ΕΡΓΑΣΙΑ ΣΤΟ ΜΑΘΗΜΑ
Λ03Β ΑΛΓΟΡΙΘΜΟΙ ΔΙΚΤΥΩΝ & ΠΟΛΥΠΛΟΚΟΤΗΤΑ
ΦΛΕΒΑΡΗΣ 2004

Παρουσίαση του paper:

Increasing the Weight of Minimum Spanning Trees

Greg N. Frederickson and Roberto Solis- Oba

Journal of Algorithms 33, 244-266 (1999)

Φοιτητής: Κώστας Μανουβέλος, ΜΠΛΑ, 200209

1.ΕΙΣΑΓΩΓΗ

Στην εργασία αυτή μελετούμε την επίδραση που η αφαίρεση τυχών ακμών καθώς και οι αυξήσεις στα βάρη των ακμών ενός γράφου, έχουν στο βάρος των MSTs του.

Το *robustness problem* (RP) για MSTs είναι η εύρεση της μέγιστης αύξησης στο βάρος των MST του *G*, η οποία μπορεί να επιτευχθεί δαπανώντας ένα συγκεκριμένο budget για να αυξήσουμε τα βάρη των ακμών

- Στην διακριτή (discrete) περίπτωση του προβλήματος, στόχος μας είναι να βρούμε, για μια δοσμένη τιμή *k*, το σύνολο *k* ακμών τις οποίες αν αφαιρέσουμε από το γράφο θα προκληθεί η μέγιστη αύξηση στο βάρος των MSTs του.
- Για την συνεχή (continuous) περίπτωση του προβλήματος, στόχος μας είναι να βρούμε, για δοσμένη τιμή *B*, την μέγιστη πιθανή αύξηση στο βάρος των MSTs που μπορεί να επιτευχθεί με συνολικές αυξήσεις στα βάρη των ακμών ίσες με *B*.
- Η διακριτή περίπτωση του προβλήματος μας, έχει εμφανιστεί στη βιβλιογραφία με το όνομα, the most vital edges in the MST problem

ΔΙΑΚΡΙΤΗ ΠΕΡΙΠΤΩΣΗ (εισαγωγής)

- Στην εργασία αυτή αποδεικνύουμε ότι η διακριτή περίπτωση του RP για γενικότερα *k* είναι NP-complete, ακόμη και αν οι ακμές έχουν βάρος 0 ή 1.
- Παρουσιάζεται ο πρώτος προσεγγιστικός αλγόριθμος για το πρόβλημα αυτό, στην περίπτωση που τα βάρη των ακμών είναι τυχαίοι μη αρνητικοί πραγματικοί αριθμοί. Ο αλγόριθμος τρέχει σε χρόνο και για k>1 βρίσκει λύση της οποίας η τιμή είναι φορές η βέλτιστη.
- Η λύση επιτυγχάνεται με την επίλυση μιας σειράς από στιγμιότυπων του *maximum components problem*.
- Δοσμένης μιας τιμής και ενός γράφου G, το πρόβλημα αυτό είναι η επιλογή ενός συνόλου ακμών των οποίων η αφαίρεση από το G μεγιστοποιεί τον αριθμό των connected components του. Θα δώσουμε έναν (1/2)-approximation algorithm για το πρόβλημα αυτό, το οποίο είναι NP-complete, με μια απλή αναγωγή από το minimum *k*-cut problem [13].

ΣΥΝΕΧΗΣ ΠΕΡΙΠΤΩΣΗ (εισαγωγής)

- Για την συνεχή περίπτωση του RP δείχνουμε ότι, δοθέντος ενός budget *B*, οποιαδήποτε βέλτιστη λύση μπορεί να κατασκευαστεί incrementally με τέτοιο τρόπο ώστε κάθε μερική συνάρτηση είναι βέλτιστη για κάποια τιμή $B' <= B$.
- Αυτό μας επιτρέπει να σχεδιάσουμε έναν αποδοτικό greedy αλγόριθμο για το πρόβλημα. Ο πυρήνας του αλγορίθμου είναι μια διαδικασία όπου επιλέγουμε ένα υποσύνολο *S* ακμών, που μεγιστοποιεί το λόγο της αύξησης στο βάρος των MST του G προς το κόστος της μετατροπής του βάρους των ακμών στο *S*.
- Η επιλογή αυτή μπορεί να πραγματοποιηθεί αποδοτικά με έναν προσεκτικό συνδυασμό edge contraction και *graph strength* [3, 11] υπολογισμών. Δείχνουμε ότι το μέγιστο βάρος των MSTs του *G* που επιτυγχάνεται αυξάνοντας τα βάρη των ακμών είναι μια piecewise linear concave συνάρτηση *f* της budget τιμής.
- Εφόσον λύνοντας την συνεχή περίπτωση του RP μπορούμε να συναντήσουμε τα βάρη ενός εκθετικού αριθμού από MST, είναι περίεργο το πώς αυτή η περίπτωση δεν είναι NP-hard όπως η διακριτή αντίστοιχή της.
- Υπάρχει μια πληθώρα από προβλήματα στη θεωρία γράφων [2, 9, 22], στο scheduling [1], και στο transportation [6] με την ίδια ιδιότητα (διακριτή περίπτωση που είναι NP-hard και συνεχής που επιλύεται αποτελεσματικά).
- Έχει ενδιαφέρον ότι η λύση που παρουσιάζουμε εδώ για την συνεχή περίπτωση του RP για MSTs, μπορεί να επεκταθεί και σε τυχαία matroids.

ΥΠΟΛΟΙΠΗ ΕΡΓΑΣΙΑ (σχεδιάγραμμα)

- Στο κεφάλαιο 2.1 παρουσιάζουμε έναν (1/2)-προσεγγιστικό αλγόριθμο για το maximum components problem.
- Στο 2.2 παρουσιάζουμε έναν $\Omega(1/\log k)$-προσεγγιστικό αλγόριθμο για τη διακριτή περίπτωση του RP για MSTs.
- Στο 2.3 προσθέτουμε ένα βήμα πριν την εκτέλεση του προσεγγιστικού αλγορίθμου μας, το οποίο μειώνει την συνολική χρονική πολυπλοκότητά του.
- Στο 2.4 αποδεικνύουμε ότι το RP για MSTs είναι NP-hard ακόμη και όταν το βάρος κάθε ακμής είναι 0 ή 1.
- Στο τέλος και στο κεφάλαιο 3 παρουσιάζουμε τον αλγόριθμο που επιλύει την συνεχή περίπτωση του RP.

2. ΔΙΑΚΡΙΤΗ ΠΕΡΙΠΤΩΣΗ

- Το διακριτό RP για MSTs είναι το εξής: Δοθέντος ενός μη κατευθυνόμενου γράφου G και μιας ακέραιας τιμής k, βρες ένα σύνολο k ακμών των οποίων η αφαίρεση από το G, μεγιστοποιεί τα βάρη των MSTs του γράφου που προκύπτει.
- Υποθέτουμε ότι η συνεκτικότητα των ακμών του G είναι μεγαλύτερη από k, αλλιώς το πρόβλημα γίνεται τετριμμένο εφόσον η αφαίρεση των ακμών σε ένα minimum cardinality cut αυξάνει τα βάρη των MSTs και του δίνει άπειρη τιμή.
- Έστω ένας μη κατευθυνόμενος γράφος $G = (V,E,w)$. Έστω μια ακμή , και έστω G' ένας υπό- γράφος του G που περιλαμβάνει μόνο τις ακμές με βάρος το πολύ $w(e)$. Εάν αφαιρέσουμε από το G οποιοδήποτε υποσύνολο S ακμών, που χωρίζει το G' σε $i+1$ connected components, τότε κάθε MST του $(V,E-S,w)$ θα είχε τουλάχιστον i ακμές βάρους μεγαλύτερο από $w(e)$.
- Χρησιμοποιούμε αυτήν την ιδέα για να κατασκευάσουμε έναν προσεγγιστικό αλγόριθμο για την διακριτή περίπτωση του RP για MSTs. Παρόλα αυτά, χρειάζεται να γνωρίζουμε για δοσμένη τιμή k,πώς να επιλέξουμε ένα υποσύνολο από k ακμές των οποίων η αφαίρεση από το γράφο τον διαχωρίζουν στο μέγιστο δυνατό αριθμό από connected components.

2.1 Το *Maximum Components problem* (MCP)

- Τροποποιούμε έναν αλγόριθμο από τους Saran και Vazirani [24] για να επιτύχουμε έναν (1/2)-προσεγγιστικό αλγόριθμο για το MCP. Ο αλγόριθμός μας χρησιμοποιεί μια greedy εκδοχή για να τεμαχίσουμε το γράφο σε όσο το δυνατό περισσότερα κομμάτια, αφαιρώντας ακμές από αυτό.

- ALGORITHM dice *(G,k)*.
- **repeat**
- Αφαίρεσε από το *G* τις ακμές σε ένα ελάχιστο cut, το οποίο διαχωρίζει ένα από τα connected components του *G*.
- **until** *k'<=k* ακμές έχουν αφαιρεθεί και κανένα επιπλέον cut δεν μπορεί να δημιουργηθεί με τις εναπομείναντες *k-k'* ακμές.
- Δώσε ως έξοδο το σύνολο των ακμών που αφαιρέθηκαν από το *G*.

- Πριν το θεώρημα 2.1 βλέπουμε το ΛΗΜΜΑ 2.1 (απόδειξη με επαγωγή στο *i*, στις σημειώσεις, σελ.5)

ΘΕΩΡΗΜΑ 2.1. *Δοθέντος ενός μη κατευθυνόμενου γράφου G, ο αλγόριθμος dice βρίσκει ένα σύνολο S το πολύ k ακμών, των οποίων η αφαίρεση από το G, το διαχωρίζουν σε d>d*/2 components, όπου d* είναι ο μέγιστος αριθμός των components που μπορούν να σχηματιστούν με την αφαίρεση οποιουδήποτε συνόλου k ακμών από το G.*

- Απόδειξη θεωρήματος 2.1. Για να απλοποιήσουμε την απόδειξη υποθέτουμε πρώτα ότι ο γράφος είναι συνεκτικός. Στην συνέχεια δείχνουμε ότι το θεώρημα ισχύει και για μη συνεκτικούς γράφους.
- Απόδειξη στις σημειώσεις, σελ.5-6

ΘΕΩΡΗΜΑ 2.2 *Ο αλγόριθμος dice τρέχει σε χρόνο* $O(km+k^2 n\log n)$.

- Απόδειξη. Μπορούμε να τροποποιήσουμε τον αλγόριθμο dice έτσι ώστε σε κάθε επανάληψη, εκτός της πρώτης, να υπολογίζει μόνον connectivity cuts για τα δύο components που δημιουργήθηκαν στην αμέσως προηγούμενη επανάληψη. Ο αλγόριθμος του Gabow [10], βρίσκει την edge connectivity λ και ένα connectivity cut, για δοσμένο γράφο, σε χρόνο $O(m+\lambda^2 n\log n(n/\lambda))$.
- Ένας απλός υπολογισμός δείχνει ότι ο αλγόριθμος dice τρέχει σε χρόνο $O(km+k^2 n\log n)$ εάν εφαρμοστεί μαζί με τον αλγόριθμο του Gabow.

2.2 *Το RP για MST*

- Δοθέντος ενός μη κατευθυνόμενου γράφου $G=(V,E,w)$ και ενός ακεραίου k, θέλουμε να βρούμε την μέγιστη αύξηση στο βάρος των MSTs του G, που επιτυγχάνεται με την αφαίρεση k ακμών από αυτό. Το σχήμα 1 παρουσιάζει ένα απλό παράδειγμα.
- Η τεχνική που ακολουθούμε για να προσεγγίσουμε την λύση του προβλήματος, είναι να βάλουμε μέσα σε κάθε MST του G, όσες πιο πολλές ακμές με «μεγάλο» βάρος γίνεται.
- Εύκολα βλέπουμε ότι μια ακμή e ανήκει σε ένα MST του G αν και μόνον αν η e είναι μια ελαχίστου βάρους ακμή σε κάποιο cut του G.
- Αυτό σημαίνει ότι αν μια ακμή e' δεν ανήκει σε κανένα MST του G, μπορούμε να διαγράψουμε τις ακμές με βάρος μικρότερο από $w(e')$ σε κάποιο cut που χωρίζει τους κόμβους της e', για να πάρουμε έναν νέο γράφο στον οποίο η e' να ανήκει σε τουλάχιστον ένα MST. Από την στιγμή που όλα τα MSTs ενός γράφου έχουν τον ίδιο αριθμό ακμών ενός δοσμένου βάρους, η διαδικασία που μόλις περιγράψαμε μπορεί να χρησιμοποιηθεί για να προσθέσουμε μια ακμή βάρους $w(e')$ σε κάθε MST του γράφου που προκύπτει.

ALGORITHM *slice_n_dice(G=(V,E,w),k)*
max_mst_wgt=0
for each distinct weight w_i in G **do**
  S=dice(Gw_i *,k)*
  mst_wgr=weight of a minimum spanning tree of (*V,E-S,w)*
  if *mst_wgt>max_mst_wgt* **then**
    max_mst_wgt=mst_wgt
$S_{s\&d}$ $=S$
  end if
end for
Output $S_{s\&d}$ and *max_mst_wgt*.

ΛΗΜΜΑ 2.2 *Εάν κάθε ακμή του G έχει βάρος της μορφής* 2^i*,για κάποιον ακέραιο i, τότε το performance ratio του slice_n_dice είναι ίσο με* $r=w(T_{s\&d})/w(T^*)>=1/(3+2\log(k+1))$.

Απόδειξη στις σημειώσεις σελ 6-7.

ΛΗΜΜΑ 2.3 *Το μέγιστο βάρος των MSTs ενός γράφου G=(V,E,w) που μπορεί να επιτευχθεί με την αφαίρεση k ακμών από το G, δεν είναι μεγαλύτερο από δύο φορές η αντίστοιχη βέλτιστη τιμή για το γράφο G=(V,E,w') ,ο οποίος δημιουργείται με στρογγυλοποίηση προς τα κάτω (rounding down), του βάρους κάθε ακμής στο E στη πλησιέστερη σε αυτό δύναμη του δύο.*

Απόδειξη σελ.7

ΘΕΩΡΗΜΑ 2.3. *Το performance ratio του slice_n_dice όταν η είσοδος (ο γράφος) έχει ακμές με τυχαία μη αρνητικά βάρη είναι* $r=w(T_{s\&d})/w(T^*)>=1/(2+4\log(k+1))$.

ΛΗΜΜΑ 2.4. *Εάν οι ακμές ενός γράφου G=(V,E,w) έχουν τυχαία destruction costs, το performance ratio του slice_n_dice είναι* $r>=1/(1+4\log m)$.

Απόδειξη σελ.9

.

2.3. *Ένας γρηγορότερος αλγόριθμος*

ΛΗΜΜΑ 2.5. $w(T_{s\&d})/w(T'_{s\&d})<2$

απόδειξη στη σελίδα .10.

Για ένα γράφο *G=(V,E,w)* με τυχαία μη αρνητικά βάρη, ο τροποποιημένος μας αλγόριθμος slice_n_dice πρώτα στρογγυλοποιεί μπρος τα κάτω το βάρος κάθε ακμής στην πλησιέστερη σε αυτό δύναμη του 2. Εάν κάποια ακμή έχει βάρος μικρότερο από 1, τότε πολλαπλασιάζει όλα τα βάρη με 2^s, όπου 2^{-s} είναι το μικρότερο βάρος ακμής που προκύπτει μετά την στρογγυλοποίηση. Αυτό το scaling του βάρους των ακμών, δεν επηρεάζει το performance ratio του αλγορίθμου και επιτρέπει στα βάρη των ακμών που είναι ακέραιοι να παραμένουν δυνάμεις του 2.

ΘΕΩΡΗΜΑ 2.4. *Το performance ratio του τροποποιημένου αλγορίθμου slice_n_dice που μόλις περιγράψαμε είναι r>=1/4(4+8log(k+1)*. Απόδειξη. Το θεώρημα προκύπτει από τα λήμματα 2.2, 2.3, και 2.5

ΘΕΩΡΗΜΑ 2.5. *Η χρονική πολυπλοκότητα του τροποποιημένου slice_n_dice αλγορίθμου, είναι O(klogn(m+knlogn)).*

Απόδειξη. Χρησιμοποιούμε τον αλγόριθμο του Gabow [10] για να επιλύσουμε την διαδικασία που χρειάζεται για να βρούμε το βάρος μιας μέγιστης non- redundant ακμής σε ένα γράφο.

2.4 *NP-Completeness*

- Σε αυτό το κεφάλαιο αποδεικνύουμε, ότι το πρόβλημα του να επιλέξουμε *k* ακμές, τις οποίες όταν αφαιρέσουμε από ένα γράφο, μεγιστοποιείται το βάρος των MSTs του, είναι NP-hard, ακόμη και όταν τα βάρη των ακμών του είναι 0 ή 1. Η αναγωγή είναι από το minimum *k-cut* problem.
- ΘΕΩΡΗΜΑ 2.6. *Έστω G=(V,E,w) ένας γράφος του οποίου οι ακμές έχουν βάρος είτε 0 είτε 1, και έστω K μια μη-αρνητική ακέραιη τιμή. Το πρόβλημα του να αποφασίσουμε εάν υπάρχει ένα σύνολο S το πολύ k ακμών, τέτοιο που οποιοδήποτε MST του γράφου (V,E-S,w) έχει βάρος τουλάχιστον K, είναι NP-complete.*
- Απόδειξη. Το πρόβλημα είναι εμφανώς στο NP. Για να δείξουμε ότι είναι NP-hard δίνουμε μια αναγωγή από το minimum *k*-cut problem, στη σελίδα 12.

3.ΣΥΝΕΧΗΣ ΠΕΡΙΠΤΩΣΗ

- Σε αυτό το κεφάλαιο μελετούμε το πρόβλημα του να καθορίσουμε την μέγιστη αύξηση στα βάρη των MSTs ενός γράφου, που δημιουργείται από μια πεπερασμένη αύξηση στα βάρη των ακμών του.
- Σε κάθε ακμή έχουμε αναθέσει ένα rate *c(e)* και υποθέτουμε μια συνάρτηση κόστους που χρεώνει *c(e)*δ*, όταν αυξηθεί το βάρος της *e* κατά *δ*.
- Το συνολικό κόστος των αλλαγών στα βάρη των ακμών δεν πρέπει να υπερβαίνει κάποια δοσμένη budget τιμή *B*.
- Στο σχήμα 3 παρουσιάζεται ένα στιγμιότυπο του προβλήματος.
- Ο αλγόριθμος για να λύσουμε το συνεχές RP για MST είναι στην σελίδα 13.

ΛΗΜΜΑ 3.1. Σελ.14

ΘΕΩΡΗΜΑ 3.1 *Ο αλγόριθμος raise_mst υπολογίζει την μέγιστη αύξηση στα βάρη των MST ενός γράφου G=(V,E,w), η οποία επιτυγχάνεται ξοδεύοντας ένα δοσμένο budget B που αυξάνει τα βάρη των ακμών του.*
Απόδειξη σελ.15.

3.1 *Βρίσκοντας ένα σύνολο ακμών με ελάχιστο κόστος ανά μονάδα αύξησης*

ΙΔΙΟΤΗΤΑ 3.1 Το T είναι MST του G αν και μόνον αν περιέχει ένα spanning tree για κάθε nontrivial component σε κάθε Gw_i.

ΑΛΓΟΡΙΘΜΟΣ OPTIMAL_SUBSET στην σελίδα 16.
ΛΗΜΜΑ 3.2. Ο αλγόριθμος optimal_subset τρέχει σε χρόνο Ο($n^2 m \log(n^2/m)$).
Απόδειξη στην σελίδα 16.

ΛΗΜΜΑ 3.3 Ο αλγόριθμος raise_mst πραγματοποιεί το πολύ (m-1)(n-1)+1 επαναλήψεις. Σελ.17

ΘΕΩΡΗΜΑ 3.2. Η χρονική πολυπλοκότητα του αλγορίθμου raise_mst είναι $O(n^3 m^2 \log(n^2/m)$.

ΛΗΜΜΑ 3.4 Η συνάρτηση f_G είναι concave, μη φθίνουσα και piecewise linear με O(mn) breakpoints. Σελ.17-18.

ΣΥΜΠΕΡΑΣΜΑΤΑ & ΒΙΒΛΙΟΓΡΑΦΙΑ