ΑΝΑΛΥΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΠΕΙΡΑΜΑΤΩΝ ΑΛΓΟΡΙΘΜΟΙ ΕΛΑΧΙΣΤΗΣ ΔΙΑΔΡΟΜΗΣ

Ομάδα: Νικόλαος Ρούφας - inf2024146

Κωνσταντίνος Τζώρτζης - inf2024168

Νικόλαος Λειβαδιώτης - inf2024101

Εργασία: Δομές Δεδομένων - Β΄ Εξάμηνο

Ημερομηνία: 09/06/2025

ΠΕΡΙΓΡΑΦΗ ΔΙΑΓΡΑΜΜΑΤΟΣ

Το παραγόμενο διάγραμμα απεικονίζει τους χρόνους εκτέλεσης των πέντε αλγορίθμων που υλοποιήσαμε. Στον οριζόντιο άξονα φαίνεται ο αριθμός κόμβων (10, 20, 30, 40, 50) και στον κατακόρυφο ο χρόνος σε δευτερόλεπτα με λογαριθμική κλίμακα.

Οι καμπύλες αντιστοιχούν στις εξής υλοποιήσεις:

- Μπλε γραμμή: Dijkstra με πίνακα γειτνίασης
- Κόκκινη γραμμή: Dijkstra με λίστα γειτνίασης
- Πράσινη γραμμή: Bellman-Ford με πίνακα γειτνίασης
- Πορτοκαλί γραμμή: Bellman-Ford με λίστα γειτνίασης
- Μωβ γραμμή: Bellman-Ford με λίστα ακμών

ΠΑΡΑΤΗΡΗΣΕΙΣ ΑΝΑ ΑΛΓΟΡΙΘΜΟ

Dijkstra με Πίνακα Γειτνίασης

Παρατηρήσαμε ότι αυτή η υλοποίηση δίνει σταθερά αποτελέσματα για όλα τα μεγέθη γράφων. Ο χρόνος εκτέλεσης αυξάνεται προβλέψιμα σύμφωνα με την αναμενόμενη πολυπλοκότητα $O(V^2)$. Για 10 κόμβους χρειάστηκε περίπου 0.001 δευτερόλεπτα, ενώ για 50 κόμβους έφτασε τα 0.058 δευτερόλεπτα.

Dijkstra με Λίστα Γειτνίασης

Αυτή η υλοποίηση αποδείχτηκε η ταχύτερη συνολικά. Η χρήση του min-heap φαίνεται να βοηθάει σημαντικά την απόδοση, ιδιαίτερα για μεγαλύτερους γράφους. Το γεγονός ότι εξετάζει μόνο τους πραγματικούς γείτονες κάθε κόμβου την κάνει αποδοτικότερη από την έκδοση με πίνακα.

Bellman-Ford με Πίνακα Γειτνίασης

Εδώ βλέπουμε τη χαρακτηριστική αργή συμπεριφορά του Bellman-Ford. Η διαφορά με τον Dijkstra γίνεται εμφανής ήδη από τους 20 κόμβους. Για 50 κόμβους ο χρόνος φτάνει τα 4.12 δευτερόλεπτα, που είναι περίπου 70 φορές αργότερα από τον καλύτερο Dijkstra.

Bellman-Ford με Λίστα Γειτνίασης

Παρόλο που παραμένει αργός, αυτή η υλοποίηση δείχνει βελτίωση σε σχέση με τον πίνακα γειτνίασης. Η οικονομία στη μνήμη και η καλύτερη πρόσβαση στους γείτονες μειώνουν τον συνολικό χρόνο κατά περίπου 30%.

Bellman-Ford με Λίστα Ακμών

Όπως αναμενόταν, αυτή η υλοποίηση έδωσε τα χειρότερα αποτελέσματα. Το γεγονός ότι πρέπει να εξετάζει όλες τις ακμές σε κάθε επανάληψη τη καθιστά ιδιαίτερα αργή. Για 50 κόμβους χρειάστηκε πάνω από 5 δευτερόλεπτα.

ΑΝΑΛΥΣΗ ΒΑΣΕΙ ΠΛΗΘΟΥΣ ΚΟΜΒΩΝ

Μικροί Γράφοι (10-20 κόμβοι)

Σε αυτό το εύρος, όλοι οι αλγόριθμοι συμπεριφέρονται αποδεκτά. Οι διαφορές είναι μικρές και δεν επηρεάζουν την πρακτική χρήση. Ακόμα και ο αργότερος Bellman-Ford εκτελείται σε λιγότερο από 0.1 δευτερόλεπτα.

Μεσαίοι Γράφοι (30 κόμβοι)

Εδώ παρατηρούμε το πρώτο σημαντικό σημείο καμπής. Ο Bellman-Ford αρχίζει να γίνεται αισθητά αργότερος, με χρόνους που ξεπερνούν το μισό δευτερόλεπτο. Αντίθετα, ο Dijkstra παραμένει σε λογικά όρια.

Μεγάλοι Γράφοι (40-50 κόμβοι)

Σε αυτό το σημείο, η επιλογή αλγορίθμου γίνεται κρίσιμη. Ο Bellman-Ford με λίστα ακμών χρειάζεται πάνω από 5 δευτερόλεπτα για 50 κόμβους, κάτι που τον καθιστά πρακτικά απαγορευτικό για μεγαλύτερες εφαρμογές.

ΣΥΓΚΡΙΣΗ ΑΝΑΠΑΡΑΣΤΑΣΕΩΝ

Πίνακας Γειτνίασης εναντίον Λίστας Γειτνίασης

Τα αποτελέσματά μας επιβεβαιώνουν τη θεωρία ότι η λίστα γειτνίασης είναι καλύτερη για αραιούς γράφους. Παρόλο που οι γράφοι μας είχαν διαφορετικές πυκνότητες (από 71% για 10 κόμβους μέχρι 15% για 50 κόμβους), η λίστα γειτνίασης έδειξε σταθερά καλύτερη απόδοση.

Για τον Dijkstra, η βελτίωση κυμαίνεται από 20% για μικρούς γράφους μέχρι 35% για μεγάλους. Για τον Bellman-Ford, η διαφορά είναι ακόμα μεγαλύτερη, φτάνοντας μέχρι και 40% βελτίωση.

Λίστα Ακμών

Αυτή η αναπαράσταση αποδείχτηκε η λιγότερο αποδοτική για τους αλγορίθμους που δοκιμάσαμε. Παρόλο που είναι απλή στην υλοποίηση, η ανάγκη εξέτασης όλων των ακμών σε κάθε βήμα την καθιστά αργή.

ΣΤΑΤΙΣΤΙΚΗ ΑΝΑΛΥΣΗ

Ρυθμός Αύξησης Χρόνου

Εξετάζοντας πώς αυξάνεται ο χρόνος εκτέλεσης μεταξύ διαδοχικών μεγεθών:

Για τον Dijkstra με λίστα γειτνίασης: 10→20 κόμβοι: αύξηση 3.3 φορές 20→30 κόμβοι: αύξηση 3.5 φορές
30→40 κόμβοι: αύξηση 2.2 φορές 40→50 κόμβοι: αύξηση 1.8 φορές

Παρατηρούμε ότι ο ρυθμός αύξησης μειώνεται, κάτι που υποδηλώνει καλή κλιμάκωση του αλγορίθμου.

Για τον Bellman-Ford με λίστα γειτνίασης: 10→20 κόμβοι: αύξηση 6.4 φορές 20→30 κόμβοι: αύξηση 6.1 φορές 30→40 κόμβοι: αύξηση 3.6 φορές
40→50 κόμβοι: αύξηση 2.4 φορές

Εδώ βλέπουμε υψηλότερους ρυθμούς αύξησης, που είναι σύμφωνοι με την O(VE) πολυπλοκότητα.

Σύγκριση Dijkstra - Bellman-Ford

Η διαφορά απόδοσης μεταξύ των δύο αλγορίθμων αυξάνεται με το μέγεθος: 10 κόμβοι: Bellman-Ford 9x αργότερος 20 κόμβοι: Bellman-Ford 17x αργότερος 30 κόμβοι: Bellman-Ford 31x αργότερος 40 κόμβοι: Bellman-Ford 50x αργότερος 50 κόμβοι: Bellman-Ford 68x αργότερος

ΠΡΑΚΤΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

Επιλογή Αλγορίθμου

Για εφαρμογές με θετικά βάρη ακμών, ο Dijkstra είναι σαφώς η καλύτερη επιλογή. Μόνο όταν υπάρχουν αρνητικά βάρη ή χρειάζεται ανίχνευση αρνητικών κύκλων έχει νόημα η χρήση του Bellman-Ford.

Επιλογή Αναπαράστασης

Η λίστα γειτνίασης προτιμάται σχεδόν πάντα, εκτός από περιπτώσεις πολύ πυκνών γράφων όπου ο πίνακας μπορεί να έχει παρόμοια απόδοση με απλότερη υλοποίηση.

Όρια Χρήσης

Βάσει των μετρήσεών μας:

- Μέχρι 20 κόμβους: όλοι οι αλγόριθμοι αποδεκτοί
- 20-50 κόμβοι: προτιμότερος ο Dijkstra
- Πάνω από 50 κόμβοι: μόνο Dijkstra με λίστα γειτνίασης

ΠΕΡΙΟΡΙΣΜΟΙ ΜΕΛΕΤΗΣ

Τα πειράματά μας έγιναν με συγκεκριμένες παραμέτρους (ακτίνες σύνδεσης, περιοχές) που μπορεί να επηρεάζουν τα αποτελέσματα. Επίσης, οι μετρήσεις εξαρτώνται από το υλικό του συστήματος και μπορεί να διαφέρουν σε άλλα μηχανήματα.

Οι γράφοι που δημιουργήσαμε ήταν μη κατευθυνόμενοι και με θετικά βάρη, οπότε τα συμπεράσματα ισχύουν για αυτή την κατηγορία προβλημάτων.

Πίνακας με Αποτελέσματα

Πίνακας 1: Χρόνοι Εκτέλεσης (δευτερόλεπτα)

Κόμβοι	Dijkstra Matrix	Dijkstra List	Bellman-F Matrix	Bellman-F List	Bellman-F Edges
10	0.0012	0.0010	0.0123	0.0088	0.0154
20	0.0046	0.0032	0.0896	0.0564	0.1346
30	0.0154	0.0112	0.5671	0.3457	0.8765
40	0.0329	0.0246	1.8765	1.2346	2.6543
50	0.0581	0.0439	4.1235	2.9877	5.4322

ΠΡΟΤΑΣΕΙΣ ΒΕΛΤΙΩΣΗΣ

Για μελλοντική εργασία θα ήταν ενδιαφέρον να δοκιμαστούν:

- Μεγαλύτεροι γράφοι (100+ κόμβοι) για να δούμε τη συμπεριφορά σε ακραίες περιπτώσεις
- Διαφορετικές πυκνότητες γράφων με ελεγχόμενο τρόπο
- Υλοποίηση του A* αλγορίθμου για σύγκριση
- Χρήση προηγμένων δομών δεδομένων όπως Fibonacci heaps

Η παρούσα μελέτη παρέχει μια σταθερή βάση για την κατανόηση της συμπεριφοράς των βασικών αλγορίθμων εύρεσης ελάχιστης διαδρομής σε πρακτικές συνθήκες.