Graph Algorithms

Παρουσίαση στα πλαίσια του μαθήματος «Παράλληλοι Αλγόριθμοι»

Καούρη Γεωργία
Μήτσου Βάλια

Περιεχόμενα

- Μεταβατικό Κλείσιμο
- Συνεκτικές συνιστώσες
- Συντομότερα μονοπάτια
- Breadth – First Spanning Trees
- Spanning Trees ελάχιστου βάρους

Τα παραπάνω προβλήματα αν και φαίνονται ασύνδετα, παρουσιάζουν μεγάλη ομοιότητα στον τρόπο αντιμετώπισής τους, όπως και με τον παράλληλο αλγόριθμο για Gaussian Elimination.

Μεταβατικό Κλείσιμο (1/5)

Δίνεται κατευθυνόμενος γράφος N – κόμβων G=(V, E) με πίνακα αντιστοίχισης A = (a_{ij}).

Το μεταβατικό κλείσιμο G^* του G ορίζεται σαν ο γράφος με κόμβους V και ακμές

E^* = {(i, j)| υπάρχει κατευθυνόμενο μονοπάτι από τον i στον j}

Sequential algorithm: $O(N^3)$

Μεταβατικό Κλείσιμο (2/5)

Η παραλληλοποίηση γίνεται σε N φάσεις:

1η φάση: Εισάγουμε την ακμή (i, j) στο γράφο ανν υπάρχουν οι ακμές (i, 1) και (1, j) στο γράφο $G = G^{(0)}$.

k – οστή φάση: Εισάγουμε την ακμή (i, j) ανν υπάρχουν οι ακμές (i, k) και (k, j) στο γράφο που σχηματίστηκε μετά την k – 1 φάση, δηλαδή στο γράφο $G^{(k-1)}$.

Στο τέλος της N φάσης προκύπτει ο ζητούμενος γράφος $G^{(N)} = G^*$.

Μεταβατικό Κλείσιμο (3/5)

- Οι γραμμές του πίνακα αντιστοίχισης εισάγονται στο mesh αντίστροφα.
- Κάθε γραμμή σταματά στην πρώτη ελεύθερη γραμμή του mesh στο 2i – 1 βήμα (Η i γραμμή σταματά στην i γραμμή του mesh).
- Η i γραμμή αρχίζει να κινείται προς τα κάτω όταν η Ν γραμμή του πίνακα την «προσπεράσει», δηλαδή στο Ν +2i -1 βήμα.
- Οι γραμμές του πίνακα G^* βγαίνουν από την τελευταία γραμμή του mesh.

Μεταβατικό Κλείσιμο (4/5)

1η φάση: Η γραμμή 1 του πίνακα Α έχει αποθηκευτεί στην πρώτη γραμμή του mesh. Καθώς μπαίνει η i γραμμή του Α στην 1η γραμμή του mesh, το κελί (1, 1) μεταδίδει την τιμή $a_{i,1}$ στα υπόλοιπα κελιά του mesh. Αν $a_{i,1} = 1$ και $a_{1,j} = 1$ το κελί αλλάζει την τιμή του a_{ij} σε 1, αλλιώς δεν την αλλάζει. Δηλαδή η i γραμμή του Α αφού έχει περάσει από την 1η γραμμή του mesh είναι πλέον η γραμμή i του πίνακα $A^{(1)}$.

k – φάση: Οι γραμμές που φτάνουν στην k γραμμή του mesh είναι οι γραμμές του $A^{(k-1)}$. Καθώς φτάνουν το κελί (k, k) μεταδίδει την τιμή $a_{i,k}^{(k-1)}$ στα άλλα κελιά της γραμμής. Το κελί (k, j) έχει αποθηκευμένη στη μνήμη του την τιμή $a_{kj}^{(k-1)}$, και κάνει τον υπολογισμό

$$a_{ij}^{(k)}=a_{ij}^{(k-1)} \vee (a_{ik}^{(k-1)} \wedge a_{kj}^{(k-1)}).$$

Μεταβατικό Κλείσιμο (5/5)

- Χρειαζόμαστε συνολικά 3N – 1 βήματα μέχρι να πάρουμε τον πίνακα A^*. Ωστόσο ο αλγόριθμος είναι semisystolic.
- Ακολουθώντας την ανάλυση της παραγράφου 1.4.6 κάνουμε retiming και ο αλγόριθμος ολοκληρώνεται μετά από 5N – 2 βήματα.
- Τελικά με pipelining χρειαζόμαστε N βήματα και N^2 processors, οπότε επιτυγχάνουμε σημαντικό speedup σε σχέση με τον sequential αλγόριθμο.

Παρατηρήσεις

Η πράξη

$$a_{ij}^{(k)}=a_{ij}^{(k-1)} \vee (a_{ik}^{(k-1)} \wedge a_{kj}^{(k-1)})$$

μοιάζει με την πράξη

$$a_{ij}^{(k)}=a_{ij}^{(k-1)} - (a_{ik}^{(k-1)} \cdot a_{kj}^{(k-1)})$$

με αντικατάσταση των – και · με ν και ^.

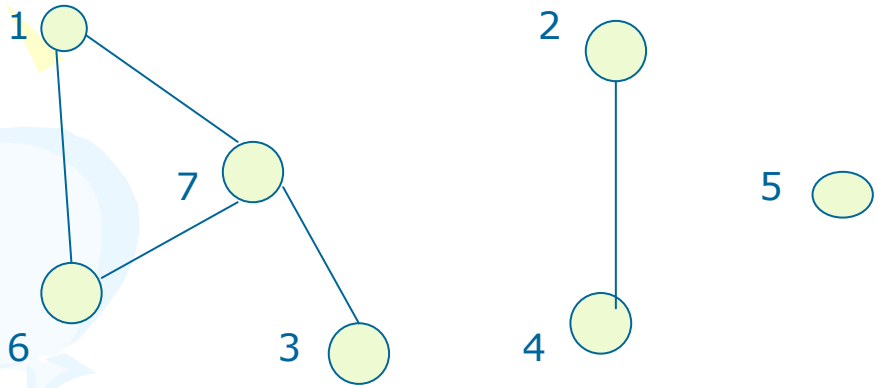
- Βλέπουμε επομένως την αντιστοιχία με την Gaussian Elimination.

Συνεκτικές συνιστώσες (1/3)

Ορισμός: Δύο κόμβοι i και j ενός μη κατευθυνόμενου γράφου ανήκουν στην ίδια συνεκτική συνιστώσα αν υπάρχει μονοπάτι από τον ένα κόμβο στον άλλο.

* Κατασκευάζοντας το μεταβατικό κλείσιμο ενός γράφου, μπορούμε να ελέγξουμε αν δύο κόμβοι i και j ανήκουν στην ίδια συνεκτική συνιστώσα ελέγχοντας την τιμή a^*_{ij}.

Συνεκτικές συνιστώσες (2/3)



G

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}$$

connected components: {1, 3, 6, 7}, {2, 4}, {5}

Συνεκτικές συνιστώσες (3/3)

- Ο παράλληλος αλγόριθμος τρέχει σε N βήματα με N^2 processors.
- Υπάρχει sequential αλγόριθμος που βρίσκει συνεκτικές συνιστώσες σε O(E) βήματα.
- Επομένως ο παραπάνω αλγόριθμος δεν είναι καλός, ακόμα κι όταν $E = \Theta(N^2)$...
- Σε επόμενο κεφάλαιο, χρησιμοποιώντας διαφορετικές τοπολογίες μπορούμε να επιτύχουμε καλύτερα αποτελέσματα.

Συντομότερα μονοπάτια (1/2)

Δίνεται κατευθυνόμενος γράφος G(V, E) με βάρη w_{ij} στις ακμές και αναζητάμε το συντομότερο μονοπάτι από τον κόμβο i στον j.

Προϋποθέσεις:

- Υπάρχουν όλες οι ακμές, εκτός από τα self loops. Για τις ακμές που δε δίνονται στον αρχικό γράφο βάζουμε $w^{ij} = \infty$.
- Ο G δεν περιέχει κύκλους αρνητικού βάρους.

Συντομότερα μονοπάτια (2/2)

Ο αλγόριθμος μοιάζει με αυτούς που έχουμε περιγράψει μέχρι τώρα. Αποτελείται από N βήματα και είναι ο εξής:

Θέτουμε $w_{ij}=w_{ij}^{(0)}$.

1η φάση: Αντικαθιστούμε την ακμή (i, j) με μονοπάτι μικρότερου βάρους από τον i στον j το οποίο διέρχεται από τον κόμβο 1 αν αυτό υπάρχει. Δηλαδή συγκρίνουμε τα w_{ij} με το $w_{i1}+w_{1j}$ και κρατάμε τη μικρότερη τιμή, την οποία θέτουμε $w_{ij}^{(1)}$.

k φάση: Υπολογίζουμε την τιμή

$$w_{ij}^{(k)}=\min\{w_{ij}^{(k-1)}, w_{ik}^{(k-1)}+w_{kj}^{(k-1)}\}$$

για να καθορίσουμε το συντομότερο μονοπάτι από τον κόμβο i στον j χρησιμοποιώντας τους κόμβους {1, 2, ..., k}.

* Μετά από N βήματα έχουμε υπολογίσει όλα τα συντομότερα μονοπάτια.

Παρατηρήσεις

- Ο αλγόριθμος που χρησιμοποιούμε είναι παρόμοιος με όσους έχουμε περιγράψει αν αντικαταστήσουμε τα min και + με ν και ^. Επομένως μπορούμε να ακολουθήσουμε την ίδια διαδικασία.
- Για να μπορέσουμε να βρούμε το μονοπάτι που χρησιμοποιήσαμε για να βρούμε το shortest path, αρκεί σε κάθε κελί να κρατάμε τον κόμβο από τον οποίο περνάει το μονοπάτι καθώς αλλάζει.
- Και εδώ χρειαζόμαστε N βήματα και N^2 processors, ενώ ο naive sequential αλγόριθμος απαιτεί $\Theta(N^3)$ βήματα.

Breadth – First Spanning Trees (1/2)

Δοθέντος ενός συνεκτικού μη κατευθυνόμενου γράφου χωρίς βάρη, με ρίζα, ένα breadth – first spanning tree είναι ένα spanning tree στο οποίο το μονοπάτι από έναν κόμβο στη ρίζα είναι το συντομότερο στον αρχικό γράφο.

Αυτό το spanning tree μπορεί να βρεθεί μέσω του προβλήματος shortest path θέτωντας βάρη στις ακμές του γράφου (1 για όσες ακμές υπάρχουν και ∞ για όσες δεν υπάρχουν). Λύνοντας το shortest path στο γράφο που προκύπτει βρίσκουμε την απόσταση κάθε κόμβου από τη ρίζα. Για να βρούμε το ζητούμενο spanning tree για κάθε κόμβο που απέχει i από τη ρίζα μαρκάρουμε την ακμή που συνδέει αυτόν τον κόμβο με κόμβο που απέχει από τη ρίζα i – 1 για κάθε i > 0.

Επειδή μπορούμε να βρούμε περισσότερες από 1 τέτοιες ακμές, το breadth – first spannong tree που προκύπτει δεν είναι μοναδικό.

Breadth – First Spanning Trees (2/2)

- Ο αλγόριθμος είναι ο ίδιος με αυτόν για τον υπολογισμό των shortest paths. Θεωρώντας όμως οτι ρίζα είναι ο κόμβος 1, αρκεί να κάνουμε update μόνο τις τιμές της πρώτης γραμμής του πίνακα A, οπότε ο αλγόριθμος χρειάζεται μόνο 2N βήματα.
- Σε N-1 ακόμα βήματα μπορούμε να βρούμε την ακμή που συνδέει κάθε κόμβο που απέχει απόσταση i από τη ρίζα με κόμβο που απέχει απόσταση i-1. Κάνοντας output αυτόν τον κόμβο έχουμε το ζητούμενο spanning tree.
- Επομένως χρειαζόμαστε συνολικά 3N-1 βήματα (με retiming 7N-2 βήματα για να γίνει systolic) και N^2 processors. Οι sequential αλγόριθμοι είναι επίσης O(N), ενώ δεν είναι γνωστοί άλλοι γρηγορότεροι παράλληλοι αλγόριθμοι.

Spanning Trees ελάχιστου βάρους (1/3)

- Ένα minimum weight spanning tree είναι ένα spanning tree με το ελάχιστο δυνατό άθροισμα ακμών.
- Γνωστοί sequential αλγόριθμοι επιτυγχάνουν πολυπλοκότητα $O(E \log |E|)$ και $O(|V|^2)$.
- Ο αλγόριθμος που θα περιγράψουμε χρειάζεται $O(E)$ βήματα, άρα δεν είναι ιδιαίτερα αποδοτικός, είναι βέλτιστος όμως χρησιμοποιώντας arrays.
- Θα υποθέσουμε οτι δεν υπάρχουν ακμές με ίδιο βάρος.

Spanning Trees ελάχιστου βάρους (2/3)

Λήμμα: Αν δεν υπάρχουν ακμές ίσου βάρους στο γράφο, τότε μια ακμή (i, j) με βάρος w_{ij} ανήκει στο mst ανν οποιοδήποτε άλλο μονοπάτι μεταξύ των i, j μήκους ≥2 περιέχει ακμή μεγαλύτερου βάρους από την w_{ij}.

Απόδειξη: Έστω T mst.

Έστω (i,j)ϵT και υπάρχει μονοπάτι P_{ij} με βάρη ακμών $\leq w_{ij}$.

Έστω οτι κάθε μονοπάτι μήκους ≥2 από τον i στον j περιέχει μια ακμή βάρους $\geq w_{ij}$, (i, j)¬ϵT.

$\Rightarrow$ $\Leftarrow$

Spanning Trees ελάχιστου βάρους (3/3)

Ο αλγόριθμος είναι ίδιος με τον υπολογισμό των shortest paths με 2 διαφορές:

1. Το βάρος ενός μονοπατιού είναι το βάρος του βαρύτερου μονοπατιού του.
2. Η εξίσωση που υπολογίζει το κάθε κελί είναι η:

$$w_{ij}^{(k)}=\min\{w_{ij}^{(k-1)}, \max(w_{ik}^{(k-1)}, w_{kj}^{(k-1)})\}$$

Υπολογίζουμε δηλαδή την ελάχιστη μέγιστη ακμή που ενώνει κάθε δύο κόμβους i, j. Επομένως από το προηγούμενο λήμμα μια ακμή $(i,j)\in msp$ αvv $w_{ij}=w_{ij}^{(N)}$.