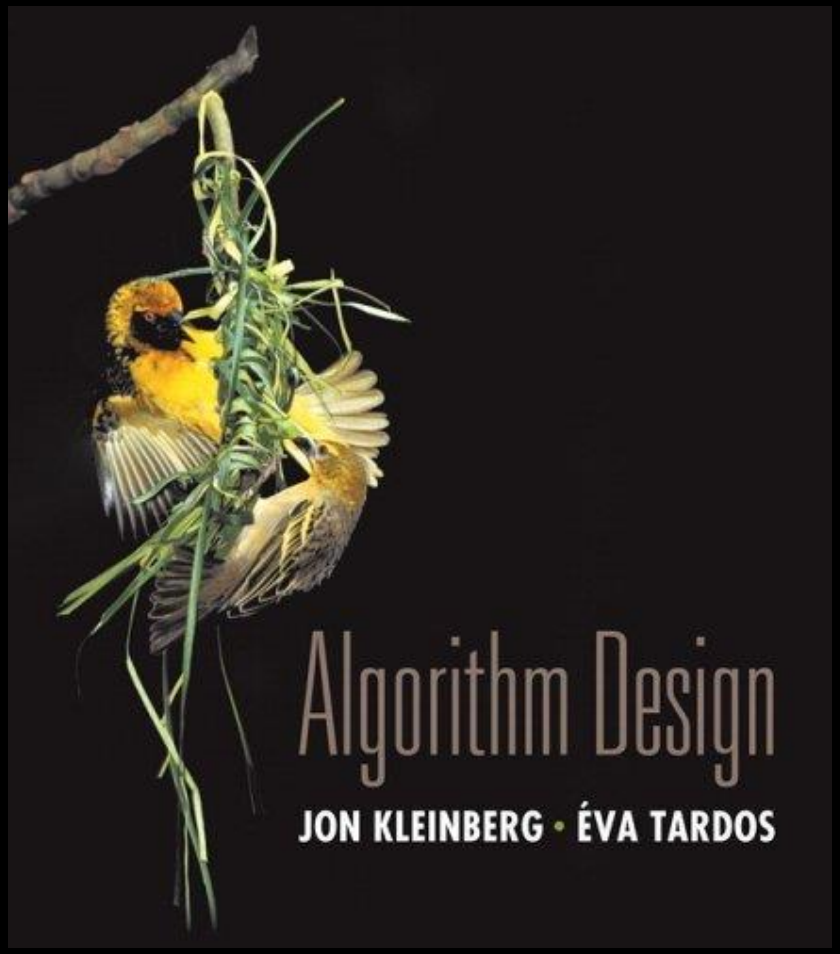


Επέκταση των Ορίων της Επιλυσιμότητας

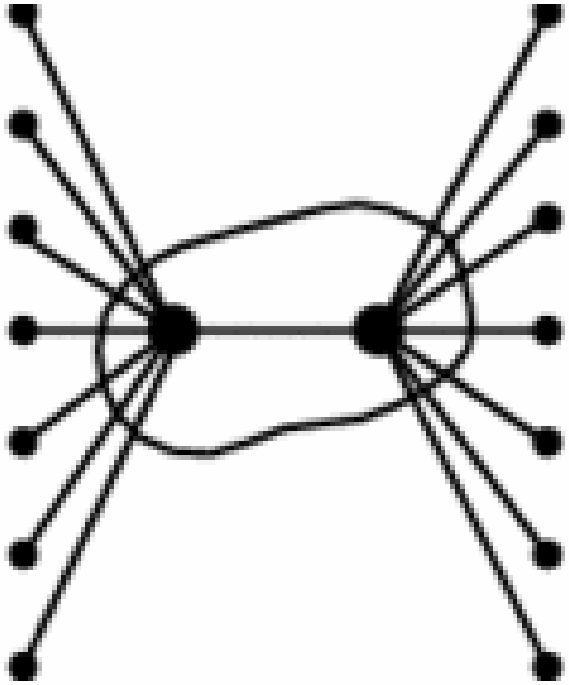
Πώς Αντιμετωπίζουμε την NP-πληρότητα;

Ε: Έστω ότι θέλω να λύσω ένα NP-πλήρες πρόβλημα. Τι πρέπει να κάνω;
Α. Η θεωρία αναφέρει ότι είναι μάλλον απίθανο να βρεις πολυωνυμικό αλγόριθμο.

Θα πρέπει να θυσιάσουμε κάτι από τα εξής:

- Βέλτιστη λύση προβλήματος.
- Επίλυση προβλήματος σε πολυωνυμικό χρόνο.
- Επίλυση αυθαίρετων στιγμιοτύπων του προβλήματος.

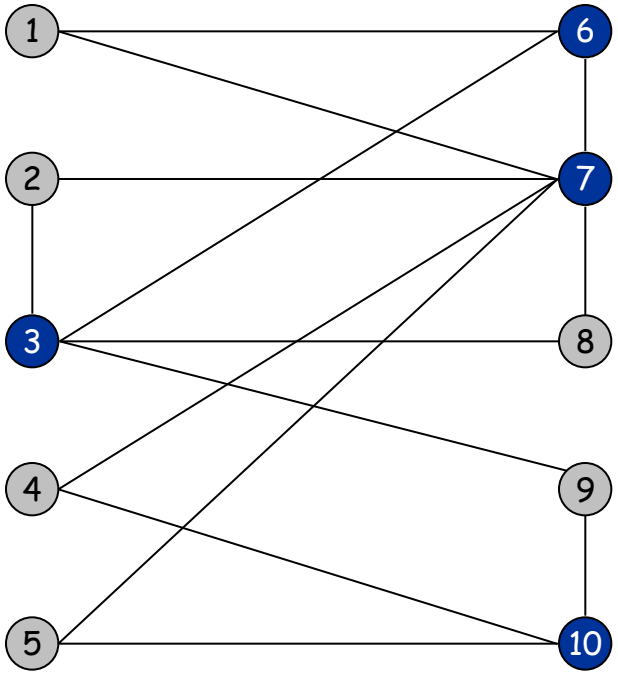
Αυτή (και η επόμενη;;;) διάλεξη: Επίλυση ειδικών περιπτώσεων NP-πλήρων προβλημάτων που εμφανίζονται στην πράξη ☺.



10.1 ΕΎΡΕΣΗ ΜΙΚΡΏΝ ΚΑΛΎΨΕΩΝ ΚΟΡΥΦΏΝ

Κάλυμμα Κορυφών

Κάλυμμα Κορυφών: Δοθέντος ενός γραφήματος $G = (V, E)$ και ενός ακεραίου k, υπάρχει υποσύνολο των κορυφών $S \subseteq V$ έτσι ώστε $|S| \leq k$, και για κάθε ακμή (u, v) θα ισχύει είτε $u \in S$, ή $v \in S$, ή και τα δύο.



$k = 4$

$S = \{ 3, 6, 7, 10 \}$

Εύρεση Μικρών Καλυμμάτων Κορυφών

Ε. Τι γίνεται αν το k είναι μικρό;

Εξαντλητική Αναζήτηση. $O(k\, n^{k+1})$.

- Δοκιμάζουμε τα $C(n, k) = O(n^k)$ υποσύνολα μεγέθους k.
- Απαιτείται $O(k\, n)$ χρόνος για έλεγχο αν ένα υποσύνολο είναι κάλυμμα κορυφών.

Στόχος: Περιορισμός της εκθετικής εξάρτησης στο k, π.χ. σε $O(2^k\, k\, n)$.

π.χ. n = 1,000, k = 10.
Εξαντλητική Αναζήτηση $k\, n^{k+1} = 10^{34} \Rightarrow$ μη διαχειρίσιμο
Καλύτερη Λύση: $2^k\, k\, n = 10^7 \Rightarrow$ διαχειρίσιμο

Σχόλιο: Αν το k είναι σταθερά, ο αλγόριθμος είναι πολυωνυμικός, και αν το k είναι μικρό, τότε είναι και πρακτικός αλγόριθμος.

Εύρεση Μικρών Καλυμμάτων Κορυφών

Ισχυρισμός. Έστω u-v μία ακμή του G. Το G έχει κάλυμμα κορυφών μεγέθους $\leq k$ αν και μόνο αν τουλάχιστον ένα από τα $G - \{u\}$ και $G - \{v\}$ έχει κάλυμμα κορυφών μεγέθους $\leq k-1$.

Διαγραφή v και όλων των προσκείμενων ακμών

Απόδειξη: $\Rightarrow$

- Έστω ότι το G έχει κάλυμμα κορυφών S μεγέθους $\leq k$.
- Το S περιέχει είτε την u ή την v (ή και τις δύο). Έστω ότι περιέχει την u.
- Το $S - \{u\}$ είναι ένα κάλυμμα κορυφών του $G - \{u\}$.

$\Leftarrow$

- Έστω ότι το S είναι κάλυμμα κορυφών του $G - \{u\}$ μεγέθους $\leq k-1$.
- Τότε το $S \cup \{u\}$ είναι ένα κάλυμμα κορυφών για το G. ▪

Ισχυρισμός: Αν το G έχει κάλυμμα κορυφών μεγέθους k, έχει $\leq k(n-1)$ ακμές.
Απόδειξη: Κάθε κορυφή καλύπτει το πολύ $n-1$ ακμές. ▪

Εύρεση Μικρών Καλυμμάτων Κορυφών: Αλγόριθμος

Ισχυρισμός. Ο ακόλουθος αλγόριθμος βρίσκει αν το G έχει κάλυμμα κορυφών μεγέθους $\leq k$ σε $O(2^k kn)$ χρόνο.

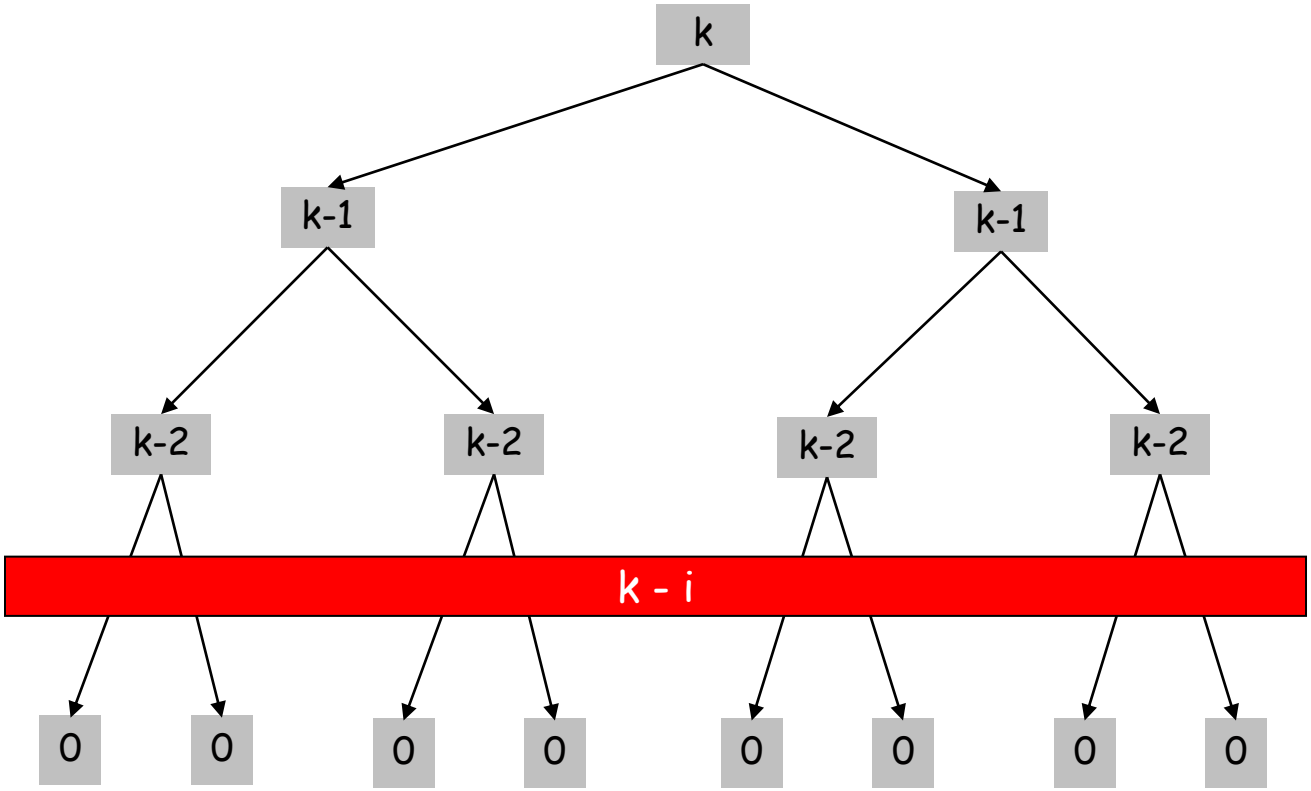
```
boolean Vertex-Cover(G, k) {
   if (G contains no edges)   return true
   if (G contains ≥ kn edges) return false

   let (u, v) be any edge of G
   a = Vertex-Cover(G - {u}, k-1)
   b = Vertex-Cover(G - {v}, k-1)
   return a or b
}
```

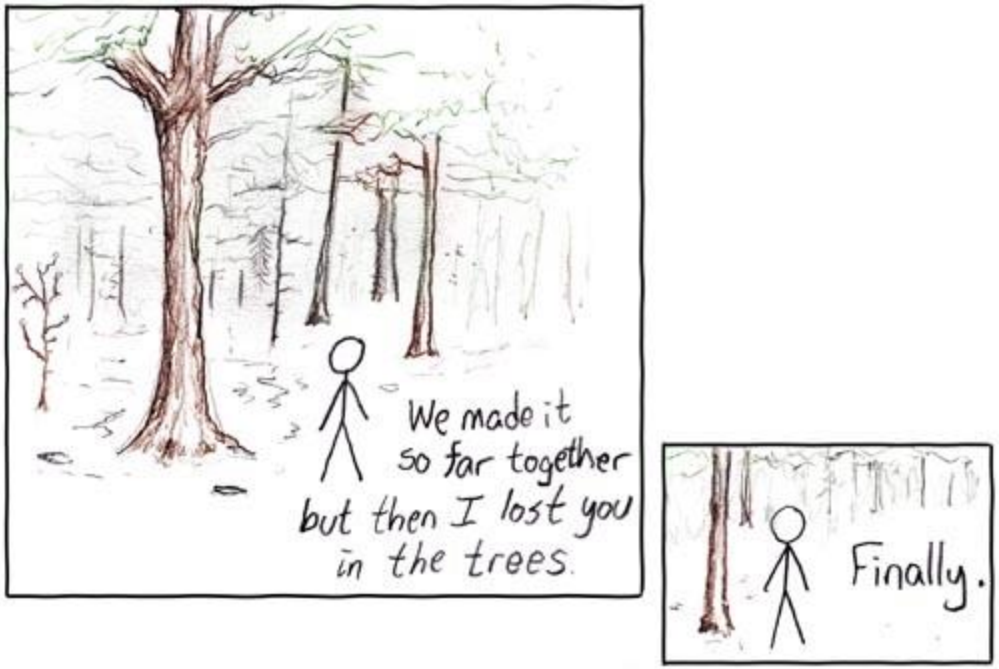
Απόδειξη.

- Η ορθότητα προκύπτει από την προηγούμενη διαφάνεια.
- Υπάρχουν $\leq 2^{k+1}$ κορυφές στο δένδρο αναδρομής και κάθε αναδρομή απαιτεί $O(kn)$ χρόνο. ▪

Εύρεση Μικρών Καλυμμάτων Κορυφών : Δένδρο Αναδρομής

$$T(n,k) \le \begin{cases} c & \text{αν } k = 0 \\ cn & \text{αν } k = 1 \\ 2T(n,k-1) + ckn & \text{αν } k > 1 \end{cases} \Rightarrow T(n,k) \le 2^k ckn$$

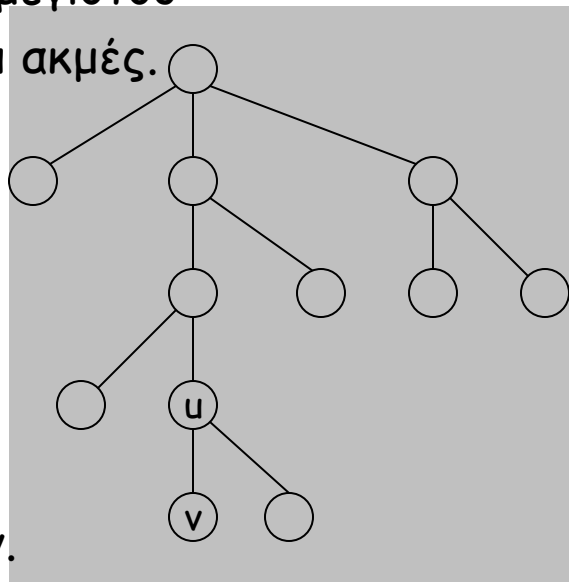




10.2 ΕΠΊΛΥΣΗ NP-ΔΎΣΚΟΛΩΝ ΠΡΟΒΛΗΜΆΤΩΝ ΣΕ ΔΈΝΔΡΑ

Ανεξάρτητο Σύνολο σε Δένδρα

Δοθέντος ενός δένδρου, βρείτε ένα υποσύνολο κόμβων μέγιστου πληθάριθμου έτσι ώστε κανένα ζεύγος να μην μοιράζεται ακμές.



Ένα δένδρο με δύο τουλάχιστον κόμβους
έχει τουλάχιστον δύο φύλλα.

βαθμός= 1

Παρατήρηση. Αν το v είναι φύλλο, υπάρχει ένα μέγιστου μεγέθους ανεξάρτητο σύνολο που περιέχει το v.

Απόδειξη. (επιχείρημα αντικατάστασης)

- Έστω ένα μέγιστου μεγέθους ανεξάρτητο σύνολο S.
- Αν $v \in S$, τότε είμαστε OK.
- Αν $u \notin S$ και $v \notin S$, τότε $S \cup \{ v \}$ είναι ανεξάρτητο $\Rightarrow$ S δεν είναι μέγιστο.
- Αν $u \in S$ και $v \notin S$, τότε $S \cup \{ v \} - \{ u \}$ είναι ανεξάρτητο. ▪

Ανεξάρτητο Σύνολο σε Δένδρα: Άπληστος Αλγόριθμος

Θεώρημα. Ο κάτωθι άπληστος αλγόριθμος βρίσκει ένα μέγιστου μεγέθους ανεξάρτητο σύνολο σε δάση, (και άρα σε δένδρα).

```
Independent-Set-In-A-Forest(F) {
    S ← φ
    while (F has at least one edge) {
        Let e = (u, v) be an edge such that v is a leaf
        Add v to S
        Delete from F nodes u and v, and all edges
            incident to them.
    }
    return S
}
```

Απόδειξη. Η ορθότητα προκύπτει από την προηγούμενη παρατήρηση. ▪

Παρατήρηση. Σε $O(n)$ χρόνο διαπερνώντας τους κόμβους σε μεταδιάταξη.

Ερώτηση: Τι θα συμβεί σε αυθαίρετα γραφήματα με τον παραπάνω αλγόριθμο;

Ανεξάρτητα Σύνολα Μέγιστης Βαρύτητας σε Δένδρα

Δοθέντος ενός δένδρου όπου οι κόμβοι έχουν βάρη $w_v > 0$, βρείτε ένα ανεξάρτητο σύνολο S που μεγιστοποιεί το $\Sigma_{v \in S} w_v$.

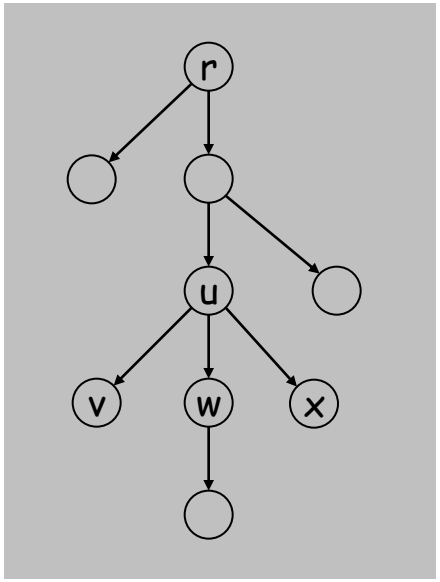
Παρατήρηση. Αν (u, v) είναι μία ακμή έτσι ώστε ο v να είναι φύλλο, τότε είτε ο OPT περιλαμβάνει τον u, ή περιλαμβάνει όλα τα φύλλα που πρόσκεινται στον u.

Δυναμικός Προγραμματισμός: Έστω η ρίζα του δένδρου r.

- OPT_{in} (u) = μέγιστου βάρους ανεξάρτητο σύνολο του υποδένδρου με ρίζα του u, περιέχοντας το u.
- OPT_{out}(u) = μέγιστου βάρους ανεξάρτητο σύνολο του υποδένδρου με ρίζα στο u, χωρίς να περιέχει το u.

$$OPT_{in}(u) = w_u + \sum_{v \in \text{children}(u)} OPT_{out}(v)$$

$$OPT_{out}(u) = \sum_{v \in \text{children}(u)} \max\{OPT_{in}(v),\ OPT_{out}(v)\}$$



children(u) = { v, w, x }

Ανεξάρτητα Σύνολα Μέγιστης Βαρύτητας σε Δένδρα: Δυναμικός Προγραμματισμός

Θεώρημα. Ο αλγόριθμος δυναμικού προγραμματισμού βρίσκει το μέγιστου βάρους ανεξάρτητο σύνολο σε ένα δένδρο σε $O(n)$ χρόνο.

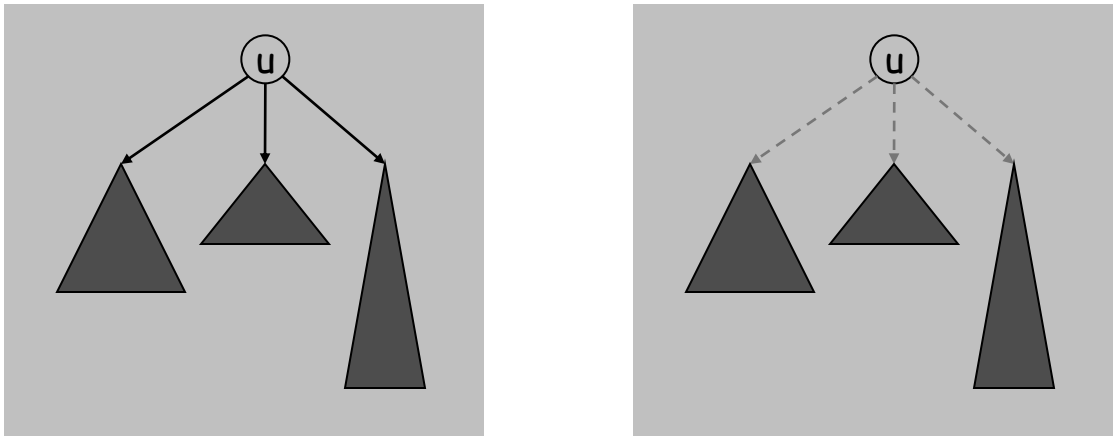
```
Weighted-Independent-Set-In-A-Tree(T) {
    Root the tree at a node r
    foreach (node u of T in postorder) {
        if (u is a leaf) {
            M_in [u] = w_u
            M_out[u] = 0
        }
        else {
            M_in [u] = w_u + Σ_{v∈children(u)} M_out[v]
            M_out[u] = Σ_{v∈children(u)} max(M_in[v], M_out[v])
        }
    }
    return max(M_in[r], M_out[r])
}
```

Εξασφαλίζει ότι επισκεπτόμαστε έναν κόμβο αφού επισκεπτούμε όλα τα παιδιά του.

Απόδειξη. Απαιτείται $O(n)$ χρόνος αφού κάθε κόμβος που επισκεπτόμαστε σε μεταδιάταξη και κάθε ακμή ελέγχεται μία φορά. ▪

Ιδέα

Ανεξάρτητο Σύνολο σε Δένδρα. Αυτή η δομική ιδιαίτερη περίπτωση είναι διαχειρίσιμη αφού μπορούμε να βρούμε έναν κόμβο που αποκόπτει την επικοινωνία μεταξύ υποπροβλημάτων σε διαφορετικά υποδένδρα.



Γραφήματα με φραγμένο δενδρικό πλάτος. (10.4 αλλά δεν θα το κάνουμε)

Γενίκευση δένδρων έτσι ώστε:

- Αντιστοιχεί σε μία πλούσια οικογένεια γραφημάτων που εμφανίζονται στην πράξη.
- Επιτρέπει την διάσπαση σε ανεξάρτητα κομμάτια.

ΤΟ ΠΡΌΒΛΗΜΑ ΤΗΣ ΚΑΤΑΝΟΜΉΣ ΚΑΤΑΧΩΡΗΤΏΝ

Κατανομή Καταχωρητών

Κατανεμητής Καταχωρητών. Μέρος του βελτιστοποιητή σε έναν μεταγλωττιστή που καθορίζει ποιες μεταβλητές αποθηκεύονται σε ποιους καταχωρητές κατά τη διάρκεια εκτέλεσης του προγράμματος.

μεταβλητές

Γράφημα Αλληλεπιδράσεων. Οι κόμβοι αντιστοιχούν σε διαστήματα ύπαρξης. Η ακμή u-v υπάρχει αν σε μία πράξη τόσο ο u όσο και ο v είναι υπαρκτοί.

Παρατήρηση. [Chaitin, 1982] Το πρόβλημα λύνεται αν και μόνο αν το γράφημα αλληλεπιδράσεων είναι k-χρωματίσιμο.

Αν δεν είναι; Αν το γράφημα δεν είναι k-χρωματίσιμο (ή δεν βρίσκουμε έναν k-χρωματισμό), κάποιες μεταβλητές τις αποθηκεύουμε στη κύρια μνήμη και τις επαναφέρουμε όταν χρειάζονται.

Τυπικά είναι μη συχνά χρησιμοποιήσιμες μεταβλητές που δεν είναι στο εσωτερικό επαναλήψεων.

Μία Καλή Ιδιότητα

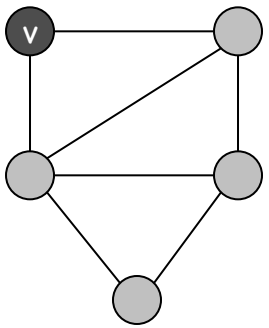
Σχόλιο. Το Πρόβλημα της Κατανομής Καταχωρητών είναι NP-δυσχερές.

Ιδιότητα. Αν ένας κόμβος v στο γράφημα G έχει <k γείτονες, το G είναι k-χρωματίσιμο αν και μόνο αν το $G - \{v\}$ είναι k-χρωματίσιμο.

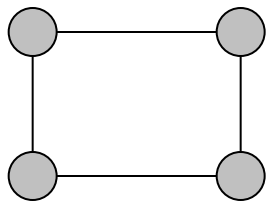
Διαγραφή του v και των προσκείμενων ακμών

Απόδειξη. Διέγραψε τον v από το G και χρωμάτισε το $G - \{v\}$.

- Αν το $G - \{v\}$ δεν είναι k-χρωματίσιμο, τότε δεν είναι και το G.
- Αν το $G - \{v\}$ είναι k-χρωματίσιμο, τότε υπάρχει τουλάχιστον ένα χρώμα για τον v. ▪



k = 3



k = 2

Το G είναι 2-χρωματίσιμο αν και όλοι οι κόμβοι έχουν βαθμό 2

Ο Αλγόριθμος του Chaitin

```
Vertex-Color(G, k) {
    while (G is not empty) {
        Pick a node v with fewer than k neighbors
        Push v on stack
        Delete v and all its incident edges
    }
    while (stack is not empty) {
        Pop next node v from the stack
        Assign v a color different from its neighboring
        nodes which have already been colored
    }
}
```

π.χ. τον κόμβο με τους λιγότερους γείτονες

Ο Αλγόριθμος του Chaitin

Θεώρημα. [Kempe 1879, Chaitin 1982] Ο αλγόριθμος του Chaitin παράγει έναν k-χρωματισμό για κάθε γράφημα με μέγιστο βαθμό k-1.
Απόδειξη. Από το γεγονός ότι κάθε κόμβος έχει <k γείτονες.

Ο αλγόριθμος επιτυγχάνει χρωματισμό ακόμα και για γραφήματα με μέγιστο βαθμό μεγαλύτερο του k.

Στην Πράξη. Ο αλγόριθμος του Chaitin είναι εξαιρετικά αποδοτικος και χρησιμοποιείται ευρέως σε μεταγλωττιστές για την κατανομή΄σε καταχωρητές.