Παράλληλα και Διανεμημένα Συστήματα

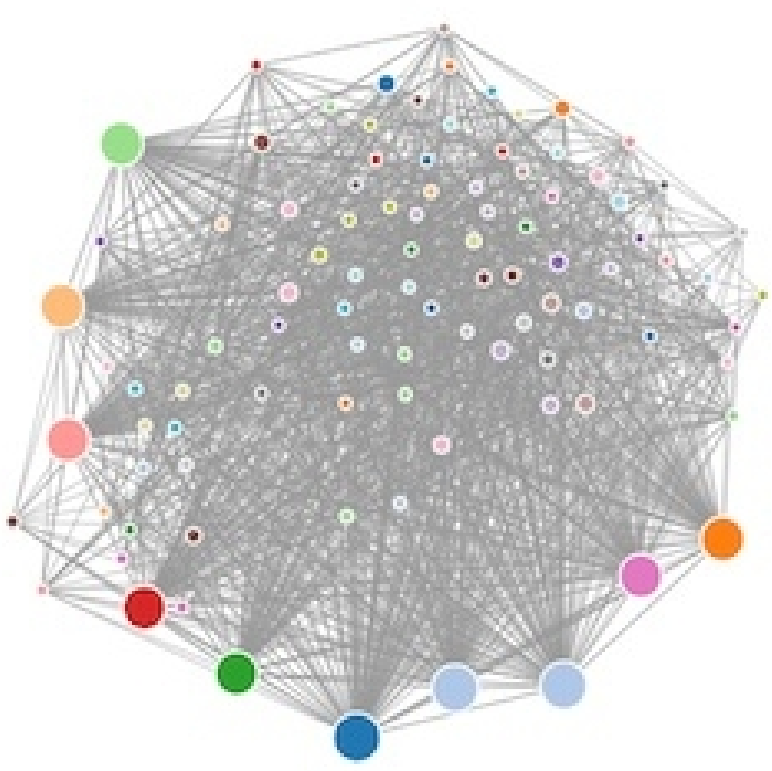
Εργασία 4η

Βλαδίμηρος Στερζεντσένκο

ΑΕΜ:7777

email:vlad.sterzentsenko@gmail.com

Θεσσαλονίκη 27/09/2015



Σκοπός της εργασίας

Δοσμένου ενός γράφου , καλούμαστε να υπολογίσουμε την πιθανότητα να βρεθεί ο χρήστης σε κάποιον κόμβο , σε κάποια μελλοντική στιγμή. Οι πιθανότητες υπολογίζονται με βάση τον παρακάτω τύπο.

$$P_i^{t+1} = d \sum_{j \in B_i} P_j^t \frac{a_{ij}}{\sum_{k \in B_j} a_{jk}} + (1-d) E_i$$

Έπειτα από το πέρας ενός αριθμού βημάτων, οι πιθανότητες σταθεροποιούνται. Οι πιθανότητες για κάθε κόμβο στην σταθερή αυτή κατάσταση ορίζουν το κριτήριο σημαντικότητας του κάθε κόμβου.

Υλοποιήση

Σαν είσοδος στο πρόγραμμα πρέπει να δοθεί ένα αρχείο αυστηρά στην μορφή **node id connected node id**. Αφού δοθεί σαν όρισμα στο πρόγραμμα το αρχείο, παράγονται 3 πίνακες.

counter_in[i] : Περιέχει τον αριθμό των κόμβων που συνδέονται στον κόμβο i.

counter_out[i] : Περιέχει τον αριθμό τον κόμβων που συνδέεται ο κόμβος i.

Conn[i][] : Περιέχει τα Ids των κόμβων που συνδέονται στον κόμβο i.

Η υλοποιήση του αλγορίθμου εύρεσης πιθανότητας ακολουθεί ακριβώς το άθροισμα που δόθηκε παραπάνω. Για τους κόμβους οι οποίοι δεν έχουν ακμές που να πηγαίνουν σε άλλους κόμβους , θεωρούμε ότι συνδέονται σε όλους τους άλλους κόμβους , και στον εαυτό τους. Είναι μια μέθοδος κανονικοποιήσης του αποτελέσματος, ώστε το άθροισμα των πιθανοτήτων σε κάθε instance του προγράμματος να ισούται με 1.

Στην παράλληλη υλοποιήση του προγράμματος, κάθε νήμα αναλαμβάνει να υπολογίσει τις πιθανότητες, για κάθε βήμα του προγράμματος, για ένα κομμάτι του πίνακα.

Επαλήθευση του αποτελέσματος

Η επαλήθευση του τελικού αποτελέσματος έγινε με χρήση του δοσμένου κώδικα, σε Matlab. Τα εξής βήματα έγιναν:

- Παραγωγή ενός dataset 200 κόμβων από τον δοσμένο κώδικα (surfer.m).
- Αποθήκευση του dataset , στην μορφή εισόδου του προγράμματος , σε ένα αρχείο .txt.
- Εκτέλεση του κώδικα σε Matlab (pagerankpow.m) και εκτέλεση του κώδικα σε C.
- Αποθήκευση των τελικών πιθανοτήτων , με ακρίβεια 10 δεκαδικών ψηφίων , σε αρχεία .txt.
- Σύγκριση των αρχείων με το πρόγραμμα diff.

Παρακάτω παρουσιάζονται τα αποτελέσματα της σειριακής και της παράλληλης υλοποιήσης του αλγορίθμου. Χρησιμοποιήθηκαν έτοιμα datasets [1] , καθώς η εκτέλεση του surfer.m για μεγάλο αριθμό κόμβων είναι χρονοβόρα.

Datasets

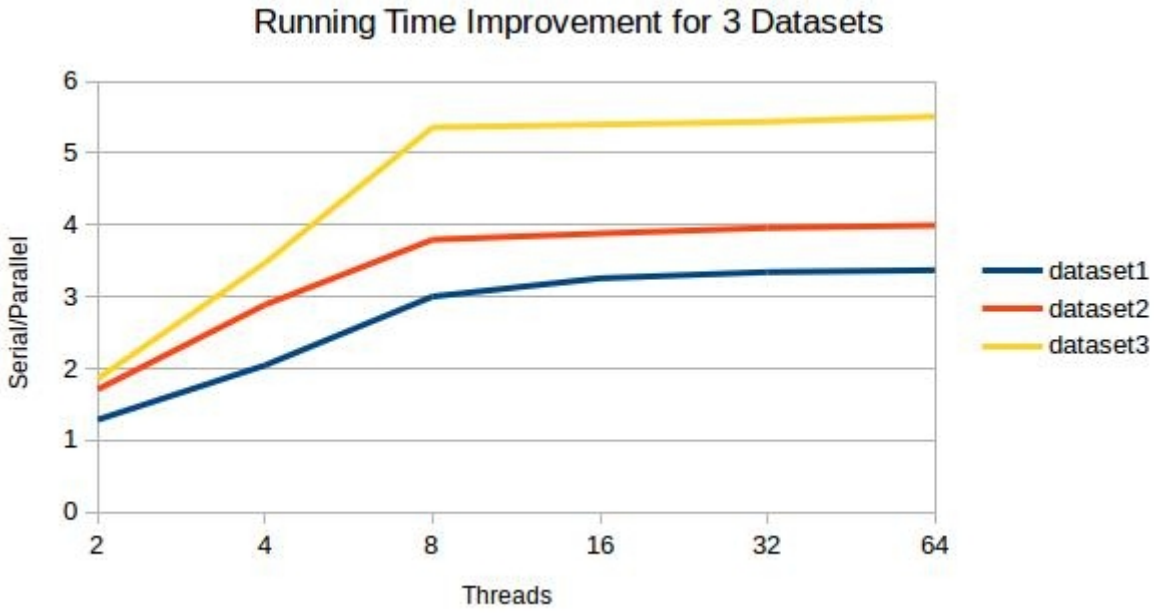
Dataset 1 : Soc-Pokec	Nodes:1,632,803	Edges:30,622,564
Dataset 2 : Web-Google	Nodes:875,713	Edges:5,105,039
Dataset 3 : Web-BerkStan	Nodes:685,230	Edges:7,600,595

Παρουσίαση Αποτελεσμάτων

Πίνακας Χρόνων εκτέλεσης προγράμματος για 3 διαφορετικά datasets (σε seconds).

	Serial	*2 thr*	*4 thr*	*8 thr*	*16 thr*	*32 thr*	*64 thr*
Set1	2.34	1.82	1.15	0.77	0.71	0.69	0.68
Set2	2.29	1.34	0.79	0.61	0.59	0.58	0.57
Set3	0.99	0.53	0.28	0.18	0.18	0.17	0.17

Διάγραμμα βελτίωσης χρόνων ($\frac{\textit{Χρόνος Σειριακής Εκτέλεσης}}{\textit{Χρόνος Παράλληλης Εκτέλεσης}}$).



Βήματα για την σύγκλιση του αλγορίθμου

Dataset 1 : 2

Dataset 2 : 7

Dataset 3 : 11

Σχόλια

- Φανερό είναι ότι , όσα περισσότερα βήματα κάνει ο αλγόριθμος να συγκλίνει, τόσο βελτιώνεται η επίδοση της παράλληλης υλοποιήσης έναντι της σειριακής.
- Το πρόγραμμα εκτελέστηκε στον diades, με αποτέλεσμα ,τα αποτελέσματα για περισσότερα απο 8 νήματα, να μην παρουσιάζουν ιδιαίτερη βελτίωση. Θεωρούμε ότι η καθυστερήσεις επέρχονται απο την επικοινωνία μεταξύ επεξεργαστών.
- Δεν ήταν δυνατή η αποφυγή ορισμένων barriers στην παράλληλη υλοποιήση, καθώς το πρόγραμμα πρέπει να είναι απόλυτα συγχρονισμένο σε κάθε instance του.

[1] http://snap.stanford.edu/data/