

02_ΜΗ-ΚΑΤΕΥΘΥΝΟΜΕΝΑ ΓΡΑΦΗΜΑΤΑ / 02_graphreprtransv

Διάσχιση κατά Βάθος (Depth-First Search)

Ιδέα

Ο αλγόριθμος βασίζεται στην παρακάτω αναδρομική διαδικασία η οποία δέχεται σαν είσοδο μία κορυφή v και επισκέπτεται αναδρομικά το αριστερό και μετά το δεξιό υποδένδρο. Σε κάθε φάση ο μονοδ. πίνακας $visited$ έχει στη θέση v τιμή $true$ αν η κορυφή v έχει 'άνακαλυφθεί.'

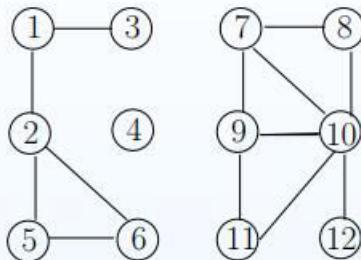
Αλγόριθμος

```
explore( $v$ ,  $visited$ )
   $visited[v] \leftarrow true;$ 
  for  $u \in N(v)$ 
    if not  $visited[u]$  then  $\text{explore}(u, visited);$ 
```

Η διαδικασία αυτή καλείται από το "κύριο" πρόγραμμα:

```
main()
  for  $v \in V$ 
     $visited[v] \leftarrow false;$ 
    for  $v \in V$ 
      if not  $visited[v]$  then  $\text{explore}(v, visited);$ 
```

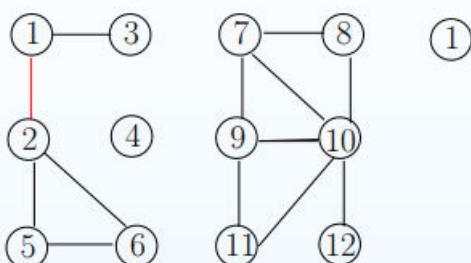
Παράδειγμα



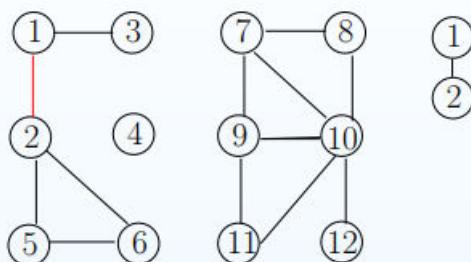
Σχήμα 3: Γράφημα

Θα εκτελέσουμε ΔκΒ στο παραπάνω γράφημα θεωρώντας ότι οι κόμβοι στη λίστα γειτνίασης εμφανίζονται με λεξικογραφική σειρά.

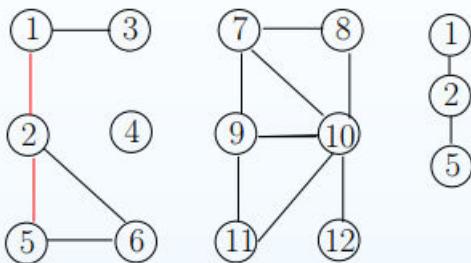
Στα παρακάτω σχήματα απεικονίζεται το χτίσιμο του γεννητορικού δάσους - οι ακμές οι οποίες οδηγούν σε κορυφές που έχει ο αλγόριθμος ήδη επισκεφτεί εμφανίζονται διακεκομμένες.



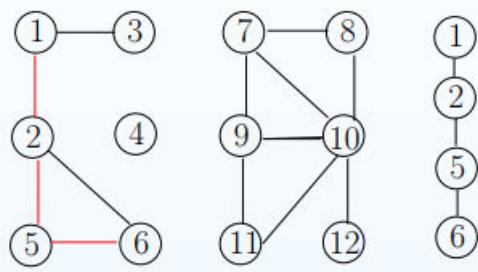
Σχήμα 4: Γράφημα - ΔκΒ



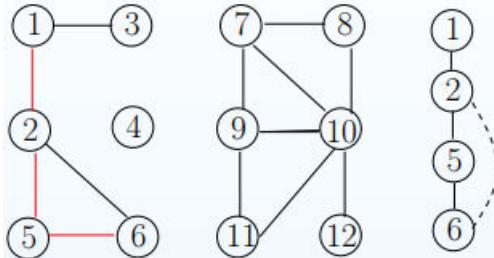
Σχήμα 4: Γράφημα - ΔκΒ



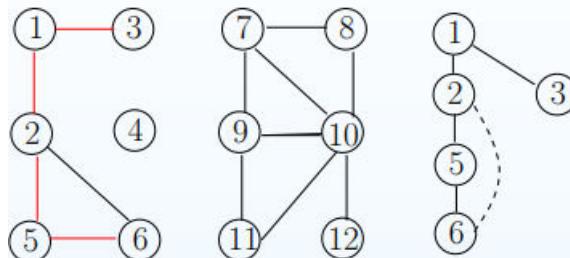
Σχήμα 4: Γράφημα - ΔκΒ



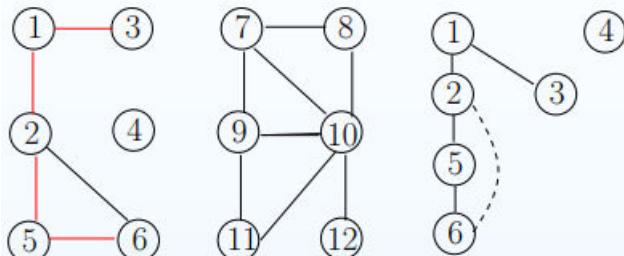
Σχήμα 4: Γράφημα - ΔκΒ



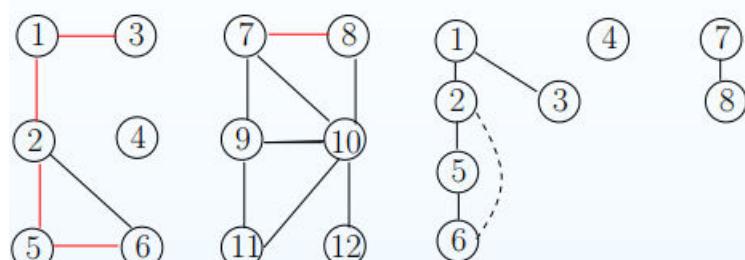
Σχήμα 4: Γράφημα - ΔκΒ



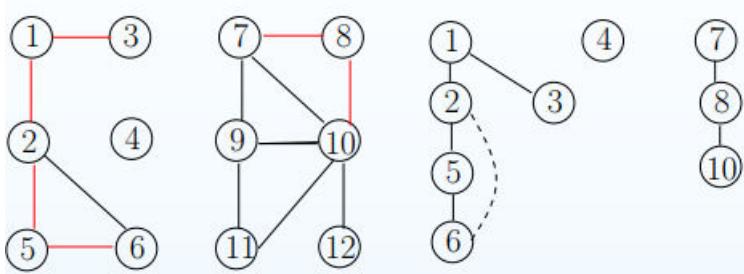
Σχήμα 4: Γράφημα - ΔκΒ



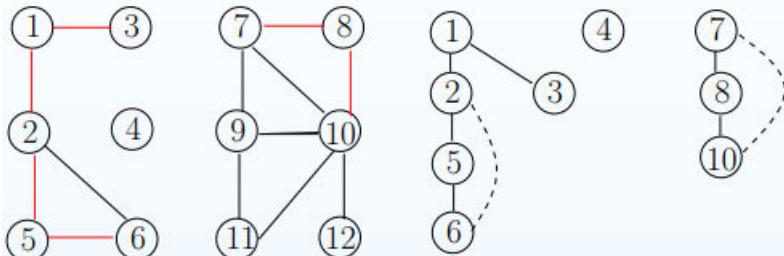
Σχήμα 4: Γράφημα - ΔκΒ



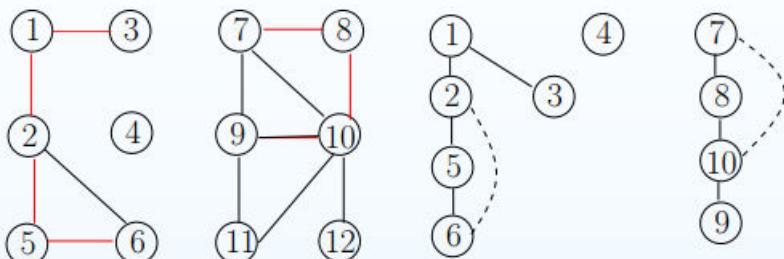
Σχήμα 4: Γράφημα - ΔκΒ



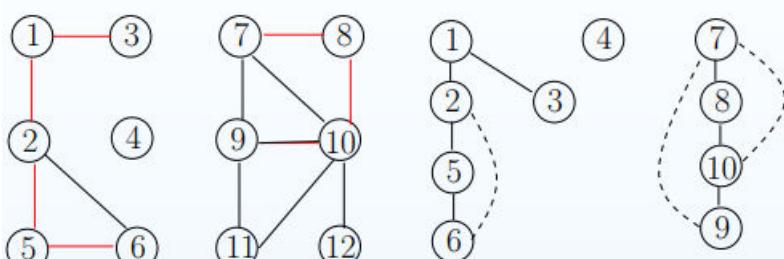
Σχήμα 4: Γράφημα - ΔκΒ



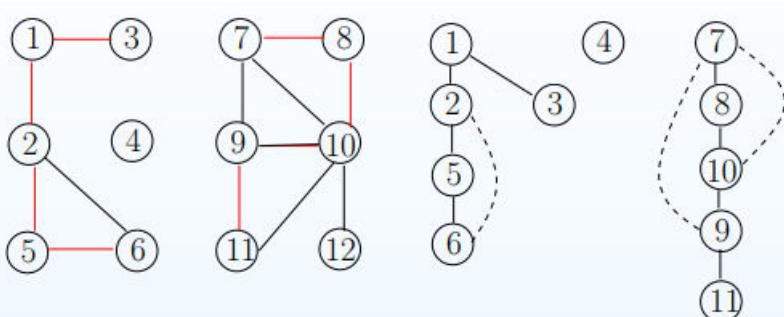
Σχήμα 4: Γράφημα - ΔκΒ



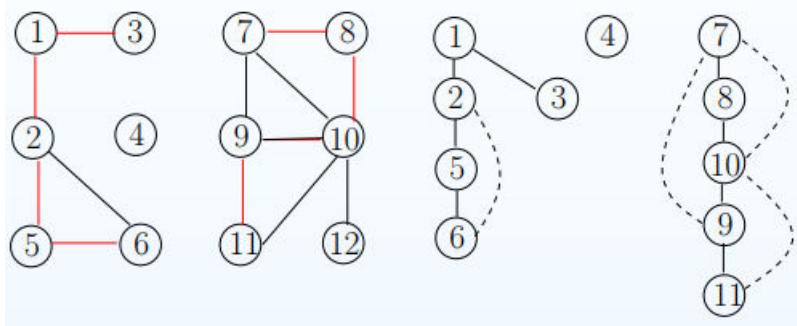
Σχήμα 4: Γράφημα - ΔκΒ



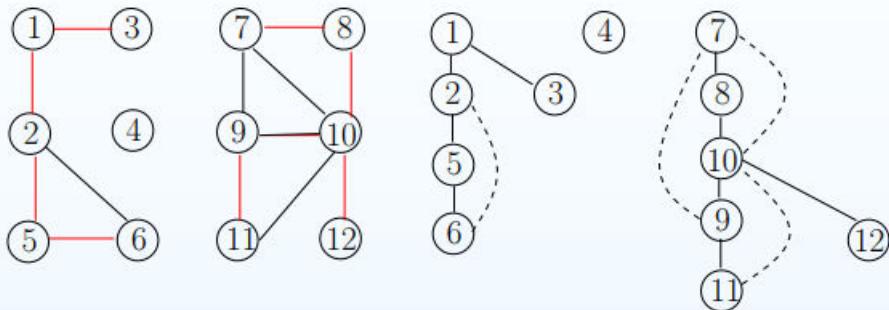
Σχήμα 4: Γράφημα - ΔκΒ



Σχήμα 4: Γράφημα - ΔκΒ

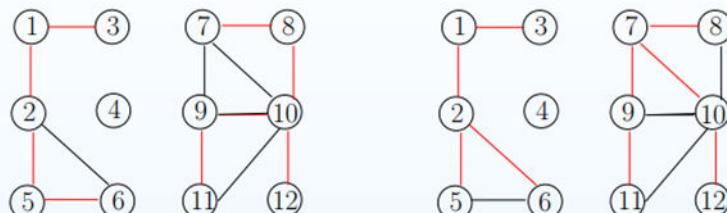


Σχήμα 4: Γράφημα - ΔκΒ



Σχήμα 4: Γράφημα - ΔκΒ

Παρατηρήσεις



Σχήμα 7: Γεννητορικά δένδρα ΔκΒ και ΔκΠ (κόκκινες ακμές)

Στο Σχήμα 7 απεικονίζονται τα δένδρα που σχημάτισαν οι δυο στρατηγικές διάσχισης. Από κατασκευής το δένδρο της ΔκΠ έχει την ιδιότητα ότι κάθε μονοπάτι που συνδέει τη ρίζα $υ$ με ένα φύλλο $ω$ είναι το συντομότερο ως προς τον αριθμό των ακμών. Για παράδειγμα το μονοπάτι ανάμεσα στις κορυφές 7, 11 με το μικρότερο αριθμό ακμών είναι το 7-9-11. Το αντίστοιχο μονοπάτι που υπολόγισε η ΔκΒ είναι το 7-8-10-9-11.

Πολυπλοκότητα

Αμφότερες οι στρατηγικές διάσχισης επισκέπτονται τις κορυφές του γραφήματος δια μέσου των ακμών.

Παρατηρούμε ότι κάθε φορά που ‘άνακαλύπτεται’ μία ακμή εκτελείται σταθερός αριθμών στοιχειωδών πράξεων. Επίσης η κάθε ακμή ‘άνακαλύπτεται’ δύο φορές (μία φορά για κάθε προσπίπουσα κορυφή). Άρα μπορούμε να θεωρήσουμε ότι για κάθε ακμή εκτελούνται το πολύ c στοιχειώδεις πράξεις. Άρα αν $T(m)$ είναι ο αριθμός των στοιχειωδών πράξεων της διάσχισης ενός γραφήματος με m ακμές, έχουμε

$$T(m) \leq B(m), \text{όπου } B(m) = B(m - 1) + c.$$

Σε κλειστή μορφή έχουμε $B(m) = c \cdot m$ και άρα

$$T(m) = O(n + m)$$

εφόσον στην $T(m)$ συνυπολογίσουμε και το κόστος αρχικοποίησης των δομών που χρησιμοποιεί η διάσχιση.

ΔκΒ με στοίβα (DFS with stack)

Ιδέα

Είναι προφανές ότι η ΔκΒ στην πραγματικότητα υλοποιεί έναν μηχανισμό στοίβας: όταν επισκέπτεται έναν κόμβο αποθηκεύει τους γείτονες σε μία ουρά Q με ιεραρχία LIFO. Μπορούμε να εξαλείψουμε την αναδρομή χρησιμοποιώντας την Q . Η διαδικασία explore γίνεται:

Αλγόριθμος

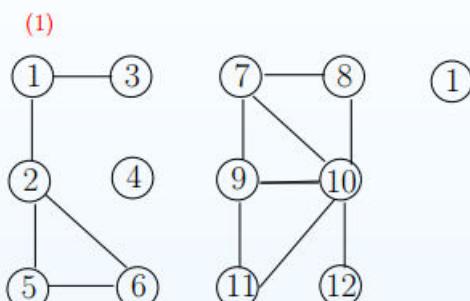
```
explore(clock, Q, visited, preorder)
  while Q ≠ ∅
    v ← pop(Q);
    if not visited[v] then
      visited[v] ← true;
      preorder[v] ← ++clock;
      for (u ∈ N(v)) and (not visited[u]) push(u, Q);
```

Η μεταβλητή *clock* χρησιμοποιείται για να μπορούμε να καταγράψουμε τη σειρά επίσκεψης. Η καταγραφή γίνεται στον μονοδιάστατο πίνακα *preorder*.

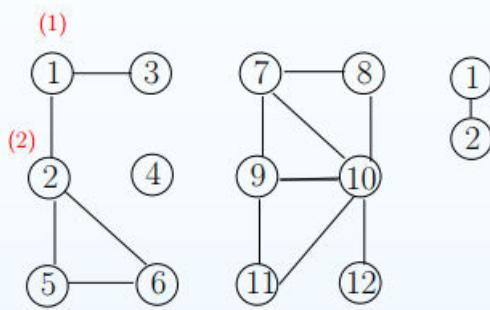
Το “κύριο” πρόγραμμα γίνεται:

```
main()
  create Q;
  clock ← 0;
  for v ∈ V
    visited[v] ← false; preorder[v] ← 0;
  for v ∈ V
    if not visited[v] then
      push(v, Q);
      explore(clock, Q, visited, preorder);
```

Παράδειγμα

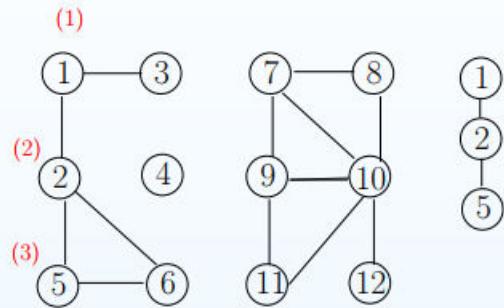


Σχήμα 5: Γράφημα - ΔκΒ



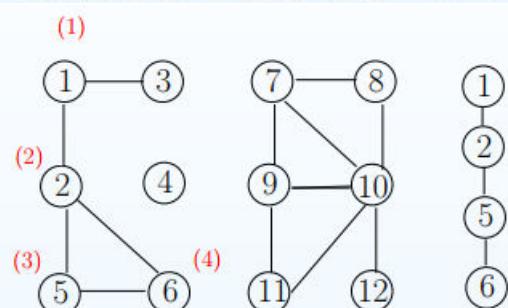
$$Q : 5 - 6 - 3$$

Σχήμα 5: Γράφημα - ΔκΒ



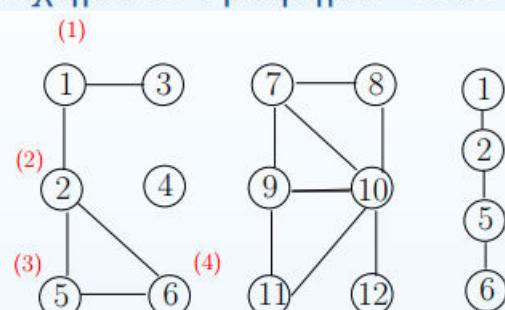
$$Q : 6 - 6 - 3$$

Σχήμα 5: Γράφημα - ΔκΒ



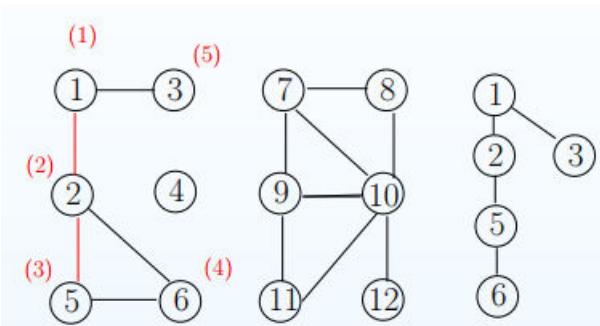
$$Q : 6 - 3$$

Σχήμα 5: Γράφημα - ΔκΒ



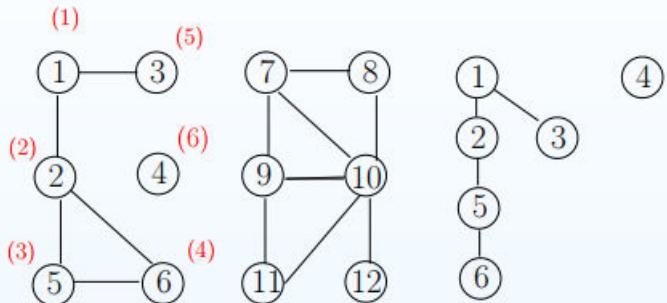
$$Q : 3$$

Σχήμα 5: Γράφημα - ΔκΒ



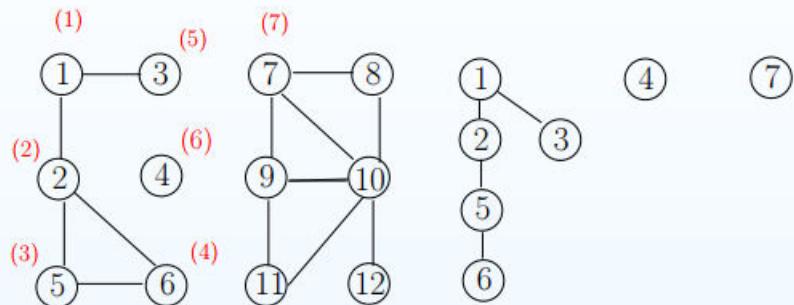
$Q :$

Σχήμα 5: Γράφημα - ΔκΒ



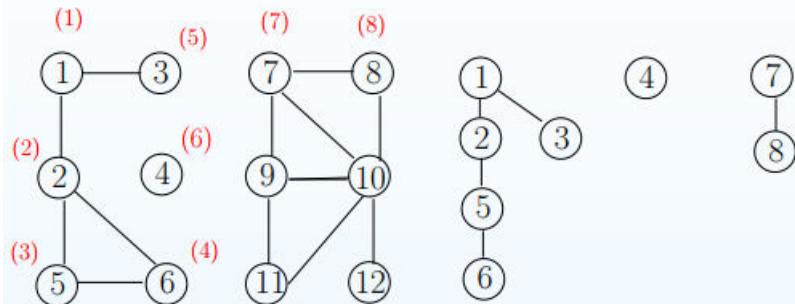
$Q :$

Σχήμα 5: Γράφημα - ΔκΒ



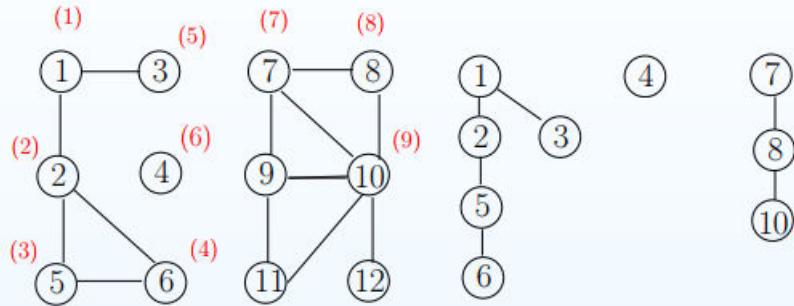
$Q : 8 - 9 - 10$

Σχήμα 5: Γράφημα - ΔκΒ



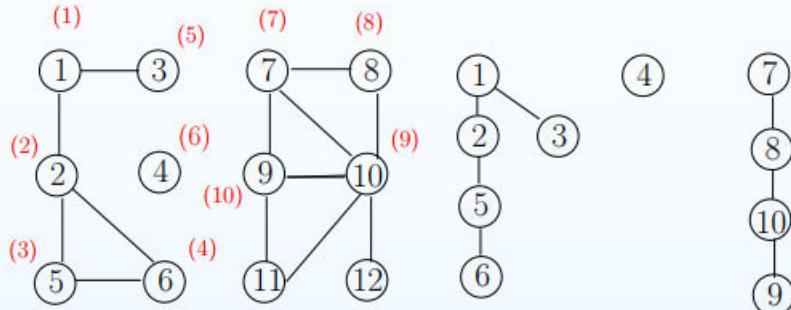
$Q : 10 - 9 - 10$

Σχήμα 5: Γράφημα - ΔκΒ



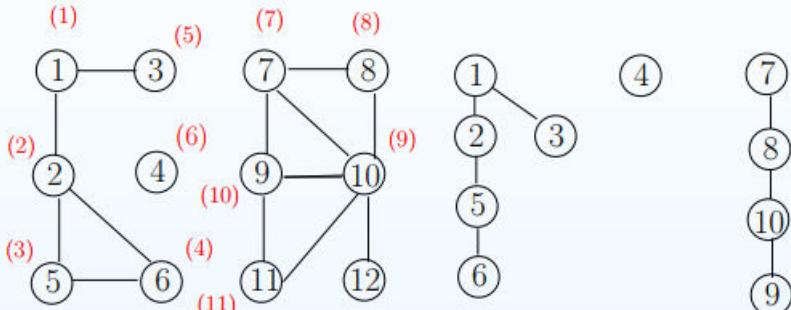
$$Q : 9 - 11 - 12 - 9 - 10$$

Σχήμα 5: Γράφημα - ΔκΒ



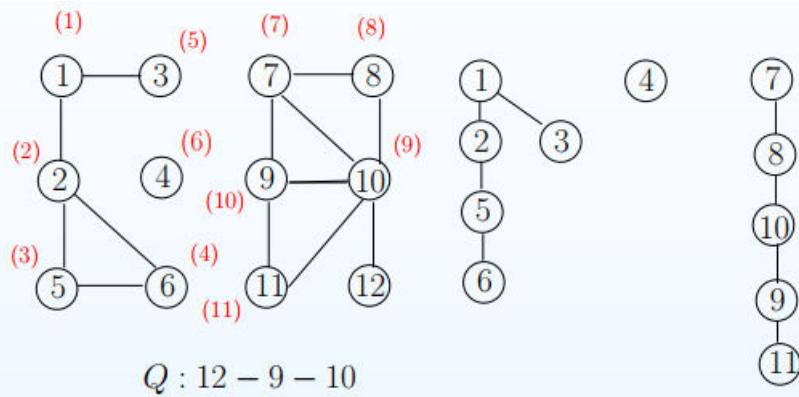
$$Q : 11 - 11 - 12 - 9 - 10$$

Σχήμα 5: Γράφημα - ΔκΒ

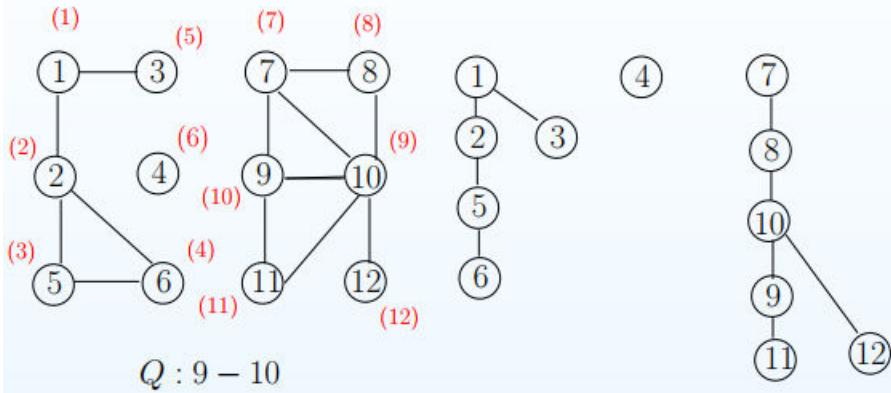


$$Q : 11 - 12 - 9 - 10$$

Σχήμα 5: Γράφημα - ΔκΒ



Σχήμα 5: Γράφημα - ΔκΒ



Σχήμα 5: Γράφημα - ΔκΒ

Πολυπλοκότητα

Αμφότερες οι στρατηγικές διάσχισης επισκέπτονται τις κορυφές του γραφήματος δια μέσου των ακμών.

Παρατηρούμε ότι κάθε φορά που ‘άνακαλύπτεται’ μία ακμή εκτελείται σταθερός αριθμόν στοιχειωδών πράξεων. Επίσης η κάθε ακμή ‘άνακαλύπτεται’ δύο φορές (μία φορά για κάθε προσπίπουσα κορυφή). Άρα μπορούμε να θεωρήσουμε ότι για κάθε ακμή εκτελούνται το πολύ c στοιχειώδεις πράξεις. Άρα αν $T(m)$ είναι ο αριθμός των στοιχειωδών πράξεων της διάσχισης ενός γραφήματος με m ακμές, έχουμε

$$T(m) \leq B(m), \text{όπου } B(m) = B(m - 1) + c.$$

Σε κλειστή μορφή έχουμε $B(m) = c \cdot m$ και άρα

$$T(m) = O(n + m)$$

εφόσον στην $T(m)$ συνυπολογίζουμε και το κόστος αρχικοποίησης των δομών που χρησιμοποιεί η διάσχιση.

Διάσχιση κατά Πλάτος (Breadth-First Search)

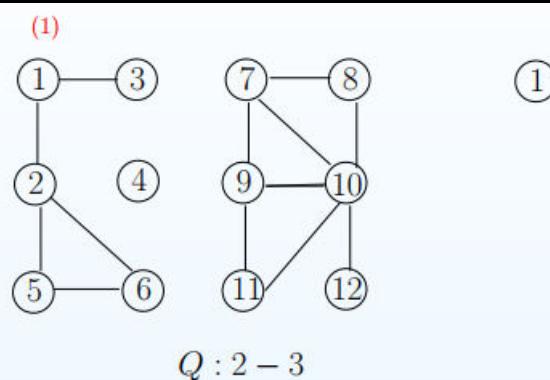
Ιδέα

Η Διάσχιση κατά Πλάτος (ΔκΠ) πρώτα επισκέπτεται όλους τους κόμβους που βρίσκονται στο ίδιο επίπεδο και μετά προχωρά στους επόμενους κόμβους. Προκειμένου να υλοποιηθεί αυτή η στρατηγική η μόνη αλλαγή που χρειάζεται να γίνει είναι στην επεξεργασία της ουράς Q . Αντί για LIFO εφαρμόζουμε FIFO. Έτσι όταν προσθέτουμε τους γείτονες ενός κόμβου τους βάζουμε στο τέλος της Q (και όχι στην αρχή.) Η διαδικασία εξερεύνησης γίνεται:

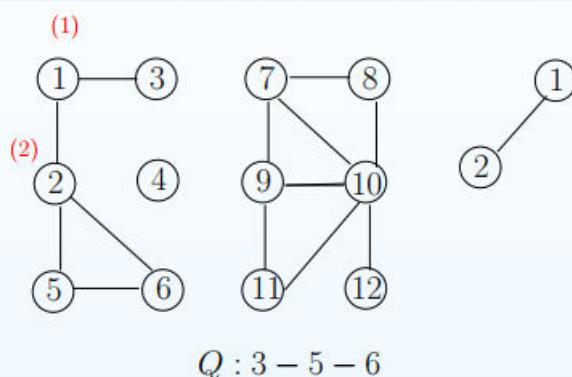
Αλγόριθμος

```
explore(clock, Q, visited, preorder)
  while  $Q \neq \emptyset$ 
     $v \leftarrow \text{pop}(Q)$ ;
    if not  $\text{visited}[v]$  then
       $\text{visited}[v] \leftarrow \text{true}$ ;
       $\text{preorder}[v] \leftarrow ++\text{clock}$ ;
      for ( $u \in N(v)$ ) and (not  $\text{visited}[u]$ ) append( $u, Q$ );
```

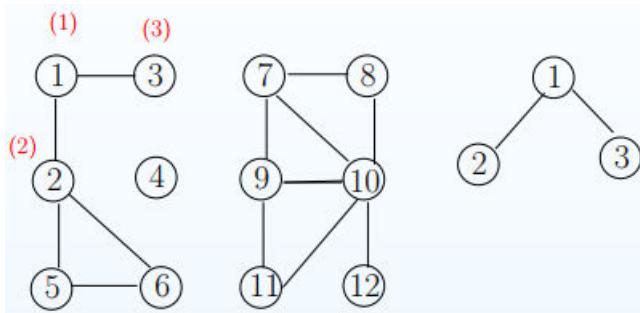
Παράδειγμα



Σχήμα 6: Γράφημα - ΔκΠ

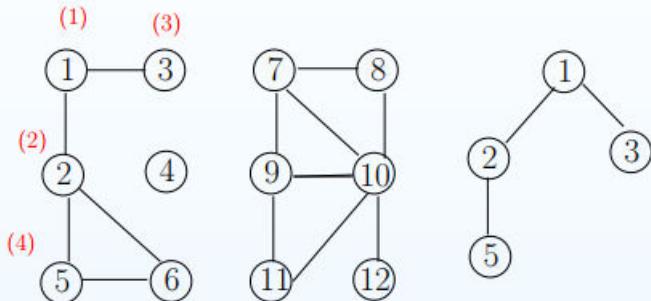


Σχήμα 6: Γράφημα - ΔκΠ



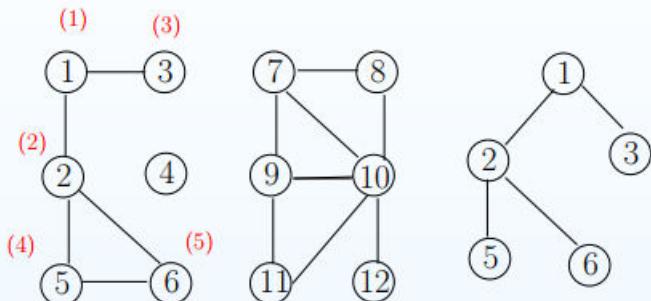
$Q : 5 - 6$

Σχήμα 6: Γράφημα - ΔκΠ



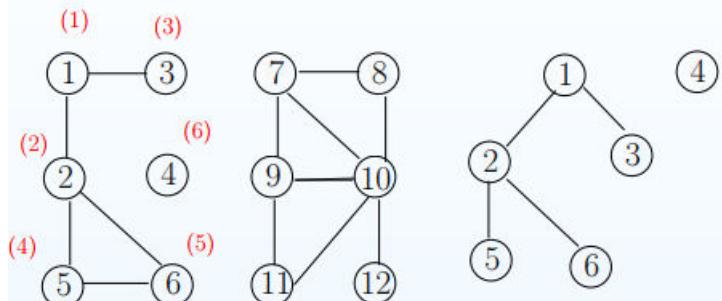
$Q : 6 - 6$

Σχήμα 6: Γράφημα - ΔκΠ



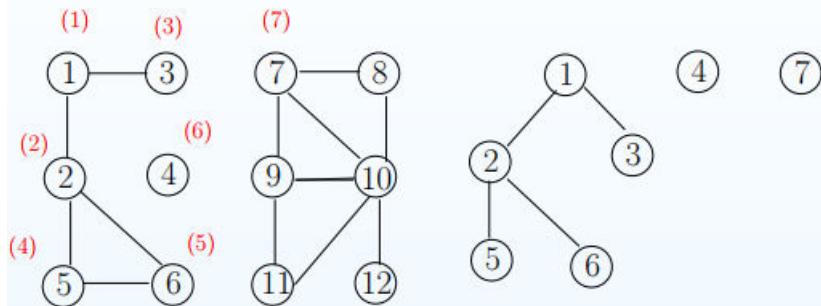
$Q : 6$

Σχήμα 6: Γράφημα - ΔκΠ



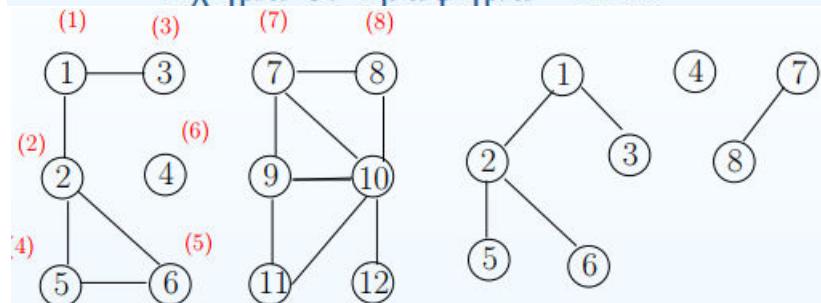
$Q :$

Σχήμα 6: Γράφημα - ΔκΠ



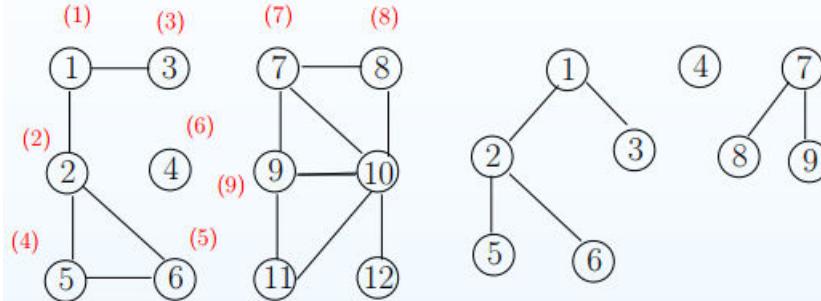
$$Q : 8 - 9 - 10$$

Σχήμα 6: Γράφημα - ΔκΠ



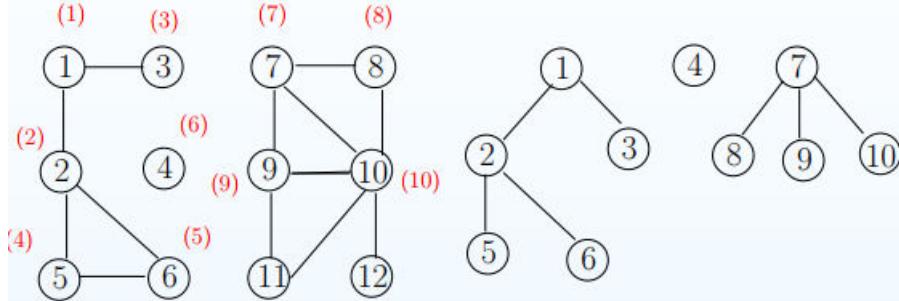
$$Q : 9 - 10 - 10$$

Σχήμα 6: Γράφημα - ΔκΠ



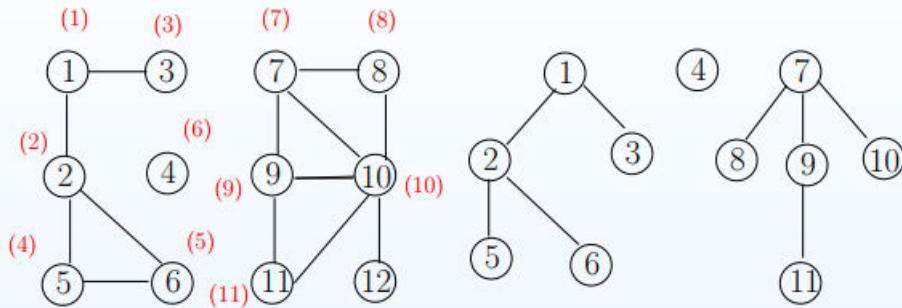
$$Q : 10 - 10 - 10 - 11$$

Σχήμα 6: Γράφημα - ΔκΠ



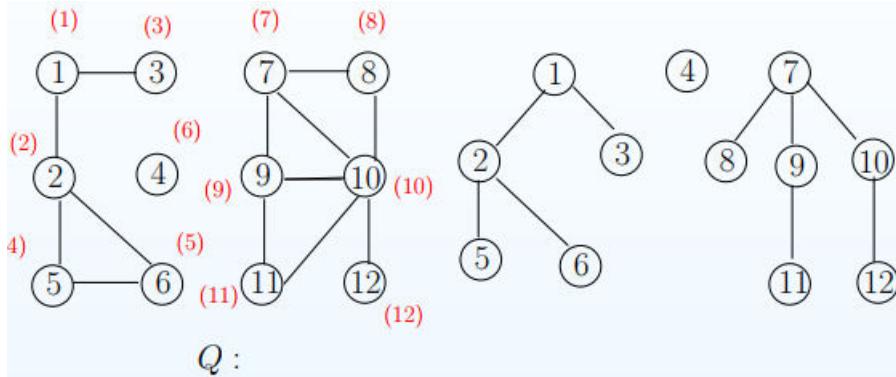
$$Q : 10 - 10 - 11 - 11 - 12$$

Σχήμα 6: Γράφημα - ΔκΠ



$$Q : 11 - 12$$

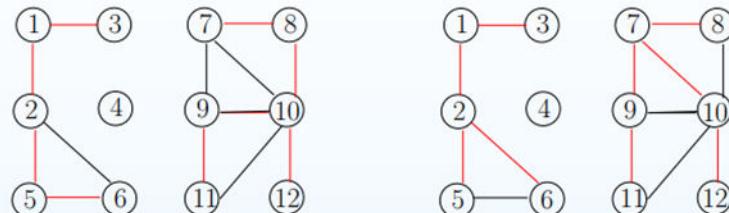
Σχήμα 6: Γράφημα - ΔκΠ



$$Q :$$

Σχήμα 6: Γράφημα - ΔκΠ

Παρατηρήσεις



Σχήμα 7: Γεννητορικά δένδρα ΔκΒ και ΔκΠ (κόκκινες ακμές)

Στο Σχήμα 7 απεικονίζονται τα δένδρα που σχημάτισαν οι δυο στρατηγικές διάσχισης. Από κατασκευής το δένδρο της ΔκΠ έχει την ιδιότητα ότι κάθε μονοπάτι που συνδέει τη ρίζα $υ$ με ένα φύλλο $ω$ είναι το συντομότερο ως προς τον αριθμό των ακμών. Για παράδειγμα το μονοπάτι ανάμεσα στις κορυφές 7, 11 με το μικρότερο αριθμό ακμών είναι το 7-9-11. Το αντίστοιχο μονοπάτι που υπολόγισε η ΔκΒ είναι το 7-8-10-9-11.

Πολυπλοκότητα

Αμφότερες οι στρατηγικές διάσχισης επισκέπτονται τις κορυφές του γραφήματος δια μέσου των ακμών.

Παρατηρούμε ότι κάθε φορά που ‘άνακαλύπτεται’ μία ακμή εκτελείται σταθερός αριθμών στοιχειωδών πράξεων. Επίσης η κάθε ακμή ‘άνακαλύπτεται’ δύο φορές (μία φορά για κάθε προσπίπουσα κορυφή). Άρα μπορούμε να θεωρήσουμε ότι για κάθε ακμή εκτελούνται το πολύ c στοιχειώδεις πράξεις. Άρα αν $T(m)$ είναι ο αριθμός των στοιχειωδών πράξεων της διάσχισης ενός γραφήματος με m ακμές, έχουμε

$$T(m) \leq B(m), \text{όπου } B(m) = B(m - 1) + c.$$

Σε κλειστή μορφή έχουμε $B(m) = c \cdot m$ και άρα

$$T(m) = O(n + m)$$

εφόσον στην $T(m)$ συνυπολογίσουμε και το κόστος αρχικοποίησης των δομών που χρησιμοποιεί η διάσχιση.

ΔΚΒ με preorder[] και parent[]

Ιδέα

Θεωρούμε την αναδρομική έκδοση του αλγόριθμου όπως αυτός αποτυπώνεται από την παρακάτω διαδικασία

```
explore( $v$ , visited)
visited[ $v$ ]  $\leftarrow$  true;
for  $u \in N(v)$ 
  if not visited[ $u$ ] then explore( $u$ , visited);
```

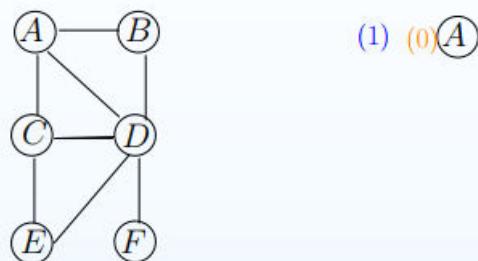
Έχουμε ήδη δει ότι η παραπάνω διαδικασία μπορεί να εμπλουτίσει με τη μεταβλητή *clock* και τον πίνακα *preorder[]*. Ο μηχανισμός αυτός καταγράφει την πρώτη φορά που συναντάμε κάθε κορυφή κατά την διάρκεια της διάσχισης. Επίσης θα χρησιμοποιήσουμε τον πίνακα *parent[]* ο οποίος για κάθε κορυφή u θα καταγράφει τον άμεσο πρόγονο της στο δένδρο που δημιουργεί η ΑΚΒ. Έτσι η ακμή $\{v, u\}$ θα ανήκει στο δένδρο ανν $v = parent[u]$. Η ρίζα του δένδρου r θα έχει $parent[r] = 0$.

Αλγόριθμος

```
explore(clock, v, visited, preorder, parent)
    visited[v] ← true;
    preorder[v] ← ++clock;
    for u ∈ N(v)
        if not visited[u] then
            parent[u] ← v;
            explore(clock, u, visited, preorder, parent);

main()
    clock ← 0;
    for v ∈ V
        visited[v] ← false;
        preorder[v] ← 0; parent[v] ← 0;
    for v ∈ V
        if not visited[v] then
            explore(clock, v, visited, preorder, parent);
```

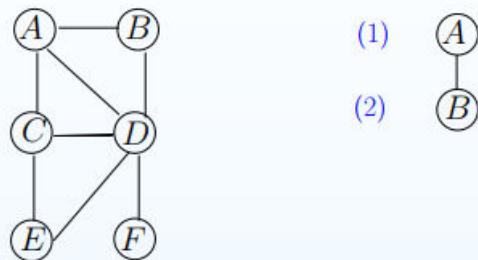
Παράδειγμα



(1) (0) A

parent = (0, 0, 0, 0, 0, 0)

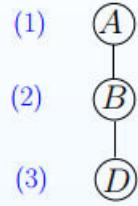
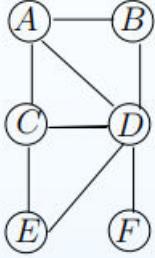
Σχήμα 9: Γράφημα - ΔκΒ



(1)
(2) A
B

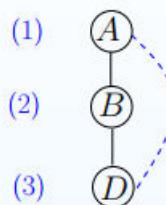
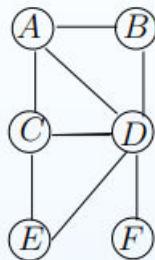
parent = (0, A, 0, 0, 0, 0)

Σχήμα 9: Γράφημα - ΔκΒ



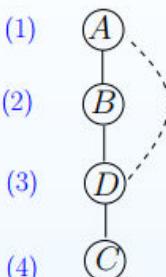
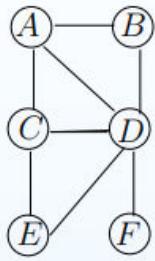
parent= (0, A, 0, B, 0, 0)

Σχήμα 9: Γράφημα - ΔκΒ



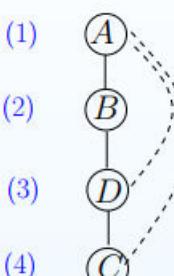
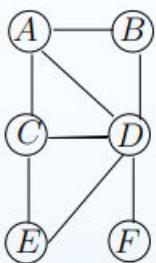
parent= (0, A, 0, B, 0, 0)

Σχήμα 9: Γράφημα - ΔκΒ



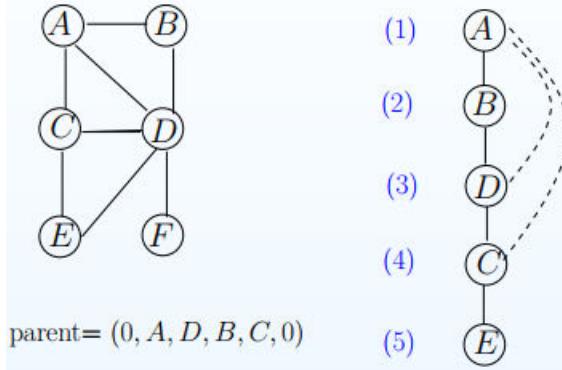
parent= (0, A, D, B, 0, 0)

Σχήμα 9: Γράφημα - ΔκΒ

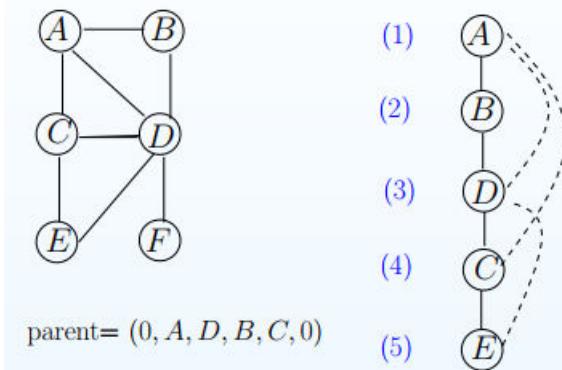


parent= (0, A, D, B, 0, 0)

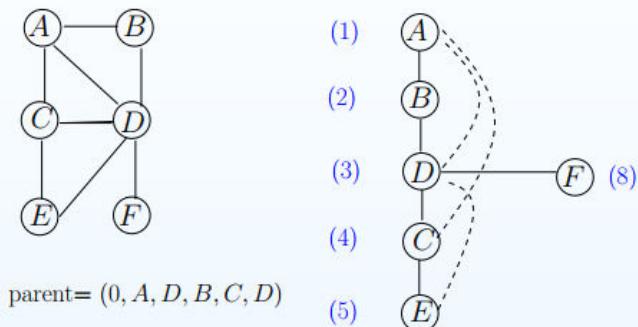
Σχήμα 9: Γράφημα - ΔκΒ



Σχήμα 9: Γράφημα - ΔκΒ



Σχήμα 9: Γράφημα - ΔκΒ



Σχήμα 9: Γράφημα - ΔκΒ

Έστω T το δένδρο της ΔκΒ. Για κάθε ακμή (v, u) (κορυφές εμφανίζονται στο ζευγάρι με τη σειρά που η ακμή προστίθεται στο δένδρο) $parent[u] = v$, αν η ακμή ανήκει στο T .

Παρατηρήσεις

Μετά την εκτέλεση της ΔκΒ, οι ακμές του G που δεν ανήκουν στο σύνολο του παραχθέντος γεννητορικού δένδρου ονομάζονται *οπισθοακμές*. Αυτές απεικονίζονται στο Σχήμα 9 σαν διακεκομμένες.

Mία οπισθοακμή πιστοποιεί την ύπαρξη κύκλου.

Εφόσον μπορούμε μέσω του μονοδιάστατου πίνακα parent να διαπιστώσουμε αν μία ακμή ανήκει στο δένδρο αμέσως γνωρίζουμε ότι αν δεν ανήκει αποτελεί οπισθοακμή και άρα στο γράφημα υπάρχει κύκλος.

Πολυπλοκότητα

Αμφότερες οι στρατηγικές διάσχισης επισκέπτονται τις κορυφές του γραφήματος δια μέσου των ακμών.

Παρατηρούμε ότι κάθε φορά που ‘άνακαλύπτεται’ μία ακμή εκτελείται σταθερός αριθμόν στοιχειωδών πράξεων. Επίσης η κάθε ακμή ‘άνακαλύπτεται’ δύο φορές (μία φορά για κάθε προσπίπτουσα κορυφή). Άρα μπορούμε να θεωρήσουμε ότι για κάθε ακμή εκτελούνται το πολύ c στοιχειώδεις πράξεις. Άρα αν $T(m)$ είναι ο αριθμός των στοιχειωδών πράξεων της διάσχισης ενός γραφήματος με m ακμές, έχουμε

$$T(m) \leq B(m), \text{όπου } B(m) = B(m - 1) + c.$$

Σε κλειστή μορφή έχουμε $B(m) = c \cdot m$ και άρα

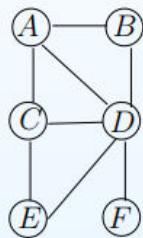
$$T(m) = O(n + m)$$

εφόσον στην $T(m)$ συνυπολογίσουμε και το κόστος αρχικοποίησης των δομών που χρησιμοποιεί η διάσχιση.

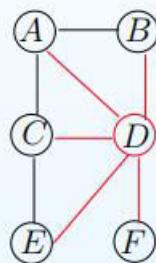
Comp_low (Εύρεση Σημείου Κοπής)

Iδέα

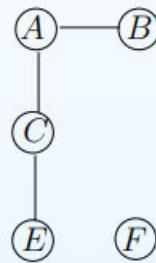
Μία από τις πολλές εφαρμογές της ΔκΒ είναι η εύρεση των σημείων κοπής, δηλαδή η εύρεση των κορυφών των οποίων η αφαίρεση αποσυνθέτει το γράφημα σε γραφικές συνιστώσες.



Σχήμα 10: Σημείο Κοπής



Σχήμα 10: Σημείο Κοπής



Σχήμα 10: Σημείο Κοπής

Θεωρούμε ότι έχει εκτελεστεί η ΔκΒ και σε κάθε κορυφή v έχει αποδοθεί τιμή $preorder[u]$.

Στη συνέχεια, για κάθε κορυφή $v \in V$, πρέπει να υπολογιστεί μία τιμή $low[v]$, η οποία να είναι η μικρότερη τιμή $preorder[u]$ στην οποία μπορεί να φτάσει ένα μονοπάτι που να ξεκινάει από την v ή κάποιο απόγονό της στο δένδρο της ΔκΒ και να χρησιμοποιεί ακριβώς μία οπισθοακμή.

Μία κορυφή $v \in V$ θα είναι σημείο κοπής αν είτε

- είναι ρίζα του δένδρου ΔκΒ και έχει πάνω από ένα κόμβο παιδί, ή,
- έχει παιδί u στο δένδρο ΔκΒ (δηλαδή γειτονική κορυφή στο δένδρο της ΔκΒ) με τιμή $low[u] \geq preorder[v]$.

Αλγόριθμος

Θεωρούμε ότι έχει ήδη εκτελεστεί ΔκΒ και οι πίνακες $preorder$, $parent$ έχουν ήδη πάρει τιμές.

```

comp_low( $v$ ,  $visited$ ,  $preorder$ ,  $parent$ ,  $low$ ,  $cut\_node$ )
     $visited[v] \leftarrow true$ ;
     $low[v] \leftarrow preorder[v]$ ;
    for  $u \in N(v)$ 
        if not  $visited[u]$  then
             $comp\_low(u, visited, preorder, parent, low, cut\_node)$ 
            if  $low[v] > low[u]$  then
                 $low[v] \leftarrow low[u]$ ;
            if  $low[u] \geq preorder[v]$  then
                 $cut\_node[v] \leftarrow true$ ;
            else /*  $visited[u] = true$  */
                if  $u \neq parent[v]$  then /*( $v, u$ ) οπισθοακμή */
                    if  $low[v] > preorder[u]$  then
                         $low[v] \leftarrow preorder[u]$ ;

```

```

main()
    clock  $\leftarrow 0$ ;
    for  $v \in V$ 
         $\text{visited}[v] \leftarrow \text{cut\_node}[v] \leftarrow \text{false}$ ;
         $\text{preorder}[v] \leftarrow \text{parent}[v] \leftarrow 0$ ;
    for  $v \in V$ 
        if not  $\text{visited}[v]$  then
             $\text{explore}(\text{clock}, v, \text{visited}, \text{preorder}, \text{parent})$ ;
        for  $v \in V$   $\text{visited}[v] \leftarrow \text{false}$ ;
        for  $v \in V$ 
            if not  $\text{visited}[v]$  then
                 $\text{comp\_low}(v, \text{visited}, \text{preorder}, \text{parent}, \text{low}, \text{cut\_node})$ ;
    for  $v \in V$ 
        if  $\text{cut\_node}[v]$  then
            if  $\text{parent}[v] \neq 0$  then
                 $\text{print}(v)$ ;
            else /*κορυφή  $v$  είναι ρίζα της ΔΚΒ*/
                 $\text{num\_root\_children} \leftarrow 0$ ;
                for  $u \in N(v)$ 
                    if  $\text{parent}[u] = v$  then  $\text{num\_root\_children}++$ ;
                if  $\text{num\_root\_children} \geq 2$  then
                     $\text{print}(v)$ ;

```

Παράδειγμα



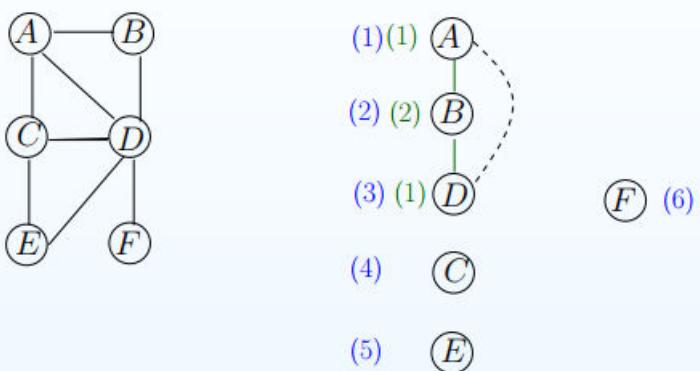
Σχήμα 12: Υπολογισμός *low*



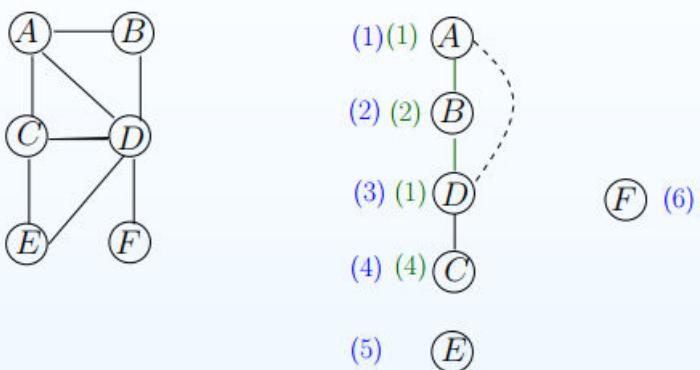
Σχήμα 12: Υπολογισμός *low*



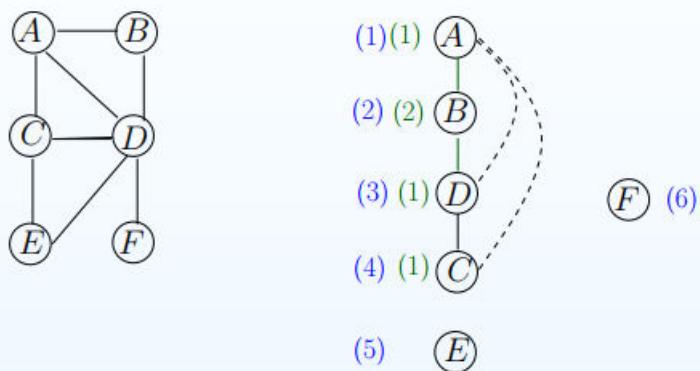
Σχήμα 12: Υπολογισμός *low*



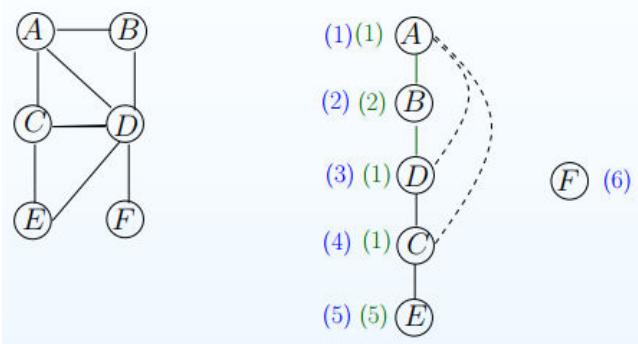
Σχήμα 12: Υπολογισμός *low*



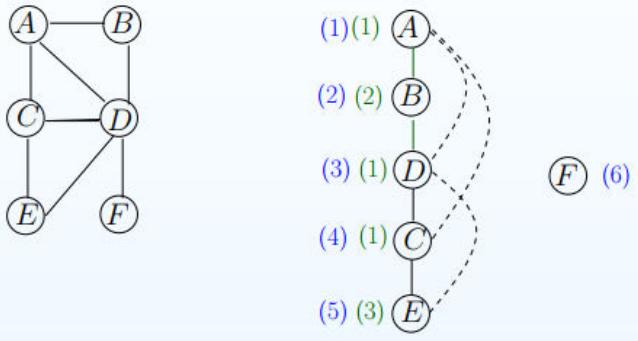
Σχήμα 12: Υπολογισμός *low*



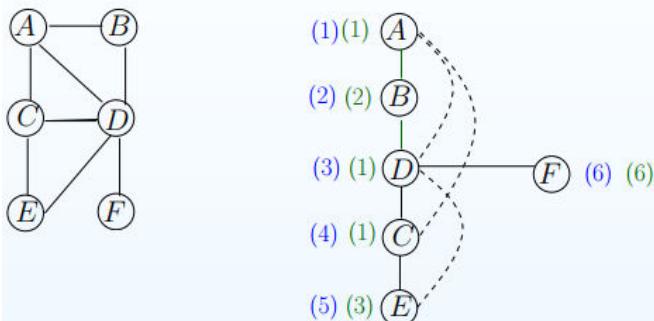
Σχήμα 12: Υπολογισμός *low*



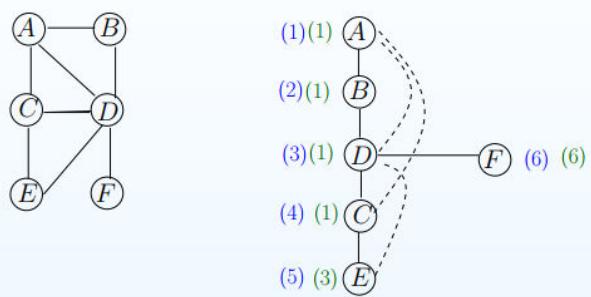
Σχήμα 12: Υπολογισμός *low*



Σχήμα 12: Υπολογισμός *low*



Σχήμα 12: Υπολογισμός *low*



Σχήμα 12: Υπολογισμός *low*

Επειδή

$$\text{low}[F] = 6 \geq \text{preorder}[D] = 3$$

η κορυφή D αναγνωρίζεται σαν σημείο κοπής. Πράγματι η διαγραφή της από το γράφημα δημιουργεί μία γραφική συνιστώσα που περιέχει την κορυφή F και μία άλλη με όλες τις υπόλοιπες κορυφές.

Διάσχιση κατά Βάθος (Depth-First Search)

Ιδέα

Το πρόβλημα της διάσχισης είναι το ίδιο όπως και στην περίπτωση των μη-κατευθυνόμενων γραφημάτων.

Πρόβλημα 1 Επίσκεψη των κορυφών του γραφήματος $G(V, E)$ με συστηματικό τρόπο.

Στην περίπτωση των κατευθυνόμενων γραφημάτων η επίσκεψη στις κορυφές γίνεται κατά τη φορά των ακμών. Δηλαδή, από μία κορυφή v επισκεπτόμαστε τις κορυφές του συνόλου $N^+(v)$ (και όχι τις κορυφές του συνόλου $N^-(v)$).

Ο αλγόριθμος λειτουργεί όπως και στην περίπτωση των μη-κατευθυνόμενων γραφημάτων. Η διαφορά βρίσκεται στην πληροφορία που καταγράφεται κατά τη διάρκεια της διάσχισης. Συγκεκριμένα, στην περίπτωση των μη-κατευθυνόμενων γραφημάτων καταγραφόταν η πρώτη φορά επίσκεψης και ο άμεσος πρόγονος μίας κορυφής στους πίνακες *preorder* και *parent*, αντίστοιχα. Στην περίπτωση των κατευθυνόμενων γραφημάτων ο πίνακας *preorder* διατηρείται ενώ ο πίνακας *parent* αντικαθίσταται από τον πίνακα *postorder*. Στον πίνακα αυτό καταγράφεται για κάθε κορυφή v η τελευταία φορά που συναντάται από τη διάσχιση. Έτσι ο χρόνος (αριθμός των βημάτων) που μία κορυφή v βρίσκεται στη στοίβα (της ΔΚΒ) είναι

$$postorder[v] - preorder[v] + 1$$

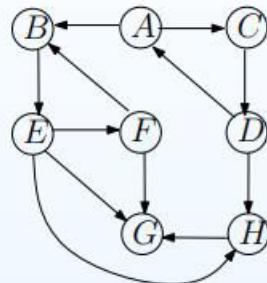
Αλγόριθμος

```
explore(clock, v, visited, preorder, postorder)
    visited[v] ← true;
    preorder[v] ← ++clock;
    for u ∈ N+(v)
        if not visited[u] then
            explore(clock, u, visited, preorder, postorder);
            postorder[v] ← ++clock;

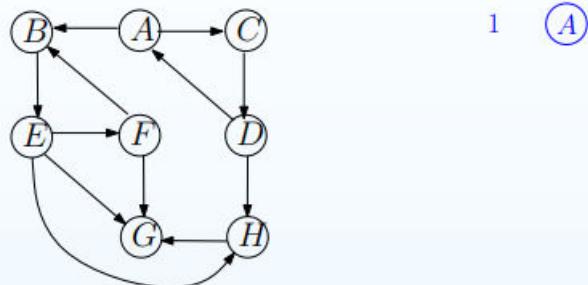
main()
    clock ← 0;
    for v ∈ V
        visited[v] ← false;
        preorder[v] ← 0; postorder[v] ← 0;
    for v ∈ V
        if not visited[v] then
            explore(clock, v, visited, preorder, postorder);
```

Παράδειγμα

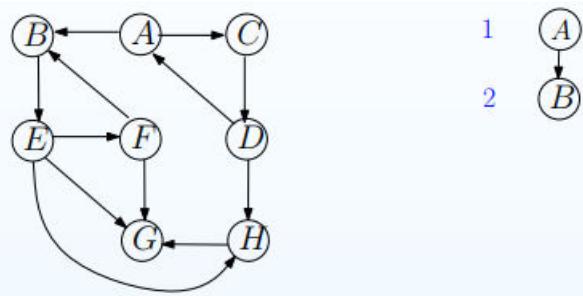
Να πραγματοποιηθεί ΔκΒ στο γράφημα του Σχήματος 3.



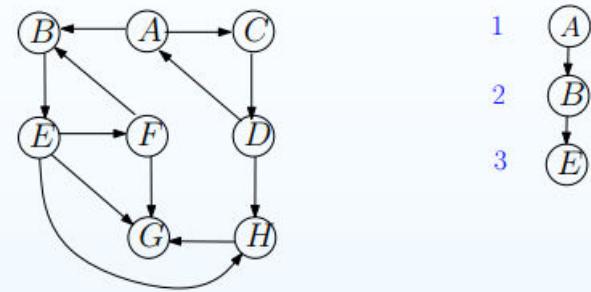
Σχήμα 3: Γράφημα - ΔκΒ



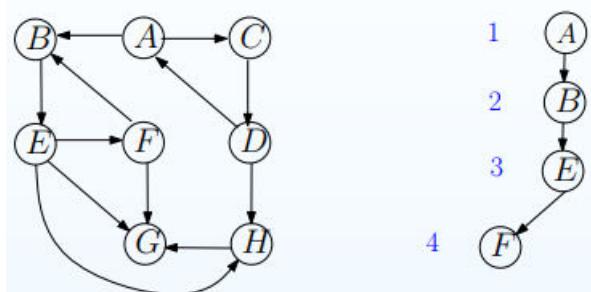
Σχήμα 4: Γράφημα - ΔκΒ



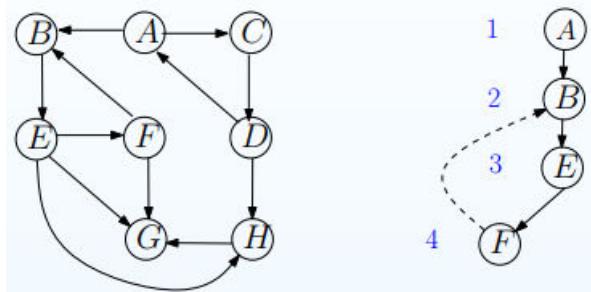
Σχήμα 4: Γράφημα - ΔκΒ



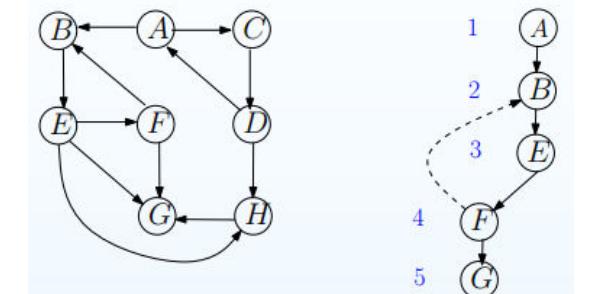
Σχήμα 4: Γράφημα - ΔκΒ



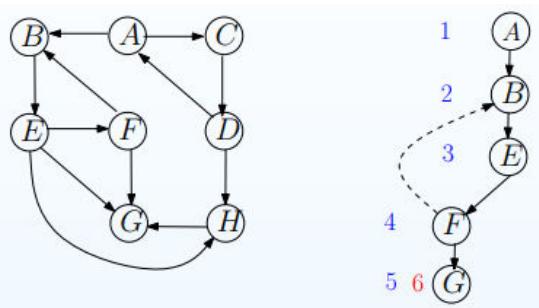
Σχήμα 4: Γράφημα - ΔκΒ



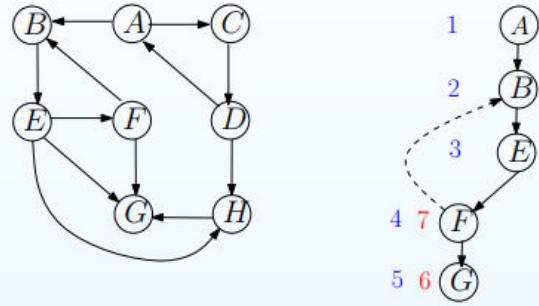
Σχήμα 4: Γράφημα - ΔκΒ



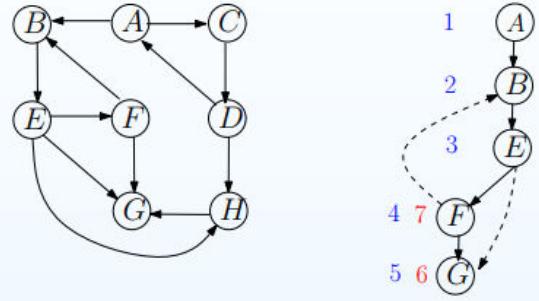
Σχήμα 4: Γράφημα - ΔκΒ



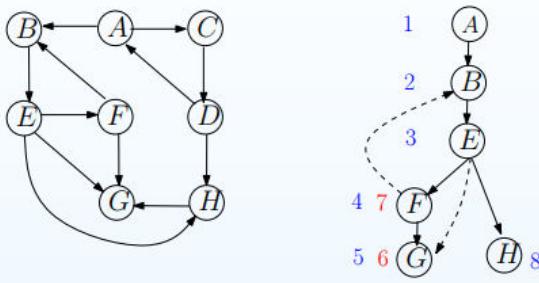
Σχήμα 4: Γράφημα - ΔκΒ



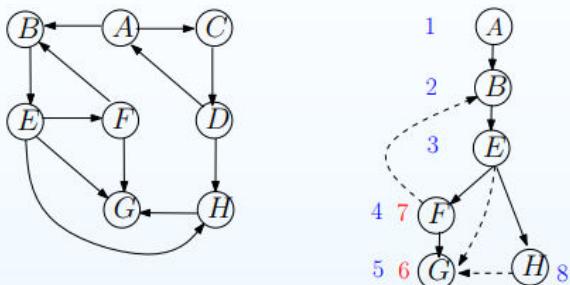
Σχήμα 4: Γράφημα - ΔκΒ



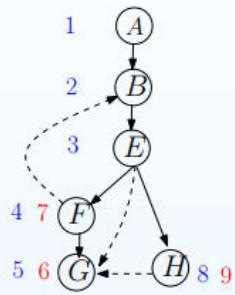
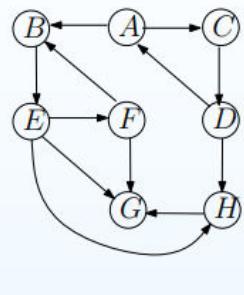
Σχήμα 4: Γράφημα - ΔκΒ



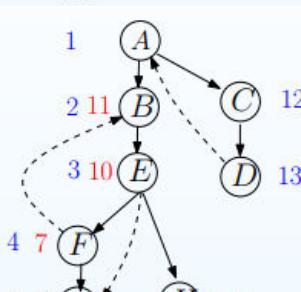
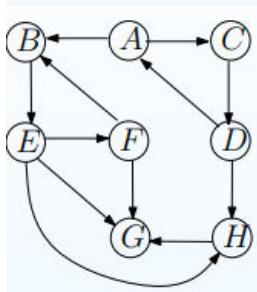
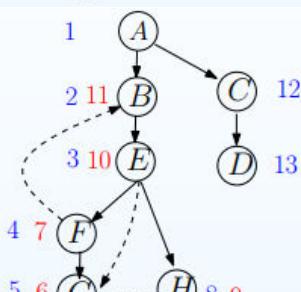
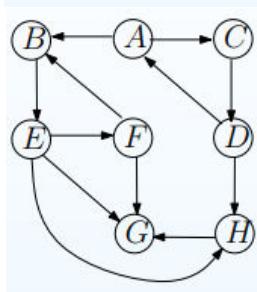
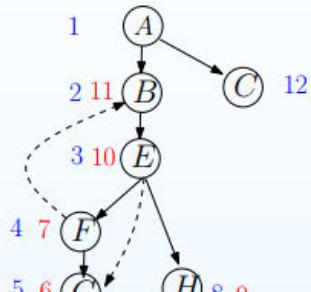
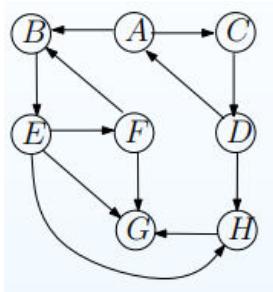
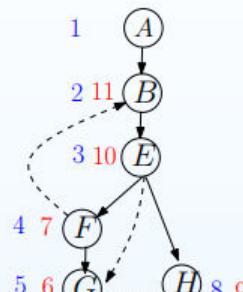
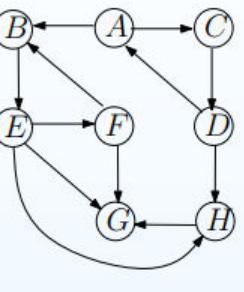
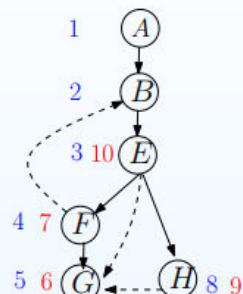
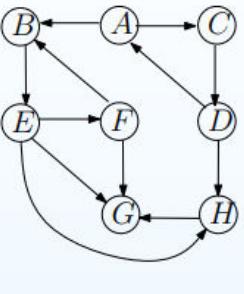
Σχήμα 4: Γράφημα - ΔκΒ

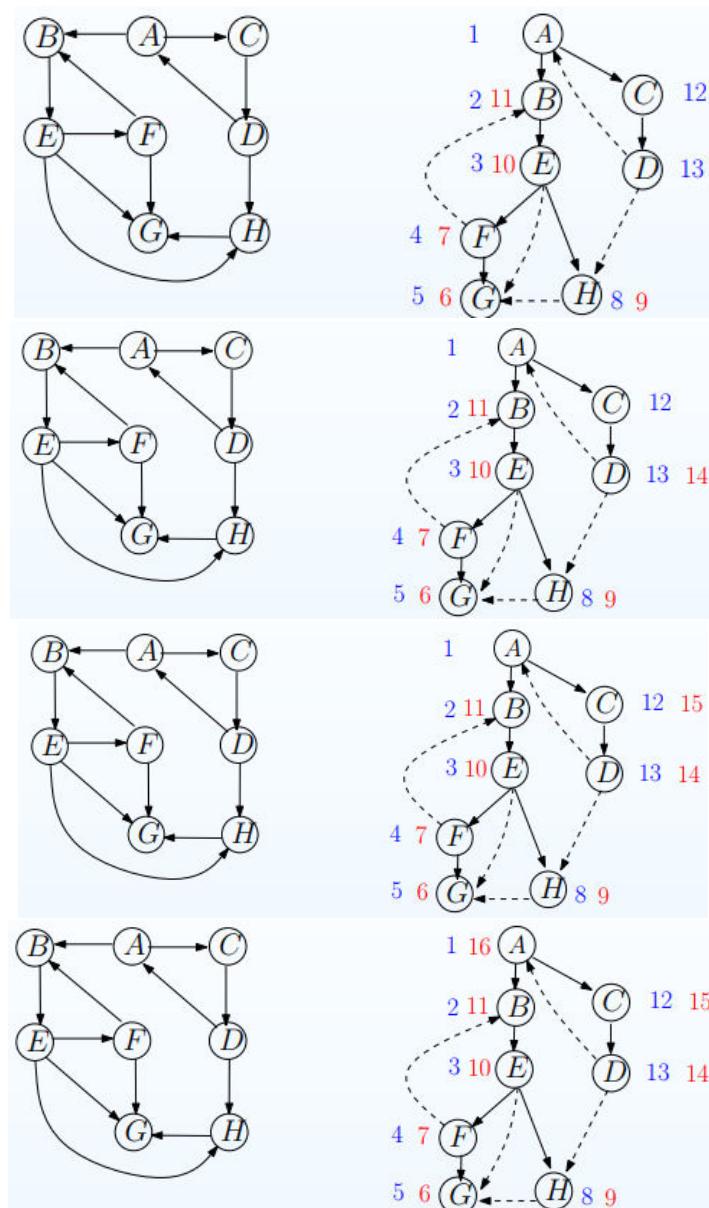


Σχήμα 4: Γράφημα - ΔκΒ



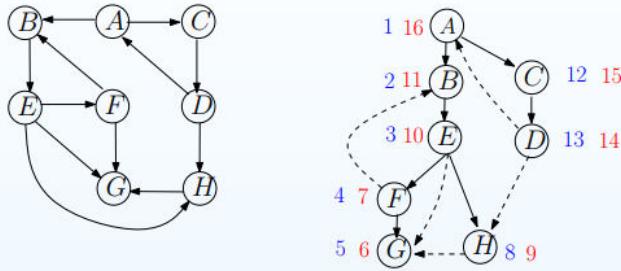
Σχήμα 4: Γράφημα - ΔΚΒ





Παρατηρήσεις

- Η ύπαρξη οπισθοακμής υποδηλώνει την ύπαρξη κατευθυνόμενου κύκλου
- Η ΔκΒ μπορεί να εντοπίσει αν υπάρχει κατευθυνόμενο μονοπάτι που να συνδέει δύο διθείσες κορυφές καθώς και κύκλο.
- Αν το γράφημα δεν έχει (κατευθυνόμενο) κύκλο τότε ονομάζεται άκυκλο κατευθυνόμενο γράφημα (ΑΚΓ).



Σχήμα 5: Γράφημα - ΔκΒ

Ακμή (v, u) είναι

- εμπροσθοακμή αν $pre[v] < pre[u] < post[u] < post[v]$,
- οπισθοακμή αν $pre[u] < pre[v] < post[v] < post[u]$,
- εγκάρσια αν $[pre[v], post[v]] \cap [pre[u], post[u]] = \emptyset$.

Οι δενδρικές ακμές είναι εμπροσθοακμές αλλά δεν ισχύει ότι κάθε εμπροσθοακμή είναι δενδρική ακμή (δες ακμή (E, G) στο Σχήμα 5)

Πολυπλοκότητα

Αμφότερες οι στρατηγικές διάσχισης επισκέπτονται τις κορυφές του γραφήματος δια μέσου των ακμών.

Παρατηρούμε ότι κάθε φορά που ‘άνακαλύπτεται’ μία ακμή εκτελείται σταθερός αριθμόν στοιχειωδών πράξεων. Επίσης η κάθε ακμή ‘άνακαλύπτεται’ δύο φορές (μία φορά για κάθε προστίπουσα κορυφή). Άρα μπορούμε να θεωρήσουμε ότι για κάθε ακμή εκτελούνται το πολύ c στοιχειώδεις πράξεις. Άρα αν $T(m)$ είναι ο αριθμός των στοιχειωδών πράξεων της διάσχισης ενός γραφήματος με m ακμές, έχουμε

$$T(m) \leq B(m), \text{όπου } B(m) = B(m-1) + c.$$

Σε κλειστή μορφή έχουμε $B(m) = c \cdot m$ και άρα

$$T(m) = O(n + m)$$

εφόσον στην $T(m)$ συνυπολογίζουμε και το κόστος αρχικοποίησης των δομών που χρησιμοποιεί η διάσχιση.

Ιδέα

Η έννοια της συνδεσιμότητας στα κατευθυνόμενα γραφήματα είναι διαφορετική από την αντίστοιχη έννοια στα μη-κατευθυνόμενα γραφήματα. Συγκεκριμένα, ένα κατευθυνόμενο γράφημα αποτελείται (μπορεί να αποσυνδεθεί σε) *Ισχυρά Συνδεδεμένες Γραφικές Συνιστώσες* (ΙΣΓΣ).

Ορισμός 2 *Mία ΙΣΓΣ είναι ένα μεγιστοτικό υπογράφημα εινός κατευθυνόμενου γραφήματος με την ιδιότητα ότι για κάθε δύο κορυφές v , u που ανήκουν σε αυτή ισχύει ότι υπάρχει κατευθυνόμενο μονοπάτι και από την v στη u αλλά και αντίστροφα.*

Ιδιότητα 1 Αν εκτελέσουμε ΔκΒ σε μία ΙΣΓΣ εκκινώντας από μία κορυφή v της Π θα μπορέσουμε να προσπελάσουμε όλους τις κορυφές της συνιστώσας αυτής.

Ιδιότητα 2 Αν εκτελέσουμε ΔκΒ σε όλο το γράφημα, η κορυφή με την μεγαλύτερη τιμή *postorder* βρίσκεται σε μία αφετηριακή ΙΣΓΣ

Ιδιότητα 3 Αν C και C' δύο ΙΣΓΣ και υπάρχει ακμή από κάποια κορυφή v πρώτης προς τη δεύτερη, τότε ο μεγαλύτερος αριθμός *postorder* στη C είναι μεγαλύτερος από τον μεγαλύτερο αριθμό *postorder* στη C' .

Η τελευταία ιδιότητα μας λέει ότι μπορούμε να κατατάξουμε τις ΙΣΓΣ σύμφωνα με το μεγαλύτερο αριθμό *postorder* που εμφανίζεται σε κάποια από τις κορυφές τους. Επίσης σύμφωνα με την Ιδιότητα 2 μπορούμε να ξεκινήσουμε από μία αφετηριακή ΙΣΓΣ. Θα εκτελέσουμε ΔκΒ σε αυτή και αφού προσπελάσουμε όλες τις κορυφές της (σύμφωνα με την Ιδιότητα 1) μπορούμε να “διαγράψουμε” τις κορυφές της και να επαναλάβουμε τη διαδικασία αυτή.

Πρόβλημα: Πως όμως θα εντοπίσουμε μία αφετηριακή ΙΣΓΣ;

Ιδέα: Θα χρησιμοποιήσουμε το γράφημα G^R . Το γράφημα αυτό παράγεται από το G αν αντιστρέψουμε τη φορά των ακμών. Παρατηρήστε ότι το G^R έχει τις ίδιες ΙΣΓΣ όπως και το G μόνο που οι αφετηριακές και τερματικές ΙΣΓΣ είναι σε αντίστροφους ρόλους.

1. Εκτελούμε ΔκΒ στο G υπολογίζοντας *postorder*.
2. Παράγουμε τον γράφημα G^R
3. Επιλέγουμε την κορυφή v με το μεγαλύτερο αριθμό *postorder* και εκτελούμε ΔκΒ στο G^R εκκινώντας από αυτή. Οι κορυφές που ανακαλύππονται από τη ΔκΒ ανήκουν στην ίδια ΙΣΓΣ.
4. Διαγράφουμε την ΙΣΓΣ.
5. Αν υπάρχουν κορυφές που δεν τις έχει επισκεφθεί η ΔκΒ επιστρέφουμε στο Βήμα 3.

Ο Αλγόριθμος που παρουσιάζουμε στη συνέχεια προϋποθέτει ότι έχει πάρει τιμές ο πίνακας *postorder*. Δηλαδή έχει προηγηθεί στο αρχικό γράφημα η ΔκΒ.

Ο Αλγόριθμος χρησιμοποιεί το σύνολο Q στον οποίο αποθηκεύει τις ΙΣΓΣ.

Αλγόριθμος

scc(*postorder*)

```

for  $v \in V$   $\text{visited}[v] \leftarrow \text{false};$ 
while  $\exists u \in V$  such that  $\text{visited}[u] = \text{false}$ 
     $v \leftarrow \text{argmax}\{\text{postorder}[u] : u \in V, \text{visited}[u] = \text{false}\};$ 
     $Q \leftarrow \emptyset;$ 
    explore( $v, \text{visited}, Q$ );
    print  $Q$ ;
```

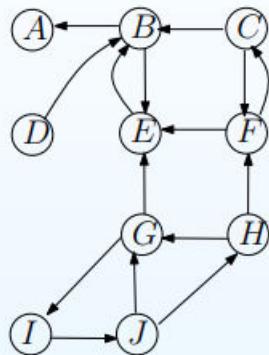
όπου

```

explore( $v, \text{visited}, Q$ );
     $\text{visited}[v] \leftarrow \text{true};$ 
     $Q \leftarrow Q \cup \{v\};$ 
    for  $u \in N^-(v)$ 
        if not  $\text{visited}[u]$  then
            explore( $v, \text{visited}, Q$ );
```

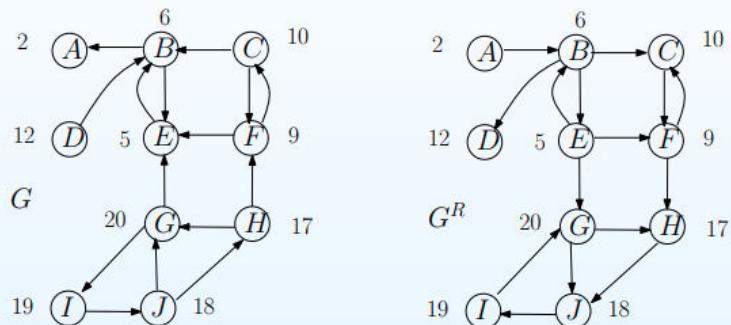
Παράδειγμα

Να βρεθούν οι ΙΣΓΣ στο γράφημα του Σχήματος 7



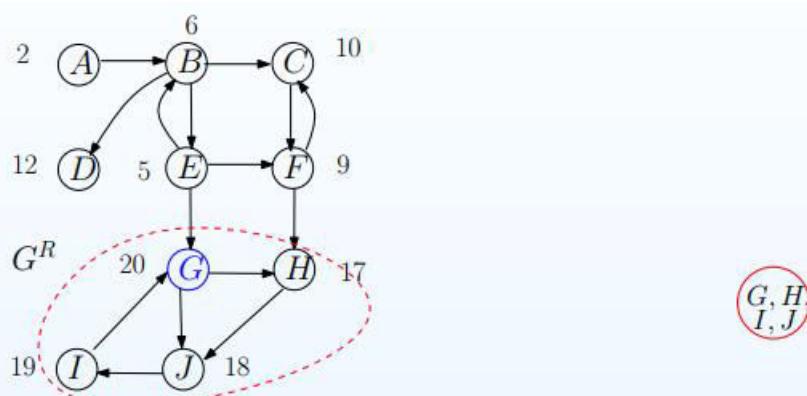
Σχήμα 7: Κατευθυνόμενο Γράφημα

Εκτελούμε ΔκΒ προκειμένου να υπολογιστούν τα στοιχεία του πίνακα *postorder*. Οι τιμές του πίνακα σημειώνονται δίπλα σε κάθε κορυφή στο Σχήμα 8.

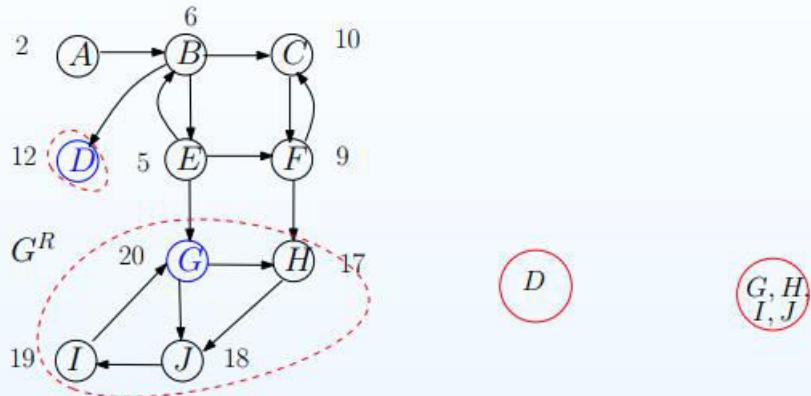


Σχήμα 8: Γραφήματα G και G^R

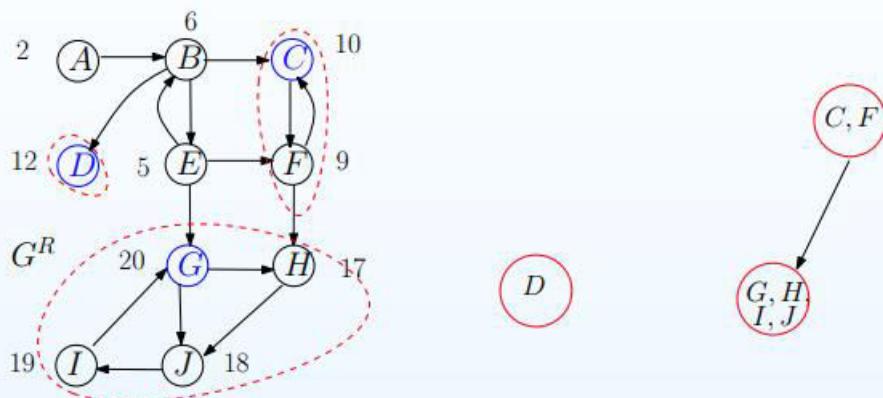
Στη συνέχεια κατασκευάζουμε το γράφημα G^R με αντιστροφή της φοράς των ακμών (Σχήμα 8). Ο Αλγόριθμος συνεχίζει με την ανάδειξη των ΙΣΓΣ ξεκινώντας από την κορυφή σε κάθε συνιστώσα με το μεγαλύτερο αριθμό *postorder*.



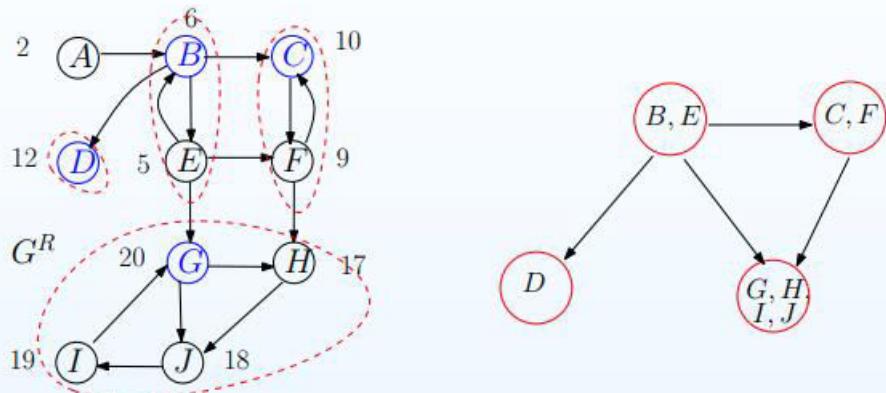
Σχήμα 9: Υπολογισμός ΙΣΓΣ - κορυφή με μεγαλύτερο συντελεστή *postorder* σε μπλε.



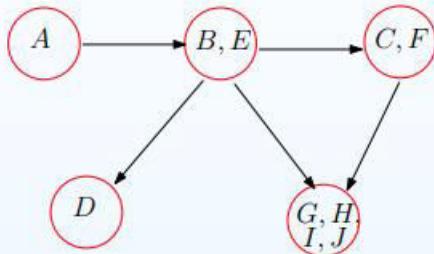
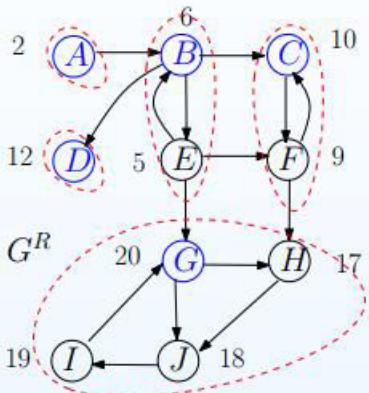
Σχήμα 9: Υπολογισμός ΙΣΓΣ - κορυφή με μεγαλύτερο συντελεστή *postorder* σε μπλε.



Σχήμα 9: Υπολογισμός ΙΣΓΣ - κορυφή με μεγαλύτερο συντελεστή *postorder* σε μπλε.



Σχήμα 9: Υπολογισμός ΙΣΓΣ - κορυφή με μεγαλύτερο συντελεστή *postorder* σε μπλε.



Σχήμα 9: Υπολογισμός ΙΣΓΣ - κορυφή με μεγαλύτερο συντελεστή *postorder* σε μπλε.

04_ΣΥΝΤΟΜΟΤΕΡΑ ΜΟΝΟΠΑΤΙΑ / 05_shpaths

Dijkstra (Εύρεση ΜΠΣΔ)

Πρόβλημα

Θα ασχοληθούμε με την εύρεση συντομότερων μονοπατιών από μία δεδομένη κορυφή s προς κάθε άλλη κορυφή ενός γραφήματος.

ΜΠΣΔ Δεδομένου ενός εμβαρούς γράφημα $G(V, E, w)$, $w \geq 0$ και μίας κορυφής του $s \in V$, να βρεθούν τα ελάχιστα μονοπάτια από το s προς κάθε άλλη κορυφή του γραφήματος

Στην περίπτωση που το γράφημα δεν είναι εμβαρές (ισοδύναμο με την περίπτωση όπου $w(e) = 1$, για κάθε $e \in E$), το πρόβλημα επιλύεται με τη διάσχιση κατά πλάτος (ΔκΠ) εκκινώντας από την κορυφή s .

Ο αλγόριθμος του Dijkstra επιλύει το πρόβλημα ΜΠΣΔ στη γενική περίπτωση.

Iδέα

Έστω d_v η μεταβλητή που θα περιέχει το μήκος του συντομότερου μονοπατιού που συνδέει την αφετηρία (κορυφή s) με την κορυφή v στο τέλος της εκτέλεσης του Αλγόριθμου. Ο Αλγόριθμος, κατά τη διάρκεια της εκτέλεσης, επανυπολογίζει την τιμή της μεταβλητής d_v , για κάθε κορυφή v , μέχρι να είναι ίση με το μήκος που αναφέραμε παραπάνω.

Βασική Ιδέα

Σε κάθε στιγμή, ο Αλγόριθμος διαχωρίζει τις κορυφές σε δύο ομάδες: στην ομάδα των μόνιμων κορυφών (σύνολο S) και στην ομάδα των μη-μόνιμων κορυφών. Στην πρώτη ομάδα ανήκουν οι κορυφές των οποίων η τιμή της μεταβλητής d δεν θα αλλάξει μέχρι το τέλος του αλγόριθμου. Δηλαδή, για κάθε μόνιμη κορυφή έχει υπολογιστεί το συντομότερο μονοπάτι από την αφετηρία. Στη δεύτερη ομάδα ανήκουν οι υπόλοιπες κορυφές.

Λειτουργία

Ο Αλγόριθμος λειτουργεί επαναληπτικά. Η κάθε επανάληψη χωρίζεται σε δύο φάσεις.

Φάση I

Από τις μη-μόνιμες κορυφές επιλέγεται αυτή που έχει το μικρότερο d και γίνεται μόνιμη. Έστω ότι αυτή είναι η κορυφή v^* . Δηλαδή,

$$v^* = \operatorname{argmin}\{d_v : v \in V \setminus S\}. \quad (1)$$

Στη συνέχεια η κορυφή v^* προστίθεται στο σύνολο S .

Φάση II

Για κάθε ΓΕΙΤΟΝΙΚΗ ΚΟΡΥΦΗ u της v^* η οποία ΑΝΗΚΕΙ ΣΤΟ ΣΥΝΟΛΟ $V \setminus S$ επανυπολογίζεται η τιμή της μεταβλητής d_u από τον τύπο

$$d_u = \min\{d_u, d_{v^*} + w(v^*, u)\}, \quad (2)$$

όπου $w(v^*, u)$ είναι το βάρος (μήκος) της ακμής (v^*, u) .

Η (2) υποδηλώνει ότι η τιμή της d_u θα αλλάξει αν το μονοπάτι που συνδέει την s με τη u και ΠΕΡΝΑΕΙ ΑΠΟ ΤΗΝ κορυφή v^* είναι συντομότερο από το μονοπάτι που μέχρι πρότινος συνέδεε την s με τη u (προηγούμενη τιμή της d_u .)

Ο Αλγόριθμος εκτελεί $|V| - 1$ επαναλήψεις. Στην Φάση I της πρώτης επανάληψης γίνεται μόνιμη η αφετηρία (κορυφή s). Αυτό είναι το αποτέλεσμα της αρχικοποίησης του Αλγόριθμου κατά την οποία τίθεται

$$d_s \leftarrow 0, \quad d_v \leftarrow \infty, \forall v \in V \setminus \{s\}, \quad S \leftarrow \emptyset.$$

Αλγόριθμος

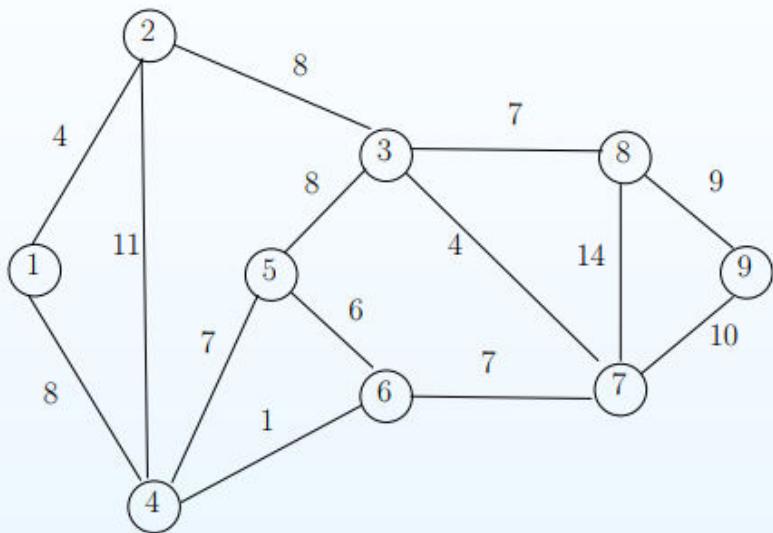
```
Require:  $w : E \rightarrow \mathbb{R}_+$ 
1: void Dijkstra( $G(V, E, w), s$ )
2: for all  $v \in V$  do
3:    $d_v \leftarrow \infty; p_v \leftarrow \text{NULL};$ 
4: end for
5:  $S \leftarrow \emptyset; d_s \leftarrow 0;$ 
6: for  $i \leftarrow 1; i \leq |V| - 1; i + +$  do
7:    $v^* \leftarrow \operatorname{argmin}\{d_v : v \in V \setminus S\};$ 
8:    $S \leftarrow S \cup \{v^*\};$ 
9:   for all  $u \in V \setminus S$  and  $u \in N(v^*)$  do
10:    if  $d_u > d_{v^*} + w(v^*, u)$  then
11:       $d_u \leftarrow d_{v^*} + w(v^*, u);$ 
12:       $p_u \leftarrow v^*;$ 
13:    end if
14:   end for
15: end for
```

Παρατηρήσεις

- Οι μεταβλητές p_v αποθηκεύουν την προτελευταία κορυφή στο συντομότερο μονοπάτι από την κορυφή s στην v . Με αυτό τον τρόπο μπορούμε να ανακτήσουμε όλο το μονοπάτι από την s στην v (όχι μόνο την απόσταση του που δίνεται από την τιμή της d_v).
- Οι μεταβλητές d_v και p_v μπορούν να υλοποιηθούν σαν μονοδιάστατοι πίνακες με πλήθος στοιχείων μεγαλύτερο-ίσο με τον πληθάριθμο του αριθμού των κορυφών.

Παράδειγμα

Να βρεθούν τα συντομότερα μονοπάτια από την κορυφή 1 προς τις υπόλοιπες κορυφές στο γράφημα του Σχήματος 2.



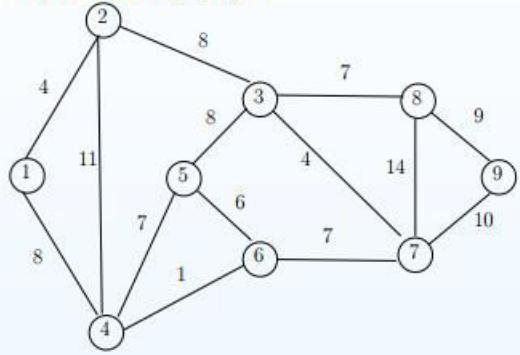
Αρχικοποίηση

$$d = [0, \infty, \infty, \infty, \infty, \infty, \infty, \infty, \infty]$$

$$p = [, , , , , , , ,]$$

$$S = \emptyset$$

Επανάληψη 1



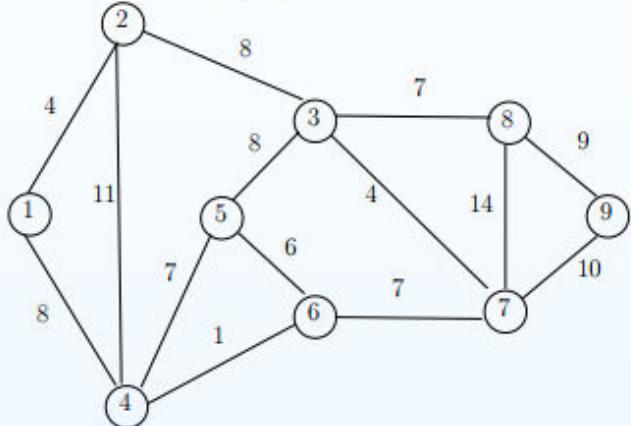
ΦΑΣΗ I

$$\begin{aligned}v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{1, \dots, 9\}\} \\&= \operatorname{argmin}\{0, \infty, \infty, \infty, \infty, \infty, \infty, \infty, \infty\} = 1, \\S &\leftarrow S \cup \{1\} = \emptyset \cup \{1\} = \{1\}\end{aligned}$$

ΦΑΣΗ II

$$\begin{aligned}d_2 &\leftarrow \min\{d_2, d_1 + w(1, 2)\} = \min\{\infty, 0 + 4\} = 4, \quad p_2 \leftarrow 1, \\d_4 &\leftarrow \min\{d_4, d_1 + w(1, 4)\} = \min\{\infty, 0 + 8\} = 8, \quad p_4 \leftarrow 1.\end{aligned}$$

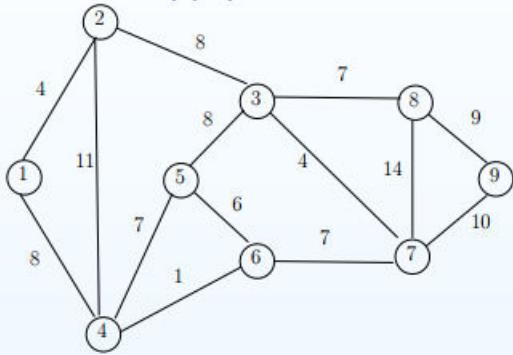
Επανάληψη 1



Έξοδος Επανάληψης

$$\begin{aligned}d &= [\textcolor{red}{0}, 4, \infty, 8, \infty, \infty, \infty, \infty, \infty] \\p &= [\quad, 1, \quad, 1, \quad, \quad, \quad, \quad, \quad] \\S &= \{1\}\end{aligned}$$

Επανάληψη 2



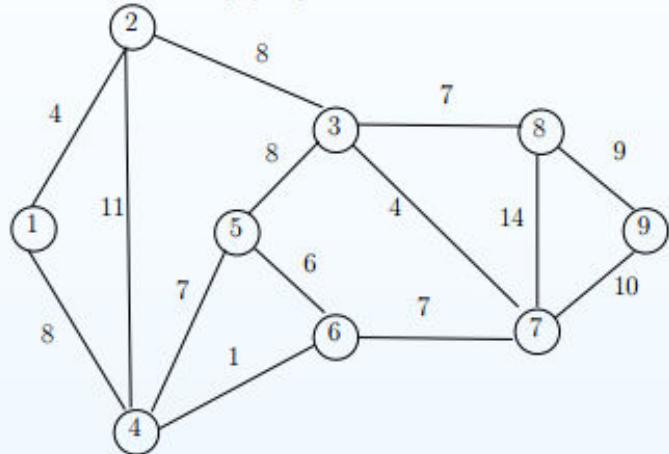
ΦΑΣΗ I

$$\begin{aligned}v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{2, \dots, 9\}\} \\&= \operatorname{argmin}\{4, \infty, 8, \infty, \infty, \infty, \infty\} = 2, \\S &\leftarrow S \cup \{1\} = \{1\} \cup \{2\} = \{1, 2\}\end{aligned}$$

ΦΑΣΗ II

$$\begin{aligned}d_3 &\leftarrow \min\{d_3, d_2 + w(2, 3)\} = \min\{\infty, 4 + 8\} = 12, \quad p_3 \leftarrow 2, \\d_4 &\leftarrow \min\{d_4, d_2 + w(2, 4)\} = \min\{8, 4 + 11\} = 8.\end{aligned}$$

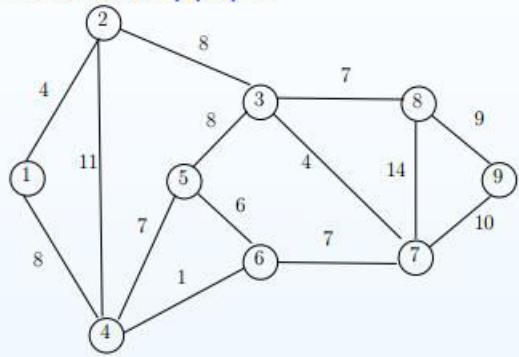
Επανάληψη 2



Έξοδος Επανάληψης

$$\begin{aligned}d &= [0, 4, 12, 8, \infty, \infty, \infty, \infty, \infty] \\p &= [, 1, 2, 1, , , , ,] \\S &= \{1, 2\}\end{aligned}$$

Επανάληψη 3



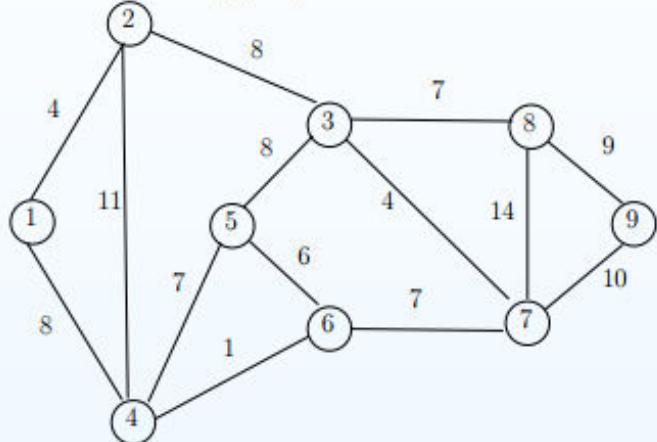
ΦΑΣΗ I

$$\begin{aligned}v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{3, \dots, 9\}\} \\&= \operatorname{argmin}\{12, 8, \infty, \infty, \infty, \infty, \infty\} = 4, \\S &\leftarrow S \cup \{4\} = \{1, 2\} \cup \{4\} = \{1, 2, 4\}\end{aligned}$$

ΦΑΣΗ II

$$\begin{aligned}d_5 &\leftarrow \min\{d_5, d_4 + w(4, 5)\} = \min\{\infty, 8 + 7\} = 15, \quad p_5 \leftarrow 4, \\d_6 &\leftarrow \min\{d_6, d_4 + w(4, 6)\} = \min\{\infty, 8 + 1\} = 9, \quad p_6 \leftarrow 4.\end{aligned}$$

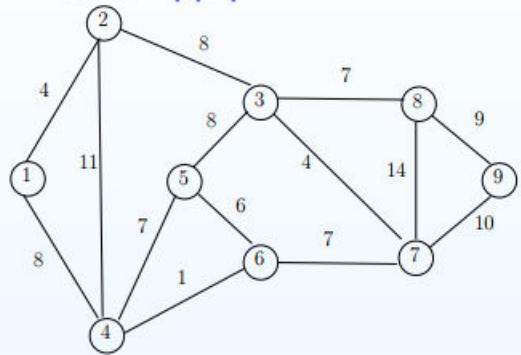
Επανάληψη 3



Έξοδος Επανάληψης

$$\begin{aligned}d &= [\textcolor{red}{0}, \textcolor{red}{4}, 12, \textcolor{red}{8}, 15, 9, \infty, \infty, \infty] \\p &= [\quad, 1, 2, 1, 4, 4, \quad, \quad, \quad] \\S &= \{1, 2, 4\}\end{aligned}$$

Επανάληψη 4



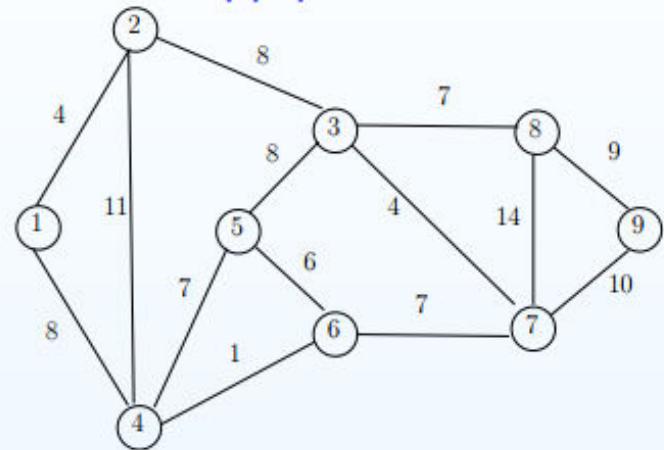
ΦΑΣΗ I

$$\begin{aligned}v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{5, \dots, 9\} \cup \{3\}\} \\&= \operatorname{argmin}\{15, 9, \infty, \infty, \infty, 12\} = 6, \\S &\leftarrow S \cup \{6\} = \{1, 2, 4\} \cup \{6\} = \{1, 2, 4, 6\}\end{aligned}$$

ΦΑΣΗ II

$$\begin{aligned}d_5 &\leftarrow \min\{d_5, d_6 + w(6, 5)\} = \min\{15, 9 + 6\} = 15, \\d_7 &\leftarrow \min\{d_7, d_6 + w(6, 7)\} = \min\{\infty, 9 + 7\} = 16, \quad p_7 \leftarrow 6.\end{aligned}$$

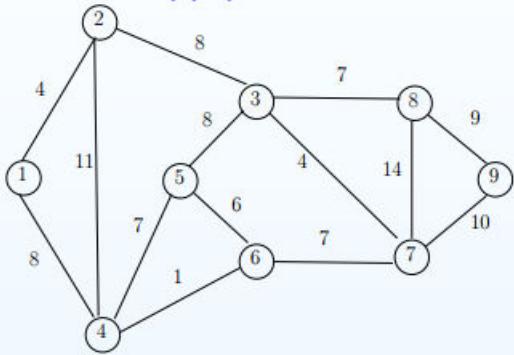
Επανάληψη 4



Έξοδος Επανάληψης

$$\begin{aligned}d &= [0, 4, 12, 8, 15, 9, 16, \infty, \infty] \\p &= [, 1, 2, 1, 4, 4, 6, ,] \\S &= \{1, 2, 4, 6\}\end{aligned}$$

Επανάληψη 5



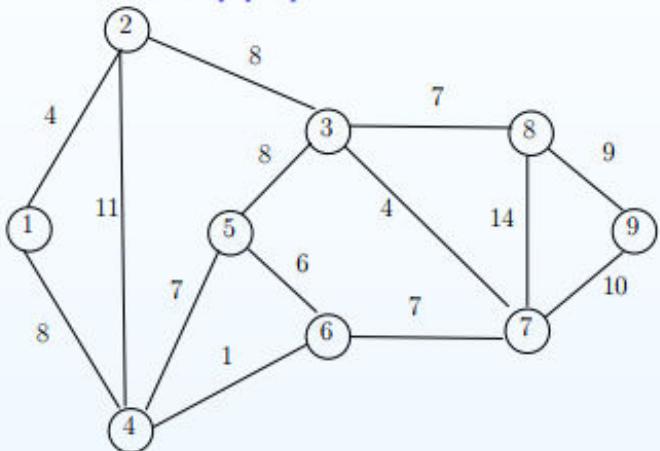
ΦΑΣΗ I

$$\begin{aligned}
 v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{3, 5, 7, 8, 9\}\} \\
 &= \operatorname{argmin}\{12, 15, 16, \infty, \infty\} = 3, \\
 S &\leftarrow S \cup \{3\} = \{1, 2, 4, 6\} \cup \{3\} = \{1, 2, 3, 4, 6\}
 \end{aligned}$$

ΦΑΣΗ II

$$\begin{aligned}
 d_5 &\leftarrow \min\{d_5, d_3 + w(3, 5)\} = \min\{15, 12 + 8\} = 15, \\
 d_7 &\leftarrow \min\{d_7, d_3 + w(7, 3)\} = \min\{16, 12 + 4\} = 16, \\
 \text{iwa.gr } d_8 &\leftarrow \min\{d_8, d_3 + w(3, 8)\} = \min\{\infty, 12 + 7\} = 19 \quad p_8 \xleftarrow{14} 3.
 \end{aligned}$$

Επανάληψη 5



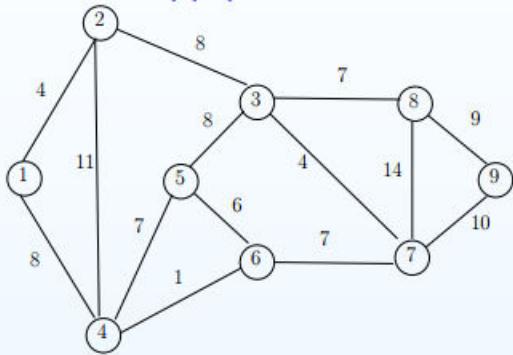
Έξοδος Επανάληψης

$$d = [0, 4, 12, 8, 15, 9, 16, 19, \infty]$$

$$p = [, 1, 2, 1, 4, 4, 6, 3,]$$

$$S = \{1, 2, 4, 6\}$$

Επανάληψη 6

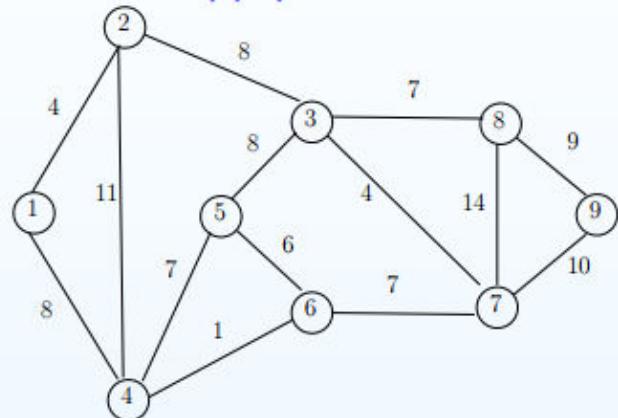


ΦΑΣΗ Ι

$$\begin{aligned}v^* &\leftarrow \operatorname{argmin}\{d_v : v \in \{5, 7, 8, 9\}\} \\&= \operatorname{argmin}\{15, 16, 19, \infty\} = 5, \\S &\leftarrow S \cup \{5\} = \{1, 2, 3, 4, 6\} \cup \{5\} = \{1, 2, 3, 4, 5, 6\}\end{aligned}$$

ΦΑΣΗ ΙΙ

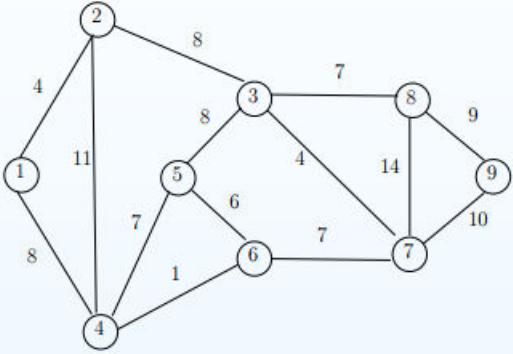
Επανάληψη 6



Έξοδος Επανάληψης

$$\begin{aligned}d &= [0, 4, 12, 8, 15, 9, 16, 19, \infty] \\p &= [, 1, 2, 1, 4, 4, 6, 3,] \\S &= \{1, 2, 3, 4, 5, 6\}\end{aligned}$$

Επανάληψη 7



ΦΑΣΗ Ι

$$v^* \leftarrow \operatorname{argmin}\{d_v : v \in \{7, 8, 9\}\}$$

$$= \operatorname{argmin}\{16, 19, \infty\} = 7,$$

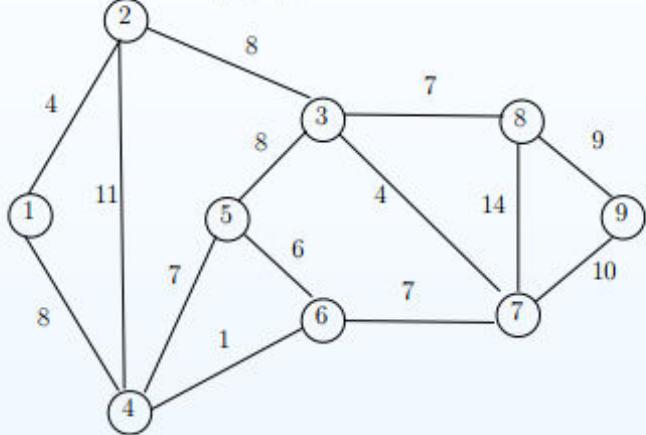
$$S \leftarrow S \cup \{7\} = \{1, 2, 3, 4, 5, 6\} \cup \{7\} = \{1, 2, 3, 4, 5, 6, 7\}$$

ΦΑΣΗ II

$$d_8 \leftarrow \min\{d_8, d_7 + w(7, 8)\} = \min\{19, 16 + 14\} = 19$$

$$d_9 \leftarrow \min\{d_9, d_7 + w(7, 9)\} = \min\{\infty, 16 + 10\} = 26 \quad p_9 \leftarrow 7.$$

Επανάληψη 7



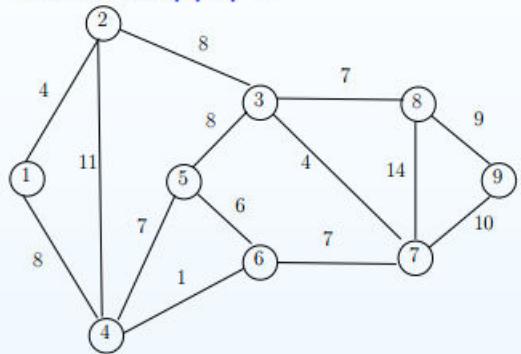
Έξοδος Επανάληψης

$$d = [0, 4, 12, 8, 15, 9, 16, 19, 26]$$

$$p = [\quad, 1, 2, 1, 4, 4, 6, 3, 7]$$

$$S = \{1, 2, 3, 4, 5, 6, 7\}$$

Επανάληψη 8



ΦΑΣΗ I

$$v^* \leftarrow \operatorname{argmin}\{d_v : v \in \{8, 9\}\}$$

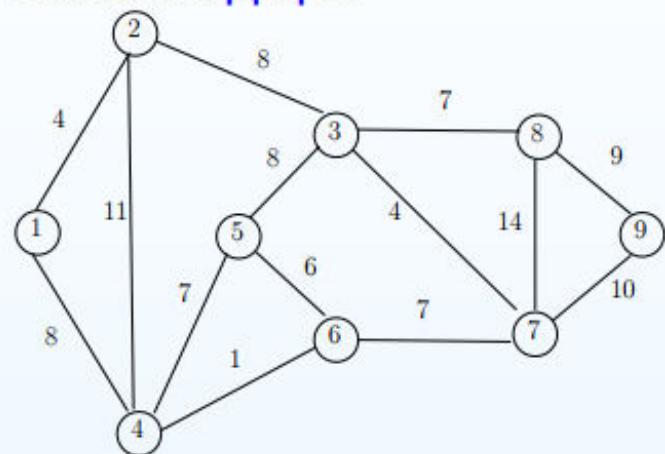
$$= \operatorname{argmin}\{19, 26\} = 8,$$

$$S \leftarrow S \cup \{8\} = \{1, 2, 3, 4, 5, 6, 7\} \cup \{8\} = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

ΦΑΣΗ II

$$d_9 \leftarrow \min\{d_9, d_8 + w(8, 9)\} = \min\{26, 19 + 9\} = 26.$$

Επανάληψη 8



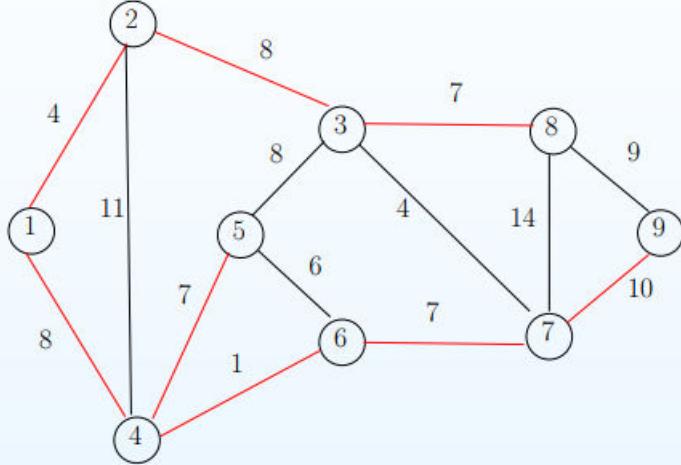
Έξοδος Επανάληψης

$$d = [0, 4, 12, 8, 15, 9, 16, 19, 26]$$

$$p = [, 1, 2, 1, 4, 4, 6, 3, 7]$$

$$S = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

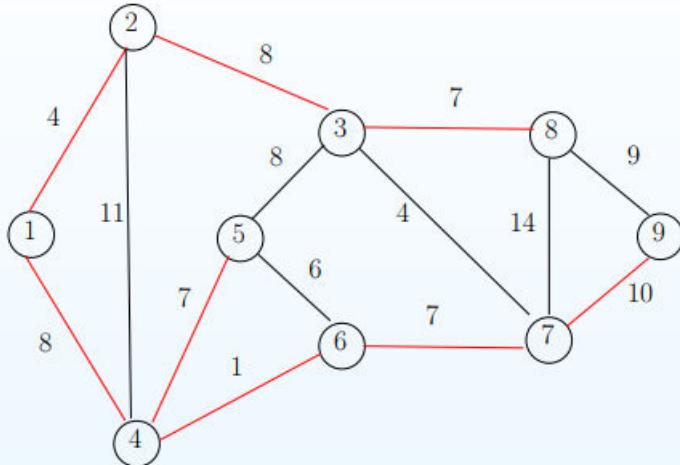
Λύση



$$d = [0, 4, 12, 8, 15, 9, 16, 19, 26]$$

$$p = [\quad, 1, 2, 1, 4, 4, 6, 3, 7]$$

Παρατηρήσεις



Ο Αλγόριθμος του Dijkstra υπολογίζει το συντομότερο μονοπάτι από την αφετηρία προς κάθε άλλο κόμβο.

Επομένως, παράγονται $n - 1$ μονοπάτια τα οποία συνιστούν ένα δένδρο συντομότερων μονοπατιών (ΔΣΜ) (λόγω της ιδιότητας της βέλτιστης υποδομής). Η πληροφορία αυτή βρίσκεται αποθηκευμένη στον πίνακα p . Στο συγκεκριμένο παράδειγμα έχουμε

$$p = [\quad, 1, 2, 1, 4, 4, 6, 3, 7].$$

Ορθότητα

Πρόταση 1 Ο Αλγόριθμος του Dijkstra υπολογίζει σωστά το ελάχιστο μονοπάτι από την αφετηρία (κορυφή s) προς κάθε άλλη κορυφή του γραφήματος

Απόδειξη Θα συμβολίσουμε με $d(s, v)$ το μήκος του συντομότερου μονοπατιού από την κορυφή s ως την κορυφή v , για κάθε $v \in V$. Για να δείξουμε το ζητούμενο, θα χρησιμοποιήσουμε επαγωγή.

Βάση Επαγωγής: $|S| = 0$, $d_s = 0 = d(s, s)$, $S \leftarrow \{s\}$.

Επαγωγικό Βήμα: Στην αρχή μίας τυχαίας επανάληψης, θα υποθέσουμε ότι για κάθε κορυφή $v \in S$ ισχύει ότι $d(s, v) = d_v$. Θα δείξουμε ότι $d(s, u) = d_u$, όπου u η κορυφή που επιλέγει ο Αλγόριθμος να κάνει μόνιμη στην παρούσα επανάληψη, ήτοι

$$u = v^* = \operatorname{argmin}\{d_v : v \in V \setminus S\}. \quad (3)$$

Έστω ότι, αντίθετα με το ζητούμενο,

$$d(s, u) < d_u. \quad (4)$$

Επομένως, υπάρχει συντομότερο μονοπάτι P_{su} από την κορυφή s στη u από αυτό που έχει ανακαλύψει ο Αλγόριθμος.

Έστω x η τελευταία κορυφή του P_{su} που ανήκει στο S και y η αμέσως επόμενη κορυφή και επομένως $y \in V \setminus S$.

Από την επαγωγική υπόθεση $d(s, x) = d_x$ και επειδή οι συντελεστές των ακμών είναι μη-αρνητικοί αριθμοί θα πρέπει

$$d_x + w(x, y) \leq d(s, u). \quad (5)$$

Όμως θα πρέπει

$$d_y \leq d_x + w(x, y) \quad (6)$$

αφού στην (όποια προηγούμενη) επανάληψη που η κορυφή x έγινε μόνιμη το d_y υπολογίστηκε από τον τύπο

$$d_y = \min\{d_y, d_x + w(x, y)\}$$

Από τις (4), (5), (6), έχουμε $d_y < d_u$. Αυτό όμως αποτελεί αντίφαση στην επιλογή του u από τον τύπο (3).

Πολυπλοκότητα

Έστω $[V] = n$.

Εξετάζουμε την k -ιοστή επανάληψη.

ΦΑΣΗ Ι

$|S| = k - 1$ και επομένως $|V \setminus S| = n - k + 1$. Ο Αλγόριθμος αναζητάει ανάμεσα σε $n - k + 1$ στοιχεία να βρει το μικρότερο. Άρα ο αριθμός των συγκρίσεων προκειμένου να βρεθεί το v^* είναι $a(k) = n - k$.

ΦΑΣΗ ΙΙ

Εφόσον $S \leftarrow S \cup \{v^*\}$ τώρα $|S| = k$ και επομένως $|V \setminus S| = n - k$. Ο Αλγόριθμος εκτελεί προσθέσεις και συγκρίσεις στη φάση αυτή. Για κάθε κορυφή $v \in N(v^*)$ εκτελείται μία πρόσθεση και μία σύγκριση. Ο αριθμός των συγκρίσεων $b(k)$ είναι ίσος με τον αριθμό των προσθέσεων $c(k)$ και φράζεται από τα επάνω από την ποσότητα $n - k$ - το φράγμα επιτυγχάνεται όταν $N(v^*) = V \setminus S$.

Ο αριθμός των στοιχειωδών πράξεων στην επανάληψη k ικανοποιεί τη σχέση

$$a(k) + b(k) + c(k) \leq 3(n - k). \quad (7)$$

Αθροίζοντας για όλες τις τιμές του k παίρνουμε το συνολικό αριθμό των στοιχειωδών πράξεων, ήτοι

$$\begin{aligned} T(n) &\leq \sum_{k=1}^{n-1} 3(n - k) = 3 \left(\sum_{k=1}^{n-1} n - \sum_{k=1}^{n-1} k \right) \\ &= 3 \left(n(n - 1) - \frac{n(n - 1)}{2} \right) = \frac{3}{2} n(n - 1) \Rightarrow \\ T(n) &= O(n^2). \end{aligned}$$

Bellman-Ford (Εύρεση ΜΠΣΔ σε Κατευθυνόμενα Γραφήματα με αρνητικά βάρη)

Αρνητικά βάρη

Ο Αλγόριθμος του Dijkstra μπορεί να εφαρμοστεί και σε κατευθυνόμενα γραφήματα αρκεί τα βάρη των ακμών να είναι μη-αρνητικοί αριθμοί. Στην περίπτωση αυτή υπολογίζονται το συντομότερα κατευθυνόμενα μονοπάτια από την αφετηρία προς όλες τις άλλες ακμές.

Αν υπάρχουν αρνητικά βάρη σε κάποιες από τις ακμές ΣΤΗΝ ΠΕΡΙΠΤΩΣΗ ΤΩΝ ΚΑΤΕΥΘΥΝΟΜΕΝΩΝ ΓΡΑΦΗΜΑΤΩΝ χρησιμοποιείται ο Αλγόριθμος των Bellman-Ford.

Σημειώνουμε ότι αν στο γράφημα υπάρχουν αρνητικά βάρη στις ακμές, τότε μπορεί να υπάρχουν και αρνητικοί κύκλοι. Σε αυτή την περίπτωση το πρόβλημα του συντομότερου μονοπατιού δεν είναι καλά ορισμένο: όσες περισσότερες φορές διασχίσουμε τον κύκλο μικραίνει η απόσταση.

Ο Αλγόριθμος των Bellman-Ford εντοπίζει και την ύπαρξη αρνητικού κύκλου.

Ιδέα

Για τη συνέχεια η αναφορά σε μονοπάτια ή κύκλους υπονοεί την έννοια ‘κατευθυνόμενα’ αφού όλα τα γραφήματα είναι κατευθυνόμενα.

- $d(s, v)$: το μήκος του συντομότερου μονοπατιού από την αφετηρία s μέχρι την κορυφή v .
- d_v^k : το μήκος του συντομότερου μονοπατιού από την αφετηρία s μέχρι την κορυφή v το οποίο αποτελείται από ΤΟ ΠΟΛΥ k ακμές.
- $d_v^k \leq d_v^{k-1}$ και $d_v^{|V|-1} = d(s, v)$.

Μπορούμε να υπολογίσουμε το d_v^k αν γνωρίζουμε τα d_u^{k-1} για κάθε κορυφή $u \in N^-(v) \cup \{v\}$ από τον ακόλουθο τύπο

$$d_v^k = \min\{d_v^{k-1}, \min\{d_u^{k-1} + w(u, v) : u \in N^-(v)\}\} \quad (8)$$

Ο επόμενος αλγόριθμος βασίζεται σε αυτές τις ιδέες.

Αλγόριθμος

```
1: void Bellman-Ford( $G(V, E, w)$ ,  $s$ )
2: for all  $v \in V$  do
3:    $d_v^0 \leftarrow \infty$ ;  $p_v^0 \leftarrow \text{NULL}$ ;
4: end for
5:  $d_s^0 \leftarrow 0$ ;
6: for  $k \leftarrow 1$ ;  $k \leq |V| - 1$ ;  $k++$  do
7:   for all  $v \in V$  do
8:      $d_v^k \leftarrow d_v^{k-1}$ ;  $p_v^k \leftarrow p_v^{k-1}$ ;
9:   end for
10:  for  $(u, v) \in E$  do
11:    if  $d_v^k > d_u^{k-1} + w(u, v)$  then
12:       $d_v^k \leftarrow d_u^{k-1} + w(u, v)$ ;
13:       $p_v^k \leftarrow u$ ;
14:    end if
15:  end for
16: end for
    ///////////////////////////////////////////////////////////////////check for a negative cycle
for  $(u, v) \in E$  do
  if  $d_v^{|V|-1} > d_u^{|V|-1} + w(u, v)$  then
    Print "Negative Cycle"
  end if
end for
```

Παρατήρηση: Οι γραμμές 7-15 υλοποιούν την (8) για κάθε $v \in V \setminus \{s\}$. Οι γραμμές 18-22 ελέγχουν την ύπαρξη αρνητικού κύκλου.

Απόδειξη

Πρόταση 2 Αν το κατευθυνόμενο γράφημα $G(V, E)$ δεν περιέχει αρνητικό κύκλο ο αλγόριθμος των Bellman-Ford υπολογίζει την ελάχιστη απόσταση από την αφετηρία προς κάθε άλλη κορυφή.

Απόδειξη Όπως και στην προηγούμενη περίπτωση $d(s, v)$ συμβολίζει το μήκος του συντομότερου μονοπατιού από την κορυφή s ως την κορυφή v , για κάθε $v \in V$. Με τη χρήση επαγωγής, στην τιμή του k , θα δείξουμε ότι d_v^k θα περιέχει το μήκος του μικρότερο μονοπατιού από την κορυφή s στην κορυφή v με αριθμό ακμών $\leq k$.

Βάση Επαγωγής: Για $k = 0$, ο αλγόριθμος επιστρέφει το σωστό αποτέλεσμα αφού

$$d_s^0 = 0 = d(s, s), d_v^0 = \infty, \forall v \in V \setminus \{s\}.$$

Επαγωγικό Βήμα: Έστω ότι για κάθε κορυφή u , d_u^{k-1} είναι το μήκος του συντομότερου μονοπατιού από την s στη u , χρησιμοποιώντας το πολύ $k - 1$ ακμές (επαγωγική υπόθεση). Έστω P το συντομότερο μονοπάτι από την s στην v , με $\leq k$ ακμές και μήκος $w(P)$. Έστω u η προτελευταία κορυφή στο μονοπάτι αυτό. Λόγω της ιδιότητας της βέλτιστης υποδομής το μονοπάτι αυτό περιέχει το συντομότερο μονοπάτι από την s στη u , έστω Q . Το μονοπάτι Q περιέχει $\leq k - 1$ κορυφές και άρα $\leq k - 1$ ακμές. Έστω $w(Q)$ το μήκος του μονοπατιού αυτού. Από την επαγωγική υπόθεση

$$w(Q) = d_u^{k-1}. \quad (9)$$

Επίσης από κατασκευής ισχύει ότι

$$w(P) = w(Q) + w(u, v). \quad (10)$$

Στην επανάληψη k επανυπολογίζουμε την απόσταση d_v^k από την (8) η οποία γράφεται

$$d_v^k = \min\{d_v^{k-1}, d_u^{k-1} + w(u, v)\}. \quad (11)$$

Άρα $d_v^k \leq d_u^{k-1} + w(u, v)$ η οποία σε συνδυασμό με (9) και στη συνέχεια με (10) παράγει

$$d_v^k \leq w(Q) + w(u, v) \Rightarrow d_v^k \leq w(P) \quad (12)$$

Από την επαγωγική υπόθεση, το d_v^{k-1} είναι το μήκος του συντομότερου μονοπατιού από την s στη v , το οποίο περιέχει το πολύ $k - 1$ ακμές. Άρα το μήκος αυτό θα είναι τουλάχιστον ίσο με το $w(P)$ αφού το μονοπάτι P μπορεί να χρησιμοποιεί παραπάνω ακμές. Δηλαδή $w(P) \leq d_v^{k-1}$.

Από την τελευταία παρατήρηση και την (11), έχουμε

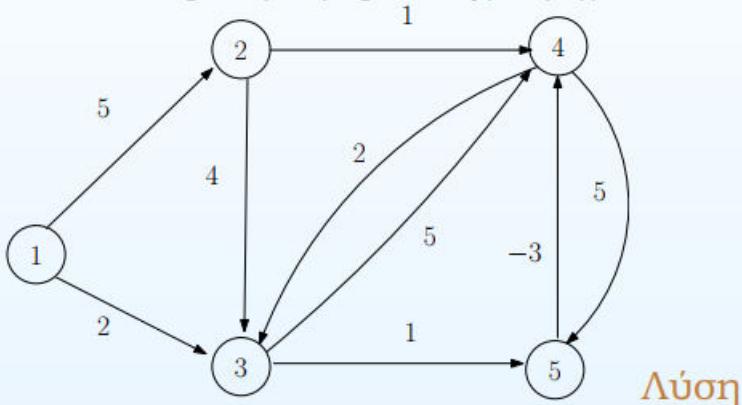
$$d_v^k = w(P).$$

Πολυπλοκότητα

Πρόταση 3 Ο αλγόριθμος Bellman-Ford εκτελεί $O(|V| \cdot |E|)$ βήματα.

Παράδειγμα

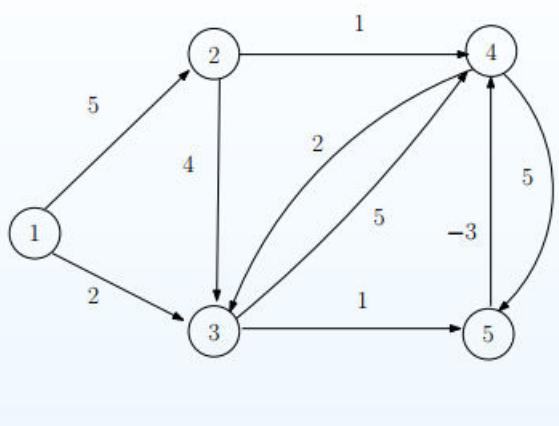
Με τη χρήση του αλγόριθμου των Bellman-Ford να βρεθούν τα συνομότερα μονοπάτια από την κορυφή 1 προς τις υπόλοιπες κορυφές στο γράφημα



Αρχικοποίηση

$$d = [0, \infty, \infty, \infty, \infty]$$

$$p = [, , , ,]$$



$$d^0 = [0, \infty, \infty, \infty, \infty]$$

$$p^0 = [, , , ,]$$

Επανάληψη $k = 1$

$$d_1^1 = d_1^0 = 0,$$

$$d_2^1 = \min\{d_2^0, d_1^0 + w(1, 2)\} = \min\{\infty, 0 + 5\} = 5, \quad p_2^1 = 1,$$

$$d_3^1 = \min\{d_3^0, d_1^0 + w(1, 3), d_2^0 + w(2, 3), d_4^0 + w(4, 3)\}$$

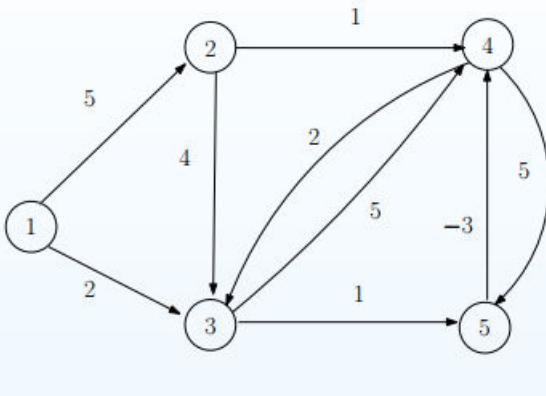
$$= \min\{\infty, 0 + 2, \infty + 4, \infty + 2\} = 2, \quad p_3^1 = 1,$$

$$d_4^1 = \min\{d_4^0, d_2^0 + w(2, 4), d_3^0 + w(3, 4), d_5^0 + w(5, 4)\}$$

$$= \min\{\infty, \infty + 1, \infty + 5, \infty - 3\} = \infty,$$

$$d_5^1 = \min\{d_5^0, d_3^0 + w(3, 5), d_4^0 + w(4, 5)\}$$

$$= \min\{\infty, \infty + 1, \infty + 5, \infty\} = \infty$$



$$d^1 = [0, 5, 2, \infty, \infty]$$

$$p^1 = [, 1, 1, ,]$$

Επανάληψη $k = 2$

$$d_1^2 = d_1^1 = 0,$$

$$d_2^2 = \min\{d_2^1, d_1^1 + w(1, 2)\} = \min\{5, 0 + 5\} = 5,$$

$$d_3^2 = \min\{d_3^1, d_1^1 + w(1, 3), d_2^1 + w(2, 3), d_4^1 + w(4, 3)\}$$

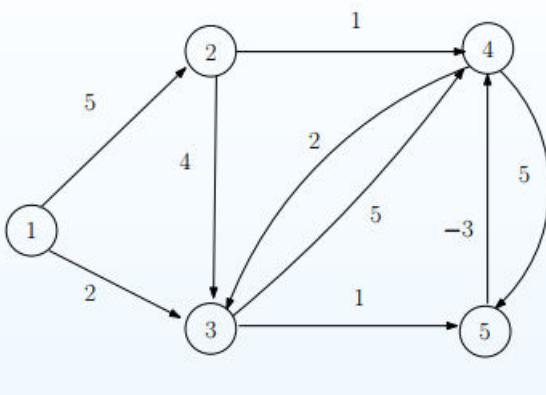
$$= \min\{2, 0 + 2, 5 + 4, \infty + 2\} = 2,$$

$$d_4^2 = \min\{d_4^1, d_2^1 + w(2, 4), d_3^1 + w(3, 4), d_5^1 + w(5, 4)\}$$

$$= \min\{\infty, 5 + 1, 2 + 5, \infty - 3\} = 6, \quad p_4^2 = 2,$$

$$d_5^2 = \min\{d_5^1, d_3^1 + w(3, 5), d_4^1 + w(4, 5)\}$$

a.gr = min{∞, 2 + 1, ∞ + 5, } = 3 p₅² = 3 2



$$d^2 = [0, 5, 2, 6, 3]$$

$$p^2 = [, 1, 1, 2, 3]$$

Επανάληψη $k = 3$

$$d_1^3 = d_1^2 = 0,$$

$$d_2^3 = \min\{d_2^2, d_1^2 + w(1, 2)\} = \min\{5, 0 + 5\} = 5,$$

$$d_3^3 = \min\{d_3^2, d_1^2 + w(1, 3), d_2^2 + w(2, 3), d_4^2 + w(4, 3)\}$$

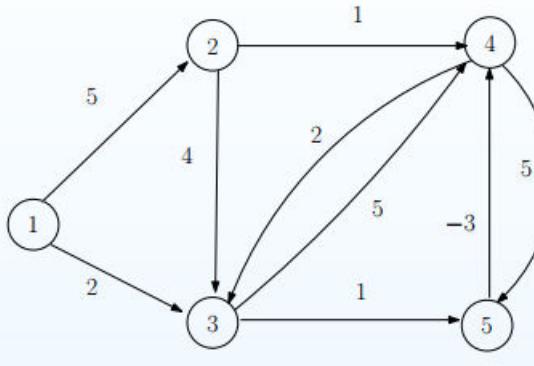
$$= \min\{2, 0 + 2, 5 + 4, 6 + 2\} = 2,$$

$$d_4^3 = \min\{d_4^2, d_2^2 + w(2, 4), d_3^2 + w(3, 4), d_5^2 + w(5, 4)\}$$

$$= \min\{6, 5 + 1, 2 + 5, 3 - 3\} = 0, \quad p_4^3 = 5,$$

$$d_5^3 = \min\{d_5^2, d_3^2 + w(3, 5), d_4^2 + w(4, 5)\}$$

wa.gr = min{3, 2 + 1, 6 + 5, } = 3 29



$$d^3 = [0, 5, 2, 0, 3]$$

$$p^3 = [, 1, 1, 5, 3]$$

Επανάληψη $k = 4$

$$d_1^4 = d_1^3 = 0,$$

$$d_2^4 = \min\{d_2^3, d_1^3 + w(1, 2)\} = \min\{5, 0 + 5\} = 5,$$

$$\begin{aligned} d_3^4 &= \min\{d_3^3, d_1^3 + w(1, 3), d_2^3 + w(2, 3), d_4^3 + w(4, 3)\} \\ &= \min\{2, 0 + 2, 5 + 4, 0 + 2\} = 2, \end{aligned}$$

$$\begin{aligned} d_4^4 &= \min\{d_4^3, d_2^3 + w(2, 4), d_3^3 + w(3, 4), d_5^3 + w(5, 4)\} \\ &= \min\{0, 5 + 1, 2 + 5, 3 - 3\} = 0, \end{aligned}$$

$$d_5^4 = \min\{d_5^3, d_3^3 + w(3, 5), d_4^3 + w(4, 5)\}$$

$$\underline{\underline{m}} = \min\{3, 2 + 1, 0 + 5, \} = 3 \quad \underline{\underline{m}}$$

Output

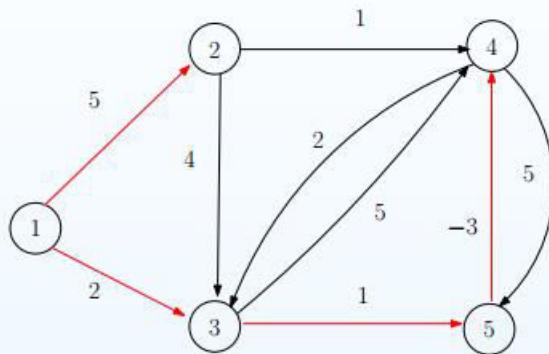
$$d = [0, 5, 2, 0, 3]$$

$$p = \begin{bmatrix} 1 & 1 \\ 1 & 1 & 2 & 3 \\ 1 & 1 & 5 & 3 \\ 1 & 1 & 5 & 3 \end{bmatrix}$$

Παρατήρηση: Στην έξοδο εμφανίζονται (υπό μορφή πίνακα p) όλα τα διανύσματα p^k για κάθε $k = 1, \dots, |V| - 1$.

Χρειαζόμαστε όλη την πληροφορία αυτή για να εντοπίσουμε τα συντομότερα μονοπάτια από την αφετηρία προς κάθε άλλη κορυφή. Για παράδειγμα, για να βρούμε τη συντομότερη διαδρομή από την κορυφή 1 στην 4 ξεκινώντας από την τελευταία κορυφή έχουμε

$$4 \leftarrow p_4^4 = 5 \leftarrow p_5^3 = 3 \leftarrow p_3^2 = 1 \leftarrow p_1^1 =$$



Σχήμα 3: Παράδειγμα Bellman-Ford : Δένδρο συντομότερων διαδρομών

Παρατήρηση

- Μπορούμε να τερματίσουμε τον αλγόριθμο αν σε δύο διαδοχικές επαναλήψεις δεν μεταβληθεί το διάνυσμα των συντομότερων αποστάσεων (d).
- Το πρόβλημα εύρεσης των συντομότερων μονοπατιών κοινής αφετηρίας σε ένα μη-κατευθυνόμενο γράφημα $G(V, E, w)$ με μη-αρνητικά βάρη στις ακμές μπορεί να λυθεί με τον αλγόριθμο των Bellman-Ford ως εξής: κατασκευάζουμε το κατευθυνόμενο γράφημα $G'(V, E', w)$, όπου για κάθε μη κατευθυνόμενη ακμή $\{v, u\} \in E$ εισάγουμε ακμές $(v, u) \in E'$ και $(u, v) \in E'$ με $w((v, u)) = w((u, v)) = w(\{v, u\})$.
- Δυστυχώς ο μετασχηματισμός αυτός παράγει ένα κατευθυνόμενο γράφημα με αρνητικό κύκλο αν το βάρος κάποιας ακμής είναι αρνητικός αριθμός. Στην περίπτωση αυτή το πρόβλημα μπορεί να επιλυθεί μέσω ενός ‘τέλειου ταιριάσματος’.

DAG (Εύρεση ΜΠΣΔ σε Κατευθυνόμενα Άκυκλα Γράφηματα)

ΚΑΓ

Το πρόβλημα ΜΠΣΔ απλοποιείται σημαντικά αν το γράφημα δεν έχει κύκλο - στην περίπτωση αυτή το γράφημα ονομάζεται Κατευθυνόμενο Άκυκλο Γράφημα (ΚΑΓ).

Βασική Ιδέα

Έστω ότι έχουμε αριθμίσει τις κορυφές του γραφήματος με τους αριθμούς $1, 2, 3, \dots, n$ κατά τρόπο ώστε να υπάρχει ακμή από μία κορυφή i σε μία κορυφή j ανν $i < j$ (στην παραπάνω αρίθμηση.) Χώρις βλάβη της γενικότητας θεωρούμε ότι η κορυφή πηγή (s) έχει αρίθμηση 1.

Τα μήκη των συντομότερων διαδρομών από την κορυφή 1 προς τις υπόλοιπες κορυφές δύνονται από τη σχέση

$$d_v = \begin{cases} 0, & v = 1, \\ \min\{d_j + w(j, v) : j < v, j \in N^-(v)\}, & v \in \{2, \dots, |V|\} \end{cases}$$

Αλγόριθμος

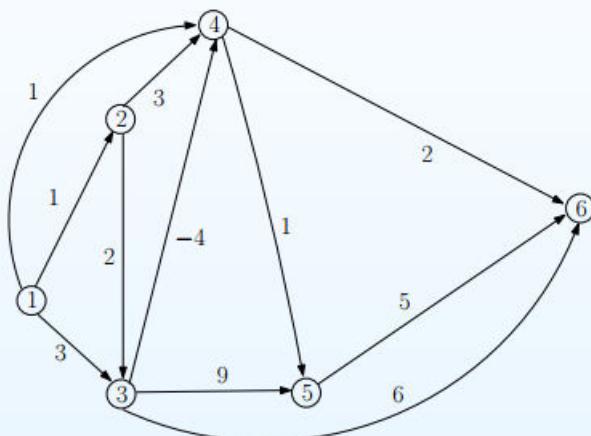
Ο Αλγόριθμος που ακολουθεί υπολογίζει τις συντομότερες διαδρομές σε ένα ΚΑΓ από την κορυφή 1.

```
1: void DAG( $G(V, E, w)$ )
2: for all  $v \leftarrow 1; v \leq |V|; v++ \text{ do}$ 
3:    $d_v \leftarrow \infty; p_v \leftarrow \text{NULL};$ 
4: end for
5:  $d_1 \leftarrow 0;$ 
6: for  $v \leftarrow 2; v \leq |V|; v++ \text{ do}$ 
7:    $d_v \leftarrow \min\{d_j + w(j, v) : j < v, j \in N^-(v)\};$ 
8:    $p_v \leftarrow \operatorname{argmin}\{d_j + w(j, v) : j < v, j \in N^-(v)\};$ 
9: end for
```

Πρόταση 4 Η επίλυση του προβληματος ΜΠΣΔ σε ΚΑΓ επιλύεται σε $O(|V|^2)$ βήματα.

Παράδειγμα

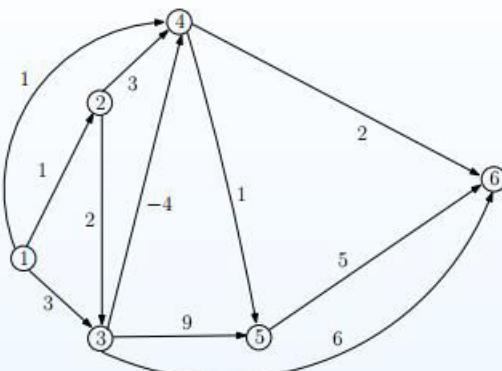
Να βρεθούν οι συντομότερες διαδρομές από την κορυφή 1 προς τις υπόλοιπες κορυφές στο γράφημα του Σχήματος 4.



Σχήμα 4: Βεβαρυμένο ΚΑΓ

Παρατήρηση

Μπορεί να υπάρχουν αρνητικά βάρη στις ακμές από τη στιγμή που το γράφημα δεν περιέχει κύκλους (άρα δεν περιέχει και αρνητικούς κύκλους)



Λύση

$$d_2 = \min\{d_1 + w(1, 2)\} = \min\{0 + 1\} = 1$$

$$d_3 = \min\{d_1 + w(1, 3), d_2 + w(2, 3)\} = \min\{0 + 3, 1 + 2\} = 3$$

$$\begin{aligned} d_4 &= \min\{d_1 + w(1, 4), d_2 + w(2, 4), d_3 + w(3, 4)\} \\ &= \min\{0 + 1, 1 + 3, 3 + (-4)\} = -1 \end{aligned}$$

$$d_5 = \min\{d_3 + w(3, 5), d_4 + w(4, 5)\} = \min\{3 + 9, -1 + 1\} = 0$$

$$d_6 = \min\{d_3 + w(3, 6), d_4 + w(4, 6), d_5 + w(5, 6)\}$$

Prim (Εύρεση ΕΣΔ)**ΕΣΔ**

ΕΣΔ Δεδομένου ενός βεβαρημένου συνδεδεμένου γραφήματος $G(V, E, w)$, να βρεθεί το συνδετικό δένδρο του G να ελαχιστοποιεί το συνολικό βάρος των ακμών που συμμετέχουν σε αυτό.

Δηλαδή αν \mathcal{T} είναι το σύνολο των συνεκτικών δένδρων του G , ζητάμε

$$\min\{w(T) : T \in \mathcal{T}\},$$

$$\text{όπου } w(T) = \sum\{w(e) : e \in E(T)\}.$$

Ιδέα μέθοδος απλησίας - σε κάθε βήμα επιλέγουμε να προσθέσουμε την επόμενη ελάχιστη ακμή που συνδέει δύο σύνολα κορυφών

Ιδιότητες

- Ένα συνδεδεμένο γράφημα G με διακριτά βάρη ακμών έχει ένα μοναδικό ελάχιστο συνδετικό δένδρο.
- Εάν το ελάχιστο βάρος ακμής είναι μοναδικό, τότε η ακμή αυτή περιέχεται σε όλα τα ελάχιστα συνδετικά δένδρα του G .
- Έστω συνδεδεμένο γράφημα G με διακριτά βάρη ακμών, κύκλος C και έστω e η ακμή μέγιστου βάρους του κύκλου. Τότε το ΕΣΔ δεν περιέχει την ακμή e .
- Έστω συνδεδεμένο γράφημα G με βάρη στις ακμές και έστω $U \subset V$. Εάν (u, v) είναι μια ακμή ελάχιστου βάρους έτσι ώστε $u \in U$ και $v \in V \setminus U$, τότε υπάρχει ένα ΕΣΔ το οποίο περιέχει την (u, v) .

Αλγόριθμος

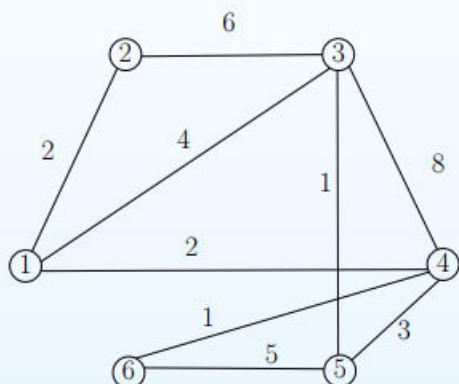
Βασικό Βήμα Έστω $V(T) \subset V$ και $E(T) \subset E$ το σύνολο των κορυφών και ακμών του ΕΣΔ που θέλουμε να υπολογίσουμε. Σε κάθε βήμα οι κορυφές του γραφήματος διαχωρίζονται σε δύο σύνολα: στο σύνολο των κορυφών που ανήκουν στο ΕΣΔ ($V(T)$) και στο σύνολο των υπόλοιπων κορυφών ($V \setminus V(T)$). Σε κάθε βήμα εμπλουτίζεται το σύνολο $V(T)$ προσθέτοντας σε αυτό την κορυφή του συνόλου $V \setminus V(T)$ που συνδέεται με την ‘έλαφρύτερη’ ακμή με τις υπόλοιπες κορυφές του συνόλου $V(T)$.

Ο αλγόριθμος κωδικοποιείται στη συνέχεια. Το σύνολο $V(T)$ αρχικοποιείται από μία οποιαδήποτε κορυφή του V .

```
1:  $V(T) \leftarrow \{v\}; E(T) \leftarrow \emptyset;$ 
2: for  $i \leftarrow 1; i \leq |V| - 1; i++$  do
3:    $(v^*, u^*) \leftarrow \operatorname{argmin}\{w(v, u) : v \in V(T), u \in V \setminus V(T), (v, u) \in E\};$ 
4:    $V(T) \leftarrow V(T) \cup \{u^*\};$ 
5:    $E(T) \leftarrow E(T) \cup \{(v^*, u^*)\};$ 
6: end for
```

Παράδειγμα

Παράδειγμα 5 Με τη χρήση του αλγόριθμου του Prim, να βρεθεί το ΕΣΔ του γραφήματος στου Σχήματος 3



Σχήμα 3: Γράφημα $G(V, E, w)$

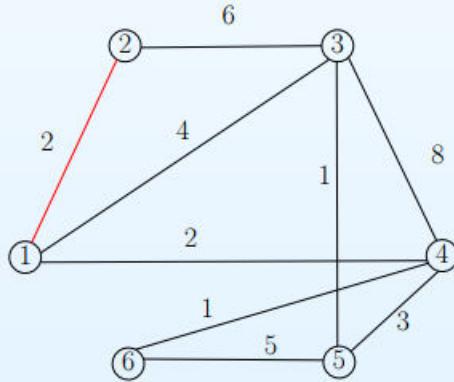
Λύση Αρχικώς έστω $V(T) = \{1\}$

Επανάληψη 1

$$\min\{w(1, 2), w(1, 3), w(1, 4)\} = \min\{2, 4, 2\} = 2.$$

$$V(T) = \{1, 2\}$$

$$E(T) = \{(1, 2)\}$$



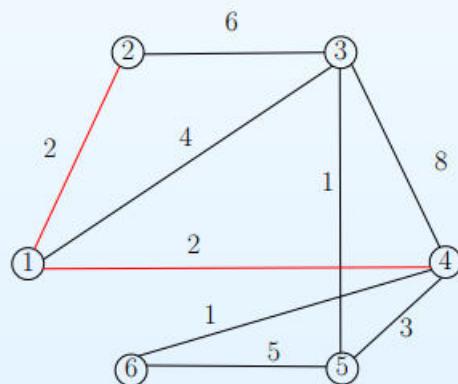
Σχήμα 3: Με κόκκινο οι ακμές του δένδρου

Επανάληψη 2

$$\min\{w(1, 3), w(1, 4), w(2, 3)\} = \min\{4, 2, 6\} = 2.$$

$$V(T) = \{1, 2, 4\}$$

$$E(T) = \{(1, 2), (1, 4)\}$$



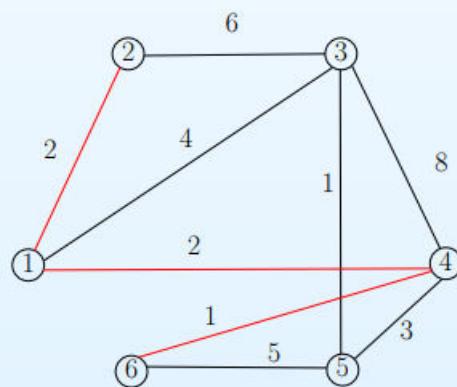
Σχήμα 3: Με κόκκινο οι ακμές του δένδρου

Επανάληψη 3

$$\begin{aligned} & \min\{w(1, 3), w(2, 3), w(4, 3), w(4, 5), w(4, 6)\} \\ &= \min\{4, 6, 8, 3, 1\} = 1. \end{aligned}$$

$$V(T) = \{1, 2, 4, 6\}$$

$$E(T) = \{(1, 2), (1, 4), (4, 6)\}$$



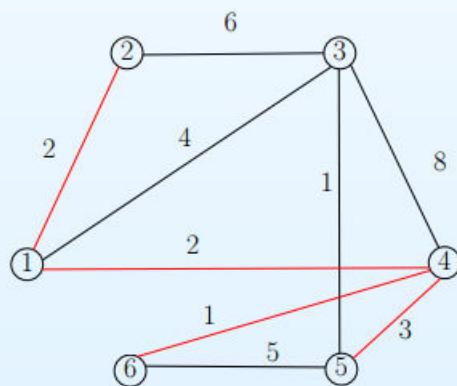
Σχήμα 3: Με κόκκινο οι ακμές του δένδρου

Επανάληψη 4

$$\begin{aligned} & \min\{w(1, 3), w(2, 3), w(4, 3), w(4, 5), w(6, 5)\} \\ &= \min\{4, 6, 8, 3, 5\} = 3. \end{aligned}$$

$$V(T) = \{1, 2, 4, 6, 5\}$$

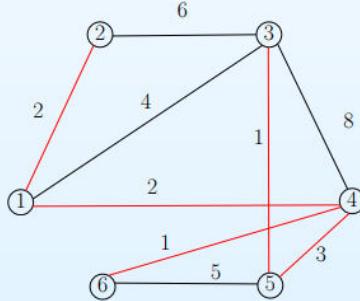
$$E(T) = \{(1, 2), (1, 4), (4, 6), (4, 5)\}$$



Σχήμα 3: Με κόκκινο οι ακμές του δένδρου

Επανάληψη 5

$$\begin{aligned} & \min\{w(1, 3), w(2, 3), w(4, 3), w(5, 3)\} \\ &= \min\{4, 6, 8, 1\} = 1. \\ V(T) &= \{1, 2, 4, 6, 5, 3\} \\ E(T) &= \{(1, 2), (1, 4), (4, 6), (4, 5), (5, 3)\} \end{aligned}$$



Σχήμα 3: Με κόκκινο οι ακμές του δένδρου

Ορθότητα

Πρόταση 6 Έστω T το συνδετικό δένδρο του συνδεδεμένου γραφήματος $G(V, E, w)$ που υπολογίζει ο αλγόριθμος του Prim. Το T είναι ένα ΕΣΔ του G .

Απόδειξη Έστω ότι T^* ένα ΕΣΔ του G . Θα δείξουμε ότι $w(T) = w(T^*)$. Αν $T = T^*$ αυτό ισχύει. Αν $T \neq T^*$ τότε θα υπάρχει (τουλάχιστον) μία ακμή $(u, v) \in E(T) \setminus E(T^*)$. Όταν ο αλγόριθμος πρόσθεσε την ακμή αυτή στο T ήταν η ακμή μικρότερου κόστους από το σύνολο $V(T)$ στο σύνολο $V \setminus V(T)$. Επειδή το T^* είναι συνδετικό δένδρο θα υπάρχει ένα μονοπάτι στο T^* , έστω $P_{u,v}$, από την κορυφή u στην κορυφή v . Στο μονοπάτι αυτό θα πρέπει να υπάρχει ακμή (x, y) τέτοια ώστε $x \in V(T)$, $y \in V \setminus V(T)$. Θα πρέπει

$$w(u, v) \leq w(x, y) \quad (1)$$

(Απόδειξη-συνχ.) Σύμφωνα με το Λήμμα 3 το $\hat{T}^* = T^* + (u, v) - (x, y)$ είναι συνδετικό δένδρο με βάρος

$$w(\hat{T}^*) = w(T^*) + w(u, v) - w(x, y).$$

Από την (1)

$$w(\hat{T}^*) \leq w(T^*) \Rightarrow w(\hat{T}^*) = w(T^*)$$

αφού το \hat{T}^* είναι συνδετικό δένδρο και το T^* είναι ένα ΕΣΔ. Επειδή $|E(T) \setminus E(\hat{T}^*)| = |E(T) \setminus E(T^*)| - 1$, μπορούμε να επαναλάβουμε τη διαδικασία αυτή για κάθε ακμή του συνόλου $E(T) \setminus E(T^*)$ μετατρέποντας το T^* σε T διατηρώντας το ίδιο συνολικό βάρος $w(T^*)$. Έτοιμος $w(T^*) = w(T)$.

Πολυπλοκότητα

Ανάλογα με την υλοποίηση και τη χρήση προχωρημένων δομών δεδομένων ο αλγόριθμος υλοποίεται με πολυπλοκότητες που απεικονίζονται στον Πίνακα 1

Δομή Δεδομένων	Πολυπλοκότητα
λίστα γειτνίασης	$O(V ^2)$
δυαδικός σωρός	$O((V + E) \log V) = O(E \log V)$
σωρός Fibonacci	$O(E + V \log V)$

Πίνακας 1: Αλγόριθμος Prim - Διαφορετικές Υλοποιήσεις

Kruskal (Εύρεση ΕΣΔ)

ΕΣΔ

ΕΣΔ Δεδομένου ενός βεβαρημένου συνδεδεμένου γραφήματος $G(V, E, w)$, να βρεθεί το συνδετικό δένδρο του G να ελαχιστοποιεί το συνολικό βάρος των ακμών που συμμετέχουν σε αυτό.

Δηλαδή αν \mathcal{T} είναι το σύνολο των συνεκτικών δένδρων του G , ζητάμε

$$\min\{w(T) : T \in \mathcal{T}\},$$

όπου $w(T) = \sum\{w(e) : e \in E(T)\}$.

Ιδέα μέθοδος απλησίας - σε κάθε βήμα επιλέγουμε να προσθέσουμε την επόμενη ελάχιστη ακμή που συνδέει δύο σύνολα κορυφών

Ιδιότητες

- Ένα συνδεδεμένο γράφημα G με διακριτά βάρη ακμών έχει ένα μοναδικό ελάχιστο συνδετικό δένδρο.
- Εάν το ελάχιστο βάρος ακμής είναι μοναδικό, τότε η ακμή αυτή περιέχεται σε όλα τα ελάχιστα συνδετικά δένδρα του G .
- Έστω συνδεδεμένο γράφημα G με διακριτά βάρη ακμών, κύκλος C και έστω e η ακμή μέγιστου βάρους του κύκλου. Τότε το ΕΣΔ δεν περιέχει την ακμή e .
- Έστω συνδεδεμένο γράφημα G με βάρη στις ακμές και έστω $U \subset V$. Εάν (u, v) είναι μια ακμή ελάχιστου βάρους έτσι ώστε $u \in U$ και $v \in V \setminus U$, τότε υπάρχει ένα ΕΣΔ το οποίο περιέχει την (u, v) .

Αλγόριθμος Kruskal

Ο αλγόριθμος κτίζει βήμα-βήμα το συνδετικό δένδρο προσθέτοντας ακμές εφαρμόζοντας κατά γράμμα την άπληστη αρχή: σε κάθε βήμα κάνει την καλύτερη επιλογή (επιλέγει την ακμή με το μικρότερο βάρος) χωρίς να παραβιάζει τους περιορισμούς (που αν την προσθέσει στο υποσύνολο των ακμών που έχει δημιουργηθεί στα προηγούμενα βήματα δεν δημιουργείται κύκλος).

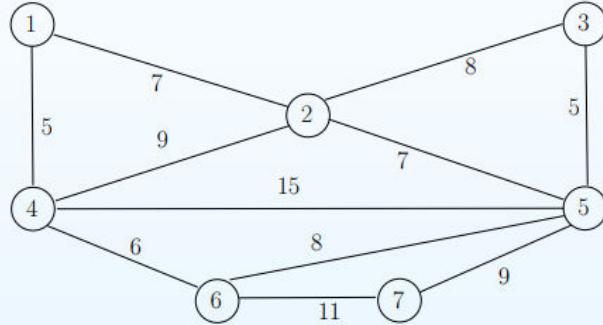
Είσοδος: Συνδεδεμένο γράφημα $G(V, E, w)$

Έξοδος: Ελάχιστο συνδετικό δένδρο T του G

- 1: $m \leftarrow |E|$;
- 2: Διέταξε τις ακμές του $E = \{e_1, e_2, \dots, e_m\}$ ώστε $w(e_1) \leq w(e_2) \leq \dots \leq w(e_m)$.
- 3: $E(T) \leftarrow \emptyset$;
- 4: $i \leftarrow 1$;
- 5: **while** $|E(T)| < |V| - 1$ **do**
- 6: **if** $E(T) \cup \{e_i\}$ δεν περιέχει κύκλο **then**
- 7: $E(T) \leftarrow E(T) \cup \{e_i\}$;
- 8: **end if**
- 9: $i++$;
- 10: **end while**

Παράδειγμα

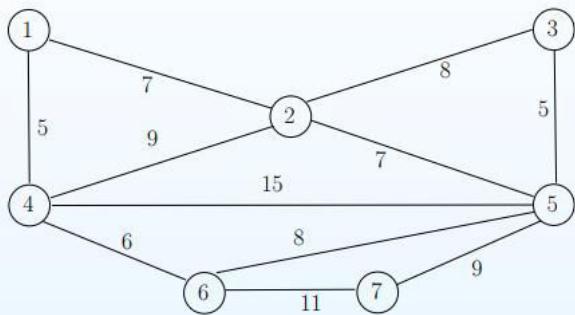
Παράδειγμα 7 Με τη χρήση του αλγόριθμου του Kruskal, να βρεθεί το ΕΣΔ του γραφήματος στου Σχήματος 4



Σχήμα 4: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Παράδειγμα - Επίλυση

Λύση

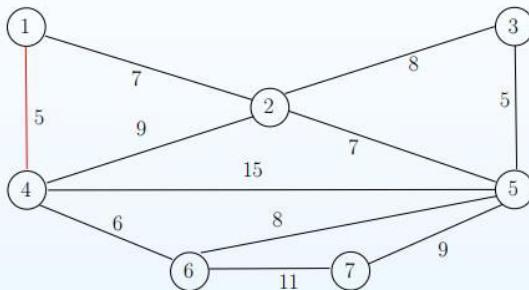


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1,4) - (3,5) - (4,6) - (1,2) - (2,5) - (2,3) \\ & - (6,5) - (2,4) - (5,7) - (6,7) - (4,5) \end{aligned}$$

Επανάληψη 1

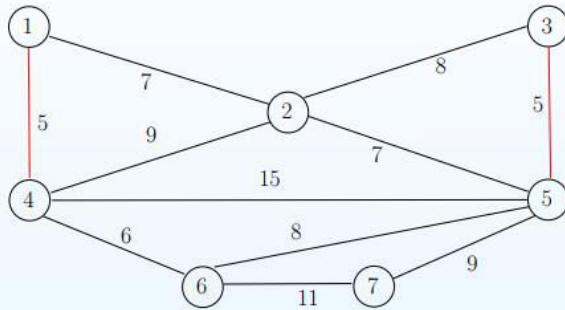


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (\mathbf{1,4}) - (3,5) - (4,6) - (1,2) - (2,5) - (2,3) \\ & - (6,5) - (2,4) - (5,7) - (6,7) - (4,5) \end{aligned}$$

Επανάληψη 2

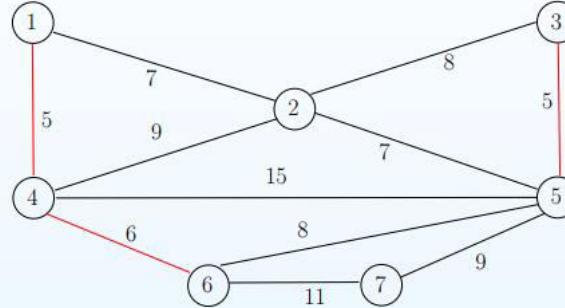


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1, 4) - (3, 5) - (4, 6) - (1, 2) - (2, 5) - (2, 3) \\ & - (6, 5) - (2, 4) - (5, 7) - (6, 7) - (4, 5) \end{aligned}$$

Επανάληψη 3

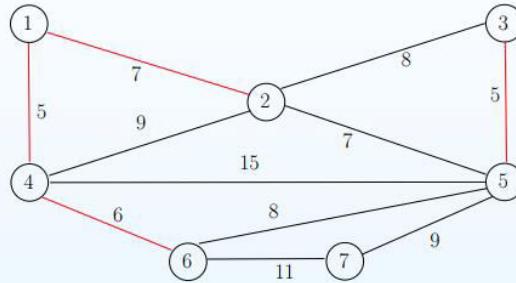


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1, 4) - (3, 5) - (4, 6) - (1, 2) - (2, 5) - (2, 3) \\ & - (6, 5) - (2, 4) - (5, 7) - (6, 7) - (4, 5) \end{aligned}$$

Επανάληψη 4

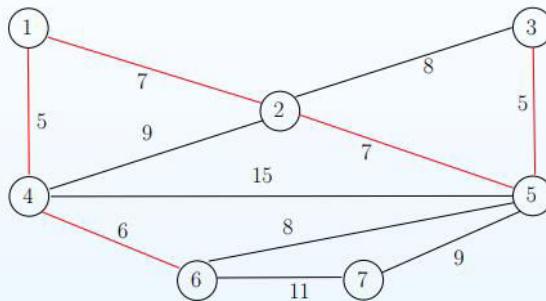


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1, 4) - (3, 5) - (4, 6) - (1, 2) - (2, 5) - (2, 3) \\ & - (6, 5) - (2, 4) - (5, 7) - (6, 7) - (4, 5) \end{aligned}$$

Επανάληψη 5

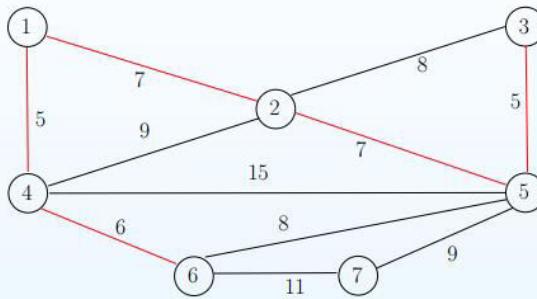


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1,4) - (3,5) - (4,6) - (1,2) - (2,5) - (2,3) \\ & - (6,5) - (2,4) - (5,7) - (6,7) - (4,5) \end{aligned}$$

Επανάληψη 6

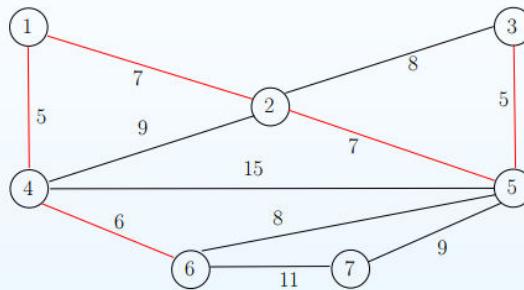


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1,4) - (3,5) - (4,6) - (1,2) - (2,5) - (2,3) \\ & - (6,5) - (2,4) - (5,7) - (6,7) - (4,5) \end{aligned}$$

Επανάληψη 7

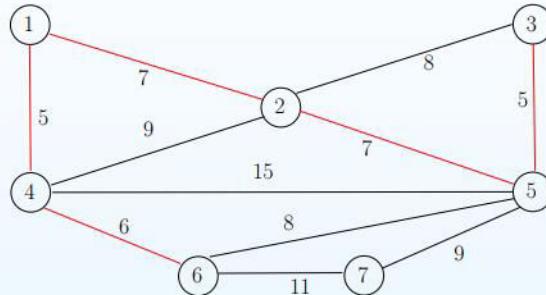


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} & (1,4) - (3,5) - (4,6) - (1,2) - (2,5) - (2,3) \\ & - (6,5) - (2,4) - (5,7) - (6,7) - (4,5) \end{aligned}$$

Επανάληψη 8

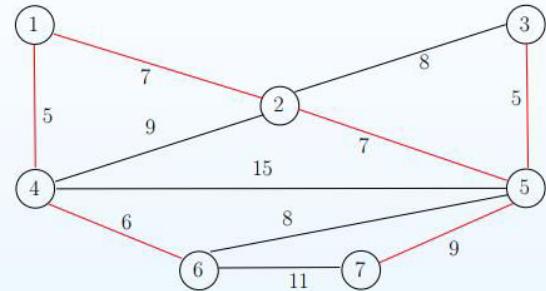


Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} &(1, 4) - (3, 5) - (4, 6) - (1, 2) - (2, 5) - (2, 3) \\ &\quad -(6, 5) - (2, 4) - (5, 7) - (6, 7) - (4, 5) \end{aligned}$$

Επανάληψη 9



Σχήμα 5: Αλγόριθμος Kruskal - κατασκευή ΕΣΔ

Οι ακμές κατά αύξουσα σειρά βάρους:

$$\begin{aligned} &(1, 4) - (3, 5) - (4, 6) - (1, 2) - (2, 5) - (2, 3) \\ &\quad -(6, 5) - (2, 4) - (5, 7) - (6, 7) - (4, 5) \end{aligned}$$

Ορθότητα

Πρόταση 8 Ο Αλγόριθμος του Kruskal παράγει ένα ΕΣΔ T του συνδεδεμένου βεβαρυμένου μη-κατευθυνομένου γραφήματος $G(V, E, w)$

Απόδειξη

- Το T δεν περιέχει κύκλους από κατασκευής
- Το T είναι γεννητορικό υπογράφημα του G (δηλαδή $V(T) = V(G)$ - δες ορισμό γεννητορικού υπογραφήματος). Στην αντίθετη περίπτωση θα υπάρχει κορυφή v η οποία δεν ανήκει στο σύνολο $V(T)$. Όμως από τις προσπιπουσες ακμές στην v αυτή που θα είχε το μικρότερο βάρος θα έπρεπε να προστεθεί σε κάποια επανάληψη στο $E(T)$ χωρίς να δημιουργείται κύκλος αφού $v \notin V(T)$.

Απόδειξη (συνεχ.)

- Το T είναι συνδεδεμένο. Στην αντίθετη περίπτωση θα έχει k συνιστώσες και έστω n_1, \dots, n_k ο αριθμός των κορυφών της κάθε συνιστώσας. Όμως τότε

$$|E(T)| \leq n_1 - 1 + n_2 - 1 + \dots + n_k - 1 = |V| - k < |V| - 1.$$

Επομένως ο αλγόριθμος δεν θα είχε τερματίσει αφού η συνθήκη στη γραμμή 4 θα ήταν αληθής και θα υπήρχαν ακμές με άκρα σε διαφορετικές συνιστώσες η προσθήκη των οπίων δεν θα δημιουργούσε κύκλο σε κάποια επόμενη επανάληψη (αφού το G είναι συνδεδεμένο)

Από τις τρεις παραπάνω παρατηρήσεις συνεπάγεται ότι το T είναι συνδετικό δένδρο του G .

Απόδειξη (συνεχ.) Στη συνέχεια θα δείξουμε ότι το T είναι ένα ΕΣΔ.

Έστω T^* ένα ΕΣΔ του G . Αν $T^* = T$ η απόδειξη ολοκληρώνεται εδώ. Αν $T^* \neq T$ θεωρούμε την ακμή $e \in E(T) \setminus E(T^*)$ που έχει το μικρότερο βάρος από όλες τις ακμές του συνόλου της διαφοράς. Το υπογράφημα $T^* \cup \{e\}$ περιέχει κύκλο C και επειδή το T είναι δένδρο υπάρχει ακμή $f \in E(C) \setminus E(T) \Rightarrow f \in E(T^*) \setminus E(T)$. Παρατηρήστε ότι

$$w(f) \geq w(e), \quad (1)$$

γιατί διαφορετικά ($w(f) < w(e)$) η ακμή f θα είχε εξετασθεί από τον αλγόριθμο σε επανάληψη πριν από την e και άρα θα είχε προστεθεί στο σύνολο $E(T)$ αντί της e .

Απόδειξη (συνεχ.) Το υπογράφημα $T' = T^* - f + e$ είναι συνδετικό δένδρο (Λήμμα 3) και περιέχει μία παραπάνω ακμή κοινή με το T από ότι το T^* . Από (1) έχουμε ότι

$$w(T') \leq w(T^*).$$

Επαναλαμβάνοντας την ίδια διαδικασία μπορούμε να μετατρέψουμε το T^* σε T χωρίς αύξηση του αθροίσματος των συντελεστών των ακμών. Επομένως,

$$w(T) \leq w(T^*),$$

και επειδή T^* είναι ΕΣΔ η παραπάνω σχέση ισχύει σαν ισότητα και άρα και το T είναι ΕΣΔ.

Πολυπλοκότητα

Πρόταση 9 Ο αλγόριθμος του Kruskal εκτελεί $O(m \lg m)$ βήματα, όπου $m = |E|$.

Απόδειξη Ο αλγόριθμος χρειάζεται $m \lg m$ βήματα για να διατάξει τις ακμές κατά αύξουσα σειρά βάρους (π.χ. ταξινόμηση μέσω της μεθόδου mergesort) στη Γραμμή 1. Για κάθε ακμή πρέπει να ελέγχεται αν το υπογράφημα με σύνολο ακμών $E(T) \cup \{e_i\}$ περιέχει κύκλο (Γραμμή 5). Αυτό μπορεί να γίνει σε σταθερό χρόνο (το πολύ δύο συγκρίσεις) ελέγχοντας αν και οι δυο κορυφές της ακμής e_i ανήλουν στο σύνολο $V(T)$.

Ο παραπάνω έλεγχος το πολύ να γίνει m φορές. Άρα συνολικά οι στοιχειώδεις πράξεις των γραμμάων 4-9 είναι τάξης $O(m)$.

06_ΔΥΝΑΜΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ / 08a Dynamic Programming Notes

Fibonacci (Υπολογισμός Σειράς Fibonacci)

Iδέα

Ένα απλό παράδειγμα, μέσω του οποίου θα αναδειχθούν τα ιδιαίτερα χαρακτηριστικά του ΔΠ αποτελεί η χρήση του στον υπολογισμό του n -ισοτού όρου ($n \geq 0$) της ακολουθίας Fibonacci. Συμβολίζουμε το πρόβλημα αυτό ως $F(n)$. Από την (4.3) γνωρίζουμε ότι για μεγάλες τιμές του n η λύση του προκύπτει από το άθροισμα των λύσεων των υποπροβλημάτων $F(n-1), F(n-2)$ ενώ η λύση του $F(n-1)$ προκύπτει από τη λύση των $F(n-2), F(n-3)$, η λύση του $F(n-2)$ προκύπτει από τη λύση των $F(n-3), F(n-4)$, κοκ. Επίσης από την ίδια σχέση, γνωρίζουμε άμεσα τη λύση του υποπροβλήματος $F(0)$ καθώς και του $F(1)$. Επομένως τα υποπροβλήματα στα οποία αποσυντίθεται το αρχικό πρόβλημα $F(n)$ είναι: $F(0), F(1), F(2), \dots, F(n-2), F(n-1)$. Θεωρούμε τα υποπροβλήματα αυτά σαν στάδια (υπολογισμού): σε καθένα από αυτά - εκτός από τα δύο πρώτα - υπολογίζεται ο αντίστοιχος όρος της ακολουθίας - λύση του υποπροβλήματος που αντιστοιχεί στο στάδιο αυτό - από τη λύση των δύο προηγουμένων σταδίων. Στην περίπτωση αυτή δεν υπάρχει θέμα αξιολόγησης εναλλακτικών αποφάσεων με βάση την οποία θα υπολογιστεί η λύση του υποπροβλήματος. Αν f_i συμβολίζει τη λύση του υποπροβλήματος $F(i)$, για $i = 0, \dots, n$, η λύση των υποπροβλημάτων καθορίζεται από την αναδρομική σχέση

$$\begin{aligned} f_i &= f_{i-1} + f_{i-2}, i = 2, \dots, n, \\ f_1 &= 1, \\ f_0 &= 0. \end{aligned}$$

Από την παραπάνω σχέση είναι άμεσο ότι προκειμένου να υπολογιστεί η τιμή της f_i , για $i \geq 2$, θα πρέπει να έχουν υπολογιστεί οι τιμές f_{i-1}, f_{i-2} . Άρα θα πρέπει ο υπολογισμός των λύσεων των υποπροβλημάτων να γίνει με τη σειρά $F(0), F(1), \dots, F(n-1), F(n)$. Επίσης γνωρίζοντας τις τιμές f_{i-1}, f_{i-2} (αποθηκεύοντας τες στη μνήμη) μπορούμε να υπολογίσουμε σε σταθερό αριθμό ΣΥΒ την f_i . Ο Αλγόριθμος 46 που επιλύει το πρόβλημα $F(n)$ προκύπτει άμεσα από την παραπάνω ανάλυση.

Αλγόριθμος

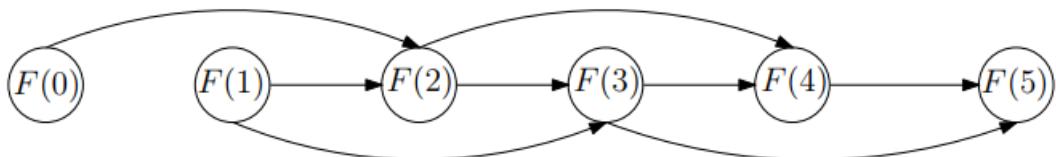
Αλγόριθμος 46 Υπολογισμός αριθμών Fibonacci.

Απαιτείται: Ακέραιος $n \geq 0$.

Επιστρέφεται: n -ιοστός αριθμός Fibonacci.

```
1: function FiboDynamic(int n)
2:   if n ≤ 1 then
3:      $f_n \leftarrow n;$ 
4:   else
5:      $f_{n-2} \leftarrow 0; f_{n-1} \leftarrow 1;$ 
6:     i ← 2;
7:     while i ≤ n do
8:        $f_n \leftarrow f_{n-1} + f_{n-2};$ 
9:        $f_{n-2} \leftarrow f_{n-1};$ 
10:       $f_{n-1} \leftarrow f_n;$ 
11:      i++;
12:    end while
13:  end if
14:  return  $f_n;$ 
15: end function
```

Παράδειγμα



Σχήμα 8.1: Κατευθυνόμενο γράφημα υποπροβλημάτων για τον υπολογισμό του πέμπτου όρου της ακολουθίας Fibonacci

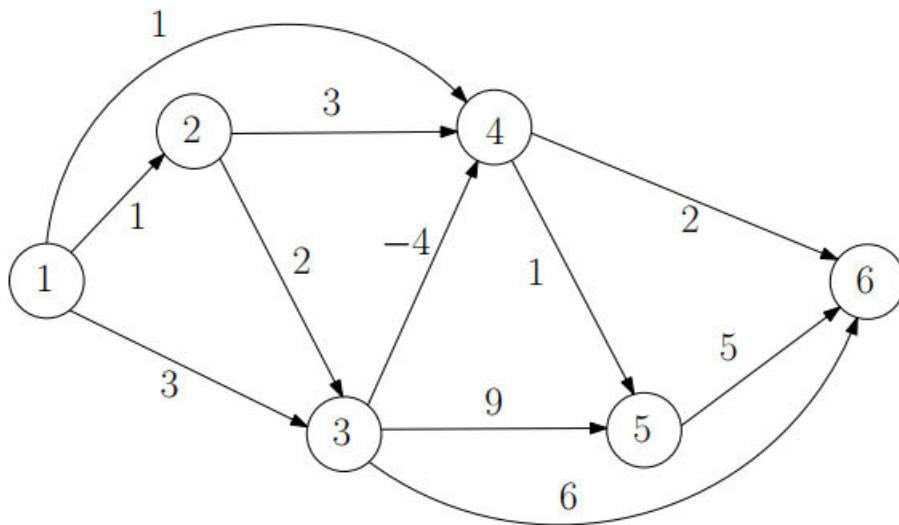
Πολυπλοκότητα

Συγκρίνοντας τον Αλγόριθμο 46 με τον Αλγόριθμο 20 παρατηρούμε ότι ο πρώτος είναι πολύ πιο αποτελεσματικός από το δεύτερο· είναι εύκολο να δείξει κανείς ότι η πολυπλοκότητα του πρώτου είναι $\Theta(n)$ ενώ του δεύτερου $\Theta(2^n)$. Αυτό είναι αποτέλεσμα της χρήσης της μνήμης από τον πρώτο αλγόριθμο αφού όποτε υπολογίζεται μία τιμή f_i αποθηκεύεται στη μνήμη και χρησιμοποιείται τόσο στον υπολογισμό της f_{i+1} όσο και στον υπολογισμό της f_{i+2} . Αντίθετα στην περίπτωση του Αλγόριθμου 20 ο υπολογισμός της τιμής αυτής γίνεται δύο φορές - μία για τον υπολογισμό της f_i και άλλη μία για τον υπολογισμό της f_2 .

Iδέα

Έστω ότι έχουμε ένα ΑΚΓ όπου κάθε κατευθυνόμενη ακμή (i, j) - ακμή η οποία εξέρχεται από την κορυφή i και εισέρχεται στην κορυφή j - σχετίζεται με έναν (αριθμητικό) συντελεστή $w(i, j)$. Το μήκος ενός (κατευθυνόμενου) μονοπατιού που ενώνει την κορυφή i με την κορυφή j ορίζεται ως το άθροισμα των συντελεστών των ακμών που συμμετέχουν σε αυτό. Δεδομένης μίας κορυφής s θέλουμε να βρούμε, για κάθε άλλη κορυφή u , το μονοπάτι εκκινεί από την s και καταλήγει στη u και έχει ελάχιστο μήκος. Θα συμβολίζουμε με d_u το μήκος αυτό για κάθε κορυφή u . Επειδή το γράφημα είναι άκυκλο $d_s = 0$.

Μπορούμε εύκολα να επιλύσουμε το πρόβλημα αυτό με ΔΠ αφού η κάθε κορυφή στο γράφημα ορίζει ένα υποπρόβλημα. Συγκεκριμένα, για κάθε κορυφή u , ορίζουμε



Σχήμα 8.2: Άκυκλο κατευθυνόμενο γράφημα - Παράδειγμα 19

ως $D(u)$ το υποπρόβλημα εύρεσης του συντομότερου μονοπατιού που εκκινεί από την κορυφή s και καταλήγει στη u (Στάδιο 1). Κάθε κορυφή v για την οποία υπάρχει η κατευθυνόμενη ακμή (v, u) αποτελεί μία επιλογή από την οπαία μπορούμε να φτάσουμε στη u . Φτάνουμε με ένα μονοπάτι από την s στη v και στη συνέχεια διασχίζουμε την ακμή (v, u) (Στάδιο 2). Έστω $N^-(u)$ το σύνολο που περιέχει όλες της κορυφές v με την ιδιότητα αυτή - $v \in N^-(u)$ αν υπάρχει η ακμή (v, u) . Άρα μπορούμε να πάρουμε την καλύτερη επιλογή που μας οδηγεί στη κορυφή u . Δηλαδή επιλέγεται η κορυφή v η οποία ελαχιστοποιεί το άθροισμα του μήκους του μονοπατιού από το s στο v συν το συντελεστή $w(v, u)$. Η σκέψη αυτή μας οδηγεί στη σχέση που συνδέει τη λύση του υποπροβλήματος $D(u)$ με τις λύσεις των υποπροβλημάτων $D(v)$, για κάθε $v \in N^-(u)$ (Στάδιο 3). Η σχέση αυτή διατυπώνεται μαθηματικά ως

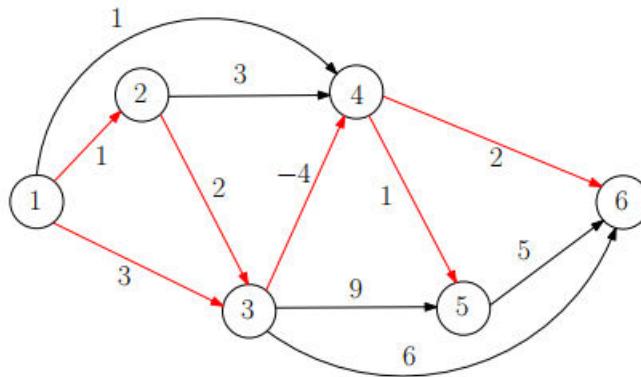
$$d_u = \min\{d_v + w(v, u) : v \in N^-(u)\}. \quad (8.4)$$

Η παραπάνω σχέση υπαγορεύει και τη σειρά με την οποία πρέπει να επιλύονται τα υποπροβλήματα: κάθε κορυφή αντιπροσωπεύει ένα υποπρόβλημα και επομένως η τοπολογική ταξινόμηση των κορυφών υποδεικνύει τη ζητούμενη σειρά επίλυσης (Στάδιο 4). Επειδή το γράφημα είναι ΑΚΓ πάντα υπάρχει μία διάταξη των κορυφών σε τοπολογικά ταξινομημένη σειρά.

Η λύση του αρχικού προβλήματος (Στάδιο 5) θα προκύψει όταν έχουν υπολογιστεί τα ελάχιστα μήκη d_u για κάθε κορυφή u . Για να μπορέσουμε να ιχνηλατήσουμε το κάθε μονοπάτι θα πρέπει μαζί με τον υπολογισμό που γίνεται από την (8.4) να κρατάμε στο διάνυσμα p την κορυφή που οδηγεί στη βέλτιστη επιλογή. Δηλαδή,

$$p_u = \operatorname{argmin}\{d_v + w(v, u) : v \in N^-(u)\}. \quad (8.5)$$

Παράδειγμα



Σχήμα 8.3: Συντομότερα μονοπάτια από την κορυφή 1 - Παράδειγμα 19

Παράδειγμα 19. Στο γράφημα του Σχήματος 8.2 θέλουμε να υπολογίσουμε τα ελάχιστα μονοπάτια από την κορυφή 1 προς όλες τις υπόλοιπες κορυφές. Παρατηρούμε ότι πρόκειται περί ενός ΑΚΓ όπου οι ετικέτες των κορυφών έχουν δοθεί σύμφωνα με την τοπολογική ταξινόμηση. Έχουμε,

$$\begin{aligned} N^-(1) &= \emptyset, & N^-(2) &= \{1\}, & N^-(3) &= \{1, 2\}, \\ N^-(4) &= \{1, 2, 3\}, & N^-(5) &= \{3, 4\}, & N^-(6) &= \{3, 4, 5\} \end{aligned}$$

Από τις (8.4), (8.5),

$$d_1 = 0,$$

$$d_2 = \min\{d_v + w(v, 2) : v \in N^-(2)\} = \min\{d_1 + w(1, 2)\} = \min\{0 + 1\} = 1, \quad p_2 = 1,$$

$$d_3 = \min\{d_v + w(v, 3) : v \in N^-(3)\} = \min\{d_1 + w(1, 3), d_2 + w(2, 3)\}$$

$$= \min\{0 + 3, 1 + 2\} = 3, \quad p_3 = 1, \text{ ή } p_3 = 2,$$

$$d_4 = \min\{d_v + w(v, 4) : v \in N^-(4)\} = \min\{d_1 + w(1, 4), d_2 + w(2, 4), d_3 + w(3, 4)\}$$

$$= \min\{0 + 1, 1 + 3, 3 + (-4)\} = -1, \quad p_4 = 3,$$

$$d_5 = \min\{d_v + w(v, 5) : v \in N^-(5)\} = \min\{d_3 + w(3, 5), d_4 + w(4, 5)\}$$

$$= \min\{3 + 9, -1 + 1\} = 0, \quad p_5 = 4,$$

$$d_6 = \min\{d_v + w(v, 6) : v \in N^-(6)\} = \min\{d_3 + w(3, 6), d_4 + w(4, 6), d_5 + w(5, 6)\}$$

$$= \min\{3 + 6, -1 + 2, 0 + 5\} = 1, \quad p_6 = 4.$$

Τα συντομότερα μονοπάτια όπως ιχνηλατούνται μέσω του διανύσματος p απεικονίζονται στο Σχήμα 8.3 (κόκκινες ακμές).

Πολυπλοκότητα

Η παραπάνω ανάλυση οδηγεί σε έναν αλγόριθμο πολυπλοκότητας $O(n^2)$, όπου n συμβολίζει τον αριθμό των κορυφών του γραφήματος. Πράγματι αν αντιστοιχίσουμε στις κορυφές τους αριθμούς $1, 2, \dots$ σε αύξουσα σειρά σύμφωνα με την τοπολογική ταξινόμηση και θεωρήσουμε ότι $s = 1$, τότε $N^-(i) \subseteq \{1, \dots, i-1\}$. Δηλαδή στη χειρότερη περίπτωση $|N^-(i)| = i - 1$. Όμως $(|N^-(i)|)$ είναι ο αριθμός των SYB που εκτελούνται για την επίλυση του στιγμιότυπου $D(i)$. Η τάξη του άθροισμα αυτών των SYB είναι $O(n^2)$.

TextJust (Στοίχιση Κειμένου)

Ιδέα

Μία παράγραφος κειμένου αποτελείται από n λέξεις. Υποθέτουμε ότι κάθε χαρακτήρας - γράμμα, σύμβολο, διάστημα - που εμφανίζεται στο κείμενο πιάνει τον ίδιο χώρο. Δεδομένου του αριθμού των χαρακτήρων που περιέχει μία γραμμή, έστω m , επιθυμούμε τη διάσπαση της παραγράφου σε γραμμές ώστε το κείμενο να εμφανίζεται όσο το δυνατόν πιο «ομοιόμορφα». Αν θεωρήσουμε ότι η παράγραφος θα πρέπει να εμφανίζεται με αριστερή στοίχιση, τότε ο όρος «ομοιόμορφα» αναφέρεται στην ελαχιστοποίηση των κενών διαστημάτων στο τέλος κάθε γραμμής.

Θεωρώντας ότι η παράγραφος αποτελείται από τις λέξεις w_1, w_2, \dots, w_n , αν μία γραμμή αποτελείται από τις λέξεις $w_i, w_{i+1}, \dots, w_{j-1}, w_j$, για $1 \leq i \leq j \leq n$, μπορούμε να συσχετίσουμε με τη γραμμή αυτή έναν αριθμό, έστω $b(i, j)$, ο οποίος εκφράζει το πόσο κακή (μη-ομοιόμορφη) είναι η γραμμή αυτή. Θα ονομάζουμε το $b(i, j)$ ως συντελεστή ανομοιομορφίας της γραμμής. Συνήθως ο συντελεστής αυτός είναι συνάρτηση του πλήθους των κενών διαστημάτων που υπάρχουν είτε στο τέλος της γραμμής ή ενδιάμεσα από τις λέξεις (αν θέλουμε πλήρη στοίχιση). Μία καλή στοίχιση θα χωρίζει την παράγραφο σε γραμμές κατά τρόπο ώστε να ελαχιστοποιείται το άθροισμα των συντελεστών ανομοιομορφίας των γραμμών (ανομοιομορφία της παραγράφου). Το ζητούμενο είναι το πως θα μπορέσει να επιτευχθεί μία καλή στοίχιση. Δηλαδή ζητείται ο χωρισμός του κειμένου σε γραμμές - ποιες λέξεις θα περιέχει η κάθε γραμμή - ώστε να ελαχιστοποιείται η ανομοιομορφία της παραγράφου.

Για να μπορέσουμε να καταστρώσουμε ένα σχήμα ΔΠ που να επιλύει το παραπάνω πρόβλημα, θεωρούμε γνωστούς τους συντελεστές $b(i, j)$. Πράγματι τους συντελεστές αυτούς μπορούμε να τους υπολογίσουμε εκ' των προτέρων. Αν για παράδειγμα μας ενδιαφέρουν τα κενά στο τέλος κάθε γραμμής, τότε θα μπορούσαμε να ορίσουμε την ανομοιομορφία της γραμμής με πρώτη λέξη την w_i και τελευταία την w_j , με $1 \leq i \leq j \leq n$, ως

$$b(i, j) = \begin{cases} (m - (j - i) - \sum_{t=i}^j |w_t|)^2, & \text{αν } m - (j - i) - \sum_{t=i}^j |w_t| \geq 0, \\ \infty, & \text{διαφορετικά,} \end{cases} \quad (8.6)$$

όπου $|w_t|$ είναι το πλήθος των χαρακτήρων της λέξης w_t και $j - i$ ο αριθμός των κενών διαστημάτων ανάμεσα στις λέξεις. Η περίπτωση $b(i, j) = \infty$ προκύπτει όταν το πλήθος των χαρακτήρων των λέξεων από w_i έως w_j με τα ενδιάμεσα κενά διαστήματα ξεπερνούν τον αριθμό των χαρακτήρων που μπορεί να χωρέσει στη γραμμή.

Τα πέντε στάδια του ΔΠ

Τα πέντε στάδια της μεθοδολογίας του ΔΠ ακολουθούν.

1. (Ορισμός υποπροβλημάτων) Έστω $D(i), i = 1, \dots, n$, το υποπρόβλημα στοίχισης κειμένου αν έχουμε στην παράγραφο τις λέξεις w_i, w_{i+1}, \dots, w_n . Δηλαδή το υποπρόβλημα $D(i)$ είναι το πρόβλημα διαχωρισμού των λέξεων αυτών σε γραμμές δεδομένου ότι η πρώτη γραμμή ξεκινάει με τη λέξη w_i . Προφανώς το σύνολο των υποπροβλημάτων που ορίζονται για κάθε τιμή i έχει επιθεματικό χαρακτήρα (συμβολικά $D(i :)$). Συνολικά υπάρχουν n τέτοια υποπροβλήματα.
2. (Καθορισμός επιλογών) Οι επιλογές που έχουμε όταν θέλουμε να επιλύσουμε το πρόβλημα $D(i)$ αφορά τις λέξεις που θα περιέχει η γραμμή με πρώτη λέξη την w_i . Οι επιλογές είναι $n - i + 1$. Δηλαδή η γραμμή μπορεί να περιέχει μόνο τη λέξη w_i ή τις λέξεις w_i, w_{i+1} ή ..., ή όλες τις λέξεις w_i, \dots, w_n . Προφανώς από κάποια λέξη και πέρα δεν μπορούν οι λέξεις να χωρέσουν στη γραμμή αφού μαζί με τα ενδιάμεσα κενά διαστήματα ξεπερνιέται η τιμή m .
3. (Αναδρομική σχέση) Συμβολίζουμε με D_i τη λύση του υποπροβλήματος $D(i)$. Δηλαδή το D_i αποτελεί μία ποσοτικοποίηση της μικρότερης δυνατής ανομοιομορφίας για την παράγραφο με λέξεις w_i, w_{i+1}, \dots, w_n χρησιμοποιώντας για κάθε γραμμή ως μέτρο το συντελεστή ανομοιομορφίας της γραμμής. Προφανώς αν $w_{j-1}, j \geq i + 1$ είναι η τελευταία λέξη της γραμμής που ξεκινάει με τη λέξη w_i . Τότε $D_i = b(i, j - 1) + D_j$. Όπως είδαμε προηγουμένως, έχουμε $n - i + 1$ επιλογές ως προς την τελευταία λέξη της γραμμής. Από αυτές θα επιλέξουμε την καλύτερη. Επομένως ισχύει ο αναδρομικός τύπος

$$D_i = \min\{b(i, j - 1) + D_j : i + 1 \leq j \leq n + 1\}, \quad (8.7)$$

όπου $D_{n+1} = 0$. Για να μπορέσουμε να ανακτήσουμε τη λύση χρησιμοποιούμε το διάνυσμα p το οποίο παίρνει τιμές ως εξής:

$$p_i = \operatorname{argmin}\{b(i, j - 1) + D_j : i + 1 \leq j \leq n + 1\}. \quad (8.8)$$

Δηλαδή η τιμή w_{p_i} είναι η λέξη που ξεκινάει την επόμενη γραμμή από τη γραμμή που ξεκινάει με τη λέξη w_i

4. (Τοπολογική ταξινόμηση) Η σειρά με την οποία επιλύονται τα υποπροβλήματα $D(i)$ είναι σε φθίνουσα διάταξη των τιμών i εκκινώντας από $i = n$.
5. (Επίλυση του αρχικού προβλήματος) Η τιμή D_1 αποτελεί τη λύση του αρχικού προβλήματος.

Αλγόριθμος

Αλγόριθμος 47 Διάσπαση κειμένου σε γραμμές με ελαχιστοποίηση της ανομοιομορφίας

Απαιτείται: Ακέραιος n , πίνακας b διάστασης $n \times n$ - το στοιχείο $b[i, j]$ ποσοτικοποιεί την ανομοιομορφία της γραμμής με πρώτη λέξη τη w_i και τελευταία τη w_j .

Επιστρέφεται: Ελάχιστη ανομοιομορφία και πίνακας p κάθε στοιχείο του οποίου υποδεικνύει την πρώτη λέξη της επόμενης γραμμής

```
1: function TextJust(int n, int b[ ][], int p[])
2:    $D[n + 1] \leftarrow 0;$ 
3:   for  $i \leftarrow n; i \geq 1; i --$  do
4:      $D[i] \leftarrow \infty;$ 
5:     for  $j \leftarrow i + 1; j \leq n + 1; j ++$  do
6:       if  $D[i] > D[j] + b[i, j - 1]$  then
7:          $D[i] \leftarrow D[j] + b[i, j - 1];$ 
8:          $p[i] \leftarrow j;$ 
9:       end if
10:      end for
11:    end for
12:    return  $D[1];$ 
13: end function
```

Παράδειγμα

Παράδειγμα 20. Θέλουμε να διατάξουμε σε γραμμές των 9 χαρακτήρων την φράση

Στη γάτα δεν αρέσει το νερό

Το πλήθος των χαρακτήρων της κάθε λέξης παρουσιάζεται στον Πίνακα 8.1. Οι συντελεστές ανομοιομορφίας όπως προκύπτουν από την (8.9) παρουσιάζονται στον Πίνακα 8.2. Από τις σχέσεις (8.7), (8.8) έχουμε

i	1	2	3	4	5	6
w_i	Στη	γάτα	δεν	αρέσει	το	νερό
$ w_i $	3	4	3	6	2	4

Πίνακας 8.1: Αριθμός γραμμάτων - Παράδειγμα 20

	1	2	3	4	5	6
1	36	1	∞	∞	∞	∞
2		25	1	∞	∞	∞
3			36	∞	∞	∞
4				9	0	∞
5					49	4
6						25

Πίνακας 8.2: Συντελεστές ανομοιομορφίας υπολογισμένοι από (8.6) για $m = 9$ - Παράδειγμα 20

$$D_7 = 0,$$

$$D_6 = 25,$$

$$D_5 = \min\{D_6 + b(5, 5), D_7 + b(5, 6)\} = \min\{25 + 49, 0 + 4\} = 4,$$

$$p_5 = 7,$$

$$D_4 = \min\{D_5 + b(4, 4), D_6 + b(4, 5), D_7 + b(4, 6)\} = \min\{4 + 9, 25 + 0, 0 + \infty\} = 13,$$

$$p_4 = 5,$$

$$D_3 = \min\{D_4 + b(3, 3), D_5 + b(3, 4), D_6 + b(3, 5), D_7 + b(3, 6)\}$$

$$= \min\{13 + 36, 4 + \infty, 25 + \infty, 0 + \infty\} = 49,$$

$$p_3 = 4,$$

$$D_2 = \min\{D_3 + b(2, 2), D_4 + b(2, 3), D_5 + b(2, 4), D_6 + b(2, 5), D_7 + b(2, 6)\}$$

$$p_2 = 4,$$

$$= \min\{49 + 25, 13 + 1, 4 + \infty, 25 + \infty, 0 + \infty\} = 14,$$

$$D_1 = \min\{D_2 + b(1, 1), D_3 + b(1, 2), D_4 + b(1, 3), D_5 + b(1, 4), D_6 + b(1, 5), D_7 + b(1, 6)\}$$

$$= \min\{14 + 36, 49 + 1, 13 + \infty, 4 + \infty, 25 + \infty, 0 + \infty\} = 50,$$

$$p_1 = 2 \text{ ή } p_1 = 3.$$

Επομένως υπάρχουν δύο τρόποι διάσπασης της φράσης σε γραμμές των εννέα χαρακτήρων που ελαχιστοποιούν τη (συνολική) ανομοιομορφία. Οι τρόποι αυτοί μαζί με τα αντίστοιχα διανύσματα p παρουσιάζονται στον Πίνακα 8.3.

Ο απλούστερος κανόνας που λέει να τοποθετήσουμε σε κάθε γραμμή όσο περισσότερες λέξεις μπορούμε - κανόνας που ακολουθούσε (ακολουθεί;) το MsWord - οδηγεί σε πολλές περιπτώσεις σε μεγαλύτερη ανομοιομορφία. Ακολουθώντας τον κανόνα αυτό στο κείμενο του Παραδείγματος 20 παίρνουμε τη διάταξη

	Στη γάτα δεν αρέσει το νερό			Στη γάτα δεν αρέσει το νερό	
$p = [2, 4, 4, 5, 7, ,]$		$p = [3, 4, 4, 5, 7, ,]$			

Πίνακας 8.3: Διαφορετικές διατάξεις γραμμών που ελαχιστοποιούν την ανομοιομορφία - Παράδειγμα 20

Στη γάτα δεν αρέσει το νερό

Υπολογίζοντας την ανομοιομορφία της παραπάνω διάταξης ως το άθροισμα των τετραγώνων των κενών διαστημάτων στο τέλος κάθε γραμμής (με αριθμό χαρακτήρων γραμμής $m = 9$) παίρνουμε την τιμή 62 η οποία είναι μεγαλύτερη από την τιμή 50 που υπολογίσαμε παραπάνω. Το αποτέλεσμα αυτό αντικατοπτρίζει την ανομοιομορφία της παραπάνω διάταξης στη δεύτερη και τέταρτη γραμμή.

Πολυπλοκότητα

Με γνωστούς τους συντελεστές ανομοιομορφίας $b(i, j)$ ο υπολογισμός της λύσης του κάθε υποπροβλήματος D_i παίρνει σταθερό αριθμό ΣΥΒ: εκτελούνται $n - i + 1$ προσθέσεις, $n - i + 1$ αφαιρέσεις (υπολογισμός της τιμής $j - 1$), $n - i$ συγκρίσεις και μία εκχώρηση. Άρα ο συνολικός αριθμός των ΣΥΒ που εκτελούνται προκειμένου να υπολογιστεί η τιμή D_1 που αποτελεί τη λύση στο αρχικό πρόβλημα είναι $\sum_{i=1}^n 3(n - i + 1) = \Theta(n^2)$. Ισοδύναμα, ο αριθμός των υποπροβλημάτων είναι n και η λύση σε κάθε υποπρόβλημα υπολογίζεται θεωρώντας το πολύ $n - 1$ επιλογές ενώ η αποτίμηση της κάθε επιλογής γίνεται σε σταθερό αριθμό ΣΥΒ. Άρα η πολυπλοκότητα του αλγόριθμου είναι $\Theta(n) \cdot O(n)\Theta(1) = O(n^2)$.

Όμως ο υπολογισμός των συντελεστών $b(i, j)$ από την (8.6) είναι τάξης $\Theta(n^3)$. Μπορεί ο υπολογισμός να γίνει σε $\Theta(n^2)$ παρατηρώντας ότι μπορούμε να υπολογίσουμε τον όρο $b(i, j)$ από τον όρο $b(i, j - 1)$ εκτελώντας σταθερό αριθμό ΣΥΒ. Πράγματι, για $i < j$,

$$\begin{aligned}
b(i, j) &= (m - (j - i) - \sum_{t=i}^j |w_t|)^2 \\
&= (m - (j - 1 - i) - 1 - \sum_{t=i}^{j-1} |w_t| - |w_j|)^2 \\
&= (\sqrt{b(i, j - 1)} - 1 - |w_j|)^2.
\end{aligned}$$

Επομένως, αντί της (8.6), μπορούμε να υπολογίσουμε τους συντελεστές ανομοιομορφίας για $1 \leq i \leq j \leq n$, από τη σχέση

$$b(i, j) = \begin{cases} (m - |w_i|)^2, & \text{αν } i = j, \\ (\sqrt{b(i, j - 1)} - 1 - |w_j|)^2, & \text{αν } i < j, \sqrt{b(i, j - 1)} - 1 - |w_j| \geq 0, \\ \infty, & \text{διαφορετικά.} \end{cases} \quad (8.9)$$

MinMatMult (Πολλαπλασιασμός Πινάκων)

Iδέα

Από τη Γραμμική Άλγεβρα είναι γνωστό το γινόμενο δύο πινάκων A, B διάστασης $p \times q$ και $q \times r$ είναι ένας πίνακας $C = A \cdot B$ όπου το κάθε στοιχείο του δίνεται από τη σχέση

$$C_{i,j} = \sum_{k=1}^q A_{i,k} \cdot B_{k,j}, i = 1, \dots, p, j = 1, \dots, r.$$

Από την παραπάνω σχέση προκύπτει εύκολα ότι κάθε στοιχείο του C προκύπτει από q πολλαπλασιασμούς και επειδή η διάσταση του είναι $p \times r$ για τον υπολογισμό του εκτελούνται $p \cdot q \cdot r$ πολλαπλασιασμοί. Όμως αν έχουμε μία σειρά από πίνακες A_1, A_2, \dots, A_n με συμβατές μεταξύ τους διαστάσεις - ο αριθμός των στήλων του πίνακα A_i είναι ίσος με τον αριθμό των γραμμών του πίνακα A_{i+1} , για $i = 1, \dots, n-1$, - πόσους πολλαπλασιασμούς χρειάζεται να εκτελέσουμε προκειμένου να βρούμε το γινόμενο

$$A_1 \cdot A_2 \cdots A_n; \quad (8.10)$$

Η απάντηση στο ερώτημα αυτό ποικίλει ανάλογα με τη σειρά που εκτελούνται οι πολλαπλασιασμοί. Για παράδειγμα, έστω ότι ο πίνακας A_i έχει k_{i-1} γραμμές και k_i στήλες. Τότε για $n = 3$ υπάρχουν δύο τρόποι για να υπολογίσουμε το γινόμενο $A_1 \cdot A_2 \cdot A_3$. Είτε το υπολογίζουμε ως $A_1 \cdot (A_2 \cdot A_3)$ ή ως $(A_1 \cdot A_2) \cdot A_3$. Στην πρώτη περίπτωση ο αριθμός των πολλαπλασιασμών που εκτελούνται είναι $k_1 \cdot k_2 \cdot k_3 + k_0 \cdot k_1 \cdot k_3$ ενώ στη δεύτερη περίπτωση $k_0 \cdot k_1 \cdot k_2 + k_0 \cdot k_2 \cdot k_3$. Δηλαδή βλέπουμε ότι το πλήθος των πολλαπλασιασμών στην πρώτη και στη δεύτερη περίπτωση δεν είναι απαραίτητα το ίδιο. Γενικότερα, ο αριθμός των τρόπων που μπορεί να υπολογιστεί το γινόμενο (8.10) είναι εκθετικός.¹ Από όλους αυτούς τους τρόπους θέλουμε να βρούμε αυτόν που υπολογίζει το γινόμενο (8.10) εκτελώντας το μικρότερο αριθμό πολλαπλασιασμών. Θα επιλύσουμε το πρόβλημα αυτό με ΔΠ. Προς τούτο ακολουθούμε τη μεθοδολογία της Ενότητας 8.2.1.

Τα πέντε στάδια του ΔΠ

- (Ορισμός υποπροβλημάτων) Θα συμβολίσουμε τον πίνακα που προκύπτει από το γινόμενο (8.10) ως $A(1, n)$ και ως $m_{1,n}$ τον αριθμό των πολλαπλασιασμών που απαιτείται για τον υπολογισμό του. Γενικότερα, για $1 \leq i \leq j \leq n$, ορίζουμε ως $A(i, j)$ το πρόβλημα εύρεσης του μικρότερου αριθμού των πολλαπλασιασμών που απαιτείται για τον υπολογισμό του πίνακα $A_{i,j}$. Στην περίπτωση όπου $i = j$ ο πίνακας $A_{i,i} = A_i$ και άρα για τον υπολογισμό του δεν λαμβάνει χώρα κανένας πολλαπλασιασμός, ήτοι $m_{i,i} = 0$.
- (Καθορισμός επιλογών) Βασική παρατήρηση είναι ότι για οποιοδήποτε $t \geq i$ και $t < j$, ο πίνακας $A_{i,j}$ μπορεί να υπολογιστεί αν είναι γνωστοί οι πίνακες $A_{i,t}$ και $A_{t+1,j}$ αφού

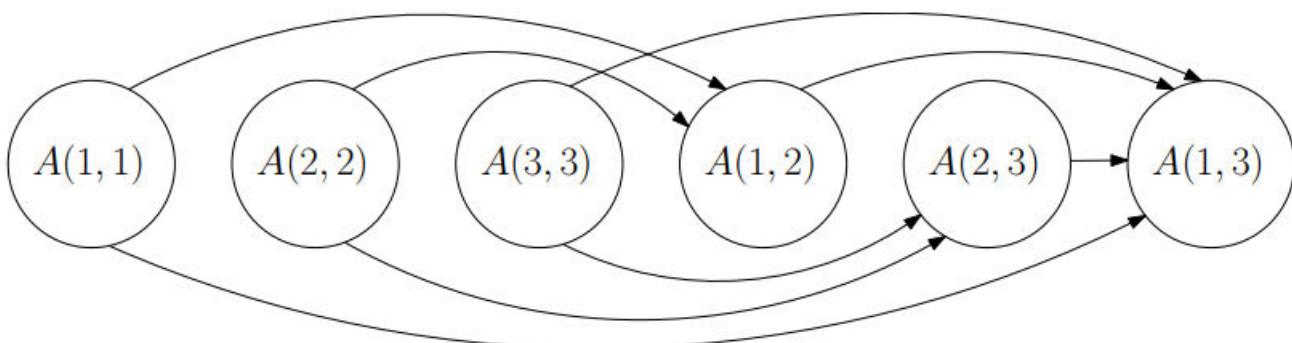
$$A_{i,j} = A_{i,t} \cdot A_{t+1,j} = (A_i \cdot \dots \cdot A_t) \cdot (A_{t+1} \cdot \dots \cdot A_j). \quad (8.11)$$

Ο αριθμός των πολλαπλασιασμών που εκτελούνται από την (8.11) είναι $k_{i-1} \cdot k_t \cdot k_j$. Από την παραπάνω σχέση είναι προφανές ότι υπάρχουν $j - i$ επιλογές για την τιμή του δείκτη t .

- (Αναδρομική σχέση) Η λύση του υποπροβλήματος $A_{i,j}$ προκύπτει από την καλύτερη επιλογή της τιμής t ώστε να προκύψει ο πίνακας $A_{i,j}$ από την (8.11). Δηλαδή για $i, j \in \{1, \dots, n\}$ έχουμε

$$m_{i,j} = \begin{cases} 0, & i = j, \\ \min_{t \in \{i, \dots, j-1\}} \{m_{i,t} + m_{t+1,j} + k_{i-1} \cdot k_t \cdot k_j\}, & 1 \leq i < j \leq n. \end{cases} \quad (8.12)$$

- (Τοπολογική ταξινόμηση) Από την (8.12) προκύπτει ότι η σειρά με την οποία θα πρέπει να λυθούν τα υποπροβλήματα υποδεικνύεται από την διαφορά των δεικτών $j - i$. Δηλαδή αρχικώς έχουμε τα υποπροβλήματα που έχουν τετριμένη λύση ($j = i \Rightarrow j - i = 0$), στη συνέχεια θα λυθούν τα υποπροβλήματα που ορίζονται για $j - i = 1$ μετά αυτά για τα οποία $j - i = 2$ κοκ. Μία οπτική αναπαράσταση του γραφήματος των υποπροβλημάτων για $n = 3$ απεικονίζεται στο Σχήμα 8.4.
- (Λύση στο αρχικό πρόβλημα) Ο ελάχιστος αριθμός πολλαπλασιασμών δίνεται από την τιμή του $m_{1,n}$.



Σχήμα 8.4: Γράφημα υποπροβλημάτων για τον πολλαπλασιασμό τριών πινάκων

Αλγόριθμος

Αλγόριθμος 48 Υπολογισμός ελάχιστου αριθμού πολλαπλασιασμών

Απαιτείται: Ακέραιος n , πίνακας k με ακέραιους στις θέσεις 0 ως n . Ο πίνακας A_i έχει $k[i - 1]$ γραμμές και $k[i]$ στήλες.

Επιστρέφεται: Ελάχιστος αριθμός πολλαπλασιασμών για τον υπολογισμό του γινόμενου των πινάκων A_1, A_2, \dots, A_n και δισδιάστατος πίνακας p για την ιχνηλάτηση της λύσης.

```
1: function MinMatMult(int n, int k[], int p[][])
2:   for i  $\leftarrow 1; i \leq n; i++$  do
3:     m[i, i]  $\leftarrow 0;$ 
4:     p[i, i]  $\leftarrow i;$ 
5:   end for
6:   for r  $\leftarrow 0; r \leq n - 1; r++$  do
7:     for i  $\leftarrow 1; i \leq n; i++$  do
8:       j  $\leftarrow i + r;$ 
9:       if j  $\leq n$  then
10:        m[i, j]  $\leftarrow \infty;$ 
11:        for t  $\leftarrow i; t \leq j - 1; t++$  do
12:          temp  $\leftarrow m[i, t] + m[t + 1, j] + k[i - 1] \cdot k[t] \cdot k[j];$ 
13:          if temp > m[i, j] then
14:            m[i, j]  $\leftarrow$  temp;
15:            p[i, j]  $\leftarrow t;$ 
16:          end if
17:        end for
18:      end if
19:    end for
20:  end for
21:  return m[1, n];
22: end function
```

Παράδειγμα

Παράδειγμα 21. Έστω πίνακες:

A_1 διάστασης 30×35 ,

A_2 διάστασης 35×15 ,

A_3 διάστασης 15×5 ,

A_4 διάστασης 5×10 ,

A_5 διάστασης 10×20 .

Από τα παραπάνω προκύπτουν οι ακόλουθοι αριθμοί γραμμών-στηλών:

$$k_0 = 30, k_1 = 35, k_2 = 15, k_3 = 5, k_4 = 10, k_5 = 20.$$

Θέλουμε να υπολογίσουμε το γινόμενο

$$A_1 \cdot A_2 \cdot A_3 \cdot A_4 \cdot A_5$$

με το μικρότερο δυνατό αριθμό πολλαπλασιασμών. Ο αριθμός των πολλαπλασιασμών που απαιτούνται για να προκύψει το παραπάνω γινόμενο αν κάνουμε τους πολλαπλασιασμούς από αριστερά προς τα δεξιά - με την προτεραιότητα που υποδηλώνουν οι παρενθέσεις:

$$(((A_1 \cdot A_2) \cdot A_3) \cdot A_4) \cdot A_5,$$

είναι ίσος με 25500. Ο αριθμός αυτός προκύπτει από το άθροισμα των αριθμών της δεύτερης στήλης του Πίνακα 8.4. Σε κάθε γραμμή του πίνακα αυτού υπολογίζεται ο αριθμός των πολλαπλασιασμών της πράξης που σημειώνεται με κόκκινο.

Σειρά εκτέλεσης	Αριθμός πολλαπλασιασμών
$A_1 \cdot A_2$	$30 \cdot 35 \cdot 15 = 15750$
$(A_1 \cdot A_2) \cdot A_3$	$30 \cdot 15 \cdot 5 = 2250$
$((A_1 \cdot A_2) \cdot A_3) \cdot A_4$	$30 \cdot 5 \cdot 10 = 1500$
$(((A_1 \cdot A_2) \cdot A_3) \cdot A_4) \cdot A_5$	$30 \cdot 10 \cdot 20 = 6000$

Πίνακας 8.4: Αναλυτικός υπολογισμός του αριθμού των πολλαπλασιασμών με σειρά από αριστερά προς τα δεξιά - Παράδειγμα 21

Για την εφαρμογή του ΔΠ αρχικώς επιλύουμε τα τετριμένα υποπροβλήματα $A_{i,i}$, ήτοι

$$\begin{aligned} m_{1,1} &= 0, & p_{1,1} &= 1, \\ m_{2,2} &= 0, & p_{2,2} &= 2, \\ m_{3,3} &= 0, & p_{3,3} &= 3, \\ m_{4,4} &= 0, & p_{4,4} &= 4, \\ m_{5,5} &= 0, & p_{5,5} &= 5. \end{aligned}$$

Στη συνέχεια υπολογίζουμε μέσω της (8.12) τις λύσεις των υποπροβλημάτων σύμφωνα με την τιμή της παράστασης $j - i$ ($j > i$) σε αύξουσα σειρά. Έχουμε,

$j - i = 1 :$

$$\begin{aligned} m_{1,2} &= \min\{m_{1,1} + m_{2,2} + k_0 \cdot k_1 \cdot k_2\} \\ &= \min\{0 + 0 + 30 \cdot 35 \cdot 15\} = 15750, & p_{1,2} &= 1, \\ m_{2,3} &= \min\{m_{2,2} + m_{3,3} + k_1 \cdot k_2 \cdot k_3\} \\ &= \min\{0 + 0 + 35 \cdot 15 \cdot 5\} = 2625, & p_{2,3} &= 2, \\ m_{3,4} &= \min\{m_{3,3} + m_{4,4} + k_2 \cdot k_3 \cdot k_4\} \\ &= \min\{0 + 0 + 15 \cdot 5 \cdot 10\} = 750, & p_{3,4} &= 3, \\ m_{4,5} &= \min\{m_{4,4} + m_{5,5} + k_3 \cdot k_4 \cdot k_5\} \\ &= \min\{0 + 0 + 5 \cdot 10 \cdot 20\} = 1000, & p_{4,5} &= 4, \end{aligned}$$

$j - i = 2 :$

$$\begin{aligned}
 m_{1,3} &= \min\{m_{1,1} + m_{2,3} + k_0 \cdot k_1 \cdot k_3, m_{1,2} + m_{3,3} + k_0 \cdot k_2 \cdot k_3\} \\
 &= \min\{0 + 2625 + 30 \cdot 35 \cdot 5, 15750 + 0 + 30 \cdot 15 \cdot 5\} \\
 &= \min\{7875, 18000\} = 7875, & p_{1,3} = 1, \\
 m_{2,4} &= \min\{m_{2,2} + m_{3,4} + k_1 \cdot k_2 \cdot k_4, m_{2,3} + m_{4,4} + k_1 \cdot k_3 \cdot k_4\} \\
 &= \min\{0 + 750 + 35 \cdot 15 \cdot 10, 2625 + 0 + 35 \cdot 5 \cdot 10\} \\
 &= \min\{6000, 4375\} = 4375, & p_{2,4} = 3, \\
 m_{3,5} &= \min\{m_{3,3} + m_{4,5} + k_2 \cdot k_3 \cdot k_5, m_{3,4} + m_{5,5} + k_2 \cdot k_4 \cdot k_5\} \\
 &= \min\{0 + 1000 + 15 \cdot 5 \cdot 20, 750 + 0 + 15 \cdot 10 \cdot 20\} \\
 &= \min\{1500, 3000\} = 1500, & p_{3,5} = 3,
 \end{aligned}$$

$j - i = 3 :$

$$\begin{aligned}
 m_{1,4} &= \min\{m_{1,1} + m_{2,4} + k_0 \cdot k_1 \cdot k_4, m_{1,2} + m_{3,4} + k_0 \cdot k_2 \cdot k_4, \\
 &\quad m_{1,3} + m_{4,4} + k_0 \cdot k_3 \cdot k_4\} \\
 &= \min\{0 + 4375 + 30 \cdot 35 \cdot 10, 15750 + 750 + 30 \cdot 15 \cdot 10, \\
 &\quad 7875 + 0 + 30 \cdot 5 \cdot 10\} \\
 &= \min\{14875, 21000, 9375\} = 9375, & p_{1,4} = 3, \\
 m_{2,5} &= \min\{m_{2,2} + m_{3,5} + k_1 \cdot k_2 \cdot k_5, m_{2,3} + m_{4,5} + k_1 \cdot k_3 \cdot k_5, \\
 &\quad m_{2,4} + m_{5,5} + k_1 \cdot k_4 \cdot k_5\} \\
 &= \min\{0 + 1500 + 35 \cdot 15 \cdot 20, 2625 + 1000 + 35 \cdot 5 \cdot 20, \\
 &\quad 4375 + 0 + 35 \cdot 10 \cdot 20\} \\
 &= \min\{12000, 7125, 11375\} = 7125, & p_{2,5} = 3,
 \end{aligned}$$

$j - i = 4 :$

$$\begin{aligned}
 m_{1,5} &= \min\{m_{1,1} + m_{2,5} + k_0 \cdot k_1 \cdot k_5, m_{1,2} + m_{3,5} + k_0 \cdot k_2 \cdot k_5, \\
 &\quad m_{1,3} + m_{4,5} + k_0 \cdot k_3 \cdot k_5, m_{1,4} + m_{5,5} + k_0 \cdot k_4 \cdot k_5\} \\
 &= \min\{0 + 7125 + 30 \cdot 35 \cdot 20, 15750 + 1500 + 30 \cdot 15 \cdot 20, \\
 &\quad 7875 + 1000 + 30 \cdot 5 \cdot 20, 9375 + 0 + 30 \cdot 10 \cdot 20\} \\
 &= \min\{28125, 26250, 11875, 15375\} = 11875, & p_{1,5} = 3.
 \end{aligned}$$

Μπορούμε να παρουσιάσουμε συγκεντρωτικά τους υπολογισμούς στους πίνακες

$$m = \begin{bmatrix} 0 & 15750 & 7875 & 9375 & 11875 \\ & 0 & 2625 & 4375 & 7125 \\ & & 0 & 750 & 1500 \\ & & & 0 & 1000 \\ & & & & 0 \end{bmatrix}, p = \begin{bmatrix} 1 & 1 & 1 & 3 & 3 \\ & 2 & 2 & 3 & 3 \\ & & 3 & 3 & 3 \\ & & & 4 & 4 \\ & & & & 5 \end{bmatrix}.$$

Επομένως μπορούμε να υπολογίσουμε το γινόμενο των πέντε πινάκων εκτελώντας μόλις 11875 πολλαπλασιασμούς.² Όμως με ποια σειρά πρέπει να εκτελέσουμε τους πολλαπλασιασμούς; Βλέπουμε ότι $p_{1,5} = 3$. Αυτό σημαίνει ότι $1,5 = 1,3 \cdot A_{4,5}$. Επειδή $p_{1,3} = 1$, έχουμε ότι $A_{1,3} = A_{1,1} \cdot A_{2,3}$ και άρα $A_{1,3} = A_1 \cdot (A_2 \cdot A_3)$. Αντίστοιχα, επειδή $p_{4,5} = 4$ έχουμε ότι $A_{4,5} = A_{4,4} \cdot A_{5,5} = A_4 \cdot A_5$. Επομένως η σειρά με την οποία πρέπει να πολλαπλασιάσουμε τους πίνακες για να επιτύχουμε τον ελάχιστο αριθμό πολλαπλασιασμών των στοιχείων τους είναι

$$(A_1 \cdot (A_2 \cdot A_3)) \cdot (A_4 \cdot A_5).$$

Πολυπλοκότητα

Η προηγούμενη ανάλυση οδηγεί στον Αλγόριθμο 48. Επειδή πέρα από τον ελάχιστο αριθμό των πολλαπλασιασμών μας ενδιαφέρει και να ιχνηλατήσουμε με ποια σειρά πρέπει να τους εκτελέσουμε προκειμένου να επιτύχουμε τον αριθμό αυτό, χρησιμοποιούμε ένα δισδιάστατο διάνυσμα p . Για $i < j$, η τιμή $p_{i,j}$ (στοιχείο $p[i, j]$ στον Αλγόριθμο 48) είναι η τιμή t για την οποία επιτυγχάνεται ο ελάχιστος αριθμός των πολλαπλασιασμών προκειμένου να υπολογιστεί ο πίνακας $A_{i,j}$ από την (8.11).

Για να υπολογίσουμε την πολυπλοκότητα του Αλγόριθμου 48, παρατηρούμε ότι στον πλέον εσωτερικό βρόχο γίνεται σταθερός αριθμός ΣΥΒ. Αν c είναι ο αριθμός αυτός, ο συνολικός αριθμός ΣΥΒ - αγνοώντας την αρχικοποίηση των δομών - δίνεται από τη σχέση

$$\begin{aligned} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \sum_{t=i}^{j-1} c &= \sum_{i=1}^{n-1} \sum_{j=i+1}^n (j-i)c = \sum_{i=1}^{n-1} \sum_{t=1}^{n-i} tc \\ &= c \cdot \sum_{i=1}^{n-1} \frac{(n-i)(n-i+1)}{2} = \frac{c}{2} \left(\sum_{i=1}^{n-1} (n-i)^2 + \sum_{i=1}^{n-1} (n-i) \right) \\ &= \frac{c}{2} \left(\sum_{i=1}^{n-1} i^2 + \sum_{i=1}^{n-1} i \right) = \Theta(n^3). \end{aligned}$$

Μέγιστη Κοινή Υπακολουθία

Ιδέα

Μια συμβολοσειρά $W = \langle w_1, w_2, \dots, w_k \rangle$ αποτελεί μία ακολουθία k συμβόλων (χαρακτήρων) όπου το πρώτο σύμβολο είναι το w_1 , το δεύτερο το w_2 , κοκ. Μια υπακολουθία W' της W είναι μία συμβολοσειρά η οποία προκύπτει από την W αν διαγραφούν από αυτή ένας οποιοσδήποτε αριθμός χαρακτήρων. Για παράδειγμα, αν $W = \langle X, A, I, P, E, T, I, S, M, A, T, A \rangle$, τότε οι συμβολοσειρές $\langle A, S, M, A, T, A \rangle$, $\langle A, I, M, A \rangle$, $\langle P, E, M, A \rangle$, $\langle A, P, A \rangle$, $\langle A, P, I, A \rangle$ αποτελούν υποακολουθίες της. Από το παράδειγμα είναι φανερό ότι οι χαρακτήρες μίας υπακολουθίας πρέπει να εμφανίζονται με την ίδια σειρά στην ακολουθία αλλά όχι απαραίτητα συνεχόμενα.

Το πρόβλημα με το οποίο θα ασχοληθούμε ορίζεται ως εξής: δεδομένων δύο ακολουθιών $X = \langle x_1, x_2, \dots, x_n \rangle$ και $Y = \langle y_1, y_2, \dots, y_m \rangle$ θέλουμε να βρούμε μία μέγιστου μήκους κοινή υπακολουθία. Εκτός των άλλων, το πρόβλημα αυτό βρίσκει εφαρμογή στον τομέα της γενετικής όπου δεδομένων δυο σειρών DNA, καθένα από τα οποία περιγράφει έναν οργανισμό, θέλουμε να καθορίσουμε το βαθμό ομοιότητας τους.

Τα πέντε στάδια του ΔΠ

- (Ορισμός υποπροβλημάτων) Για να ορίσουμε τα υποπροβλήματα, θα χρησιμοποιήσουμε τη λογική των επιθεμάτων. Έστω $X(i :) = \langle x_i, x_{i+1}, \dots, x_n \rangle$ και $Y(j :) = \langle y_j, y_{j+1}, \dots, y_m \rangle$. Συμβολίζουμε με $Z(i, j)$ το υποπρόβλημα που προκύπτει από το αρχικό αν θέλουμε να βρούμε τη μεγαλύτερη κοινή υπακολουθία των $X(i :), Y(j :)$. Θα συμβολίζουμε με $L_{i,j}$ το μήκος της υπακολουθίας αυτής. Δηλαδή, $L_{i,j}$ αποτελεί τη λύση, σε αριθμό χαρακτήρων, του υποπροβλήματος $Z(i, j)$.
- (Καθορισμός επιλογών) Όταν πάμε να επιλύσουμε το υποπρόβλημα $Z(i, j)$ υπάρχουν δύο περιπτώσεις: είτε $x_i \neq y_j$ ή $x_i = y_j$. Στην πρώτη περίπτωση υπάρχουν δύο επιλογές: είτε η μέγιστη κοινή υπακολουθία αποτελεί λύση του υποπροβλήματος $Z(i, j+1)$ ή του $Z(i+1, j)$. Στη δεύτερη περίπτωση δεν υπάρχει επιλογή: αφού ταιριάζουν οι χαρακτήρες x_i και y_j μένει να επιλυθεί το υποπρόβλημα $Z(i+1, j+1)$.
- (Αναδρομική σχέση). Στην πρώτη περίπτωση που εξετάσαμε προηγουμένως ($x_i \neq y_j$) υπάρχουν δύο επιλογές σε σχέση με την τιμή της $L_{i,j}$ και άρα θα πάρουμε την καλύτερη. Μαθηματικά, $L_{i,j} = \max\{L_{i+1,j}, L_{i,j+1}\}$. Στην δεύτερη περίπτωση ($x_i = y_j$), το μήκος της μεγαλύτερης κοινής συμβολοσειράς ισούται με την τιμή $L_{i+1,j+1}$ - λύση του υποπροβλήματος $Z(i+1, j+1)$ επαυξημένη κατά 1. Επίσης αν έχουν εξεταστεί όλοι οι χαρακτήρες της μίας από τις δύο συμβολοσειρές - περίπτωση κατά την οποία είτε $i = n+1$, ή $j = m+1$ - δεν υπάρχουν επιπλέον χαρακτήρες που να ανήκουν σε κοινή συμβολοσειρά. Άρα θέτουμε $L(n+1, j) = 0, j = 1, \dots, m+1$, και $L(i, m+1) = 0, i = 1, \dots, n+1$. Συνολικά,

$$L_{i,j} = \begin{cases} 0, & i = n+1 \text{ ή } j = m+1, \\ 1 + L_{i+1,j+1}, & x_i = y_j, \\ \max\{L_{i+1,j}, L_{i,j+1}\}, & x_i \neq y_j. \end{cases} \quad (8.13)$$

Παρατηρούμε ότι αυτή η αναδρομική σχέση διαφέρει από τις αναδρομικές σχέσεις που αναπτύχθηκαν στις προηγούμενες ενότητες. Η διαφορά έγκειται στο γεγονός ότι υπάρχει συνθήκη - σχέση ανάμεσα σε x_i, y_j - που υπαγορεύει στις λύσεις ποιών υποπροβλημάτων θα βασίζεται η λύση του παρόντος υποπροβλήματος.

- (Τοπολογική ταξινόμηση). Επειδή ακολουθήθηκε η θεώρηση των υποπροβλημάτων με τη λογική των επιθεμάτων η σειρά επίλυσης γίνεται με τους δείκτες i, j να αρχικοποιούνται στις τιμές n, m , αντίστοιχα, και να βαίνουν μειωμένοι. Δηλαδή η σειρά που θα επιλυθούν τα υποπροβλήματα είναι

$$\begin{aligned} Z(n, m) &\rightarrow Z(n, m-1) \rightarrow \dots \rightarrow Z(n, 1) \rightarrow Z(n-1, m) \rightarrow \\ &\dots \rightarrow Z(n-1, 1) \rightarrow Z(n-2, m) \rightarrow \dots \end{aligned}$$

- (Επίλυση του αρχικού προβλήματος) Το μήκος της μεγαλύτερης κοινής υπακολουθίας είναι η τιμή $L_{1,1}$. Η υπακολουθία μπορεί να ιχνηλατηθεί αν η επιλογή που γίνεται κάθε φορά για τον υπολογισμό της λύσης από την (8.13), καταχωρείται σε ένα δισδιάσταστο διάνυσμα p . Για να μπορέσουμε να εκφράσουμε τη λύση σαν ένα μονοπάτι σε γράφημα θα θεωρήσουμε ότι το στοιχείο $p_{i,j}$ θα παίρνει τις παρακάτω τιμές

$$p_{i,j} = \begin{cases} '↖', & \text{av } L_{i,j} = 1 + L_{i+1,j+1}, \\ '↑', & \text{av } L_{i,j} = L_{i+1,j}, \\ '←', & \text{av } L_{i,j} = L_{i,j+1}. \end{cases}$$

Παράδειγμα

Παράδειγμα 22. Θεωρούμε τις ακολουθίες $= < \Pi, O, \Lambda, \Lambda, A >$ και $= < P, O, \Lambda, A >$. Η μεγαλύτερη κοινή υπακολουθία μέσω ΔΠ προκύπτει από τους υπολογισμούς:

$$\begin{array}{ll} L_{5,4} = 1 + L_{6,5} = 1 + 0 = 1, & p_{5,4} = '↖', \\ L_{5,3} = \max\{L_{6,3}, L_{5,4}\} = \max\{0, 1\} = 1 & p_{5,3} = '←', \\ L_{5,2} = \max\{L_{6,2}, L_{5,3}\} = \max\{0, 1\} = 1 & p_{5,2} = '←', \\ L_{5,1} = \max\{L_{6,1}, L_{5,2}\} = \max\{0, 1\} = 1 & p_{5,1} = '←', \end{array}$$

$$\begin{array}{ll} L_{4,4} = \max\{L_{5,4}, L_{4,5}\} = \max\{1, 0\} = 1, & p_{4,4} = '↑', \\ L_{4,3} = 1 + L_{5,4} = 1 + 1 = 2 & p_{4,3} = '↖', \\ L_{4,2} = \max\{L_{5,2}, L_{4,3}\} = \max\{1, 2\} = 2 & p_{4,2} = '←', \\ L_{4,1} = \max\{L_{5,1}, L_{4,2}\} = \max\{1, 2\} = 2 & p_{4,1} = '←', \end{array}$$

$$\begin{array}{ll} L_{3,4} = \max\{L_{4,4}, L_{3,5}\} = \max\{1, 0\} = 1, & p_{3,4} = '↑', \\ L_{3,3} = 1 + L_{4,4} = 1 + 1 = 2 & p_{3,3} = '↖', \\ L_{3,2} = \max\{L_{4,2}, L_{3,3}\} = \max\{2, 2\} = 2 & p_{3,2} = '↑' \text{ ή } p_{3,2} = '←', \\ L_{3,1} = \max\{L_{4,1}, L_{3,2}\} = \max\{2, 2\} = 2 & p_{3,1} = '↑' \text{ ή } p_{3,1} = '←', \end{array}$$

$$\begin{array}{ll} L_{2,4} = \max\{L_{3,4}, L_{2,5}\} = \max\{1, 0\} = 1, & p_{2,4} = '↑', \\ L_{2,3} = \max\{L_{3,3}, L_{2,4}\} = \max\{2, 1\} = 2 & p_{2,3} = '↑' \\ L_{2,2} = 1 + L_{3,3} = 1 + 2 = 3 & p_{2,2} = '↖', \\ L_{2,1} = \max\{L_{3,1}, L_{2,2}\} = \max\{2, 3\} = 3 & p_{2,1} = '←', \end{array}$$

$$\begin{array}{ll} L_{1,4} = \max\{L_{2,4}, L_{1,5}\} = \max\{1, 0\} = 1, & p_{1,4} = '↑', \\ L_{1,3} = \max\{L_{2,3}, L_{1,4}\} = \max\{2, 1\} = 2 & p_{1,3} = '↑' \\ L_{1,2} = \max\{L_{2,2}, L_{1,3}\} = \max\{3, 2\} = 3 & p_{1,2} = '↑', \\ L_{1,1} = \max\{L_{2,1}, L_{1,2}\} = \max\{3, 3\} = 3 & p_{1,1} = '↑' \text{ ή } p_{1,1} = '←'. \end{array}$$

	P	O	Λ	A
Π	3 ↑	3 ↑	2 ↑	1 ↑
O	3 ↑	3 ↖	2 ↑	1 ↑
Λ	2 ↑	2 ↑	2 ↖	1 ↑
Λ	2 ↑	2 ↑	2 ↖	1 ↑
A	1 ↑	1 ↖	1 ↖	1 ↖

Πίνακας 8.5: Μέγιστη κοινή υπακολουθία - Παράδειγμα 22

Στον Πίνακα 8.5 το μήκος της μέγιστης κοινής υπακολουθίας σε κάθε υποπρόβλημα. Επίσης για κάθε υποπρόβλημα έχει απεικονιστεί και το αντίστοιχο στοιχείο του πίνακα p το οποίο δείχνει από ποιο (προηγούμενο) υποπρόβλημα προήρθε η λύση αυτή. Ακολουθώντας σε ανάποδη φορά το μονοπάτι (τα μονοπάτια) που καταλήγουν στο κελί της πρώτης γραμμής και πρώτης στήλης μπορούμε να βρούμε την κοινή συμβολοσειρά: κάθε γράμμα της υποδεικνύεται από μία από τις «διαγώνιες ακμές» του μονοπατιού. Πίνακας 8.5 υπάρχουν δύο μονοπάτια που καταλήγουν στο κελί της πρώτης γραμμής, πρώτης στήλης τα οποία εκκινούν από το κελί της γραμμής $n = 5$, στήλης $m = 4$ (κόκκινες ακμές στον πίνακα). Και στα δύο μονοπάτια οι διαγώνιες ακμές είναι ίδιες και υποδεικνύουν την υπακολουθία $< O, \Lambda, A >$ με πλήθος χαρακτήρων $L_{1,1} = 3$.

Πολυπλοκότητα

Η πολυπλοκότητα της μεθόδου μπορεί να υπολογιστεί πολύ εύκολα: ο αριθμός των υποπροβλημάτων είναι $n \cdot m$ και η επίλυση καθενός από αυτά απαιτεί σταθερό αριθμό ΣΥΒ. Επομένως η διαδικασία είναι τάξης $\Theta(n \cdot m)$.

07_ΑΠΛΗΣΤΟΙ ΑΛΓΟΡΙΘΜΟΙ / 10a_greedy_notes

JobsProg (Προγραμματισμός Εργασιών)

Εισαγωγή

Έστω ότι έχουμε ένα σύνολο εργασιών $T = \{1, \dots, n\}$. Η κάθε εργασία $t \in T$ χαρακτηρίζεται από έναν χρόνο έναρξης s_t καθώς και από ένα χρόνο περαίωσης f_t - θεωρούμε ότι $s_t < f_t$. Όλες οι εργασίες εκτελούνται σε μία μηχανή, μία εργασία κάθε χρονική στιγμή και κατά τρόπο ώστε αν έχει ξεκινήσει η επεξεργασία της εργασίας θα πρέπει να ολοκληρωθεί (δηλαδή δεν μπορεί να μείνει στη μέση και να ξεκινήσει μία επόμενη εργασία).

Ζητείται να βρεθεί το μεγαλύτερο σύνολο εργασιών T' που μπορούν να εκτελεστούν στη μηχανή.

Για τους σκοπούς της ανάλυσης θεωρούμε δύο από τις εργασίες, έστω $i, j \in T$, τέτοιες ώστε η i να προηγηθεί της εργασίας j στην επεξεργασία μέσω της μηχανής. Για να μπορεί να συμβεί αυτό, θα πρέπει $f_i < s_j$. Στην περίπτωση αυτή λέμε ότι οι εργασίες i, j είναι συμβατές. Ένα υποσύνολο εργασιών του T που αποτελείται από συμβατές μεταξύ τους εργασίες ονομάζεται συμβατό. Το ζητούμενο είναι να βρούμε το συμβατό υποσύνολο του T με το μεγαλύτερο πληθύριθμο.

Έστω δύο συμβατές εργασίες i, j . Μία άλλη εργασία k μπορεί να παρεμβληθεί ανάμεσα στις i, j αν

$$f_i \leq s_k \text{ και } f_k \leq s_j. \quad (9.1)$$

Με βάση την (9.1) μπορούμε να ορίσουμε το σύνολο των εργασιών

$$T_{i,j} = \{k \in T : f_i \leq s_k \text{ και } f_k \leq s_j\}. \quad (9.2)$$

Δηλαδή το σύνολο $T_{i,j}$ περιέχει κάθε εργασία η οποία μπορεί να παρεμβληθεί ανάμεσα στις i, j .

Για απλότητα μπορούμε να θεωρήσουμε ότι οι εργασίες δεικτοδοτούνται κατά αύξουσα σειρά σε σχέση με τον χρόνο περαίωσης. Δηλαδή, ισχύει η σειρά

$$f_1 \leq f_2 \leq \dots \leq f_n. \quad (9.3)$$

Κάτω από την υπόθεση αυτή ισχύει ότι το σύνολο $T_{i,j} = \emptyset$ για $j \leq i + 1$.

Ορίζουμε ως $A(i, j)$ το πρόβλημα εύρεσης του συνόλου συμβατών εργασιών με τον μεγαλύτερο πληθύριθμο για τις οποίες ισχύει ότι έπονται της i και προηγούνται της j . Συμβολίζουμε με $A_{i,j}$ το σύνολο αυτό - δηλαδή το $A_{i,j}$ είναι το σύνολο με το μεγαλύτερο αριθμό συμβατών εργασιών από αυτές που ανήκουν στο $T_{i,j}$. Παρατηρήστε ότι μπορεί να υπάρχουν παραπάνω από ένα σύνολα $A_{i,j}$.

Αν $T_{i,j} \neq \emptyset$ για οποιοδήποτε στοιχείο $k \in T_{i,j}$ ισχύει ότι

$$T_{i,j} = T_{i,k} \cup \{k\} \cup T_{k,j}. \quad (9.4)$$

Επίσης μπορούμε να θεωρήσουμε δύο ψευδοεργασίες, την υπ' αριθμόν 0 και την υπ' αριθμόν $n + 1$ για τις οποίες ισχύει ότι $f_0 < s_1$ και $s_{n+1} > f_n$. Οπότε $T = T_{0,n+1}$ και το αρχικό πρόβλημα περιγράφεται ως $A(0, n + 1)$. η λύση του, ονομαστικά $A_{0,n+1}$, είναι το σύνολο που περιέχει το μεγαλύτερο αριθμό συμβατών εργασιών που ανήκουν στο $T_{0,n+1}$ (και επομένως στο T). Η επόμενη πρόταση διατυπώνει δύο ιδιότητες κρίσιμες για την ανάλυση του προβλήματος.

Λήμμα

Λήμμα 6. Έστω εργασίες $i, j \in T$ τέτοιες ώστε $T_{i,j} \neq \emptyset$ και

$$r = \operatorname{argmin}\{f_k : k \in T_{i,j}\}. \quad (9.5)$$

1. υπάρχει σύνολο $A_{i,j}$ τέτοιο ώστε $r \in A_{i,j}$,
2. $T_{i,r} = \emptyset$.

Απόδειξη.

1. Έστω $t^* = \operatorname{argmin}\{f_t : t \in A_{i,j}\}$. Αν $t^* \neq r$ τότε μπορούμε να αφαιρέσουμε από το $A_{i,j}$ την εργασία t^* και να προσθέσουμε την εργασία r χωρίς να δημιουργηθεί ασυμβατότητα.
2. Εφόσον η εργασία r είναι αυτή που ολοκληρώνεται νωρίτερα από κάθε άλλη εργασία του συνόλου $T_{i,j}$ δεν υπάρχει άλλη εργασία που μπορεί να παρεμβληθεί ανάμεσα σε αυτή και την εργασία i . Επομένως $T_{i,r} = \emptyset$.

□

Πόρισμα

Πόρισμα 7. Έστω συμβατές εργασίες $i, j \in T$. Τότε

$$T_{i,j} = \{r\} \cup T_{r,j},$$

όπου το r δίνεται από την (9.5).

Απόδειξη. Εφόσον $r \in T_{i,j}$ και $T_{i,r} = \emptyset$ (Λήμμα 6), από την (9.4), έχουμε

$$_{i,j} = T_{i,r} \cup \{r\} \cup T_{r,j} = \emptyset \cup \{k\} \cup T_{r,j} = \{r\} \cup T_{r,j}.$$

□

Βέλτιστη Υποδομή

Λήμμα 7. Αν $i, j \in T$ αποτελούν συμβατές εργασίες, τότε υπάρχει σύνολο $A_{i,j}$ που αποτελεί λύση του προβλήματος $A(i, j)$ για το οποίο ισχύει ότι

$$A_{i,j} = \{r\} \cup A_{r,j}, \quad (9.6)$$

όπου το r δίνεται από την (9.5).

Απόδειξη. Από το Λήμμα 6, γνωρίζουμε ότι για κάποιο $A_{i,j}$ ισχύει ότι $r \in A_{i,j}$. Έστω ότι το $A_{i,j} \setminus \{r\}$ δεν αποτελεί βέλτιστη λύση για το πρόβλημα $A(r, j)$. Άρα οποιοδήποτε σύνολο $A_{r,j}$ το οποίο αποτελεί λύση του προβλήματος $A(r, j)$ περιέχει περισσότερες εργασίες από το $A_{i,j} \setminus \{r\}$. Δηλαδή, έχουμε

$$|A_{r,j}| > |A_{i,j} \setminus \{r\}|. \quad (9.7)$$

Παρατηρούμε ότι κάθε εργασία στο σύνολο $A_{r,j}$ είναι συμβατή με την r . Επομένως το σύνολο $\{r\} \cup A_{r,j}$ περιέχει συμβατές εργασίες για το πρόβλημα με σύνολο εργασιών το $\{r\} \cup T_{r,j} = T_{i,j}$ - η ισότητα προκύπτει από το Πόρισμα 7. Όμως αυτό έρχεται σε αντίφαση με το ότι το σύνολο $A_{i,j}$ είναι η βέλτιστη λύση για το πρόβλημα $A(i, j)$ αφού από την (9.7) έχουμε ότι

$$|A_{r,j} \cup \{r\}| > |(A_{i,j} \setminus \{r\}) \cup \{r\}| = |A_{i,j}|.$$

□

Ιδέα

Το Λήμμα 7 λέει ότι η βέλτιστη λύση του υποπροβλήματος $A(i, j)$ περιέχει τη βέλτιστη λύση του υποπροβλήματος $A(r, j)$, όπου $i < r$ και r δίνεται από την (9.5). Δηλαδή, το πρόβλημα έχει την ιδιότητα της βέλτιστης υποδομής όταν χρησιμοποιείται η επιλογή του συντομότερου χρόνου περαίωσης. Περαιτέρω, βλέπουμε ότι η βέλτιστη λύση του αρχικού προβλήματος μπορεί να προκύψει αν θέσουμε αρχικά $i = 0, j = n + 1$ και επαγωγικά εφαρμόσουμε την (9.6). Δηλαδή, κάνοντας την άπληστη επιλογή - ανάμεσα σε συμβατές εργασίες - σε σχέση με το μικρότερο χρόνο περαίωσης επαναληπτικά μπορούμε να κατασκευάσουμε τη (βέλτιστη) λύση του αρχικού προβλήματος. Ο Αλγόριθμος 49 - τάξης $O(n)$ - υλοποιεί την ιδέα. Ο αλγόριθμος παράγει σαν έξodo το σύνολο S καθώς και τον πληθάριθμο του - μεταβλητή $m + 1$. Παρατηρούμε ότι το S αρχικοποιείται από την εργασία 1 και επαυξάνεται από την εργασία $j = \operatorname{argmin}\{f_k : k \in T_{i,n+1}\}$, όπου i είναι η εργασία που είχε προστεθεί στο S πιο πρόσφατα.

Αλγόριθμος

Algorithm 49 Εκτύπωση μεγαλύτερου συνόλου εργασιών που μπορούν να εκτελεστούν στη μηχανή.

Απαιτείται: Ακέραιος n , πίνακες s, f που περιέχουν τους χρόνους εκκίνησης και περαίωσης, αντίστοιχα, n εργασίες. Οι εργασίες είναι δεικτοδοτημένες κατά αύξουσα σειρά των χρόνων περαίωσης: $f[1] \leq f[2] \leq \dots \leq f[n]$.

Επιστρέφεται: Το πλήθος του μεγαλύτερου αριθμού των εργασιών που μπορούν να εκτελεστούν από τη μηχανή και γίνεται εκτύπωση τους.

```
1: function JobsProg(int n, int s[], int f[])
2:   i  $\leftarrow 1$ ; m  $\leftarrow 0$ ;
3:   S  $\leftarrow \{1\}$ ;
4:   for j  $\leftarrow 2$ ; j  $\leq n$ ; j  $\leftarrow \leftarrow$  do
5:     if  $s[j] \geq f[i]$  then
6:       S  $\leftarrow S \cup \{j\}$ ;
7:       i  $\leftarrow j$ ;
8:       m  $\leftarrow \leftarrow$ ;
9:     end if
10:   end for
11:   print S;
12:   return m + 1;
13: end function
```

Απληστη Επιλογή

Λήμμα 8. Ο Αλγόριθμος 49 υπολογίζει το μέγιστο αριθμό των εργασιών που μπορούν να εκτελεστούν στη μηχανή.

Απόδειξη. Από το Λήμμα 7 έχουμε ότι

$$\begin{aligned} A_{0,n+1} &= \{1\} \cup A_{1,n+1} = \{1, r_1\} \cup A_{r_1,n+1} \\ &= \{1, r_1, r_2\} \cup A_{r_2,n+1} = \dots = \{1, r_1, r_2, \dots, r_m\} \cup A_{r_m,n+1} \\ &= \bigcup_{i=0}^m \{r_i\} \cup A_{r_m,n+1} \end{aligned} \quad (9.8)$$

όπου

$$r_i = \begin{cases} 1, & i = 0, \\ \operatorname{argmin}\{f_k : k \in T_{r_{i-1},n+1}\}, & m \geq i > 0, \end{cases}$$

και $m = \operatorname{argmax}\{i : T_{r_{i-1}, n+1} \neq \emptyset\}$. Από τον ορισμό της παραμέτρου m είναι προφανές ότι $A_{r_m, n+1} = \emptyset$ αφού $T_{r_m, n+1} = \emptyset$. Από την (9.8) έχουμε

$$A_{0, n+1} = \{1, r_1, r_2, \dots, r_m\}$$

Παρατηρούμε ότι τα στοιχεία του συνόλου αυτού είναι ακριβώς αυτά που επιλέγει ο αλγόριθμος για να «χτίσει» το S . \square

Παράδειγμα

Παράδειγμα 23. Θεωρούμε 5 εργασίες με χρόνους έναρξης-περαίωσης όπως απεικονίζονται στον Πίνακα 9.1 Παρατηρούμε ότι οι εργασίες έχουν διαταχθεί σε αύξουσα σειρά σε σχέση με το χρόνο περαίωσης. Εκτελώντας τον Αλγόριθμο 49 βλέπουμε ότι η πρώτη εργασία που θα επιλεγεί είναι η 1, δεύτερη η 4 και τρίτη η 5.

Εναλλακτικά, για να προγραμματίσουμε τις εργασίες στη μηχανή προκειμένου να εκτελεστεί ο μεγαλύτερος αριθμός αυτών θεωρούμε τις ψευδοεργασίες 0 και 6 με $f_0 = 0$ $s_6 = \infty$. Θέλουμε να βρούμε τη βέλτιστη λύση στο πρόβλημα $A(0, 6)$. Η λύση δίνεται από την αναδρομική εφαρμογή της (9.6) ήτοι

$$\begin{aligned} A_{0,6} &= \{r_0 = \operatorname{argmin}\{f_k, k \in T_{0,6}\}\} \cup A_{r_0,6} = \{1\} \cup A_{1,6} \\ &= \{1\} \cup (\{r_1 = \operatorname{argmin}\{f_k, k \in T_{1,6}\}\} \cup A_{r_1,6}) = \{1\} \cup (\{4\} \cup A_{4,6}) \\ &= \{1, 4\} \cup (\{r_2 = \operatorname{argmin}\{f_k, k \in T_{4,6}\}\} \cup A_{r_2,6}) = \{1, 4\} \cup (\{5\} \cup A_{5,6}) \\ &= \{1, 4, 5\} \cup \emptyset = \{1, 4, 5\}. \end{aligned}$$

Η παραπάνω επίλυση αποτελεί εξειδίκευση της διαδικασίας απόδειξης του Λήμματος 8 - στη συγκεκριμένη περίπτωση έχουμε $m = 2$.

Μία βασική παρατήρηση είναι ότι η ιδιότητα της άπληστης επιλογής ισχύει για το πρόβλημα μόνο αν εφαρμόσουμε το κριτήριο επιλογής σε σχέση με το μικρότερο χρόνο περαίωσης. Αν επιλεγεί διαφορετικό κριτήριο - π.χ. επιλογή με βάση με τον μικρότερο χρόνο έναρξης - δεν είναι απαραίτητο ότι θα ικανοποιείται η ιδιότητα αυτή. Αν για παράδειγμα, επιλέξουμε άπληστα με βάση τον μικρότερο χρόνο έναρξης επειδή δεν ισχύει η ιδιότητα της άπληστης επιλογής καταλήγουμε σε μία λύση - σύνολο $\{2, 5\}$ - που δεν περιέχει το μεγαλύτερο αριθμό εργασιών που μπορούν να εκτελεστούν στη μηχανή.

Χρόνος Παραμονής σε Κατάστημα

Ιδέα

Σε ένα κατάστημα οι πελάτες πληρώνουν σε ένα ταμείο. Έστω ότι ένα σύνολο πελατών $I = \{1, \dots, n\}$ συσσωρεύεται στο ταμείο και είναι γνωστό εκ' των προτέρων ότι ο πελάτης $i \in I$ για να απασχολήσει το ταμείο t_i λεπτά. Θέλουμε να βρούμε τη σειρά με την οποία θα πρέπει να εξυπηρετηθούν οι πελάτες έτσι ώστε ο μέσος χρόνος αναμονής να είναι ο μικρότερος δυνατός. Για να κατανοήσουμε καλύτερα το πρόβλημα θεωρούμε το επόμενο παράδειγμα.

Παράδειγμα

Παράδειγμα 24. Έστω $I = \{1, 2, 3\}$ με χρόνο εξυπηρέτησης (σε λεπτά) $t_1 = 1, t_2 = 5, t_3 = 3$. Αν η σειρά εξυπηρέτησης είναι $3-2-1$ τότε ο πελάτης 3 θα περιμένει 3 λεπτά ο πελάτης 2 $3+5 = 8$ λεπτά και ο πελάτης 3 $5+3+2 = 10$ λεπτά. Στην περίπτωση αυτή ο μέσος χρόνος αναμονής στο κατάστημα είναι το άθροισμα των χρόνων αναμονής δια του συνολικού αριθμού των πελατών, ήτοι $(3 + 8 + 10)/3 = 7$. Αν θεωρήσουμε ως σειρά εξυπηρέτησης την $2-1-3$ τότε πρώτα θα εξυπηρετηθεί ο πελάτης 2 με χρόνο αναμονής 5 λεπτά, στη συνέχεια ο πελάτης 1 με χρόνο αναμονής $5 + 1 = 6$ λεπτά και στο τέλος ο πελάτης 3 με χρόνο αναμονής $5 + 6 + 3 = 14$ λεπτά. Επομένως ο μέσος χρόνος αναμονής, στην περίπτωση αυτή, είναι $(5 + 6 + 14)/3 = 25/3 = 8,333$ λεπτά. Είναι φανερό ότι διαφορετική σειρά εξυπηρέτησης οδηγεί σε διαφορετικό μέσο χρόνο αναμονής στο κατάστημα. Αναζητούμε λοιπόν από τους $3! = 6$ διαφορετικούς τρόπους εξυπηρέτησης (σειρές εξυπηρέτησης) αυτόν ο οποίος ελαχιστοποιεί το μέσο χρόνο αναμονής στο κατάστημα.

Απληστη Επιλογή

Μία βασική παρατήρηση είναι ότι αφού ο αριθμός των πελατών είναι ο ίδιος σε κάθε πιθανή σειρά εξυπηρέτησης το πρόβλημα είναι ισοδύναμο με αυτό της εύρεσης της σειράς που ελαχιστοποιεί το συνολικό χρόνο αναμονής στο κατάστημα.

Ένα προφανές κριτήριο για την επίλυση του παραπάνω προβλήματος είναι να δίνεται προτεραιότητα κάθε φορά στον πελάτη με το μικρότερο χρόνο εξυπηρέτησης. Η εφαρμογή του κριτήριου είναι πολύ εύκολη: οι πελάτες εξυπηρετούνται με τη σειρά $i_1 - i_2 - \dots - i_n$ όπου $\{i_1, i_2, \dots, i_n\} = I$ και $t_{i_1} \leq t_{i_2} \leq \dots \leq t_{i_n}$. Η εξυπηρέτηση με τη σειρά αυτή περιγράφει έναν άπληστο αλγόριθμο. Παρατηρούμε ότι ο αλγόριθμος αυτός οδηγεί σε μία σειρά (λύση) στην οποία όλοι οι πελάτες εξυπηρετούνται. Ο συνολικός χρόνος αναμονής στο κατάστημα με βάση αυτή τη σειρά εξυπηρέτησης δίνεται από τον τύπο

$$\begin{aligned} T &= t_{i_1} + (t_{i_1} + t_{i_2}) + \dots + (t_{i_1} + t_{i_2} + \dots + t_{i_{n-1}}) + (t_{i_1} + t_{i_2} + \dots + t_{i_n}) \\ &= n \cdot t_{i_1} + (n-2) \cdot t_{i_2} + \dots + t_{i_n} \\ &= \sum_{j=1, \dots, n} (n-j+1)t_{i_j} \end{aligned} \tag{9.9}$$

Βέλτιστη Υποδομή

Πρόταση 3. Οποιαδήποτε σειρά εξυπηρέτησης έχει συνολικό χρόνο αναμονής στο κατάστημα μεγαλύτερο-ίσο με το χρόνο που προκύπτει από την (9.9).

Απόδειξη. Θεωρούμε δύο πελάτες i_v, i_u με $v < u$ (και άρα $t_{i_v} \leq t_{i_u}$) όπου ο i_u εξυπηρετείται πριν τον i_v . Αν οι υπόλοιποι πελάτες εξυπηρετούνται με αύξουσα σειρά σε σχέση με το χρόνο εξυπηρέτησης, έχουμε το συνολικό χρόνο αναμονής στο σύστημα

$$T' = (n-v+1)t_{i_u} + (n-u+1)t_{i_v} + \sum_{j \in I \setminus \{v, u\}} t_{i_j} \tag{9.10}$$

Αφαιρώντας τη (9.10) από τη (9.9) παίρνουμε

$$\begin{aligned} T - T' &= (n-v+1)(t_{i_v} - t_{i_u}) + (n-u+1)(t_{i_u} - t_{i_v}) \\ &= (n-v+1)(t_{i_v} - t_{i_u}) - (n-u+1)(t_{i_v} - t_{i_u}) \\ &= (u-v)(t_{i_v} - t_{i_u}) \leq 0, \end{aligned}$$

αφού $u-v > 0$ και $t_{i_v} - t_{i_u} \leq 0$. Άρα $T \leq T'$.

Εφαρμόζοντας επαγωγικά την αλλαγή στη προτεραιότητα εξυπηρέτησης σε οποιοδήποτε ζευγάρι πελατών i_v, i_u , με $v < u$, μπορούμε να παράγουμε οποιαδήποτε άλλη σειρά εξυπηρέτησης από τις $n!$ σειρές. Όμως ο συνολικός χρόνος αναμονής θα είναι στην καλύτερη περίπτωση ίσος με το χρόνο αναμονής της σειράς που παράγει ο άπληστος αλγόριθμος. \square

Παράδειγμα

Επομένως για το στιγμιότυπο του Παραδείγματος 24 η σειρά εξυπηρέτησης που ελαχιστοποιεί το μέσο χρόνο αναμονής είναι $1 - 3 - 2$ με μέσο χρόνο εξυπηρέτησης $\frac{1 \cdot 3 + 3 \cdot 2 + 5 \cdot 1}{3} = \frac{14}{3} = 4,666$.

Huffman (Απροθηματικοί κώδικες)

Πρόβλημα

Ένα από τα βασικότερα προβλήματα στη συμπίεση δεδομένων (data compression) είναι αυτό της αναπαράστασης μίας ακολουθίας χαρακτήρων με τον ελάχιστο αριθμό δυαδικών ψηφίων. Για να επιτύχουμε κάτι τέτοιο μπορούμε να αντιστοιχίσουμε κάθε χαρακτήρα σε μία διαφορετική δυαδική συμβολοσειρά (κωδικολέξη). Για παράδειγμα,

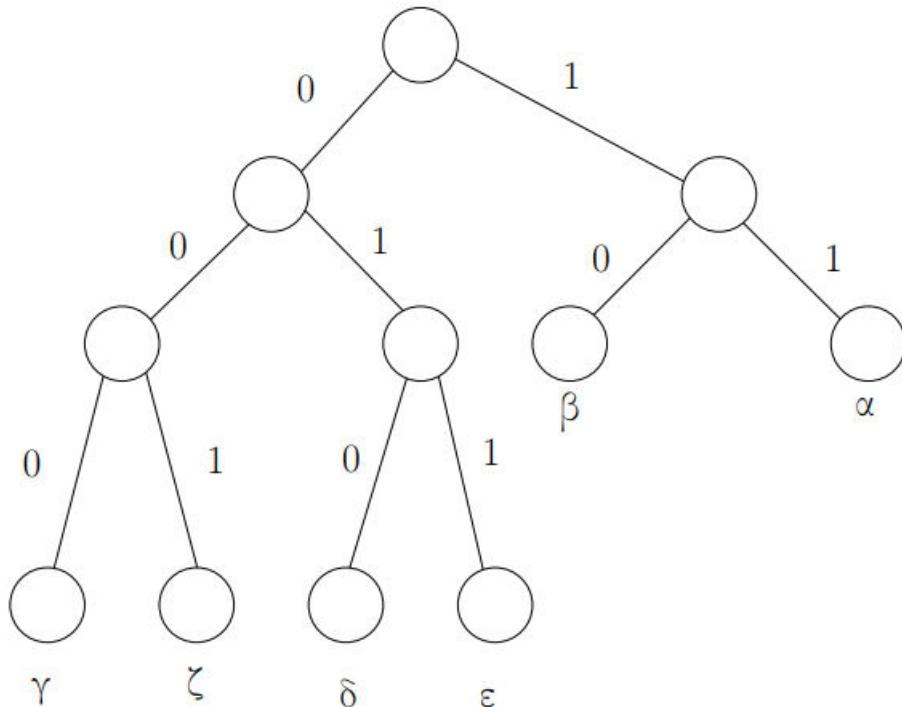
Παράδειγμα

αν χρησιμοποιούνται μόνο οι χαρακτήρες από το σύνολο $\{\alpha, \beta, \gamma, \delta, \varepsilon, \zeta\}$, μπορούμε να αντιστοιχίσουμε τα γράμματα αυτά σε συμβολοσειρές σταθερού μήκους τριών ψηφίων όπως αυτές που απεικονίζονται στην τρίτη γραμμή του Πίνακα 9.2. Αν οι συχνότητες που εμφανίζονται οι χαρακτήρες σε κάποιο κείμενο είναι αυτές που απεικονίζονται στη δεύτερη γραμμή του πίνακα, τότε ο αριθμός των δυαδικών ψηφίων που κωδικοποιεί το κείμενο χρησιμοποιώντας την κωδικοποίηση αυτή είναι

$$(344 + 96 + 12 + 16 + 9 + 5) \cdot 3 = 1446.$$

Εναλλακτικά, μπορούμε να χρησιμοποιήσουμε την κωδικοποίηση μεταβλητού μήκους που παρουσιάζεται στην τελευταία γραμμή του πίνακα. Τότε το κείμενο κωδικοποιείται σε μία δυαδική συμβολοσειρά με μήκος

$$344 \cdot 2 + 96 \cdot 2 + 12 \cdot 3 + 16 \cdot 3 + 9 \cdot 3 + 5 \cdot 3 = 1006.$$



Σχήμα 9.1: Κωδικοποίηση μεταβλητού μήκους Πίνακα 9.2

Παρατηρήσεις

Παρατηρούμε ότι η κωδικοποίηση των χαρακτήρων στην τελευταία γραμμή του πίνακα είναι απροθηματική: καμία συμβολοσειρά δεν αποτελεί πρόθημα κάποιας άλλης. Πλέονέκτημα της χρήσης μίας τέτοιας κωδικοποίησης είναι η εύκολα αποκωδικοποίηση του κειμένου. Δεδομένου ότι καμία κωδικολέξη δεν συμπίπτει με το πρόθεμα κάποιας άλλης η εναρκτήρια κωδικολέξη σε ένα κωδικοποιημένο κείμενο είναι μονοσήμαντα ορισμένη. Επομένως μπορούμε απλώς να προσδιορίσουμε την πρώτη κωδικολέξη και αφού την αποκωδικοποίησουμε να επαναλάβουμε τη διαδικασία για το υπόλοιπο κωδικοποιημένο κείμενο.

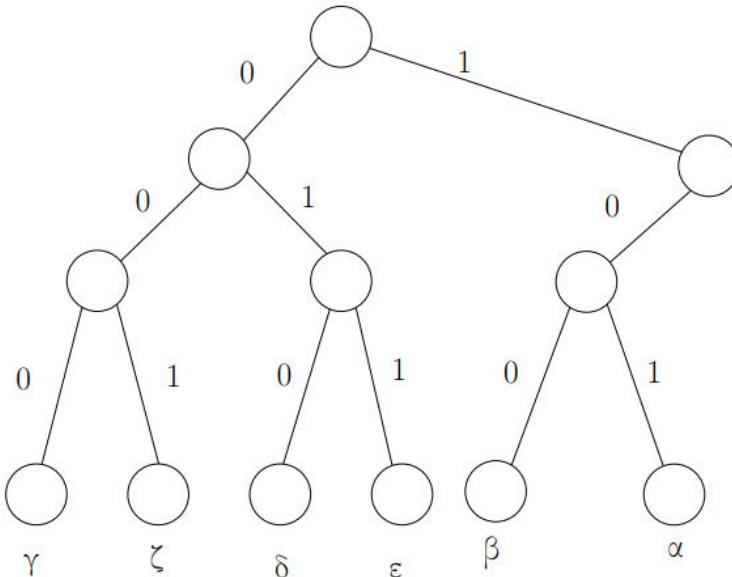
	α	β	γ	δ	ε	ζ
Συχνότητα	344	96	12	16	9	5
Σταθ. Μήκος	000	001	010	011	100	101
Μετ. Μήκος	11	10	000	010	011	001

Πίνακας 9.2: Απροθηματική Κωδ. Σταθ. και Μετ. Μήκους

Για τη διαδικασία αποκωδικοποίησης μπορούμε να χρησιμοποιήσουμε μία αναπαράσταση του σχήματος κωδικοποίησης η οποία θα μας δίνει τη δυνατότητα να εντοπίσουμε με ευκολία την πρώτη κωδικολέξη. Μία τέτοια αναπαράσταση είναι μέσω ενός δυαδικού δένδρου όπου τα φύλλα του αντιστοιχούν στα γράμματα. Σε ένα τέτοιο δένδρο η δυαδική κωδικολέξη ενός χαρακτήρα συνίσταται στη απλή διαδρομή από τη ρίζα του δένδρου μέχρι το αντίστοιχο φύλλο, όπου το 0 σημαίνει μετάβαση κατά μήκος της αριστερής ακμής και 1 μέσω της δεξιάς. Για παράδειγμα, το δένδρο που αντιστοιχεί στην κωδικοποίηση μεταβλητού μήκους του Πίνακα 9.2 απεικονίζεται στο Σχήμα 9.1. Με αντίστοιχο τρόπο η κωδικοποίηση σταθερού μήκους του Πίνακα 9.2 απεικονίζεται στο Σχήμα 9.2.

Παρατηρούμε ότι το δένδρο του Σχήματος 9.1 είναι πλήρες δυαδικό ενώ αυτό του Σχήματος 9.2 είναι μεν δυαδικό αλλά όχι πλήρες δυαδικό. Γενικά ισχύει ότι ένα βέλτιστο σχήμα κωδικοποίησης - δηλαδή μία κωδικοποίηση βάσει της οποίας το παραγόμενο κείμενο έχει το μικρότερο αριθμό bits - αναπαρίσταται πάντα με ένα πλήρες δυαδικό δένδρο. Επομένως για να βρούμε ένα βέλτιστο σχήμα κωδικοποίησης θα πρέπει να εξετάσουμε απροθηματικά σχήματα που αναπαρίστανται με πλήρη δυαδικά δένδρα.

Αν γνωρίζουμε το δένδρο T που αντιστοιχεί σε ένα σχήμα κωδικοποίησης κάποιου συνόλου χαρακτήρων C , μπορούμε να υπολογίσουμε το μήκος της συμβολοσειράς του



Σχήμα 9.2: Κωδικοποίηση σταθερού μήκους Πίνακα 9.2

κωδικοποιημένου κειμένου. Αυτό δίνεται από τον τύπο

$$B(T) = \sum_{c \in C} f(c)d_T(c),$$

όπου $f(c)$ είναι η συχνότητα εμφάνισης του χαρακτήρα c στο κείμενο και $d_T(c)$ το βάθος του φύλλου - αριθμός ακμών στο μονοπάτι από την κορυφή αυτή μέχρι την ρίζα - που αντιστοιχεί στον χαρακτήρα c . Η ποσότητα $B(T)$ ονομάζεται κόστος του δένδρου T . Η μέθοδος Huffman [9] υπολογίζει το δένδρο με το ελάχιστο κόστος. Για να την περιγράψουμε ορίζουμε:

\mathcal{T} : ένα σύνολο από δένδρα με φύλλα (κάποιους από) τους χαρακτήρες του αλφάβητου C ,

T_S : ένα δυαδικό δένδρο με φύλλα τους χαρακτήρες του συνόλου $S \subseteq C$,

\oplus : ένας τελεστής ο οποίος όταν εφαρμόζεται στα δένδρα T_{S_1}, T_{S_2} , όπου $S_1 \cup S_2 \subseteq C, S_1 \cap S_2 = \emptyset$, τα συνενώνει: δημιουργεί ένα δυαδικό δένδρο $T_{S_1 \cup S_2}$ υπάγοντας τις δύο ρίζες των T_{S_1}, T_{S_2} σε κοινή ρίζα.

Επίσης για κάθε δυαδικό δένδρο, η συχνότητα μίας κορυφής v ορίζεται αναδρομικά ως

$$f(v) = \begin{cases} f(c), & \text{αν } v \text{ είναι το φύλλο που} \\ & \text{αντιπροσωπεύει το χαρακτήρα } c \in C, \\ f(u) + f(w), & \text{αν } v \text{ είναι εσωτερική και οι κορυφές } u, w \\ & \text{είναι άμεσοι απόγονοι της } v. \end{cases} \quad (9.11)$$

Η συχνότητα ενός δένδρου T , ονομαστικά $f(T)$, είναι συχνότητα της ρίζας του.

Iδέα

Η μέθοδος Huffman κατασκευάζει ένα δένδρο κωδικοποίησης T (δένδρο Huffman) επιλέγοντας κάθε φορά, από ένα σύνολο δένδρων \mathcal{T} , να συνενώσει τα δύο δένδρα με τις μικρότερες ρίζες συχνότητες ρίζας. Στη συνέχεια αντικαθιστά τα δένδρα αυτά με το δένδρο που κατασκεύασε στο σύνολο \mathcal{T} . Αρχικοποιώντας το σύνολο \mathcal{T} με δένδρα που αποτελούνται από μία κορυφή, δηλαδή $\mathcal{T} = \cup_{c \in C} T_{\{c\}}$ και εφαρμόζοντας επαναληπτικά την διαδικασία που περιγράφηκε προηγουμένως, στο τέλος, καταλήγει το σύνολο \mathcal{T} να περιέχει μόνο ένα δένδρο. Αυτό αποτελεί την έξοδο του Αλγόριθμου 50 ο οποίος κωδικοποιεί τη μέθοδο.

Αλγόριθμος

Algorithm 50 Αλγόριθμος Huffman

Απαιτείται: Αλφάβητο C , πίνακας f όπου στη θέση c περιέχει την συχνότητα εμφάνισης του χαρακτήρα $c \in C$ στα κείμενο.

Επιστρέφεται: Δένδρο Huffman

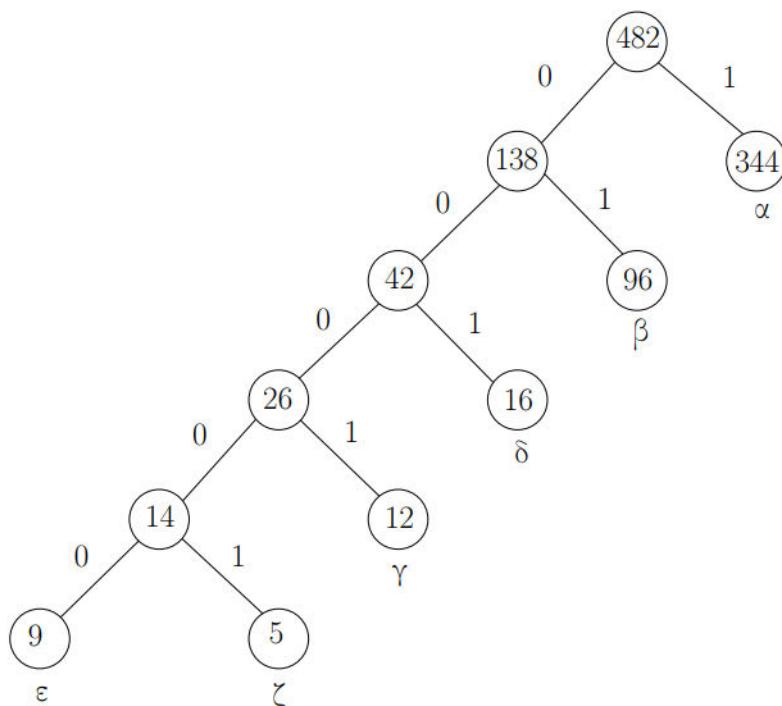
```
1: function Huffman(set  $C$ , int  $f[]$ )
2:    $\mathcal{T} \leftarrow \emptyset;$ 
3:   for all  $c \in C$  do
4:      $\mathcal{T} \leftarrow \mathcal{T} \cup \{T_{\{c\}}\};$ 
5:   end for
6:   while  $|\mathcal{T}| > 1$  do
7:      $T_{S_1} \leftarrow \text{argmin}\{f(T_S) : T_S \in \mathcal{T}\};$ 
8:      $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T_{S_1}\};$ 
9:      $T_{S_2} \leftarrow \text{argmin}\{f(T_S) : T_S \in \mathcal{T}\};$ 
10:     $\mathcal{T} \leftarrow \mathcal{T} \setminus \{T_{S_2}\};$ 
11:     $T \leftarrow T_{S_1} \oplus T_{S_2};$ 
12:     $\mathcal{T} \leftarrow \mathcal{T} \cup \{T\};$ 
13:   end while
14:   return  $T;$ 
15: end function
```

Παράδειγμα & Παρατηρήσεις

Το δένδρο Huffman για το στιγμιότυπο που περιγράφεται στον Πίνακα 9.2 - σε κάθε κορυφή απεικονίζεται η συχνότητα της από την (9.11) παρουσιάζεται στο Σχήμα 9.3. Το μήκος της δυαδικής συμβολοσειράς που κωδικοποιεί το κείμενο με βάση το σχήμα αυτό είναι

$$344 \cdot 1 + 96 \cdot 2 + 16 \cdot 3 + 12 \cdot 4 + 9 \cdot 5 + 5 \cdot 5 = 702.$$

Η μέθοδος Huffman είναι ένας άπληστος αλγόριθμος· σε κάθε επανάληψη συνενώνει τα δύο δένδρα με τη μικρότερη συχνότητα ρίζας. Θα αποδείξουμε ότι με βάση το κριτήριο αυτό ισχύουν τόσο η ιδιότητα της άπληστης επιλογής όσο και της βέλτιστης υποδομής.



Σχήμα 9.3: Δένδρο Huffman για το στιγμιότυπο του Πίνακα 9.2

Απληστη Επιλογή

Λήμμα 9. (Απληστη επιλογή) Έστω x', y' , οι χαρακτήρες του αλφάβητου C με τις μικρότερες συχνότητες. Υπάρχει βέλτιστο απροθηματικό δένδρο T για το αλφάβητο C τέτοιο ώστε οι χαρακτήρες αυτοί να αντιπροσωπεύονται από φύλλα μεγαλύτερου βάθους και έχουν κοινή κορυφή ως άμεσο πρόγονο.

Απόδειξη. Έστω T^* ένα βέλτιστο απροθηματικό δένδρο για το αλφάβητο C και x, y οι χαρακτήρες που αντιστοιχούν σε φύλλα μεγαλύτερου βάθους με κοινό άμεσο πρόγονο. Προφανώς αν $\{x, y\} = \{x', y'\}$ το δένδρο T^* είναι το ζητούμενο δένδρο T . Στην αντίθετη περίπτωση, μπορούμε να υποθέσουμε ότι $f(x) \leq f(y)$ και $f(x') < f(y')$. Θεωρούμε ως T το δένδρο που παράγεται από το T^* εναλλάσσοντας τον x με τον x'

στα φύλλα που τους αντιπροσωπεύουν και αντίστοιχα τον y με τον y' .

$$\begin{aligned} B(T^*) - B(T) &= \sum_{c \in C} f(c) \cdot d_{T^*}(c) - \sum_{c \in C} f(c) \cdot d_T(c) \\ &= f(x')(d_{T^*}(x') - d_T(x')) + f(y')(d_{T^*}(y') - d_T(y')) \\ &\quad + f(x)(d_{T^*}(x) - d_T(x)) + f(y)(d_{T^*}(y) - d_T(y)) \tag{9.12} \\ &= f(x')(d_{T^*}(x') - d_T(x')) + f(x)(d_{T^*}(x) - d_T(x)) \\ &\quad + f(y')(d_{T^*}(y') - d_T(y')) + f(y)(d_{T^*}(y) - d_T(y)). \end{aligned}$$

Επειδή

$$\begin{aligned} d_{T^*}(x) - d_T(x) &= -(d_{T^*}(x') - d_T(x')), \\ d_{T^*}(y) - d_T(y) &= -(d_{T^*}(y') - d_T(y')), \end{aligned}$$

η (9.12) γίνεται

$$\begin{aligned} B(T^*) - B(T) &= (f(x') - f(x))(d_{T^*}(x') - d_T(x')) \\ &\quad + (f(y') - f(y))(d_{T^*}(y') - d_T(y')). \tag{9.13} \end{aligned}$$

Λαμβάνοντας υπόψη ότι

$$f(x') \leq f(x), f(y') \leq f(y), d_{T^*}(x') \leq d_T(x'), d_{T^*}(y') \leq d_T(y'),$$

η (9.13) συνεπάγεται $B(T^*) - B(T) \geq 0$. Επειδή το T^* είναι βέλτιστο δένδρο κωδικοποίησης η τελευταία σχέση ισχύει σαν ισότητα. \square

Βέλτιστη Υποδομή

Λήμμα 10. (Βέλτιστη υποδομή) Έστω $x, y \in C$ οι δύο χαρακτήρες με τη μικρότερη συχνότητα. Θεωρούμε ένα νέο χαρακτήρα $z \notin C$ με $f(z) = f(x) + f(y)$, βάσει του οποίου ορίζουμε το αλφάβητο $C' = C \setminus \{x, y\} \cup \{z\}$. Αν T' είναι το βέλτιστο απροθηματικό δένδρο για το αλφάβητο C' , τότε το βέλτιστο απροθηματικό δένδρο, έστω T , για το αλφάβητο C προκύπτει από το T' αν η κορυφή που αντιστοιχεί στο χαρακτήρα z γίνει εσωτερική αποκτώντας δύο άμεσους απογόνους (φύλλα) που αντιστοιχούν στους χαρακτήρες x, y .

Απόδειξη. Παρατηρούμε ότι για κάθε $c \in C \setminus \{x, y\}$ ισχύει $f(c)d_T(c) = f(c)d_{T'}(c)$. Αθροίζοντας κατά μέλη, παίρνουμε

$$\sum_{c \in C \setminus \{x, y\}} f(c)d_T(c) = \sum_{c \in C \setminus \{x, y\}} f(c)d_{T'}(c) \quad (9.14)$$

Επίσης ισχύει ότι

$$\begin{aligned} f(x)d_T(x) + f(y)d_T(y) &= (f(x) + f(y))(d_{T'}(z) + 1) \\ &= f(z)d_{T'}(z) + f(x) + f(y). \end{aligned} \quad (9.15)$$

Προσθέτοντας τις (9.14), (9.15) έχουμε

$$B(T) = B(T') + f(x) + f(y) \Rightarrow B(T') = B(T) - f(x) - f(y). \quad (9.16)$$

Έστω ότι το δεν είναι βέλτιστο. Τότε υπάρχει βέλτιστο δένδρο T^* για το αλφάβητο C τέτοιο ώστε $B(T^*) < B(T)$. Σύμφωνα με το Λήμμα 9, μπορούμε να θεωρήσουμε ότι οι κορυφές που αντιστοιχούν στους χαρακτήρες x, y στο T^* έχουν κοινό άμεσο πρόγονο. Έστω T^{**} το δένδρο που προκύπτει από το T^* αν αντικαταστήσουμε την κορυφή που αποτελεί άμεσα πρόγονο από ένα φύλλο z με $f(z) = f(x) + f(y)$. Στην περίπτωση αυτή

$$B(T^{**}) = B(T^*) - f(x) - f(y) < B(T) - f(x) - f(y) = B(T'),$$

όπου η τελευταία ισότητα προκύπτει άμεσα από τη (9.16). Η ανισότητα $B(T^{**}) < B(T')$ οδηγεί σε άτοπο αφού το T' είναι εξ' υποθέσεως βέλτιστο για το αλφάβητο C' . Επομένως το T είναι βέλτιστο για το αλφάβητο C . \square

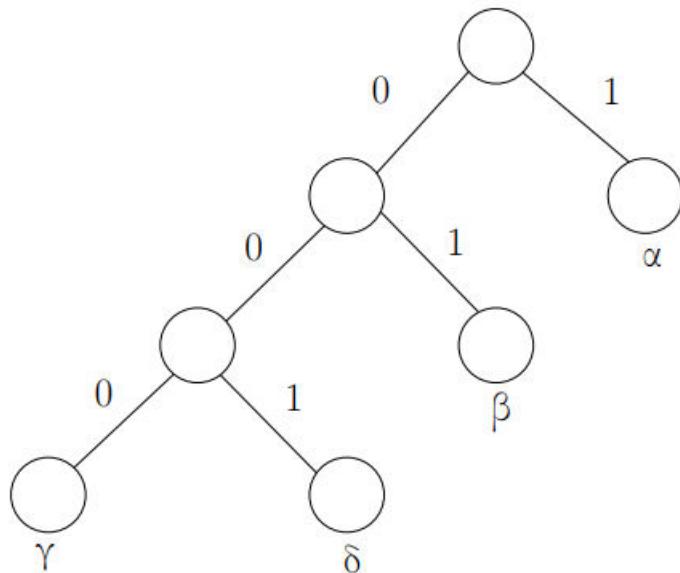
Παρατηρήσεις

Άμεση συνέπεια του Λήμματος 10 είναι ότι αν T αποτελεί το βέλτιστο δένδρο για το αλφάβητο C τότε το δένδρο αυτό «περιέχει» το βέλτιστο δένδρο για το αλφάβητο $C \setminus \{x, y\}$, όπου x, y οι κορυφές που αντιστοιχούν στα βαθύτερα φύλλα του δένδρου. Το τελευταίο μπορούμε να το ανακτήσουμε αν από το T διαγράψουμε τις κορυφές αυτές μαζί με τον άμεσα πρόγονο τους στο δένδρο και συγχωνεύσουμε την εσωτερική κορυφή που συνδέει το φύλλο που αντιστοιχεί στην κορυφή $p = \operatorname{argmin}\{f(c) : c \in C \setminus \{x, y\}\}$, με τον άμεσο πρόγονο της. Αν, για παράδειγμα, εφαρμόσουμε τα παραπάνω στο δένδρο του Σχήματος 9.2 θα πάρουμε το δένδρο του Σχήματος 9.3.

Με επαγγεική εφαρμογή της διαδικασίας που χρησιμοποιήθηκε στην απόδειξη του Λήμματος 9, από ένα βέλτιστο απροθηματικό δένδρο T^* μπορούμε να παράγουμε το δένδρο του Αλγόριθμου 50 χωρίς να μεταβληθεί το κόστος σε κανένα από τα «ενδιάμεσα δένδρα». Συνεπώς από τα Λήμματα (10), (9) προκύπτει άμεσα η επόμενη πρόταση

Πόρισμα

Πόρισμα 8. Το δένδρο Huffman παράγει το βέλτιστο σχήμα κωδικοποίησης.



Σχήμα 9.4: Δένδρο Huffman για το αλφάριθμο του Πίνακα 9.2 χωρίς τους χαρακτήρες ‘ε’ και ‘ζ’

08_ΒΑΣΙΚΕΣ ΚΛΑΣΕΙΣ ΠΡΟΒΛΗΜΑΤΩΝ P, NP / 15a_Basic Classes Notes, 16_npcexer

SAT

SAT

Στιγμιότυπο: Τύπος ϕ σε κανονική συζευτική μορφή.

Ερώτηση: Είναι ο ϕ ικανοποιήσιμος;

Απόδειξη

Για αν εφαρμόσουμε την παραπάνω μεθοδολογία θα πρέπει να γνωρίζουμε ένα τουλάχιστον πρόβλημα το οποίο να ανήκει σε αυτή την κλάση. Το πρόβλημα αυτό, ονομάζεται SAT και αφορά την αποτίμηση ενός λογικού τύπου σε κανονική συζευτική μορφή. Το πρόβλημα περιγράφεται με βάση τους επόμενους ορισμούς. Μία λογική μεταβλητή x παίρνει τιμές στο σύνολο {True, False}. Σε κάθε λογική μεταβλητή x αντιστοιχούν δύο όροι: ήτοι x , \bar{x} . Οι όροι αυτοί είναι αντίθετοι: αν μία αποτίμηση δίνει στον έναν όρο την τιμή True τότε αυτομάτως ο άλλος όρος παίρνει την τιμή False. Ο συνδυασμός λογικών όρων με τους τελεστές διάζευξης \vee (OR), σύζευξης \wedge (AND) και με τη χρήση παρενθέσεων δημιουργεί τους λογικούς τύπους. Για παράδειγμα, με τη χρήση των λογικών μεταβλητών x_1, x_2, x_3, x_4 μπορούμε να γράψουμε τους τύπους

$$\phi_1 = (x_1 \vee \bar{x}_2) \wedge (x_1 \wedge \bar{x}_4 \wedge x_3), \quad (11.1)$$

$$\phi_2 = (x_2 \wedge x_4) \vee (\bar{x}_1 \wedge x_3 \wedge \bar{x}_2), \quad (11.2)$$

$$\phi_3 = (\bar{x}_3 \vee \bar{x}_2 \vee x_4) \wedge (x_2 \vee x_1 \vee \bar{x}_4). \quad (11.3)$$

Θα αναφερόμαστε σε κάθε υπο-τύπο που εμπεριέχεται μέσα σε μια παρένθεση ως (*λογική*) *πρόταση*. Για παράδειγμα, στον τύπο που ορίζεται στην (11.1), προτάσεις αποτελούν οι $(x_1 \vee \bar{x}_2)$ και $(x_1 \wedge \bar{x}_4 \wedge x_3)$. Ένας τύπος είναι σε κανονική συζευτική μορφή αν αποτελεί σύζευξη προτάσεων μέσα στις οποίες οι όροι συνδέονται με διάζευξη. Ο τύπος που ορίζεται από την (11.3) είναι σε κανονική συζευτική μορφή. Ο συμμετρικός ορισμός αποτελεί δίνει τύπους σε κανονική διαζευτική μορφή. Έχουμε διάζευξη προτάσεων στις οποίες οι όροι συνδέονται με σύζευξη, πχ., τύπος που ορίζεται από την (11.2).

Για κάθε απόδοση τιμών από το σύνολο {True, False} στους λογικούς όρους ενός τύπου, έχουμε μία αποτίμηση του. Για παράδειγμα, αν θέσουμε $x_1 = \text{True}$, $x_2 = \text{False}$, $x_3 = \text{True}$, $x_4 = \text{False}$, οι αποτιμήσεις των παραπάνω τύπων είναι $\phi_1 = \text{True}$, $\phi_2 = \text{False}$, $\phi_3 = \text{True}$. Ένας τύπος ονομάζεται *ικανοποιήσιμος* αν υπάρχει μία απόδοση τιμών στους όρους του ώστε ο τύπος να αποτιμάται σε True. Το πρόβλημα SAT αφορά ακριβώς αυτό:

SAT

Στιγμιότυπο: Τύπος ϕ σε κανονική συζευτική μορφή.

Ερώτηση: Είναι ο ϕ ικανοποιήσιμος;

Όπως τονίστηκε προηγουμένως το πρόβλημα SAT είναι το πρώτο πρόβλημα το οποίο αποδείχθηκε ότι ανήκει στην κλάση NP-complete. Θα χρησιμοποιήσουμε το πρόβλημα αυτό σαν «σημείο εκκίνησης» προκειμένου να αποδείξουμε και για κάποια άλλα προβλήματα ότι ανήκουν στην κλάση αυτή. Η μεθοδολογία που θα χρησιμοποιηθεί είναι αυτή που περιγράφηκε στην προηγούμενη ενότητα.

3SAT

Πρόβλημα

Το πρόβλημα 3SAT είναι μία περιορισμένη έκδοση του SAT όπου οι προτάσεις που συνιστούν τον τύπο ϕ περιέχουν ακριβώς τρεις όρους· για παράδειγμα, ο τύπος που περιγράφεται από την (11.3) είναι τέτοιας μορφής. Τυπικά το πρόβλημα περιγράφεται ως εξής.

Περιγραφή

3SAT

Στιγμιότυπο: Τύπος ϕ σε κανονική συζευτική μορφή όπου σε κάθε πρόταση εμφανίζονται τρεις όροι.

Ερώτηση: Είναι ο ϕ ικανοποιήσιμος;

Απόδειξη 3SAT ∈ NP

Λήμμα 11. 3SAT ∈ NP.

Απόδειξη. Θεωρούμε ότι ο τύπος ϕ αποτελεί σύζευξη m προτάσεων, όπου $m \in \mathbb{N}$, σε κάθε μία από τις οποίες περιέχονται ακριβώς τρεις όροι. Αν συμβολίσουμε την i -οστή πρόταση ως ϕ^i τότε μπορούμε να γράψουμε τον τύπο ως $\phi = \wedge_{i=1}^m \phi^i$, όπου $\phi^i = \lambda_1^i \vee \lambda_2^i \vee \lambda_3^i$, και $\lambda_1^i, \lambda_2^i, \lambda_3^i$ οι τρεις όροι της πρότασης. Επομένως, αν αποδώσουμε τιμές στους όρους, για να αποτιμήσουμε κάθε πρόταση ϕ^i εκτελούμε δύο ΣΥΒ - δυο λογικές πράξεις διάζευξης. Στη συνέχεια χρειάζεται να εκτελέσουμε $m - 1$ πράξεις σύζευξης για να αποτιμήσουμε τον τύπο ϕ . Άρα για να ελέγχουμε αν μία απόδοση τιμών στους όρους κάνει τον τύπο ϕ True εκτελούνται $2m$ - δηλαδή $\Theta(m)$ - ΣΥΒ. \square

Απόδειξη 3SAT ∈ NP-hard

Λήμμα 12. 3SAT ∈ NP-hard.

Απόδειξη. Θα δείξουμε αναγωγή από το πρόβλημα SAT. Δηλαδή από ένα τυχαίο στιγμιότυπο του προβλήματος SAT θα κατασκευάσουμε ένα στιγμιότυπο του προβλήματος 3SAT κατά τρόπο ώστε το στιγμιότυπο του SAT θα είναι αποτιμάται σε True αν και μόνο αν το ίδιο συμβαίνει για το στιγμιότυπο 3SAT. Επιπλέον θα δείξουμε ότι το στιγμιότυπο 3SAT έχει πολυωνυμικό μέγεθος σε σχέση με αυτό του στιγμιότυπου SAT.

Έστω τύπος $\hat{\phi}$ ο οποίος αποτελεί στιγμιότυπο του προβλήματος SAT. Θεωρούμε ότι ο τύπος αποτελεί τη σύζευξη m διαζευκτικών προτάσεων, δηλαδή

$$\hat{\phi} = \hat{\phi}^1 \wedge \hat{\phi}^2 \wedge \cdots \wedge \hat{\phi}^m,$$

όπου

$$\hat{\phi}^i = \lambda_1^i \vee \lambda_2^i \vee \cdots \vee \lambda_t^i, \forall i \in \{1, \dots, m\}.$$

Δηλαδή η πρόταση $\hat{\phi}^i$ αποτελείται από τη διάζευξη t όρων, ήτοι των $\lambda_1^i, \lambda_2^i, \dots, \lambda_t^i$.

Για κάθε πρόταση $\hat{\phi}^i$, θα γράψουμε είτε μία πρόταση ή μία σύζευξη προτάσεων, ονομαστικά ϕ^i , με τις ιδιότητες

1. η πρόταση (προτάσεις της) ϕ^i να αποτελούνται από τρεις όρους η κάθε μία και
2. η ϕ^i να αποτιμάται σε True αν και μόνο αν το ίδιο συμβαίνει για την ϕ^i .

Παρατηρούμε ότι η δεύτερη ιδιότητα διασφαλίζει ότι το παραγόμενο στιγμιότυπο 3SAT είναι ΝΑΙ-στιγμιότυπο αν και μόνο αν το στιγμιότυπο SAT είναι ΝΑΙ-στιγμιότυπο. Διακρίνουμε τις παρακάτω περιπτώσεις σε σχέση με το πλήθος των όρων (παράμετρος t) που περιέχει η ϕ .

$t = 1$: Η ϕ^i περιέχει τη σύζευξη του ιδίου όρου:

$$\phi^i = \lambda_1^i \vee \lambda_2^i \vee \lambda_3^i.$$

$t = 2$: Η ϕ^i αποτελείται από τη σύζευξη των όρων της $\hat{\phi}^i$ επαυξημένη με τον πρώτο από τους δύο:

$$\phi^i = \lambda_1^i \vee \lambda_2^i \vee \lambda_1^i.$$

$t = 3$: Η ϕ^i είναι ακριβώς ίδια με την $\hat{\phi}^i$:

$$\phi^i = \hat{\phi}^i = \lambda_1^i \vee \lambda_2^i \vee \lambda_3^i.$$

$t \geq 4$: Για να γράψουμε την ϕ^i θα χρησιμοποιήσουμε $t - 3$ νέες μεταβλητές - μεταβλητές που δεν εμφανίζονται στον τύπο $\bar{\phi}$ - ονομαστικά z_1, \dots, z_{t-3} . Η ϕ^i θα αποτελείται από τη σύζευξη $t - 4 + 2 = t - 2$ προτάσεων στις οποίες χρησιμοποιούνται $2(t - 3)$ νέοι όροι, ήτοι

$$\begin{aligned} \phi^i = & (\lambda_1^i \vee \lambda_2^i \vee z_1) \wedge (\bar{z}_1 \vee \lambda_3^i \vee z_2) \wedge \\ & \cdots \wedge (\bar{z}_{k-2} \vee \lambda_k^i \vee z_{k-1}) \wedge \cdots \\ & (\bar{z}_{t-4} \vee \lambda_{t-2}^i \vee z_{t-4}) \wedge (\bar{z}_{t-3} \vee \lambda_{t-1}^i \vee \lambda_t^i). \end{aligned} \quad (11.4)$$

Για $t \leq 3$, είναι προφανές ότι ϕ^i αποτιμάται σε True αν και μόνο αν το ίδιο συμβαίνει για την $\hat{\phi}^i$. Θα δείξουμε ότι αυτό ισχύει και στην περίπτωση κατά την οποία $t \geq 4$. Για την περίπτωση αυτή, αρχικώς, θα δείξουμε ότι αν $\hat{\phi}^i$ αποτιμάται σε True μπορούμε να δώσουμε (λογικές) τιμές στους νέους όρους ώστε και ϕ^i να αποτιμάται σε True. Παρατηρούμε ότι για να αποτιμάται $\hat{\phi}^i$ σε True θα πρέπει ένας τουλάχιστον από τους όρους της - δηλαδή ένας από τους όρους $\lambda_1^i, \lambda_2^i, \dots, \lambda_t^i$ - να έχει τιμή True.

Αν $t > 4$, χωρίς βλάβη της γενικότητας θεωρούμε ότι ο όρος αυτός είναι ο λ_k - δες (11.4). Θέτουμε,

$$z_j = \text{True}, \forall j = 1, \dots, k - 2, \quad (11.5)$$

$$z_j = \text{False}, \forall j = k - 1, \dots, t - 3. \quad (11.6)$$

Παρατηρούμε ότι από την παραπάνω ανάθεση τιμών όλες οι προτάσεις που προηγούνται της $(\bar{z}_{k-2} \vee \lambda_k^i \vee z_{k-1})$ στην (11.4) αποτιμώνται σε True αφού σε κάθε μία ο όρος z_j έχει τιμή True. Αντίστοιχα, για όλες τις προτάσεις που βρίσκονται δεξιότερα της παραπάνω πρότασης αποτιμώνται σε True αφού σε κάθε μία ο όρος \bar{z}_j έχει τιμή True. Επίσης η πρόταση $(\bar{z}_{k-2} \vee \lambda_k^i \vee z_{k-1})$ αποτιμάται σε True αφού (εξ' υποθέσεως) ο όρος $\lambda_k^i = \text{True}$.

Αν $t = 4$, η (11.4) γράφεται ως

$$\phi^i = (\lambda_1^i \vee \lambda_2^i \vee z_1) \wedge (\bar{z}_1 \vee \lambda_3^i \vee \lambda_4^i)$$

Αν ο όρος που αποτιμάται σε True είναι κάποιος από τους λ_3^i, λ_4^i η z_1 παίρνει τιμή από την (11.5), διαφορετικά από την (11.6).

Άρα όταν $t \geq 4$, η απόδοση τιμών από τις (11.5), (11.6), διασφαλίζει ότι η ϕ^i να αποτιμάται σε True όταν $\hat{\phi}^i$ αποτιμάται σε True. Στη συνέχεια θα δείξουμε το αντίστροφο: αν $\hat{\phi}^i$ αποτιμάται σε False δεν υπάρχει απόδοση τιμών στους όρους

z_1, \dots, z_{t-3} (και επομένως στους $\bar{z}_1, \dots, \bar{z}_{t-3}$) ώστε να αποτιμάται η ϕ σε True. Παρατηρούμε ότι αν $\hat{\phi}^i = \text{False}$, θα πρέπει $\lambda_1^i = \lambda_2^i = \dots = \lambda_t^i = \text{False}$. Για να έχουμε $\phi^i = \text{True}$, θα πρέπει κάθε πρόταση της (11.4) να έχει έναν τουλάχιστον όρο με τιμή True. Άρα θα πρέπει να θέσουμε $z_1 = \text{True}$ για να αποτιμηθεί η πρώτη πρόταση της (11.4) σε True, στη συνέχεια, να θέσουμε $z_2 = \text{True}$ για να αποτιμηθεί η δεύτερη πρόταση της (11.4) σε True, κοκ. Όμως έτσι θα φτάσουμε να έχουμε στην τελευταία πρόταση της (11.4) όλους τους όρους με τιμή False.

Από τα παραπάνω προκύπτει ότι ο τύπος ϕ αποτιμάται σε True αν και μόνο αν το ίδιο συμβαίνει για τον ϕ .

Έστω n ο αριθμός των μεταβλητών που εμφανίζονται στον τύπο ϕ . Θα δείξουμε ότι ο αριθμός των όρων και των προτάσεων που εμφανίζονται στον τύπο ϕ φράζεται από τα επάνω από ένα πολυώνυμο των n, m . Παρατηρούμε ότι οι όροι σε κάθε πρόταση ϕ^i δεν μπορούν να ξεπερνούν σε αριθμό το n : διαφορετικά η πρόταση αποτιμάται σε True αφού θα εμφανίζονται και οι δύο όροι για κάποια μεταβλητή - η πρόταση γίνεται tautology. Άρα εφόσον στην πρόταση ϕ^i εμφανίζονται το-πολύ n όροι, η ϕ^i αποτελείται από το-πολύ $n - 2$ προτάσεις² με τρεις όρους η κάθε μία. Επομένως ο αριθμός των προτάσεων στο στιγμιότυπο 3SAT είναι τάξης $O((n - 2)m) = O(nm)$. Αντίστοιχα, σε κάθε ϕ^i εμφανίζονται το-πολύ $2(n - 3)$ νέοι όροι και άρα ο συνολικός αριθμός των όρων στην ϕ^i δεν μπορεί να ξεπερνάει τους $2(n - 3) + n = 3(n - 2)$. Επομένως ο αριθμός των όρων του στιγμιότυπου 3SAT είναι τάξης $O(3(n - 2)m) = O(nm)$. \square

1 – 3SAT

Πρόβλημα

Το πρόβλημα αυτό αποτελεί παραλλαγή του 3SAT. Δηλαδή και στην περίπτωση αυτή ρωτάμε για την ικανοποιησιμότητα ενός τύπου $\phi = \wedge_{i=1}^m \phi^i$, όπου κάθε ϕ^i αποτελείται από τρεις όρους. Όμως σε ένα ΝΑΙ-στιγμιότυπο το ϕ αποτιμάται σε True αν και μόνο αν ακριβώς ένας από τους τρεις όρους σε κάθε πρόταση παίρνει την τιμή True. Έχουμε τον ακόλουθο ορισμό.

Περιγραφή

1 – 3SAT

Στιγμιότυπο: Τύπος ϕ σε κανονική συζευτική μορφή όπου σε κάθε πρόταση εμφανίζονται τρεις όροι.

Ερώτηση: Είναι ο ϕ ικανοποιήσιμος από μία αποτίμηση που δίνει τιμή True σε ακριβώς έναν όρο σε κάθε πρόταση;

Απόδειξη 1-3SAT ∈ NP

Για να γίνει ευκολότερα αντιληπτός ο παραπάνω ορισμός, ο τύπος

$$\phi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3)$$

ενώ αποτελεί ένα ΝΑΙ-στιγμιότυπο 3SAT (αφού η απόδοση τιμών $x_1 = \text{True}$, $x_2 = \text{False}$, $x_3 = \text{False}$ τον ικανοποιεί) αποτελεί ένα ΟΧΙ-στιγμιότυπο 1 – 3SAT αφού δεν υπάρχει απόδοση τιμών στους όρους που να τον ικανοποιεί και σε κάθε πρόταση να υπάρχει ένας μόνο όρος να έχει τιμή True.

Μία απόδοση τιμών στους όρους των προτάσεων που απαρτίζουν τον τύπο ϕ αποτελεί ένα πιστοποιητικό για το ότι ο τύπος αποτελεί ένα ΝΑΙ-στιγμιότυπο. Θεωρώντας ότι το στιγμιότυπο αποτελείται από m προτάσεις, σε χρόνο $\Theta(m)$ επαληθεύεται ότι $\phi = \text{True}$ (δες απόδειξη Λήμματος 11). Στη συνέχεια, για κάθε πρόταση αρχικοποιείται (σε τιμή μηδέν) ένας μετρητής ο οποίος αυξάνει κάθε φορά που συναντάται ένας όρος με τιμή True. Σε ένα ΝΑΙ-στιγμιότυπο, ο μετρητής της κάθε πρότασης πρέπει να επιστρέψει τιμή ίση με 1. Ο έλεγχος αυτός μπορεί να εκτελεστεί σε $\Theta(m)$ ΣΥΒ αφού για κάθε πρόταση εκτελείται σταθερός αριθμός ΣΥΒ και ο τύπος περιέχει m προτάσεις.

Απόδειξη $1 - 3SAT \in NP\text{-hard}$

Λήμμα 13. $1 - 3SAT \in NP\text{-hard}$

Απόδειξη. Θα δείξουμε ότι $3SAT \leq_P 1 - 3SAT$. Θεωρούμε ένα στιγμιότυπο $3SAT$ με m προτάσεις: δηλαδή, τύπο $\phi = \wedge_{i=1}^m \phi^i$, όπου ϕ^i είναι μία πρόταση με τρεις όρους. Από αυτό θα κατασκευάσουμε ένα στιγμιότυπο $1 - 3SAT$ τέτοιο ώστε να είναι ΝΑΙ-στιγμιότυπο αν και μόνο αν το $3SAT$ είναι ΝΑΙ-στιγμιότυπο. Μπορούμε να γράψουμε κάθε πρόταση ϕ^i του στιγμιότυπου $3SAT$ ως

$$\phi^i = a^i \vee b^i \vee c^i, \quad i = 1, \dots, m.$$

Για κάθε τέτοια πρόταση, εισάγουμε στο στιγμιότυπο $1 - 3SAT$ την σύζευξη τεσσάρων προτάσεων

$$\psi^i = (\bar{a}_i \vee y_1^i \vee z_1^i) \wedge (\bar{b}_i \vee y_2^i \vee z_2^i) \wedge (\bar{c}_i \vee y_3^i \vee z_3^i) \wedge (y_1^i \vee y_2^i \vee y_3^i),$$

όπου $\bar{a}_i, \bar{b}_i, \bar{c}_i$ οι αντίθετοι των όρων a_i, b_i, c_i , αντίστοιχα, και οι y_1^i, y_2^i, y_3^i και z_1^i, z_2^i, z_3^i είναι νέοι όροι που δεν εμφανίζονται στον τύπο ϕ .

Αν το στιγμιότυπο $3SAT$ είναι ένα ΝΑΙ-στιγμιότυπο τότε υπάρχει μία απόδοση τιμών στους όρους για την οποία σε κάθε πρόταση ϕ^i υπάρχει ένας τουλάχιστον όρος με τιμή $True$. Χωρίς βλάβη της γενικότητας θεωρούμε ότι στην πρόταση ϕ^i ο όρος $a^i = True$. Θα δείξουμε ότι μπορούμε να δώσουμε τιμές στους νέους όρους που εισάγαμε παραπάνω ώστε η ψ^i να αποτιμάται σε $True$ και ταυτόχρονα να παίρνει τιμή $True$ μόνο ένας από τους όρους σε κάθε πρόταση της ψ^i . Η ανάθεση είναι

$$\begin{aligned} y_1^i &= True, \quad y_2^i = False, \quad y_3^i = False, \\ z_1^i &= False, \quad z_2^i = b^i, \quad z_3^i = c^i, \end{aligned}$$

όπου οι δύο τελευταίες σχέσεις υποδηλώνουν ότι οι όροι z_2^i, z_3^i παίρνουν την ίδια τιμή που έχουν οι όροι b^i, c^i , αντίστοιχα, στην απόδοση τιμών που ικανοποιεί τον τύπο ϕ . Παρατηρούμε ότι με δεδομένο ότι $a_i = True$, η παραπάνω απόδοση τιμών (όποιες και αν είναι οι τιμές των όρων b^i, c^i) αποτιμά την ψ^i σε $True$ θέτοντας ακριβώς έναν όρο σε τιμή $True$ σε κάθε μία από της τέσσερις προτάσεις που την αποτελούν.

Αντίστροφα, αν το $3SAT$ είναι ΟΧΙ-στιγμιότυπο, θα υπάρχει πρόταση ϕ^i όπου όλοι οι όροι της θα έχουν την τιμή $False$. Στην περίπτωση αυτή σε κάθε μία από τις τρεις πρώτες προτάσεις της ψ^i θα έχουμε έναν από τους αντίθετους όρους σε τιμή $True$. Παρατηρούμε, ότι επειδή απαιτείται να υπάρχει μόνο ένας όρος με τιμή $True$ σε κάθε πρόταση, θα πρέπει όλοι οι νέοι όροι να τεθούν σε τιμή $False$. Όμως αυτή η απόδοση τιμών κάνει την τέταρτη πρόταση της ψ^i να αποτιμάται σε $False$ και επομένως και η ψ^i αποτιμάται σε $False$. Συνεπώς, και το παραγόμενο $1 - 3SAT$ στιγμιότυπο είναι ΟΧΙ-στιγμιότυπο.

Όσον αφορά το μέγεθος του $1 - 3SAT$ στιγμιότυπου που κατασκευάσαμε, παρατηρούμε ότι έχει $4m$ προτάσεις και $6m$ νέους όρους. Επομένως το μέγεθος του είναι πολυωνυμικό σε σχέση με το μέγεθος του $3SAT$ από το οποίο ξεκινήσαμε. \square

INDSET

Πρόβλημα

Θυμηθείτε ότι σε ένα μη-κατευθυνόμενο γράφημα $G(V, E)$ ένα ανεξάρτητο σύνολο είναι ένα υποσύνολο κορυφών V' τέτοιο ώστε να υπάρχει στο γράφημα ακμή ανάμεσα σε οποιοδήποτε ζευγάρι κορυφών του. Το πλήθος των κορυφών του V' ονομάζεται μέγεθος του ανεξάρτητου συνόλου. Με βάση τα παραπάνω ορίζουμε το πρόβλημα INDSET ως:

Περιγραφή

INDSET

Στιγμιότυπο: Μη-κατευθυνόμενο γράφημα $G(V, E)$, ακέραιος k .

Ερώτηση: Υπάρχει στο γράφημα G ένα ανεξάρτητο σύνολο μεγέθους μεγαλύτερου-ίσου του k ;

Απόδειξη $\text{INDSET} \in \text{NP}$

Λήμμα 14. $\text{INDSET} \in \text{NP}$

Απόδειξη. Αν $k > |V|$ το στιγμιότυπο έχει απάντηση αρνητική - η λύση είναι ΟΧΙ. Διαφορετικά (δηλαδή $k \leq |V|$), χωρίς βλάβη της γενικότητας μπορούμε να θεωρήσουμε ότι $V = \{1, 2, \dots\}$. Έτσι σε περίπτωση θετικής απάντησης - το συγκεκριμένο στιγμιότυπο έχει απάντηση ΝΑΙ - ένα πιστοποιητικό αποτελείται από ένα υποσύνολο του V μεγέθους τουλάχιστον k και το-πολύ ίσου με $|V|$. Επομένως σε $O(|V| \lg |V|)$ ΣΥΒ³ μπορούμε να ελέγχουμε αν ένα δοσμένο σύνολο S πληροί στις ιδιότητες αυτές. Στη συνέχεια, ελέγχουμε αν για κάθε ζευγάρι κορυφών $v, u \in S$ υπάρχει στο G ακμή που συνδέει τις κορυφές αυτές. Αυτό μπορεί να διαπιστωθεί εκτελώντας $O(|S|^2)$ ΣΥΒ - μία ερώτηση (σύγκριση) για κάθε ζευγάρι κορυφών του S . Επειδή $|S| \leq |V|$ οι παραπάνω έλεγχοι εκτελούνται σε $(|V|^2)$ ΣΥΒ. □

Απόδειξη $\text{INDSET} \in \text{NP-hard}$

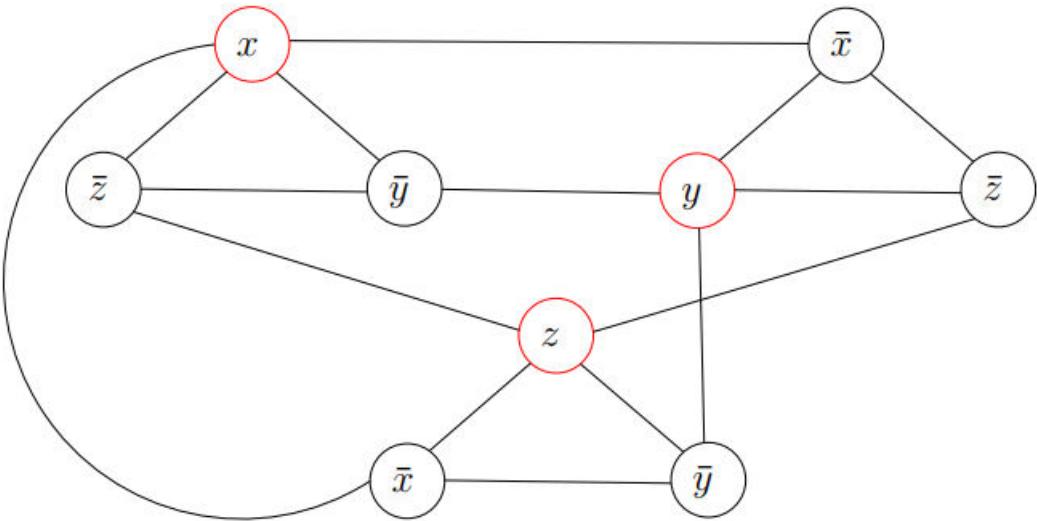
Λήμμα 15. $\text{INDSET} \in \text{NP-hard}$

Απόδειξη. Θα δείξουμε ότι $\text{3SAT} \leq_P \text{INDSET}$. Προς τούτο, χωρίς βλάβη της γενικότητας, θεωρούμε ένα 3SAT στιγμιότυπο με k πρότασεις. Θα κατασκευάσουμε μη-κατευθυνόμενο γράφημα $G(V, E)$ κατά τρόπο ώστε αυτό θα έχει ένα ανεξάρτητο μεγέθους k αν και μόνο αν το 3SAT στιγμιότυπο είναι ικανοποιησιμό. Η κατασκευή είναι η εξής. Για κάθε πρόταση του 3SAT εισάγουμε μία τριάδα κορυφών, μία κορυφή για κάθε όρο της πρότασης. Στο γράφημα υπάρχουν ακμές ανάμεσα στις κορυφές της ίδιας τριάδας καθώς και ακμές ανάμεσα σε κορυφές διαφορετικών τριάδων που αντιστοιχούν σε αντίθετους όρους. Για παράδειγμα για το στιγμιότυπο 3SAT

$$\phi = (x \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z)$$

το γράφημα απεικονίζεται στο Σχήμα 11.3. Παρατηρούμε ότι το γράφημα αυτό περιέχει ένα ανεξάρτητο σύνολο τριών κορυφών (κόκκινες κορυφές). Αν θέσουμε τους αντίστοιχους όρους στο στιγμιότυπο 3SAT σε True παρατηρούμε ότι ο τύπος ϕ αποτιμάται σε True . Θα αποδείξουμε ότι τόσο αυτό όσο και το αντίστροφο ισχύει στη γενική περίπτωση.

Έστω ότι το στιγμιότυπο $\text{3SAT } \phi = \bigwedge_{i=1}^k \phi^i$ αποτιμάται σε ΝΑΙ. Τότε, σε κάθε πρόταση ϕ^i ένας τουλάχιστον όρος έχει τιμή True . Αν επιλέξουμε για κάθε πρόταση έναν από αυτούς τους όρους και παρατηρήσουμε τις αντίστοιχες κορυφές στο γράφημα $G(V, E)$ θα δούμε ότι αποτελούν ένα ανεξάρτητο σύνολο με k κορυφές: κάθε κορυφή ανήκει σε διαφορετική τριάδα (αφού έχουμε επιλέξει έναν όρο από κάθε πρόταση) και αν υπήρχε ακμή ανάμεσα σε δύο από αυτές αυτό θα σήμαινε ότι σε δύο αντίθετους όρους θα έχει δοθεί η τιμή True (άτοπο).



Σχήμα 11.3: Μη-κατευθυνόμενο γράφημα που περιέχει ανεξάρτητο σύνολο τριών κορυφών

Αντίστροφα, αν υπάρχει στο γράφημα ένα ανεξάρτητο σύνολο από k κορυφές τότε κάθε κορυφή θα ανήκει σε διαφορετική τριάδα - οι κορυφές της ίδιας τριάδας συνδέονται μεταξύ τους με ακμές - άρα οι αντίστοιχοι όροι εμφανίζονται ο καθένας σε διαφορετική πρόταση ϕ^i . Επίσης μπορούμε να τους θέσουμε όλους στην τιμή True - αποτιμώντας έτσι τον τύπο ϕ σε True - χωρίς να υπάρχει περίπτωση να έχουμε θέσει δύο αντίθετους όρους στην τιμή True αφού κορυφές που ανήκουν σε διαφορετικές τριάδες συνδέονται με ακμή μόνο αν αντιστοιχούν σε αντίθετους όρους.

Το γράφημα έχει $|V| = 3k$ κορυφές και επομένως η αναγωγή είναι πολυωνυμική. \square

KAT – MONOPATI – XAMIATON

Πρόβλημα

Σε ένα κατευθυνόμενο γράφημα $G(V, E)$ ένα (απλό) μονοπάτι που εκκινεί από μία κορυφή, έστω s , και καταλήγει σε μία κορυφή, έστω t , έχοντας «περάσει» από κάθε άλλη κορυφή του γραφήματος ονομάζεται μονοπάτι Χάμιλτον. Με βάση αυτό τον ορισμό ορίζουμε το πρόβλημα

Περιγραφή

KAT – MONOPATI – XAMIATON

Στιγμιότυπο: Κατευθυνόμενο γράφημα $G(V, E)$

Ερώτηση: Υπάρχει στο G μονοπάτι Χάμιλτον;

Απόδειξη KAT-MONOPATI-XAMIATON \in NP

Λήμμα 16. KAT – MONOPATI – XAMIATON \in NP.

Απόδειξη. Ένα σύντομο πιστοποιητικό για κάποιο ΝΑΙ-στιγμιότυπο αποτελεί μία ακολουθία κορυφών, έστω $S = < v_0, v_1, \dots, v_{n-1} >$, η οποία περιλαμβάνει ακριβώς τις κορυφές του V . Ταξινομώντας τις κορυφές του V και τις κορυφές του S μπορούμε να ελέγχουμε ότι αυτό ισχύει αν οι δύο ταξινομημένες σειρές ακολουθιών ταυτίζονται. Η διαδικασία αυτή μπορεί τυπικά να ολοκληρωθεί σε $O(n \lg n)$ ΣΥΒ, όπου $n = |V|$. Στη συνέχεια ελέγχουμε αν $(v_i, v_{i+1}) \in E(G)$, για $i = 0, \dots, n - 1$. Η διαδικασία αυτή ολοκληρώνεται σε γραμμικό χρόνο ($O(n)$). \square

Απόδειξη KAT-MONOΠΑΤΙ-ΧΑΜΙΛΤΟΝ ∈ NP-hard

Λήμμα 17. KAT – ΜΟΝΟΠΑΤΙ – ΧΑΜΙΛΤΟΝ ∈ NP-hard

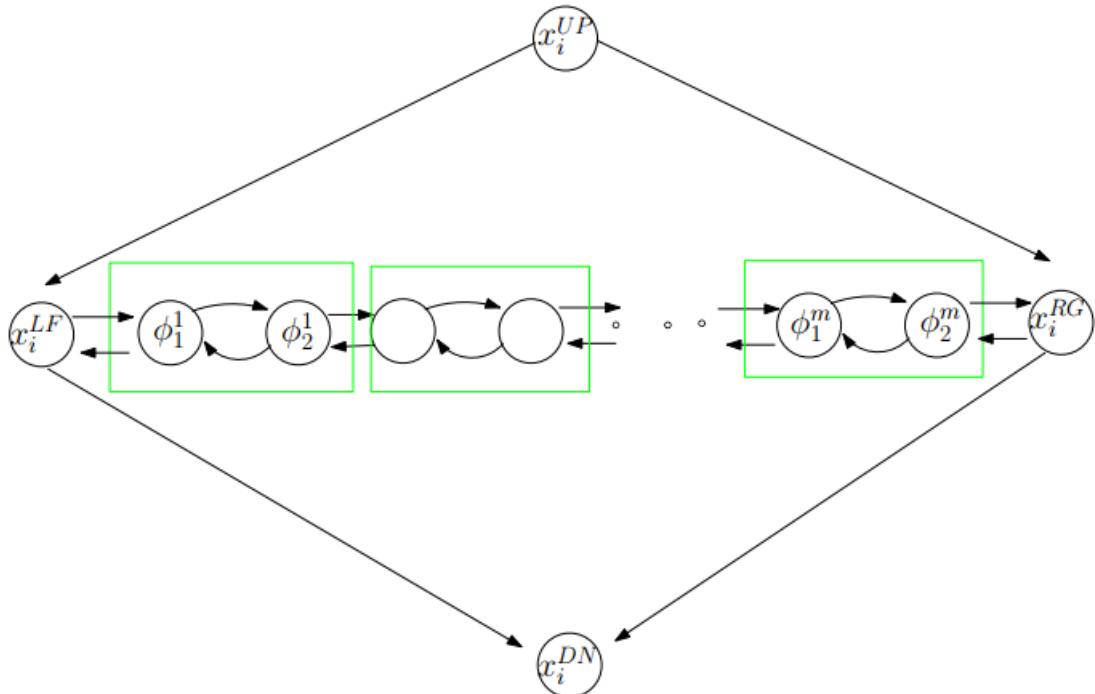
Απόδειξη. Θα δείξουμε ότι $1 - 3SAT \leq_P KAT - \text{ΜΟΝΟΠΑΤΙ} - \text{ΧΑΜΙΛΤΟΝ}$. Θεωρούμε ότι το στιγμιότυπο $1 - 3SAT$ περιέχει n λογικές μεταβλητές (άρα $2n$ σύμβολα τα οποία δεν είναι ανάγκη να εμφανίζονται όλα) και m προτάσεις. Θα κατασκευάσουμε ένα κατευθυνόμενο γράφημα $G(V, E)$ το οποίο θα διαθέτει μονοπάτι του Χάμιλτον ANN το στιγμιότυπο $1-3IKAN$ είναι NAI. Το γράφημα θα περιέχει γραφικές συνιστώσες τύπου «διαμάντι»- μία για κάθε λογική μεταβλητή. Μία συνιστώσα τέτοιου τύπου για τη (λογική) μεταβλητή x_i αποτελείται καταρχάς από τέσσερις κορυφές, έστω $x_i^{UP}, x_i^{DN}, x_i^{LF}, x_i^{RG}$, και τις κατευθυνόμενες ακμές

$$(x_i^{UP}, x_i^{LF}), (x_i^{LF}, x_i^{DN}), (x_i^{UP}, x_i^{RG}), (x_i^{RG}, x_i^{DN}), \forall i \in \{1, \dots, n\}.$$

Επιπλέον σε κάθε τέτοια συνιστώσα υπάρχει ένα ζευγάρι κορυφών για κάθε πρόταση $\phi^j, j \in \{1, \dots, m\}$, ήτοι οι κορυφές ϕ_1^j, ϕ_2^j . Κάθε τέτοιο ζευγάρι συνδέεται με δύο ακμές; η μία κατευθύνεται από το ϕ_1^j στο ϕ_2^j και η άλλη αντίστροφα. Επίσης υπάρχουν δύο ακμές ανάμεσα σε κάθε ζευγάρι και στο επόμενο του ως εξής:

$$(\phi_2^j, \phi_1^{j+1}), (\phi_1^{j+1}, \phi_2^j), \forall j \in \{1, \dots, m-1\}.$$

Παρατηρούμε ότι όλα τα ζευγάρια κορυφών ϕ_1^j, ϕ_2^j συνδέονται σε μία «αλυσίδα» που επιτρέπει τη διάσχιση των κορυφών είτε από αριστερά προς τα δεξιά είτε από δεξιά προς τα αριστερά. Το «διαμάντι» ολοκληρώνεται με τη διασύνδεση της αλυσίδας με τις κορυφές x_i^{LF}, x_i^{RG} . Αυτό γίνεται με την προσθήκη των ακμών $(x_i^{LF}, \phi_1^1), (\phi_1^1, x_i^{LF})$ και $(x_i^{RG}, \phi_2^m), (\phi_2^m, x_i^{RG})$. Μία συνιστώσα διαμάντι απεικονίζεται στο Σχήμα 11.4. Ο



Σχήμα 11.4: Συνιστώσα τύπου «διαμάντι». Μία τέτοια συνιστώσα για κάθε μεταβλητή αριθμός των κορυφών και ακμών σε κάθε τέτοια συνιστώσα είναι

$$|V_i| = 4 + 2m, |E_i| = 4 + 2m + 2(m-1) + 4 = 4m - 6. \quad (11.7)$$

Όπως είδαμε και προηγουμένως, οι ακμές είναι τοποθετημένες με τέτοιο τρόπο ώστε για να διασχίσουμε όλους τις κορυφές μίας αλυσίδας είτε θα κινηθούμε από αρι-

στερά προς τα δεξιά είτε ανάποδα. Η φορά αυτή υπονοεί ότι η μεταβλητή που αντιστοιχεί στη συνιστώσα παίρνει τιμή True ενώ η αντίστροφη φορά σημαίνει τιμή False. Οι συνιστώσες «διαμάντι» συνδέονται μεταξύ τους προσθέτοντας τις ακμές

$$(x_i^{DN}, x_{i+1}^{UP}), i \in \{1, \dots, n-1\}. \quad (11.8)$$

Το γράφημα ολοκληρώνεται περιλαμβάνοντας τη νέες κορυφές - μία για κάθε πρόταση ϕ^j . Για κάθε τέτοια κορυφή εισάγονται έξι ακμές - δύο για κάθε μεταβλητή που εμφανίζεται στην πρόταση. Αν εμφανίζεται ο όρος x στην πρόταση ϕ^j τότε εισάγονται οι ακμές (ϕ_1^j, ϕ^j) και (ϕ^j, ϕ_2^j) , όπου οι κορυφές ϕ_1^j, ϕ_2^j είναι αυτοί που εμφανίζονται στο διαμάντι της συγκεκριμένης μεταβλητής x για την πρόταση ϕ^j . Συμμετρικά, αν εμφανίζεται ο όρος \bar{x} στην πρόταση ϕ^j τότε εισάγονται οι ακμές (ϕ_2^j, ϕ^j) και (ϕ^j, ϕ_1^j) . Το γράφημα $G(V, E)$ που απεικονίζεται στο Σχήμα 11.5 περιέχει όλες τις κορυφές ϕ^j και για τις ϕ^1, ϕ^m περιέχει τις ακμές που υποδηλώνουν ότι ο όρος x_1 εμφανίζεται στην πρώτη ενώ ο \bar{x}_1 στη δεύτερη. Όπως έχουμε ήδη πει το γράφημα θα είναι ολοκληρωμένο όταν για κάθε ϕ^j έχουμε τρία ζευγάρια ακμών - για κάθε όρο που εμφανίζεται στην πρόταση ένα ζευγάρι ακμών προς τις κορυφές ϕ_1^j, ϕ_2^j του διαμαντιού που αντιστοιχεί στη μεταβλητή του όρου αυτού.

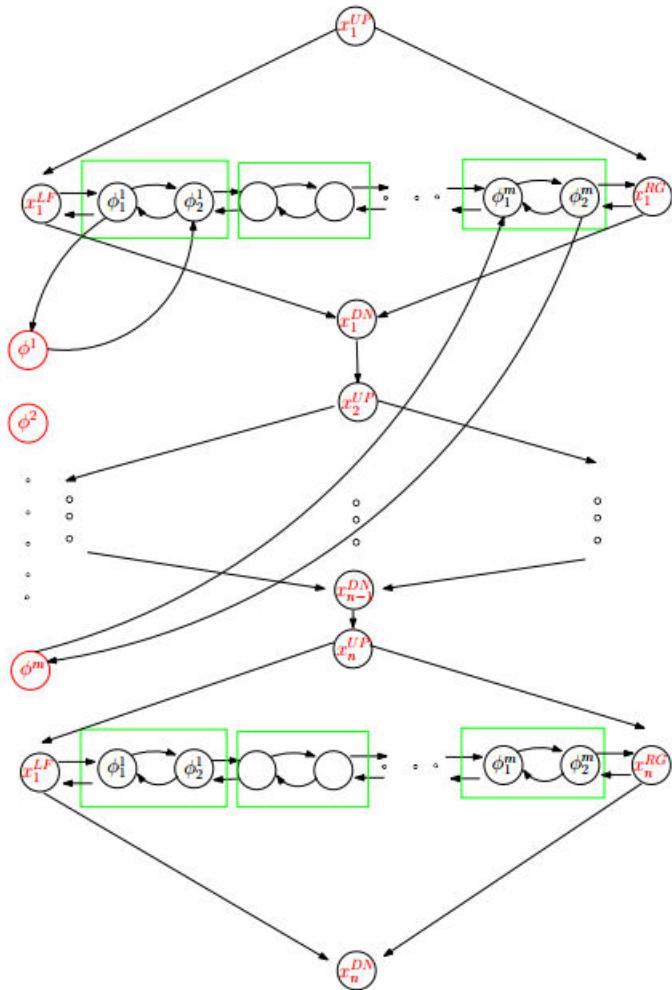
Έστω ότι έχουμε μία αποτίμηση η οποία να ικανοποιεί το στιγμιότυπο 1 – 3SAT. Τότε μπορούμε να θεωρήσουμε ότι ο μονοπάτι που οποίο ξεκινάει από το x_1^{UP} καταλήγει στο x_n^{DN} και διασχίζει τις κορυφές κάθε «διαμαντιού» είτε από αριστερά προς τα δεξιά - αν η μεταβλητή στην συγκεκριμένη αποτίμηση πάρει τιμή True - ή από δεξιά προς τα αριστερά - αν πάρει τιμή False. Προφανώς το μονοπάτι αυτό περνάει από όλους τις κορυφές των διαμαντιών μία ακριβώς φορά και δεν περιλαμβάνει τις κορυφές που αντιστοιχούν στις προτάσεις. Όμως κάθε τέτοια κορυφή μπορούμε να την συμπεριλάβουμε στο μονοπάτι επιλέγοντας για κάθε μία πρόταση τον όρο που σε αυτή γίνεται True. Επειδή το στιγμιότυπο είναι 1 – 3SAT κάθε εξωτερική κορυφή ϕ^j περιλαμβάνεται μόνο μια φορά (στο μονοπάτι).

Αντίστροφα τώρα, έστω ότι στο γράφημα υπάρχει όντας μονοπάτι του Χάμιλτον. Αυτό δεν μπορεί να εγκαταλείπει όντας διαμάντι χωρίς να διασχίζει όλες τις οριζόντιες κορυφές του. Γιατί όμως δεν μπορεί να συμβαίνει αυτό; Προσέξτε ότι ο μόνος τρόπος να γίνει αυτό είναι μέσω κάποιας κορυφής ϕ^j . Έστω λοιπόν ότι είμαστε στο «διαμάντι» που σχετίζεται με τη μεταβλητή x_i . και το μονοπάτι κάνει διάσχιση από τα αριστερά προς τα δεξιά. Έστω ότι το μονοπάτι χρησιμοποιεί την ακμή (ϕ_1^j, ϕ^j) και η επόμενη ακμή οδηγεί έξω από το «διαμάντι», Κάποια στιγμή το μονοπάτι θα πρέπει να επισκεφθεί και την κορυφή ϕ_2^j έχοντας μεταπηδήσει από κάποιο άλλο «διαμάντι» πίσω στο συγκεκριμένο «διαμάντι» (αφού είναι μονοπάτι του Χάμιλτον και άρα όλες οι κορυφές πρέπει να υπάρχουν πάνω σε αυτό). Όμως για να γίνει αυτό θα πρέπει να φτάσει στην κορυφή ϕ_2^j από την κορυφή ϕ_1^{j+1} και άρα η επόμενη κορυφή στο μονοπάτι θα είναι ο ϕ_1^j - άτοπο αφού έχουμε ήδη επισκεφτεί την κορυφή αυτή αφού χρησιμοποιήσαμε την ακμή (ϕ_1^j, ϕ^j) . Η περίπτωση στην οποία διασχίζουμε το διαμάντι από δεξιά προς τα αριστερά είναι απολύτως συμμετρική.

Επομένως το μονοπάτι του Χάμιλτον διασχίζει τις κορυφές ενός διαμαντιού στη σειρά είτε από αριστερά προς τα δεξιά είτε από τα δεξιά προς τα αριστερά. Στην πρώτη περίπτωση θέτουμε την μεταβλητή στην τιμή True ενώ στη δεύτερη στην τιμή False. Αν χρησιμοποιείται στη διάσχιση και την κορυφή της πρότασης ϕ^j τότε η αποτίμηση που δείξαμε χρησιμοποιείται για να ικανοποιεί τη συγκεκριμένη πρόταση.

Τέλος παρατηρούμε ότι στο γράφημα $G(V, E)$ έχουμε

- $4 + 2m$ κορυφές σε κάθε διαμάντι (11.7) και έχουμε n διαμάντια και



Σχήμα 11.5: Γράφημα που αντιστοιχεί σε στιγμότυπο με x_1, \bar{x}_1 να εμφανίζονται στις προτάσεις ϕ^1, ϕ^m αντίστοιχα.

- m κορυφές εξωτερικές (ένας για κάθε πρόταση C_j) από τα διαμάντια.

Συνολικά ο αριθμός των κορυφών είναι $(4+2m)n+m = O(mn)$. Επίσης στο γράφημα $G(V, E)$ έχουμε

- $4m - 6$ ακμές σε κάθε διαμάντι (11.7) και έχουμε n διαμάντια και
- $n - 1$ ακμές για την σύνδεση των διαμαντιών(11.8) και
- $6m$ ακμές οι οποίες συνδέουν τις εξωτερικές κορυφές με τις κορυφές τα διαμάντια.

Συνολικά ο αριθμός των ακμών είναι $(4m - 6)n + n - 1 + 6m = 4mn - 5n + 6m - 1 = O(mn)$.

Από τα παραπάνω προκύπτει ότι η τάξη και το μέγεθος του γραφήματος είναι $O(mn)$ και επομένως ο μετασχηματισμός είναι πολυωνυμικός. \square

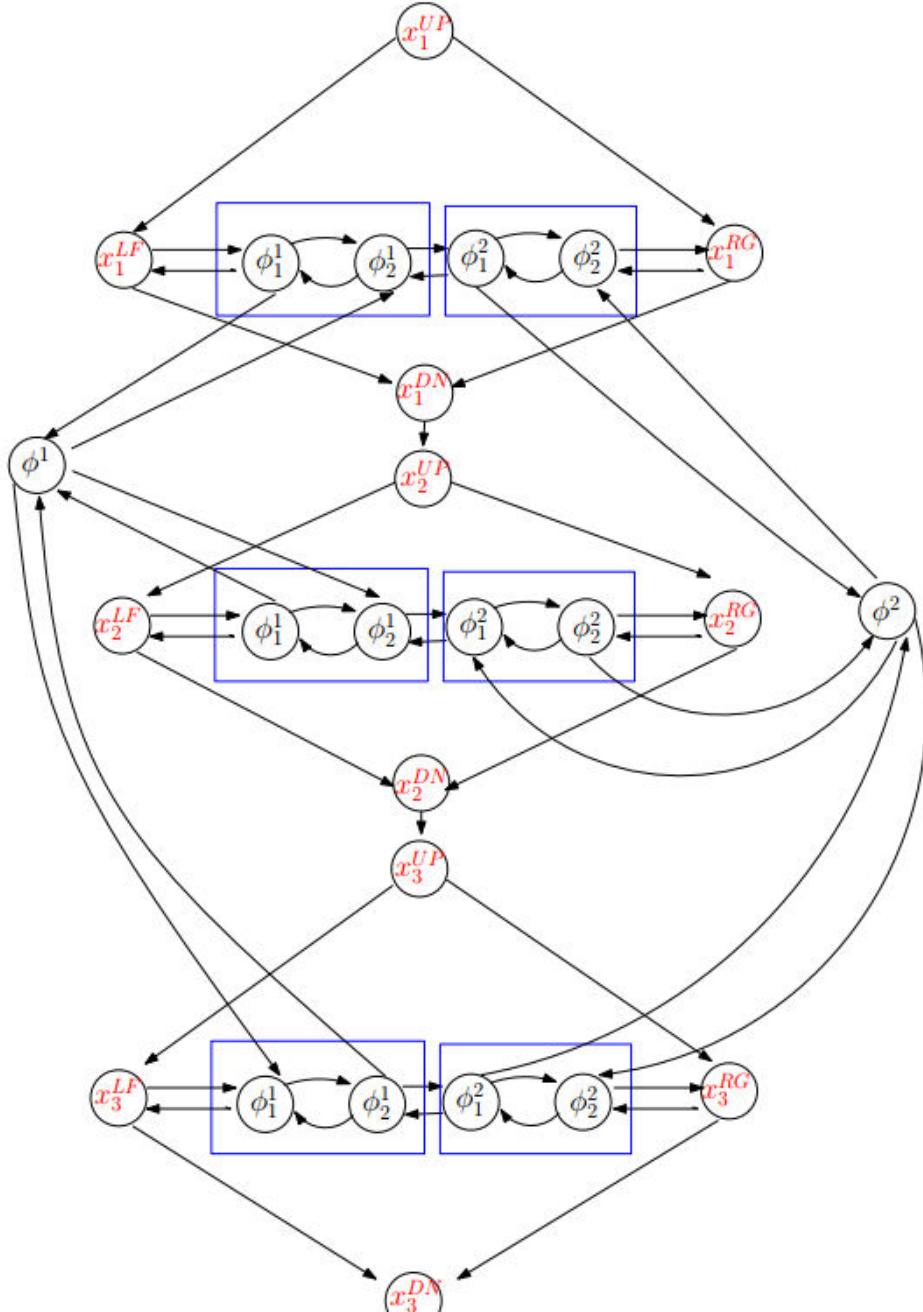
Παράδειγμα

Παράδειγμα 25. Θεωρούμε το ΝΑΙ-στιγμιότυπο του 1 – 3SAT

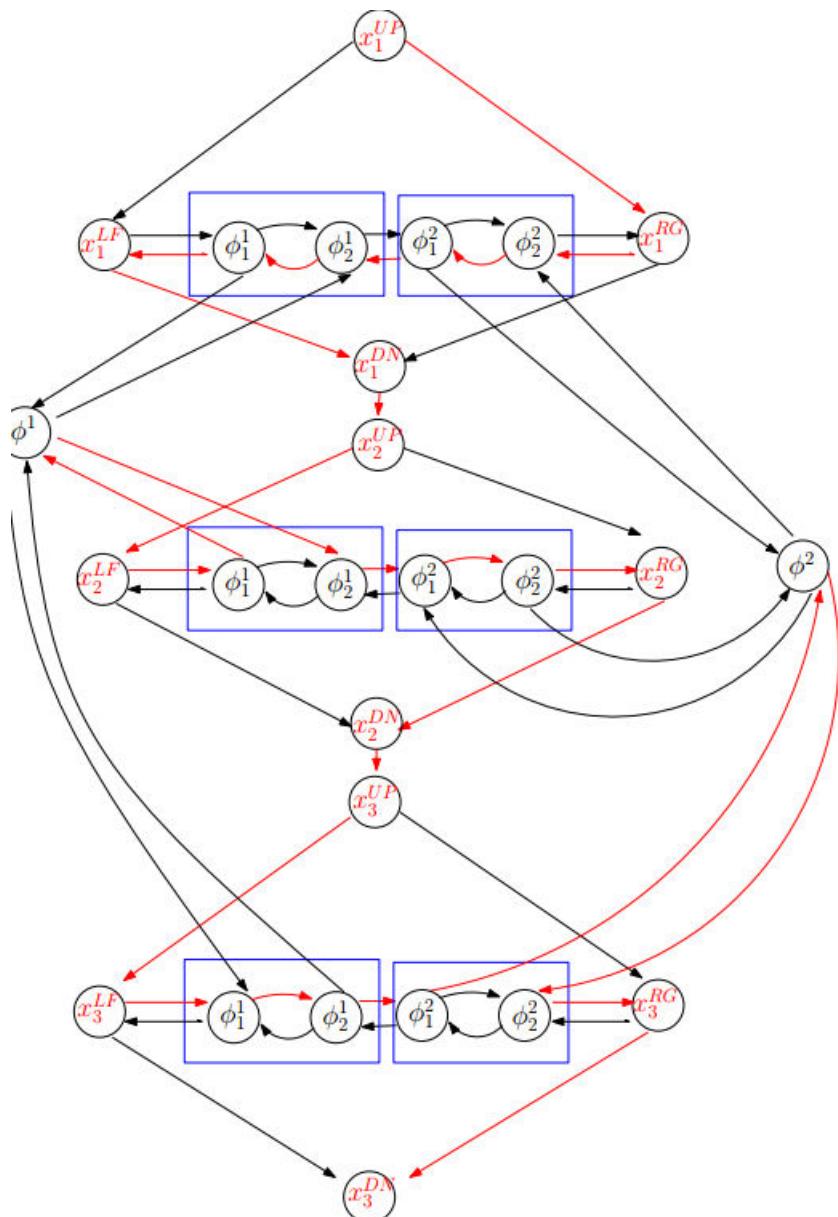
$$x_1 \vee x_2 \vee \bar{x}_3$$

$$x_1 \vee \bar{x}_2 \vee x_3.$$

Το αντίστοιχο γράφημα απεικονίζεται στο Σχήμα 11.6. Έστω τώρα η αποτίμηση $x_1 = \text{False}$, $x_2 = \text{True}$, $x_3 = \text{True}$ η οποία ικανοποιεί το στιγμιότυπο του παραδείγματος. Αντί να περιλάβουμε την ακμή (ϕ_1^1, ϕ_2^1) στο μονοπάτι (στο δεύτερο «διαμάντι») μπορούμε να περιλάβουμε και την κορυφή ϕ^1 στο μονοπάτι χρησιμοποιώντας τις ακμές (ϕ_1^1, ϕ^1) , (ϕ^1, ϕ_2^1) . Ομοίως αντικαθιστούμε στο μονοπάτι την ακμή (ϕ_1^2, ϕ_2^2) με τις (ϕ_1^2, ϕ^2) , (ϕ^2, ϕ_2^2) . Το μονοπάτι απεικονίζεται στο Σχήμα 11.7.



Σχήμα 11.6: $G(V, E)$ για ένα ΝΑΙ-στιγμιότυπο 1 – 3SAT



Σχήμα 11.7: Οι κόκκινες ακμές ανήκουν στο μονοπάτι Χάμιλτον που προκύπτει από την αποτίμηση $x_1 = \text{False}$, $x_2 = \text{True}$, $x_3 = \text{True}$.

01ΙΣΟΤΗΤΑ

Πρόβλημα

Το πρόβλημα αυτό ορίζεται με βάση έναν πίνακα A με στοιχεία από το σύνολο $\{0, 1\}$. Θα αναφερόμαστε σε μία τέτοια δομή ως 0/1-πίνακα ενώ αν ο πίνακας έχει μόνο μία στήλη ως 0/1-διάνυσμα.

Περιγραφή

01ΙΣΟΤΗΤΑ

Στιγμιότυπο: 0/1-πίνακας A

Ερώτηση: Υπάρχει 0/1-διάνυσμα y τέτοιο ώστε $Ay = e$; (όπου e ένα διάνυσμα με στοιχεία ίσα με 1)

Απόδειξη 01ΙΣΟΤΗΤΑ \in NP

Λήμμα 18. 01ΙΣΟΤΗΤΑ \in NP

Απόδειξη. Θεωρούμε ότι ο πίνακας A έχει m γραμμές και n στήλες, δηλαδή $A \in \{0, 1\}^{m \times n}$. Για ένα ΝΑΙ-στιγμιότυπο του παραπάνω προβλήματος, πιστοποιητικό αποτελεί το διάνυσμα y . Μπορούμε να ελέγχουμε την εγκυρότητα του πιστοποιητικού αυτού εκτελώντας $\Theta(mn)$ ΣΥΒ. Αρχικώς διαπιστώνουμε σε $\Theta(n)$ ΣΥΒ αν το y περιέχει ακριβώς n στοιχεία από το σύνολο $\{0, 1\}$. Στη συνέχεια, υπολογίζουμε το διάνυσμα $z = Ay$. Κάθε στοιχείο του διανύσματος αυτού προκύπτει από n πολλαπλασιασμούς και $n - 1$ προσθέσεις. Επειδή το z έχει m στοιχεία, για τον υπολογισμό του απαιτούνται $\Theta(mn)$ ΣΥΒ. Τέλος, συγκρίνονται ένα-προς-ένα τα στοιχεία του z με αυτά του e - δηλαδή εκτελούνται m συγκρίσεις. Συνολικά η διαδικασία επαλήθευσης του πιστοποιητικού y πραγματοποιείται σε $\Theta(mn)$ ΣΥΒ. \square

Απόδειξη 01ΙΣΟΤΗΤΑ \in NP-hard

Λήμμα 19. 01ΙΣΟΤΗΤΑ \in NP-hard

Απόδειξη. Θα πραγματοποιήσουμε αναγωγή από το πρόβλημα 1 – 3SAT: από ένα στιγμιότυπο του 1 – 3SAT θα κατασκευάσουμε ένα στιγμιότυπο του 01ΙΣΟΤΗΤΑ τέτοιο ώστε το δεύτερο να είναι ΝΑΙ-στιγμιότυπο αν και μόνο αν το πρώτο είναι ΝΑΙ-στιγμιότυπο.

Η κατασκευή του πίνακα A γίνεται ως εξής. Οι στήλες αντιστοιχούν στους όρους των μεταβλητών του στιγμιότυπου 1 – 3SAT· ο πίνακας έχει $2n$ στήλες εφόσον το στιγμιότυπο 1 – 3SAT περιλαμβάνει όρους από n (λογικές) μεταβλητές. Αντίστοιχα, ο πίνακας έχει $k = n + m$ γραμμές όπου m ο αριθμός των προτάσεων η σύζευξη των οποίων συνιστά τον τύπο ϕ του στιγμιότυπου 1 – 3SAT. Κάθε μία από τις n πρώτες γραμμές του πίνακα αντιστοιχεί σε μία λογική μεταβλητή. Έχει 1 στις δύο στήλες που αντιστοιχούν στους αντίθετους όρους της μεταβλητής και 0 στις υπόλοιπες. Κάθε μία από τις υπόλοιπες m γραμμές αντιστοιχεί σε μία από τις προτάσεις η σύζευξη των οποίων συνιστά τον τύπο ϕ . Σε κάθε τέτοια πρόταση υπάρχουν τρεις όροι· τα στοιχεία της γραμμής που αντιστοιχούν στους όρους αυτούς έχουν τιμή 1 ενώ τα υπόλοιπα 0.

Παρατηρούμε ότι για να έχει νόημα το γινόμενο Ay , το διάνυσμα y θα πρέπει να έχει $2n$ στοιχεία· δηλαδή, όσοι και οι όροι του στιγμιότυπου 1 – 3SAT. Εφόσον το 1 – 3SAT είναι ΝΑΙ-στιγμιότυπο τότε υπάρχει απόδοση τιμών στους όρους που ικανοποιεί το στιγμιότυπο στην οποία ακριβώς ένας όρος σε κάθε πρόταση έχει τιμή True. Θέτουμε τα αντίστοιχα στοιχεία στο διάνυσμα y στην τιμή 1 ενώ τα υπόλοιπα στην τιμή 0. Για να είναι έγκυρη η απόδοση τιμών στους όρους που ικανοποιεί το στιγμιότυπο 1 – 3SAT δεν μπορεί αντίθετοι όροι να παίρνουν την τιμή True. Αυτό σημαίνει ότι το εσωτερικό γινόμενο κάθε μίας από τις πρώτες n γραμμές του πίνακα A με το y είναι ίσο με 1. Ακολούθως το εσωτερικό γινόμενο για κάθε μία από τις επόμενες m γραμμές του πίνακα A με το διάνυσμα y είναι ίσο με 1· η γραμμή και το διάνυσμα y έχουν και οι δύο 1 μόνο στη θέση του στοιχείου που αντιστοιχεί στο μοναδικό όρο που είναι True στην πρόταση που αντιπροσωπεύει η γραμμή.

Αντίστροφα, έστω ότι έχουμε ένα διάνυσμα y που ικανοποιεί το σύστημα $Ay = e$. Συγκρίνοντας κάθε μία από τις m τελευταίες γραμμές του πίνακα A με το διάνυσμα y παρατηρούμε ότι υπάρχει μόνο μία θέση όπου αμφότερα έχουν τιμή 1. Αποδίδουμε τιμή True στον όρο που αντιστοιχεί στη θέση αυτή και False στους υπόλοιπους όρους που αντιστοιχούν σε θέσεις με τιμή 1 στη γραμμή αυτή. Άρα υπάρχει μόνο ένας όρος που παίρνει τιμή True σε κάθε μία από τις προτάσεις του στιγμιότυπου 1 – 3SAT. Αυτή η απόδοση τιμών είναι έγκυρη αφού το γεγονός ότι το εσωτερικό γινόμενο κάθε μία από τις n πρώτες γραμμές με το διάνυσμα y είναι ίσο με 1 διασφαλίζει ότι δεν αποδίδεται ταυτόχρονα σε κάποιον όρο και στον αντίθετο του η τιμή True.

Τέλος η αναγωγή είναι πολυωνυμική αφού ο αριθμός των στοιχείων του πίνακα A (μέγεθος του στιγμιότυπου 01ΙΣΟΤΗΤΑ) είναι ίσος με $2n(n + m)$. \square

Παράδειγμα

Θα δείξουμε την κατασκευή του πίνακα A που πραγματοποιήθηκε στην απόδειξη του Λήμματος 19 με ένα παράδειγμα.

Παράδειγμα 26. Θεωρούμε το ακόλουθο ΝΑΙ-στιγμιότυπο του 1 – 3SAT

$$\phi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \bar{x}_3).$$

Ο πίνακας A που κατασκευάζεται στην απόδειξη του Λήμματος 19 είναι

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

, όπου οι στήλες αντιστοιχούν (με τη σειρά εμφάνισης) στους όρους $x_1, x_2, x_3, \bar{x}_1, \bar{x}_2, \bar{x}_3$.

Η απόδοση τιμών με ακριβώς έναν όρο να παίρνει τιμή True σε κάθε πρόταση είναι

$$\bar{x}_1 = \text{True}, \bar{x}_2 = \text{True}, \bar{x}_3 = \text{True}, \quad y = (0, 0, 0, 1, 1, 1)^T.$$

ΥΠΟΣΥΝΑΘΡ

Πρόβλημα

Το πρόβλημα αυτό είναι καθαρά αριθμητικό και διατυπώνεται ως εξής.
ΤΠΟΣΤΥΝΑΘΡ

Περιγραφή

Στιγμιότυπο: Ακέραιοι d_1, d_2, \dots, d_n, t

Ερώτηση: Υπάρχει υποσύνολο $J \subseteq \{1, \dots, n\}$ τέτοιο ώστε $\sum_{j \in J} d_j = t$;

Απόδειξη ΥΠΟΣΥΝΑΘΡ \in NP

Λήμμα 20. ΤΠΟΣΤΥΝΑΘΡ \in NP.

Απόδειξη. Προφανώς το πιστοποιητικό σε ένα ΝΑΙ-στιγμιότυπο είναι ένα σύνολο J . Εκτελώντας $O(n)$ ΣΥΒ μπορούμε να διαπιστώσουμε αν $J \subseteq \{1, \dots, n\}$ και στη συνέχεια εκτελώντας $O(n)$ προσθέσεις και μία σύγκριση μπορούμε να υπολογίσουμε το άθροισμα $\sum_{j \in J} d_j$ και να το συγκρίνουμε με το t \square

Απόδειξη ΥΠΟΣΥΝΑΘΡ \in NP-hard

Λήμμα 21. ΤΠΟΣΤΥΝΑΘΡ \in NP-complete

Απόδειξη. Θα κάνουμε αναγωγή από το πρόβλημα 01ΙΣΟΤΗΤΑ. Θεωρούμε έναν 0/1-πίνακα με m γραμμές και n στήλες. Θα ορίσουμε αριθμούς d_1, \dots, d_n και t τέτοιους ώστε να υπάρχει $J \subseteq \{1, \dots, n\}$ για το οποίο να ισχύει ότι $\sum_{j \in J} d_j = t$ αν και μόνο αν υπάρχει 0/1-διάνυσμα y τέτοιο ώστε $Ay = e$, όπου e είναι ένα διάνυσμα που περιέχει μόνο 1.

Θεωρούμε ένα αριθμητικό σύστημα κωδικοποίησης με βάση $b \geq n+1$ και ορίζουμε $t = 1 \cdots 1$. Αντίστοιχα θεωρούμε ότι η στήλη $A_j, j \in \{1, \dots, n\}$, του πίνακα A κωδικοποιεί τον αριθμό d_j στο αριθμητικό σύστημα με βάση b .

Αν υπάρχει 0/1-διάνυσμα y τέτοιο ώστε $Ay = e$, τότε οι θέσεις στις οποίες το y περιέχουν τιμή 1 υποδεικνύουν τους αριθμούς (υποσύνολο J του συνόλου $\{1, \dots, n\}$) το άθροισμα των οπίων ισούται με το t (όπως αυτό κωδικοποιείται στο αριθμητικό σύστημα με βάση το b). Αντίστροφα, θεωρούμε τους αριθμούς d_1, \dots, d_n οι οποίοι προκύπτουν από τις στήλες ενός 0/1-πίνακα A (διάστασης $m \times n$) αν θεωρήσουμε κάθε στήλη ως την κωδικοποίηση ενός αριθμού στο αριθμητικό σύστημα με βάση b . Επίσης θεωρούμε τον αριθμό t που κωδικοποιεί στο σύστημα αυτό το διάνυσμα $(1, 1, \dots, 1)$.

Προφανώς αν κάποιοι αριθμοί από τους d_1, \dots, d_n έχουν άθροισμα τον t τότε το αντίστοιχο διάνυσμα δεικτών J υποδεικνύει τις θέσεις σε ένα 0/1-διάνυσμα y (με n στοιχεία) που θα έχουν τιμή 1. Το εσωτερικό γινόμενο του y με κάθε γραμμή του πίνακα A θα είναι ίση με 1.

Η απαίτηση $b \geq n + 1$ διασφαλίζει ότι το άθροισμα των υποδεικνυόμενων στηλών από το διάνυσμα y διασφαλίζει ότι δεν μπορεί να ξεπεραστεί ο αριθμός που κωδικοποιείται ως $(1, 1, \dots, 1)$. Αυτό θα μπορούσε να συμβεί αν δεν ίσχυε η παραπάνω απαίτηση από την ύπαρξη κρατουμένων στο άθροισμα των στηλών.

Παράδειγμα

Παράδειγμα 27. Θεωρούμε τον πίνακα

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Αν θεωρήσουμε ότι οι στήλες εκφράζουν δεκαδικούς αριθμούς (σύστημα αρίθμησης με βάση $b = 10$) τότε

$$\begin{aligned} d_1 &= A_1 = 1 \cdot 10^4 + 0 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 1 \cdot 10^0 = 10011, \\ d_2 &= A_2 = 0 \cdot 10^4 + 1 \cdot 10^3 + 0 \cdot 10^2 + 0 \cdot 10^1 + 0 \cdot 10^0 = 1000, \\ d_3 &= A_3 = 0 \cdot 10^4 + 0 \cdot 10^3 + 1 \cdot 10^2 + 0 \cdot 10^1 + 0 \cdot 10^0 = 100, \\ d_4 &= A_4 = 1 \cdot 10^4 + 0 \cdot 10^3 + 0 \cdot 10^2 + 0 \cdot 10^1 + 0 \cdot 10^0 = 10000, \\ d_5 &= A_5 = 0 \cdot 10^4 + 1 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 0 \cdot 10^0 = 1010, \\ d_6 &= A_6 = 0 \cdot 10^4 + 0 \cdot 10^3 + 1 \cdot 10^2 + 0 \cdot 10^1 + 1 \cdot 10^0 = 101. \end{aligned}$$

Επίσης

$$t = 1 \cdot 10^4 + 1 \cdot 10^3 + 1 \cdot 10^2 + 1 \cdot 10^1 + 1 \cdot 10^0 = 11111.$$

Το t παράγεται από το διάνυσμα $y = (1, 1, 1, 0, 0, 0)$ ως

$$t = 1 \cdot 10011 + 1 \cdot 1000 + 1 \cdot 100 + 0 \cdot 10000 + 0 \cdot 1010 + 0 \cdot 101 = 11111$$

και αντίστοιχα από το διάνυσμα $y = (0, 0, 0, 1, 1, 1)$ ως

$$t = 0 \cdot 10011 + 0 \cdot 1000 + 0 \cdot 100 + 1 \cdot 10000 + 1 \cdot 1010 + 1 \cdot 101 = 11111.$$

Άρα από το σύνολο των αριθμών $\{10011, 1000, 100, 10000, 1010, 101\}$ ο αριθμός $t = 11111$ παράγεται είτε ως άθροισμα των τριών πρώτων ($J = \{1, 2, 3\}$) ή των τριών τελευταίων ($J = \{4, 5, 6\}$).

Πρόβλημα

Λύση Η δομή kIKAN υπονοεί το πρόβλημα IKAN όταν υπάρχουν ακριβώς k όροι σε κάθε πρόταση.

Απόδειξη $k\text{IKAN} \in \text{NP}$

Επομένως η απόδειξη ότι $k\text{IKAN}$ ανήκει στην τάξη NP είναι ακριβώς ανάλογη με την αντίστοιχη απόδειξη για 3IKAN και παραλείπεται.

Απόδειξη $k\text{IKAN} \in \text{NP-hard}$

τατ. Στη συνέχεια θα δείξουμε $3\text{IKAN} \leq_P k\text{IKAN}$. Θεωρούμε ένα στιγμιότυπο 3IKAN σε ΣΚΜ, δηλαδή,

$$\phi = \bigwedge_{j=1}^m C_j \quad (1)$$

όπου

$$C_j = a_j \vee b_j \vee c_j, \quad j \in \{1, \dots, m\}. \quad (2)$$

Το αντίστοιχο στιγμιότυπο $k\text{IKAN}$ αποτελείται από σύζευξη των προτάσεων

$$C'_j = a_j \vee b_j \vee c_j \vee z_1 \vee \dots \vee z_{k-3}, \quad j \in \{1, \dots, m\}, \quad (3)$$

όπου $z_1 \vee \dots \vee z_{k-3}$ νέες μεταβλητές που δεν υπάρχουν στο σύστημα 3IKAN. Μία αποτίμηση T στο στιγμιότυπο $k\text{IKAN}$ είναι ακριβώς ίδια όπως και στο 3IKAN για όλες τις μεταβλητές που υπάρχουν και στα δύο στιγμιότυπα ενώ για τις επιπλέον μεταβλητές z , έχουμε $T(z_1) = T(z_2) = \dots = T(z_{k-3}) = \text{False}$. Το στιγμιότυπο $k\text{IKAN}$ έχει ακριβώς τον ίδιο αριθμό προτάσεων με το στιγμιότυπο 3IKAN ενώ έχει παραπάνω $k - 3$ μεταβλητές και επομένως η αναγωγή είναι πολυωνυμική.

ΔΙΠΛΗ-ΙΚΑΝ

Περιγραφή

Στιγμιότυπο: Έκφραση ϕ σε ΚΣΜ.

Ερώτηση: Υπάρχουν τουλάχιστον δύο αποτιμήσεις που να ικανοποιούν την ϕ .

Απόδειξη $\Delta\text{ΙΚΑΝ} \in \text{NP}$

Λύση Η απόδειξη ότι $\Delta\text{ΙΚΑΝ} \in \text{NP}$ είναι ανάλογη της αντίστοιχης απόδειξης για το πρόβλημα IKAN και επομένως παραλείπεται.

Απόδειξη ΔΙΠΛΗ-ΙΚΑΝΕΝP-hard

Θα δείξουμε $\text{IKAN}_{\leq P} \text{ΔΙΠΛΗ-ΙΚΑΝ}$. Θεωρούμε ένα στιγμιότυπο ΙΚΑΝ σε ΣΚΜ (βλέπε (1), (2)). Κατασκευάζουμε ένα αντίστοιχο στιγμιότυπο ΔΙΠΛΗ-ΙΚΑΝ ως εξής. Για κάθε πρόταση C_j εισάγουμε δύο προτάσεις, ήτοι

$$A_j = C_j \vee z, \bar{A}_j = C_j \vee \neg z,$$

όπου z είναι μία νέα μεταβλητή. Το στιγμιότυπο ΔΙΠΛΗ-ΙΚΑΝ είναι

$$\phi' = \bigwedge_{j=1}^m (A_j \wedge \bar{A}_j).$$

Η αναγωγή είναι σωστή αφού $A_j \wedge \bar{A}_j$ ικανοποιείται ANN C_j ικανοποιείται όποια αποτίμηση και να θεωρήσουμε για τη μεταβλητή z . Επίσης είναι πολυωνυμική αφού η ϕ' περιέχει $2m$ προτάσεις και μία επιπλέον μεταβλητή (τη z).

ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ

Περιγραφή

Στιγμιότυπο: Έκφραση ϕ σε ΚΣΜ με ακριβώς τέσσερις όρους σε κάθε πρόταση.

Ερώτηση: Υπάρχει κάποια αποτίμηση που να ικανοποιεί την ϕ και να μην δίνει σε όλους τους όρους κάθε πρότασης την ίδια τιμή;

Απόδειξη ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝΕΝP

Λύση Και για τα δύο προβλήματα είναι εύκολο να δειχθεί ότι ανήκουν στην τάξη NP και επομένως η αντίστοιχη απόδειξη παραλείπεται. Προχωράμε για

Απόδειξη ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝΕΝP-hard

1. Θα δείξουμε ότι $\text{3IKAN}_{\leq P} \text{ ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ}$. Θεωρούμε ένα 3IKAN στιγμιότυπο της μορφής (1), (2). Για κάθε πρόταση C_j εισάγουμε στο ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ στιγμιότυπο την πρόταση

$$A_j = C_j \vee z,$$

όπου z είναι μία νέα μεταβλητή. Για οποιαδήποτε αποτίμηση T , θεωρούμε την επέκταση της με $T(z) = \text{False}$. Προσέξτε ότι A_j ικανοποιείται ANN C_j ικανοποιείται στην περίπτωση κατά την οποία τουλάχιστον ένας από τους όρους a_j, b_j, c_j παίρνει την τιμή True . Όμως τότε η πρόταση A_j ικανοποιείται και έχει και έναν όρο με άλλη τιμή (αφού $T(z) = \text{False}$, για κάθε αποτίμηση T).

Περιγραφή

Στιγμιότυπο: Έκφραση ϕ σε KSM με ακριβώς τρεις όρους σε κάθε πρόταση.

Ερώτηση: Υπάρχει κάποια αποτίμηση που να ικανοποιεί την ϕ και να μην δίνει σε όλους τους όρους κάθε πρότασης την ίδια τιμή;

Απόδειξη ΟΧΙ-ΟΛΑ-ΙΔΙΑ-3ΙΚΑΝ ∈ NP

Λύση Και για τα δύο προβλήματα είναι εύκολο να δειχθεί ότι ανήκουν στην τάξη NP και επομένως η αντίστοιχη απόδειξη παραλείπεται. Προχωράμε για

Απόδειξη ΟΧΙ-ΟΛΑ-ΙΔΙΑ-3ΙΚΑΝ ∈ NP-hard

- Θα δείξουμε ότι $\text{ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ} \leq_p \text{ΟΧΙ-ΟΛΑ-ΙΔΙΑ-3ΙΚΑΝ}$. Εστω το στιγμιότυπο ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ που περιγράφεται από την (1) και από

$$C_j = a_j \vee b_j \vee c_j \vee d_j, \quad j \in \{1, \dots, m\}.$$

Για κάθε τέτοια πρόταση, εισάγουμε στο στιγμιότυπο ΟΧΙ-ΟΛΑ-ΙΔΙΑ-3ΙΚΑΝ τις δύο προτάσεις

$$\begin{aligned} A_j &= a_j \vee b_j \vee w_j, \\ B_j &= c_j \vee d_j \vee \neg w_j, \end{aligned}$$

όπου w_j είναι νέα μεταβλητή. Προσέξτε ότι επειδή η ϕ αποτελεί στιγμιότυπο ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ τουλάχιστον ένας από τους όρους της C_j θα παίρνει την τιμή *False* και ένας την τιμή *True* σε κάθε αποτίμηση T που την ικανοποιεί. Χωρίς βλάβη της γενικότητας, λόγω συμμετρίας, μπορούμε να θεωρήσουμε ότι $T(a_j) = \text{False}$ και $T(c_j) = \text{True}$. Επεκτείνοντας την αποτίμηση θέτουμε

$$T(w_j) = \begin{cases} \text{True}, & \text{av } (T(b_j) = \text{False}) \vee (T(d_j) = \text{True}), \\ \text{False}, & \text{διαφορετικά,} \end{cases}$$

το στιγμιότυπο ΟΧΙ-ΟΛΑ-ΙΔΙΑ-3ΙΚΑΝ

$$\phi' = \bigwedge_{j=1}^m (A_j \wedge B_j)$$

ικανοποιείται ANN το ΟΧΙ-ΟΛΑ-ΙΔΙΑ-4ΙΚΑΝ ικανοποιείται. Επίσης τόσο στην πρόταση A_j όσο και στην πρόταση B_j δεν έχουν όλοι οι όροι τις ίδιες τιμές. Η έκφραση ϕ' έχει $2 * m$ προτάσεις και m νέες μεταβλητές και άρα ο μετασχηματισμός είναι πολυωνυμικός.

ΑΝΣΥΝΟΛΟ

Πρόβλημα

Άσκηση 8 Σε ένα μη-κατευθυνόμενο γράφημα $G(V, E)$ ένα ανεξάρτητο σύνολο είναι ένα σύνολο $V' \subseteq V$ τέτοιο ώστε για κάθε $v, u \in V'$ δεν υπάρχει ακμή (v, u) . Μία κάλυψη κόμβων είναι ένα σύνολο $\hat{V} \subseteq V$ τέτοιο ώστε για κάθε ακμή (v, u) είτε $v \in \hat{V}$ ή $u \in \hat{V}$. Ορίζουμε τα ακόλουθα προβλήματα

Περιγραφή

Στιγμιότυπο: Μη-κατευθυνόμενο γράφημα $G(V, E)$, ακέραιος k

Ερώτηση: Υπάρχει ένα ανεξάρτητο σύνολο μεγέθους k ;

Απόδειξη ΑΝΣΥΝΟΛΟ \in NP-complete

- Κατασκευάζουμε το συμπληρωματικό γράφημα του $G(V, E)$ το οποίο ορίζεται σαν $G(\bar{V}, \bar{E})$ όπου $\bar{V} = V$ και $\bar{E} = \{(v, u) : v, u \in V, (v, u) \notin E\}$. Προφανώς ένα clique μεγέθους k στο $G(\bar{V}, \bar{E})$ αποτελεί ένα ανεξάρτητο σύνολο στο $G(V, E)$.

ΚΑΛΥΨΗΚΟΜΒΩΝ

Πρόβλημα

Άσκηση 8 Σε ένα μη-κατευθυνόμενο γράφημα $G(V, E)$ ένα ανεξάρτητο σύνολο είναι ένα σύνολο $V' \subseteq V$ τέτοιο ώστε για κάθε $v, u \in V'$ δεν υπάρχει ακμή (v, u) . Μία κάλυψη κόμβων είναι ένα σύνολο $\hat{V} \subseteq V$ τέτοιο ώστε για κάθε ακμή (v, u) είτε $v \in \hat{V}$ ή $u \in \hat{V}$. Ορίζουμε τα ακόλουθα προβλήματα

Περιγραφή

Στιγμιότυπο: Μη-κατευθυνόμενο γράφημα $G(V, E)$, ακέραιος k

Ερώτηση: Υπάρχει μία κάλυψη μεγέθους k ;

Απόδειξη ΚΑΛΥΨΗΚΟΜΒΩΝ \in NP-complete

- Έστω $S \subseteq V$ ένα ανεξάρτητο σύνολο του $G(V, E)$. Τότε το σύνολο $V \setminus S$ αποτελεί μία κάλυψη κόμβων του $G(V, E)$ (γιατί;) Άρα αρκεί να αναζητήσουμε ένα ανεξάρτητο σύνολο μεγέθους $n - k$ στο $G(V, E)$, όπου n ο αριθμός των κόμβων του $G(V, E)$.

3ΧΡΩΜΑΤΙΣΜΟΣ

Περιγραφή

Στιγμιότυπο: Μη-κατευθυνόμενο γράφημα $G(V, E)$, χρώματα T, F, B

Ερώτηση: Υπάρχει αντιστοίχηση χρωμάτων στους κόμβους του γραφήματος τέτοια ώστε κάθε ζευγάρι κόμβων που συνδέεται με ακμή να χρωματίζεται με διαφορετικό χρώμα;

Απόδειξη 3ΧΡΩΜΑΤΙΣΜΟΣεNP-complete

Λύση Θα κάνουμε αναγωγή από το πρόβλημα 3IKAN. Θεωρούμε λοιπόν ένα στιγμιότυπο 3IKAN στη γενική μορφή των (1) και (2). Θα κατασκευάσουμε ένα γράφημα $G(V, E)$ το οποίο θα είναι χρωματίσιμο με τρία χρώματα ANN το 3IKAN είναι ικανοποιήσιμο. Θεωρούμε το σύνολο των τριών χρωμάτων

$$\{T(\text{true}), F(\text{false}), B(\text{ase})\}. \quad (6)$$

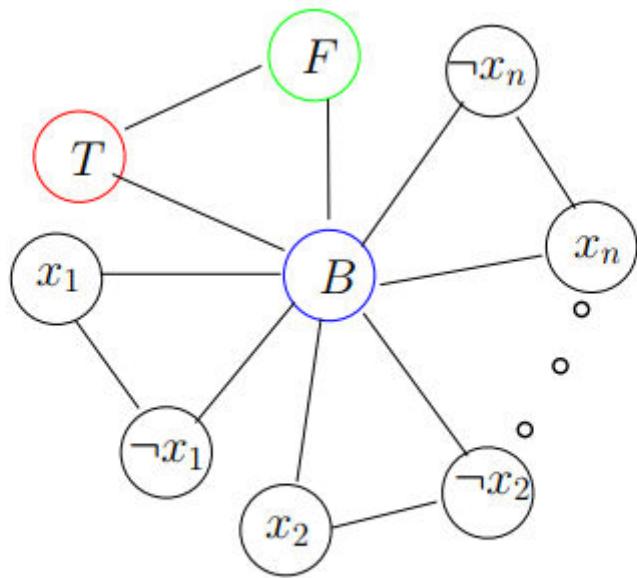
Θα χρησιμοποιήσουμε δύο γραφικές συνιστώσες· η πρώτη αφορά τους όρους και η δεύτερη τις προτάσεις. Η γραφική συνιστώσα που αφορά τους όρους απεικονίζεται στο Σχήμα 8.

Για κάθε πρόταση $C_j = a_j \vee b_j \vee c_j$ έχουμε μία γραφική (προτασιακή) συνιστώσα που απεικονίζεται στο Σχήμα 9. Παρατηρείστε ότι αν οι κόμβοι a, b, c λάβουν το χρώμα *False* (δηλαδή η C_j και άρα η ϕ δεν είναι ικανοποιήσιμη) τότε ο κόμβος C_j στο γράφημα (κόμβος που αντιστοιχεί στην πρόταση $a \vee b \vee c$) αναγκαστικά πρέπει να πάρει το ίδιο χρώμα (δηλαδή $F(\text{false})$). Διαφορετικά, αν τουλάχιστον ένας από τους όρους πάρει την τιμή *True* υπάρχει χρωματισμός των υπολοίπων κόμβων (από το σύνολο χρωμάτων (6)) ώστε ο κόμβος αυτός να πάρει το χρώμα *True*.

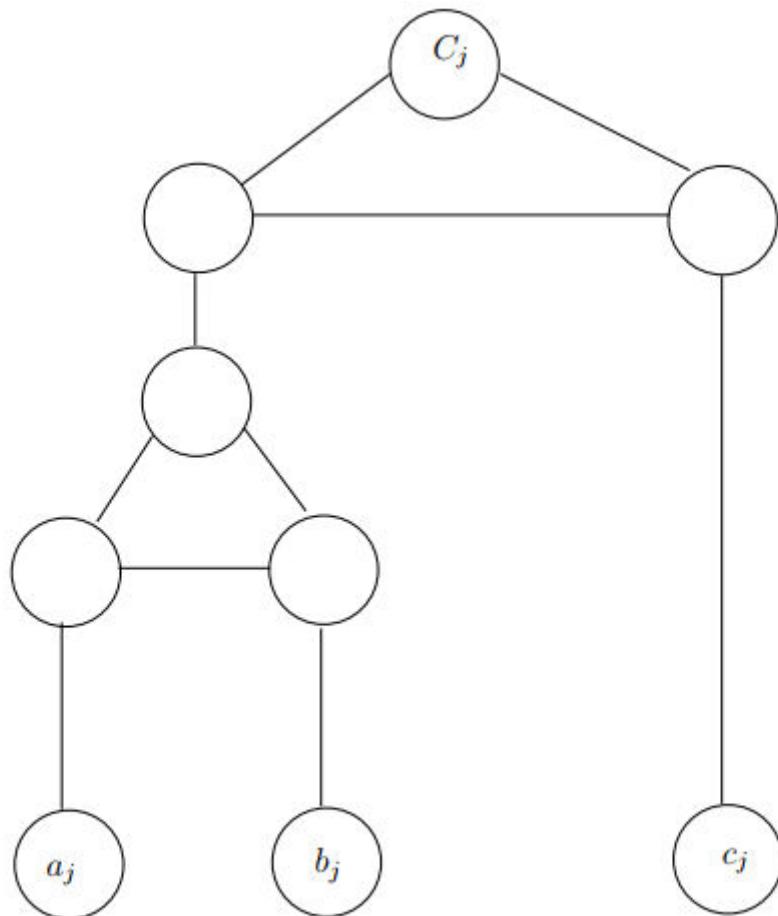
Το γράφημα $G(V, E)$ απεικονίζεται στο Σχήμα 10 με μία συνιστώσα που αφορά την πρόταση C_j . Το γράφημα έχει m τέτοιες συνιστώσες - μία για κάθε τιμή του δείκτη j - με τον αντίστοιχο κόμβο C_j να συνδέεται με ακμή με τους κόμβους B και $F(\text{false})$.

Αν η ϕ (δες (1)) είναι ικανοποιήσιμη τότε οι κόμβοι που αφορούν τις μεταβλητές παίρνουν τα χρώματα *True*, *False* από την αποτίμηση που ικανοποιεί την έκφραση. Το ίδιο συμβαίνει για τους κόμβους a_j, b_j, c_j σε κάθε προτασιακή συνιστώσα - θα πρέπει να χρησιμοποιηθούν τα χρώματα από την αποτίμηση και για τους κόμβους αυτούς. Τέλος οι εσωτερικοί κόμβοι της κάθε προτασιακής συνιστώσας χρωματίζονται κατάλληλα από το σύνολο των χρωμάτων (6) ώστε ο κάθε κόμβος C_j να παίρνει το χρώμα $T(\text{true})$. Αντίστροφα, αν το γράφημα είναι 3-χρωματίσιμο τότε αναγκαστικά οι κόμβοι C_j παίρνουν το ίδιο χρώμα (δηλαδή $T(\text{true})$). Επίσης υπάρχει χρωματισμός όπου κάθε κόμβος με βαθμό 1 (σε κάθε προτασιακή συνιστώσα) παίρνει το ίδιο χρώμα με τον κόμβο που αντιστοιχεί στον ίδιο όρο στη γραφική συνιστώσα που αφορά τις μεταβλητές (αφού οι κόμβοι αυτοί δεν ενώνονται με ακμή).

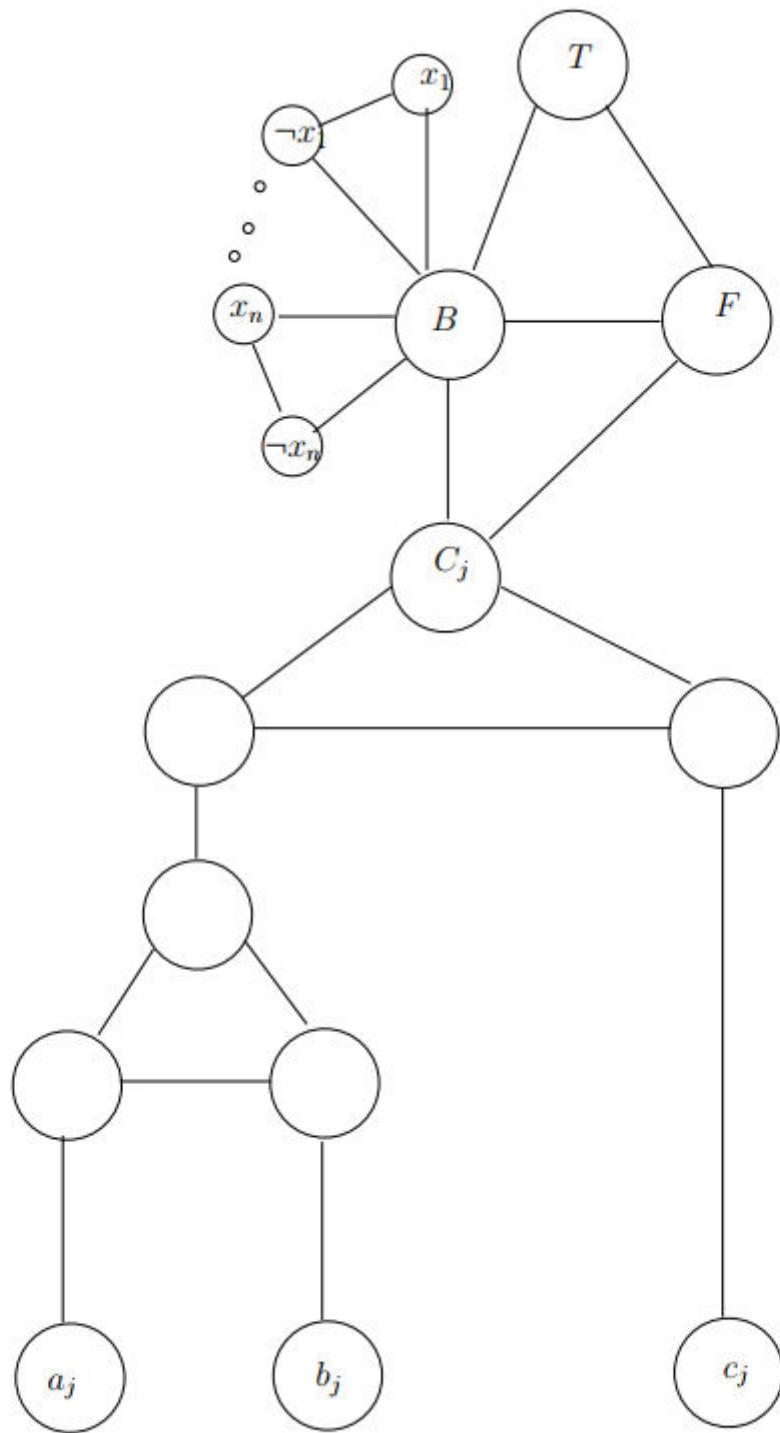
Το $G(V, E)$ έχει m συνιστώσες που αφορούν προτάσεις - κάθε μία έχει 9 κόμβους, 2 n κόμβους για τους όρους και 3 κόμβους με τα χρώματα $B(\text{ase})$, $F(\text{false})$, $T(\text{true})$. Συνολικά το G περιέχει $9m + 2n + 3$ κόμβους και άρα η μετατροπή είναι πολυωνυμική.



Σχήμα 8: Συνιστώσα που αφορά τους όρους (3ΧΡΩΜΑΤΙΣΜΟΣ)



Σχήμα 9: Συνιστώσα για την πρόταση C_j (3ΧΡΩΜΑΤΙΣΜΟΣ)



Σχήμα 10: Το $G(V, E)$ σε απλοποιημένη έκδοση - με μία πρόταση C_j (3ΧΡΩΜΑΤΙΣΜΟΣ)

ΔΙΑΧΩΡΙΣΜΟΣ

Περιγραφή

Στιγμιότυπο: Σύνολο ακεραιών S

Ερώτηση: Υπάρχουν σύνολα $S_1, S_2 \subset S$ με $S_1 \cup S_2 = S, S_1 \cap S_2 = \emptyset$ τέτοια ώστε $\sum\{s : s \in S_1\} = \sum\{s : s \in S_2\}$; Να δείξετε ότι τα παραπάνω προβλήματα

Απόδειξη ΔΙΑΧΩΡΙΣΜΟΣΕΝΠ

Λύση Προφανώς το πρόβλημα ανήκει στην τάξη ΝΠ.

Απόδειξη ΔΙΑΧΩΡΙΣΜΟΣΕΝP-hard

νυμική αναγωγή από το πρόβλημα ΥΠΟΣΥΝΑΘΡ. Σε ένα NAI στιγμιότυπο του προβλήματος αυτού μας δίνεται μία τιμή t και ένα σύνολο από ακέραιους $D = \{d_1, \dots, d_n\}$ και υπάρχει $J \subset \{1, \dots, n\}$ τέτοιο ώστε

$$\sum_{j \in J} d_j = t. \quad (7)$$

Δημιουργούμε το σύνολο $S = D \cup \{2t - s\}$ όπου $s = \sum_{i=1, \dots, n} d_i$. Το άθροισμα των στοιχείων του S είναι $2t$. Επίσης μέσα στο D υπάρχει ένα υποσύνολο στοιχείων που αθροίζει σε t (αυτά που δεικτοδοτούνται από το J) και άρα τα υπόλοιπα αθροίζουν πάλι σε t (αφού το άθροισμα του συνόλου των στοιχείων του S είναι $2t$). Άρα υπάρχει ο ζητούμενος διαχωρισμός του S αν το στιγμιότυπο του ΥΠΟΣΥΝΑΘΡ είναι NAI.

Αντίστροφα αν ισχύει ότι το S διαχωρίζεται σε δύο σύνολα S_1, S_2 με $\sum\{d : d \in S_1\} = \sum\{d : d \in S_2\}$, θα δείξουμε ότι υπάρχει σύνολο $J \subset \{1, \dots, n\}$ τέτοιο ώστε να ισχύει η (7). Εφόσον ισχύει ο παραπάνω διαχωρισμός του S σε S_1, S_2 , μόνο το ένα από τα δύο αυτά σύνολα έχει το επιπλέον στοιχείο $2t - s$. Έστω ότι αυτό είναι το σύνολο S_2 . Επομένως,

$$\sum\{d : d \in S_1\} = \sum\{d : d \in S_2 \setminus \{2t - s\}\} + 2t - s.$$

Παρατηρούμε ότι $S_2 \setminus \{2t - s\} \subset \{d_1, \dots, d_n\}$ και επομένως

$$\sum\{d : d \in S_1\} = s - \sum\{d : d \in S_2 \setminus \{2t - s\}\}.$$

Από τις δύο τελευταίες σχέσεις προκύπτει ότι

$$s - \sum\{d : d \in S_2 \setminus \{2t - s\}\} = \sum\{d : d \in S_2 \setminus \{2t - s\}\} + 2t - s,$$

και επομένως

$$\begin{aligned} t &= s - \sum\{d : d \in S_2 \setminus \{2t - s\}\} \Rightarrow \\ t &= \sum_{j \in J} d_j, \end{aligned}$$

όπου J δεικτοδοτεί αποκλειστικά στοιχεία από το σύνολο $\{d_1, \dots, d_n\}$.