



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

3η Γραπτή Άσκηση

Αλγόριθμοι & Πολυπλοκότητα

Σπουδαστής:
Διονύσης ΖΗΝΔΡΟΣ (06601)
<dionyziz@gmail.com>

Διδάσκοντες:
Στάθης ΖΑΧΟΣ
Δημήτρης ΦΩΤΑΚΗΣ

24 Ιανουαρίου 2012

Άσκηση 1

Το πρόβλημα μοντελοποιείται αν το αντιπροσωπεύσουμε με έναν γράφο στον οποίο κάθε ταινία είναι ένας κόμβος και κάθε συνδρομητής είναι μία ακμή ανάμεσα στις δύο ταινίες που θέλει να δει. Στη συνέχεια, αρκεί να χρωματίσουμε το γράφο με δύο χρώματα έτσι ώστε καμία ακμή να μην πατάει στο ίδιο χρώμα δύο φορές· με αυτό τον τρόπο ελέγχουμε αν ο γράφος μας είναι διμερής που είναι ουσιαστικά και το ζητούμενο του προβλήματος.

Για να ελέγξουμε αν ο γράφος είναι διμερής, χρωματίζουμε μία κορυφή αυθαίρετα και στη συνέχεια για κάθε χρωματισμένη κορυφή χρωματίζουμε τους γείτονές της με το αντίθετο χρώμα έως ότου ολοκληρωθεί η διαδικασία ή καταλήξουμε σε αντίφαση. Αυτό πρέπει να το επαναλάβουμε για κάθε συνεκτική συνιστώσα του γράφου μας.

Λήμμα. *Ο αλγόριθμος ελέγχου διμερούς γράφου είναι ορθός.*

Απόδειξη. Η ορθότητα του αλγορίθμου προκύπτει ως εξής. Ο αρχικός κόμβος κάθε συνιστώσας σε έναν έγκυρο διμερή διαχωρισμό, αν υπάρχει, μπορεί να χρωματιστεί αυθαίρετα. Στη συνέχεια, αν ο γράφος μας είναι πραγματικά διμερής, η αναζήτηση στη συγκεκριμένη συνιστώσα θα πρέπει να χρωματίσει με σωστό τρόπο όλους τους κόμβους χωρίς να αποτύχει. Άρα αν ο γράφος είναι πραγματικά διμερής, τότε ο χρωματισμός θα πρέπει να είναι επιτυχής. Αντίστροφα, αν έχουμε επιτυχή χρωματισμό, και επειδή κάθε χρώμα είναι αμετάκλητο, τότε θα έχουμε ελέγξει όλες τις ακμές για αντιφάσεις και καμία δεν θα έχει βρεθεί. Συνεπώς ο γράφος θα είναι διμερής. Αυτό ισχύει για κάθε συνεκτική συνιστώσα του γράφου και έτσι ολοκληρώνεται η απόδειξη. \square

Algorithm 1 Άσκηση 1

```
1: procedure BIPARTITE( $V, E$ )
2:    $unexplored \leftarrow \{V\}$ 
3:   for  $v \in V$  do
4:      $color[v] \leftarrow \emptyset$ 
5:   while  $unexplored \neq \emptyset$  do
6:      $u \leftarrow$  an item of  $unexplored$ 
7:      $unexplored \leftarrow unexplored \setminus \{u\}$ 
8:      $S \leftarrow \text{EMPTYSTACK}()$ 
9:      $\text{PUSH}(S, u)$ 
10:     $color[u] \leftarrow \top$ 
11:    while  $\neg \text{EMPTY}(S)$  do
12:       $s \leftarrow \text{POP}(S)$ 
13:       $unexplored \leftarrow unexplored \setminus \{s\}$ 
14:       $paint \leftarrow \neg color[s]$ 
15:      for neighbour  $t$  of  $s$  do
16:        if  $color[t] \neq \emptyset$  then
17:          if  $color[t] \neq paint$  then
18:            return  $\perp$ 
19:          else
20:             $color[t] \leftarrow paint$ 
21:             $\text{PUSH}(S, t)$ 
22:  return  $\top$ 
```

Ο χρόνος εκτέλεσης του αλγορίθμου θα είναι $O(|V| + |E|)$. Αυτό προκύπτει από το γεγονός ότι διασχίζουμε όλες τις ακμές του γράφου από το πολύ δύο φορές, μία για κάθε άκρη τους, και άρα έχουμε $O(|E|)$, ενώ για κάθε συνιστώσα θα πρέπει να ξεκινήσουμε από κάποιον κόμβο της το οποίο κοστίζει $O(|V|)$.

Άσκηση 2

Διασχίζουμε το γράφο χρησιμοποιώντας αναζήτηση κατά πλάτος και διατηρούμε κάθε φορά το πλήθος των τρόπων με τους οποίους φτάσαμε σε κάθε κόμβο.

Λήμμα. Ο αλγόριθμος υπολογισμού πλήθους συντομότερων μονοπατιών είναι ορθός.

Απόδειξη. Η ορθότητα του αλγορίθμου προκύπτει ως εξής. Επειδή οι ακμές δεν έχουν βάρη, η μέθοδος αναζήτησης κατά πλάτος θα τις επισκευθεί σε αύξουσα σειρά απόστασης. Αν σε κάποια κορυφή μπορούμε να φτάσουμε με μεγαλύτερη απόσταση απ' ό,τι έχουμε ήδη φτάσει, αυτός ο τρόπος δεν μας ενδιαφέρει διότι δεν θα αποτελεί μέρος του συντομότερου μονοπατιού. Αυτό είναι και η αρχή της βελτιστότητας του προβλήματος. Άρα μπορούμε να αποθηκεύσουμε σε κάθε

κόμβο το πλήθος των τρόπων με τους οποίους φτάσαμε σε αυτόν με ελάχιστο κόστος. Επισκεπτόμενοι τους γείτονές του, στη συνέχεια, ενημερώνουμε όσους δεν έχουμε επισκευθεί, ή έχουμε επισκευθεί με το ίδιο κόστος και προσθέτουμε στο πλήθος των τρόπων με τους οποίους μπορούμε να φτάσουμε σε αυτούς το πλήθος των τρόπων που αντιστοιχούν στον κόμβο από τον οποίο ήρθαμε.

Η αμετάβλητη ιδιότητα της επανάληψης είναι ότι, σε κάθε βήμα i , έχουμε ενημερώσει σωστά το πλήθος των ελάχιστων μονοπατιών με τα οποία μπορούμε να φτάσουμε σε κόμβους που απέχουν ακριβώς i από τον αρχικό. Πράγματι, για $i = 0$, αυτό ισχύει τετριμμένα αφού η απόσταση από το s στο s είναι 0 και το πλήθος των τρόπων είναι 1.

Έστω, τώρα, ότι η αμετάβλητη ιδιότητα ισχύει στο i -οστό βήμα. Τότε, στο $i + 1$ -οστό βήμα, οι κόμβοι με απόσταση $i + 1$ είναι προσβάσιμοι από τους κόμβους με απόσταση i . Οι κόμβοι με μεγαλύτερες αποστάσεις τους οποίους έχουμε ήδη επισκευθεί δεν ενημερώνονται αφού τα μονοπάτια δεν είναι συντομότερα. Οι κόμβοι που επισκεπτόμαστε σε αυτό το βήμα αποκτούν απόσταση $i + 1$ και το πλήθος των επισκέψεων για κάποιο νέο κόμβο που επισκεπτόμαστε περιλαμβάνει το άθροισμα από τα πλήθη που έχουμε βρει για όλους τους κόμβους που οδηγούν σε αυτόν. Έτσι, η αμετάβλητη ιδιότητα θα ισχύει και στο βήμα $i + 1$.

Χρησιμοποιώντας μαθηματική επαγωγή, αυτό ολοκληρώνει την απόδειξη του αμετάβλητου. Τέλος, όταν το i ισούται με το μήκος του συντομότερου μονοπατιού από την αφετηρία στον προορισμό, το πλήθος που έχουμε υπολογίσει είναι η λύση. \square

Algorithm 2 Άσκηση 2

```
1: procedure SHORTESTCOUNT( $V, E, s, t$ )
2:   for  $v \in V$  do
3:      $visited[v] \leftarrow 0$ 
4:      $distance[v] \leftarrow \infty$ 
5:    $visited[s] \leftarrow 1$ 
6:    $distance[s] \leftarrow 0$ 
7:    $q \leftarrow \text{EMPTYQUEUE}()$ 
8:    $\text{ENQUEUE}(q, s)$ 
9:   while  $\neg \text{EMPTY}(q)$  do
10:     $u \leftarrow \text{DEQUEUE}(q)$ 
11:    if  $u = t$  then
12:      return  $visited[t]$ 
13:    for neighbour  $v$  of  $u$  do
14:      if  $distance[v] = \infty$  then
15:         $\text{ENQUEUE}(q, u)$ 
16:      if  $distance[u] + 1 \leq distance[v]$  then
17:         $distance[v] \leftarrow distance[u] + 1$ 
18:         $visited[v] \leftarrow visited[v] + visited[u]$ 
```

Η πολυπλοκότητα του αλγορίθμου είναι η ίδια με αυτή της αναζήτησης κατά πλάτος, δηλαδή $O(|E|)$.

Άσκηση 3

(α)

Ο γράφος της άσκησης 4 με $L = \{C\}$ είναι ένα παράδειγμα γράφου που έχει διαφορετικό ΕΣΔ υπό περιορισμούς.

(β)

Για κάθε κόμβο για τον οποίο υπάρχει απαίτηση να είναι φύλλο κρατάμε την φθηνότερη ακμή που το συνδέει με κάποιον κόμβο που δεν υπάρχει απαίτηση να είναι φύλλο. Στη συνέχεια βρίσκουμε το ΕΣΔ κατά τα γνωστά.

Λήμμα. Ο αλγόριθμος υπολογισμού ΕΣΔ υπό περιορισμούς είναι ορθός.

Απόδειξη. Κάθε φύλλο θα συνδέεται με το υπόλοιπο δέντρο με μία ακμή. Αυτή η ακμή θα πρέπει να είναι η ελάχιστη ανάμεσα σε εκείνες που μπορούν να συνδέσουν το φύλλο με το υπόλοιπο δέντρο. Αυτό προκύπτει ως εξής. Αν η ακμή αυτή δεν ήταν η ελάχιστη, τότε αντικαθιστώντας την με την ελάχιστη θα προέκυπτε ένα φθηνότερο ΕΣΔ υπό τους ίδιους περιορισμούς, πράγμα άτοπο.

Επιπλέον, το ΕΣΔ υπό περιορισμούς θα περιέχει ένα ΕΣΔ του τμήματος του γράφου που δεν είμαστε υποχρεωμένοι να χρησιμοποιήσουμε μόνο ως φύλλα. Αυτό φαίνεται ως εξής. Έστω ότι το τμήμα αυτό δεν ήταν ΕΣΔ. Τότε θα ήταν ένα συνδετικό δέντρο μη ελάχιστο. Αντικαθιστώντας το με κάποιο ΕΣΔ θα προέκυπτε ένα συνολικά φθηνότερο ΕΣΔ υπό περιορισμούς, πράγμα άτοπο.

Έτσι, το ΕΣΔ υπό περιορισμούς μπορεί να προκύψει υπολογίζοντας το απλό ΕΣΔ αφού αφαιρέσουμε τις ακμές που γνωρίζουμε ότι πρέπει να απαγορευθούν, δηλαδή τις ακμές που συνδέουν φύλλα και δεν είναι ελάχιστες. \square

Algorithm 3 Άσκηση 3

```

1: procedure CONSTRAINEDMST( $V, E, L$ )
2:   if  $|V| \leq 2$  then
3:     return  $E$ 
4:   for  $v \in V$  do
5:      $adj[v] \leftarrow \emptyset$ 
6:      $best[v] \leftarrow \infty$ 
7:   for  $e = (u, v, w) \in E$  do
8:     if  $u \notin L \vee v \notin L$  then
9:       if  $u \in L \vee v \in L$  then
10:        for  $s \in \{u, v\}$  do
11:          if  $s \in L$  then
12:            if  $w < best[s]$  then
13:               $best[s] \leftarrow w$ 
14:               $adj[s] \leftarrow \{e\}$ 
15:          else
16:             $adj[v] \leftarrow adj[v] \cup \{e\}$ 
17:   return PRIM( $V, adj$ )

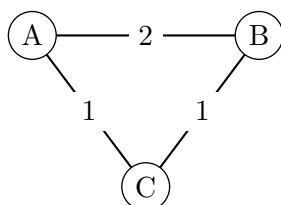
```

Η πολυπλοκότητα του αλγορίθμου προκύπτει ως εξής. Για κάθε ακμή ενημερώνουμε την βέλτιστη ακμή που είναι γνωστή για κάποιο φύλλο, αν συνδέεται με κάποιο, σε σταθερό χρόνο. Ο συνολικός χρόνος γι' αυτό απαιτεί να περάσουμε από όλες τις ακμές σε συνολικό χρόνο $O(|E|)$. Ο αλγόριθμος του Prim τρέχει σε $O(|E| + |V| \log |V|)$ που είναι και άνω φράγμα για όλο τον αλγόριθμο.

Άσκηση 4

(α)

Ο ακόλουθος γράφος έχει μοναδικό ΕΣΔ, αλλά περιέχει ακμές ίδιου βάρους:



(β)

Ο γράφος του υποερωτήματος (α) έχει μοναδικό ΕΣΔ αλλά περιέχει ακμές ίδιου βάρους που είναι οι ελάχιστες που διασχίζουν το ίδιο κόψιμο.

Λήμμα. Κάθε ακμή ενός ΕΣΔ ενός συνεκτικού μη κατευθυνόμενου ζυγισμένου γράφου είναι η ελάχιστη που διασχίζει κάποιο κόψιμο.

Απόδειξη. Έστω τέτοιος γράφος $G = (V, E, w)$ με ένα ΕΣΔ του $T \subseteq E$ και έστω ότι υπάρχει κάποια ακμή $e = (u, v) \in T$ τέτοια ώστε να μην υπάρχει κόψιμο του Γ το οποίο να είναι η ελάχιστη που το διασχίζει. Τότε αφαιρώντας αυτή την τομή από το T προκύπτει ένα κόψιμο S του αρχικού γράφου σε δύο συνεκτικές συνιστώσες που είναι δέντρα. Το κόψιμο S θα διασχίζεται από κάποια ακμή f του G που θα είναι ελάχιστη ακμή που διασχίζει το S . Επειδή η e δεν είναι η ελάχιστη για κανένα κόψιμο, θα είναι $e \neq f$ και $w(f) < w(e)$. Όμως στο T η ακμή e μπορεί να αντικατασταθεί από την f και το δέντρο να παραμείνει συνεκτικό και με βάρος $w(T \cup \{f\} \setminus \{e\}) < w(T)$. Άρα το T δεν ήταν ΕΣΔ, κάτι το οποίο αποτελεί αντίφαση. \square

Λήμμα. Κάθε συνεκτικός μη κατευθυνόμενος ζυγισμένος γράφος στον οποίο για κάθε κόψιμο η ακμή ελάχιστου βάρους που το διασχίζει είναι μοναδική έχει μοναδικό ΕΣΔ.

Απόδειξη. Έστω τέτοιος γράφος $G = (V, E, w)$ και δύο ΕΣΔ του, $T \subseteq E$ και $T' \subseteq E$, με $T \neq T'$ και $w(T) = w(T')$.

Τότε, επειδή $T \neq T'$, θα υπάρχει κάποια ακμή $e = (u, v)$ με $e \in T \wedge e \notin T'$. Από το παραπάνω λήμμα, η e θα είναι η ελάχιστη που διασχίζει κάποιο κόψιμο του T , έστω S .

Τώρα, επειδή το T' είναι συνεκτικό δέντρο, θα υπάρχει μοναδικό μονοπάτι από τον κόμβο u στον κόμβο v που διασχίζει το S με κάποια ακμή, έστω $e' \neq e$. Αφού $w(e) < w(e')$, αντικαθιστώντας την ακμή e' με την e στο T' παίρνουμε ένα συνεκτικό δέντρο με βάρος $w(T' \cup \{e\} \setminus \{e'\}) < w(T')$. Άρα το T' δεν είναι ελάχιστο συνεκτικό δέντρο, κάτι το οποίο αποτελεί αντίφαση.

Άρα το ΕΣΔ είναι μοναδικό. \square

(γ)

Λήμμα. Το ΕΣΔ ενός συνεκτικού ζυγισμένου μη κατευθυνόμενου γράφου είναι μοναδικό ανν κάθε ακμή του είναι η μοναδική ελάχιστη σε κάποιο κόψιμο του γράφου.

Απόδειξη. Έστω $G = (V, E, w)$ ένας τέτοιος γράφος και έστω K ένα ΕΣΔ του.

Θα δείξουμε ότι αν κάθε ακμή του K είναι η μοναδική ελάχιστη σε κάποιο κόψιμο, τότε το δέντρο είναι μοναδικό. Πράγματι, έστω ότι υπάρχει ΕΣΔ $T \neq K$. Τότε θα υπάρχει ακμή $e = (u, v) \in K \wedge e \notin T$. Από την υπόθεση, υπάρχει κόψιμο S το οποίο η e διασχίζει ως μοναδική ελάχιστη. Επειδή το T είναι συνδετικό, θα υπάρχει μοναδικό μονοπάτι που συνδέει τους κόμβους u και v το οποίο διασχίζει το κόψιμο S χρησιμοποιώντας κάποια ακμή $f \neq e$. Επειδή η e είναι ελάχιστη, θα είναι $w(e) < w(f)$. Αντικαθιστώντας την f με την e στο T προκύπτει ΕΣΔ με $w(T \cup \{e\} \setminus \{f\}) < w(T)$ κάτι το οποίο αποτελεί αντίφαση. Άρα το K είναι μοναδικό.

Αντίστροφα, έστω ότι το K είναι μοναδικό. Έστω, τώρα, ότι υπάρχει κάποια ακμή $e \in K$ που δεν είναι η μοναδική ελάχιστη σε κανένα κόψιμο του G . Τότε έστω το κόψιμο S που δημιουργείται αν αφαιρέσουμε την e από το K . Από το γεγονός ότι το K είναι ελάχιστο (βλ. προηγούμενο θεώρημα), η e θα είναι η ελάχιστη στο S . Έστω, τώρα, ότι υπάρχει και άλλη ελάχιστη ακμή $f \neq e$ που διασχίζει το S με $w(f) = w(e)$. Αντικαθιστώντας την e με την f στο K παίρνουμε ένα συνδετικό δέντρο με βάρος $w(T \cup \{f\} \setminus \{e\}) = w(T)$. Άρα το K δεν ήταν μοναδικό συνδετικό δέντρο. Αυτό αποτελεί αντίφαση, και άρα όλες οι ακμές είναι ελάχιστες σε κάποιο κόψιμο. \square

(δ)

Ξεκινάμε υπολογίζοντας ένα ΕΣΔ του γράφου. Στη συνέχεια ελέγχουμε αν υπάρχει εναλλακτικό ΕΣΔ εργαζόμενοι ως εξής. Κατασκευάζουμε έναν πίνακα $V \times V$ που δείχνει τη μέγιστη ακμή του μονοπατιού ανάμεσα σε οποιουδήποτε δύο κόμβους του ΕΣΔ. Τέλος, ελέγχουμε για κάθε ακμή του αρχικού γράφου (u, v) αν μπορεί να αντικαταστήσει κάποια ακμή του ΕΣΔ και αυτό να παραμείνει ελάχιστο. Αυτό γίνεται πλέον εύκολα ελέγχοντας την τιμή της μέγιστης ακμής στο μονοπάτι από το u στο v πάνω στον αρχικό γράφο.

Η ορθότητα του αλγορίθμου προκύπτει άμεσα από το παρακάτω λήμμα:

Λήμμα. Ένα ΕΣΔ T ενός συνδεδεμένου μη κατευθυνόμενου ζυγισμένου γράφου $G = (V, E, w)$ είναι μοναδικό ανν $\forall (u, v) \in E \setminus T : w(u, v) > w(\maxEdge(p(u, v)))$ όπου $p(u, v)$ το μονοπάτι από την κορυφή u στην κορυφή v στο δέντρο T .

Απόδειξη. Θα αποδείξουμε την ισοδυναμία με δύο αντίστροφες μεταξύ τους αποδείξεις με απαγωγή σε άτοπο.

Έστω ότι το T δεν είναι μοναδικό, τότε θα υπάρχει $T' \neq T$ που διαφέρει κατά κάποια ακμή $e = (u, v)$ με $e \notin T \wedge e \in T'$. Τότε έστω το κόψιμο S που προκύπτει αν αφαιρέσουμε την e από το T' . Αυτό το κόψιμο θα διασχίζει μια ακμή $f \neq e$ του T που είναι μέρος του μοναδικού μονοπατιού που οδηγεί από την u στη v στο T αφού το T είναι ΕΣΔ. Οι e και f θα έχουν το ίδιο βάρος (βλ. παραπάνω θεώρημα) δηλαδή $w(e) = w(f)$ διότι διαφορετικά τα T και T' δεν θα ήταν ελάχιστα. Άρα η e δεν μπορεί να ξεπερνά το $w(\maxEdge(p(u, v)))$. Αυτό είναι αντίφαση.

Αντίστροφα, έστω ότι υπάρχει ακμή $e = (u, v) \in E \setminus T$ τέτοια ώστε $w(e) \leq w(\maxEdge(p(u, v)))$ και έστω $f = \maxEdge(p(u, v))$. Προφανώς θα είναι $e \neq f$. Τότε το $T \cup \{e\} \setminus \{f\}$ θα είναι συνδετικό δέντρο. Αν $w(e) < w(f)$ τότε το νέο δέντρο έχει μικρότερο βάρος από το T , πράγμα αδύνατο. Άρα $w(e) = w(f)$. Αντικαθιστώντας την f με την e στο T προκύπτει ένα ΕΣΔ διαφορετικό από το αρχικό. Συνεπώς το T δεν είναι μοναδικό. Αυτό είναι αντίφαση.

Τα δύο σκέλη αυτά μαζί ολοκληρώνουν την απόδειξη ισοδυναμίας. \square

Algorithm 4 Άσκηση 4

```

1: procedure UNIQUEMST( $V, E, w$ )
2:    $T \leftarrow \text{PRIM}(V, E, w)$ 
3:   for  $(u, v) \in V^2$  do
4:      $\text{dist}[u][v] \leftarrow \infty$ 
5:   for  $v \in V$  do
6:      $S \leftarrow \text{EMPTYSTACK}()$ 
7:      $\text{PUSH}(S, v)$ 
8:     while  $\neg \text{EMPTY}(S)$  do
9:        $u \leftarrow \text{POP}(S)$ 
10:      for neighbour  $r$  of  $u$  in  $T$  do
11:        if  $\text{dist}[v][u] = \infty$  then
12:           $\text{dist}[v][u] \leftarrow \max(\text{dist}[v][r], w(r, u))$ 
13:           $\text{PUSH}(S, u)$ 
14:   for  $(u, v) \in E \setminus T$  do
15:     if  $w(u, v) = \text{dist}[u][v]$  then
16:       return  $\perp$ 
17:   return  $\top$ 

```

Ο αλγόριθμος του Prim τρέχει σε $O(|V|^2)$. Η κατασκευή του πίνακα μέγιστων ακμών ανά μονοπάτι μπορεί να γίνει σε χρόνο $O(|V|^2)$ χρησιμοποιώντας αναζήτηση κατά πλάτος πάνω στο δέντρο. Η αναζήτηση ακμής αντικατάστασης γίνεται στη συνέχεια σε $O(|E|)$. Έτσι ο αλγόριθμος που προκύπτει είναι $O(|V|^2)$.

Άσκηση 5

(α)

Λήμμα. Σε ένα συνεκτικό κατευθυνόμενο ζυγισμένο γράφο, για κάθε κύκλο υπάρχει ΕΣΔ που δεν περιέχει την ακμή μέγιστου βάρους του κύκλου αυτού.

Απόδειξη. Έστω τέτοιος γράφος $G = (V, E, w)$ με κάποιον κύκλο C και έστω ένα ΕΣΔ $T \subseteq E$ που περιέχει την ακμή e μέγιστου βάρους του C . Τότε έστω το δάσος $T \setminus \{e\}$ που θα αποτελείται από δύο συνδεδεμένους υπογράφους. Θα υπάρχει μία ακμή $f \in E$ που θα συνδέει αυτούς τους δύο συνδεδεμένους υπογράφους με $f \neq e$ και άρα $w(f) \leq w(e)$. Αντικαθιστώντας την e με την f στο T προκύπτει ένα συνεκτικό δέντρο με βάρος $w(T \cup \{f\} \setminus \{e\}) \leq w(T)$. Άρα το $T \cup \{f\} \setminus \{e\}$ είναι ΕΣΔ. \square

(β)

Λήμμα. Ο αντίστροφος αλγόριθμος του Kruskal είναι ορθός.

Απόδειξη. Σε κάθε βήμα i , ο αλγόριθμος διατηρεί τον ελάχιστου κόστους συνδετικό υπογράφο G_i με ακριβώς $j(i) = m - i + 1$ ακμές. Πράγματι, στο 1ο βήμα, ο υπογράφος είναι ίσος με τον αρχικό γράφο και άρα συνδετικός και ελάχιστος. Έστω ότι ο υπογράφος είναι συνδετικός και ελάχιστος στο i -οστό βήμα. Τότε έστω $e = (u, v)$ η επόμενη ακμή που θα επιλεγεί.

Στην περίπτωση που η e είναι γέφυρα του G_i ανάμεσα σε δύο συνδεδεμένα τμήματα C_1 και C_2 , δηλαδή διασχίζει το κόψιμο $S = (C_1, C_2)$ του αρχικού γράφου, η ακμή αυτή δε θα λείπει από κανέναν ελάχιστο συνδετικό υπογράφο G_k με $k > i$. Αυτό φαίνεται ως εξής. Έστω ότι η e έλειπε από τον G_k , τότε αυτός θα περιείχε κάποια ακμή $f \neq e$ που διασχίζει το S για να είναι συνδεδεμένος. Όμως θα είναι $w(f) > w(e)$ διότι, λόγω άπληστης ιδιότητας, η f είχε αφαιρεθεί πριν το i -οστό βήμα αφού η e ήταν γέφυρα του G_i . Αντικαθιστώντας την f με την e λαμβάνουμε ένα συνδετικό υπογράφο ίδιου πλήθους ακμών με βάρος $w(G_k \cup \{e\} \setminus \{f\}) < w(G_k)$. Άρα το G_k δεν θα ήταν ελάχιστο χωρίς την ύπαρξη της e . Συνεπώς μπορούμε με ασφάλεια να διατηρήσουμε την e στον υπογράφο και να προχωρήσουμε στην επόμενη ακμή στο ίδιο βήμα.

Στην περίπτωση που η e δεν είναι γέφυρα του G_i , τότε ανήκει σε ένα κύκλο του G_i . Εργαζόμενοι παρόμοια όπως στο παραπάνω θεώρημα, είναι εμφανές ότι η e δεν ανήκει στον υπογράφο G_{i+1} . Την αφαιρούμε, λοιπόν, και προχωράμε στο επόμενο βήμα.

Χρησιμοποιώντας, τώρα, τη μαθηματική επαγωγή, έχουμε αποδείξει την αμετάβλητη ιδιότητα της επανάληψης. Τέλος, ο ελάχιστος συνδετικός υπογράφος με ακριβώς $n - 1$ ακμές είναι ΕΣΔ. \square

(γ)

Algorithm 5 Άσκηση 5

```
1: procedure REVERSEKRUSKAL( $V, E, w$ )  
2:    $E \leftarrow$  sort  $E$  decreasingly by weight  
3:    $S \leftarrow E$   
4:   for  $e \in E$  do  
5:     if  $\neg \text{BRIDGE}(S, e)$  then  
6:        $S \leftarrow S \setminus \{e\}$   
7:   return  $S$ 
```

Η υλοποίηση του $\text{BRIDGE}(S, e)$ ελέγχει αν η e είναι γέφυρα του S και μπορεί να γίνει με διάφορους τρόπους. Ο απλούστερος τρόπος θα ήταν ένας αλγόριθμος αναζήτησης κατά βάθος ή κατά πλάτος που ελέγχει αν υπάρχει και άλλο μονοπάτι ανάμεσα στις δύο κορυφές που συνδέει η ακμή. Τότε η συνολική πολυπλοκότητα του αλγορίθμου είναι $O(E^2)$. Αντικαθιστώντας τον έλεγχο για γέφυρες με κάποια πιο αποδοτική υλοποίηση (βλ. [1]) μπορούμε να πετύχουμε την αισθητά καλύτερη πολυπλοκότητα $O(E \log E (\log \log E)^3)$.

Βιβλιογραφία

- [1] Mikkel Thorup, *Near-optimal fully-dynamic graph connectivity*. Addison Wesley, Massachusetts, Proc. 32nd ACM Symposium on Theory of Computing, 2000.