρία μαζί, και συνεπώς έχουμε έναν τρόπο να λάβουμε διάγνωση για το α σε πολυωνυμικό χρόνο. Με άλλα λόγια, «ανάγοντας» την επίλυση του προβλήματος Α στην επίλυση τον προβλήματος Β, αποδεικνύουμε την «ευκολία» του Α μέσω της «ευκολίας» του Β.

Δεδομένου ότι η NP-πληρότητα αφορά το να αποδειχθεί όχι πόσο εύκολο, αλλά πόσο δύσκολο είναι ένα πρόβλημα, για να δειχθεί ότι κάποιο πρόβλημα είναι NP-πλήρες, οι αναγωγές πολυωνυμικού χρόνου θα πρέπει να χρησιμοποιηθούν με τον αντίθετο τρόπο. Προεκτείνοντας λίγο τη σχετική ιδέα, ας δούμε πώς θα μπορούσαμε χρησιμοποιώντας τις πολυωνυμικού χρόνου αναγωγές να δείξουμε ότι δεν είναι δυνατόν να υπάρχει αλγόριθμος πολυωνυμικού χρόνου για κάποιο συγκεκριμένο πρόβλημα Β. Ας υποθέσουμε ότι έχουμε ένα πρόβλημα διάγνωσης Α για το οποίο γνωρίζουμε ήδη ότι δεν είναι δυνατόν να υπάρχει αλγόριθμος πολυωνυμικού χρόνου. (Το πώς μπορεί να βρεθεί έναν τέτοιο πρόβλημα Α δεν θα μας απασχολήσει προς το παρόν.) Ας υποθέσουμε επιπλέον ότι έχουμε μια αναγωγή πολυωνυμικού χρόνου η οποία μετασχηματίζει στιγμιότυπα του Α σε στιγμιότυπα του Β. Στην περίπτωση αυτή μπορούμε να αποδείξουμε εύκολα με την εις άτοπον απαγωγή ότι δεν είναι δυνατόν να υπάρχει αλγόριθμος πολυωνυμικού χρόνου για το Β. Έστω ότι αυτό δεν ισχύει, δηλ. έστω ότι υπάρχει αλγόριθμος πολυωνυμικού χρόνου για το Β. Επομένως, χρησιμοποιώντας τη μέθοδο που απεικονίζεται στο Σχήμα 2.15, θα είχαμε τη δυνατότητα να επιλύσουμε το πρόβλημα Α σε πολυωνυμικό χρόνο, πράγμα που αντιφάσκει με την παραδοχή μας ότι δεν υπάρχει αλγόριθμος πολυωνυμικού χρόνου για το Α.

Όσον αφορά την NP-πληρότητα, δεν μπορούμε να υποθέσουμε ότι αποκλείεται να υπάρχει αλγόριθμος πολυωνυμικού χρόνου για το Α. Η αποδεικτική μεθοδολογία όμως είναι παρόμοια, με την έννοια ότι αποδεικνύουμε πως το πρόβλημα Β είναι NP-πλήρες με την παραδοχή ότι το πρόβλημα Α είναι επίσης NP-πλήρες.

Ένα πρώτο NP-πλήρες πρόβλημα

Καθώς η τεχνική της αναγωγής απαιτεί να έχουμε ένα πρόβλημα για το οποίο ήδη γνωρίζουμε ότι είναι NP-πλήρες προκειμένου να αποδείξουμε ότι κάποιο άλλο πρόβλημα είναι επίσης NP-πλήρες, χρειαζόμαστε ένα «πρώτο» NP-πλήρες πρόβλημα. Το πρόβλημα που θα χρησιμοποιήσουμε θα είναι η αληθινότητα κυκλώματος: μας δίνεται ένα λογικό συνδυαστικό κύκλωμα που απαρτίζεται από πύλες ΚΑΙ, Ή και ΟΧΙ, και το ζητούμενο είναι να βρεθεί εάν υπάρχει κάποιο σύνολο από λογικές τιμές εισόδου στο κύκλωμα για τις οποίες το κύκλωμα να δίνει ως έξοδο 1.



Σχήμα 2.15 : Επίλυση ενός προβλήματος διάγνωσης Α σε πολυωνυμικό χρόνο με τη βοήθεια ενός αναγωγικού αλγορίθμου πολυωνυμικού χρόνου, όταν διαθέτουμε ένα διαγνωστικό αλγόριθμο πολυωνυμικού χρόνου για κάποιο άλλο πρόβλημα Β. Σε πολυωνυμικό χρόνο, μετασχηματίζουμε ένα στιγμιότυπο α του Α σε κάποιο στιγμιότυπο β του Β, επιλύουμε το Β, και χρησιμοποιούμε την απάντηση για το β ως απάντηση για το α. Πηγή : Εισαγωγή στους αλγορίθμους.

Θέματα Αποκρισιμότητας

Σε μία μέθοδο απόφασης για στοιχειώδη άλγεβρα και γεωμετρία, ο Tarski έδειξε με την μέθοδο της απάλειψης των ποσοδεικτών (∀, ∃), ότι η θεωρία πρώτης τάξης των πραγματικών αριθμών υπό την πρόσθεση και πολλαπλασιασμό, είναι αποκρίσιμη (decidable). Αυτό είναι ένα πολύ περίεργο αποτέλεσμα. Ο Alonzo Church απέδειξε το 1936 ότι η αριθμητική του Peano δεν είναι αποφάνσιμη. Η αριθμητική Peano δεν είναι πλήρης (Θεώρημα μη Πληρότητας Gödel 1931).

Διάσταση – Dimension

Η διάσταση ενός συνόλου μερικής διάταξης (poset) είναι ο ελάχιστος αριθμός όλων των αλυσίδων των οποίων η τομή συνεπάγεται τη μερική διάταξη. Μερικές φορές αυτή η έννοια καλείται και διάσταση διάταξης ή διάσταση Dushnik–Miller. Το 1941 οι Dushnik και Miller μελέτησαν για πρώτη φορά τη διάσταση διάταξης. Για εκτενέστερη αναφορά σε αυτό το θέμα βλέπε Trotter (1992).

Ορισμός : Η διάσταση ενός poset P είναι ο ελάχιστος ακέραιος t για τον οποίο υπάρχει η οικογένεια,

$R = <_1, \dots, <_t$

για τη γραμμική επέκταση του P έτσι ώστε, για κάθε x και y στο P, το x προηγείται του y στο P, εάν και μόνο εάν το x προηγείται του y σε κάθε γραμμική επέκταση. Αυτό είναι:

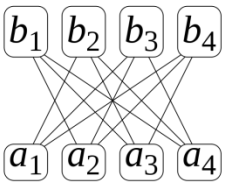
$P = \quad R = \bigcap_{i=1}^{t} <_i$

Πραγματοποιήσιμα :

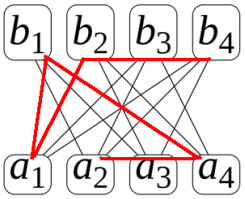
Μία οικογένεια $R = <_1, \dots, <_t$ από γραμμικές διατάξεις στο X καλούνται πραγματοποιήσιμα για το poset $P = X, <_p$ *εάν* $<_p =$ R, στην οποία λέγεται ότι κάποιο x και y στο X, $x <_p y$ ακριβώς όταν $x <_1 y$, $x <_2 y$, ... , και $x <_t y$. Μπορεί να δειχθεί ότι για οποιαδήποτε τέτοια οικογένεια R είναι πραγματοποιήσιμο εάν και μόνο εάν, για κάθε κρίσιμο ζεύγος (x,y) του P, $y <_i x$ για κάποια διάταξη $<_i$ στο R.

Γραμμικές Επεκτάσεις :

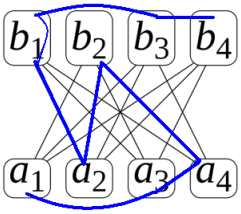
Γραμμική επέκταση είναι μία γραμμική σειρά που περνά από όλες τις κορυφές-κόμβους μία φορά και ικανοποιεί τη διάταξη του γράφου.



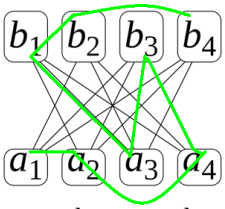
$a_2 < a_3 < a_4 < b_1 < a_1 < b_2 < b_3 < b_4$
$a_1 < a_3 < a_4 < b_2 < a_2 < b_1 < b_3 < b_4$
$a_1 < a_2 < a_4 < b_3 < a_3 < b_1 < b_2 < b_4$
$a_1 < a_2 < a_3 < b_4 < a_4 < b_1 < b_2 < b_3$



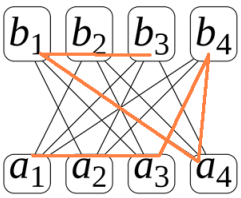
$a_2 < a_3 < a_4 < b_1 < a_1 < b_2 < b_3 < b_4$



$a_1 < a_3 < a_4 < b_2 < a_2 < b_1 < b_3 < b_4$



$a_1 < a_2 < a_4 < b_3 < a_3 < b_1 < b_2 < b_4$



$a_1 < a_2 < a_3 < b_4 < a_4 < b_1 < b_2 < b_3$

Σχήμα 2.16 : Σχηματική αναπαράσταση του Poset S_4 σε διάγραμμα Hasse. Πραγματοποιείται από τις τέσσερις αλυσίδες. D=4.

Ο Μιχάλης Γιαννακάκης το 1982 διατύπωσε το θεώρημα που συσχετίζει την εύρεση της διάστασης ενός poset με το χρωματισμό του γράφου.

Θεώρημα :

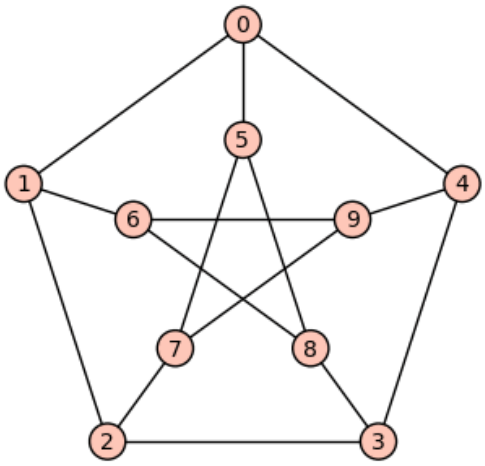
Είναι εφικτό να υπολογίσεις σε πολυωνυμικό χρόνο οποτεδήποτε σου δίνεται ένα πεπερασμένο poset που έχει διάσταση το πολύ δύο, για υπόδειξη, με τη δοκιμή εάν το γράφημα σύγκρισης του poset είναι ένα γράφημα μετάθεσης. Ωστόσο, για κάθε $D \geq 3$, είναι NP-complete για να ελέγξουμε κατά πόσο η διάσταση της διάταξης είναι το πολύ k.

Χρωματισμός Γράφου:

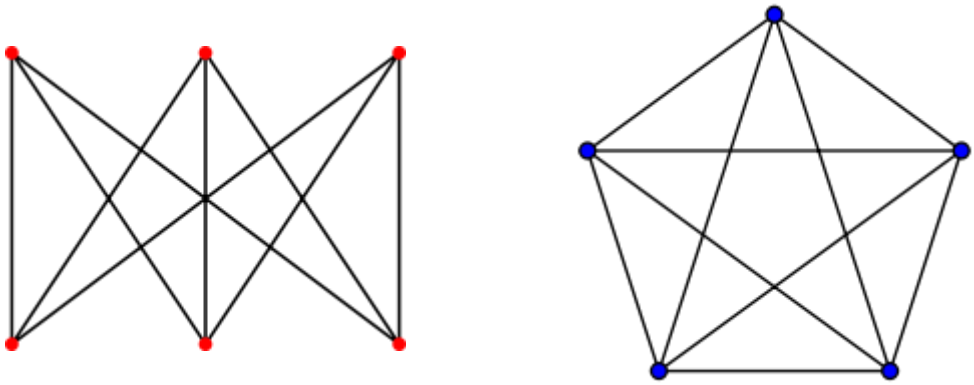
Στη θεωρία γράφων, ο χρωματισμός γράφου είναι μια ειδική κατηγορία του ονοματισμού γράφου. Ποίος είναι ο ελάχιστος αριθμός των χρωμάτων που χρειάζεται για το χρωματισμό ενός γράφου; Χρειάζονται N χρώματα για K_N.

Σχήμα 2.17 : Παράδειγμα ισομορφισμού με αστέρες γράφους.

Για παράδειγμα ο γράφος του Petersen εμπεριέχει τον $K_{3,3}$ και τον K_5. Δηλαδή:



Σχήμα 2.18 : Γράφος του Petersen, Πηγή : Wikipedia

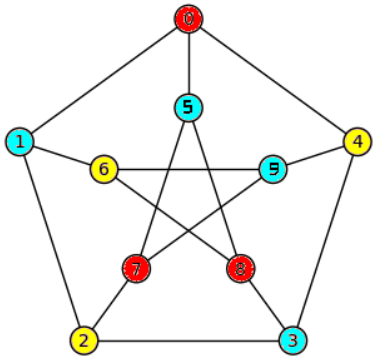


$K_{3,3}$ K_5

Απόδειξη ότι ο γράφος του Petersen χρειάζεται τρία χρώματα για να ζωγραφιστεί:

Α) Έστω 5 (Β) ⇒ 0 (Κ) και 8 (Κ) ⇒ 1 (Β) και 6 (Β) Άτοπο!

Β)



Ζωγραφίζεται με τρία χρώματα.

Χρ ≤ 3

$A)\quad X\rho > 2$

$B)\quad X\rho \leq 3$

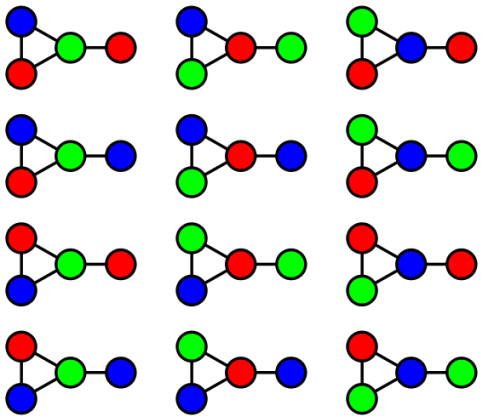
$X\rho$ 3

Άρα χρειάζονται τρία χρώματα για να ζωγραφιστεί ο γράφος του Petersen. □

Θεώρημα :

Ο χρωματισμός γράφου έχει ίση πολυπλοκότητα με το dimension 3. (Γιαννακάκης)

Εφόσον τα προβλήματα του χρωματισμού των γράφων είναι ισόμορφα με το πρόβλημα της διάστασης D ≥ 3.



Σχήμα 2.19 : Μη ισομορφικοί χρωματισμοί του γράφου.

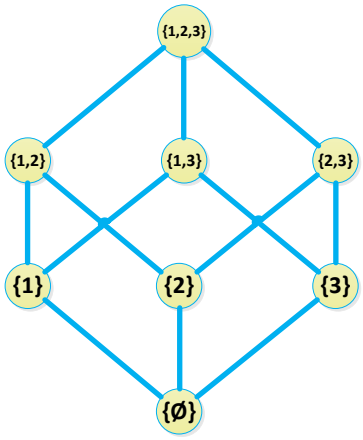
Εικασία SSU (Sidney-Sidney-Urrutia Conjecture)

Η εικασία Sidney-Sidney-Urrutia Conjecture γνωστή ως SSU αναφέρεται στο αν μπορεί ένα Poset να αναπαρασταθεί με κύκλους. Δηλαδή, δοθέντος ενός Poset μπορώ να βρω ένα συνδυασμό από κύκλους τέτοιο ώστε να ικανοποιούνται οι σχέσεις. Είναι γνωστό ότι το πρώτο γνωστό αντιπαράδειγμα είναι το $P_{14.}$ και αποδείχτηκε το 1989 από τους Brightwell & Winkler. Η εικασία SSU δηλώνει ότι αυτό το αντιπαράδειγμα είναι το μικρότερο σε μέγεθος στοιχείων.

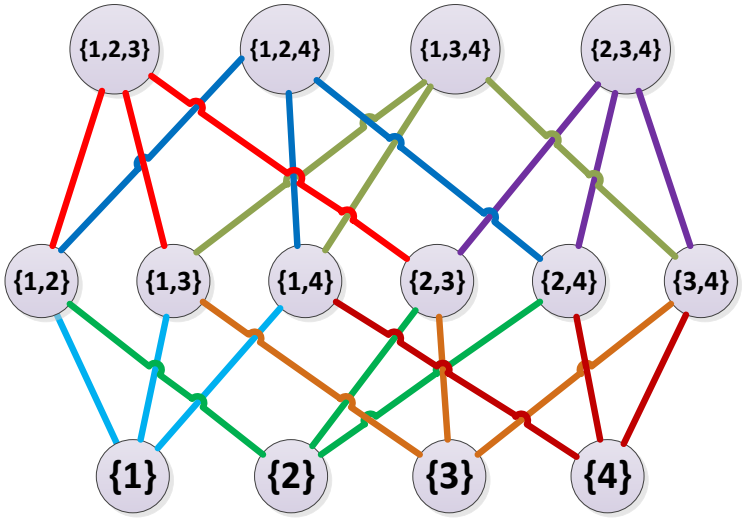
Γενικότερα, έχει αποδειχτεί ότι σίγουρα θα υπάρχει ένα μη κυκλικά αναπαραστάσιμο Poset όταν $P \geq 2^{d+2} - 2$, όπου $\boldsymbol{d}$ είναι η χωρική διάσταση. Για $\boldsymbol{d} = 2$ έχουμε κύκλους στο επίπεδο, για $\boldsymbol{d} = 3$ έχουμε σφαίρες στο χώρο και ούτω καθεξής.

Πως εφαρμόζεται το παραπάνω στο P_{14}:

$$2^{d+2} - 2 = 2^{2+2} - 2 = 2^4 - 2 = 16 - 2 = 14$$

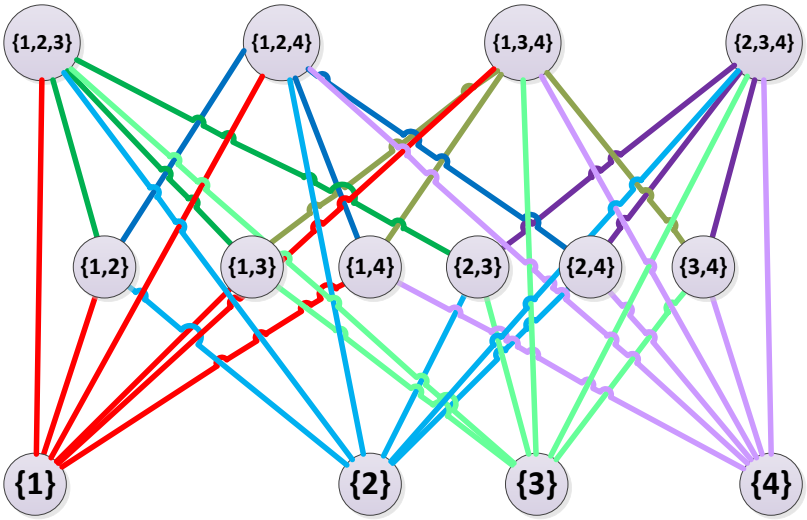


Διάγραμμα 2.4 : P_8 Κυκλικά αναπαραστάσιμος κύβος.

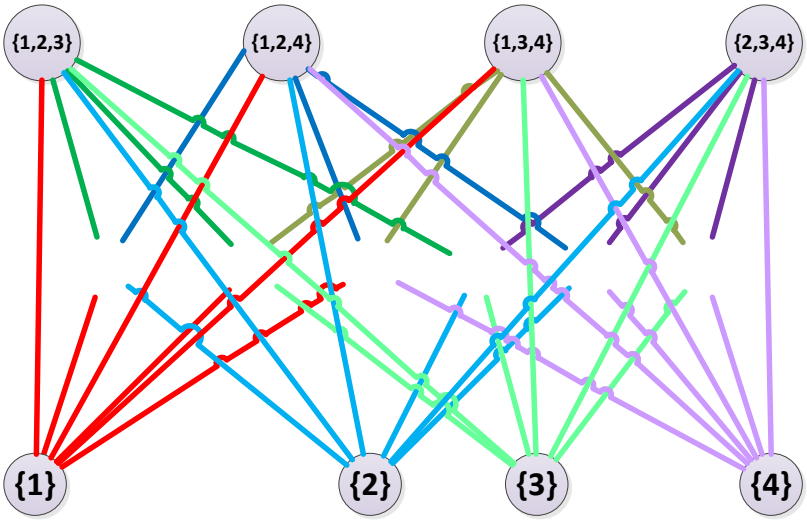


Διάγραμμα 2.5 : Hasse διάγραμα του P_{14} Κυκλικά μη αναπαραστάσιμο .

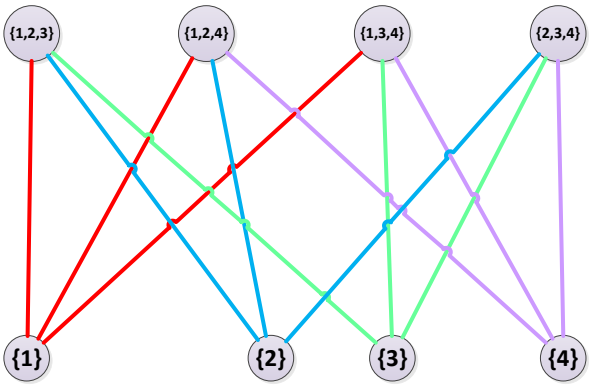
Μελέτη εάν ο S_4 είναι ελάσσονας γράφος του P_{14}.



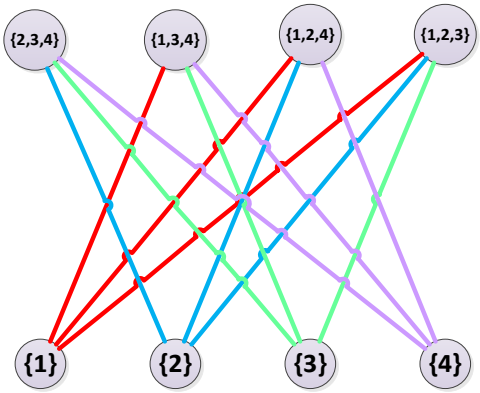
Ανάπτυξη Hasse διαγράμματος του P_{14}



Ενδιάμεση φάση κατά την απαλοιφή κόμβων



Απαλοιφή ακμών



Διάγραμμα του S_4

Σχήμα 2.20 : Στο σύμπλεγμα αυτό των σχημάτων γίνεται μελέτη εάν το S_4 είναι ελάσσονας γράφος του γράφου P_{14}

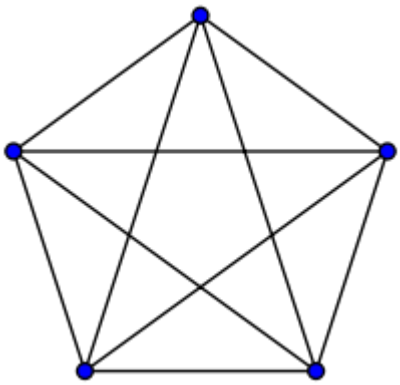
Γεγονός : Ο γράφος S_4 είναι ελάσσονας γράφος του γράφου $P_{14.}$

Γεγονός : Ο γράφος S_4 είναι υπογράφος του γράφου $P_{14.}$

Έστω P_{min} το μικρότερο Poset το οποίο δεν έχει κυκλική αναπαράσταση. Το P_{min} έχει απαραίτητα τις επόμενες ιδιότητες

- (B&L&S 2006) $11 \leq card\ \ P_{min}\ \leq 14$ (B&W 1989)
- (D&M 1941) $3 \leq dim(\mathrm{P_{min}}) \leq 4$ αν min είναι ως προς τη διάσταση (dim) αλλιώς $3 \leq dim(\mathrm{P_{min}}) \leq 6$ (H 1955)

Κλίκες : Στις κλίκες όλοι οι κόμβοι πρέπει να συνδέονται μεταξύ τους όπως τα K_N. (KURATOWSKI)



K_5

Διάγραμμα Hasse K_5

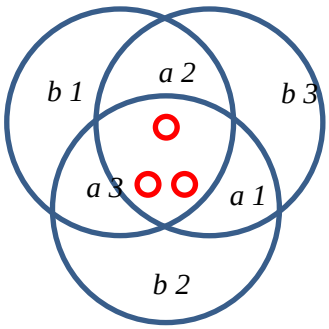
Σχήμα 2.21 : Παράδειγμα κλίκας

Αλυσίδα : Είναι ένα σύνολο με απόλυτη διάταξη.

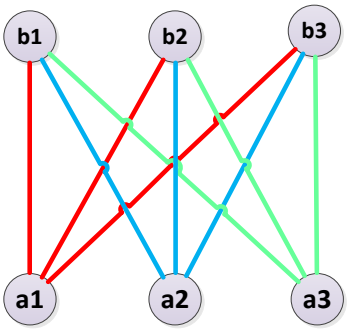
Γεγονός : Όλα τα K_N έχουν διάσταση 1 , $dim(K_N) = 1$, άρα έχουν κυκλική αναπαράσταση μέσω του θεωρήματος Dilworth & Miller (1941).

Λήμμα : Τα $K_{N,N}$ έχουν κυκλική αναπαράσταση.

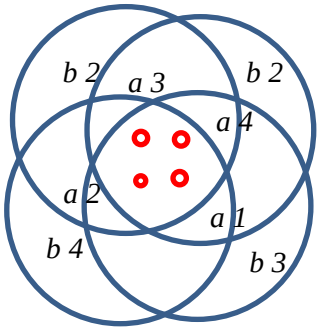
Απόδειξη : Τα $K_{N,N}$ είναι διμερείς γράφοι (bipartite graphs) κατά συνέπεια έχουν ύψος 2. Τα $K_{N,N}$ εμπεριέχουν τα S_N άρα dim $K_{N,N} \geq N$. Όπως $width\ K_{N,N} = N$ έχουμε dim $K_{N,N} = N$. Όλοι οι κύκλοι των ατόμων βρίσκονται στην κοινή τομή των κύκλων των συνατόμων.



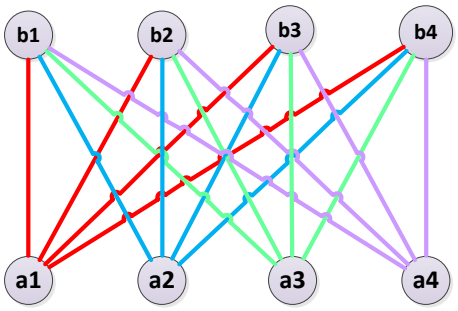
Κυκλική αναπαράσταση του $K_{3,3}$



Διάγραμμα $K_{3,3}$



Κυκλική αναπαράσταση του $K_{4,4}$



Διάγραμμα $K_{4,4}$

Σχήμα 2.22 : Κυκλική αναπαράσταση των $K_{3,3}$, $K_{4,4}$.

Θεώρημα :

Ένα poset P με πάχος w μπορεί να επιμεριστεί σε w αλυσίδες. (Dilworth, 1950)

Δυϊκό θεώρημα :

Ένα poset P με ύψος h μπορεί να επιμεριστεί σε h αντιαλυσίδες.

Θεώρημα : Για κάθε X , Y που είναι Poset, το C είναι αλυσίδα μέσα στο Poset και το x ανήκει στο X.

1) $\dim X - x \leq \dim X \leq 1 + \dim(X - x)$, ισχύει με την κρισιμότητα <
2) $\dim X \leq 2 + \dim(X - C)$, Βγάζω από το Poset μία αλυσίδα.
3) $\dim X \leq width\ X$, Θεώρημα Hiraguchi
4) $\dim X \leq \frac{X}{2}$, Θεώρημα Hiraguchi 1951
5) $\dim X \times Y \leq \dim X + \dim(Y)$

Ένα Poset έχει διάσταση - dimension 1 αν και μόνο αν είναι αλυσίδα. Μία αντιαλυσίδα που έχει τουλάχιστο 2 στοιχεία έχει dimension 2.

Σημειώνουμε εδώ ότι το |X| υποδηλώνει την πληθικότητα (cardinality), δηλαδή, πόσες κορυφές-στοιχεία περιέχει. Το width υποδηλώνει πόσο είναι το μέγιστο πλάτος. Δηλαδή, είναι ο μέγιστος αριθμός στοιχείων που εμφανίζονται σε μία γραμμή όταν απεικονιστεί το Poset με τη σύμβαση Hasse .

Λήμμα :

Υποθέτουμε x και y είναι ασύγκριτα σημεία σε ένα Poset X, αλλά για κάθε $z \in X - x, y, z > x$ *εάν* $z > y$ *και* $z < x$ *εάν* $z < y$. Τότε $\dim X - x = \dim X$ *εκτός και αν* $X - x$ είναι μία αλυσίδα.

Απόδειξη :

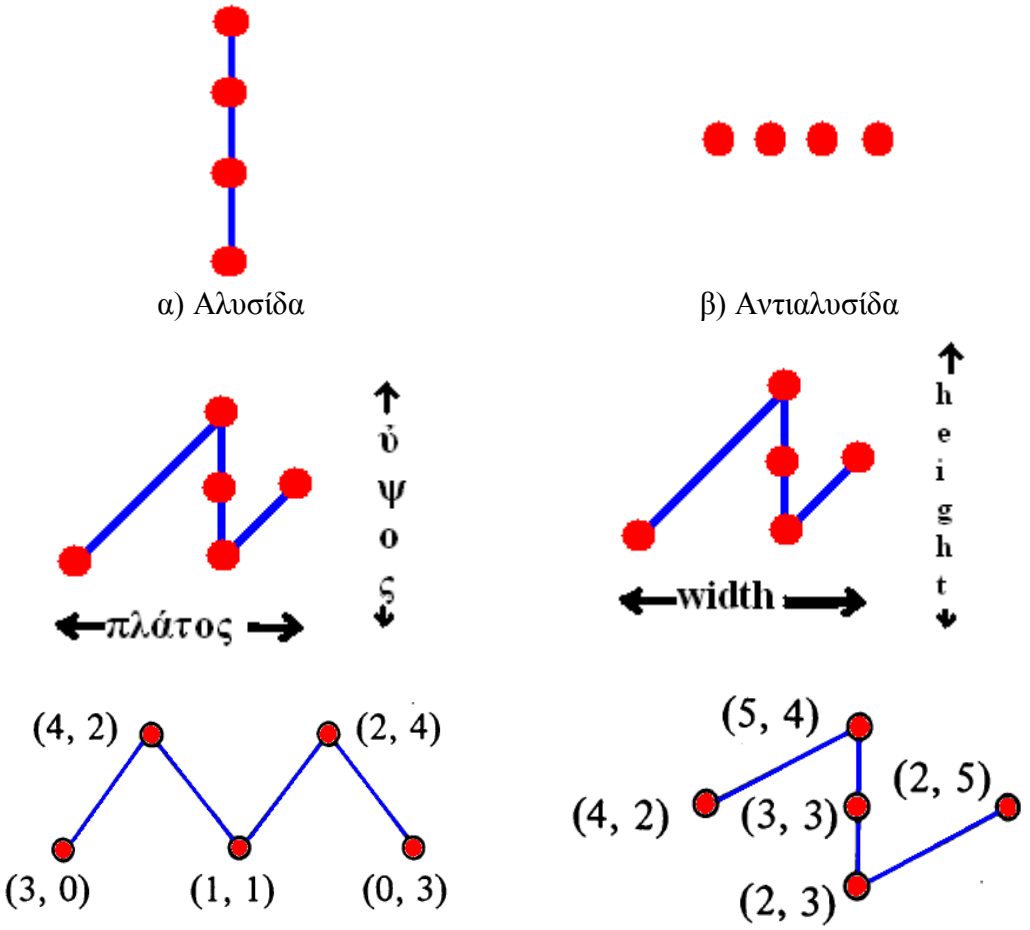
Εάν $X - x$ δεν είναι μία αλυσίδα τότε $\dim X - x \geq 2$, έστω $L_1, L_2, \dots, L_t$ είναι γραμμική επέκταση του $P|X - x| = P'$ του οποίου η διατομή είναι P'. Στο $L_1, L_2, \dots, L_{t-1}$ εισάγεται το y άμεσα πάνω από το x, και L_{t-1} εισάγεται το y άμεσα κάτω από το x. Το αποτέλεσμα της γραμμικής επέκτασης του P τέμνεται και δίνει το P και αυτό $\dim X \leq \dim(X - x)$. Εμείς σημειώνουμε ότι εάν $X - x$ είναι μία αλυσίδα, τότε $\dim X - x = \dim X - y = 1$, αλλά $\dim X = 2$. □

Λήμμα :

Υποθέτουμε $x > y$ μέσα στο P αλλά για κάθε $z \in X - x, y, z > x$ εάν $z > y$ και $z < x$ εάν $z < y$. Τότε $\dim X = \dim X - x = \dim(X - y)$.

Λήμμα :

Εάν το $X - A = 2$ τότε $\dim X = 2$, όπου το A είναι αντιαλυσίδα.



Σχήμα 2.23 : Εμπνευσμένο από Inequalities in dimension theory for Posets.

Θεώρημα : Εάν το $X - A \geq 2$ τότε $\dim X \leq |X - A|$

Θεώρημα : $\dim X \leq 2 \cdot width\ X - E + 1$

Θεώρημα : $\dim X \leq 2 \cdot width\ X - A + 1$

Θεώρημα : Για κάθε $n \geq 3$ έχουμε $\dim S_n \text{ x } S_n = 2n - 2$

Θεώρημα (Dushnik & Miller (1941)) :

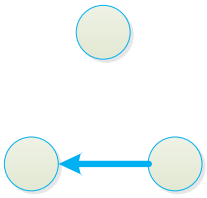
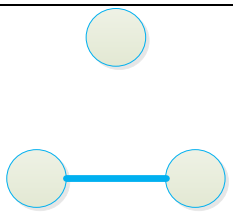
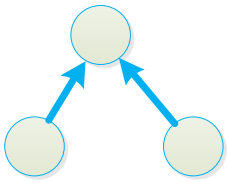
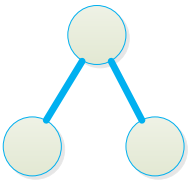
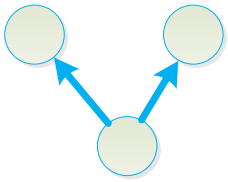
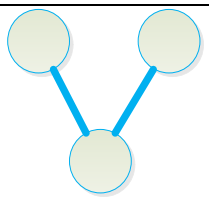
$\dim P \leq 2 \rightarrow P\ is\ Circular\ Representable$

$\dim P \leq 2 \rightarrow P$ *είναι κυκλικά αναπαραστάσιμο*

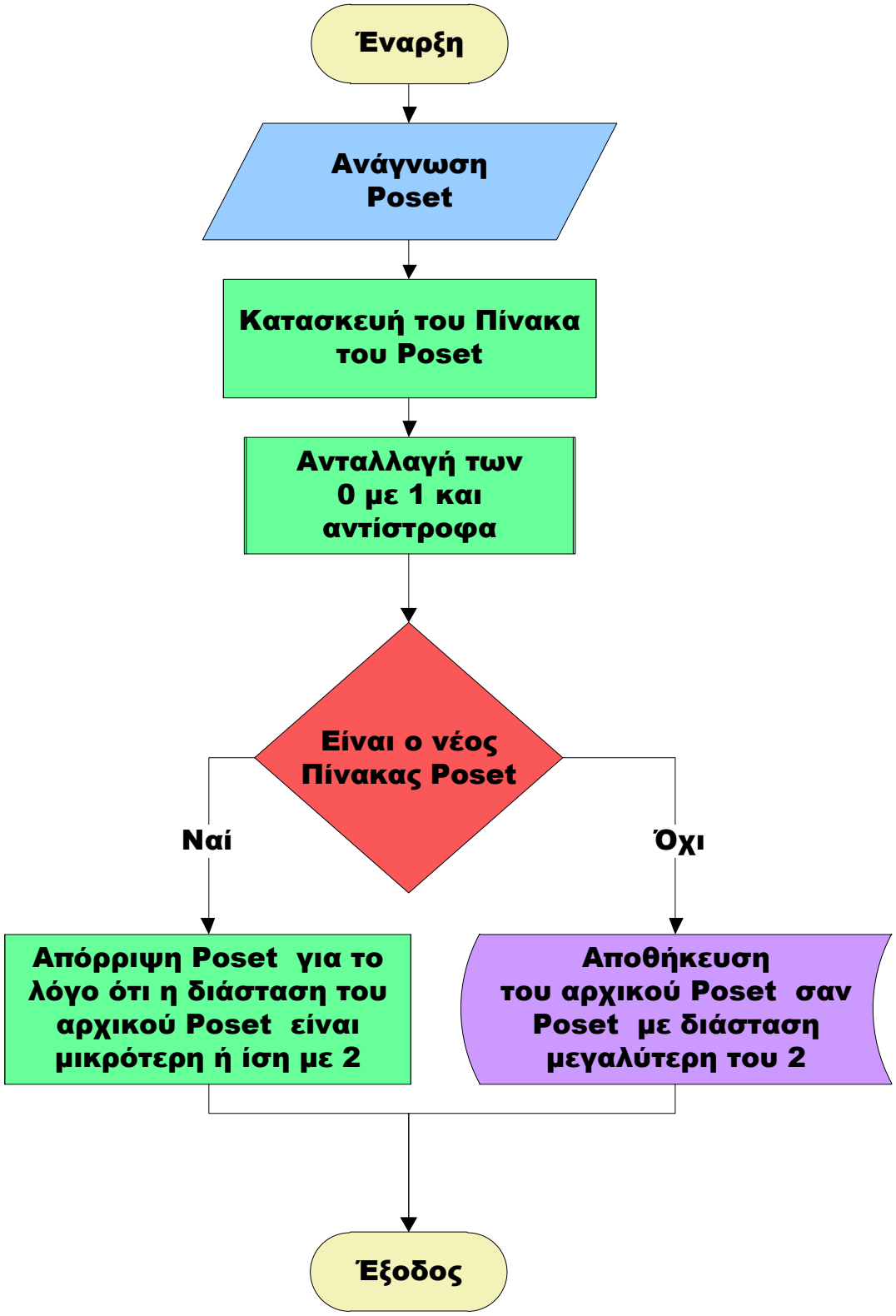
Μέσω αυτού του θεωρήματος ξέρουμε ότι όταν το $\dim P \leq 2$ μπορεί να αναπαρασταθεί με κύκλους.

Θεώρημα :

Εάν το συμπληρωματικό (Conjugate) ενός Poset είναι Poset τότε και μόνο τότε το Poset έχει διάσταση δύο. Συμπληρωματικό εννοούμε όταν αλλάζουμε τα 0 με τα 1. $\dim P \leq 2 \quad \exists Conj(P)$

	Διάγραμμα Poset	*Πίνακας Πρόσπτωσης*	*Διάγραμμα Hasse*
Αρχικό P		0 0 0 1 0 0 0 0 0	
Conj(P)		0 0 1 0 0 1 0 0 0	
Conj(P)		0 0 0 0 0 0 1 1 0	

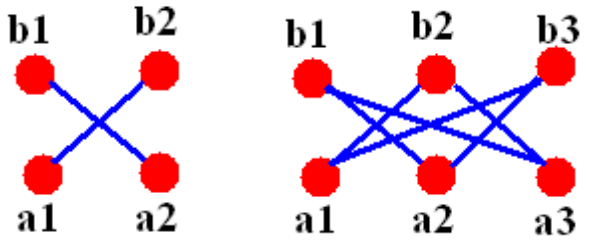
Σχήμα 2.24 : Παράδειγμα εφαρμογής Dushnik & Miller.



Διάγραμμα Ροής 2.1: Έλεγχος εάν ένα Poset έχει διάσταση μικρότερη ή ίση με 2.

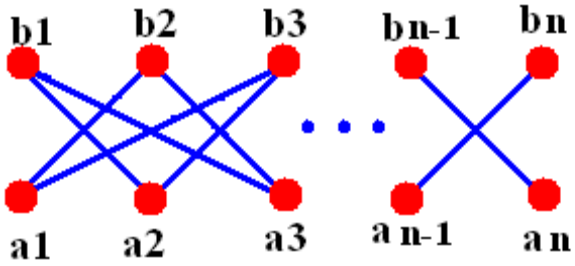
Standard n

Θεώρημα : Ένα Poset είναι Poset τμήματος εάν και μόνο εάν δεν εμπεριέχει το Standard S_2. (Fishburn) Το Standard S_n έχει διάσταση n, dim $S_n = n$.



Standard S_2 Standard S_3

Σχήμα 2.25 : Σχηματική αναπαράσταση των S_2 και S_3.



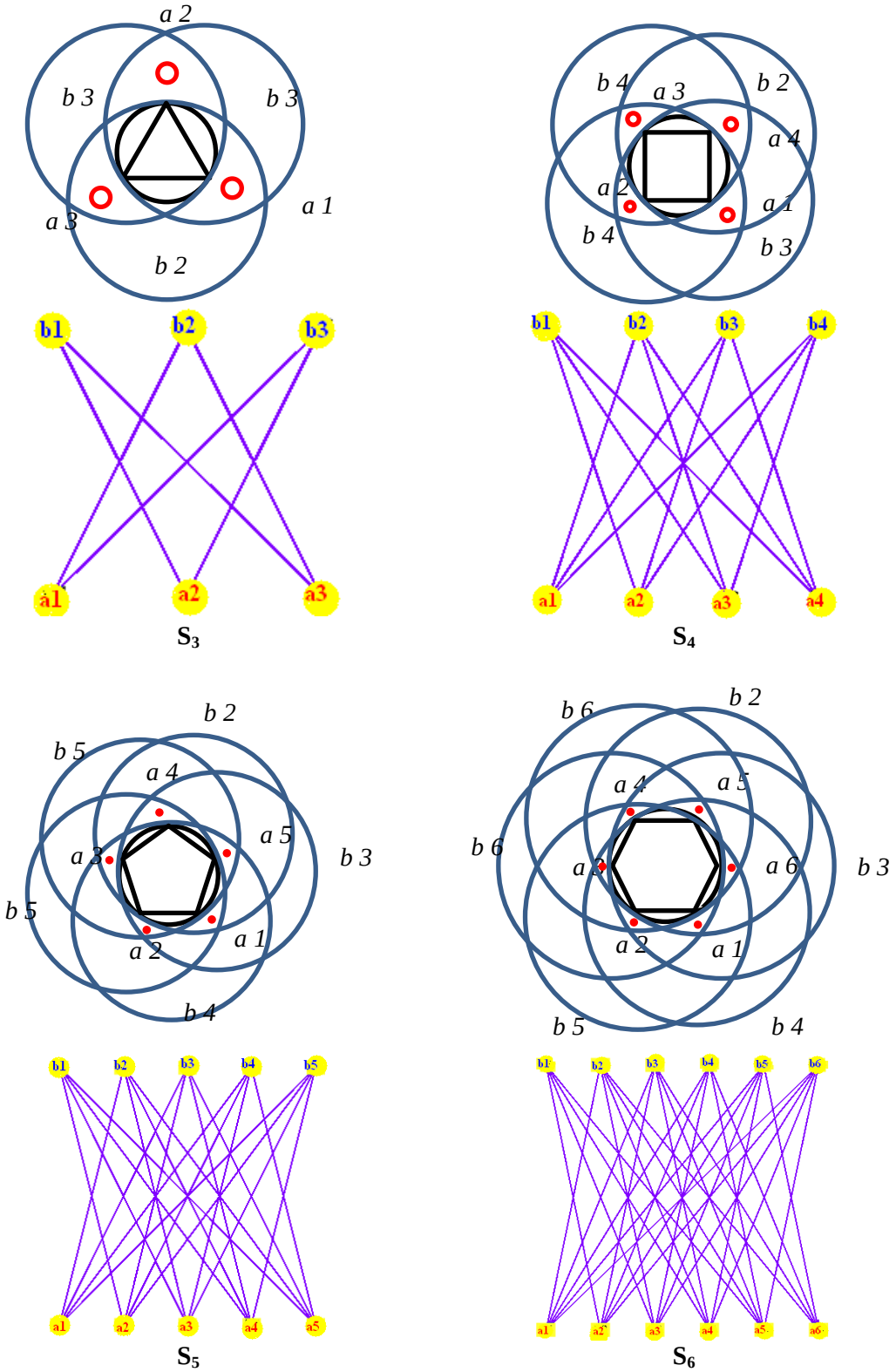
Σχήμα 2.26 : Standard n (S_n)

Το Standard n υλοποιείται ως εξής: Σχεδιάζουμε 2n κορυφές. Τις πρώτες n τις ονομάζουμε a_1 έως a_n και τις σχεδιάζουμε σε μία ευθεία. Τις υπόλοιπες n τις ονομάζουμε από b_1 έως b_n και τις σχεδιάζουμε σε μία ευθεία παράλληλη της προηγούμενης έτσι ώστε τα a_1 και b_1 να είναι στην κοινή κάθετη. Έπειτα ενώνουμε τις κορυφές ως ακολούθως: Κάθε a_i όπου $i \in 1,n$ συνδέεται με όλα τα b_k όπου το $k \in 1,n \setminus \{i\}$.

Θεώρημα : Κάθε S_n έχει κυκλική αναπαράσταση. (Ν.Λυγερός)

Απόδειξη :

Έστω ένα κανονικό πολύγωνο (τάξης *n*) εγγεγραμμένο, σ' ένα κύκλο ακτίνας ρ. Από κάθε κορυφή, κατασκευάζουμε κύκλο ακτίνας 2ρ. Κάθε νέος κύκλος εφάπτεται στον αρχικό. Μετά, αν θεωρήσουμε τον άξονα που δημιουργούν τα κέντρα του αρχικού κύκλου και ενός κύκλου με διπλή ακτίνα τότε τα σημεία που βρίσκονται εκτός του αρχικού κύκλου και εκτός του δεδομένου κύκλου, κοντά στο σημείο που εφάπτονται οι δύο κύκλοι, ανήκουν σε όλους τους άλλους κύκλους. □

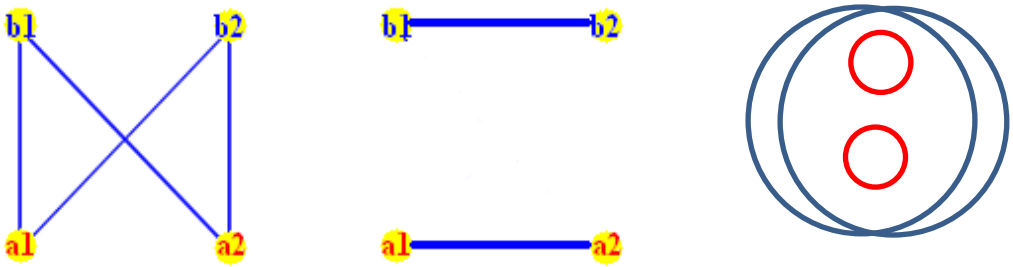


Σχήμα 2.27 : Κυκλική αναπαράσταση των S_3,S_4,S_5,S_6

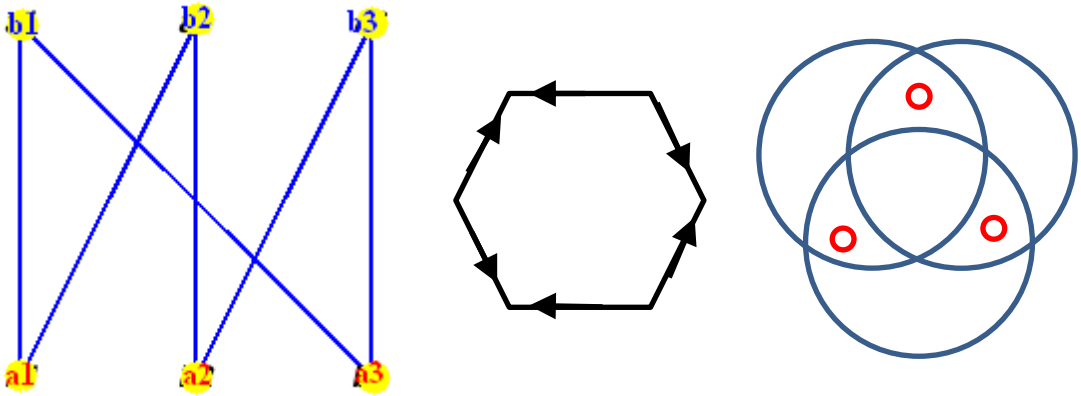
Κορώνες

Ορισμός :

Κορώνες ονομάζονται τα Poset τα οποία έχουν την εξής συνδεσμολογία: Έχουν ύψος 2 και τα a_i ενώνονται με τα b_i και b_{i+1}. Το τελευταίο a_n συνδέεται πάνω στο b_1.

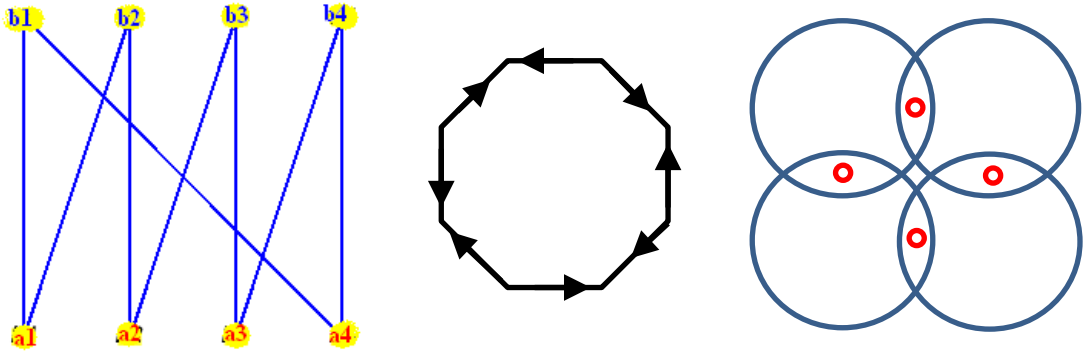


Σχήμα 2.28 : Διάγραμμα C_4 , Conjugate του C_4 και κυκλική αναπαράστασή του.



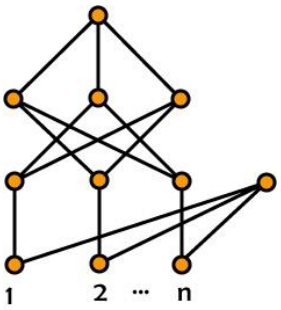
Σχήμα 2.29 : Διάγραμμα C_6 και κυκλική αναπαράστασή του.

1) dim C_6 $\leq width\ C_6$ $= 3$ (Hiraguchi)
2) dim C_6 $\leq \frac{C_6}{2} = 3$ (Hiraguchi)
3) dim C_6 $= 3$ (Chaunier & Lygeros 1991)
4) Εδώ αξίζει να σημειωθεί ότι το $C_6 \cong S_3$ δηλαδή το C_6 είναι ισόμορφο με το S_3.



Σχήμα 2.30 : Διάγραμμα C_8 και κυκλική αναπαράστασή του.

Κρισιμότητα των Posets



Διάγραμμα 2.6: Οικογένεια κρίσιμων Posets.
Πηγή : Trotter

Αυτά τα Posets έχουν πάχος και διάσταση n+1 και αν βγάλουμε οποιοδήποτε στοιχείο τότε η διάσταση γίνεται n.

Θεώρημα :

Έστω $P = (X, <)$ ένα poset όπου X είναι αριθμήσιμο. Τότε $dim\ P \leq 2$ εάν και μόνο εάν P είναι s-αναπαραστάσιμο όταν s είναι μία οικογένεια από κλειστά και φραγμένα τμήματα του **R**.

Θεώρημα :

Εάν $n \geq 4$ και $P \leq 2 \cdot n + 1$, τότε $\dim P < n$ εκτός και εάν υπάρχει το S_n. (Trotter)

Εφαρμογή :

Στην εικασία που θα ασχοληθούμε αργότερα θα μελετήσουμε τι γίνεται με το $N = 11$. Για αυτό καλό θα ήταν να δούμε την εφαρμογή του παραπάνω θεωρήματος. $N = 2 \cdot n + 1$. $P = N = 11 \rightarrow n = 5$. Συμπεραίνουμε, δηλαδή, ότι η διάσταση του P είναι μικρότερη του πέντε, $\dim P < 5$, εκτός και εάν περιέχει το S_5.

Θεώρημα :

Εάν το Hasse διάγραμμα του P είναι δέντρο, τότε $\dim P \leq 3$. (Trotter και Moore)

Θεώρημα :

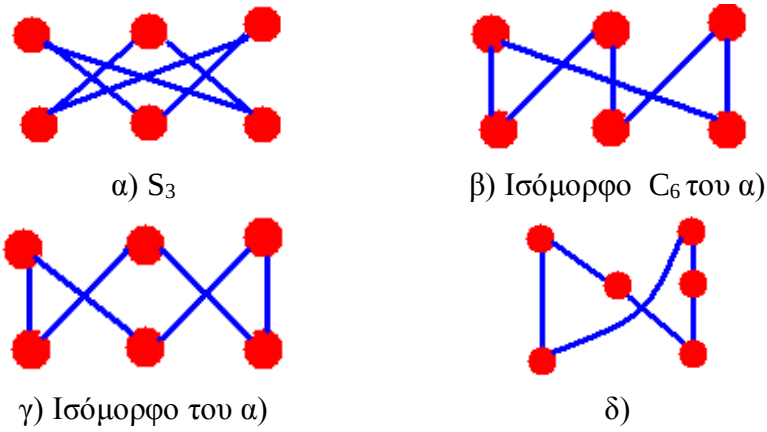
Εάν το διάγραμμα του P είναι επίπεδο, και P έχει και τα δύο στοιχεία μικρότερο και μεγαλύτερο τότε $dim\ P \leq 2$.

Ανοικτό Πρόβλημα 1971 :

- Δεν είναι γνωστό εάν <u>κάθε</u> Poset εμπεριέχει ένα ζευγάρι σημείων το οποίο αν αφαιρεθεί κατεβάζει τη διάσταση κατά μία μονάδα τουλάχιστο. Εκτός από αλυσίδα και αντιαλυσίδα.
  Εάν $P \geq 3$, $\dim P \leq 1 + \dim(P - x - y)$.

Παρατήρηση:

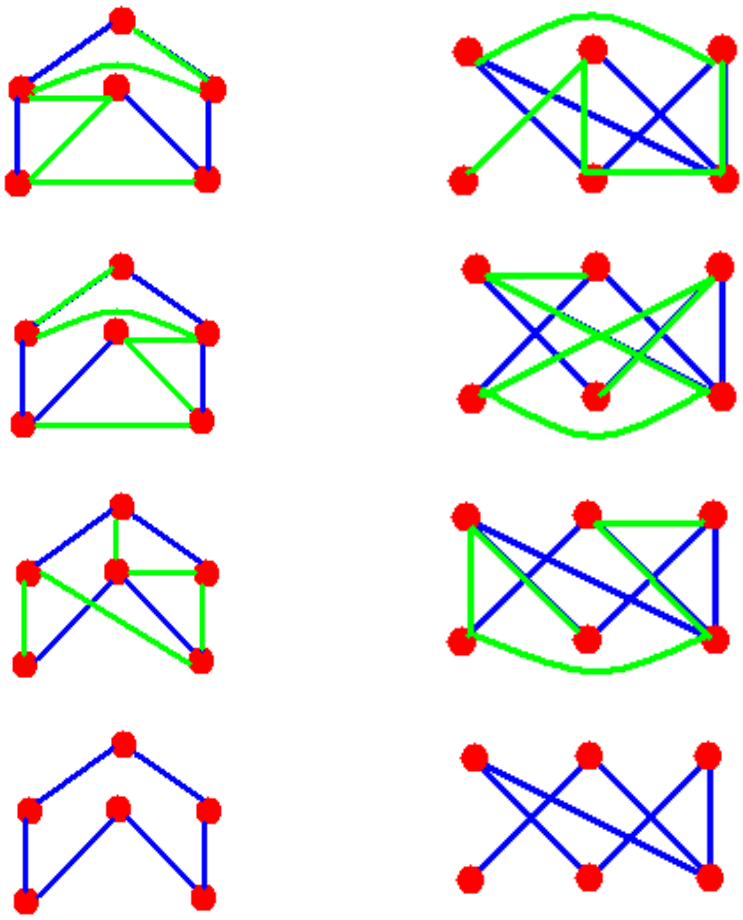
Το N = 6 είναι σημαντικό γιατί όλα όσα είναι διάστασης τρία $\dim P = 3$ είναι κρίσιμα. Το N = 8 είναι σημαντικό. Γιατί όλα όσα είναι διάστασης τέσσερα $\dim P = 4$ είναι κρίσιμα. Ιδιαίτερο ενδιαφέρον είναι και το γεγονός ότι στο N = 8 υπάρχει μόνο ένα κρίσιμο και αυτό είναι το S_4.



α) S_3 β) Ισόμορφο C_6 του α)

γ) Ισόμορφο του α) δ)

Σχήμα 2.31 : Κρίσιμα Poset N = 6



Θεώρημα : Έστω P, poset. Αν $\nexists Conj\ (P)$ και $width\ P\ \leq 3$ τότε dim $P\ = 3$

Απόδειξη : 1 $\nexists\ Conj\ P$ dim $P\ > 2$

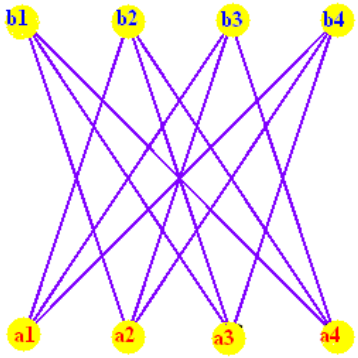
2 dim $P\ \leq width\ P\ = 3$

1 και 2 dim $P\ = 3$ □

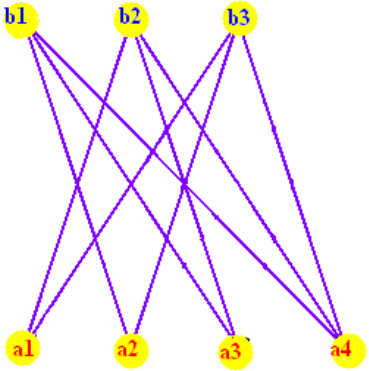
Λήμμα : Το θεώρημα είναι βέλτιστο.

Απόδειξη :

Έστω ένα poset με 8 στοιχεία και διάστασης 4. Λόγω του θεωρήματος του Hiraguchi dim $P\ \leq \frac{P}{2}$ αυτό το Poset είναι κρίσιμο με την έννοια της διάστασης. Έστω S_4, έχουμε προφανώς width(S_4) = 4.



Κατά τον Trotter αυτό το poset έχει διάσταση 4 και ξέρουμε ότι είναι κρίσιμο διότι έχει 8 στοιχεία. Ας αποδείξουμε μέσω των αλυσίδων ότι έχει διάσταση 3.



Αλυσίδες :

$$\mathbf{a_2 < a_3 < a_4 < b_1 < a_1 < b_2 < b_3}$$

$$\mathbf{a_1 < a_3 < a_4 < b_2 < a_2 < b_1 < b_3}$$

$$\mathbf{a_1 < a_2 < a_4 < b_3 < a_3 < b_1 < b_2}$$

Αυτό το Poset έχει διάσταση 3.

Άρα : $\exists P \mid$ dim $P\ = 3$ και $width\ P\ = 4$

Κατά συνέπεια το θεώρημα είναι βέλτιστο. □

Απαρίθμηση – Enumeration

Κόμβοι – N : Ο αριθμός των κόμβων που υπάρχουν στο γράφο.

Σχέσεις – Relation: Ο αριθμός των ακμών – συνδέσμων που θα υπάρχουν στο γράφο. Ο μέγιστος αριθμός ακμών που μπορούν να υπάρξουν σε ένα Poset είναι $Relation = \frac{N \cdot (N-1)}{2}$.

Ο αριθμός των διαφορετικών Poset που μπορούν να υπάρξουν για ένα δοσμένο N είναι αποτέλεσμα συνδυαστικής και δίνεται από την σχέση $P_N = \sum_{R=0}^{\frac{N\ N-1}{2}} P_N^R$. Υπάρχουν ειδικές περιπτώσεις που ξέρουμε αμέσως τον αριθμό των Poset που υπάρχουν όπως για παράδειγμα:

- $Relation\ 0 \rightarrow 1\ Poset$
- $Relation\ 1 \rightarrow 1\ Poset$
- $Relation \frac{N\ N-1}{2} - 1 \rightarrow (N-1)\ Poset$
- $Relation \frac{N\ N-1}{2} \rightarrow 1\ Poset$
- Γενικότερα μπορούμε να υπολογίσουμε το κίτρινο τρίγωνο.

N	Μέγιστες Σχέσεις -												
	0	1	2	3	4	5	6	7	8	9	10	11	12
0	1												
1	1												
2	1	1											
3	1	2	1										
4	1	3	3	4	3	1	1						
5	1	4	6	9	12	10	10	6	3	1	1		
6	1	5	10	17	28	35	44	46	43	36	25	16	7
7	1	6	15	29	54	83	123	168	204	239	249	243	209
8	1	7	21	46	94	166	274	434	629	874	1136	1402	1621
9	1	8	28	69	153	300	541	939	1528	2386	3551	5054	6897
10	1	9	36	99	237	506	986	1836	3240	5492	8974	14096	21414
11	1	10	45	137	353	811	1694	3351	6306	11385	19897	33614	55127

Πίνακας 2.1 : Κατανομή των Posets

Εικασία Fraïssé : Η κατανομή των Posets αναφορικά με τον αριθμό των σχέσεών τους (relations) παρουσιάζει μια μοναδική κορυφή.

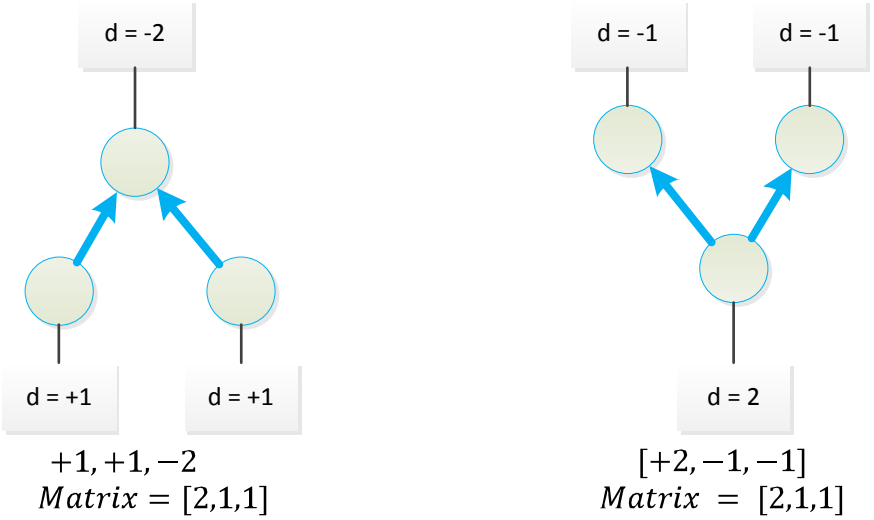
Αυτό έχει ελεγχθεί υπολογιστικά για N ≤ 16.

Φιλτράρισμα – Filter : Είναι η διαδικασία κατά την οποία τα Posets βαθμού μικρότερου και ίσου του δύο απαλείφονται. Σύμφωνα με το θεώρημα Dushinik & Miller όλα τα Posets με διάσταση μικρότερη και ίση του δύο είναι κυκλικά αναπαραστάσιμα. Επίσης στο φιλτράρισμα αφαιρούνται όλα τα μη συνεκτικά Posets δια το λόγο ότι έχουν λυθεί για μικρότερα N.

Έλεγχος $\dim(P) \leq 2$: Ο έλεγχος του αν ένα Poset έχει διάσταση μικρότερη ή ίση του δύο αρκεί να αποδειχτεί ότι το συμπληρωματικό (conjugate) είναι Poset.

Βαθμός – Degree : Ο βαθμός ενός κόμβου είναι το άθροισμα των εξερχόμενων ακμών με βάρος +1 και το άθροισμα των εισερχόμενων ακμών με βάρος -1.

Πίνακας Βαθμών – Matrix : Ο πίνακας βαθμών ή πίνακας είναι ο πίνακας με τους βαθμούς των κόμβων κατά απόλυτο τιμή και σε φθίνουσα σειρά.



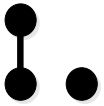
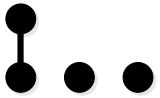
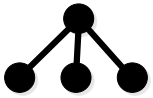
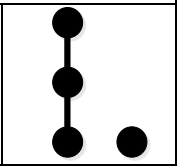
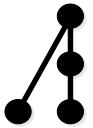
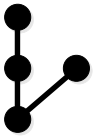
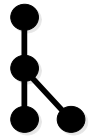
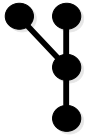
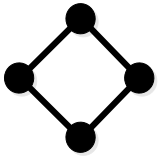
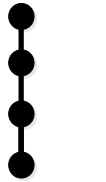
$+1, +1, -2$
$Matrix = [2,1,1]$

$[+2, -1, -1]$
$Matrix = [2,1,1]$

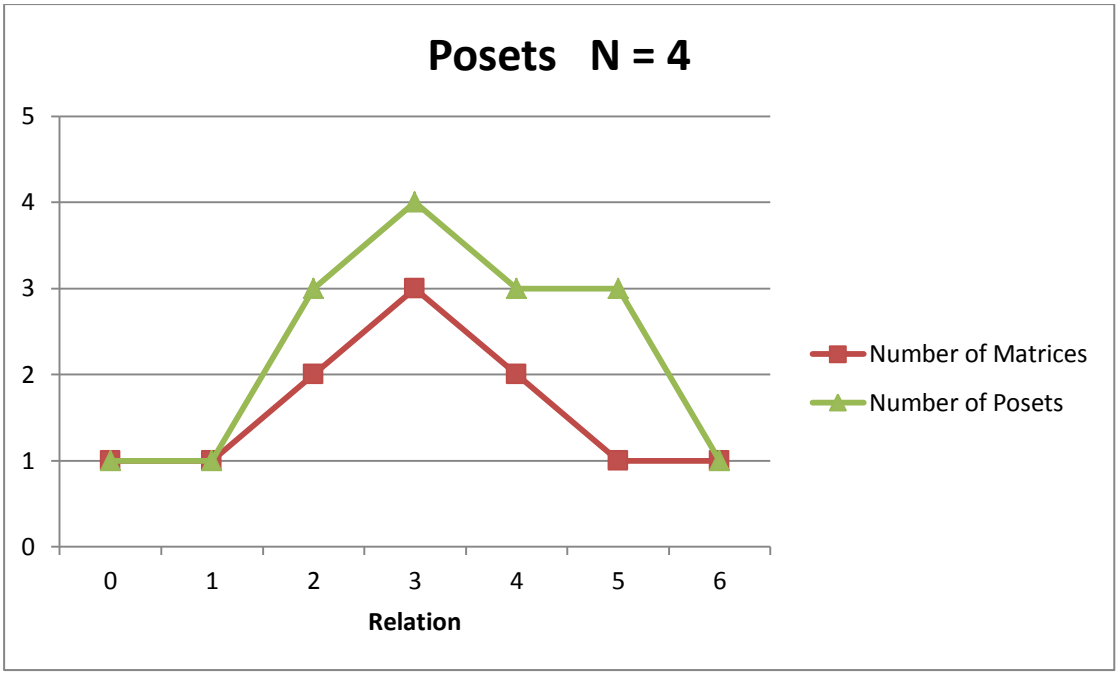
Σχήμα 2.32 : Αναπαράσταση P_3 με Relation 2. Παρατήρηση τα δύο Poset είναι δυϊκά – Duals

.

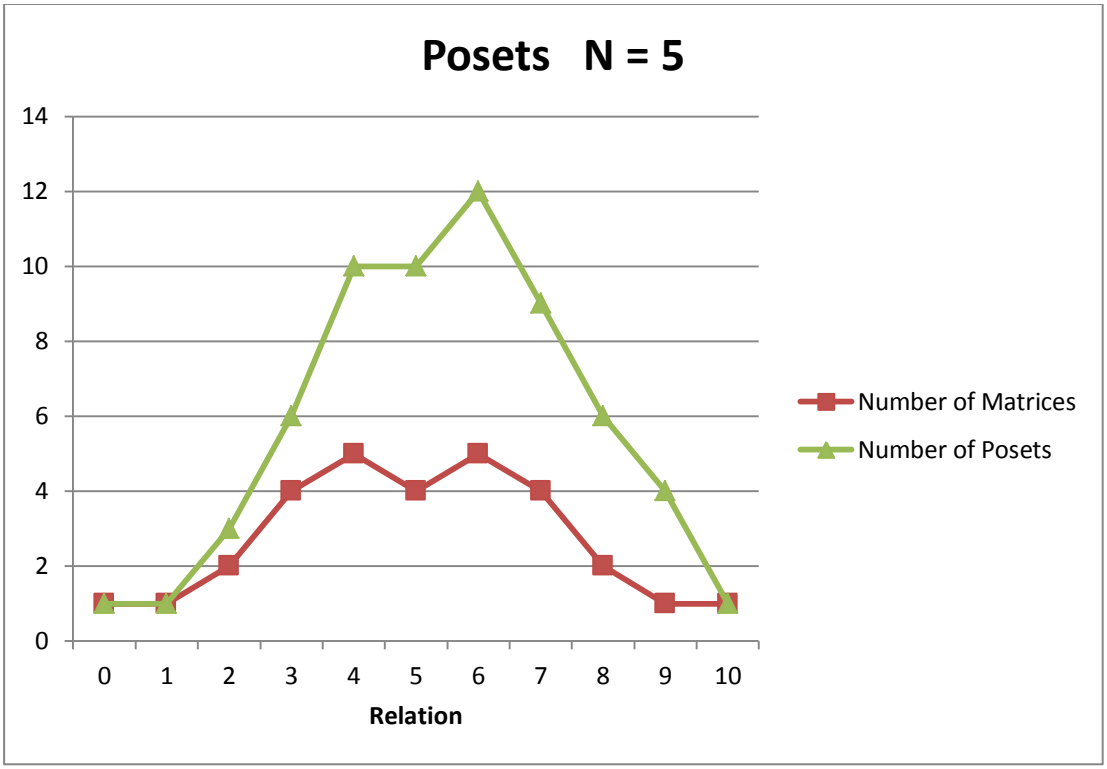
Δυϊκά – Duals : Τα δυϊκά Posets προέρχονται από την αλλαγή, την καθολική αλλαγή, των βελών κατεύθυνσης. Συνέπεια του γεγονότος αυτού τα Posets αυτά έχουν τον ίδιο Matrix. Η ιδιότητα αυτή είναι πολύ χρήσιμη κατά την παραλληλοποίηση του αλγορίθμου εύρεσης δυικών γιατί όσα είναι δυικά θα βρίσκονται στην ίδια κατηγορία.

N	Relations			
	0	1	2	3
1				
2				
	Αυτοδυικό	Αυτοδυικό		
3				
	Αυτοδυικό	Αυτοδυικό	Δυικά	Αυτοδυικό
4				
	Αυτοδυικό	Αυτοδυικό	Αυτοδυικό	Δυικό
				
	Δυικό	Αυτοδυικό	Αυτοδυικό	
	Relations			
	4		**5**	**6**
							
	Δυικά		Αυτοδυικό	Δυικά		Αυτοδυικό	Αυτοδυικό

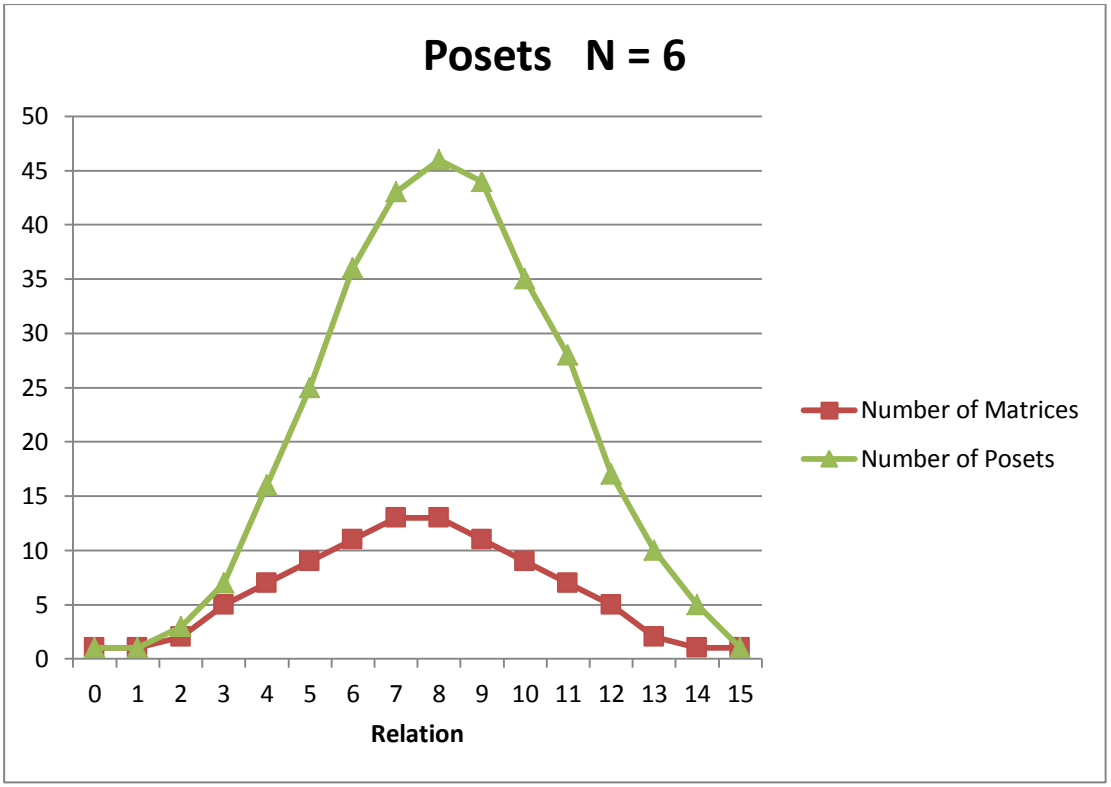
Σχήμα 2.33 : Διαγράμματα Hasse των posets τάξης μικρότερης ή ίσης του τέσσερα.



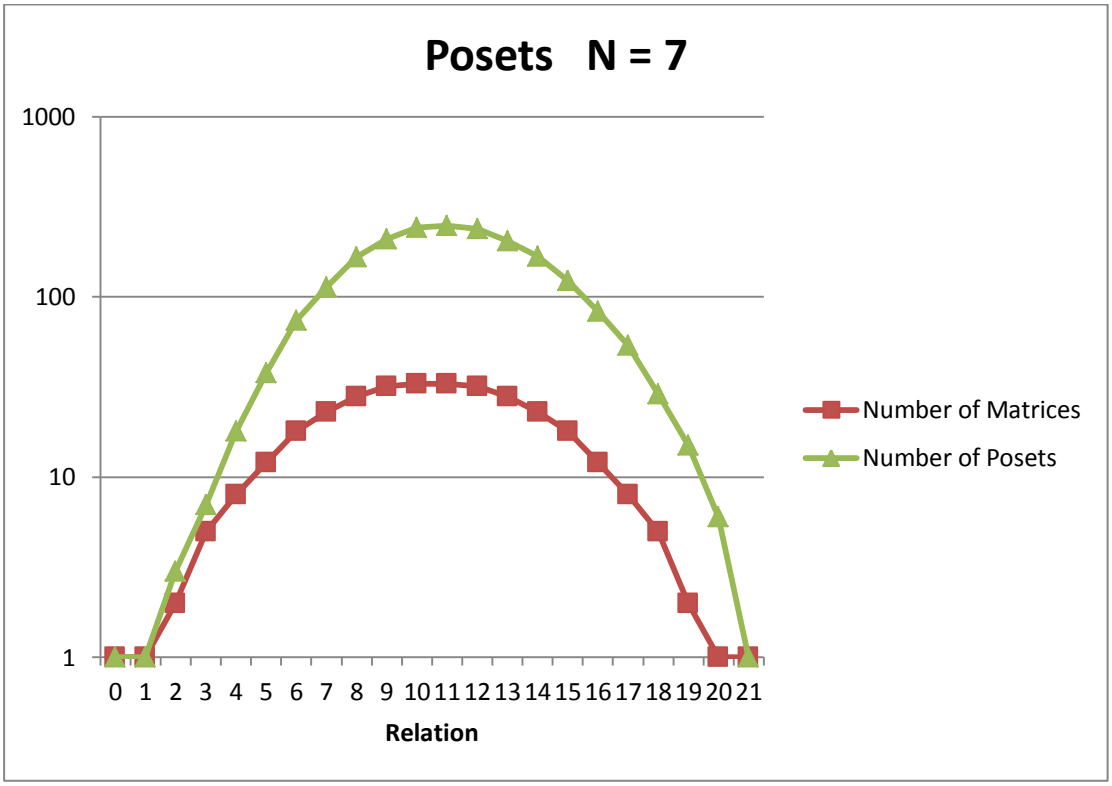
Πλήθος Πινάκων : 11
Πλήθος Posets : 16



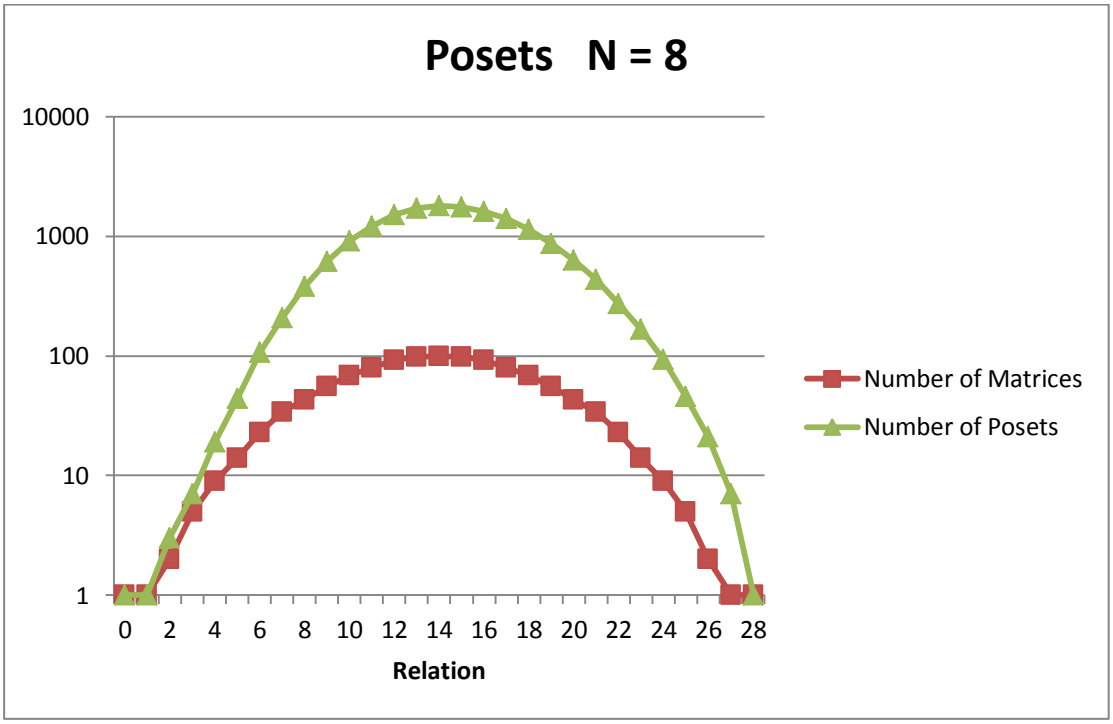
Πλήθος Πινάκων : 30
Πλήθος Posets : 63



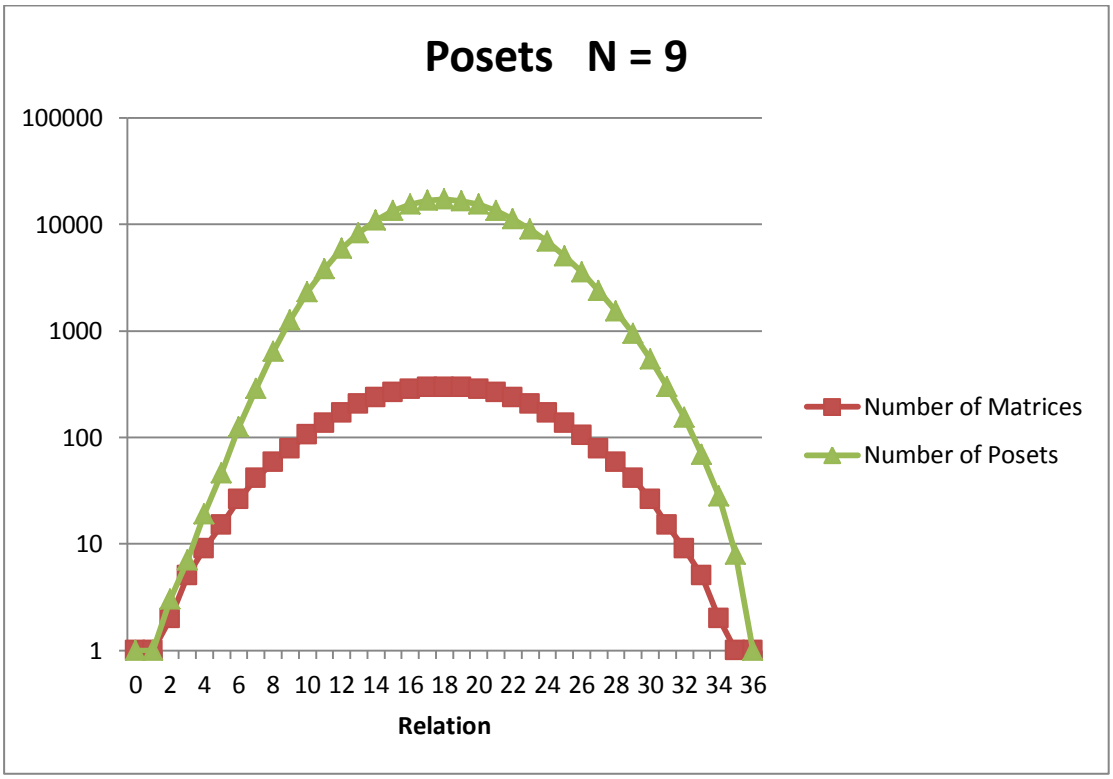
Πλήθος Πινάκων : 98
Πλήθος Posets : 318



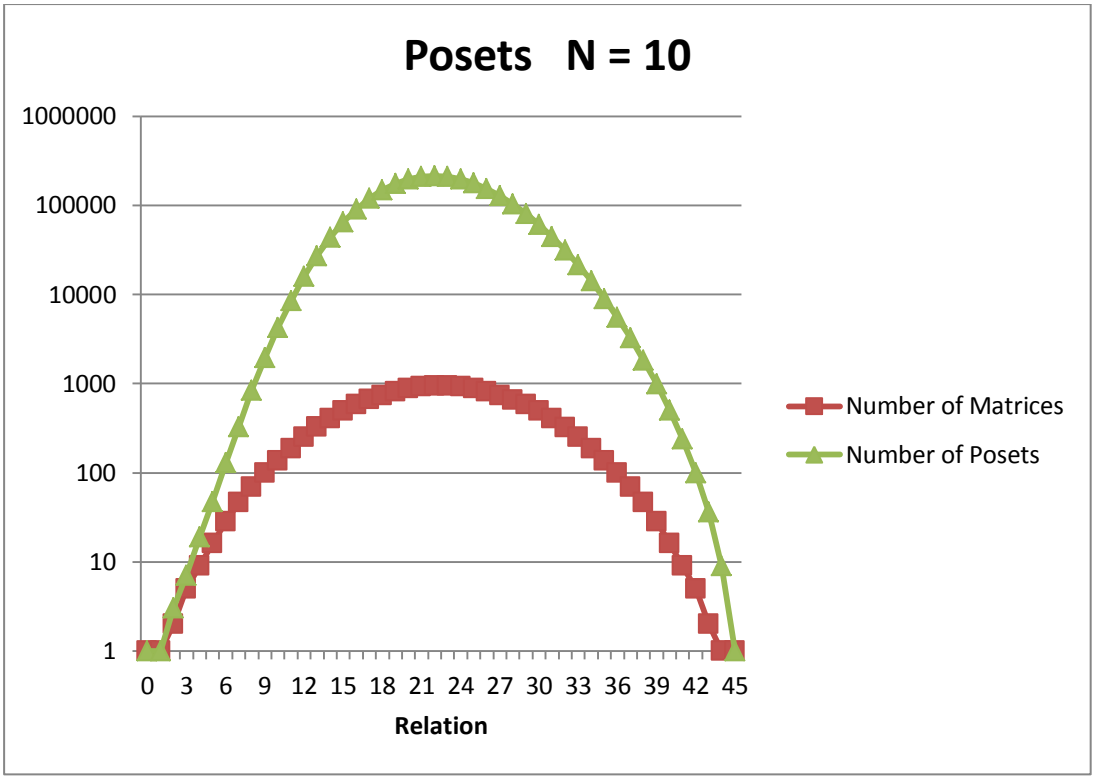
Πλήθος Πινάκων : 326
Πλήθος Posets : 2045



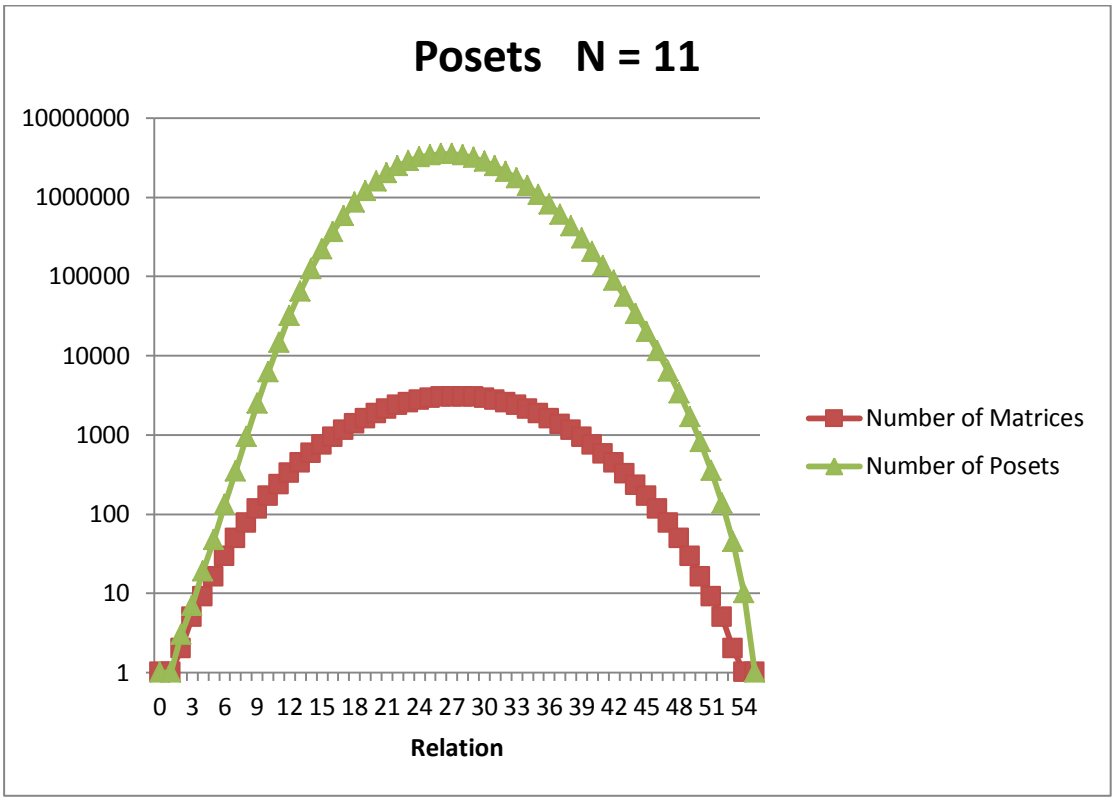
Πλήθος Πινάκων : 1154
Πλήθος Posets : 16999



Πλήθος Πινάκων : 4142
Πλήθος Posets : 183231



Πλήθος Πινάκων : 15207
Πλήθος Posets : 257284



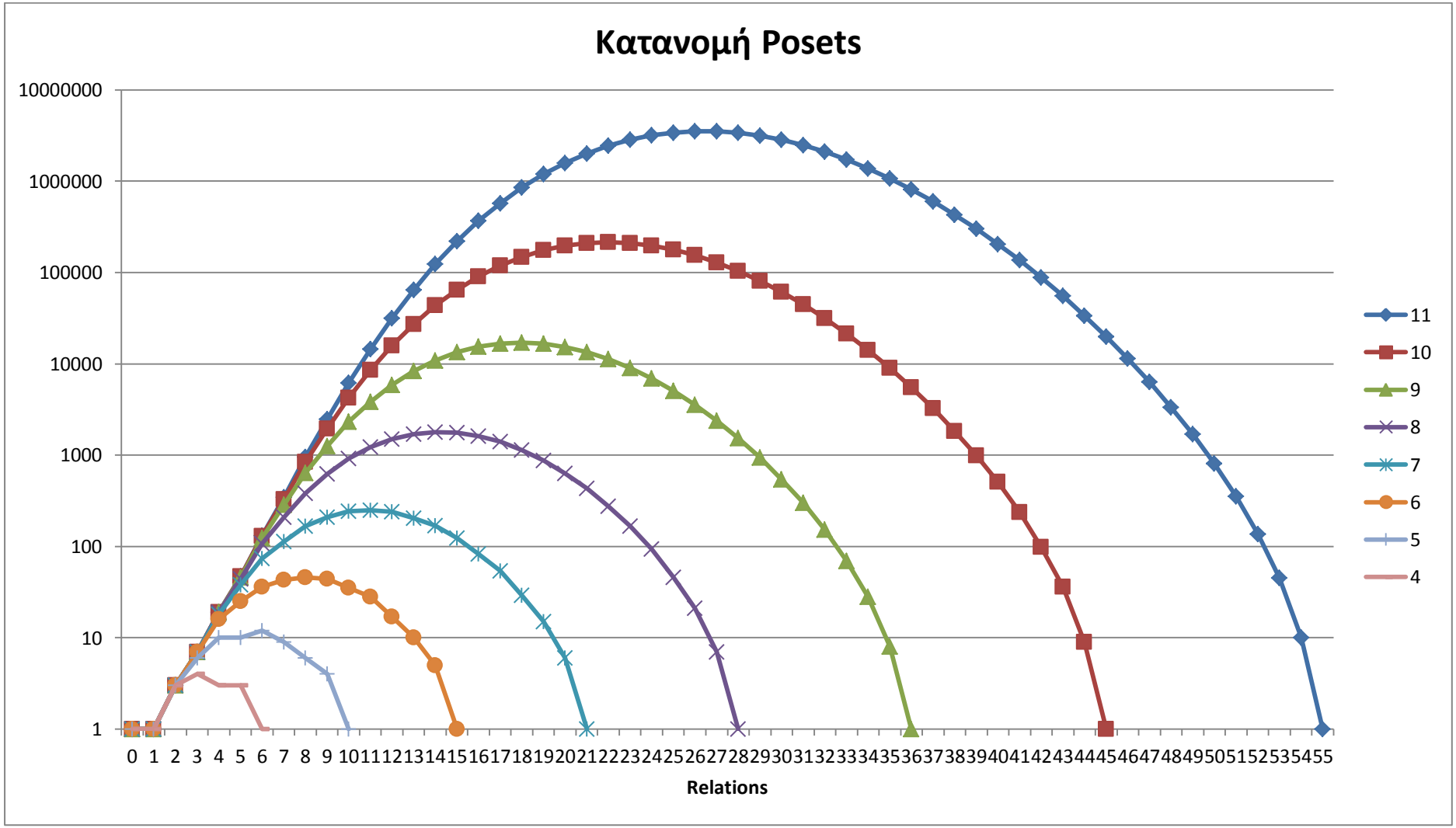
Πλήθος Πινάκων : 56375
Πλήθος Posets : 46749427

N = 11		
Relations	**Number of Matrices**	**Number of Posets**
0	1	1
1	1	1
2	2	3
3	5	7
4	9	19
5	16	47
6	29	132
7	49	346
8	76	950
9	115	2468
10	167	6171
11	232	14411
12	324	31724
13	439	64772
14	577	123620
15	742	219868
16	932	366672
17	1136	574347
18	1356	849968
19	1597	1190889
20	1842	1587016
21	2093	2015412
22	2330	2446957
23	2548	2844542
24	2742	3174558
25	2893	3405232
26	2971	3518608
27	3007	3505930
28	3006	3374784
29	2969	3141073
30	2880	2831400
31	2733	2473385
32	2547	2096755
33	2326	1725908
34	2085	1380922
35	1836	1074413
36	1585	813564
37	1349	599653
38	1132	430451
39	926	300981
40	739	204974
41	573	135976
42	437	87786
43	322	55127
44	230	33614
45	166	19897
46	115	11385
47	76	6306
48	49	3351
49	29	1694
50	16	811
51	9	353
52	5	137
53	2	45
54	1	10
55	1	1

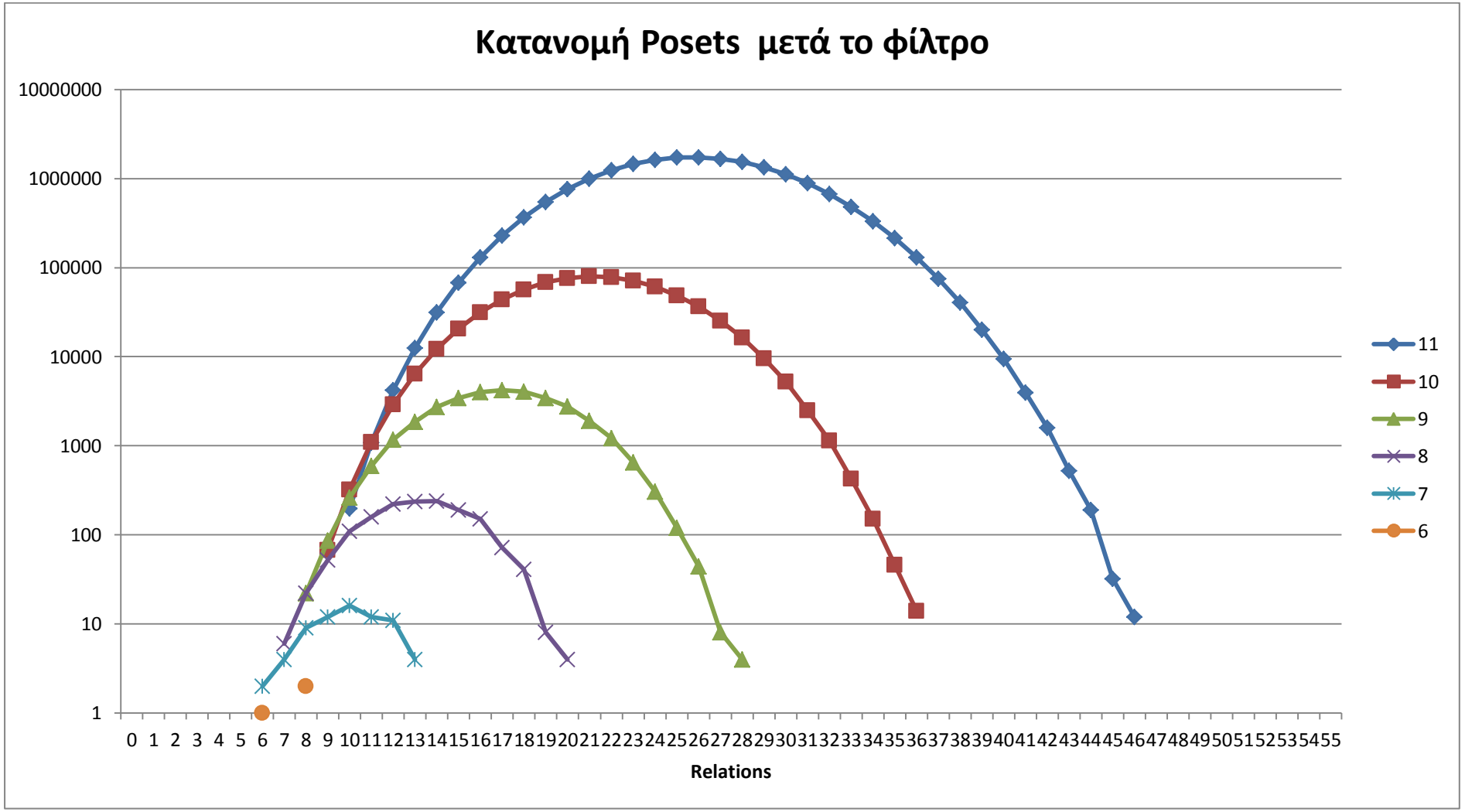
Πίνακας 2.2 : Πίνακας κατανομής Posets και των πινάκων τους για N = 11

N = 11		
Relations	**Number of Matrices**	**Number of Posets**
11	34	542
12	60	2130
13	89	6307
14	135	15802
15	185	33969
16	271	65544
17	364	113936
18	472	183158
19	580	272587
20	710	380791
21	836	499413
22	962	619792
23	1078	727301
24	1181	811482
25	1248	859796
26	1286	868669
27	1312	835253
28	1281	766836
29	1225	670222
30	1133	559473
31	1029	444079
32	901	336298
33	769	241564
34	652	165222
35	537	106656
36	424	65417
37	338	37484
38	267	20288
39	196	10105
40	141	4733
41	92	1982
42	59	797
43	37	261
44	18	96
45	5	16
46	1	6

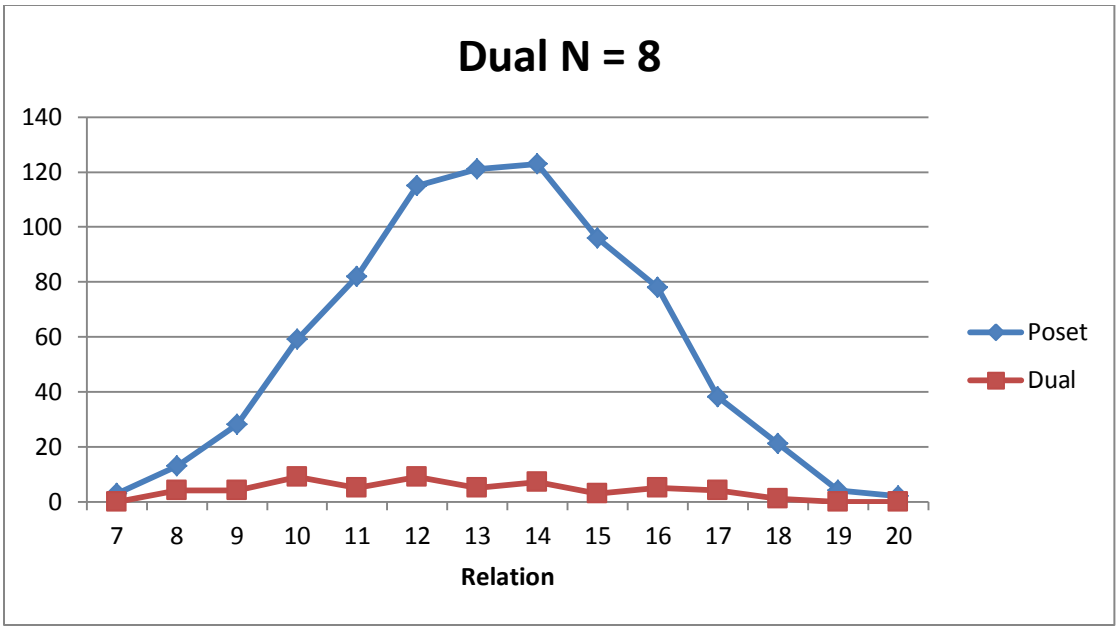
Πίνακας 2.3 **:** *Πίνακας κατανομής Posets και των πινάκων τους για N = 11 μετά από το φιλτράρισμα και την αφαίρεση των δυικών.*



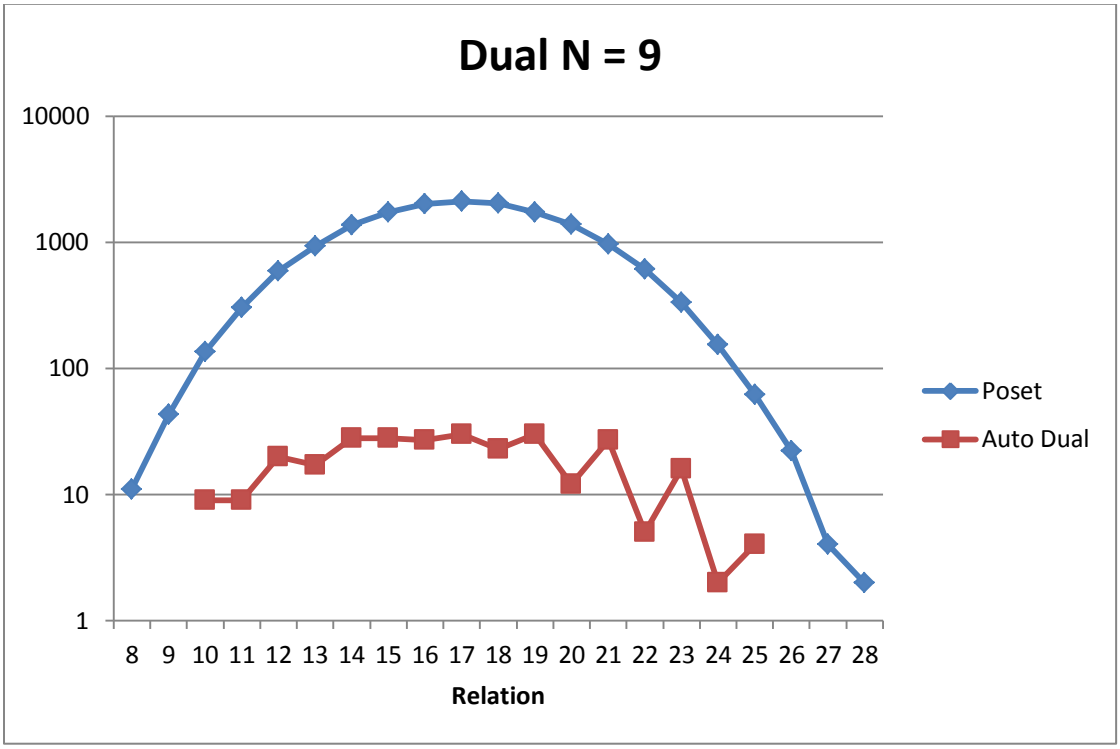
Σχήμα 2.34 : Γραφική Παράσταση με κατανομή ανά σχέση των αριθμών των Posets.



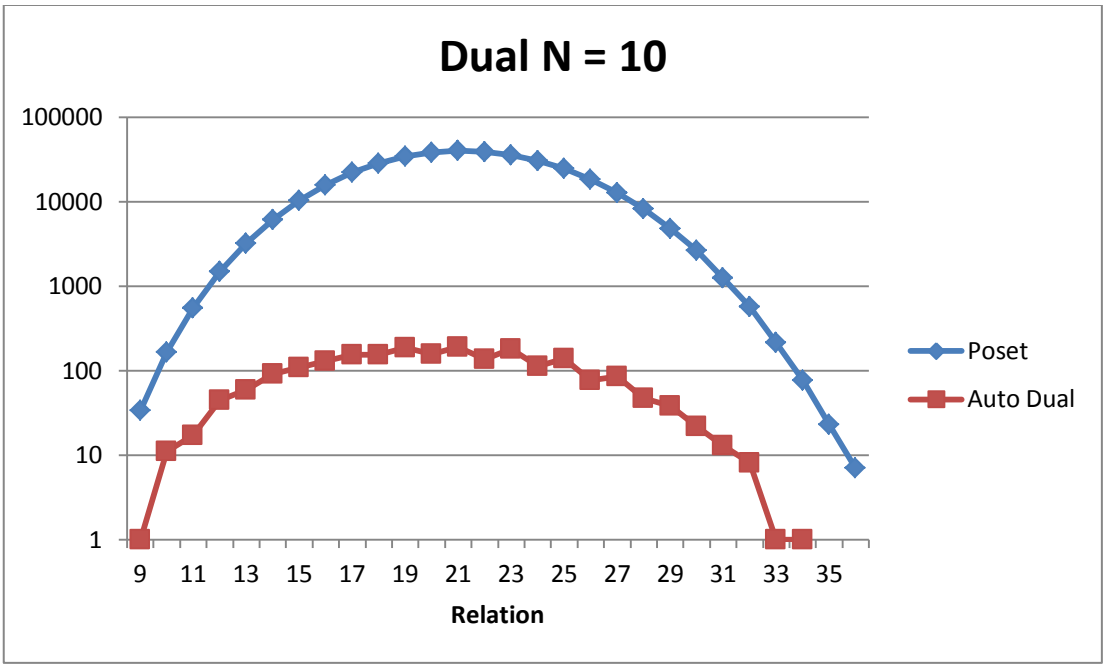
Σχήμα 2.35 : Γραφική Παράσταση με κατανομή ανά σχέση των αριθμών των Posets μετά από το φίλτρο.



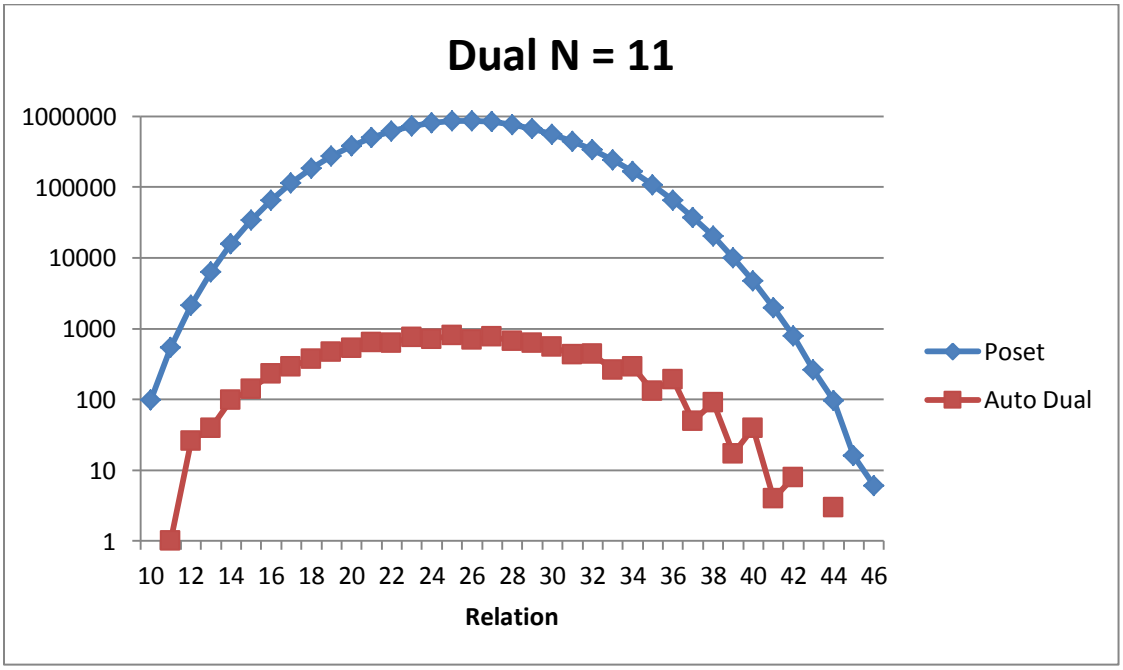
Πλήθος Πινάκων : 221
Πλήθος Auto Dual : 56
Πλήθος Posets : 783



Πλήθος Πινάκων : 1119
Πλήθος Auto Dual : 287
Πλήθος Posets : 16537



Πλήθος Πινάκων : 4859
Πλήθος Auto Dual : 2162
Πλήθος Posets : 379603



Πλήθος Πινάκων : 19925
Πλήθος Auto Dual : 11018
Πλήθος Posets : 9728106

N	Αρχικά P	$F\ (P)$	$\frac{F\ (P)}{Αρχικά\ P}$	$Dual(F(P))$	$\frac{\text{Dual(F(P))}}{F(P)}$	$\frac{\text{Dual(F(P))}}{\text{Αρχικά P}}$
1	1	0	0,00 %	-	-	-
2	2	0	0,00 %	-	-	-
3	5	0	0,00 %	-	-	-
4	16	0	0,00 %	-	-	-
5	63	0	0,00 %	-	-	-
6	318	3	0,94 %	2	66,67 %	0,62 %
7	2045	70	3,42 %	39	55,71 %	1,90 %
8	16999	1510	8,88 %	783	51,85 %	4,60 %
9	183231	32787	17,89 %	16537	50,43 %	9,02 %
10	2567284	757044	29,48 %	379603	50,14 %	14,78 %
11	46749427	19445194	41,59 %	9728106	50,02 %	20,80 %

Πίνακας 2.4: Παρουσίαση αριθμού Posets κατά τις διάφορες φάσεις επεξεργασίας $F\ P\ = Filter(P)$. *Τα ποσοστά αντιπροσωπεύουν τα Posets που έχουν περάσει τους ελέγχους του φίλτρου και του δυικού.*

Παλαιότερες Εργασίες

Ο Wierman και ο Schneinerman απέδειξαν ότι το $\mathbf{Z}^3$ δεν είναι κυκλικά αναπαραστάσιμο.

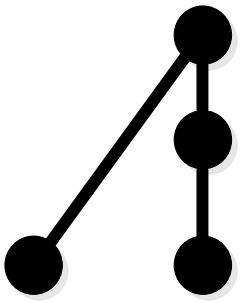
Ο G.Hulbert απέδειξε το 1988 (Order, 5,1988 (235-237)) ότι το $\mathbf{N}^3$ δεν είναι κυκλικά αναπαραστάσιμο.

Οποιαδήποτε μέθοδος που χρησιμοποιεί τη διάσταση σαν τελικό αποτέλεσμα δεν μπορεί να αποδείξει την εικασία SSU.

Θεώρημα Urrutia : Όλα τα Posets διάστασης μικρότερης και ίσης του τρία είναι αναπαραστάσιμα με κανονικά πολύγωνα.
$\forall Poset, \dim P \leq 3 \leftrightarrow Representable\ by\ inclusion\ with\ Regular\ Polygons$

Ο Urrutia αφού απέδειξε ότι όλα τα πεπερασμένα Posets με διάσταση μέγιστη τρία, $\dim P \leq 3$ είναι αναπαραστάσιμα με κανονικά **Ν-γωνα,** (Κανονικά Πολύγωνα με Ν γωνίες).

Παράδειγμα:



Διάγραμμα Hasse

Αναπαράσταση με Κανονικά τρίγωνα

Σχήμα 2.36 : Παράδειγμα εφαρμογής αναπαράστασης Poset με τρίγωνα.

Εικασία: Κάθε Poset με διάσταση μικρότερη και ίση του τρία είναι κυκλικά αναπαραστάσιμο. $\forall P | dim(P) \leq 3\ is\ C.R$ Urrutia (1989). Σημειώνουμε ότι αυτή η εικασία δεν είναι ίση με το θεώρημα του Urrutia.

Το 1998, οι Felsner, Fishburn και Trotter απέδειξαν ότι υπάρχει poset P με $\dim P = 3$ το οποίο δεν είναι κυκλικά αναπαραστάσιμο (C.R.). Η απόδειξη έγινε μέσω θεωρίας Ramsey. Μάλιστα οι συγγραφείς εκτιμούν ότι υπάρχει τέτοιο Poset με λιγότερο από 100 στοιχεία.

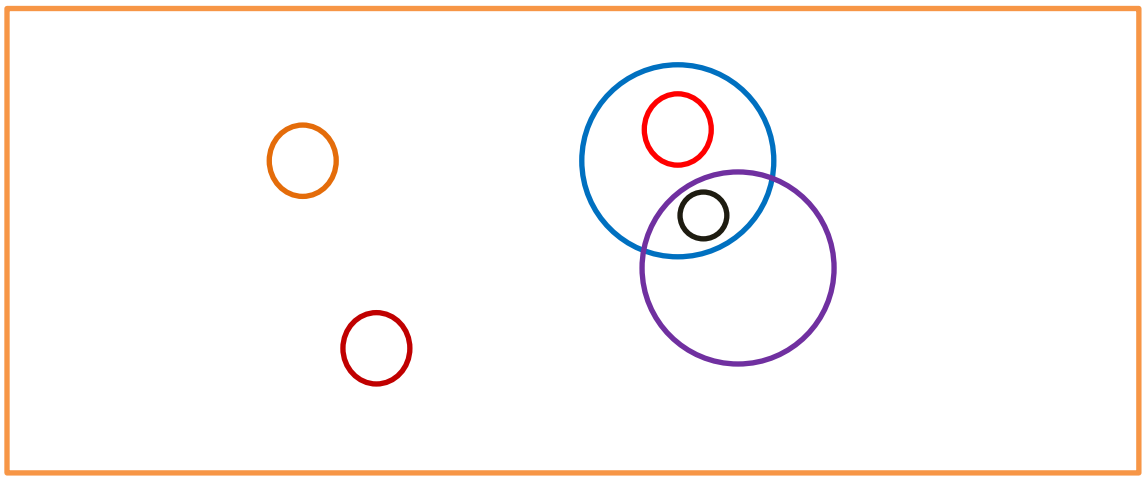
Κεφάλαιο 3

Υπολογιστική Επίλυση

Στο κεφάλαιο αυτό παραθέτουμε το γενετικό αλγόριθμο, ο οποίος αποτελεί τον πυρήνα της επίλυσης της κυκλικής αναπαράστασης των συνόλων μερικής διάταξης. Στον αλγόριθμο αυτό δεν έγινε χρήση διασταύρωσης, αλλά μόνο κριτηρίων ευγενισμού. Σε αυτό περιγράφεται η ευριστική μέθοδος ταιριάσματος. Επιπλέον δίνονται οι συναρτήσεις για την εισαγωγή μιας σχέσης και την εξαγωγή μιας σχέσης στον πίνακα πρόσπτωσης των συνόλων μερικής διάταξης. Τέλος δίνονται τα διαγράμματα ροής του αλγορίθμου.

Περιγραφή γενετικού αλγόριθμου

Αυτό το πρόγραμμα προέρχεται από τους γενετικούς αλγόριθμους. Για να βρούμε μία αναπαράσταση ενός δοσμένου poset με N στοιχεία, αρχίζουμε με μία τυχαία παραγωγή πληθυσμού αντικειμένων. Ένα αντικείμενο είναι ένα σύνολο με N κύκλους που επαληθεύουν μερικές συνθήκες inclusion. Κάθε αντικείμενο δημιουργείται με ένα τυχαίο τρόπο. Οι κύκλοι που αντιπροσωπεύουν στοιχεία μεγαλύτερου βαθμού (degree) έχουν μικρότερες ακτίνες. Μετά προσδιορίζουμε το ταίριασμα (fitness) κάθε αντικειμένου:



Σχήμα 3.1 : Τυχαίο Παράδειγμα

Ταίριασμα – Fitness

Όσο μεγαλύτερο είναι το fitness, τόσο το αντικείμενο είναι κοντά στην αναπαράσταση του poset P. Ένα αντικείμενο του οποίου η συνάρτηση ταιριάσματος αγγίζει τη μέγιστη τιμή, είναι λύση. Για να υπολογίσουμε τη συνάρτηση ταιριάσματος ενός αντικειμένου, βάζουμε ένα πόντο **N+1** για κάθε παρούσα σχέση στο αντικείμενο η οποία είναι παρούσα στο poset P. Με τον ίδιο τρόπο δίνεται ένας πόντος τιμής **1** για κάθε μη συγκρίσιμο ζευγάρι στοιχείων του αντικειμένου όταν είναι μη συγκρίσιμα στο P.

Αντιθέτως, εάν μία σχέση λείπει στο αντικείμενο τότε βάζουμε πρόστιμο **2N**. Για τις σχέσεις που είναι μέσα στο αντικείμενο, αλλά όχι στο αρχικό poset, διακρίνουμε δύο περιπτώσεις:

- Έστω το ύψος των στοιχείων στο διάγραμμα Hasse του poset P όπου τα ελάχιστα του P έχουν ύψος 0, τα μέγιστα έχουν μέγιστο ύψος.

- Έστω χ μικρότερο του ψ (χ<ψ) να είναι μια παρούσα σχέση στο αντικείμενο, αλλά όχι στο P. Εάν το ύψος του χ είναι μικρότερο από το ύψος του ψ, τότε η τιμή του προστίμου είναι **2N** .

Σημειώνουμε ότι αυτός ο έλεγχος είναι στοιχειώδης χάρη στην κανονική απεικόνιση του poset λόγω του αλγορίθμου απαρίθμησης (enumeration). Άρα, το ταίριασμα (fitness) κάθε αντικειμένου είναι το μέγιστο: $\frac{N\ N-1}{2} + N \cdot (το\ πλήθος\ των\ σχέσεων\ του\ P)$. Έχουμε ισότητα μόνο και μόνο όταν το αντικείμενο είναι μια αναπαράσταση του P.

Γενιές – Generations

Τα επόμενα 3 βήματα αποτελούν τη λεγόμενη δημιουργία (generation). Θέτουμε ένα μέγιστο αριθμό δημιουργιών στην αρχή. Εάν ο αλγόριθμος δεν βρεί αναπαράσταση μετά από αυτό το πλήθος δημιουργιών τότε σταματάει.

Βήμα 1: Κάθε αντικείμενο, εκτός από τα καλύτερα 2, θα υποστεί τοπικές διαταραχές έτσι ώστε να καλυτερέψει το ταίριασμά του.

Βήμα 2: Επανεξετάζουμε το ταίριασμα κάθε αντικειμένου.

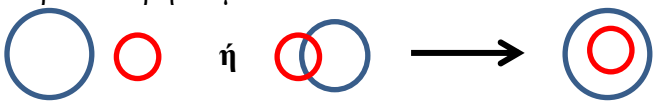
Βήμα 3: Τα δύο χειρότερα αντικείμενα αντικαθιστούνται με καινούρια χωρίς διαδικασία διασταύρωσης (crossing).

Προγραμματίσαμε δύο είδη τοπικών διαταραχών.

- Ο αλγόριθμος μπορεί να σβήσει μια inclusion.

Για να σβήσουμε μια inclusion του K_1 μέσα στο K_2, μετακινούμε τον κύκλο που έχει το μικρότερο score στη διαδικασία της συνάρτησης ταιριάσματος του αντικειμένου. Το κέντρο του κύκλου μεταφέρεται σε μία τυχαία κατεύθυνση με το minimum που σβήνει τη σχέση (inclusion).

- Ο αλγόριθμος μπορεί να σβήσει μια inclusion.

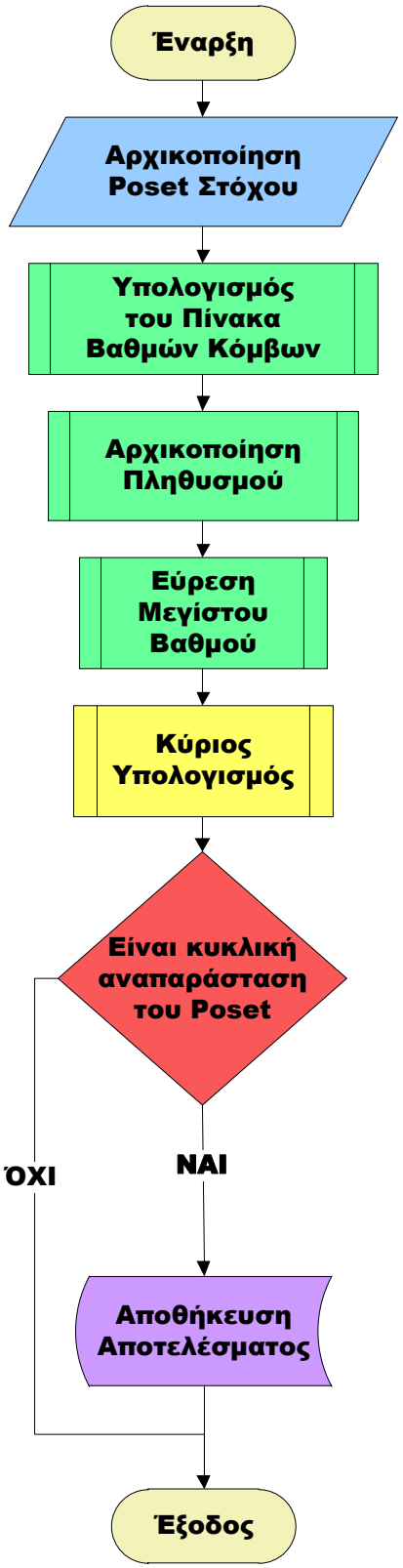


Για να αναγκάσουμε την εμφάνιση της σχέσης (inclusion) του K_1 στο K_2 και πάλι μετακινούμε το κύκλο με το μικρότερο score.

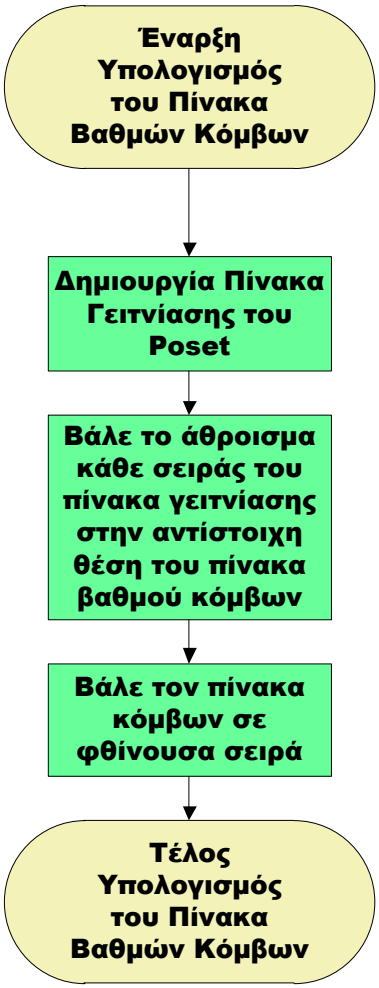
Ο εξαναγκασμός έχει ως στόχο να ελαχιστοποιήσει τις αλλαγές της κατάστασης. Η ακτίνα του $K_1(\rho_1)$ μετατρέπεται σε (ακτίνα του K_1) x (ακτίνα του K_2) $(\rho_1 \cdot \rho_2)$, όπου το ρ είναι τυχαία επιλογή $\frac{1}{2}$ και $\frac{3}{4}$. Τότε το κέντρο του $K_1(\kappa_1)$ μεταφέρεται στην κατεύθυνση που ορίζεται από τα 2 κέντρα, έτσι ώστε να ελαχιστοποιήσει την κίνηση και να ικανοποιήσει το inclusion. Μεταξύ δύο δημιουργιών μερικά αντικείμενα του πληθυσμού μπορούν να υποστούν μερικές μεταλλάξεις (mutation). Αυτά τα αντικείμενα είναι επιλεγμένα τυχαία και μία mutation αποτελείται από μία εξαναγκασμένη inclusion η οποία είναι απαραίτητη, και δεν την έχει πετύχει το αντικείμενο.

Παρατήρηση:

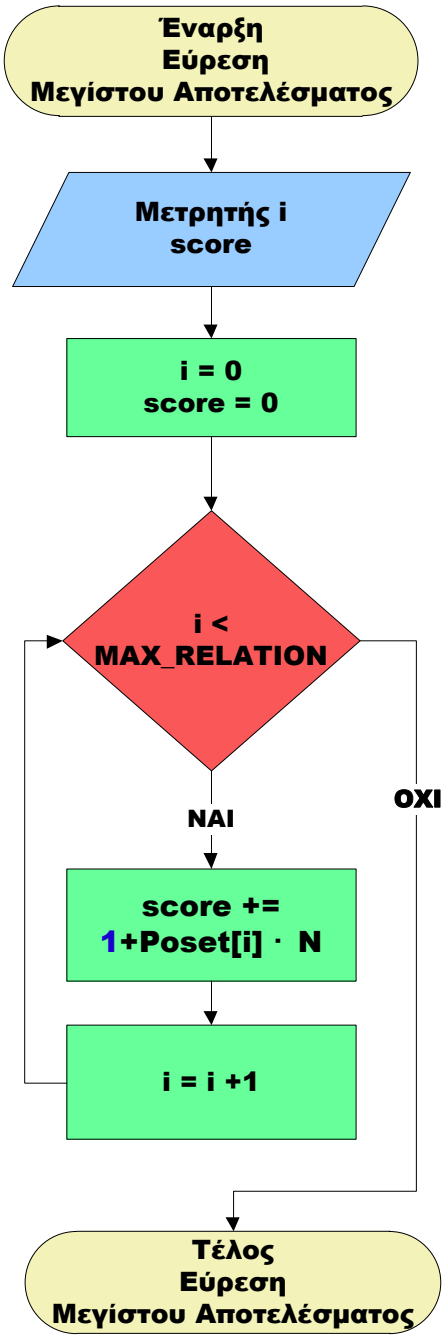
Σημειώνουμε τη σημασία της τοπικότητας των διαταραχών. Για παράδειγμα όταν μετακινούμε έναν κύκλο για εξαναγκασμό ή σβήσιμο. Αν μετακινήσουμε όλους τους κύκλους που περιέχει ο κύκλος Κ, ή τους κύκλους που περιέχουν Κ, τότε ο αλγόριθμος είναι πολύ πιο αδύναμος.



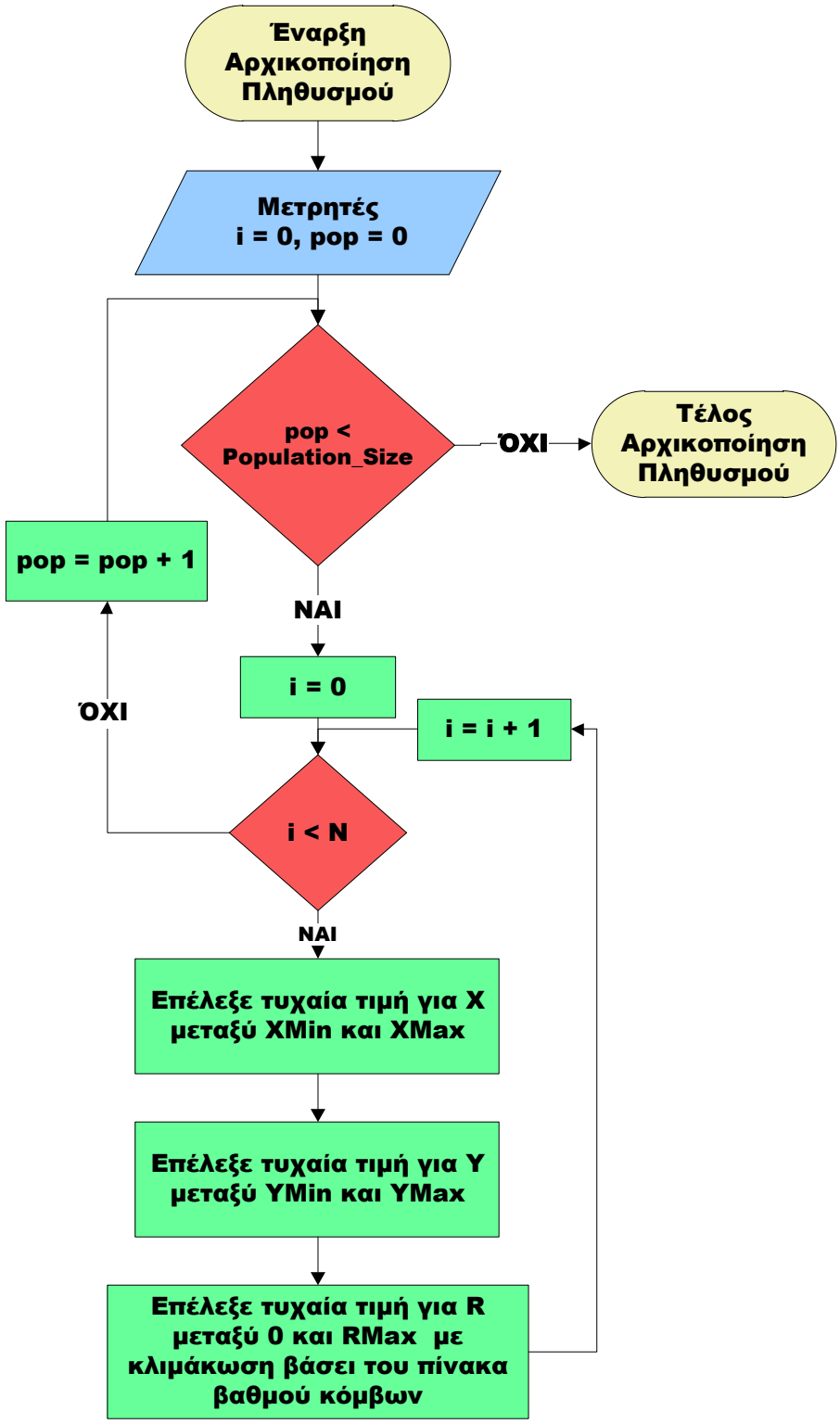
Διάγραμμα Ροής 3.1 : Γενετικός Αλγόριθμος



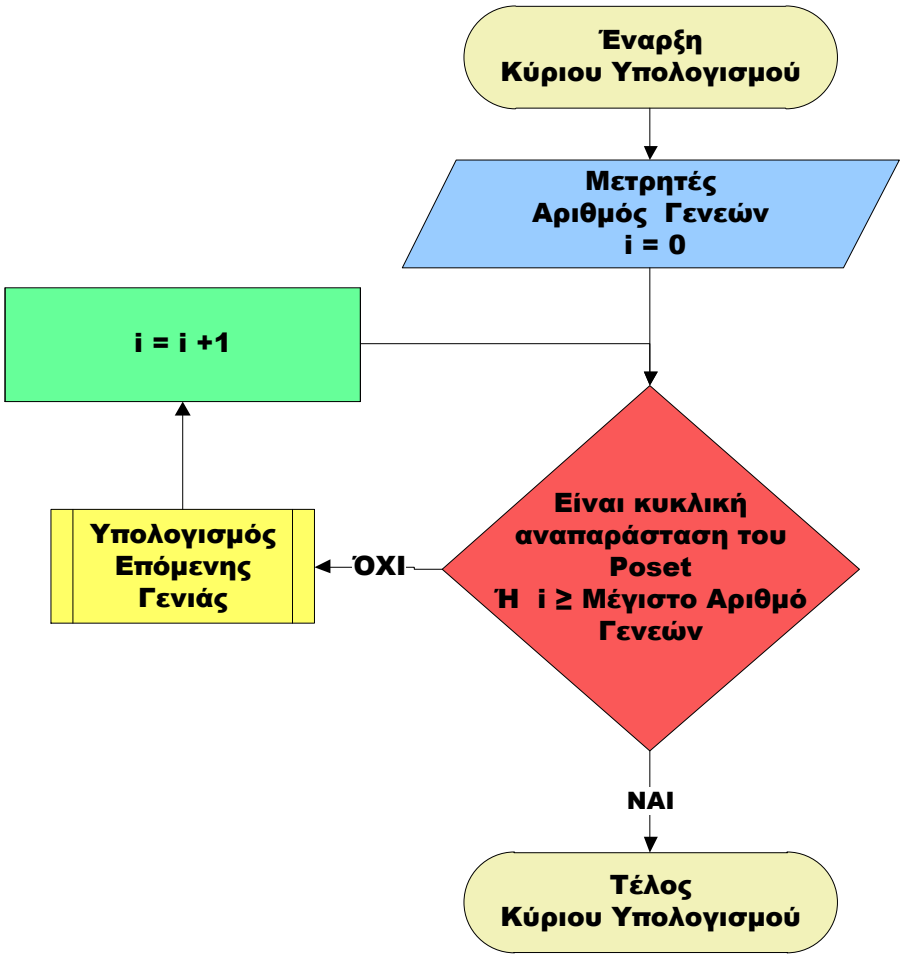
Διάγραμμα Ροής 3.2 **:** *Υπολογισμός του Πίνακα Βαθμών Κόμβων*



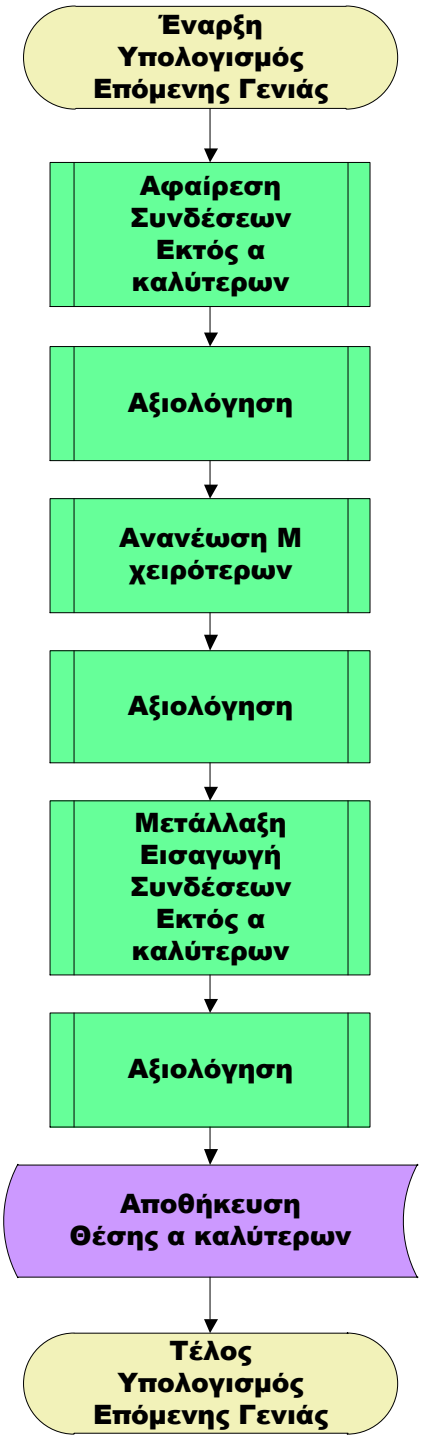
Διάγραμμα Ροής 3.3 : Εύρεση Μέγιστου Αποτελέσματος



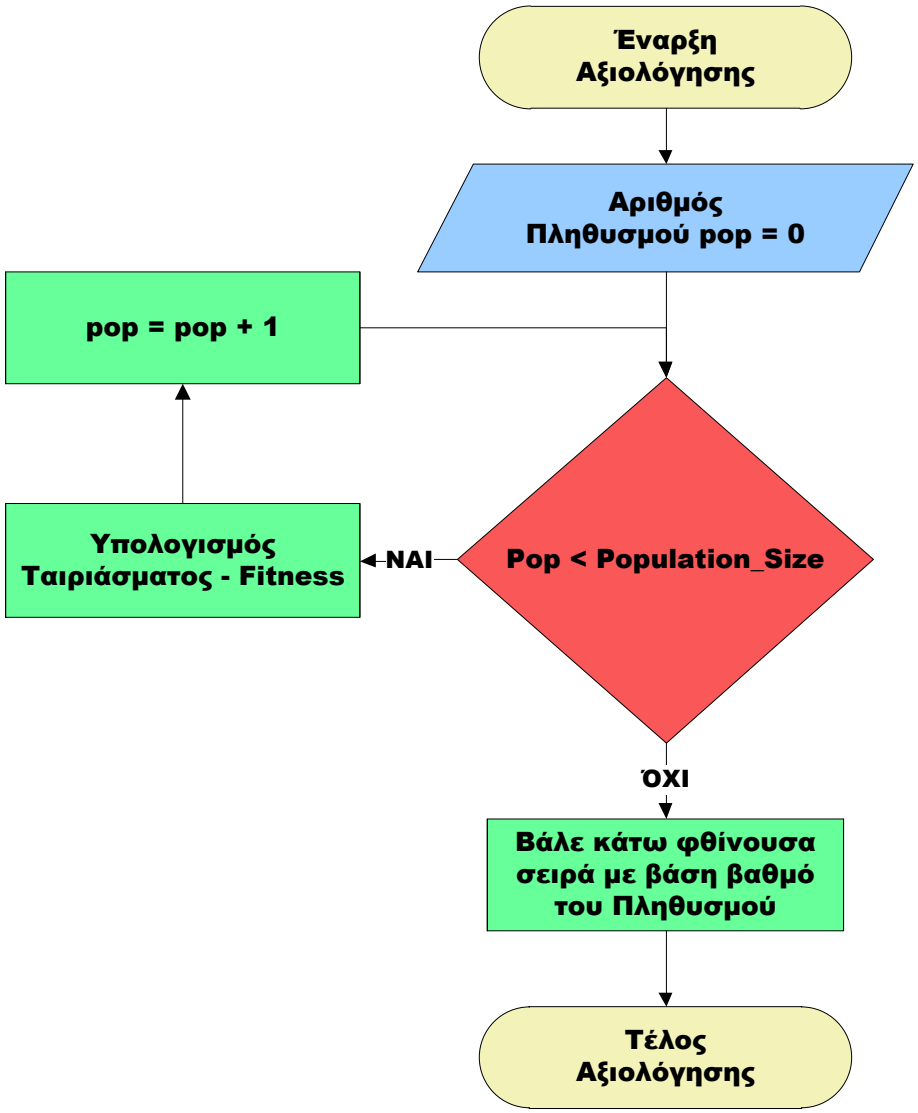
Διάγραμμα Ροής 3.4 : Αρχικοποίηση του Πληθυσμού



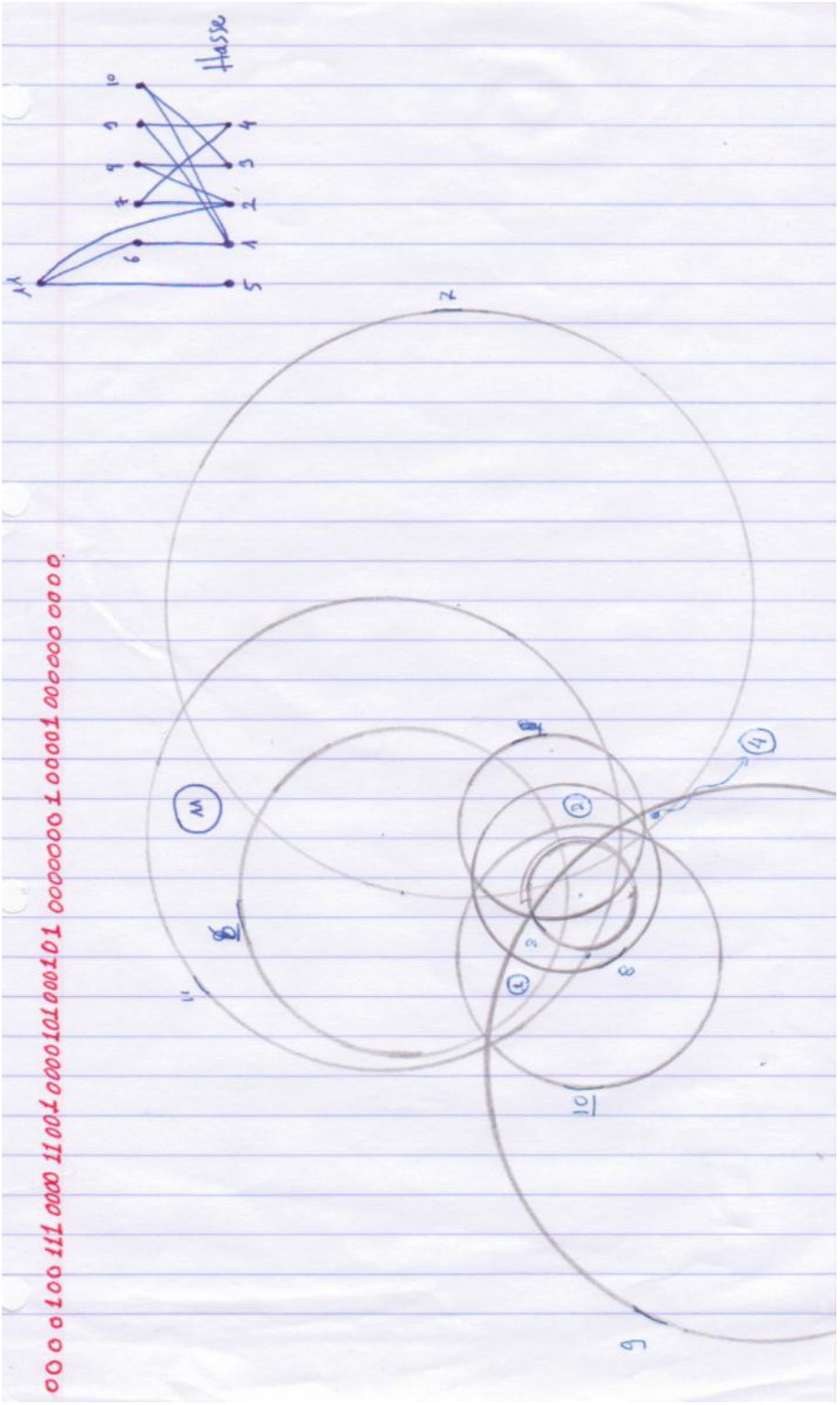
Διάγραμμα Ροής 3.5 **:** *Κύριος Υπολογισμός*



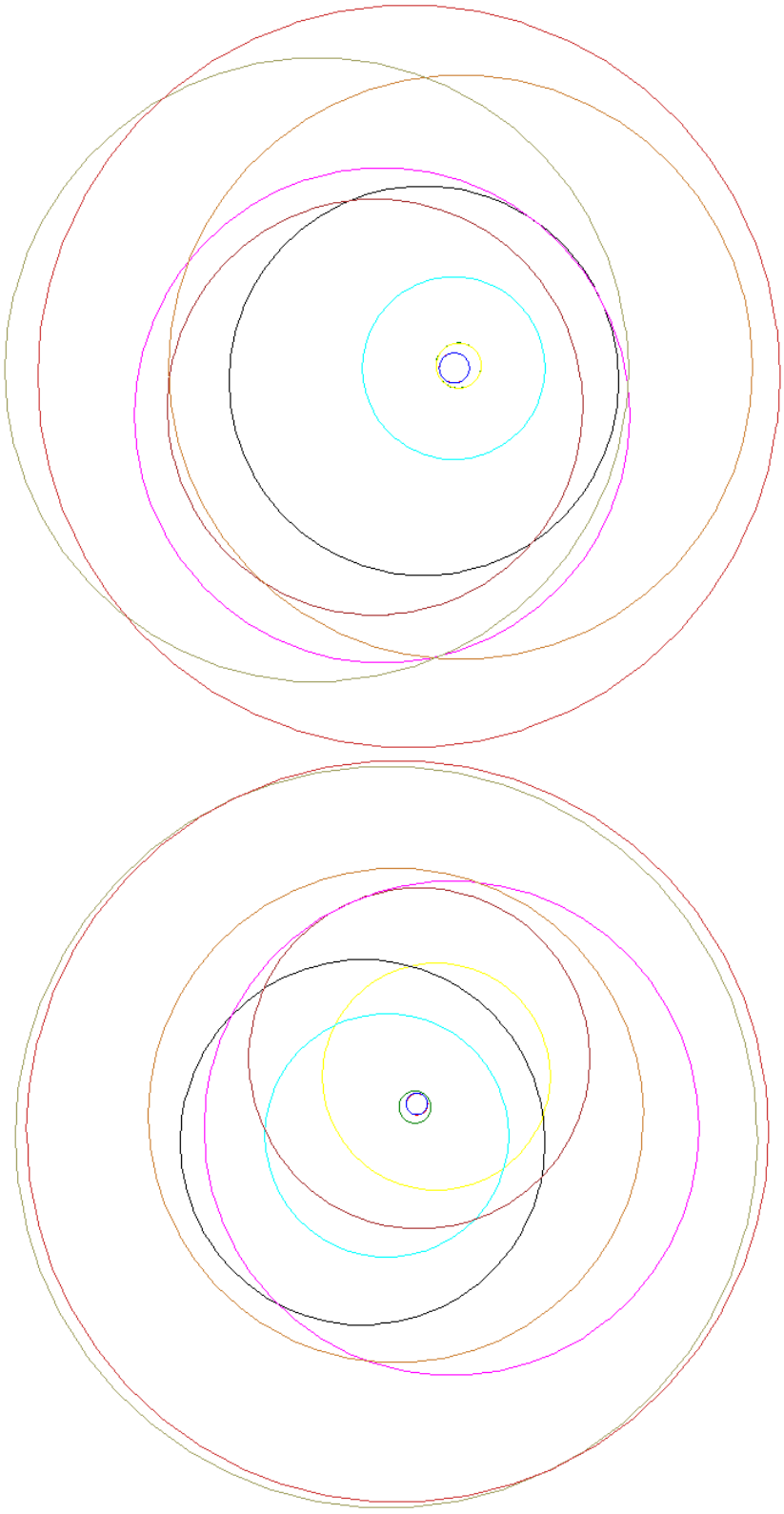
Διάγραμμα Ροής 3.6 : Υπολογισμός Επόμενης Γενιάς



Διάγραμμα Ροής 3.7 : Αξιολόγηση Πληθυσμού



Σχήμα 3.2 : Παράδειγμα εύρεσης της κυκλικής αναπαράστασης με το χέρι.



Σχήμα 3.3 : Παράδειγμα εύρεσης της κυκλικής αναπαράστασης με το γενετικό αλγόριθμο και χρήση Maple για την απεικόνιση.

Κεφάλαιο 4

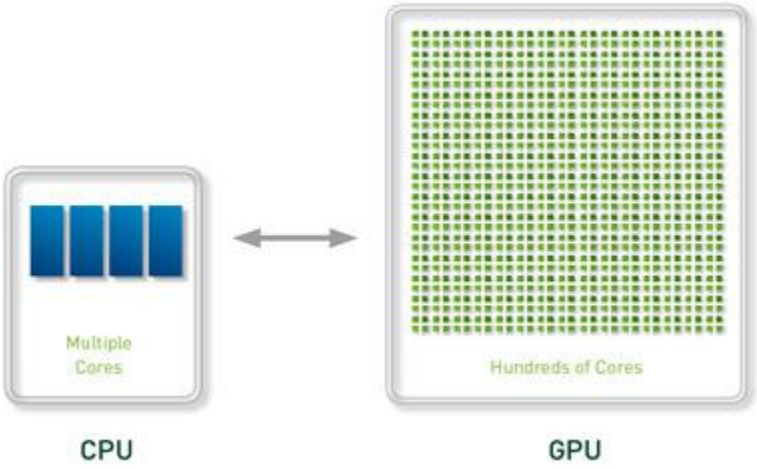
Υλοποίηση σε GPU

Στο κεφάλαιο αυτό θα δούμε τι σημαίνει ο όρος GPU. Υπάρχουν μερικές βασικές διαφορές μεταξύ της CPU και GPU. Στη συνέχεια θα δούμε την εξέλιξη της GPU δια μέσου του χρόνου. Για τον προγραμματισμό των GPU, η NVidia δημιούργησε μία προγραμματιζόμενη αρχιτεκτονική, γνωστή ως Cuda. Έπειτα, θα δούμε την εξέλιξη της αρχιτεκτονικής αυτής και πώς είναι η νέα αρχιτεκτονική Fermi στις κάρτες Tesla. Παραθέτονται μερικά χαρακτηριστικά του προγραμματιστικού μοντέλου και μερικές ιδιότητες και τεχνικά χαρακτηριστικά που λήφθησαν υπόψη κατά τον προγραμματισμό. Τέλος περιγράφονται οι τρόποι που χρησιμοποιήθηκαν για την υλοποίηση του γενετικού αλγορίθμου στις GPU και ποιες παραδοχές έγιναν. Δίνονται ενδεικτικά διαγράμματα κατανομής υπολογιστικού χρόνου ανά συνάρτηση της GPU και της CPU. Σημειώνουμε ότι ο παραλληλισμός της εύρεσης της κυκλικής διάταξης των Posets κρίνεται απαραίτητος λόγω του ότι ο αριθμός των Posets, όπως είδαμε και στο Κεφάλαιο 2, αυξάνεται εκθετικά συναρτήσει της πληθικότητας.

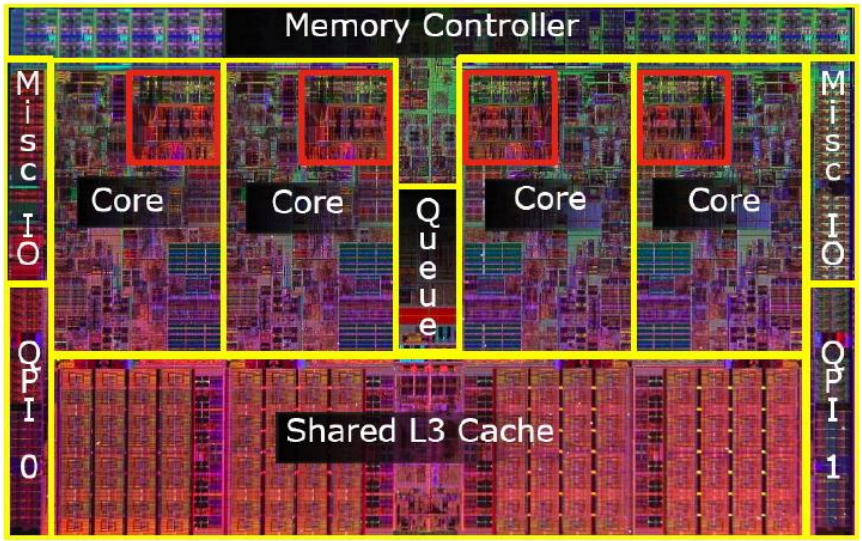
Τι είναι οι GPUs.

Ο υπολογιστής GPU ή GPGPU είναι η χρήση μιας GPU (graphics processing unit) για να εκτελεί γενικής χρήσης επιστημονικούς και μηχανολογικούς υπολογισμούς.

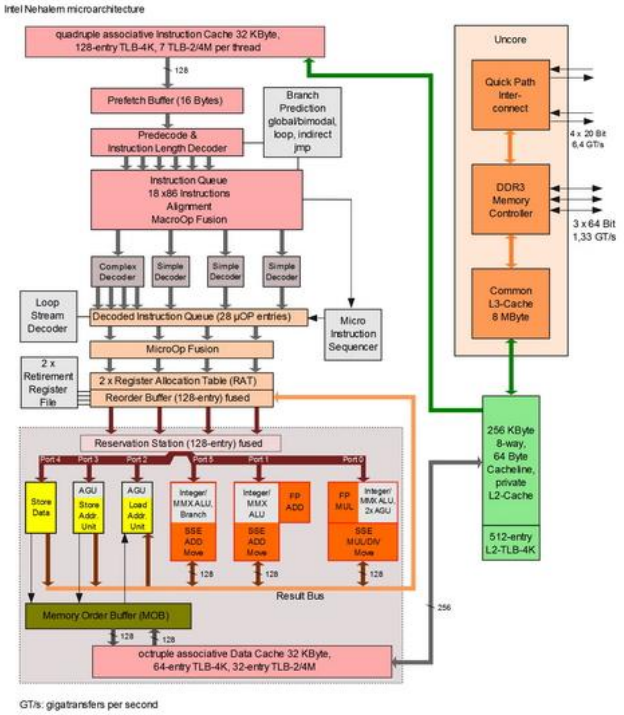
Το υπολογιστικό μοντέλο GPU είναι η χρήση CPU και GPU μαζί σε ένα ετερογενή συνεπεξεργαστικό υπολογισμό. Το σειριακό μέρος του προγράμματος τρέχει στην CPU και το βαρύ υπολογιστικό μέρος επιταχύνεται από την GPU. Από την πλευρά του χρήστη το πρόγραμμα απλά τρέχει γρηγορότερα, επειδή χρησιμοποιεί τη μεγάλη υπολογιστική ισχύ της GPU για να επιταχύνει την απόδοση.



Σχήμα 4.1 : Συνεργασία CPU με GPU. (Πηγή : NVidia Corporation)



Σχήμα 4.2 : Επεξεργαστής Intel's Core i7 (το chip με το κωδικό όνομα Bloomfield, βασισμένο στη μικροαρχιτεκτονική Nehalem) (Πηγή : Intel Corporation except red highlighting)



Σχήμα 4.3 : Αρχιτεκτονική Nehalem. (Πηγή: Wikipedia)

Η τεχνολογία της GPU έχει εξελιχτεί μέσα στα χρόνια με αποτέλεσμα να έχει την ικανότητα να εκτελεί δέκα εις την δωδεκάτη πράξεις κινητής υποδιαστολής το δευτερόλεπτο (teraflops). Η NVIDIA έκανε την επανάσταση στις GPGPUs το 2006-2007 με την εισαγωγή ενός νέου πακέτο μαζικού προγραμματισμού που ονομάστηκε CUDA. Η αρχιτεκτονική CUDA ("Compute Unified Device Architecture") αποτελείται από εκαντοντάδες υπολογιστικούς πυρήνες οι οποίοι λειτουργούν μαζί στην κρίσιμη στιγμή που τα δεδομένα δίδονται από την εφαρμογή.

Η επιτυχία των GPGPUs τα περασμένα χρόνια προήλθε από τον εύκολο προγραμματισμό που σχετίζεται με το παράλληλο προγραμματιστικό μοντέλο CUDA. Στο προγραμματιστικό μοντέλο, ο προγραμματιστής μετατρέπει την εφαρμογή έτσι ώστε το βαρύ υπολογιστικό κομμάτι να προσαρμοστεί στην GPU. Το υπόλοιπο πρόγραμμα παραμένει να εκτελείται στην CPU. Η προσαρμογή μιας συνάρτησης στην GPU συνεπάγεται επανεγγραφή της συνάρτησης για να είναι αποδοτική στον παραλληλισμό και προσθέτοντας "C" λέξεις κλειδιά για την ανταλλαγή δεδομένων με την GPU. Ο προγραμματιστής εκκινεί τον πυρήνα κι αυτός δρομολογεί δεκάδες έως χιλιάδες νήματα. Το υλικό μέρος της GPU διαχωρίζει τα νήματα αυτά και υλοποιεί τη νηματική δρομολόγηση.

Η σειρά Tesla 20 GPU είναι βασισμένη στην αρχιτεκτονική FERMI, η οποία είναι η πιο πρόσφατη αρχιτεκτονική CUDA. Η Fermi αρχιτεκτονική είναι βελτιστοποιημένη για επιστημονικές εφαρμογές με ιδιαίτερα χαρακτηριστικά, όπως 500+ gigaflops από το IEEE πρότυπο για διπλής ακρίβειας κινητή υποδιαστολή (με υποστήριξη υλικού), κρυφή μνήμη επιπέδου 1 και 2 (L1,L2 caches), κώδικα διόρθωσης

λαθών για προστασία της μνήμης από λάθη (ECC). Επίσης υποστηρίζει τον τοπικό χρήστη ώστε να διαχειρίζεται τις κρυφές μνήμες στη μορφή της μοιραζόμενης μνήμης διασκορπισμένης σε όλη την GPU, συγχωνευμένες προσβάσεις στη μνήμη και άλλα πολλά.

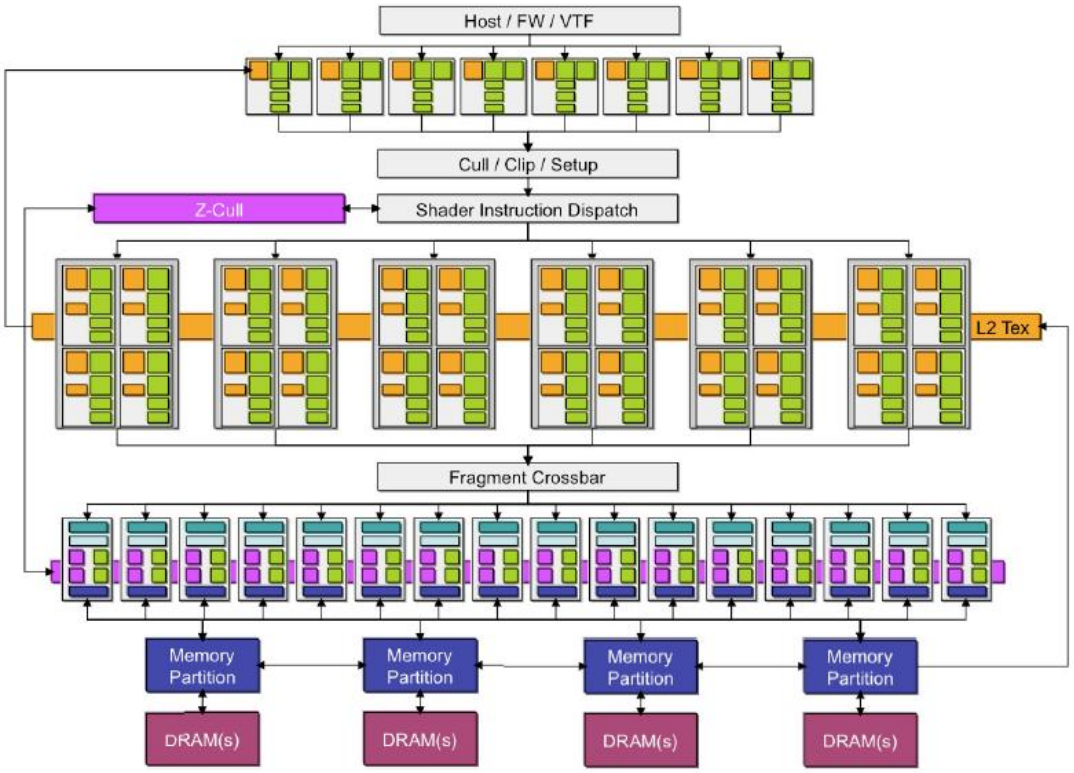
Η Ιστορία της GPU

Είναι ένα πράγμα να αναγνωρίζει κάποιος τις μελλοντικές δυνατότητες μιας νέας αρχιτεκτονικής επεξεργασίας και είναι άλλο πράγμα να οικοδομεί μια αγορά πριν μπορέσει να επιτευχθεί αυτή η δυνατότητα. Υπήρξαν προσπάθειες για την οικοδόμηση chip-scale παράλληλων επεξεργαστών στη δεκαετία του 1990, αλλά οι περιορισμένοι προϋπολογισμοί τρανζίστορ σε εκείνη την εποχή ευνόησε περισσότερο εξελιγμένα σχέδια μονού-πυρήνα (single-core).

Η πραγματική πορεία προς την υπολογιστική επεξεργασία γραφικών (GPU computing) ξεκίνησε όχι με τις GPU αλλά με τις μη προγραμματιζόμενες 3G-κάρτες γραφικών. Μηχανές πολλαπλών ψηφίδων (multi-chip) 3D απόδοσης είχαν αναπτυχθεί από πολλές εταιρείες ξεκινώντας από τη δεκαετία του 1980, αλλά από τα μέσα της δεκαετίας του 1990 κατέστη δυνατόν να ενσωματωθούν όλα τα ουσιώδη στοιχεία σε ένα μόνο chip. Από το 1994 μέχρι το 2001, αυτά τα chips προχώρησαν από τις πιο απλές λειτουργίες σχεδίασης εικονοστοιχείου (pixel- drawing) σε εφαρμογή της πλήρους 3D (τριών διαστάσεων) επεξεργασίας (pipeline): μετασχηματισμούς (transforms), φωτισμό (lighting), πλαισιοποίηση (rasterization), υφή (texturing), δοκιμή βάθους (depth testing) και απεικόνιση (display).

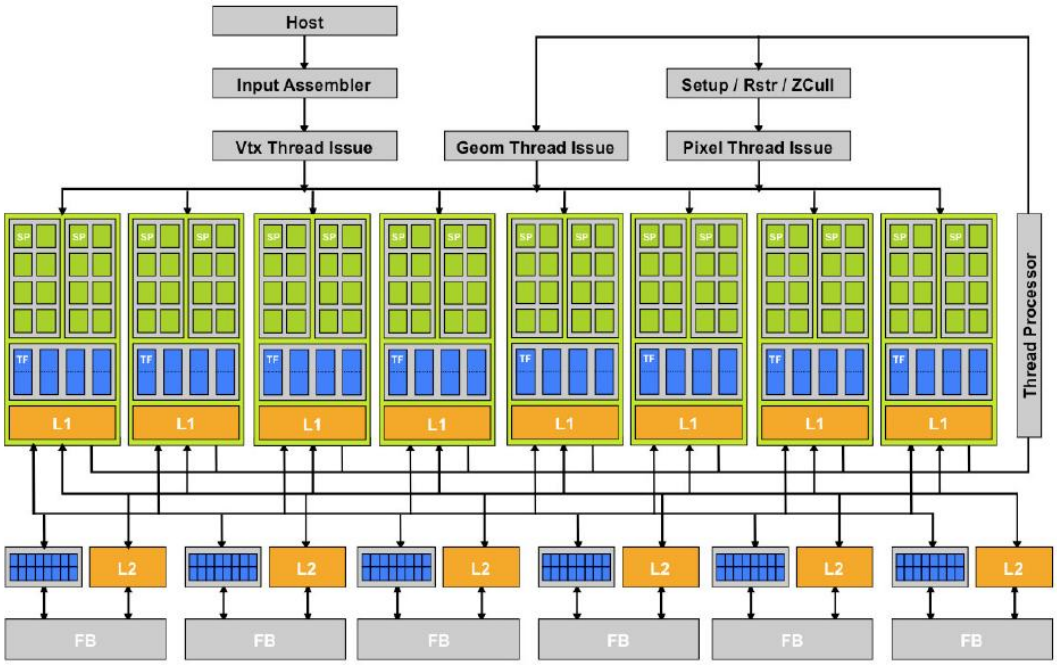
Η GeForce 3 της NVIDIA το 2001 παρουσίασε προγραμματιζόμενα εικονοστοιχεία σκίασης (programmable pixel shading) στην καταναλωτική αγορά. Η δυνατότητα προγραμματισμού αυτού του chip ήταν πολύ περιορισμένη, αλλά αργότερα τα προϊόντα GeForce έγιναν πιο ευέλικτα και πιο γρήγορα, προσθέτοντας ξεχωριστές προγραμματιζόμενες μηχανές για “vertex and geometry shading”. Αυτή η εξέλιξη είχε ως αποτέλεσμα τη GeForce 7800 που φαίνεται στο Σχήμα 4.4.

Ο λεγόμενος γενικής χρήσης προγραμματισμός GPU (GPGPU=General Purpose GPU) αναπτύχθηκε ως ένας τρόπος για να εκτελούν επεξεργασία μη γραφικών σε αυτές τις βελτιστοποιημένες για γραφικά αρχιτεκτονικές, τυπικά με το τρέξιμο προσεκτικά κατασκευασμένου κώδικα δυνατότητας σκίασης σε σχέση με τα στοιχεία που παρουσιάζονται ως πληροφορίες κορυφής ή υφής (vertex or texture information) και την ανάκτηση των δεδομένων σε μεταγενέστερο στάδιο στον αγωγό (pipeline). Αν και μερικές φορές αδέξιος, ο GPGPU φάνηκε πολλά υποσχόμενος.



Σχήμα 4.4 : Η GeForce 7800 είχε τριών ειδών προγραμματιζόμενες μηχανές για διάφορα στάδια του τρισδιάστατου αγωγού (3D pipeline) συν μερικά επιπρόσθετα στάδια διαμορφώσιμης και σταθερής λειτουργίας λογικής (configurable and fixed-function logic) (Πηγή: NVidia Corporation)

Η διαχείριση τριών διαφορετικών προγραμματιζόμενων μηχανών σε ένα μόνο τρισδιάστατο (3D) αγωγό οδήγησε σε απρόβλεπτες συμφορήσεις (bottlenecks). Καταβλήθηκε μεγάλη προσπάθεια στην εξισορρόπηση της απόδοσης κάθε σταδίου. Το 2006 η NVIDIA παρουσίασε τη GeForce 8800, όπως δείχνει το Σχήμα 4.5. Αυτόν το σχεδιασμό τον χαρακτήρισε μια ενοποιημένη αρχιτεκτονική δυνατότητας σκίασης (unified shader architecture) με 128 στοιχεία επεξεργασίας κατανεμημένα ανάμεσα σε οκτώ πυρήνες δυνατότητας σκίασης (shader). Κάθε πυρήνας σκίασης (ή δυνατότητας σκίασης) (shader core) θα μπορούσε να αντιστοιχιστεί σε οποιοδήποτε έργο του shader, εξαλείφοντας την ανάγκη για εξισορρόπηση βήμα προς βήμα και βελτιώνοντας κατά πολύ τη συνολική απόδοση.



Σχήμα 4.5 : Η GeForce 8800 εισήγαγε μια ενοποιημένη αρχιτεκτονική σκίασης (unified shader architecture) με ακριβώς ένα είδος στοιχείου προγραμματιζόμενης επεξεργασίας που θα μπορούσε να χρησιμοποιηθεί για πολλούς σκοπούς. Μερικές απλές λειτουργίες γραφικών χρησιμοποιούσαν ακόμα ειδικού σκοπού λογική (special-purpose logic) (Πηγή: NVidia Corporation)

Η 8800 επίσης εισήγαγε την CUDA, το πρώτο περιβάλλον ανάπτυξης για GPUs στη βιομηχανία βασισμένο σε C (CUDA αρχικά δήλωνε "Compute Unified Device Architecture" αλλά το μεγαλύτερο όνομα δε χρησιμοποιείται πλέον). Η CUDA έδωσε ένα ευκολότερο και πιο αποτελεσματικό μοντέλο προγραμματισμού από ότι οι προηγούμενες προσεγγίσεις GPGPU.

Για να φέρει τα πλεονεκτήματα της αρχιτεκτονικής 8800 και της CUDA σε νέες αγορές όπως στην HPC, η NVIDIA παρουσίασε τη σειρά προϊότων Tesla. Τρέχοντα προϊόντα Tesla χρησιμοποιούν την πιο πρόσφατη αρχιτεκτονική GT200.

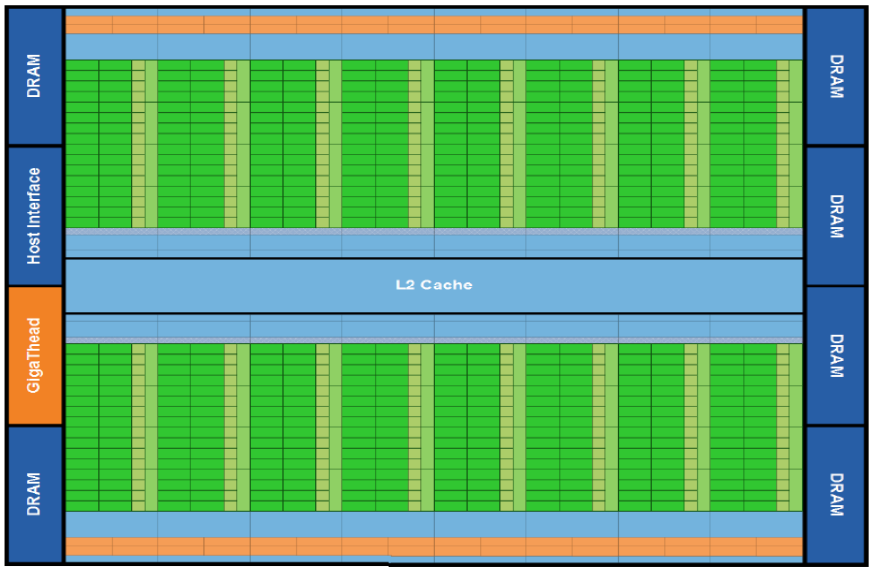
Σε αντίθεση με παλαιότερες προσπάθειες πολυεπεξεργασίας σε κλίμακα chip πίσω στη δεκαετία του 90, η Tesla ήταν από την αρχή μια υψηλού όγκου πλατφόρμα υλικού (high-volume hardware platform). Αυτό οφείλεται εν μέρει στη στρατηγική της NVIDIA για υποστήριξη της πλατφόρμας ανάπτυξης λογισμικού CUDA στα προϊόντα GeForce και Quadro της εταιρείας, καθιστώντας τα προσιτά σε ένα πολύ ευρύτερο κοινό προγραμματιστών. Η NVIDIA λέει ότι έχει στείλει πάνω από 100 εκατομμύρια πλακίδια CUDA-capable (με δυνατότητα CUDA).

Παρουσιάζοντας τη Fermi

Η πληροφορική GPU δεν αποσκοπεί στην υποκατάσταση των υπολογιστών CPU. Κάθε προσέγγιση έχει πλεονεκτήματα για ορισμένα είδη λογισμικού. Όπως εξηγήθηκε προηγουμένως, οι CPUs έχουν βελτιστοποιηθεί για εφαρμογές όπου το μεγαλύτερο μέρος της εργασίας γίνεται από ένα περιορισμένο αριθμό νημάτων (threads), ιδιαίτερα όπου τα νήματα εμφανίζουν υψηλή συγκέντρωση δεδομένων, ένα μίγμα διαφορετικών λειτουργιών και ένα υψηλό ποσοστό κλάδων υπό συνθήκη.

Ο σχεδιασμός GPU στοχεύει στο άλλο άκρο του φάσματος: εφαρμογές με πολλαπλά νήματα που κυριαρχούνται από μεγαλύτερες αλληλουχίες υπολογιστικών οδηγιών. Κατά τα τελευταία λίγα χρόνια, οι GPUs έχουν γίνει πολύ καλύτερες στο χειρισμό νήματος, στην προσωρινή αποθήκευση δεδομένων (data caching), στη διαχείριση της εικονικής μνήμης (virtual memory management), τον έλεγχο ροής (flow control) και σε άλλα γνωρίσματα χαρακτηριστικά της CPU, αλλά η διάκριση μεταξύ υπολογιστικά εντατικού λογισμικού (computationally intensive software) και του ελέγχου ροής εντατικού λογισμικού (control-flow intensive software) είναι θεμελιώδης.

Η εξέλιξη της τεχνολογίας στο σχεδιασμό GPU εκπροσωπείται από την αρχιτεκτονική νέας γενιάς CUDA της NVIDIA, με την κωδική ονομασία Fermi. Το σχήμα 4.6 δείχνει ένα υψηλού επιπέδου διάγραμμα του πρώτου chip Fermi.



Σχήμα 4.6 : Η αρχιτεκτονική Fermi GPU της NVIDIA αποτελείται από πολλαπλούς πολυεπεξεργαστές ροής (Streaming Multiprocessors (SMs)), ο καθένας αποτελούμενος από 32 πυρήνες, έκαστος των οποίων μπορεί να εκτελέσει μια κινητής υποδιαστολής (floating -point) ή ακεραίου εντολή ανά ρολόι. Οι SMs υποστηρίζονται από μια κρυφή μνήμη δευτέρου επιπέδου (second-level cache), διεπαφή προσδέκτη (host interface), προγραμματιστή (scheduler) GigaThread, και πολλαπλές διασυνδέσεις μνημών DRAM (Πηγή: NVidia Corporation)

Σε αυτό το επίπεδο σύνοψης, η GPU μοιάζει με θάλασσα υπολογιστικών μονάδων με μόνο λίγα στοιχεία υποστήριξης - ένα παράδειγμα του βασικού στόχου σχεδιασμού της GPU , που είναι να μεγιστοποιήσει την απόδοση κινητής υποδιαστολής (floating-point throughput).

Δεδομένου ότι τα περισσότερα από τα κυκλώματα μέσα σε κάθε πυρήνα είναι αφιερωμένα στον υπολογισμό παρά σε κερδοσκοπικά χαρακτηριστικά που στόχο έχουν να ενισχύσουν τη "single-threaded" απόδοση, το μεγαλύτερο μέρος της περιοχής "die" και η ισχύς που καταναλώνεται από τη Fermi, πηγαίνει στην πραγματική αλγοριθμική εργασία της εφαρμογής.

Αρχιτεκτονικές GPUs.

Η αρχιτεκτονική CUDA είναι κτισμένη γύρω από ένα κλιμακωτό φάσμα πολυνηματικών πολυεπεξεργαστών. Όταν ένα πρόγραμμα CUDA στη φιλοξενούσα κεντρική μονάδα επεξεργασίας επικαλείται ένα πλέγμα πυρήνα, τα μπλοκ του πλέγματος απαριθμούνται και διανέμονται σε πολυεπεξεργαστές με διαθέσιμη δυναμικότητα εκτέλεσης. Τα νήματα ενός νηματικού μπλοκ εκτελούνται ταυτόχρονα σε ένα πολυεπεξεργαστή και πολλαπλά νηματικά μπλοκ μπορούν να εκτελούνται ταυτόχρονα σε ένα πολυεπεξεργαστή. Καθόσον τα νηματικά μπλοκ τερματίζουν, νέα μπλοκ προωθούνται στους κενωθέντες πολυεπεξεργαστές.

Ένας πολυεπεξεργαστής σχεδιάζεται να εκτελεί εκατοντάδες νήματα ταυτόχρονα. Για τη διαχείριση ενός τόσου μεγάλου ποσού νημάτων, χρησιμοποιεί μια μοναδική αρχιτεκτονική που ονομάζεται SIMT (Single-Instruction=Μία Εντολή, Multiple-Thread=Πολλαπλά Νήματα). Οι οδηγίες είναι συνεχούς διοχέτευσης, για να προάγουν τον παραλληλισμό επιπέδου εντολών μέσα σε ένα μόνο νήμα, καθώς και τον παραλληλισμό επιπέδου νήματος διεξοδικά με ταυτόχρονο εξοπλισμό. Σε αντίθεση με πυρήνες CPU, αυτές εκδίδονται ωστόσο σε σειρά και δεν υπάρχει καμία πρόβλεψη κλάδου και καμία κερδοσκοπική εκτέλεση.

Ο πολυεπεξεργαστής δημιουργεί, διαχειρίζεται, σχεδιάζει (χρονοδιαγράμματα) και εκτελεί τα νήματα σε ομάδες των 32 παράλληλων νημάτων που ονομάζονται στημόνια (warps). Τα μεμονωμένα νήματα που συνθέτουν ένα στημόνι ξεκινούν μαζί στην ίδια διεύθυνση προγράμματος, αλλά έχουν το δικό τους μετρητή διευθύνσεων εντολών (instruction address counter) και κατάσταση μητρώου (register state) και είναι συνεπώς ελεύθερα να διακλαδώνονται και να εκτελούνται ανεξάρτητα. Ο όρος "στημόνι=warp" προέρχεται από την ύφανση, την πρώτη τεχνολογία παράλληλου νήματος. Ένα μισό-στημόνι (half-warp) είναι είτε το πρώτο είτε το δεύτερο μισό του στημονιού. Ένα τέταρτο του στημονιού (quarter-warp) είναι είτε το πρώτο, το δεύτερο, το τρίτο ή το τέταρτο τέταρτο του στημονιού.

Όταν δίνονται ένα ή περισσότερα μπλοκ νημάτων να εκτελεστούν σε ένα πολυεπεξεργαστή, τα χωρίζει σε στημόνια (warps) που προγραμματίζονται για εκτέλεση από ένα προγραμματιστή στημονιού (warp scheduler). Ο τρόπος που ένα μπλοκ χωρίζεται σε στημόνια είναι πάντοτε ο ίδιος· κάθε στημόνι περιλαμβάνει νήματα διαδοχικών, αυξανομένων ταυτοτήτων (IDs) νημάτων με το πρώτο στημόνι να περιλαμβάνει το νήμα 0.

Ένα στημόνι εκτελεί μια κοινή εντολή κάθε φορά, έτσι επιτυγχάνεται πλήρης αποδοτικότητα, όταν όλα τα 32 νήματα του στημονιού συμφωνούν πάνω στην πορεία εκτέλεσής τους. Αν τα νήματα του στημονιού αποκλίνουν μέσω δεδομένων που εξαρτώνται από τη διακλάδωση, το στημόνι εκτελεί σειριακά κάθε μονοπάτι διακλάδωσης, απενεργοποιώντας τα νήματα που δεν βρίσκονται στο μονοπάτι, και όταν όλα τα μονοπάτια συμπληρωθούν, τα νήματα συγκλίνουν και πάλι στο ίδιο μονοπάτι εκτέλεσης. Απόκλιση κλάδου συμβαίνει μόνο μέσα σε ένα στημόνι· διαφορετικά στημόνια εκτελούνται ανεξάρτητα από το αν εκτελούν κοινά ή κομματιασμένα μονοπάτια κώδικα.

Η αρχιτεκτονική SIMT είναι παρόμοια με τις SIMD (Single Instruction, Multiple Data) οργανώσεις διανυσμάτων στο ότι μια μόνο εντολή ελέγχει πολλαπλά στοιχεία επεξεργασίας. Μια βασική διαφορά είναι το ότι οι οργανώσεις διανυσμάτων SIMD εκθέτουν το πλάτος SIMD στο λογισμικό, ενώ οι οδηγίες SIMT προσδιορίζουν την εκτέλεση και τη συμπεριφορά διακλάδωσης ενός μόνο νήματος. Σε αντίθεση με τις διανυσματικές μηχανές SIMD, το SIMT επιτρέπει στους προγραμματιστές να γράψουν παράλληλο κώδικα επιπέδου νημάτων για ανεξάρτητα, βαθμωτά νήματα, καθώς και κώδικα για δεδομένα παράλληλα για συντονισμένα νήματα. Για σκοπούς ορθότητας, ο προγραμματιστής μπορεί να αγνοήσει ουσιαστικά τη SIMT συμπεριφορά.

Εάν μια μη ατομική εντολή εκτελούμενη από ένα στημόνι γράφει στην ίδια θέση σε global ή κοινόχρηστη (shared) μνήμη για περισσότερα από ένα νήματα του στημονιού, ο αριθμός των σειριακών γραφών που συμβαίνει σε αυτή τη θέση διαφέρει ανάλογα με την υπολογιστική ικανότητα της συσκευής και είναι απροσδιόριστο πιο νήμα εκτελεί την τελική γραφή (ή πιο νήμα γράφει τελευταίο).

Εάν μια ατομική εντολή που εκτελείται από ένα στημόνι διαβάζει, τροποποιεί και γράφει στην ίδια θέση σε global μνήμη για περισσότερο από ένα από τα νήματα του στημονιού, κάθε ανάγνωση, τροποποίηση, γραφή σε εκείνη τη θέση συμβαίνει και είναι όλα σε συνέχειες, αλλά η σειρά με την οποία εμφανίζονται είναι απροσδιόριστη.

Το Μοντέλο Προγραμματισμού

Η πολυπλοκότητα της αρχιτεκτονικής Fermi διευθύνεται από ένα πολυεπίπεδο (multi-level) μοντέλο προγραμματισμού τον οποίο επιτρέπει στους προγραμματιστές λογισμικού να επικεντρωθούν στο σχεδιασμό του αλγορίθμου παρά στις λεπτομέρειες του πώς θα αντιστοιχίσουν τον αλγόριθμο στο υλικό (hardware), βελτιώνοντας έτσι την παραγωγικότητα. Αυτό είναι ένα πρόβλημα που οι συμβατικοί CPUs έχουν ακόμη να αντιμετωπίσουν γιατί οι δομές τους είναι απλές και τακτικές: ένας μικρός αριθμός πυρήνων (cores) παρουσιασμένοι σαν λογικά ομότιμοι (logical peers) σε ένα εικονικό δίαυλο (on a virtual bus).

Στην πλατφόρμα λογισμικού CUDA της NVIDIA, καθώς και στο βιομηχανοποιημένο πρότυπο OpenCL πλαίσιο, τα υπολογιστικά στοιχεία των αλγορίθμων είναι γνωστά ως πυρήνες (kernels) (ένας όρος που υιοθετήθηκε εδώ από τη χρήση του στην επεξεργασία σήματος και όχι από τα λειτουργικά συστήματα). Μια εφαρμογή (application) ή μια λειτουργία βιβλιοθήκης (library function) μπορεί να αποτελείται από ένα ή περισσότερους πυρήνες (kernels). Οι πυρήνες μπορεί να γραφτούν στη γλώσσα C (συγκεκριμένα, την ANSI-standard C99 dialect) επεκτεινόμενη με επιπρόσθετες λέξεις κλειδιά για να εκφράσουν τον παραλληλισμό άμεσα και όχι μέσω των συνηθισμένων κατασκευών βρόχων (looping).

Εφόσον μεταγλωττισθούν, οι πυρήνες (kernels) αποτελούνται από πολλά νήματα που εκτελούν το ίδιο πρόγραμμα παράλληλα: ένα νήμα είναι σαν μια επανάληψη βρόχου. Σε ένα αλγόριθμο επεξεργασίας εικόνας, για παράδειγμα, ένα νήμα μπορεί να λειτουργεί σε ένα εικονοστοιχείο, ενώ όλα τα νήματα μαζί – ο πυρήνας (kernel) – μπορεί να λειτουργεί σε ολόκληρη την εικόνα.

Πολλαπλά νήματα ομαδοποιούνται σε μπλοκ νημάτων που περιέχουν μέχρι 1,536 νήματα. Όλα τα νήματα σε ένα νηματικό μπλοκ θα τρέχουν σε ένα μόνο SM, έτσι μέσα στο νηματικό μπλοκ, τα νήματα μπορούν να συνεργάζονται και να μοιράζονται τη μνήμη. Τα νηματικά μπλοκ μπορούν να συντονίσουν τη χρήση της "global shared" μνήμης ανάμεσά τους, αλλά μπορεί να εκτελούν με οποιαδήποτε σειρά, ταυτόχρονα ή διαδοχικά.

Τα νηματικά μπλοκ είναι χωρισμένα σε στημόνια (warps) των 32 νημάτων. Το στημόνι είναι η θεμελιώδης μονάδα αποστολής μέσα σε ένα ενιαίο SM. Στη Fermi δύο στημόνια από διαφορετικά νηματικά μπλοκ (ακόμη και διαφορετικούς πυρήνες) μπορούν να εκδίδονται και να εκτελούνται ταυτόχρονα , αυξάνοντας τη χρησιμοποίηση του εξοπλισμού (hardware) και την ενεργειακή απόδοση.

Τα μπλοκ νημάτων είναι ομαδοποιημένα σε πλέγματα (grids), καθένα από τα οποία εκτελεί ένα μοναδικό πυρήνα.

Τα μπλοκ νημάτων και τα νήματα έχουν αναγνωριστικά στοιχεία (IDs) που καθορίζουν τη σχέση τους με τον πυρήνα. Αυτά τα αναγνωριστικά χρησιμοποιούνται μέσα σε κάθε νήμα σαν δείκτες στα αντίστοιχα δεδομένα εισόδου και εξόδου, στις θέσεις "shared" μνήμης και ούτω καθεξής.

Ανά πάσα στιγμή, ολόκληρη η συσκευή Fermi είναι αφιερωμένη σε μια μόνο εφαρμογή. Όπως αναφέρθηκε πιο πάνω, μια εφαρμογή μπορεί να περιλαμβάνει πολλαπλούς πυρήνες. Η Fermi υποστηρίζει ταυτόχρονη εκτέλεση πολλαπλών πυρήνων από την ίδια εφαρμογή, κάθε πυρήνας διαμοιραζόμενος σε ένα ή περισσότερους SMs στη συσκευή. Αυτή η ικανότητα αποφεύγει την κατάσταση στην οποία ένας πυρήνας είναι ικανός μόνο να χρησιμοποιεί μέρος της συσκευής και το υπόλοιπο να μένει αχρησιμοποίητο.

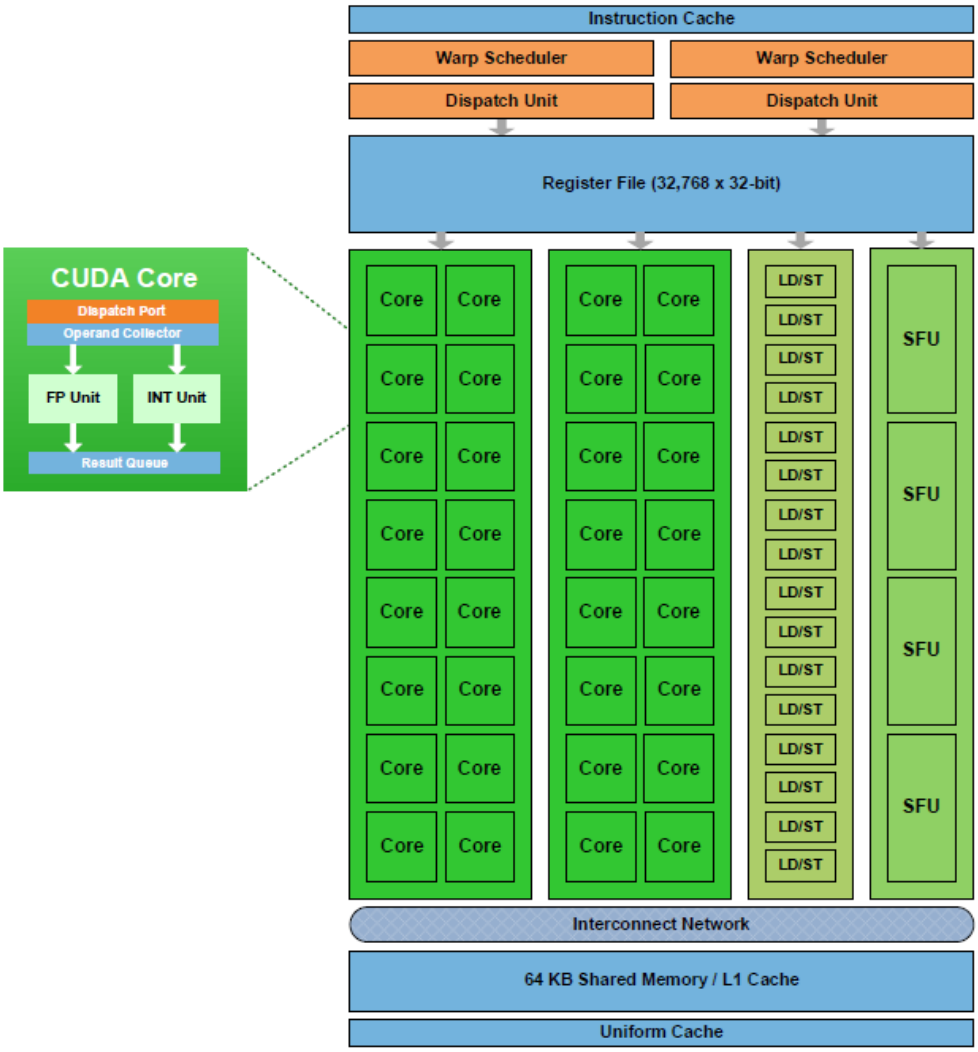
Η εναλλαγή από τη μια εφαρμογή στην άλλη είναι περίπου 20 φορές γρηγορότερη στη Fermi (ακριβώς 25 μικροδευτερόλεπτα) από ότι σε GPUs προηγούμενης γενιάς. Αυτός ο χρόνος είναι αρκετά βραχύς ώστε μια GPU Fermi μπορεί ακόμη να διατηρήσει υψηλό βαθμό χρησιμοποίησης ακόμη και όταν τρέχει πολλαπλές εφαρμογές, όπως ένα μίγμα υπολογιστικού κώδικα και κώδικα γραφικών.

Αυτή η εναλλαγή διαχειρίζεται από το chip-level GigaThread hardware thread scheduler, ο οποίος διαχειρίζεται ταυτόχρονα 1,536 ενεργά νήματα για κάθε πολυεπεξεργαστή ροής (streaming multiprocessor) σε 16 πυρήνες (kernels).

Αυτός ο κεντρικός προγραμματιστής (centralized scheduler) είναι άλλο ένα σημείο εξόδου από το συμβατικό σχεδιασμό CPU. Σε ένα διακομιστή (server) πολυπύρηνο (multicore) ή πολυεπεξεργαστή (multiprocessor) καμιά CPU δεν είναι υπεύθυνη ("in charge"). Όλες οι εργασίες συμπεριλαμβανομένου του λειτουργικού συστήματος του ίδιου του πυρήνα (kernel), μπορεί να τρέχουν σε οποιαδήποτε διαθέσιμη CPU. Με τις Fermi, οι επιδιωκόμενες εφαρμογές, οι αρχές της επεξεργασίας ροής και το μοντέλο πυρήνα (kernel) και νήματος, ήταν όλα γνωστά εκ των προτέρων, έτσι ώστε μια περισσότερο αποδοτική μέθοδος προγραμματισμού θα μπορούσε να εφαρμοστεί στη μηχανή GigaThread.

Ο Πολυεπεξεργαστής Ροής - The Streaming Multiprocessor

Οι πολυεπεξεργαστές ροής Fermi, που φαίνονται στο Σχήμα 4.7, περιλαμβάνουν 32 πυρήνες, κάθε ένας από αυτούς μπορεί να εκτελεί πράξεις κινητής υποδιαστολής (floating - point) και ακεραίων (integer), μαζί με 16 μονάδες "load-store" για λειτουργίες μνήμης, τέσσερις μονάδες "special-function"και 64K τοπικούς SRAM που μοιράζονται μεταξύ "cache" και "local" μνήμης.



Σχήμα 4.7 : Κάθε Πολυεπεξεργαστής Ροής Fermi περιλαμβάνει, 32 πυρήνες, 16 μονάδες "load/store", 4 μονάδες "special-function", ένα αρχείο μητρώου 32K – λέξεων, 64K διαμορφώσιμης RAM, και "thread control logic". Κάθε πυρήνας έχει και τις δύο μονάδες εκτέλεσης "floating-point"και "integer" (Πηγή: NVidia Corporation)

Οι πράξεις "floating-point" ακολουθούν το πρότυπο IEEE 754-2008 floating-point. Κάθε πυρήνας (core) μπορεί να εκτελέσει μία "single-precision fused multiply-add" λειτουργία σε κάθε περίοδο ρολογιού και μία "double-precision FMA" σε δύο περιόδους ρολογιού. Στο επίπεδο chip, η Fermi εκτελεί περισσότερες από 8x "double-precision" λειτουργίες ανά ρολόι παρά η προηγούμενη γενιά GT200, όπου η επεξεργασία "double-precision" τύγχανε χειρισμού από μια ειδική μονάδα ανά SM με πολύ χαμηλότερη απόδοση.

Η συμμόρφωση προς το "IEEE floating-point" περιλαμβάνει τέσσερις τρόπους στρογγυλοποίησης, και οι "subnormal numbers" (αριθμοί πιο κοντά στο μηδέν από ότι

μια κανονικοποιημένη μορφή μπορεί να αναπαριστά) τυγχάνουν ορθού χειρισμού από τον εξοπλισμό (hardware) Fermi παρά να ξεπλυθούν στο μηδέν (being flushed to zero) ή να απαιτήσουν επιπρόσθετη επεξεργασία σε ένα χειριστή εξαιρέσεων λογισμικού (software exception handler).

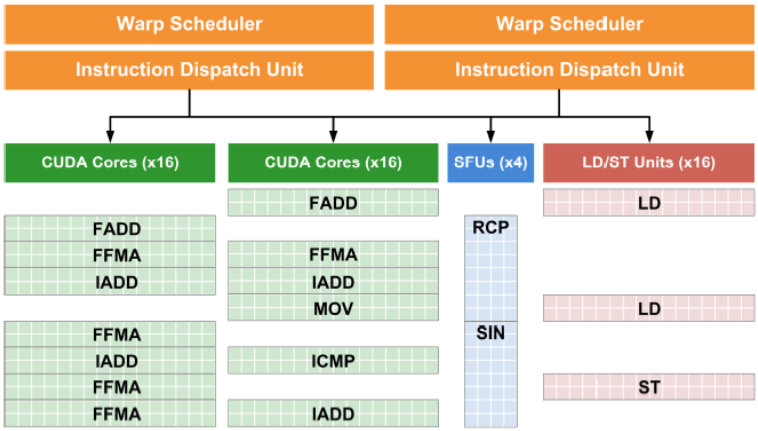
Η υποστήριξη FMA (Fused multiply-add) αυξάνει επίσης την ακρίβεια και την απόδοση άλλων μαθηματικών πράξεων όπως η διαίρεση και η τετραγωνική ρίζα, και πιο σύνθετες λειτουργίες όπως "extended precision arithmetic", "interval arithmetic" και γραμμική άλγεβρα (linear algebra).

Ο ακέραιος ALU υποστηρίζει τις συνήθεις μαθηματικές και λογικές πράξεις, συμπεριλαμβανομένου του πολλαπλασιασμού και στις δύο τιμές 32-bit και 64-bit.

Σε κάθε πολυεπεξεργαστή ροής (SM) οι λειτουργίες μνήμης τυγχάνουν χειρισμού από ένα σύνολο 16 μονάδων "load-store". Οι εντολές "load-store" μπορούν τώρα να αναφερθούν στη μνήμη από την άποψη δισδιάστατων συστοιχιών, παρέχοντας διευθύνσεις από πλευράς τιμών x και y. Τα δεδομένα μπορούν να μετατραπούν από μία μορφή σε άλλη (π.χ. από ακέραια σε "floating-point" ή αντίστροφα) καθώς περνούν μεταξύ της DRAM και των καταχωρητών του πυρήνα (core registers) σε πλήρη ρυθμό. Αυτά τα χαρακτηριστικά μορφοποίησης και μετατροπής είναι περαιτέρω παραδείγματα βελτιστοποιήσεων μοναδικά στις GPUs – όχι άξια λόγου στις γενικής χρήσης CPUs, αλλά εδώ θα χρησιμοποιηθούν αρκετά συχνά για να δικαιολογήσουν τη συμπερίληψή τους.

Ένα σύνολο τεσσάρων Μονάδων Ειδικής Λειτουργίας (Special Function Units - SFUs) είναι επίσης διαθέσιμο για να χειρισθεί υπερβατικές και άλλες ειδικές λειτουργίες, όπως το ημίτονο (sin), συνημίτονο (cos), εκθετικές (exp) και αντίστροφες (rcp). Τέσσερις από αυτές τις πράξεις μπορούν να εκδοθούν ανά κύκλο σε κάθε SM.

Μέσα στον SM, οι πυρήνες (cores) χωρίζονται σε δύο μπλοκ εκτέλεσης (execution blocks) με 16 πυρήνες στο καθένα. Μαζί με την ομάδα των 16 "load-store" μονάδων και των τεσσάρων SFUs, υπάρχουν τέσσερα μπλοκ εκτέλεσης ανά SM. Σε κάθε κύκλο, ένα σύνολο 32 εντολών μπορούν να διεκπεραιωθούν από ένα ή δύο στημόνια (warps) σε αυτά τα μπλοκ. Χρειάζονται 2 κύκλοι για τις 32 οδηγίες στο κάθε στημόνι να εκτελεστούν στους πυρήνες (cores) ή στις "load/store" μονάδες. Ένα στημόνι 32 "special-function" οδηγιών εκδίδεται σε ένα μόνο κύκλο, αλλά χρειάζεται οκτώ κύκλους να ολοκληρωθεί στις τέσσερις SFUs. Το Σχήμα 4.8 δείχνει μια ακολουθία οδηγιών κατανεμημένη μεταξύ των διαθέσιμων μπλοκ εκτέλεσης.



Σχήμα 4.8 : Ένα σύνολο 32 οδηγιών από ένα ή δύο στημόνια μπορούν να αποσταλούν σε κάθε κύκλο σε οποιαδήποτε δύο από τα τέσσερα μπλοκ εκτέλεσης μέσα σε ένα Fermi SM: δύο μπλοκς από 16 πυρήνες το καθένα, ένα μπλοκ από τέσσερις "special-function" Μονάδες, και ένα μπλοκ από 16 "load/store" μονάδες. Αυτό το σχήμα δείχνει πώς εκδίδονται οι οδηγίες στα μπλοκ εκτέλεσης (Πηγή: NVidia Corporation)

Βελτιώσεις ISA - ISA Improvements

Η Fermi κάνει την αρχή με την παράλληλη εκτέλεση νήματος (Parallel Thread eXecution(PTX)) 2.0 instruction set architecture (ISA). Η PTX 2.0 ορίζει ένα σετ οδηγιών και μια νέα αρχιτεκτονική εικονικής μηχανής (virtual machine) που ισοδυναμεί με ένα εξιδανικευμένο επεξεργαστή σχεδιασμένο για παράλληλη λειτουργία νήματος.

Επειδή αυτό το μοντέλο εικονικής μηχανής δεν μοντελοποιεί κυριολεκτικά τον εξοπλισμό (hardware) Fermi, μπορεί να μεταφέρεται από τη μια γενιά στην επόμενη. Η NVIDIA σκοπεύει το PTX 2.0 να καλύψει πολλαπλές γενιές υλικού GPU και πολλαπλά μεγέθη GPU σε κάθε γενιά, ακριβώς όπως έκανε το PTX 1.0.

Όλες από τις αρχιτεκτονικά ορατές βελτιώσεις στο Fermi αντιπροσωπεύονται στο PTX 2.0. Η βεβαίωση (predication) είναι μια από τις περισσότερο σημαντικές βελτιώσεις στο νέο ISA.

Όλες οι οδηγίες υποστηρίζουν "predication". Κάθε εντολή μπορεί να εκτελεστεί ή να παραληφθεί με βάση τους κώδικες υπό όρους "condition codes". Το "predication" επιτρέπει σε κάθε νήμα - κάθε πυρήνα (core) – να εκτελέσει διάφορες λειτουργίες ανάλογα με τις ανάγκες, ενώ η εκτέλεση συνεχίζει σε πλήρη ταχύτητα. Εκεί όπου το "predication" δεν είναι επαρκές, η Fermi υποστηρίζει επίσης τη συνηθισμένη "if – then – else" δομή με δηλώσεις κλάδου (branch statements).

Οι περισσότεροι CPUs βασίζονται αποκλειστικά σε κλάδους υπό συνθήκη (conditional branches) και ενσωματώνουν υλικό (hardware) πρόβλεψης κλάδου (branch-prediction) για να επιτρέψει εικασία κατά μήκος της πιθανής πορείας (likely path). Αυτή

είναι μια εύλογη λύση για "branch – intensive serial code", αλλά λιγότερο αποδοτική από το "predication" για εφαρμογές ροής (streaming applications).

Μια άλλη σημαντική βελτίωση στις Fermi και PTX 2.0 είναι ένα νέο μοντέλο ενοποιημένης διευθυνσιοδότησης. Όλες οι διευθύνσεις στη GPU παραχωρούνται από ένα συνεχή 40-bit (ένα terabyte) χώρο διευθύνσεων (address space). Οι global, shared και local διευθύνσεις ορίζονται ως περιοχές μέσα σε αυτό το χώρο διευθύνσεων και μπορεί να προσεγγισθεί με κοινές οδηγίες "load/store". (Οι οδηγίες "load/store" υποστηρίζουν διευθύνσεις 64-bit για να καταστήσουν δυνατή τη μελλοντική ανάπτυξη).

Η Cache και η Ιεραρχία Μνήμης - The Cache and Memory Hierarchy

Όπως προγενέστερες GPUs, η αρχιτεκτονική Fermi προβλέπει τη "local" μνήμη σε κάθε SM. Κάτι νέο στη Fermi είναι η ικανότητα να χρησιμοποιήσει κάποια από αυτή τη "local" μνήμη σαν "first-level (L1) cache" για αναφορές "global" μνήμης. Η "local" μνήμη είναι μεγέθους 64K, και μπορεί να χωρισθεί 16K/48K ή 48K/16K ανάμεσα στην L1 cache και τη "shared" μνήμη.

Η ''shared" μνήμη, η παραδοσιακή χρήση για τη "local SM" μνήμη, παρέχει "low- latency" πρόσβαση σε μέτριες ποσότητες δεδομένων (όπως ενδιάμεσα αποτελέσματα σε μια σειρά από υπολογισμούς, μια γραμμή ή στήλη δεδομένων για πράξεις πίνακα (matrix), μια γραμμή βίντεο, κλπ). Επειδή η καθυστέρηση πρόσβασης (access latency) σε αυτή τη μνήμη είναι επίσης απολύτως προβλέψιμη, μπορεί να γραφούν αλγόριθμοι για να παρεμβληθούν φορτία, υπολογισμοί και αποθήκες (stores) με μέγιστη αποτελεσματικότητα.

Η απόφαση για εκχώρηση 16K ή 48K της "local" μνήμης σαν "cache" συνήθως εξαρτάται από δύο παράγοντες: πόση "shared" μνήμη χρειάζεται και πόσο προβλέψιμες είναι πιθανόν να είναι οι προσβάσεις του πυρήνα (kernel) στη "global" μνήμη (συνήθως η "off – chip DRAM"). Μια μεγαλύτερη ανάγκη για "shared-memory" απαιτεί λιγότερη "cache". Πιο συχνές ή απρόβλεπτες προσβάσεις σε μεγαλύτερες περιοχές της DRAM απαιτούν περισσότερη "cache".

Κάθε Fermi GPU είναι επίσης εξοπλισμένη με μια L2 cache (768 KB σε μέγεθος για ένα 512-core chip). Η L2 cache καλύπτει την GPU local DRAM όπως επίσης τη μνήμη του συστήματος.

Το υποσύστημα L2 cache εφαρμόζει επίσης ένα άλλο χαρακτηριστικό που δεν βρίσκεται στους CPUs: ένα σύνολο λειτουργιών μνήμης read-modify – write που είναι ατομικές – δηλαδή αδιάλειπτες – και έτσι είναι ιδανικές για τη διαχείριση της πρόσβασης σε δεδομένα που πρέπει να μοιράζονται ανάμεσα σε νηματικά μπλοκ ή ακόμη πυρήνες (kernels). Κανονικά αυτή η λειτουργία παρέχεται μέσω μιας διαδικασίας δύο σταδίων:

Μία CPU χρησιμοποιεί μια ατομική εντολή test – and – set για να διαχειρίζεται ένα σημαφόρο (semaphore) και ο σημαφόρος διαχειρίζεται την πρόσβαση σε μια προκαθορισμένη τοποθεσία ή περιοχή στη μνήμη.

Η Fermi μπορεί να υλοποιήσει την ίδια λύση όταν χρειάζεται, αλλά είναι πολύ πιο απλό από την πλευρά του λογισμικού να μπορεί να εκδώσει μια τυποποιημένη λειτουργία ακεραίου ALU (a standard integer ALU operation) που να αποδίδει την ατομική λειτουργία (atomic operation) απευθείας, παρά να περιμένει μέχρι να γίνει διαθέσιμος ένας σημαφόρος.

Οι ατομικές πράξεις Fermi (Fermi's atomic operations) εφαρμόζονται με μια σειρά ακεραίων ALUs λογικά (a set of integer ALUs logically) που μπορεί να κλειδώσει την πρόσβαση σε μια μόνο διεύθυνση μνήμης, ενώ η ακολουθία read – modify - write έχει ολοκληρωθεί. Αυτή η διεύθυνση μνήμης μπορεί να είναι στη μνήμη του συστήματος, στην τοπικά συνδεδεμένη DRAM του GPU ή ακόμη στο χώρο μνήμης άλλων PCI Express συνδεδεμένων συσκευών. Κατά το σύντομο διάστημα κλειδώματος, το υπόλοιπο της μνήμης συνεχίζει να λειτουργεί κανονικά. Κλειδώματα στη μνήμη του συστήματος είναι ατομικά σε σχέση με τις λειτουργίες της GPU που εκτελούσαν την ατομική λειτουργία· συγχρονισμός λογισμικού συνήθως χρησιμοποιείται για να εκχωρεί περιοχές μνήμης στον έλεγχο της GPU, αποφεύγοντας έτσι αντιφατικές γραφές από τη CPU ή άλλες συσκευές.

Το τελικό στάδιο της ιεραρχίας "local" μνήμης είναι η DRAM που συνδέεται απευθείας με τη GPU. Η Fermi παρέχει έξι 64-bit DRAM κανάλια που υποστηρίζουν SDDR3 και GDDR5 DRAMs. Μέχρι 6GB της GDDR5 DRAM μπορούν να συνδεθούν στο chip για μια σημαντική ώθηση της χωρητικότητας (capacity) και του εύρους ζώνης (bandwidth) σε σχέση με προηγούμενα προϊόντα της NVIDIA.

Η Fermi είναι η πρώτη GPU που παρέχει ECC (Error Correcting Code – Κώδικας Διόρθωσης Σφάλματος) προστασία για DRAM· τα φάϊλς μητρώου του chip (register files), οι shared μνήμες, οι L1 και L2 caches είναι επίσης ECC προστατευμένες. Το επίπεδο προστασίας είναι γνωστό ως "SECDED: single (bit) error correction, double error detection". Το SECDED είναι το σύνηθες επίπεδο προστασίας στα περισσότερα συστήματα που είναι εξοπλισμένα με ECC.

Η προστασία ECC της Fermi για την DRAM είναι μοναδική μεταξύ των GPUs· έτσι είναι και η εφαρμογή της. Αντί για κάθε δίαυλο μνήμης 64-bit που μεταφέρει οκτώ έξτρα bits για πληροφορίες ECC, η NVIDIA έχει μια δική της λύση (και απόρρητη) για συσκευασία των bits του ECC για αποκλειστικές γραμμές μνήμης.

Υλικό μέρος για Πολυνηματικά - Hardware Multithreading

Το πλαίσιο εκτέλεσης (program counters = μετρητές προγράμματος, registers = καταχωρητές, κλπ.) για κάθε στημόνι που τυγχάνει επεξεργασίας από ένα πολυεπεξεργαστή (multiprocessor) διατηρείται on-chip κατά τη διάρκεια όλης της ζωής του στημονιού. Επομένως, η εναλλαγή από το ένα πλαίσιο εκτέλεσης σε άλλο δεν έχει κόστος, και σε κάθε στιγμή έκδοσης μιας εντολής, ένας δρομολογητής στημονιού (warp scheduler) διαλέγει ένα στημόνι που έχει νήματα έτοιμα να εκτελέσουν την επόμενη εντολή (the active threads of the warp =τα ενεργά νήματα του στημονιού) και εκδίδει την εντολή σε εκείνα τα νήματα.

Ειδικότερα, κάθε πολυεπεξεργαστής έχει ένα σετ από καταχωρητές 32-bit που είναι χωρισμένοι μεταξύ των στημονιών, και μια παράλληλη κρυφή μνήμη δεδομένων (parallel data cache) ή κοινόχρηστη μνήμη (shared memory) που είναι χωρισμένη μεταξύ των μπλοκ των νημάτων.

Ο αριθμός των μπλοκ και στημονιών που μπορούν να διαμένουν και να τυγχάνουν επεξεργασίας από κοινού στον πολυεπεξεργαστή για ένα δεδομένο πυρήνα (for a given kernel) εξαρτάται από την ποσότητα των καταχωρητών και την κοινόχρηστη μνήμη (shared memory) που χρησιμοποιείται από τον πυρήνα και τη διαθέσιμη ποσότητα καταχωρητών και κοινόχρηστη (shared) μνήμη στον πολυεπεξεργαστή. Υπάρχει επίσης ένας μέγιστος αριθμός μπλοκ που διαμένουν και ένας μέγιστος αριθμός διαμενόντων στημονιών ανά πολυεπεξεργαστή. Αυτά τα όρια καθώς και η ποσότητα των καταχωρητών και της διαθέσιμης κοινόχρηστης μνήμης στον πολυεπεξεργαστή είναι συνάρτηση της υπολογιστικής ικανότητας της συσκευής. Εάν δεν υπάρχουν αρκετοί καταχωρητές ή διαθέσιμη κοινόχρηστη μνήμη ανά πολυεπεξεργαστή, για να επεξεργασθεί τουλάχιστον ένα μπλοκ, ο πυρήνας (kernel) θα αποτύχει να ξεκινήσει.

Ο συνολικός αριθμός στημονιών σε ένα μπλοκ Wblock έχει ως εξής:

$$W_{block} = ceil(\frac{T}{W_{size}}, 1)$$

- T είναι ο αριθμός νημάτων ανά μπλοκ,
- W_{size} είναι το μέγεθος του στημονιού, το οποίο είναι ίσο με 32,
- ceil(x,y) είναι ίσο με x στρογγυλοποιημένο προς το πλησιέστερο πολλαπλάσιο του y.

Ο ολικός αριθμός των καταχωρητών R_{block} που παραχωρείται για ένα μπλοκ έχει ως εξής:

Για συσκευές υπολογιστικής ικανότητας 1.x:

$$R_{block} = ceil\ \ ceil\ W_{block}, G_W\ \cdot W_{size} \cdot R_k, G_T$$

Για συσκευές υπολογιστικής ικανότητας 2.x:

$$R_{block} = ceil\ \ R_k \cdot W_{size}\ \ \cdot W_{block}$$

- G_W είναι οι λεπτομέρειες κατανομής στημονιών, ίσο με 2 (υπολογιστική ικανότητα 1.x μόνο),
- R_k είναι ο αριθμός καταχωρητών που χρησιμοποιείται από τον πυρήνα
- G_T είναι οι λεπτομέρειες κατανομής νημάτων, ίσο με 256 για συσκευές υπολογιστικής ικανότητας 1.0 και 1.1, 512 για συσκευές υπολογιστικής ικανότητας 1.2 και 1.3, και 64 για συσκευές υπολογιστικής ικανότητας 2.x.

Η ολική ποσότητα κοινόχρηστης (shared) μνήμης S_{block} σε bytes που διατίθεται για ένα μπλοκ έχει ως ακολούθως:

$$Size_{block} = ceil(S_k, G_s)$$

- S_k είναι η ποσότητα κοινόχρηστης (shared) μνήμης που χρησιμοποιείται από τον πυρήνα σε bytes,
- G_s είναι οι λεπτομέρειες κατανομής της κοινόχρηστης μνήμης, η οποία ισούται με 512 για συσκευές υπολογιστικής ικανότητας 1.x και 128 για συσκευές υπολογιστικής ικανότητας 2.x.

Απόδοση Εγγενών Αριθμητικών Εντολών στις GPUs.

Πίνακας 4.1 : Απόδοση Εγγενών Αριθμητικών Εντολών (Λειτουργίες ανά Κύκλο Ρολογιού ανά Πολυεπεξεργαστή)

	Υπολογιστική Ικανότητα 1.x	Υπολογιστική Ικανότητα 2.0	Υπολογιστική Ικανότητα 2.1
32-bit floating-point add, multiply, multiply-add	8	32	48
64-bit floating-point add, multiply, multiply-add	1	16	4
32-bit integer add, logical operation	8	32	48
32-bit integer shift, compare	8	16	16
32-bit integer multiply, multiply-add, sum of absolute difference	Multiple instructions	16	16
24-bit integer multiply (**__[u]mul24**)	8	Multiple instructions	Multiple instructions
32-bit floating-point reciprocal, reciprocal square root, base-2 logarithm (**__log2f**), base-2 exponential (**exp2f**), sine (**__sinf**), cosine (**__cosf**)	2	4	8
Type conversions	8	16	16

Μετατροπή Τύπου - Type Conversion

Μερικές φορές, ο μεταγλωττιστής πρέπει να εισάγει οδηγίες μετατροπής, εισάγοντας επιπρόσθετους κύκλους εκτέλεσης. Αυτό ισχύει για:

- Συναρτήσεις που λειτουργούν σε μεταβλητές τύπου **char** ή **short** των οποίων γενικά οι τελεστές χρειάζεται να μετατραπούν σε **int**,
- Διπλής ακρίβειας κινητής υποδιαστολής σταθερές (δηλαδή εκείνες οι σταθερές που ορίζονται χωρίς καμιά κατάληξη τύπου) που χρησιμοποιούνται σαν input σε υπολογισμούς single-precision floating – point (σύμφωνα με την εντολή προτύπων (C/C++).

Αυτή η τελευταία περίπτωση μπορεί να αποφευχθεί με τη χρήση single-precision κινητής υποδιαστολής σταθερών, που ορίζονται με μια κατάληξη **f** όπως, *2502.1987***f**, *3004.1968***f**, *3.141592653589793***f**.

Αριθμητική Ακεραίων - Integer Arithmetic

Σε συσκευές υπολογιστικής ικανότητας 1.x, ένας 32-bit ακέραιος πολλαπλασιασμός υλοποιείται με τη χρήση πολλαπλών εντολών καθόσον δεν υποστηρίζεται εγγενώς. Ωστόσο ένας 24-bit ακέραιος πολλαπλασιασμός υποστηρίζεται εγγενώς μέσω του εσωτερικού (intrinsic) **_[u]mul24**. Η χρήση του **_[u]mul24** αντί για 32-bit τελεστή πολλαπλασιασμού, όπου είναι δυνατό συνήθως βελτιώνει την απόδοση για πυρήνες δεσμευμένους με εντολές. Ωστόσο, μπορεί να έχει το αντίθετο αποτέλεσμα σε περιπτώσεις, όπου η χρήση του **_[u]mul24** αναστέλλει βελτιστοποιήσεις μεταγλωττιστή.

Σε συσκευές με υπολογιστική ικανότητα 2.x, ο 32-bit ακέραιος πολλαπλασιασμός υποστηρίζεται εγγενώς, αλλά ο 24-bit ακέραιος πολλαπλασιασμός δεν υποστηρίζεται. Επομένως, η **_[u]mul24** υλοποιείται με τη χρήση πολλαπλών οδηγιών και δεν θα έπρεπε να χρησιμοποιείται.

Η ακέραιη διαίρεση κι η λειτουργία modulo είναι δαπανηρές: δεκάδες εντολές σε συσκευές υπολογιστικής ικανότητας 1.x, κάτω από 20 εντολές σε συσκευές με υπολογιστική ικανότητα 2.x. Αυτές μπορούν να αντικατασταθούν με "bitwise" πράξεις σε μερικές περιπτώσεις: Αν το **n** είναι μια δύναμη του 2, **(i/n)** είναι ισοδύναμο με **(i>>log$_2$(n))** και **(i%n)** είναι ισοδύναμο με**(i&(n-1))**· ο μεταγλωττιστής θα εκτελέσει αυτές τις μετατροπές, αν το **n** είναι κυριολεκτικός (literal).

Οι **_brev**, **_brevll**, **_popc**, και **_popcll** μεταγλωττίζουν σε δεκάδες εντολών για συσκευές με υπολογιστική ικανότητα 1.x, αλλά οι **_brev** και **_popc** χαρτογραφούν (map to) σε μια μόνο εντολή για συσκευές με υπολογιστική ικανότητα 2.x και οι -**_brevll** και **_popcll** σε λίγες μόνο.

Εντολές Ελέγχου Ροής - Control Flow Instructions

Οποιαδήποτε εντολή ελέγχου ροής (**if**, **switch**, **do**, **for**, **while**) μπορεί να επηρεάσει σημαντικά την αποτελεσματική απόδοση της εντολής προκαλώντας απόκλιση νημάτων του ίδιου στημονιού (δηλαδή να ακολουθούν διαφορετικές διαδρομές εκτέλεσης). Αν συμβεί αυτό, οι διάφορες διαδρομές εκτελέσεων πρέπει να είναι σε συνέχειες, αυξάνοντας το συνολικό αριθμό εντολών που εκτελούνται για αυτό το στημόνι. Όταν όλες οι διαφορετικές διαδρομές εκτέλεσης έχουν συμπληρωθεί, τα νήματα συγκλίνουν πίσω στο ίδιο μονοπάτι εκτέλεσης.

Για να επιτύχουμε τη βέλτιστη απόδοση σε περιπτώσεις, όπου η ροή ελέγχου εξαρτάται από την ταυτότητα του νήματος (threadID), η ελέγχουσα κατάσταση θα πρέπει να γραφτεί έτσι ώστε να ελαχιστοποιεί τον αριθμό των αποκλινόντων στημονιών. Αυτό είναι δυνατό γιατί η κατανομή των στημονιών σε όλο το μπλοκ είναι ντετερμινιστική (deterministic). Ένα τετριμμένο παράδειγμα είναι όταν η ελέγχουσα κατάσταση εξαρτάται μόνον από (**threadIdx / warpSize**), όπου **warpSize** είναι το μέγεθος του στημονιού. Σε αυτές τις περιπτώσεις, κανένα στημόνι δεν αποκλίνει καθόσον η κατάσταση ελέγχου είναι τέλεια ευθυγραμμισμένη με τα στημόνια.

Εύρεση της κυκλικής αναπαράστασης των Posets σε GPUs.

Για την εύρεση της κυκλικής αναπαράστασης των Posets σε GPUs υλοποιήθηκαν δύο προγράμματα βασισμένα στον αλγόριθμο των Bayon, Lygeros, Sereni.

1) Υβριδικό Μοντέλο :

Το πρώτο πρόγραμμα χρησιμοποιεί την CPU και την GPU για υπολογισμό. Ο υπολογισμός της αξιολόγησης (του fitness) γίνεται μέσω κλήσης ενός πυρήνα της GPU. Ο πυρήνας επιστρέφει το ταίριασμα (fitness). Στο διάγραμμα ροής 3.6 που παρουσιάζει τον υπολογισμό μιας γενιάς του αλγορίθμου μπορούν να διακριθούν οι τρείς αξιολογήσεις.

Μελετήθηκαν δύο ισόμορφοι τρόποι υπολογισμού του ταιριάσματος (fitness) του γενετικού αλγορίθμου με τη χρήση GPU.

Τρόπος A : Με πολλαπλασιασμό και εκτέλεση αμέσως των πράξεων.

Τρόπος B : Αποθήκευση του πρότυπου Poset και του τωρινού Poset σε καταχωρητή 64 bit και το αποτέλεσμα βγαίνει μέσω λίγων λογικών πράξεων.

Αποτελέσματα:

Τα αποτελέσματα μεταξύ αυτών των δύο τρόπων χρονικά είναι πανομοιότυπα. Στη συνέχεια παρατίθεται ο κώδικας των δύο τρόπων.

```
for(i = 0 ; i < N ; i++){
        for(j = i+1 ; j < N ; j++){

           d_circle.z  =  circles [j].z   - circles [i].z;
           d_circle.x  =  circles [i].x  -  circles [j].x;
           d_circle.y  =  circles [i].y  -  circles [j].y;

           d    = d_circle.x * d_circle.x + d_circle.y * d_circle.y;
           dr2 = d_circle.z  *  d_circle.z;
// ---------------------------------------------------------------------
            incl_val.x =  !((signbit( dr2 - d)) || (signbit(d_circle.z))) ;
           d_circle.z = - d_circle.z;
           incl_val.y =  !((signbit( dr2 - d)) || (signbit(d_circle.z))) ;

          flag  =   Poset_int_dev[0].element[c];

          semi_flag    = (flag ^ incl_val.x);
          score        +=  (( flag * (!(semi_flag)) ) -  (semi_flag)  -  ( !(flag) *  (semi_flag) ) – 2 * incl_val.y  )
                           * N +  !(semi_flag);
// ---------------------------------------------------------------------
           c++;
        }
}
```

Κώδικας 4.1: Κώδικας με πολλαπλασιαμό και εκτέλεση αμέσως των πράξεων.

```
for(i = 0 ; i < N ; i++){
        for(j = i+1 ; j < N ; j++){

          d_circle.z  =  pop[j].z   -  pop[i].z;
          d_circle.x  =  pop[i].x  -  pop[j].x;
          d_circle.y  =  pop[i].y  -  pop[j].y;

          d    =  d_circle.x * d_circle.x  +  d_circle.y * d_circle.y;
          dr2 =  d_circle.z * d_circle.z;
          flag.x = ((signbit( dr2 - d)) || (signbit(d_circle.z))) ;
          d_circle.z = - d_circle.z;
          flag.y =  ((signbit( dr2 - d)) || (signbit(d_circle.z))) ;

          flag.x  =  (flag.x-1);
          flag.y  =  (flag.y-1);

          poset_out = orbit(poset_out,( bit_def.bit[c] & flag.x));
          rev_poset_out = orbit(rev_poset_out, (bit_def.bit[c] & flag.y));
        // ----------------------
          c++;
         }
}
// ---------------------------------------------------------------------------
         score  =  - __popcll(rev_poset_out);
        score   =  score * 2;
       // ----------------------
         xor_bit_register  = xorbit(poset_out,Poset_dev[0].poset);
      // ----------------------
         and_bit_register = andbit(notbit(xor_bit_register),Poset_dev[0].poset);

     // ----------------------
       score  +=  __popcll(and_bit_register);
      // ----------------------
       score  -=  __popcll(xor_bit_register);
     // ----------------------
         and_bit_register  =  andbit(xor_bit_register,notbit(Poset_dev[0].poset));
    // ----------------------
        score  -=  __popcll(and_bit_register);
         score  = score * N;
         xor_bit_register = notbit(xor_bit_register);
// ----------------------
score  +=  __popcll(xor_bit_register) - (64-MAX_RELATION);
```

Κώδικας 4.2 : Κώδικας με χρήση πράξεων bit.

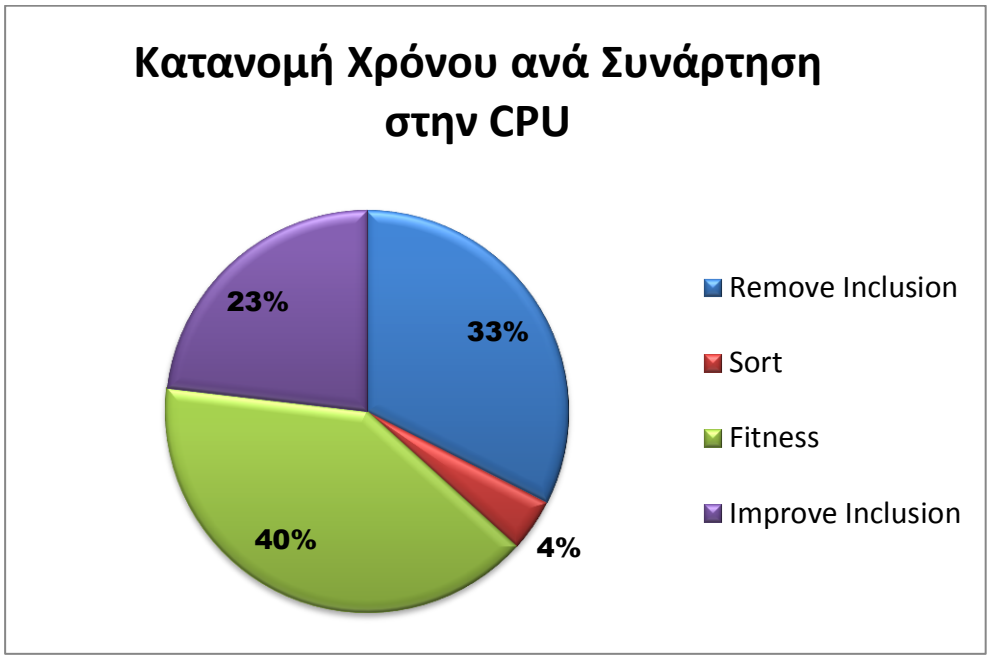
Ιδέα :

Μπορούμε να χρησιμοποιήσουμε και τις δύο μεθόδους για να επιτύχουμε μεγαλύτερη αξιοποίηση του SMD. Από το Πίνακα 4.1. (Απόδοση Εγγενών Αριθμητικών Εντολών) παρατηρούμε ότι οι πράξεις του πολλαπλασιασμού και των λογικών ακεραίων εκτελούνται ξεχωριστά. Κατά συνέπεια, μπορούμε να βάλουμε ένα Warp να εκτελεί μία μεθοδολογία και άλλο Warp την άλλη.

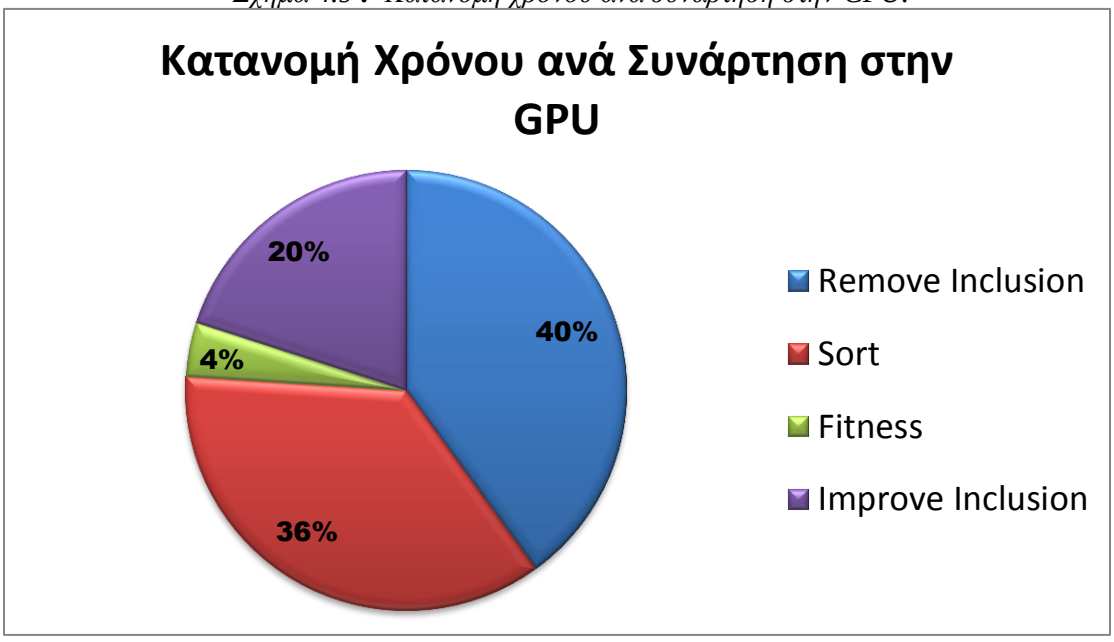
Αποτέλεσμα : Καμία ουσιαστική αύξηση της επίδοσης δεν σημειώθηκε.

2) *Μοντέλο με αποκλειστική χρήση GPU.*

Το δεύτερο πρόγραμμα χρησιμοποιεί εξολοκλήρου την GPU για εκτέλεση του γενετικού αλγορίθμου. Λόγω υπολογιστικών δυσκολιών ο γενετικός αλγόριθμος έπρεπε να τροποποιηθεί. Η κύρια τροποποίηση που έγινε είναι η εξής : δεν τίθεται ο πληθυσμός κατά φθίνουσα σειρά στην ολότητά του, αλλά για κάθε Thread Block.



Σχήμα 4.9 : Κατανομή χρόνου ανά συνάρτηση στην CPU.



Σχήμα 4.10 : Κατανομή χρόνου ανά συνάρτηση στην GPU.

Η τροποποίηση αυτή θεωρήρηκε αναγκαία για έχουμε συμβατότητα με όλες τις κάρτες αρχιτεκτονικής Fermi ανεξαρτήτος του αριθμού των πολυεπεξεργαστών που διαθέτουν.

Από τα παραπάνω σχήματα είναι εμφανής η αλλαγή στο υπολογιστικό ισοζύγιο. Εδώ αξίζει να σημειωθεί ότι στην CPU έκδοση χρησιμοποιήθηκε quick sort για στοίχιση του βαθμού. Στην GPU μία τέτοια μεθοδολογία ήταν δύσκολο να εφαρμοστεί, έτσι χρησιμοποιήθηκε η μέθοδος φυσαλίδα - Bubble sort σε παράλληλη έκδοση, γνωστή και σαν Odd-Even Transition Sorting, για τη στοίχηση του βαθμού ταιριάσματος του πληθυσμού. Αυτό έγινε με γνώμονα, το να μπορεί να γίνει η στοίχιση στην ίδια κλήση του πυρήνα. Υπήρχαν καλύτερες τεχνικές από άποψη χρονικής πολυπλοκότητας για μεγάλο αριθμό στοιχείων, αλλά θα απαιτούσε πολλαπλές εκκινήσεις πυρήνων. Θα ήταν καλύτερα αν υπήρχε περισσότερος χρόνος να εξεταστεί η τεχνική του Bitonic Merge Sorting Network.

Κεφάλαιο 5

Μελέτη Επιτάχυνσης

Στο κεφάλαιο αυτό παρατίθενται οι μετρήσεις και τα υπολογιστικά πειράματα που έγιναν. Ο σκοπός αυτών των πειραμάτων είναι η μελέτη της ανταπόκρισης του αλγόριθμού μας στις διάφορες αρχιτεκτονικές. Εξετάζουμε στην αρχή την απόδοση που έχει η CPU χωρίς τη χρήση GPU. Μελετάμε εάν η επιτάχυνση που επιτυχάνει ο αλγόριθμος είναι ανάλογος των επεξεργαστικών πόρων ή όχι. Και στη συνέχεια εξετάζουμε πως συμπεριφέρονται η CPU και η GPU μαζί σε ένα υβριδικό μοντέλο. Τέλος αναλύουμε την απόδοση του αλγορίθμου όταν λειτουργεί με GPU, μετά την εκτέλεση της πρώτης φάσης με CPU.

Σενάρια Μετρήσεων

Θέλοντας να εξετάσουμε τη συμπεριφορά του προγράμματος στις διάφορες εκδόσεις, δημιουργήσαμε μερικά σενάρια μετρήσεων. Τα σενάρια είναι τα εξής:

- Έλεγχος του παράλληλου προγράμματος που έγινε με χρήση της OPENMP σε επεξεργαστές Intel Xeon . Το μηχάνημα έχει δύο τέτοιους επεξεργαστές.
- Έλεγχος του παράλληλου προγράμματος που έγινε με χρήση της OPENMP και της CUDA. Η συνάρτηση ταιριάσματος τρέχει στην GPU, ενώ ο υπόλοιπος κώδικας στην CPU. Η κλήση της συνάρτησης γίνεται με σύγχρονο διαδοχικό τρόπο, δηλαδή η CPU περιμένει την απάντηση της GPU για να συνεχίσει.
- Έλεγχος του παράλληλου προγράμματος που έγινε με χρήση της OPENMP και της CUDA. Η συνάρτηση ταιριάσματος τρέχει στην GPU, ενώ ο υπόλοιπος κώδικας στην CPU. Η κλήση της συνάρτησης γίνεται με ασύγχρονο διαδοχικό τρόπο, δηλαδή η CPU δεν περιμένει την απάντηση της GPU για να συνεχίσει.
- Έλεγχος του παράλληλου προγράμματος που έγινε μόνο με χρήση CUDA. Η GPU εκτελεί εξ ολοκλήρου τον γενετικό αλγόριθμο με εξαίρεση την αρχική αρχικοποίηση του πληθυσμού και το τελικό κομμάτι της επιβεβαίωσης εάν οι κύκλοι που βρήκε ικανοποιούν όντως το αρχικό Poset.

Στις εκδόσεις με OPENMP ο παραλληλισμός έγινε σε επίπεδο πινάκων, δηλαδή το πακέτο επεξεργασίας περιείχε τα Posets, τα οποία έχουν ίδιο πίνακα βαθμών τους.

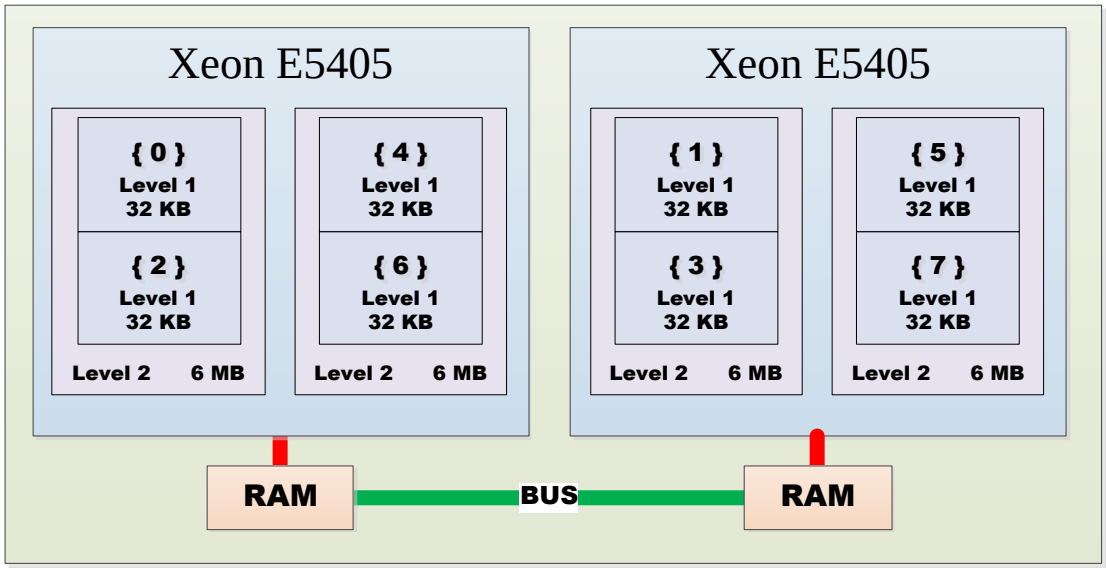
Παραδοχές :

Για να μπορέσουν να γίνουν οι μετρήσεις αυτές σε μικρή κλίμακα και να εξεταστεί η χειρότερη περίπτωση εκτέλεσης του κώδικα έγιναν μερικές παραδοχές. Πρώτον, οι πίνακες έχουν το ίδιο σύνολο Posets. Δεύτερον, ακόμα και αν βρεί λύση πρέπει να συνεχίσει την εκτέλεση θεωρώντας ότι δεν βρήκε λύση έως ότου τελειώσει ο αριθμός των προκαθορισμένων γενεών.

Σενάριο 1ο

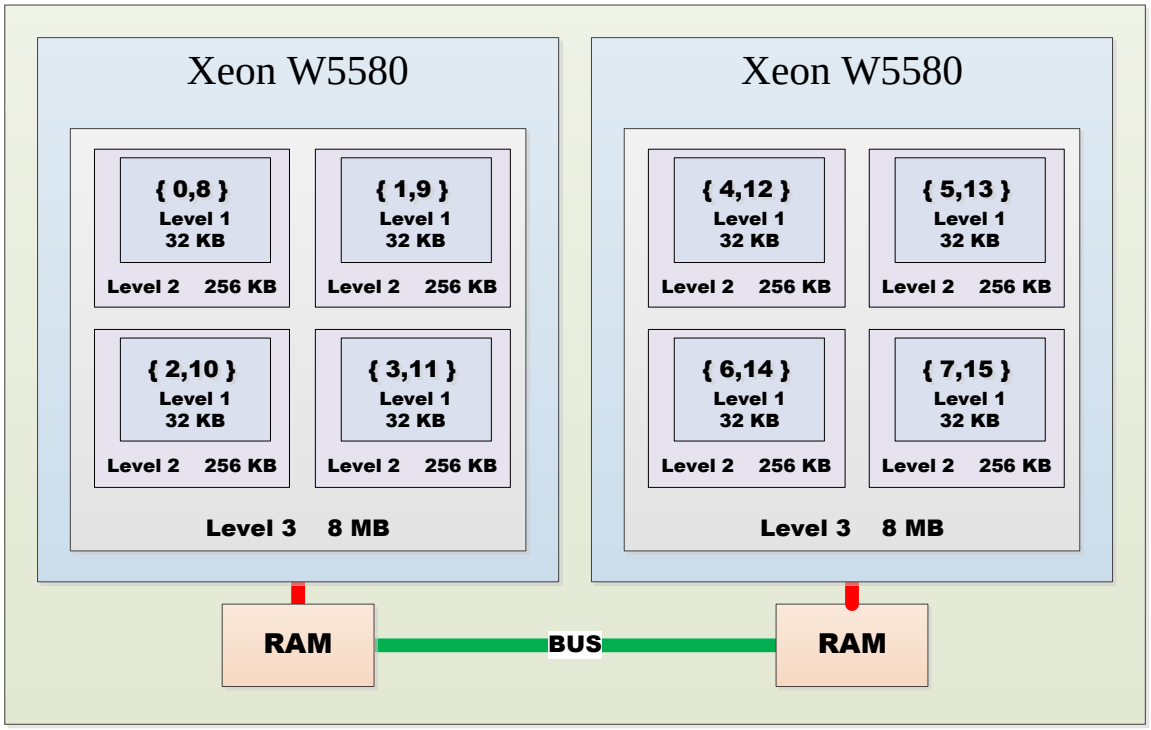
Εξέταση της απόδοσης του προγράμματος σε υπολογιστές με τα ακόλουθα χαρατκηριστικά.

Πίνακας 5.1 : Τεχνικά Χαρακτηριστικά	
Μοντέλο	Xeon E5405
"Harpertown" (45 nm) βασισμένο στην αρχιτεκτονική Intel Core.	
Πυρήνες	4
Συχνότητα	2 GHz
L1 Instruction Cache ανά Πυρήνα	32 KB
L1 Data Cache ανά Πυρήνα	32 KB
L2 Unified Cache	2 X 6MB



Σχήμα 5.1 : Σχηματική αναπαράσταση του υπολογιστή με Xeon E5405.

Πίνακας 5.2 : Τεχνικά Χαρακτηριστικά	
Μοντέλο	Xeon W5580
"Gainestown" (45 nm) βασισμένο στην αρχιτεκτονική Nehalem .	
Πυρήνες	4
Νήματα	8
Συχνότητα	2,2 GHz
L1 Instruction Cache ανά Πυρήνα	32 KB
L1 Data Cache ανά Πυρήνα	32 KB
L2 Unified Cache ανά Πυρήνα	256 KB
L3 Unified Cache	8 MB



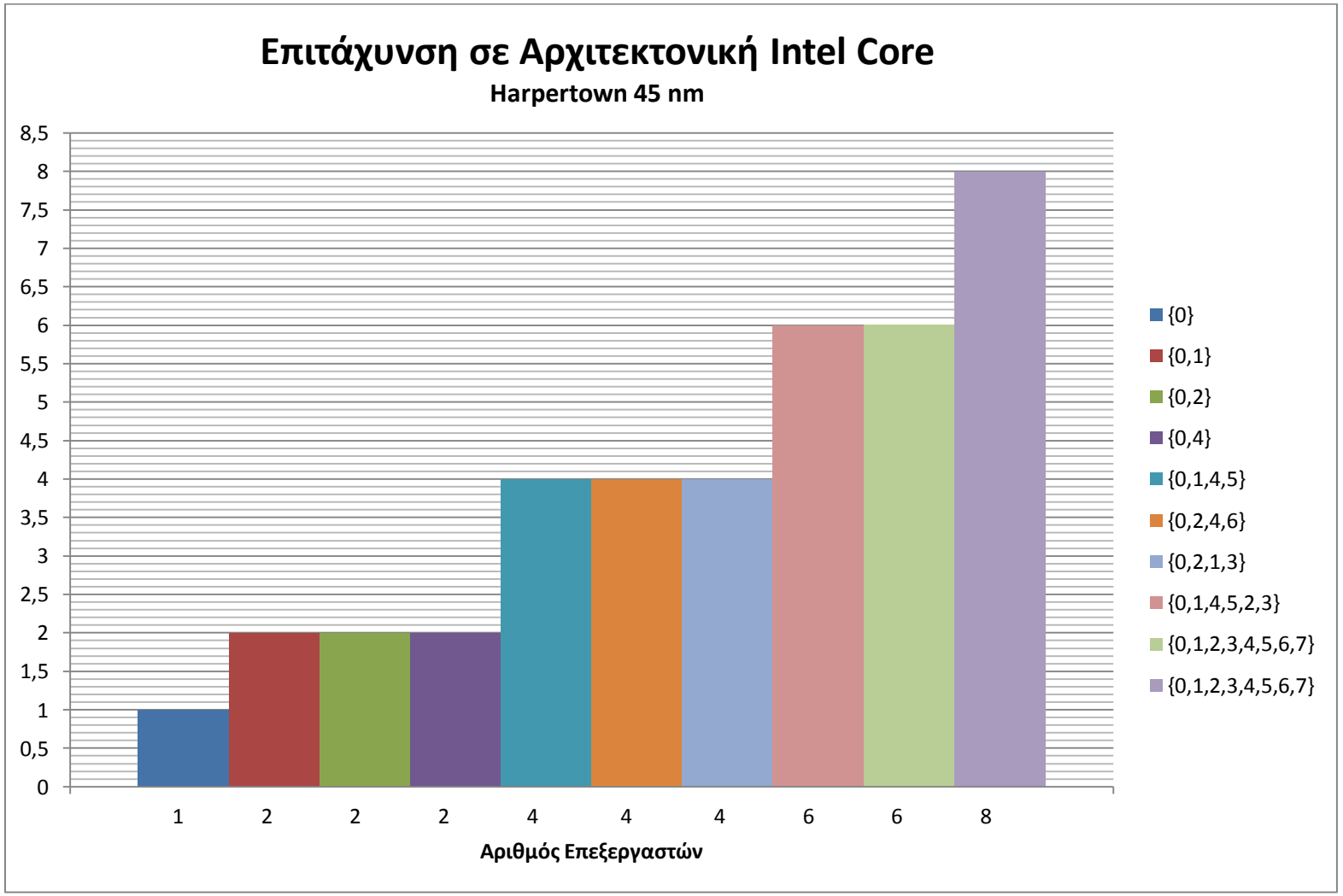
Σχήμα 5.2 : Σχηματική αναπαράσταση του υπολογιστή με Xeon W5580 .

Πίνακας 5.3 : Τεχνικά Χαρακτηριστικά	
Μοντέλο	Xeon X7460
"Dunnington" (45 nm) βασισμένο στην αρχιτεκτονική Penryn.	
Πυρήνες	12
Συχνότητα	2,66 GHz
L1 Instruction Cache ανά Πυρήνα	32 KB
L1 Data Cache ανά Πυρήνα	32 KB
L2 Unified Cache ανά Πυρήνα	3 MB
L3 Unified Cache	16 MB

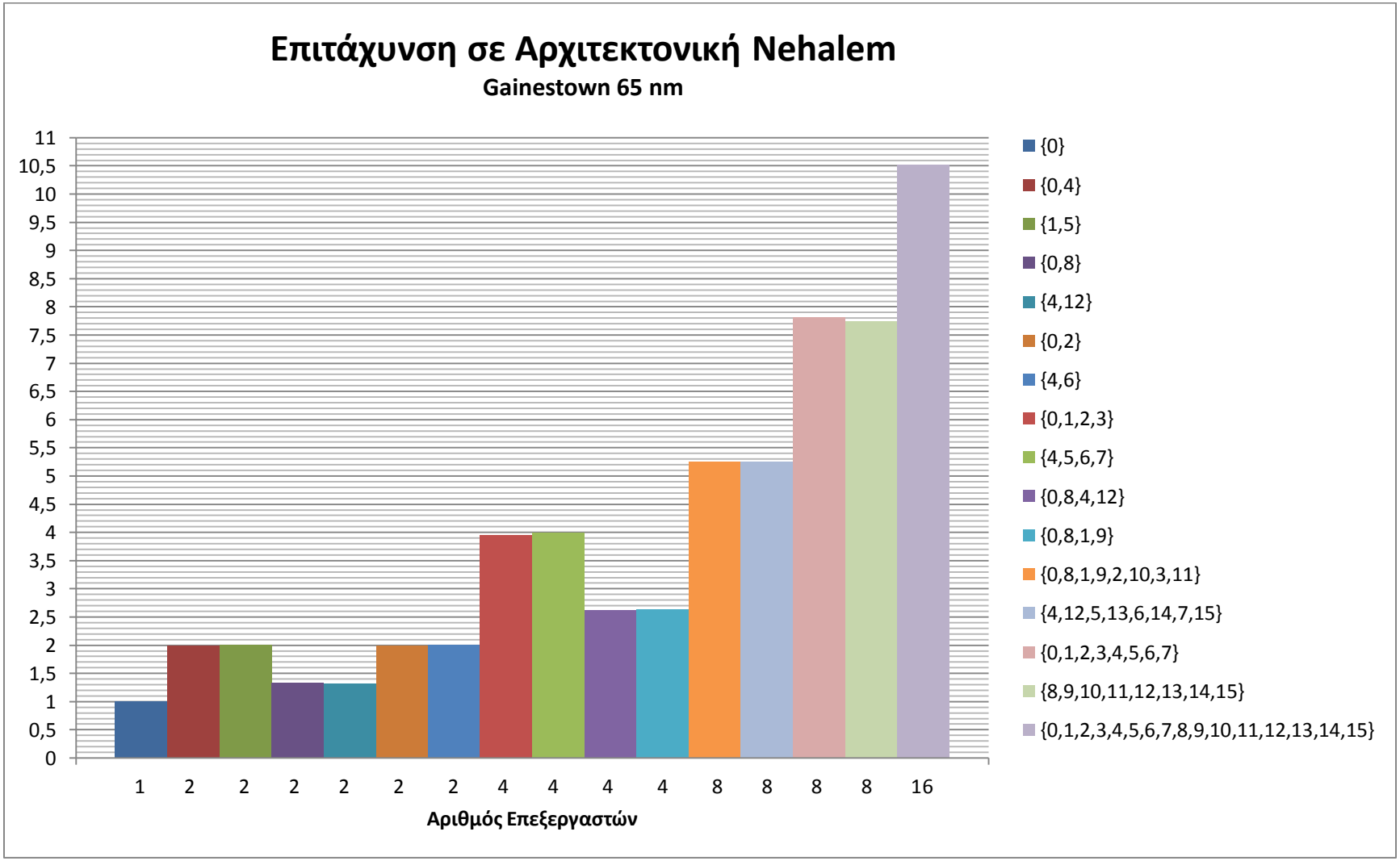
Τα σενάρια 2,3,4 πάρθηκαν από τις ακόλουθες αρχιτεκτονικές.

Πίνακας 5.4 : Τεχνικά Χαρακτηριστικά	
Μοντέλο	Opteron 180
"Denmark" (90 nm)	
Πυρήνες	2
Συχνότητα	2,4 GHz
L1 Instruction Cache ανά Πυρήνα	64 KB
L1 Data Cache ανά Πυρήνα	64 KB
L2 Unified Cache ανά Πυρήνα	1 MB
GPU	NVidia 9600 GT, 430GT

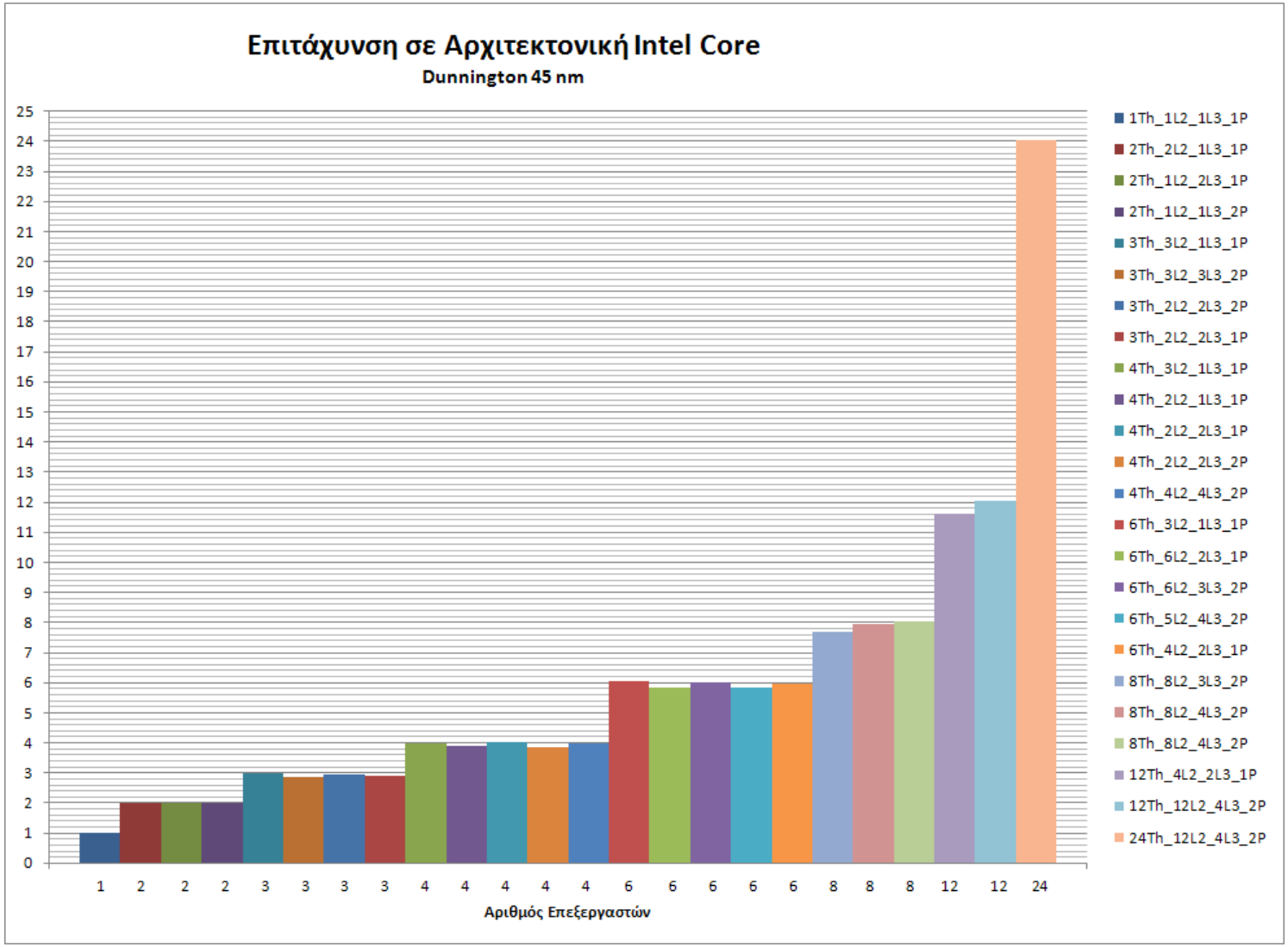
Πίνακας 5.5 : Τεχνικά Χαρακτηριστικά	
Μοντέλο	Xeon X5650
"Gulftown" (32 nm) βασισμένο στην αρχιτεκτονική Westmere .	
Πυρήνες	6
Νήματα	12
Συχνότητα	2,6 GHz
L1 Instruction Cache ανά Πυρήνα	32 KB
L1 Data Cache ανά Πυρήνα	32 KB
L2 Unified Cache ανά Πυρήνα	256 KB
L3 Unified Cache	12 MB
GPU	TESLA M2050



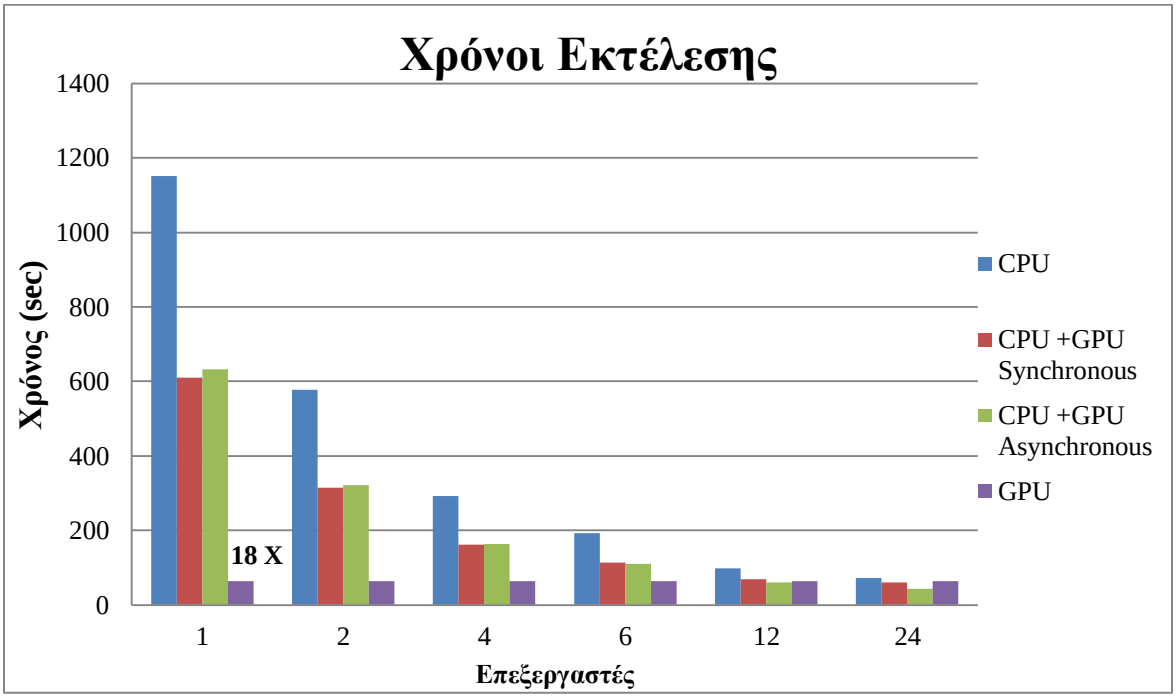
Σχήμα 5.3 : Επιτάχυνση του αλγορίθμου σε αρχιτεκτονική Penryn. Επεξεργαστής Xeon E5405



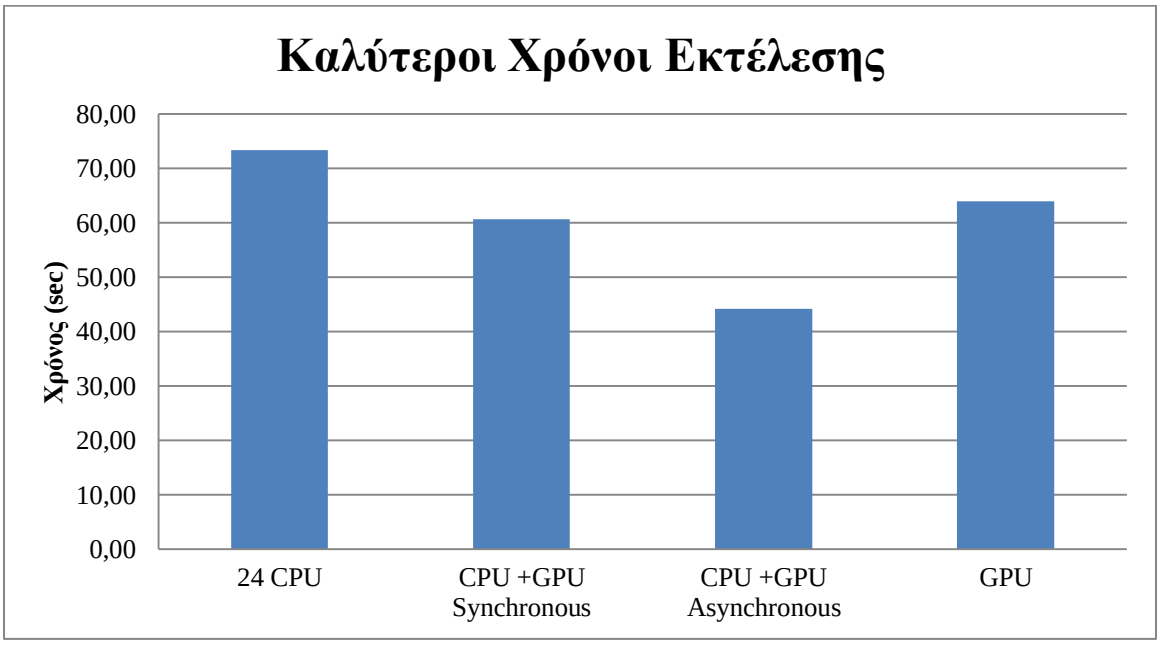
Σχήμα 5.4 : Επιτάχυνση σε Αρχιτεκτονική Nehalem. Επεξεργαστής Xeon W5580



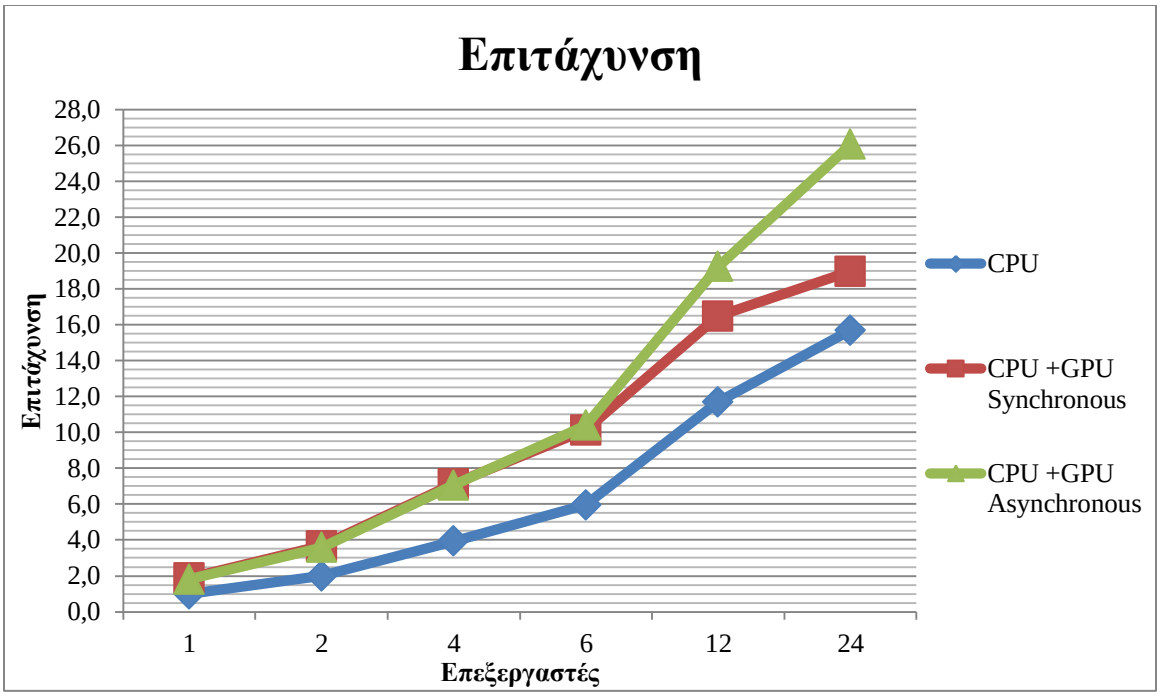
Σχήμα 5.5 : Επιτάχυνση σε Αρχιτεκτονική Penryn. Επεξεργαστής Xeon X7460



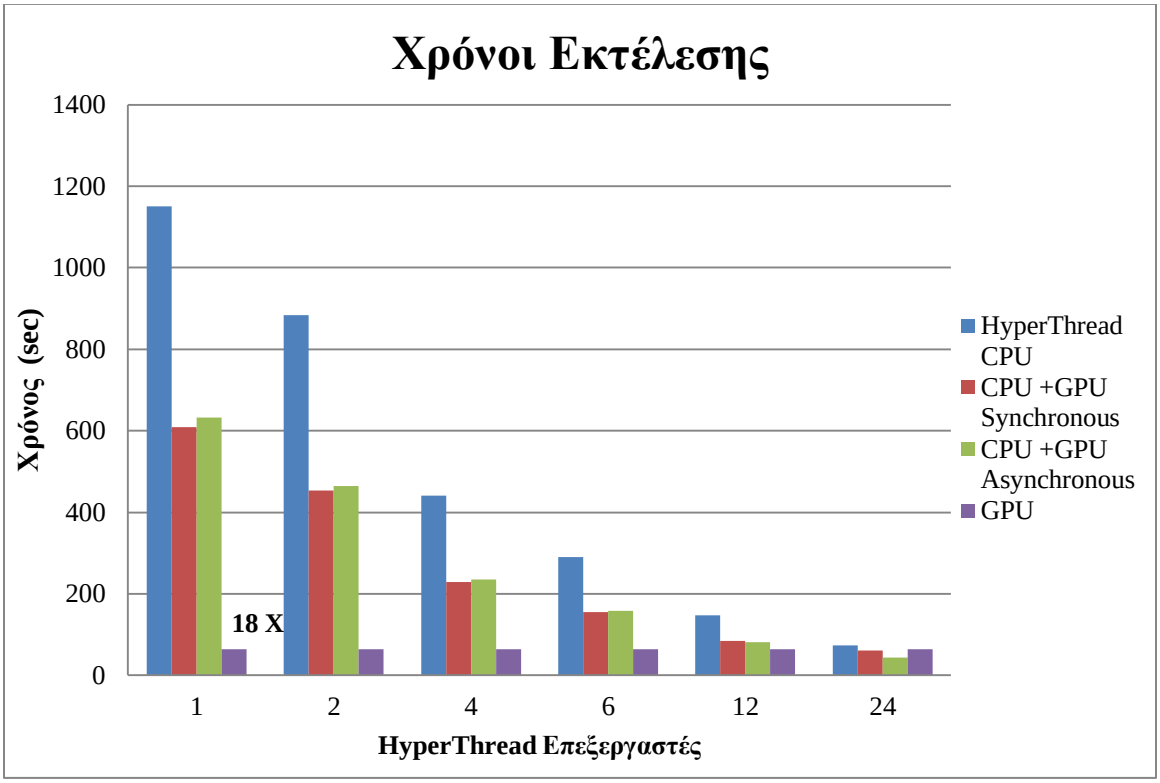
Σχήμα 5.6 : Συγκριτικοί Χρόνοι Εκτέλεσης για διάφορα πλήθει επεξεργαστών.



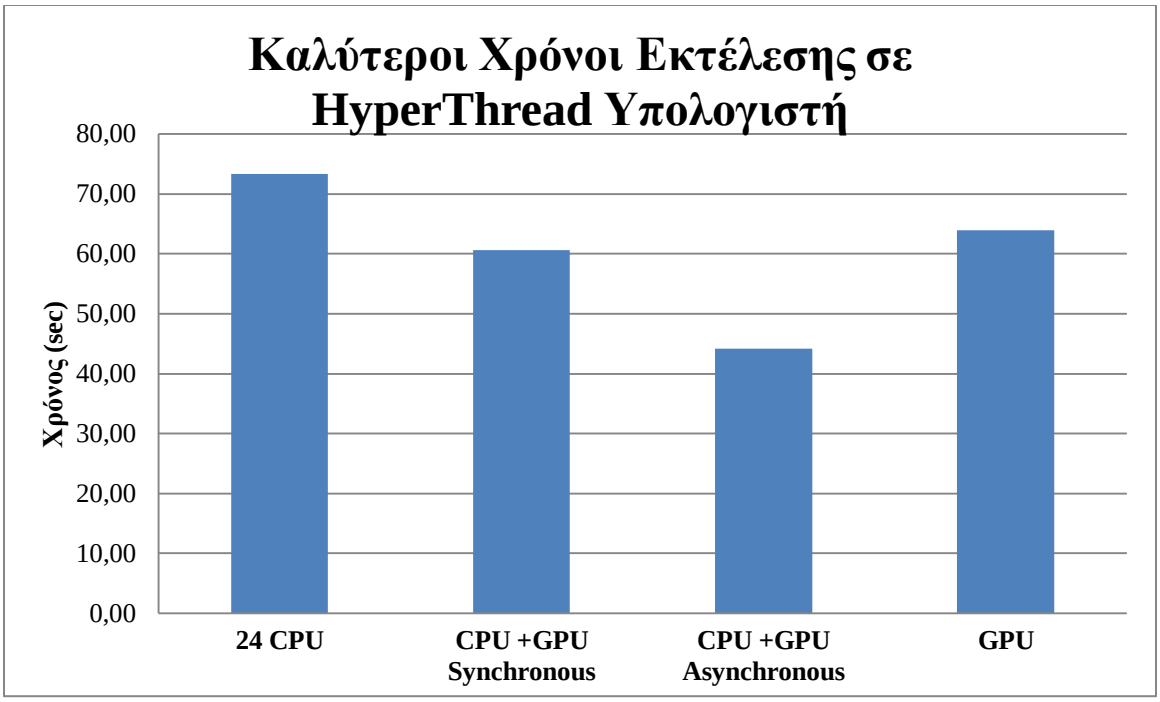
Σχήμα 5.7 : Καλύτεροι χρόνοι εκτέλεσης.



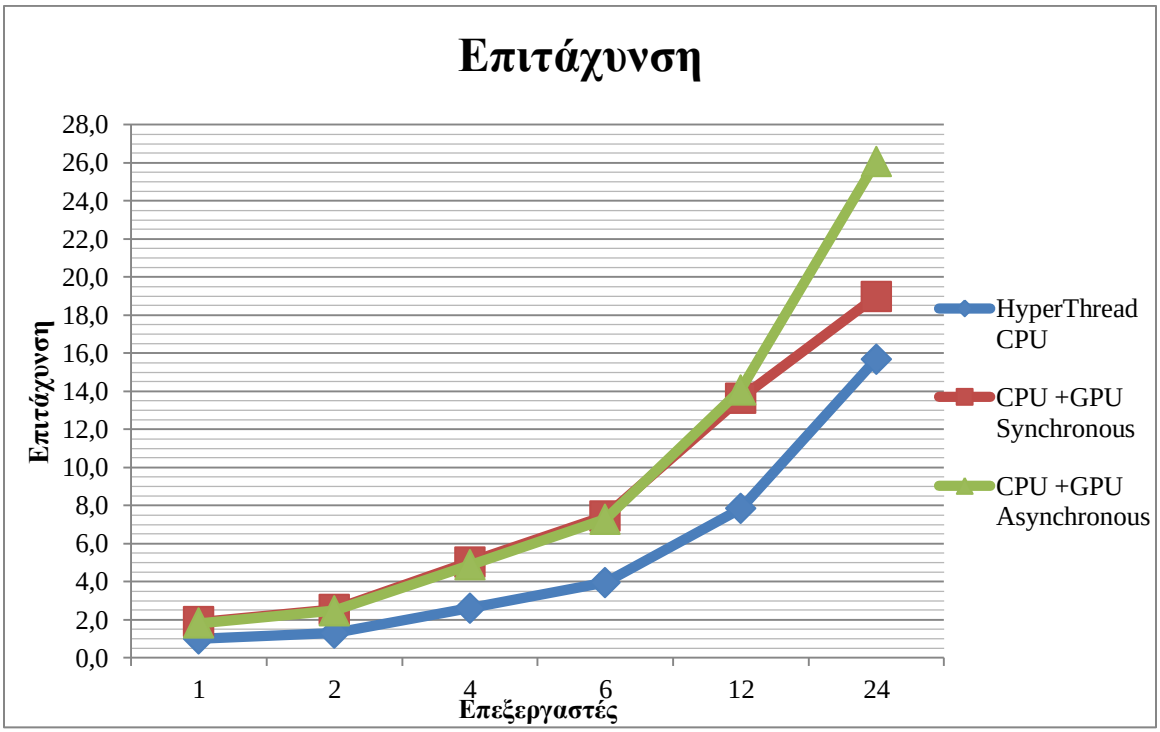
Σχήμα 5.8 : Γραφική παράσταση παρουσίασης επιταχύνσεων CPU και CPU + GPU.



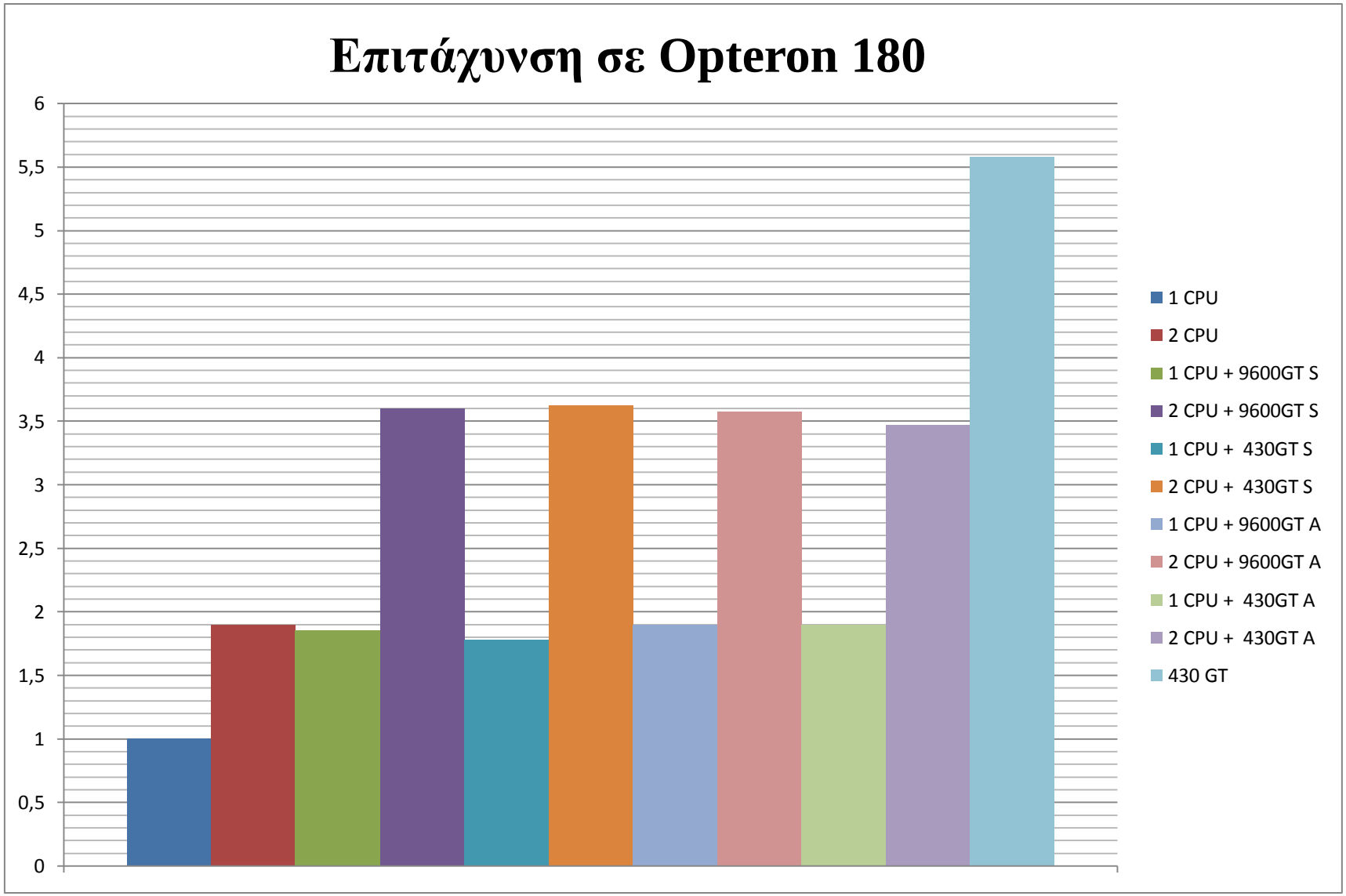
Σχήμα 5.9 : Συγκριτικοί Χρόνοι Εκτέλεσης για διάφορα πλήθη επεξεργαστών με τεχνολογία HT.



Σχήμα 5.10 : Καλύτεροι χρόνοι εκτέλεσης.



Σχήμα 5.11 : Γραφική παράσταση παρουσίασης επιταχύνσεων CPU και CPU + GPU με τεχνολογία HyperThread.



Σχήμα 5.12 : Επιταχύνσεις σε Opteron 180 και κάρτες γραφικών 9600GT και 430GT.

Κεφάλαιο 6

Συμπεράσματα

και

Μελλοντικές Κατευθύνσεις

Στο παρόν κεφάλαιο καταγράφονται τα συμπεράσματα των πειραμάτων για τη μελέτη της απόδοσης του γενετικού αλγορίθμου σε πολυπύρηνους επεξεργαστές. Επίσης, δίνονται τα συμπεράσματα που προέκυψαν από τα πειράματα σε υβριδικά συστήματα, συνδυασμένων CPU και GPU, καθώς και από πειράματα σχεδόν αποκλειστικά σε GPU. Τέλος, αναφέρονται ιδέες ως μελλοντικές κατευθύνσεις στην έρευνα.

Συμπεράσματα

Οι υλοποιήσεις που κατασκευάσαμε κλιμακώνουν αρκετά καλά, δηλαδή η αύξηση της ταχύτητας εκτέλεσης του προγράμματος αυξάνει ευθέως ανάλογα με το πλήθος των πυρήνων. Η προϋπόθεση για αυτή την κλιμάκωση είναι η κρυφή μνήμη δευτέρου ή τρίτου επιπέδου να χωρεί όλο τον πληθυσμό περίπου 800 KB.

Η χρήση υβριδικού μοντέλου οδηγεί σε σχεδόν διπλάσια ταχύτητα εκτέλεσης σε σχέση με το σειριακό. Αυτό ισχύει μέχρι και έξι επεξεργαστές, μετά από το σημείο αυτό η Tesla M2050 φτάνει σε κορεσμό. Η χρήση υβριδικού μοντέλου απαιτεί μόνο μία κάρτα γραφικών σε σχέση με το μοντέλο με σχεδόν αποκλειστική εκτέλεση σε GPU. Αυτό δίνει τη δυνατότητα να εκτελείται σε οποιοδήποτε υπολογιστή που διαθέτει κάρτα γραφικών της NVidia με υποστήριξη CUDA.

Η απόδοση του ασύγχρονου υβριδικού μοντέλου δεν είχε καμία σημαντική βελτίωση μέχρι να κορεστεί η GPU. Η εκτέλεση του ταιριάσματος με GPU έχει πολύ μικρό χρόνο υπολογισμού, αλλά η αρχιτεκτονική Fermi υποστηρίζει μέχρι και 16 ταυτόχρονες εκτελέσεις πυρήνων. Για να μπορούν να εκτελεστούν πραγματικά ταυτόχρονα θα πρέπει όταν εκτελούνται όλες μαζί να απαιτούν το πολύ όλη την υπολογιστική ισχύ, σε αντίθετη περίπτωση σειριοποιούνται. Μετά το σημείο κορεσμού το ασύγχρονο μοντέλο αρχίζει να υπερέχει έναντι του σύγχρονου. Αξίζει να σημειωθεί εδώ ότι η αποστολή του πληθυσμού στο ασύγχρονο μοντέλο γίνεται με 4 stream τα οποία αποστέλλουν διαδοχικά ανά ¼ του πληθυσμού.

Το μοντέλο με σχεδόν αποκλειστική εκτέλεση σε GPU παρουσιάζει πολύ καλή απόδοση. Για παράδειγμα, η 430GT διαθέτει μόνο 2 πολυεπεξεργαστές και είναι χαμηλών επιδόσεων. Η Tesla M2050 παρόλο που διαθέτει 16 πολυεπεξεργαστές θα περίμενε κάποιος να είναι 8 φορές πιο γρήγορη από την 430GT, όμως δεν είναι. Υπάρχουν δύο πιθανοί λόγοι :

α) Η 430GT έχει 3/2 φορές μεγαλύτερη απόδοση σε αριθμούς 32bit κινητής υποδιαστολής και ακεραίους σε σχέση με την Tesla M2050.

β) Η 430GT δεν είχε ECC μνήμη, ενώ η Tesla M2050 έχει και ήταν ενεργοποιημένη. Είναι γνωστό ότι όταν είναι ενεργοποιημένη η ECC τότε υπάρχει μείωση του ρυθμού ανταλλαγής δεδομένων.

Αν υπάρχει επιπλέον κάρτα γραφικών με υποστήριξη CUDA καλό θα είναι να δουλεύει πλήρως το γενετικό αλγόριθμο μαζί με ένα επεξεργαστή και οι υπόλοιποι επεξεργαστές θα εκτελούν ανεξάρτητα το δικό τους μερίδιο του υπολογισμού.

Θεωρούμε πλέον απαραίτητη την εφαρμογή της υλοποίησής μας σε ένα δίκτυο υπολογιστών μεγάλης εμβέλειας.

Μελλοντικές Κατευθύνσεις

1) Εξέλιξη του προγράμματος έτσι ώστε να εκτελείται στην πλατφόρμα BOINC.

2) Αναθεμελίωση της συνάρτησης Fitness του γενετικού αλγορίθμου, έτσι ώστε να ενσωματώνει μία fuzzy μεθοδολογία.

3) Η μελέτη των συνόλων μερικής διάταξης ύψους 2 είναι σημαντική στη θεωρία διάταξης λόγω της ακόλουθης ιδέας:

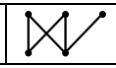
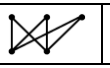
Γενικά ένας γράφος αποτελείται από κορυφές και ακμές. Είναι δυνατόν να κωδικοποιήσουμε ένα τέτοιο γράφο μ' ένα σύνολο μερικής διάταξης ύψους 2, με τον επόμενο τρόπο, ο οποίος κατασκευάζει και έναν ισομορφισμό. Όλες οι κορυφές ερμηνεύονται ως άτομα του συνόλου μερικής διάταξης και οι ακμές ως συνάτομα. Οι σχέσεις καθορίζονται μεταξύ ατόμων και συνατόμων μέσω της κάθε ακμής που ενώνει δύο κορυφές. Γι' αυτό το λόγο μελετούμε τα σύνολα μερικής διάταξης ύψους 2.

Το 1991, οι Hazim-Sharif και Lygeros απαρίθμησαν τα σύνολα μερικής διάταξης έως την τάξη 11 με το πάχος κι ανάλογα με το ύψος τους. Τα σύνολα μερικής διάταξης ύψους 2, όσον αφορά στο πλήθος τους είναι τα εξής:

N = 2	P = 1	N = 7	P = 163
N = 3	P = 3	N = 8	P = 556
N = 4	P = 8	N = 9	P = 2222
N = 5	P = 20	N = 10	P = 10765
N = 6	P = 55	N = 11	P = 64955

Πίνακας 0.1: Πίνακας με πλήθος Posets με ύψος 2.

Και τα διαγράμματα Hasse έως την τάξη 5 είναι τα εξής:

	Relations																			
N	1	2			3			4						5		6	
2																	
3																	
4																	
5																	

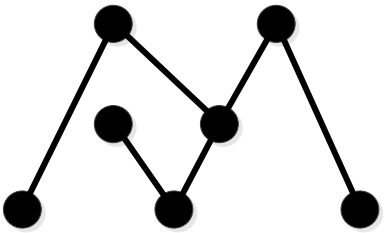
Πίνακας 0.2 Διαγράμματα Posets με ύψος 2.

Μέσω του θεωρήματος του Hiraguchi του 1955 γνωρίζουμε ότι η διάταξη του είναι μικρότερη του 2. Και μέσω του θεωρήματος Dushnik-Miller του 1941, ξέρουμε ότι όλα έχουν μία κυκλική αναπαράσταση.

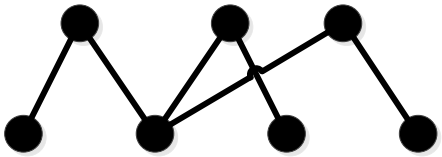
Το ίδιο ισχύει για τις εξής οικογένειες συνόλων μερικής διάταξης C_n,S_n, $K_{n,n}$. Καθώς αυτά τα σύνολα ανήκουν στο σύνολο της μελέτης μας λόγω των S_n το κριτήριο της θεωρίας διάταξης δεν επαρκεί. Στο πλαίσιο των συνόλων μερικής διάταξης ύψους 2, ανήκουν και τα δέντρα. Αυτά ερμηνεύονται ως τέτοια σύνολα, δίχως μεταβατικές σχέσεις, όπως είναι κι όλα τα σύνολα ύψους 2. Κατά συνέπεια τίθεται το ερώτημα αν τα δέντρα έχουν κυκλική αναπαράσταση. Χάρις στο θεώρημα του Trotter το 1974, έχουμε το εξής αποτέλεσμα. Όλα τα δέντρα μερικής διάταξης έχουν μέγιστη διάσταση 3. Και συνεπώς έχουμε δύο κατηγορίες δέντρων T.
$\dim(T) \leq 2$, τα οποία έχουν κυκλική αναπαράσταση λόγω των Dushnik-Miller.

Οι Sidney – Sidney – Urrutia το 1988 απέδειξαν ότι ο αριθμός διασταύρωσης ενός συνόλου μερικής διάταξης, διάστασης n είναι μικρότερος ή ισούται με n-1. Την ίδια χρονιά έκαναν την εξής εικασία: όλα τα σύνολα μερικής διάταξης με αριθμό διασταύρωσης 2, έχουν κυκλική αναπαράσταση από την οποία συνεπάγεται ότι όλα τα σύνολα μερικής διάταξης με διάσταση 3, έχουν κυκλική αναπαράσταση. Όμως το 1999, οι Fischburn, Flesner και Trotter ανακάλυψαν την ύπαρξη ενός πεπερασμένου συνόλου μερικής διάταξης με διάσταση 3, το οποίο να μην έχει κυκλική αναπαράσταση. Κατά συνέπεια, οι δύο εικασίες είναι λανθασμένες.

Ένας τρόπος για να λύσουμε το πρόβλημά μας περί δέντρων είναι να τα βυθίσουμε στο πλαίσιο του Trotter ο οποίος γενικεύει τα κλασσικά δέντρα με την έννοια του δέντρου στο διάγραμμα Hasse. Απέδειξε ότι τα δέντρα έχουν διάσταση 2 αν και μόνο αν δεν εμπεριέχουν ως υποσύνολα μερικής διάταξης τα εξής σύνολα J_1 και J_2



J_1



J_2

Ή τα δυικά τους J_1 και J_2.

Κατά συνέπεια τα δέντρα ύψους 2 δεν εμπεριέχουν ποτέ J_1 ή J_1. Έτσι το πρόβλημα περιορίζεται στα δέντρα που εμπεριέχουν J_2 ή J_2.

Αναφορές

[1] G. Brightwell and P. Winkler, Sphere orders. Order, 6(3) (1989), 235–240.

[2] C. Chaunier and N. Lygeros, The number of orders with thirteen elements. Order, 9(3) (1992), 203–204.

[3] C. Chaunier and N. Lygeros, Le nombre de posets à isomorphie près ayant 12 éléments. Theoret. Comput. Sci., 123(1) (1994), 89–94. Number theory, combinatorics and applications to computer science (Marseille, 1991).

[4] C. Chaunier and N. Lygeros, Posets minimaux ayant un groupe d'automorphismes d'ordre premier. C.R.A.S.Paris, t.318, s.I, p.695-698, 4/94

[5] T.H.Cormen , C.E. Leiserson , R.L.Rivest , C. Stein, Εισαγωγή στους Αλγορίθμους. Πανεπιστημιακές Εκδόσεις Κρήτης (2006)

[6] B. Dushnik and E. W. Miller, Partially ordered sets. Amer. J. Math., 63 (1941), 600–610.

[7] S. Felsner, P. C. Fishburn, and W. T. Trotter, Finite three-dimensional partial orders which are not sphere orders. Discrete Math., 201(1-3) (1999), 101–132.

[8] P. C. Fishburn and W. T. Trotter, Geometric containment orders: a survey. Order, 15(2) (1998/99), 167–182.

[9] R. Fraissé and N. Lygeros, Petits posets: dénombrement, représentabilité par cercles et "compenseurs". C. R. Acad. Sci. Paris Sér. I Math., 313(7) (1991), 417–420.

[10] M. C. Golumbic, Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York, 1980.

[11] W. Hazim-Sharif, N. Lygeros, Unimodalité des petits posets. Singularité, vol 2 n8 p.7-8 1991

[12] T. Hiraguchi, On the dimension of partially ordered sets. Sci. Rep. Kanazawa Univ., 1 (1951), 77–94. 22 Orders with Ten Elements

[13] G. Hulbert, A short proof that N3 is not a circle containment order. Order, 5 (1988), 235–237.

[14] N. Lygeros, Calculs exhaustifs sur les posets d'au plus 7 éléments. Singularité, vol2 n4 p.10-24, 04/1991

[15] N. Lygeros, Panorama: Fractals et Posets en Relativité. An. de la Fondation L. de Broglie, vol.19 n4, p.273-290, 12/94.

[16] N. Lygeros, M. Mizony, Construction de posets dont le groupe d'automorphismes est isomorphe à un groupe donné. C.R.A.S.Paris, t.322, s.I, p.203-206, 96.

[17] N. Lygeros, Répercussions globales d'un impact local dans une structure hiérarchisée. Perfection vol.3 2, 2/2002

[18] Επαλήθευση της απόδειξης του Καραθεοδωρή περι ανεξαρτησίας των αξιωμάτων της θεωρίας των **Somas** μέσω της θεωρίας των συνόλων μερικής διάταξης. Perfection vol.8 9, 10/2007

[19] Ν. Λυγερός , Ν. Χατζηγεωργίου, Ικανή συνθήκη περί δίαστασης 3 για τα σύνολα μερικής διάταξης. Perfection 12 7 7/2011

[20] Ν. Λυγερός , Ν. Χατζηγεωργίου, Κατασκευή συνόλων μερικής διάταξης τάξης n με κυκλική αναπαράσταση. Perfection 12 7 7/2011

[21] Ν. Λυγερός , Ν. Χατζηγεωργίου, Κυκλική αναπαράσταση και σύνολα μερικής διάταξης ύψους 2. Perfection 12 7 7/2011

[22] E. R. Scheinerman and J. C. Wierman, On circle containment orders. Order, 4(4) (1988), 315–318.

[23] J. B. Sidney, S. J. Sidney, and J. Urrutia, Circle orders, *n*-gon orders and the crossing number. Order, 5(1) (1988), 1–10.

[24] W. T. Trotter, Dimension of the crown S_n^K. Disc. Math. 8, 85-103 (1974)

[25] W.T. Trotter, Inequalities in dimension theory for posets. Proceedings of the American Mathematical Society, Volume 47, Number 2, February, 1975

[26] J. Urrutia, Partial orders and Euclidean geometry. In Algorithms and Order (Ottawa, ON, 1987), pages 387–434. Kluwer Acad. Publ., Dordrecht, 1989

Αναφορές στο Διαδίκτυο

[1] http://mathworld.wolfram.com/GraphMinor.html

[2] http://mathworld.wolfram.com/Robertson-SeymourTheorem.html

[3] http://oeis.org/A000112/graph

[4] http://oeis.org/A000112/list

[5] http://oeis.org/A001035

[6] http://plato.stanford.edu/entries/tarski/

[7] http://plato.stanford.edu/entries/tarski-truth/

[8] http://plato.stanford.edu/entries/consequence-algebraic/

[9] http://plato.stanford.edu/entries/philosophy-mathematics/

[10] http://plato.stanford.edu/entries/logic-provability/

[11] http://plato.stanford.edu/entries/category-theory/

[12] http://www.math.nthu.edu.tw/~amen/2007/060209-3.pdf

[13] http://mathworld.wolfram.com/PeanoArithmetic.html

[14] http://mathworld.wolfram.com/PeanosAxioms.html

[15] http://www.maths.bris.ac.uk/~maxng/thesis.pdf

[16] http://www.math.tu-berlin.de/~felsner/Paper/oldim.pdf

[17] http://www.vuse.vanderbilt.edu/~spin/open.html

[18] http://arxiv.org/abs/math/0602226

[19] http://dimacs.rutgers.edu/Workshops/Lattices/Markowsky.pdf

[20] http://www.mit.edu/~parrilo/pubs/talkfiles/Parrilo-ISL.pdf

[21] http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=777890

[22] http://www.cds.caltech.edu/~murray/VaVmuri/images/3/31/Sp08-cdc.pdf

[23] http://www.liafa.jussieu.fr/~pandit/Ancestry-Tight.pdf

[24] http://www.jucs.org/jucs_10_10/architectural_abstraction_as_transformation/Denford_M.pdf

[25] http://www.nvidia.com/object/GPU_Computing.html

[26] http://boinc.berkeley.edu/