Multicut and Integer Multicomodity Flow in Trees (chap. 18)

Αγγελής Γιώργος

Εισαγωγή

- Εύρεση αλγορίθμου με approx ratio 2 και ½ για τα προβλήματα “minimum multicut” και “integer multicommodity flow” αντίστοιχα
  - Χρήση primal-dual schema

Ορισμοί

- **<u>Minimum multicut</u>**: Έστω G=(V,E) μη κατευθυνόμενος γράφος, κάθε ακμή e του οποίου έχει χωρητικότητα $c_e>0$.

  Έστω $(s_1, t_1)\dots(s_k, t_k)$ ένα σύνολο k ζευγών πηγής-προορισμού. (το κάθε ζεύγος είναι ξεχωριστό αλλά όχι απαραίτητα και όλοι οι κόμβοι.. μεταξύ τους βλ. παράδειγμα)

  Ορίζουμε ως *multicut* ένα σύνολο ακμών, που αν αφαιρεθούν διαχωρίζουν κάθε ένα από τα k ζεύγη.

  Στόχος: η *εύρεση multicut ελαχίστου κόστους*

- Το πρόβλημα "minimum multicut" είναι <u>**γενίκευση**</u> των "minimum s-t cut"(για k=1) και "multiway cut"(ο χωρισμός των s_1, s_2, …, s_k ισοδυναμεί με χωρισμό όλων των ζευγών (${s_i}$, s_j), $1\leq i\leq j \leq k$. ή αλλιώς τα (s_i, s_j) είναι

  κλίκα)

- Ξέρουμε ότι το "multiway cut" είναι NP-hard για k=3 άρα και το **"minimum multicut" είναι NP-hard για k≥3**

! Προσοχή: περιοριζόμαστε στην περίπτωση που ο γράφος μας είναι **<u>δέντρο</u>** για τον 2-προσσεγγιστικό αλγόριθμο

(στη γενική περίπτωση υπάρχει O(logk) προσσεγ-γιστικός αλγόριθμος)

Καταστρώνουμε το primal integer LP

$$
\begin{array}{lll}
\text{minimize} & \sum_{e \in E} c_e d_e & \\
\text{subject to} & \sum_{e \in p_i} d_e \geq 1, & i \in \{1, \ldots, k\} \\
& d_e \in \{0, 1\}, & e \in E
\end{array}
$$

- d_e = παίρνω/δεν παίρνω ακμή στο cut
- Δέντρο -> Μοναδικό μονοπάτι p_i μεταξύ s_i, t_i
- τι σημαίνει η 2η συνθήκη…

Παίρνουμε το relaxation του

$$\begin{aligned} &\text{minimize} && \sum_{e \in E} c_e d_e && \\ &\text{subject to} && \sum_{e \in p_i} d_e \geq 1, && i \in \{1, \ldots, k\} \\ & && d_e \geq 0, && e \in E \end{aligned}$$

- Εδώ d_e -> “ποσοστό κατά το οποίο βάζουμε την ακμή e στο cut”
- {Άθροισμα d_e κατά μήκος p_i } ≥ 1

Ορισμός Dual προβλήματος

- **Multicommodity flow problem**: από κάθε πηγή s_i στέλνουμε f_i μονάδες ενός ξεχωριστού αγαθού i, κατά μήκος του p_i προς τον προορισμό t_i.
- Η f_i είναι η dual μεταβλητή μας
- Σκοπός η μεγιστοποίηση της συνολικής ροής
- Ροές με αντίθετη κατεύθυνση σε μια ακμή αθροίζονται(βλ. Παράδειγμα)

Διατύπωση του dual προβλήματος

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{i=1}^{k} f_i & \\ \text{subject to} & \displaystyle\sum_{i:\ e\in p_i} f_i \le c_e, & e \in E \\ & f_i \ge 0, & i \in \{1,\ldots,k\} \end{array}$$

- Όμως αυτό είναι fractional πρόβλημα.
- Εμείς στον αλγόριθμό μας θέλουμε και το dual να είναι integral!

- Είναι ακριβώς το ίδιο αλλά οι χωρητικότητες των ακμών και οι ποσότητες που στέλνονται είναι **ακέραιες**
- Και σε αυτό θα περιοριστούμε σε γράφους που είναι **δέντρα**

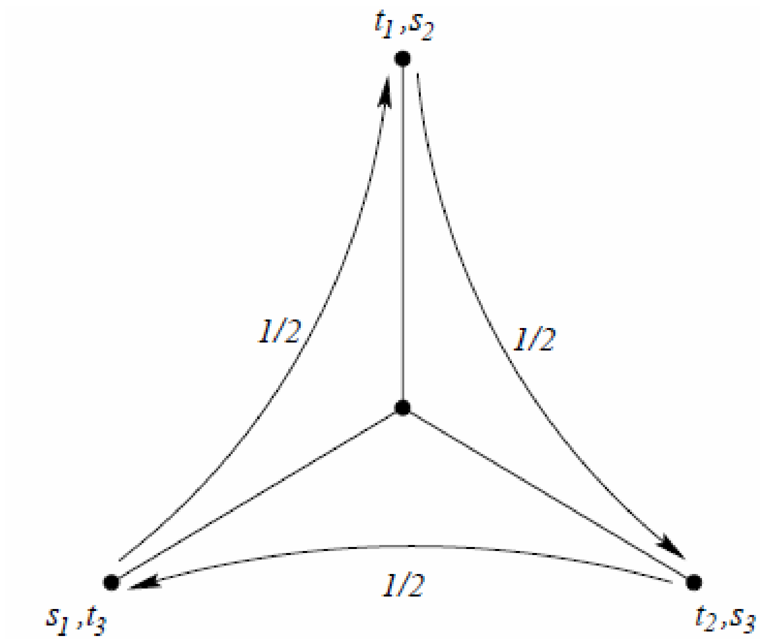
- Το integral primal εφόσον είναι πρόβλημα ελαχιστοποίησης θα φράσεται από κάτω από το fractional primal
- Αλλά και το integral dual επειδή είναι μεγιστοποίησης θα φράσεται από πάνω από το αντίστοιχα fractional του
- Τέλος τα optima των fractional προβλημάτων θα συμπίπτουν λόγω του LP – duality theorem

- Συνοψίζοντας:

$$OPT_{i,dual} \leq OPT_{f,dual} = OPT_{f,prim} \leq OPT_{i,prim}$$

Παράδειγμα

- Έστω ο ακόλουθος γράφος με c_e = 1 για κάθε ακμή



- Για τις δοσμένες ροές -> flow 3/2
- Παίρνοντας d_e = 1/2 -> cut κόστους 3/2

Άρα OPT_f =3/2

Παράδειγμα (συνέχεια)

- Αντίθετα:
  - Η μέγιστη ροή που μπορούμε να στείλουμε στο Integral προβλημα είναι 1
  - Η τομή ελαχίστου κόστους 2!

Primal-dual based Algorithm

- Ο Αλγόριθμος βρίσκει ταυτόχρονα ένα multicut και ένα Integer multicommodity flow τα οποία απέχουν το πολύ 2 και ½ αντίστοιχα απο τα αντίστοιχα optima.
- Προυπόθεση: ο γράφος είναι δέντρο!
- <u>saturated edge</u>: ροή = χωρητικότητα

- Από τις comlementary slackness conditions:

τη μεν μια την χρησιμοποιούμε tight, δηλ με a=1:

Primal conditions: For each $e \in E$, $d_e \neq 0 \Rightarrow \sum_{i:\, e \in p_i} f_i = c_e$.
Equivalently, *any edge picked in the multicut must be saturated.*

Η συνθήκη αυτή είναι αναγκαία για κάθε ακμή στο cut μας! Πρακτικά θα την χρησιμοποιήσουμε σαν κριτήριο για το ποιες ακμές θα βάζουμε στο cut

- Το δεύτερο σετ conditions το χρησιμοποιούμε relaxed, β=2:

Relaxed dual conditions: Let us relax the dual conditions to:

For each $i \in \{1, \ldots, k\}$, $f_i \neq 0 \Rightarrow \sum_{e \in p_i} d_e \leq 2$.

Equivalently, *at most two edges can be picked from a path carrying non-zero flow.*

Ο λόγος που το χρησιμοποιούμε **relaxed** και όχι **tight**: αυτό είναι το καλύτερο που μπορεί να εγγυηθεί ο αλγόριθμός μας!(αν ισχυε και αυτή **tight** θα παίρναμε το **optimal!**)

Επίσης ορίζουμε:

- vertex depth: το μήκος του μονοπατιού του κόμβου από την ρίζα του δέντρου
- lowest common ancestor δύο κόμβων – lca(u,v) – είναι ο κόμβος ελαχίστου βάθους στο μονοπάτι μεταξύ u-v.

Ο Αλγόριθμος

Algorithm 14.8 (Multicut and integer multicommodity flow in trees)

1. **Initialization:** $f \leftarrow \mathbf{0}$; $D \leftarrow \emptyset$
2. **Flow routing:** For each vertex v, in non-increasing order of depth, do:

   For each pair (s_i, t_i) such that $\mathrm{lca}(s_i, t_i) = v$, greedily route integral flow from s_i to t_i.

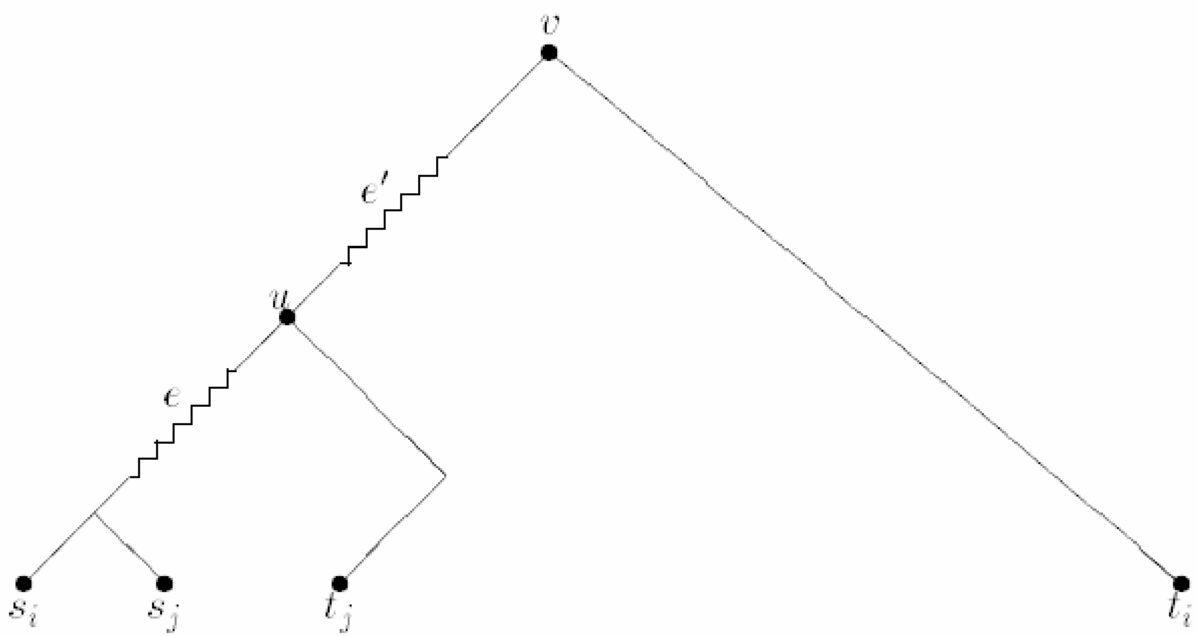
   Add to D all edges that got saturated in the current iteration, in arbitrary order.
3. Let $e_1, e_2, \ldots, e_l$ be the ordered list of edges in D.
4. **Reverse delete:** For $j = l$ downto 1 do:

   If $D - \{e_j\}$ is a multicut in G, then $D \leftarrow D - \{e_j\}$
5. Output the flow and multicut D.

- *Lemma: Let (s_i,t_i) be a pair with nonzero flow, and let lca(s_i,t_i) = v. At most one edge is picked in the multicut from each of the two paths, s_i-v, t_i-v*

- *Δηλαδή η απαίτηση της 2ης complementary slackness conditon*
- *Είναι αυτό που εξασφαλίζει το reverse delete step!*

Proof : The argument is the same for each path. Suppose two edges e and e' are picked from the s_i–v path, with e being the deeper edge. Clearly, e' must be in D all through reverse delete. Consider the moment during reverse delete when edge e is being tested. Since e is not discarded, there must be a pair, say (s_j, t_j), such that e is the only edge of D on the s_j–t_j path. Let u be the lowest common ancestor of s_j and t_j. Since e' does not lie on the s_j–t_j path, u must be deeper than e', and hence deeper than v. After u has been processed, D must contain an edge from the s_j–t_j path, say e''.



Since non-zero flow has been routed from s_i to t_i, e must be added during or after the iteration in which v is processed. Since v is an ancestor of u, e is added after e''. So e'' must be in D when e is being tested. This contradicts the fact that at this moment e is the only edge of D on the s_j–t_j path. $\square$

- *Theorem: Ο αλγόριθμός μας πετυχαίνει approx <u>gurantees 2 και ½</u> για το <u>minimum multicut</u> προβλημα και το <u>maximum integer multicommodity flow</u> αντίστοιχα*

Proof : The flow found at the end of Step 2 is maximal, and since at this point D contains all the saturated edges, D is a multicut. Since the reverse delete step only discards redundant edges, D is a multicut after this step as well. Thus feasible solutions have been found for both, the flow and the multicut.

Since each edge in the multicut is saturated, the primal conditions are satisfied. By lemma 14.9, at most two edges have been picked in the multicut from each path carrying non-zero flow. Therefore, the relaxed dual conditions are also satisfied. Hence, by Proposition 13.1, the capacity of the multicut found is within twice the flow. Since a feasible flow is a lower bound on the optimal multicut, and a feasible multicut is an upper bound on the optimal integer multicommodity flow, the claim follows. $\square$

$$\left.\begin{array}{l} Feas\ flow \le OPT\ mul \\ OPT\ flow \le Feas.Mul \\ Feas.Mul \le 2\,Feas.flow \end{array}\right\} OPT\ flow \le Feas\ mul \le 2\,Feas\ flow \le 2OPT\ mul$$

- Για γενικούς γράφους έχουμε approx ratio O(log k) για το <u>minimum multicut</u> προβλημα (Βασίλης)
- Αντίθετα για το integer multicommodity flow problem δεν υπάρχει μη-τετριμμένος προσσεγγιστικός αλγόριθμος για γενικούς γράφους
- Το integrality gap του maximum flow για επίπεδους γράφους με k ζεύγη πηγής-προορισμού είναι k/2

Ισοδυναμίες/Μετασχηματισμοί Προβλημάτων

1. Το πρόβλημα minimum weight vertex cover σε γενικό γράφο είναι ισοδύναμο με την εύρεση minimum multicut σε δέντρα ύψους 1
2. Το minimum multicut πρόβλημα σε δέντρα μπορεί να ειδωθεί σαν (μια περιορισμένη κατηγορία) minimum weight set cover

1. <u>minimum weight vertex cover - minimum multicut</u>

- ακμή δέντρου **->** κόμβος γράφου ίσου κόστους
- ζεύγος ακμών που ενώνουν πηγή – προορισμό στο δέντρο **->** ακμή στον γράφο μεταξύ αντίστοιχων κόμβων

2. minimum multicut - minimum weight set cover

- Τα στοιχεία που θέλουμε να καλύψουμε στο set cover είναι τα (μοναδικά) s_i-t_i paths.
- Τα σύνολα αντιστοιχίζονται στις ακμές: ένα σύνολο περιλαμβάνει σαν στοιχεία του όλα τα paths στα οποία η ακμή αυτή ανήκει. Το βάρος του συνόλου είναι το βάρος της ακμής

- Σημ.: Δεν μπορούν πάντως όλα τα set cover προβλήματα να εκφραστούν σαν minimum multicut πρόβλημα σε δέντρο (αυτά που μπορούν λέγονται tree representable set systems)