

Εθνικό Μετσόβιο Πολυτεχνείο  
Σχολή ΗΜ&ΜΥ  
Αλγόριθμοι και Πολυπλοκότητα  
7<sup>ο</sup> εξάμηνο, Ροή Λ  
Ακαδημαϊκή περίοδος: 2011-2012



## 4<sup>η</sup> Σειρά Γραπτών Ασκήσεων

Γερακάρης Βασίλης  
<vgerak@gmail.com>  
Α.Μ.: 03108092

19 Φεβρουαρίου 2012

# 1 Παιχνίδι Επιλογής Ακμών σε Κατευθυνόμενο Ακυκλικό Γράφημα

FILL ME 1!

## 2 Σχεδιασμός Ταξιδιού (DPV 4.13)

- i) Μία τέτοια διαδρομή είναι εφικτό να βρεθεί (ή να αποδειχτεί η μη ύπαρξή της) σε γραμμικό χρόνο. Κάνοντας ένα BFS στο γράφημα απορρίπτωντας κάθε ακμή η οποία έχει βάρος  $w > L$  βρίσκουμε μία τέτοια διαδρομή, αν υπάρχει σε χρόνο  $O(|E| + |V|)$ .
- ii) Για να υπολογίσουμε την ελάχιστη αυτονομία θα χρησιμοποιήσουμε μια τροποποιημένη μορφή του αλγορίθμου του Dijkstra. Όπως ο Dijkstra κρατάει ως δεδομένο την ελάχιστη απόσταση ως ένα κόμβο και "αναπτύσσει" τον κόμβο με τη λιγότερη απόσταση, εδώ θα κρατάμε ως δεδομένο την ελάχιστη αυτονομία καυσίμου (το ελάχιστο από τα μέγιστα βάρη ακμών) για να φτάσουμε σε ένα κόμβο και θα συνεχίζουμε την αναζήτηση από τον κόμβο με τη μικρότερη τιμή. Δηλαδή:

---

**Algorithm 1** Minimum tank capacity

---

```
1: procedure minCapacity( $(G(V, E, w), s, t)$ )
2:   for all  $u \in V$  do
3:      $D[u] \leftarrow \infty$ 
4:    $D[s] \leftarrow 0; S \leftarrow \emptyset$ 
5:   while  $|S| < |V|$  do
6:      $u \notin S : D[u] = \min_{v \notin S} \{D[v]\}$ 
7:      $S \leftarrow S \cup \{u\}$ 
8:     for all  $v \in AdjList[u]$  do
9:       if  $D[v] > \max(D[u], w[u, v])$  then
10:         $D[v] \leftarrow \max(D[u], w[u, v])$ 
11:   return  $D[t]$ 
```

---

Ο αλγόριθμος αυτός έχει πολυπλοκότητα  $O(E \log V)$  (με binary heaps) ή  $O(E + V \log V)$  (με fibonacci heaps), όσο η εκτέλεση του αλγορίθμου του Dijkstra

## 3 Διαχωρισμός Γραφήματος

- i) Έστω ότι δεν είναι το συντομότερο μονοπάτι. Θα υπάρχει δηλαδή μια ακμή  $e' \notin T$  με  $w(e') < w(e)$ , που ακουμπάει στον κόμβο  $u$ . Αυτό είναι άτοπο, αφού ο αλγόριθμος του Kruskal (που ταξινομεί τις ακμές) θα την είχε επιλέξει για το MST προτού επεξεργαστεί την  $e$  που θεωρούμε πως ανήκει στο MST.
- ii) Υπολογίζουμε το MST  $T$  με έναν από τους γνωστούς αλγορίθμους (σε ελάχιστο χρόνο  $O(|E| + |V| \log |V|)$ ). Στη συνέχεια βρίσκουμε σε γραμμικό χρόνο τη μεγαλύτερου βάρους ακμή και την αφαιρούμε από το  $T$ . Οι κόμβοι στις 2 συνεκτικές συνιστώσες που προκύπτουν ανήκουν στα υποσύνολα  $S_1, S_2$  της επιθυμητής διαμέρισης.

Το βέλτιστο της επιλογής μας προκύπτει λογικά: Οι μεγαλύτερου βάρους ακμές που δεν είναι αναγκαίες για τη συνεκτικότητα του δέντρου αφαιρούνται από τη διαδικασία εύρεσης του MST. Σκοπός μας είναι να χωρίσουμε το γράφο σε 2 συνιστώσες, μεγιστοποιώντας την απόσταση μεταξύ των συνιστωσών. Το ελάχιστο

κόστος που μπορούμε λοιπόν να "πληρώσουμε" είναι όσο μια ακμή που ανήκει στο MST, και για να μεγιστοποιήσουμε αυτή την ποσότητα, επιλέγουμε τη μέγιστη.

## **4 Παιχνίδια εξουσίας**

FILL ME 4!

## **5 Αναγωγές και NP-Πληρότητα**

### **5.1 Dense subgraph**

Dense fill!

### **5.2 Μακρύ μονοπάτι**

Μακρύ indeed!

### **5.3 Feedback Vertex Set - Directed**

directed!

### **5.4 Feedback Vertex Set - Undirected**

undirected!