Ανάπτυξη Λογισμικού Πληροφορικής
Χειμερινό Εξάμηνο 2014 - 2015

“Σύστημα Ανάλυσης Κοινωνικών δικτύων” part 2

Ημερομηνία παράδοσης: 05/12/2014

ΠΕΡΙΕΧΟΜΕΝΑ

Γενική περιγραφή

Αυτό το επίπεδο της εργασίας αποτελείται από δύο τμήματα. Στο πρώτο τμήμα θα υλοποιήσετε μερικούς από τους αλγορίθμους που χρησιμοποιούνται συχνά για να κάνουν διαφόρων ειδών μετρήσεις σε γράφους. Στο δεύτερο τμήμα θα φτιάξετε τις πρώτες σας συναρτήσεις που θα υλοποιούν την λειτουργικότητα ερωτημάτων που θέλουμε να απαντήσουμε σχετικά με το γράφημα.

Και στα δύο τμήματα θα χρησιμοποιήσετε δομές και συναρτήσεις που έχετε φτιάξει στο προηγούμενο επίπεδο. Επιπροσθέτως το δεύτερο τμήμα θα χρειαστεί και συναρτήσεις από το πρώτο τμήμα αυτού του επιπέδου. Τέλος, για κάποιες λειτουργίες θα είναι απαραίτητο να φτιάξετε και δικές σας επιπλέον δομές.

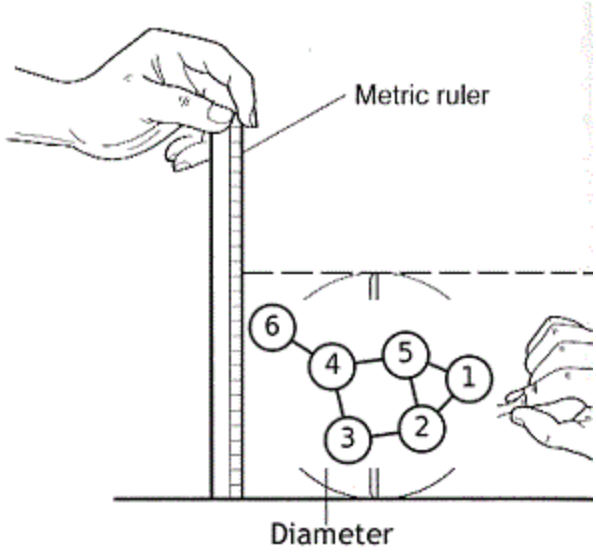
Όπως έχει ήδη ειπωθεί, αν κατά τη διάρκεια υλοποίησης αυτού του επιπέδου διαπιστώσετε ανεπάρκειες στον κώδικα που ήδη έχετε παραδώσει, μπορείτε να τον αλλάξετε για να λειτουργήσει σωστά σε αυτό το επίπεδο.

Περιγραφή παραδοτέων δεύτερου επιπέδου

Στις ενότητες που ακολουθούν δίνεται αναλυτικά η περιγραφή των τμημάτων του δεύτερου επιπέδου.

1. Μετρήσεις και στατιστικά σε γράφους.

Για να διαλέξουμε τί είδους αλγόριθμο θα χρησιμοποιήσουμε για να απαντήσει ένα ερώτημα σωστά και αποδοτικά, χρειάζεται να ξέρουμε κάποιες πληροφορίες σχετικά με το γράφο που έχουμε. Πληροφορίες δεν είναι μόνο το πλήθος των κόμβων και των ακμών του, και συνήθως δεν αρκεί να ξέρουμε μόνο αυτά, αλλά και άλλες ιδιότητες που μπορούμε να μετρήσουμε.



Σχήμα 1. Χειρωνακτική προσέγγιση μέτρησης του γράφου

Στα πλαίσια της συγκεριμένης ενότητας θα υλοποίησετε αλγορίθμους που βρίσκουν τις μετρικές που αναφέρουμε στη συνέχεια.

- **<u>Κατανομή βαθμού (degree distribution).</u>**
  Βαθμός (degree) ενός κόμβου σε ένα δίκτυο είναι ο αριθμός των ακμών που έχει αυτός ο κόμβος προς άλλους κόμβους. Έστω n_k το πλήθος των κόμβων που έχουν βαθμό k και n το συνολικό πλήθος των κόμβων του γράφου. Τότε η κατανομή βαθμού ορίζεται ως $P(k) = n_k / n$.
  Για την υλοποίηση αυτού του ερωτήματος θα πρέπει να χρησιμοποιήσετε gnuplot για να κάνετε οπτικοποίηση της κατανομής.
  - *void degreeDistribution(Graph g*);*

- **<u>Διάμετρος (diameter).</u>**
  Απόσταση (distance) μεταξύ δύο κόμβων είναι το μήκος του κοντινότερου μονοπατιού μεταξύ τους. Η διάμετρος ενός γράφου είναι η μεγαλύτερη απόσταση μεταξύ δύο οποιονδήποτε κόμβων.
  - *int diameter(Graph g*);*

- **<u>Μέσο μήκος μονοπατιού (average path length).</u>**
  Το μέσο μήκος μονοπατιού ορίζεται ως η μέση τιμή των αποστάσεων μεταξύ οποιονδήποτε δύο κόμβων του γράφου.
  - *double averagePathLength(Graph* g);*

- **<u>Πλήθος συνεκτικών γραφημάτων (connected components).</u>**
  Θεωρούμε ένα μη κατευθυνόμενο γράφημα. Συνεκτικό γράφημα ονομάζεται ένα υποσύνολο κόμβων του αρχικού γράφου, για το οποίο ισχύει ότι για κάθε δύο κόμβους του υποσυνόλου υπάρχει μονοπάτι που τους συνδέει. Σε ένα γράφο μπορεί να υπάρχουν περισσότερα από ένα συνεκτικά γραφήματα.
  - *int numberOfCCs(Graph* g);*

- **<u>Πλήθος κόμβων μέγιστου συνεκτικού γραφήματος.</u>**
  Από όλα τα συνεκτικά γραφήματα του γράφου, υπολογίστε και επιστρέψτε το πλήθος των κόμβων του μεγαλύτερου συνεκτικού γραφήματος.
  - *int maxCC(Graph* g);*

- **<u>Πυκνότητα (density).</u>**
  Ένας γράφος ονομάζεται πυκνός (dense) όταν ο αριθμός των ακμών του πλησιάζει το μέγιστο αριθμό ακμών που θα μπορούσε να έχει. Έστω |V| το πλήθος των κόμβων και |E| το πλήθος των ακμών. Η πυκνότητα σε ένα μη κατευθυνόμενο γράφημα ορίζεται ως εξής:
  $$D = \frac{2|E|}{|V|\,(|V|-1)}$$
  - *double density(Graph* g);*

- **<u>Κεντρικότητα (centrality)</u>**
  Κεντρικότητα (centrality) είναι μια μετρική που οι τιμές της καθορίζουν τους πιο σημαντικούς κόμβους σε ένα γράφο. Επειδή η σημαντικότητα είναι σχετική και ορίζεται με διάφορους τρόπους, έχουμε και διαφορετικούς τύπους centrality. Σε αυτό το επίπεδο θα ασχοληθούμε με τους εξής δύο: a) closeness και betweenness.

a) *closeness centrality*

Διαισθητικά η μετρική αυτή μετράει τη σημαντικότητα ενός κόμβου ως εξής: όσο μικρότερη είναι η συνολική του απόσταση από τους υπόλοιπους κόμβους, τόσο πιο σημαντικός είναι ο κόμβος αυτός. Ορίζεται ως:

$$C_c(i) = \sum_{j=1}^{N} [d(i,j)]^{-1}$$

όπου i ο κόμβος του οποίου την κεντρικότητα μετράμε.

Η μετρική αυτή γίνεται normalize με τον παρακάτω τύπο: $C_C^{'}(i) = (C_C(i))/(N-1)$, όπου N είναι ο αριθμός των κόμβων.

- *double closenessCentrality(Node* n, Graph* g);*

b) *betweenness centrality*

Η μετρική αυτή ποσοτικοποιεί τη σημαντικότητα ενός κόμβου ως εξής: όσο περισσότερες φορές ο κόμβος λειτουργεί ως γέφυρα (bridge) ενδιάμεσα στα κοντινότερα μονοπάτια οποιονδήποτε δύο άλλων κόμβων, τόσο πιο σημαντικός είναι. Ορίζεται με τον παρακάτω τύπο:

$$C_B(i) = \sum_{s \neq i \neq t \, \epsilon V} \frac{\sigma_{st}(i)}{\sigma_{st}}$$

όπου σ_{st} είναι ο συνολικός αριθμός των κοντινότερων μονοπατιών από τον κόμβο *s* στον κόμβο *t*, και $\sigma_{st}(i)$ ο αριθμός των κοντινότερων μονοπατιών που περνούν από τον κόμβο *i*. Η μετρική γίνεται normalize αν διαιρέσουμε με τον αριθμό των ζευγαριών των κόμβων που δεν περιλαμβάνουν τον *i* :

$$C'_B(i) = C_B(i)/[(\mathrm{N}-2)(\mathrm{N}-1)/2]$$

όπου το N το πλήθος των κόμβων στο γράφο.

- *double betweennessCentrality(Node* n, Graph* g);*

2. Ερωτήματα σε δεδομένα γραφου.

Σε αυτό το τμήμα θα υλοποιήσετε αλγόριθμους που θα υλοποιούν τα παρακάτω ερωτήματα.

- **Query 1**

“Matching definition: a person or thing that equals or resembles another in some respect; a complement.”

Τα περισσότερα κοινωνικά δίκτυα που υπάρχουν σήμερα έχουν ένα μηχανισμό για να προτείνουν στους χρήστες τους να κάνουν befriend άλλους χρήστες που δε γνωρίζουν. Φανταστείτε ότι έχετε μια εφαρμογή που προτείνει σε χρήστες άτομα του αντίθετου φύλου που φαίνεται ότι έχουν όμοια ενδιαφέροντα.

Δύο άτομα για να θεωρήσουμε ότι μοιάζουν πρέπει να

★ μένουν στο ίδιο μέρος ή να δουλεύουν/σπουδάζουν στον ίδιο οργανισμό,
★ έχουν πάνω από *k* κοινά ενδιαφέροντα (**τα ενδιαφέροντα ορίζονται από τη σχέση hasInterest [Person → Tag] στη βάση),
★ να έχουν διαφορά ηλικίας το πολύ *x* χρόνια και
★ να είναι του αντίθετου φύλου.

Επιπλέον πρέπει να απέχουν μέχρι h βήματα γνωριμίας. Ως μέτρο ομοιότητας να χρησιμοποιήσετε την απόσταση Jaccard. Η συνάρτηση θα επιστρέφει *limit* αριθμό από τα υψηλότερα Matches, το οποίο είναι η δομή που περιέχει το id του χρήστη και το similarity score.

- *Matches* matchSuggestion(Node* n, int k, int h, int x, int limit, Graph* g);*

- **Query 2**

"Stalker definition: a person who harasses or persecutes someone with unwanted and obsessive attention."

Όπως και σε όλες τις κοινωνίες, έτσι και σε ένα κοινωνικό δίκτυο συχνά παρατηρούνται "ανάρμοστες" ή "ανήθικες" συμπεριφορές. Ένα δίκτυο από κακόβουλα άτομα είναι καλό να ανακαλύπτεται, και επιπλέον να μελετάται η συμπεριφόρα μεταξύ των μελών του, για να γίνεται πρόβλεψη κακόβουλων δραστηριότητων. Θα παραποιήσουμε λίγο τον παραπάνω ορισμό και θα ορίσουμε ως stalker ένα άτομο

★ που έχει κάνει παραπάνω από *x* αριθμό likes στα posts ενός άλλου ατόμου και
★ με το οποίο δε γνωρίζεται προσωπικά.

Υπολογίστε ποιοί είναι οι top *k* central stalkers και σε ποιά forum εμφανίζονται. Η συνάρτηση επιστρέφει τον γράφο των stalkers και θα γεμίζει τη δομή Stalkers, η οποία περιέχει το id του stalker και το centrality score του. Με το *centralityMode* θα διαλέξετε το είδος centrality (1 = για closeness και 2 = για betweenness).

Αφού καλέσετε τη συνάρτηση, υπολογίστε και συγκρίνετε με βάση τα στατιστικά που υλοποιήσατε στο πρώτο τμήμα, τον γράφο των stalkers και τον αρχικό γράφο g.

- *Graph* getTopStalkers(int k, int x, int centralityMode, Graph* g, Stalkers* st);*

- **Query 3**

"Trend: (of a topic) is the subject of many posts on a social media website."

Τα κοινωνικά δίκτυα αντικατοπρίζουν σε ένα βαθμό τις τάσεις και τα ενδιαφέροντα της κοινωνίας. Από το δίκτυο των ατόμων που γνωρίζονται μεταξύ τους, βρείτε τα *k* μεγαλύτερα ενδιαφέροντα στο δίκτυο. Το μέγεθος του ενδιαφέροντος αντιστοιχεί στο μέγεθος του μεγαλύτερου συνεκτικού γραφήματος με το συγκεκριμένο ενδιαφέρον. Βρείτε τις τάσεις για άντρες και για γυναίκες αντίστοιχα.

- *void findTrends(int k, Graph* g, char** womenTrends, char** menTrends);*

- **Query 4**

"Trust: believe in the reliability, truth, ability, or strength of somebody or something."

Η εμπιστοσύνη είναι ένα ενδιαφέρον φαινόμενο το οποίο παρατηρείται έμμεσα στις διάφορες συναναστροφές των ατόμων. Ανάλογα με τον τύπο των συναναστροφών μας αυξάνεται ή μειώνεται. Έχει παρατηρηθεί επίσης ότι έχει την εξής ιδιότητα: είναι μεταβατική. Μπορεί δηλαδή άμεσα να μη γνωρίζεις ένα άτομο, αλλά αν κάποιοι γνωστοί σου το εμπιστεύονται να τον εμπιστευτείς και εσύ.

Από τα άτομα που ανήκουν σε ένα φόρουμ, δημιουργήστε ένα δίκτυο εμπιστοσύνης που θα έχει κατευθυνόμενες ακμές trust μεταξύ δύο χρηστών *i* και *j*, και βάρος το ποσοστό της εμπιστοσύνης του *i* στον *j*. Υποθέστε ότι η εμπιστοσύνη δημιουργείται ως εξής: trust(i,j) = 30% #likes(i,j) + 70% #replies(i,j).

Φανταστείτε ότι ένας χρήστης Α κάνει ένα post στο φόρουμ και του απαντάει ο χρήστης Β, τον οποίο δε γνωρίζει προσωπικά. Μπορεί ο Α να εμπιστευτεί την απάντησή του; Υλοποιήστε μια παραλλαγή του αλγορίθμου TidalTrust που κάνει εκτίμηση της εμπιστοσύνης του Α προς τον Β.

- *Graph* buildTrustGraph(char* forum, Graph* g);*
- *double estimateTrust(Node* a, Node* b, Graph* trustGraph);*

***** Επεξηγήσεις***

---

1. Στα πρότυπα των συναρτήσεων όπου αναφέρεται το πρότυπο Graph* g, θα δίνεται ως όρισμα ο γράφος που προκύπτει από την οντότητα Person και τη σχέση knows[Person<->Person].

---

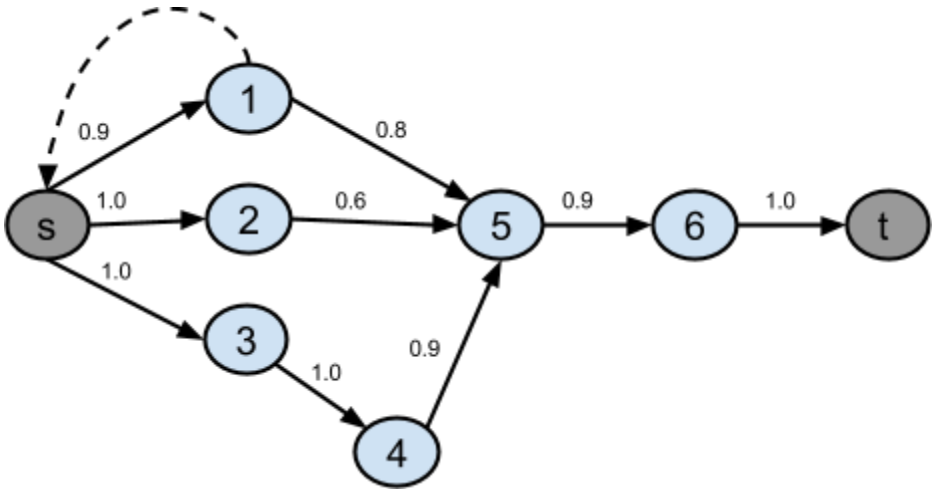
2. Παραδειγματικά αρχεία με τα δεδομένα της άσκησης μπορείτε να κατεβάσετε από αυτόν τον σύνδεσμο.

---

3. Παραλλαγή TidalTrust

Ο αλγόριθμος δέχεται ως είσοδο ένα γράφο και δύο κόμβους: τον αρχικό *s* και το τελικό *t*. Υπολογίζει την εμπιστοσύνη trust(s,t) από τον s → t, λαμβάνοντας υπόψιν του τα μεταξύ τους ακυκλικά μονοπάτια με το μικρότερο μήκος.

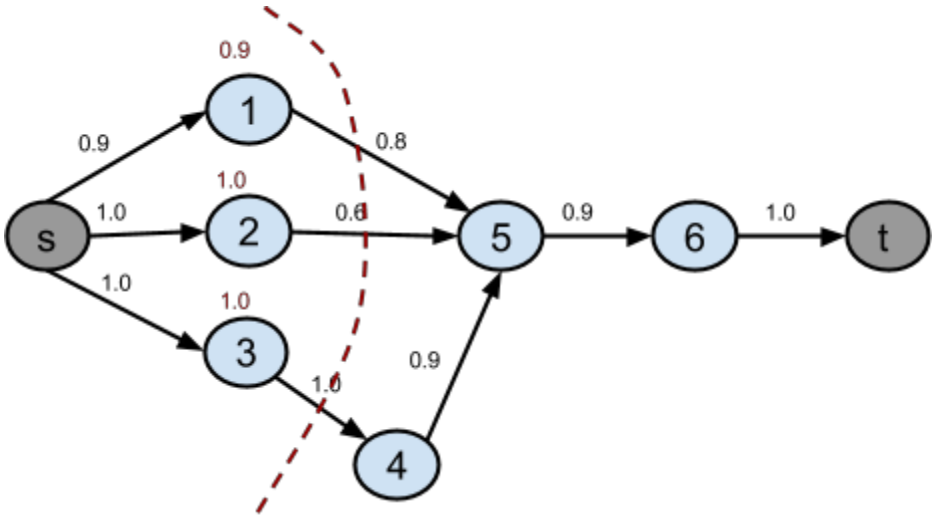
Ένα παράδειγμα εκτέλεσης του αλγορίθμου φαίνεται στο παρακάτω σχήμα. Παρατηρήστε ότι στο σχήμα παραλείπονται οι ανάποδες ακμές εμπιστοσύνης (όπως η ακμή 1→ s) για λόγους απλότητας. Ο αλγόριθμος δεν θα τις χρησιμοποιήσει στον υπολογισμό της εμπιστοσύνης καθώς ανήκουν σε μονοπάτια που δημιουργούν κύκλους.



Σχήμα 1. Αρχικός γράφος εμπιστοσύνης με τα αντίστοιχα βάρη στις ακμές.

Ο αλγόριθμος ξεκινάει την αναζήτηση του τελικού κόμβου *t* από τον κόμβο *s*. Κάθε ενδιάμεσος κόμβος αποθηκεύει την εμπιστοσύνη που του έχει ο s μέχρι εκείνη τη στιγμή.

Στο πρώτο βήμα, ο αλγόριθμος κοιτάει τους άμεσους γείτονες του {1,2,3} και ενημερώνει τις τιμές τους.



Σχήμα 2. Γράφος εμπιστοσύνης μετά το πρώτο βήμα.

Στο δεύτερο βήμα του αλγορίθμου εξετάζονται οι κόμβοι {4, 5}.

Η εμπιστοσύνη $trust_{i\to j}$ μεταξύ δύο κόμβων i, j με ενδιάμεσο κόμβο k, υπολογίζεται ως το γινόμενο των επιμέρους βαρών των ακμών i → k και k →j, από τον παρακάτω τύπο.

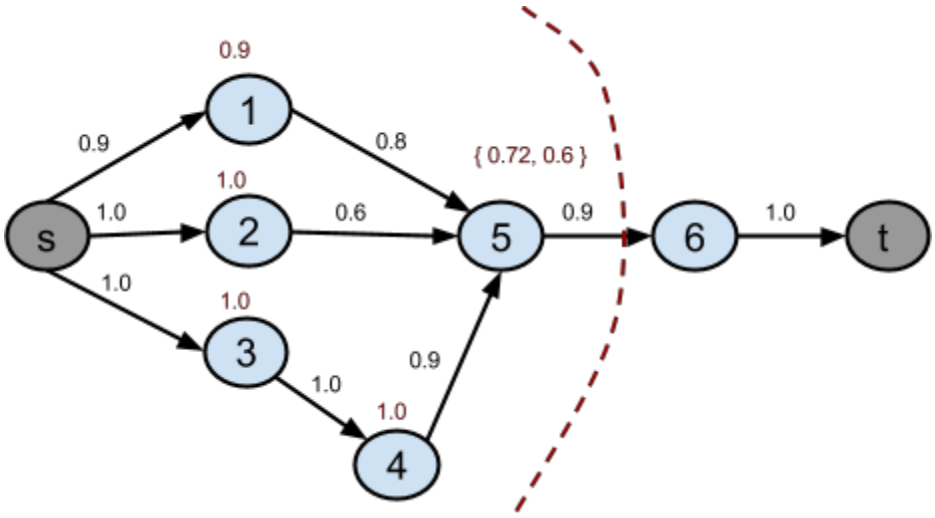
$trust_{i\to j} = trust_{i\to k} * trust_{k\to j}$.

Η εκτίμηση εμπιστοσύνης $trust_{s\to 4}$, υπολογίζεται από το μονοπάτι

- s→ 3 → 4, με τιμή $t_{s\to 4} = t_{s\to 3} * t_{3\to 4} = 1.0$.

Η εκτίμηση εμπιστοσύνης $trust_{s\to 5}$, υπολογίζεται από δύο μονοπάτια

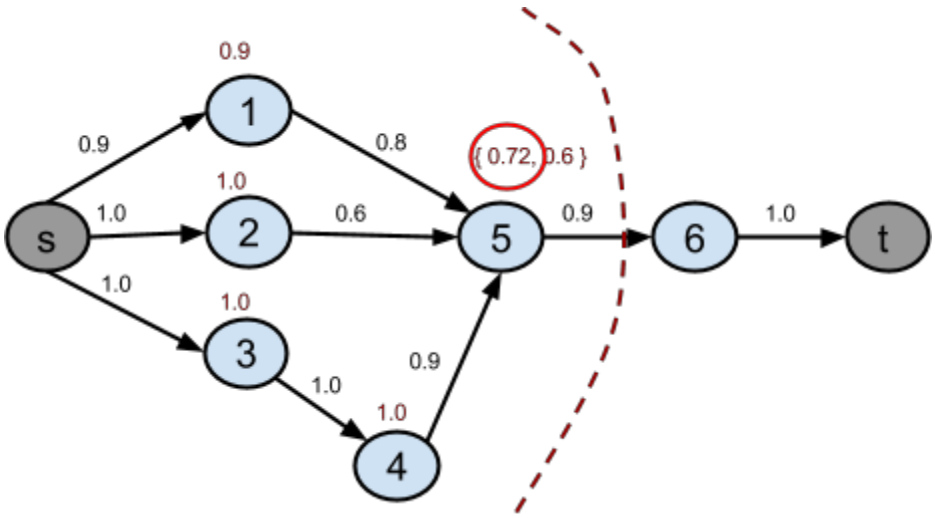
- s→ 1 → 5, με τιμή $t_{s\to 5} = t_{s\to 1} * t_{1\to 5} = 0.72$ και
- s→ 2 → 5, με τιμή $t_{s\to 5} = t_{s\to 2} * t_{2\to 5} = 0.6$.



Σχήμα 3α. Γράφος εμπιστοσύνης μετά το δεύτερο βήμα.

Αν στον ίδιο κόμβο καταλήγουν παραπάνω από μία ακμές εμπιστοσύνης, τότε επιλέγεται η μεγαλύτερη τιμή, $trust_{i \to j} = max\{ trust_{i \to k \to j}, \ldots, trust_{i \to m \to j} \}$.

Οπότε, όπως φαίνεται στο σχήμα 3β, έχουμε τελική εκτίμηση εμπιστοσύνης
$t_{s \to 5} = \max \{ t_{s \to 1 \to 5}, t_{s \to 2 \to 5} \} = \max \{ 0.72, 0.6 \} = 0.72$ για το μονοπάτι $s \to 5$.



Σχήμα 3β. Γράφος εμπιστοσύνης μετά το δεύτερο βήμα.

Στο τρίτο βήμα ο αλγόριθμος θα εξετάσει τους κόμβους {5, 6}.

Η εκτίμηση εμπιστοσύνης $trust_{s \to 5}$, υπολογίζεται και πάλι, αυτή τη φόρα από το μονοπάτι

- ❏ $s \to 3 \to 4 \to 5$, με τιμή $trust_{s \to 5} = trust_{s \to 4} * trust_{4 \to 5} = 1.0 * 0.9 = 0.9$.

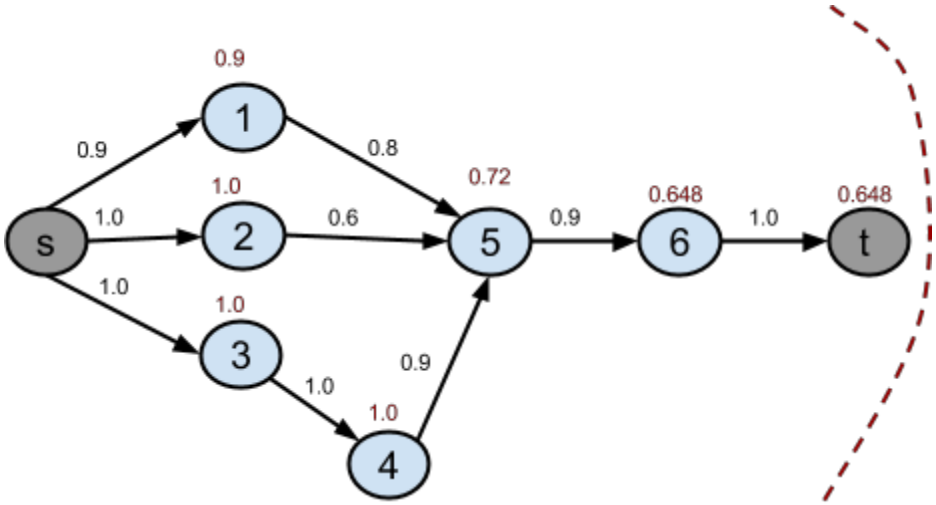
Η νέα τιμή 0.9 δε θα αντικαταστήσει την προηγούμενη εκτίμηση 0.72 (παρ'όλο που είναι μεγαλύτερη) γιατί το μονοπάτι s→ 3 → 4 → 5 έχει μήκος 3, μεγαλύτερο του κοντινότερου μονοπατιού (που έχει μήκος 2).

Η εκτίμηση εμπιστοσύνης $trust_{s \to 6}$, υπολογίζεται από το μονοπάτι

- ❏ s→ 5 → 6, με τιμή $trust_{s \to 6} = trust_{s \to 5} * trust_{5 \to 6} = 0.648$.

Τέλος, ο αλγόριθμος θα τερματίσει μετά από ένα ακόμα βήμα με τελική τιμή εμπιστοσύνης

- ❏ s→ 6 → t, με τιμή $trust_{s \to t} = trust_{s \to 6} * trust_{6 \to t} = 0.648$.



Σχήμα 4. Υπολογισμός τελικής εκτίμησης εμπιστοσύνης s→ t.