Λύσεις Προβλημάτων

- Τα προβλήματα της κλάσεως **NPC** αντιμετωπίζονται με
  - **Προσεγγιστικούς αλγορίθμους**
    - προσπαθούν να βρουν την χρυσή τομή μεταξύ χρόνου και ποιότητας λύσεως
  - **Ευρετικές τεχνικές**
    - προσπαθούν να βρουν ακριβή λύση με συστηματική αναζήτηση στον χώρο των λύσεων

Προβλήματα Βελτιστοποιήσεως

- **Όριο αναλογίας μεγιστοποίησης/ελαχιστοποίησης**
  - ο λόγος C^*/C της βέλτιστης προς την προσεγγιστική λύση
  - Εάν ψάχνουμε για ελάχιστο, χρησιμοποιούμε τον λόγο C/C^*
- **ε-προσεγγιστικός αλγόριθμος**

$$|C^* - C|/C^* \leq \varepsilon(n)$$

Κατηγορίες **NPC** βάσει ε

- **Πλήρως Προσεγγίσιμα**
  - διαθέτουν ε-προσεγγιστικό αλγόριθμο για οποιοδήποτε ε (σπάνια κατηγορία)
- **Μερικώς Προσεγγίσιμα**
  - διαθέτουν ε-προσεγγιστικό αλγόριθμο για *ορισμένες τιμές του* ε (εφ' όσον **P≠NPC**)
- **Μη Προσεγγίσιμα**
  - δεν διαθέτουν ε-προσεγγιστικό αλγόριθμο για *κανένα* ε (εκτός κι αν **P=NPC**)

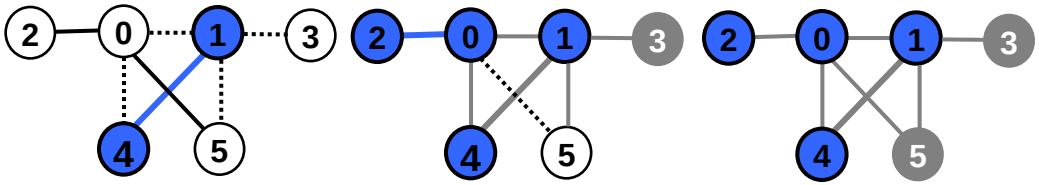
Κατηγορίες βάσει χρόνου

- **Πολυωνυμικού Χρόνου**
  - Εάν, για *καθορισμένο ε,* τρέχουν σε πολυωνυμικό χρόνο
- **Πλήρως Πολυωνυμικού Χρόνου**
  - εάν ο χρόνος τους είναι πολυωνυμικός ως προς τα n, $1/\varepsilon$

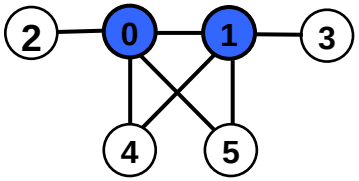
Κάλυμμα Κορυφής

- Πολύ απλός αλγόριθμος
  - Διαλέγουμε μία ακμή *e*
  - Τοποθετούμε τα άκρα της *e* στο κάλυμμα
  - Διαγράφουμε τις προσπίπτουσες στην *e* ακμές
  - Συνεχίζουμε μέχρι να διαγραφούν όλες οι ακμές

Προσεγγιστική Λύση



Βέλτιστη Λύση



Ανάλυση Πολυπλοκότητας

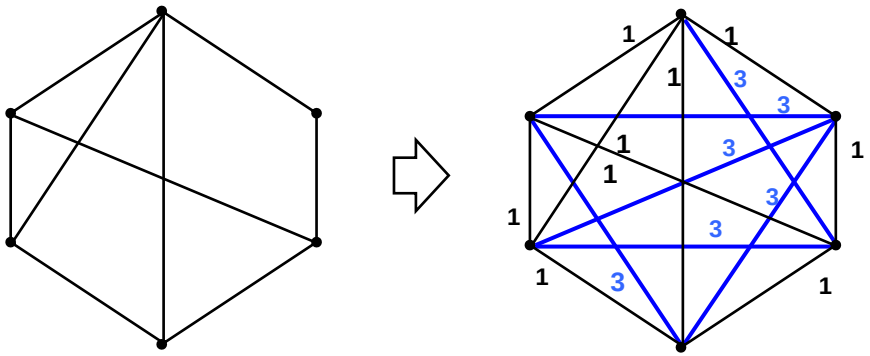
- Είναι 1-προσεγγιστικός:
  - Μέγεθος προσεγγιστικής λύσης $K = 2A$, όπου A το # ακμών που επιλέχθηκαν
  - $A \leq K^*$ (πρέπει να καλύπτει όλες τις ακμές, ενώ οι ακμές που επιλέχθηκαν δεν έχουν κοινές κορυφές) $\Rightarrow K/2 \leq K^* \Rightarrow K \leq 2K^*$

Περιοδεύων Πωλητής (TSP)

- Μη προσεγγίσιμο, γιατί διαφορετικά **P = NPC**

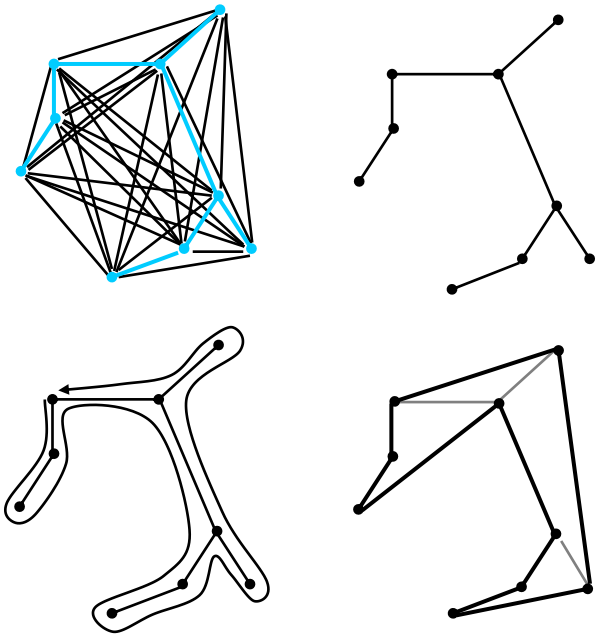
  Αναγωγή του ΚύκλοςHamilton σε TSP

  - Βάρη στο πλήρες: 1 ή 2+$V\varepsilon$ εάν ανήκει ή όχι στο αρχικό
  - Εάν υπάρχει κύκλος, τότε έχουμε κόστος ακριβώς V
  - Διαφορετικά, έχουμε κόστος ≥ V-1+2+$V\varepsilon$ =1+V(1+ε).
  - Οπότε, συγκρίνοντας το κόστος με το 1+V(1+ε), καταλαβαίνουμε εάν υπάρχει ή όχι κύκλος στο αρχικό γράφημα



Προσέγγιση TSP για Ευκλείδειες Αποστάσεις

- Υπόθεση: το γράφημα είναι πλήρες
- Σχηματισμός ΕΕΔ T
- Διαπέραση Euler, αποφεύγοντας τις κορυφές που έχουμε ήδη επισκεφτεί



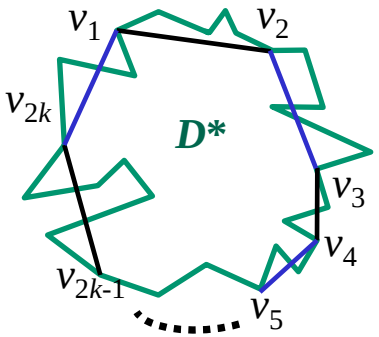
Ανάλυση

- Χρονική πολυπλοκότητα
  - πολυωνυμική
- Ποιότητα Αλγορίθμου
  - 1-προσεγγιστικός:
    - εάν από τον βέλτιστο γύρο D^* αφαιρεθεί μία ακμή, προκύπτει ένα Επικαλύπτον Δένδρο
    - Άρα, κόστος(T) ≤ κόστος (D^*)
    - Όμως,

$$\text{κόστος}(D) \leq 2\text{κόστος}(T) \leq 2\text{κόστος}(D^*)$$

Καλύτερη Προσέγγιση

- Σχηματισμός ΕΕΔ T
- Ελαφρύτερο ταίριασμα M των περιττού βαθμού κορυφών του T
- Εύρεση κύκλου Euler στο $T \cup M$ (γιατί;)
  - $c(T) \leq c(D^*)$ και
  - $c(D) \leq c(M)+c(T) \leq c(D^*)+0.5\, c(D^*)$, αφού:



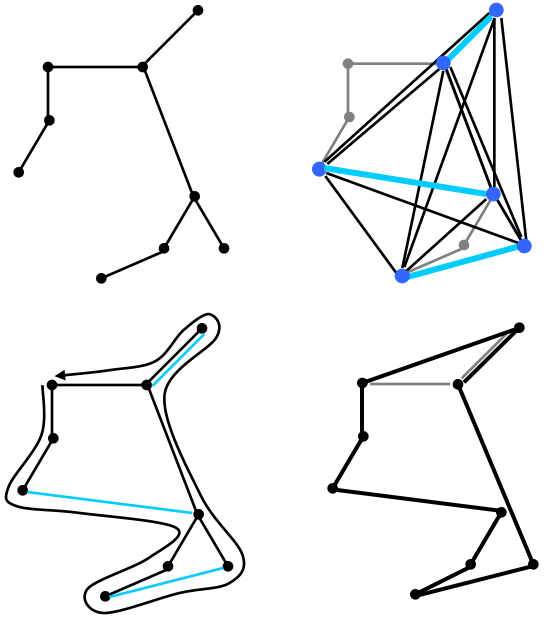
$D^*=(p_1 v_1 p_2 v_2 \ldots p_{2k} v_{2k}) \Rightarrow$

$M_1=(\{v_1,v_2\}\ldots\{v_{2k-1},v_{2k}\})$ και

$M_2=(\{v_2,v_3\}\{v_4,v_5\}\ldots\{v_{2k},v_1\})$

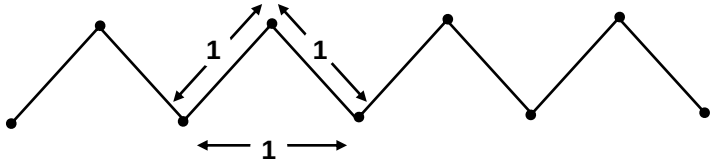
ταιριάσματα $\Rightarrow$

$2c(M) \leq c(M_1) + c(M_2) \leq c(D^*)$

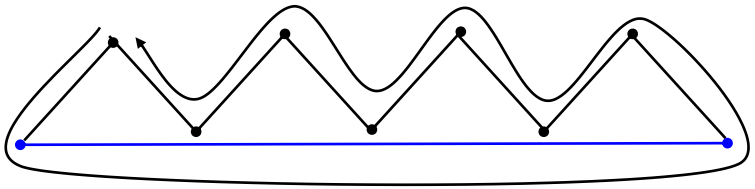


Παρατήρηση

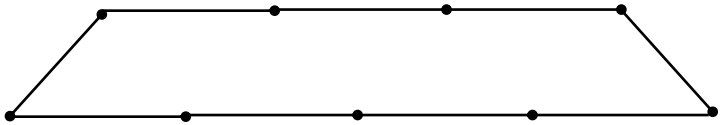
- Το όριο είναι «σφικτό»



Προσεγγιστική



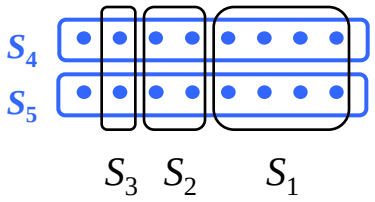
Βέλτιστη



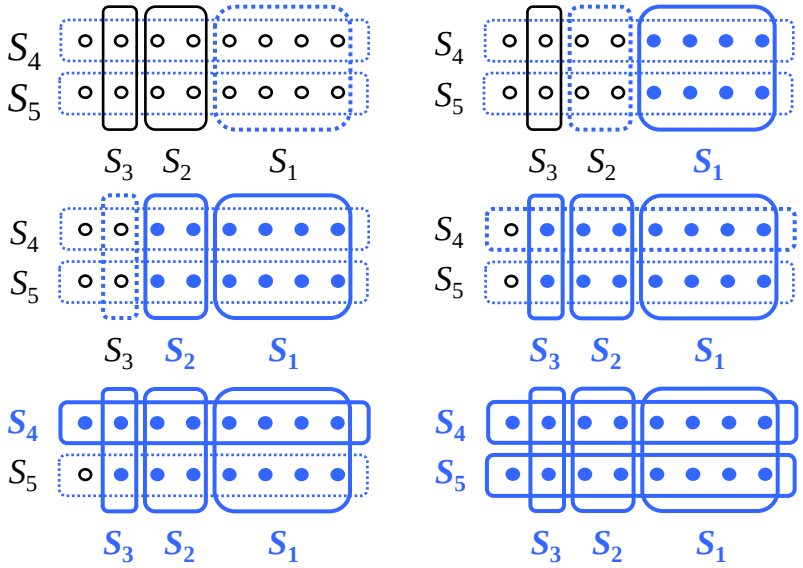
Κάλυμμα Συνόλου

- Όσο υπάρχουν ακάλυπτα σύνολα, διαλέγουμε αυτό με το μέγιστο πλήθος ακάλυπτων μελών

Βέλτιστη λύση



Προσεγγιστική



Ανάλυση

- Με χρήση επιμερισμένης αναλύσεως (amortization):
  - Κάθε φορά που διαλέγουμε ένα σύνολο S_i με x ακάλυπτα στοιχεία, πληρώνουμε μία μονάδα
  - Έτσι, κάθε στοιχείο s χρεώνεται $d(s) = 1/x$ μονάδες Άρα,

$$|C| = \Sigma_s d(s)$$

  - Έστω $S_j^* = \{s_{j_1},\ldots,s_{j_k}\}$ διάταξη των στοιχείων του S_j^* βάσει προσεγγιστικής καλύψεως. Τότε:
    - $d(S_j^*) \leq \Sigma_l 1/(|S_j^*|-l+1)=O(\log|S_j^*|) \Rightarrow \Sigma_{S_j^*} d(S_j^*) \leq |C^*|\log \max(|S_j^*|)$
    - $|C| = \Sigma_s d(s) \leq \Sigma_{S_j^*} d(S_j^*) \leq |C^*|\log |S_j^*| \leq |C^*|\log n$

Σακκίδιο0/1

- Βάσει της τεχνικής της στρογγυλοποιήσεως (rounding):
  - Στρογγυλοποιούμε προς τα κάτω τις αξίες των αντικειμένων προς το κοντινότερο πολλαπλάσιο του $p=\varepsilon v_{max}/2n$ (ε η προσέγγιση που θέλουμε, v_{max} η μέγιστη αξία)
  - Η μέγιστη ωφέλεια του μετασχηματισμένου είναι n $2n/\varepsilon$ $p=2n^2p/\varepsilon$

Σακκίδιο0/1 (συν.)

- Άρα, έχουμε την σχέση

$$W_{k,i}=\min\{W_{k-1,i},\ w_k+W_{k-1,i-vk/p}\}$$

- $v_{\max} \le V^* \le nv_{\max}$
- $V^*-\varepsilon v_{\max}/2 \le V^*-\varepsilon V^*/2 \le V' \Rightarrow$

$$V^* \le (1-\varepsilon/2)^{-1} V' \le (1+\varepsilon) V'$$

Άθροισμα Υποσυνόλου

- Δοθέντων ενός συνόλου S ακεραίων και ενός ακεραίου t, αιτείται η εύρεση $S' \subseteq S$, με το μέγιστο άθροισμα μελών που δεν ξεπερνά τον t
- Είναι **NPC,** καθώς σε αυτό ανάγεται το κάλυμμα κορυφής
- Τεχνική του διαμερισμού διαστήματος
- Πλήρης λύση:
  - Να σχηματίζουμε τα αθροίσματα όλων των υποσυνόλων και να αφαιρούμε όσα ξεπερνούν το όριο.
  - Π.χ., για S={108,101,208,200} και t=411 (L_i = αθροίσματα υποσυνόλων με τα i πρώτα στοιχεία που δεν ξεπερνούν το όριο)

    L_1={0,108},

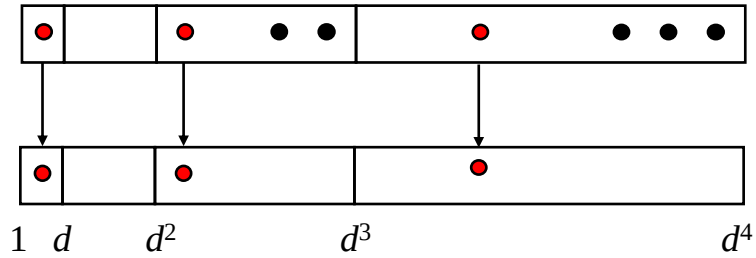
    L_2={0,101,108,209},

    L_3={0,101,108,208,209,309,316,~~417~~},

    L_4={0,101,108,200,208,209,301,308,309,316,408,409,~~509,516~~}
- Χρόνος $\Omega(2^n)$

Προσέγγιση

- Έστω $\delta=\varepsilon/n$ και $d=(1-\delta)^{-1}$
- Το διάστημα $[1,t]$ κατατμείται, σε υποδιαστήματα, ως ακολούθως:



- Κρατάμε μία τιμή ανά διάστημα, την μικρότερη σε αυτό:

  $(y-z)/z \le (d^i-d^{i-1})/d^{i-1} = 1-1/d = \delta,\ z \le y,\ \forall y, z \in [d^{i-1},d^i]$
- Το πλήθος k των διαστημάτων είναι $O(n\log t/\varepsilon)$

Προσέγγιση (συν.)

- Και πάλι σαρώνουμε την λίστα, *αλλά κόβουμε τις τιμές που είναι μικρότερες κατά* $(1-\delta)^{-1}$ *φορές από την τελευταία αποδεκτή*
- Π.χ., για $\varepsilon = 0.1$ και $\delta = 0.025$

  L_1={0,108},

  L_2={0,101,108,209},

  L_3={0,101,108,208,~~209~~,309,~~316~~,~~417~~},

  L_4={0,101,108,200,208,301,~~308~~,~~309~~, **408**,~~509~~}
- Χρόνος $O(nk)$
  - n περάσματα, και σε κάθε πέρασμα $k = n\log t/\varepsilon$ διαστήματα σαρώνονται

Ανάλυση

- Για κάθε $y \in L_i^*$ υπάρχει αντιπρόσωπός του $z \in L_i$ με σχετικό σφάλμα:

$$(1-\varepsilon/n)^i y \leq z \leq y$$

  - Με επαγωγή στον αριθμό i του βήματος της επανάληψης:

$$(1-\varepsilon/n)^{i-1} y \leq z \leq y \Rightarrow (1-\varepsilon/n)^{i-1}(y+x_i) \leq z+x_i \leq y+x_i \ (i\text{-στό βήμα})$$

z', z'' αντιπρόσωποι των z και $z+x_i$

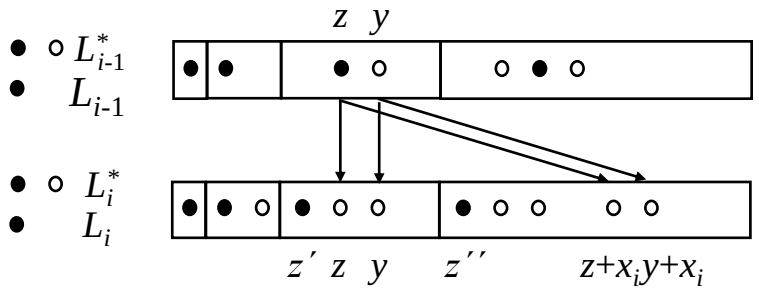
$$(1-\varepsilon/n)z \leq z' \leq z \text{ και } (1-\varepsilon/n)(z+x_i) \leq z'' \leq z+x_i$$

$$(1-\varepsilon/n)^i y \leq z' \leq z \leq y$$

$$(1-\varepsilon/n)^i (y+x_i) \leq z'' \leq z+x_i \leq y+x_i$$

  - Οπότε, για $i=n$, παίρνουμε την προσεγγιστική λύση T^*

$$(1-\varepsilon)T^* \leq (1-\varepsilon/n)^n T^* \leq T \leq T^*$$



Πλήρωση Δοχείου

- Να τοποθετηθούν αντικείμενα στον ελάχιστο αριθμό δοχείων, ατομικής χωρητικότητας 1
- Είναι **NPC** διότι σε αυτό ανάγεται το σακκίδιο0/1

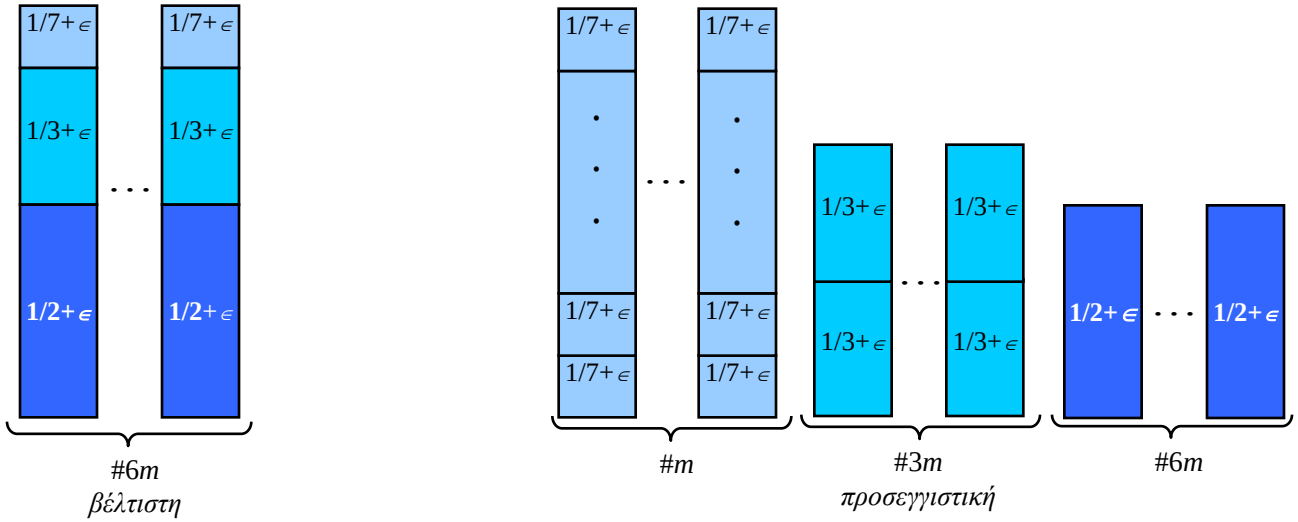
Πρώτο Ταίριασμα (First Fit)

- Κάθε αντικείμενο τοποθετείται στο πρώτο δοχείο που το χωρά, διαφορετικά σε νέο άδειο δοχείο
- Κάθε δοχείο είναι τουλάχιστον κατά το ήμισυ γεμάτο. Άρα

$$FF \leq 2\Sigma s_i \leq 2B$$

(θεωρήστε πως κάθε αντικείμενο έχει μέγεθος 1/2+ε)

Χειρότερη Περίπτωση FF



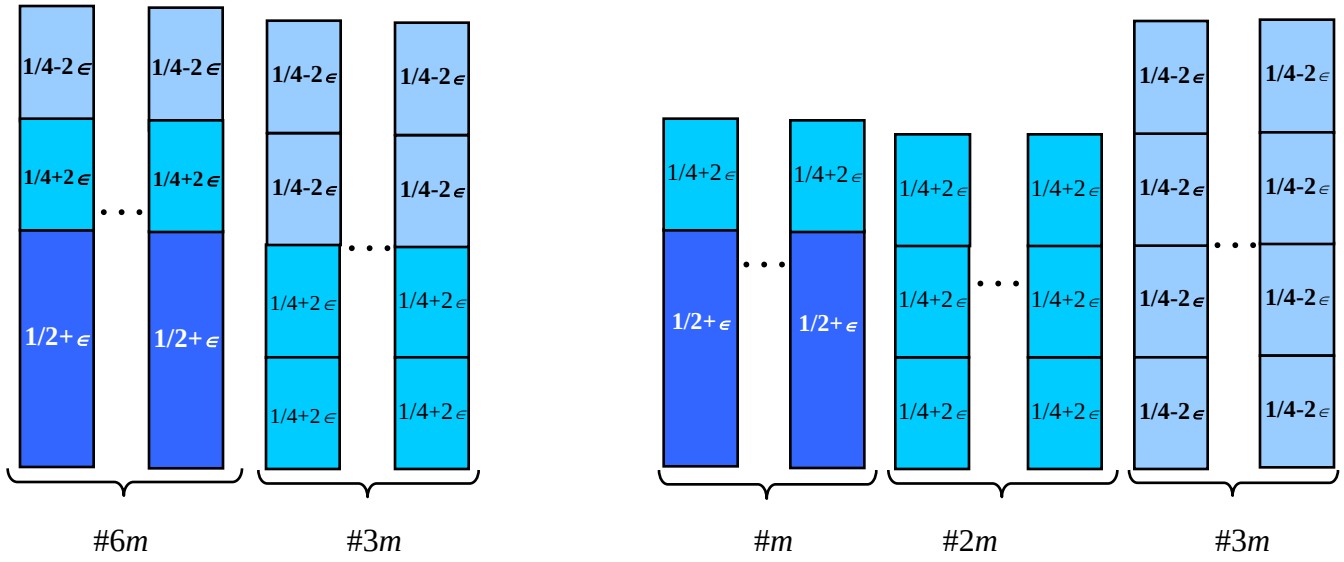
Καλύτερο Ταίριασμα (Best Fit)

- Τοποθέτηση στο πιο γεμάτο δοχείο που το χωρά. Διαφορετικά, σε νέο άδειο
- Και πάλι δεν ξεπερνά το διπλάσιο -στην πράξη, όμως, δουλεύει καλύτερα

Φθίνοντος Πρώτου Ταιριάσματος (First Fit decreasing)

- Καλύτερη Λύση:
  - Τα αντικείμενα διατάσσονται κατά φθίνουσα τάξη μεγέθους
  - Κατόπιν, εκτελείται πρώτο ταίριασμα
- FFD $\leq 11/9B+4$ (δύσκολη απόδειξη!)

Χειρότερη Περίπτωση



βέλτιστη

προσεγγιστική

Ευκολότερη Απόδειξη: 4/3+1/(3B)

- Ο προσεγγιστικός τοποθετεί στα (επιπλέον) FFD-B δοχεία αντικείμενα μεγέθους ≤ 1/3

  Έστω s_i το πρώτο αντικείμενο του (B+1) δοχείου με μέγεθος >1/3. Τότε, ≤ 2 αντικείμενα ανά δοχείο

  $b_1 \ldots b_k$ μονά δοχεία

  $b_{k+1} \ldots b_B$ διπλά δοχεία

  Η ίδια διάρθρωση και για τον βέλτιστο, άρα το s_i δεν χωρά (άτοπον)

Ευκολότερη Απόδειξη: 4/3+1/(3B) (συν.)

- Τα αντικείμενα που τοποθετούνται στα επιπλέον (*FFD*-*B*) δοχεία είναι το πολύ B-1

  Έστω B τα αντικείμενα s_1, …, s_B και στο πρώτο δοχείο έχουν μπει μεγέθους s_{a_1} αντικείμενα, ..., στο B-στό μεγέθους s_{a_B} αντικείμενα. Τότε,

$$s_{a_1} + s_{i_1} > 1 \text{ ,…, } s_{a_B} + s_{i_B} > 1$$

$$B \geq \sum s_i \geq \sum s_{a_i} + \sum s_i > \sum (s_{a_i} + s_i) > B \text{ (άτοπο)}$$

Ευκολότερη Απόδειξη: $4/3+1/(3B)$ (συν.)

- Άρα, έχουμε:

$$FFD \leq B + (B\text{-}1)/3$$

$$FFD/B \leq [B+(B\text{-}1)/3]/B \leq 4/3+1/(3B)$$

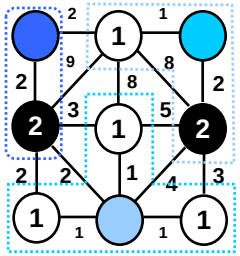
k-κέντρο

- Δοθέντος ενός γραφήματος G, να βρεθούν k κορυφές-κέντρα, έτσι ώστε η μέγιστη απόσταση κάθε κορυφής από τα κέντρα να ελαχιστοποιείται
- Ορισμοί
  - Γειτονιά ενός κέντρου
    - οι κορυφές που την έχουν ως κοντινότερο κέντρο
  - Απόσταση κέντρου
    - η μέγιστη απόσταση από τις κορυφές της γειτονιάς του
  - Απόσταση μποτιλιαρίσματος (bottleneck)
    - η μέγιστη απόσταση κέντρου

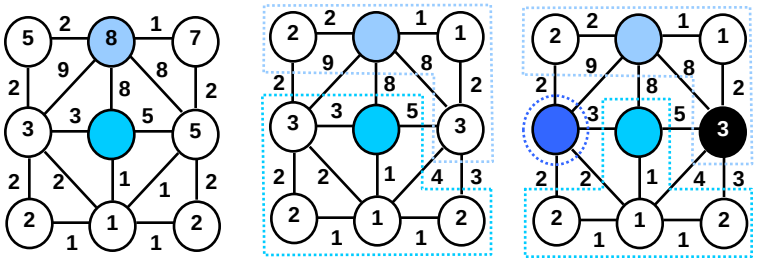
1-προσεγγιστικός Αλγόριθμος

- Διαδοχικά
  - υπολογίζουμε γειτονιές
  - στο υπό διαμόρφωση σύνολο τοποθετούμε την κορυφή μποτιλιαρίσματος

Βέλτιστη:



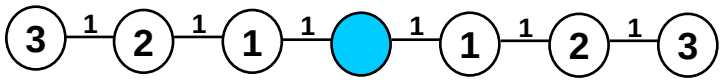
Προσεγγιστική:



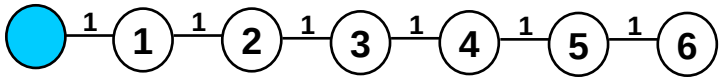
Χειρότερη Περίπτωση

- 1-κέντρο

Βέλτιστη



Προσεγγιστική



Ανάλυση

- Έστω:

  $C^o=\{c_1^o,\ldots, c_k^o\}$ η βέλτιστη, με απόσταση μποτιλιαρίσματος $D(C^o)$

  $C^h=\{c_1^h,\ldots, c_k^h\}$ η προσεγγιστική, με απόσταση μποτιλιαρίσματος $D(C^h)$

  c_b^h κορυφή μποτιλιαρίσματος
- $\text{dis}(c_i^h,c_j^h) \geq D(C^h)$ (φθίνουν οι αποστάσεις)
- $|C^h \cup \{c_b^h\}| = k+1 \Rightarrow$

  Υπάρχουν c_i^h,c_j^h στην ίδια γειτονιά c_μ^o
- Άρα,

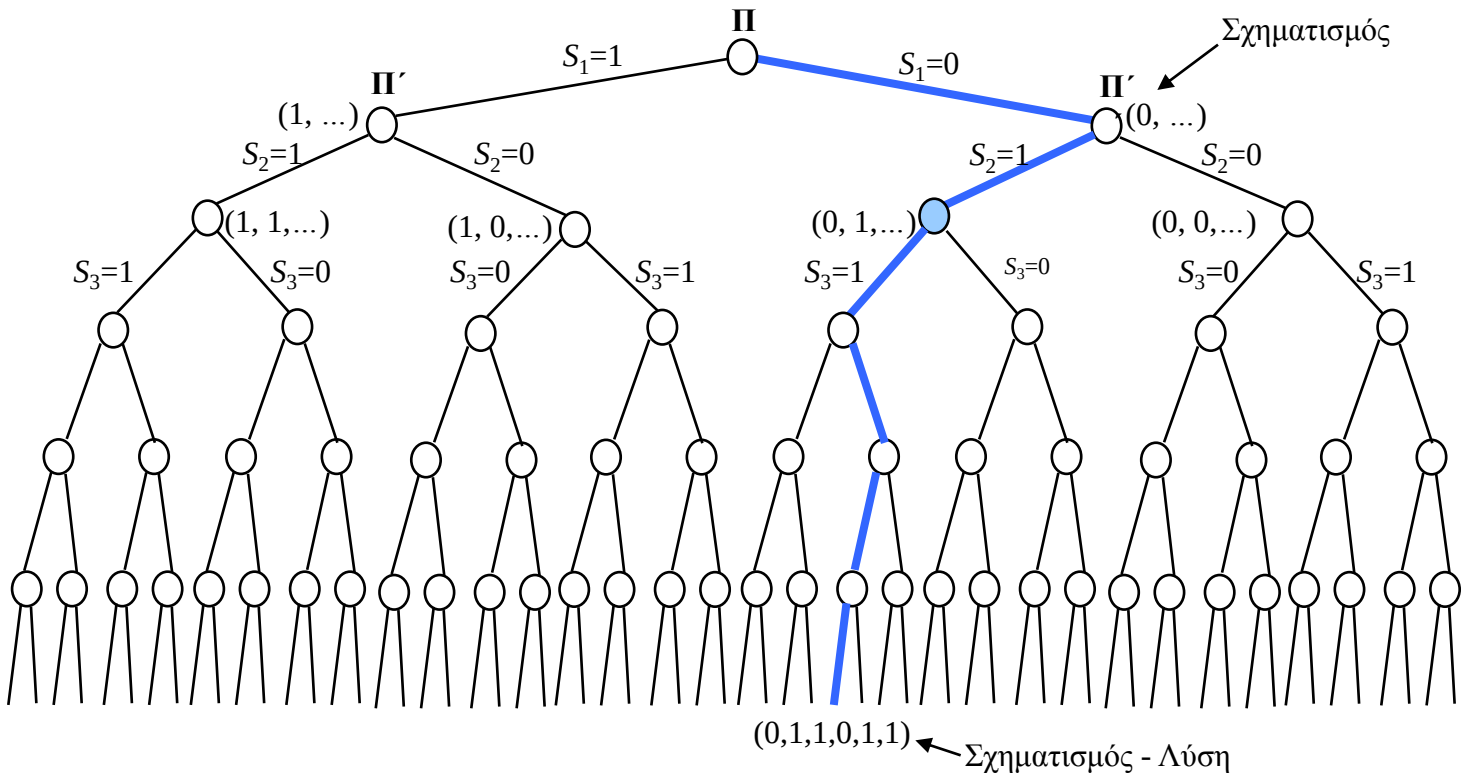
$$D(C^h) \leq \text{dis}(c_i^h,c_j^h) \leq \text{dis}(c_i^h,c_\mu^o) + \text{dis}(c_j^h,c_\mu^o) \leq 2\, D(C^o)$$

Απεικονίσεις Προβλημάτων **NPC**

- Χώρος ενδεχόμενων λύσεων εκθετικός
- Οι ληφθείσες αποφάσεις αναπαρίστανται μέσω ***σχηματισμών*** (***configurations***)
- Κάθε *μν*-επιλογή οδηγεί σε διαφορετικό σχηματισμό (configuration) και, ισοδύναμα, σε υποσύνολο του αρχικού συνόλου ενδεχόμενων λύσεων

Παράδειγμα

- Άθροισμα Υποσυνόλου



Αιτούμενο

- Εξερεύνηση του χώρου των ενδεχόμενων λύσεων με ***έξυπνο*** και ***συστηματικό*** τρόπο, ώστε να εντοπιστεί η λύση με καλή πολυπλοκότητα χρόνου
- Τεχνικές
  - Οπισθοδρόμηση (Backtrack)
    - για «απλά» προβλήματα μοναδικής λύσεως
  - Διακλάδωση και Οριοθέτηση (Branch & Bound)
    - για προβλήματα βελτιστοποιήσεως

Οπισθοδρόμηση

Algorithm backtrack(configuration c)

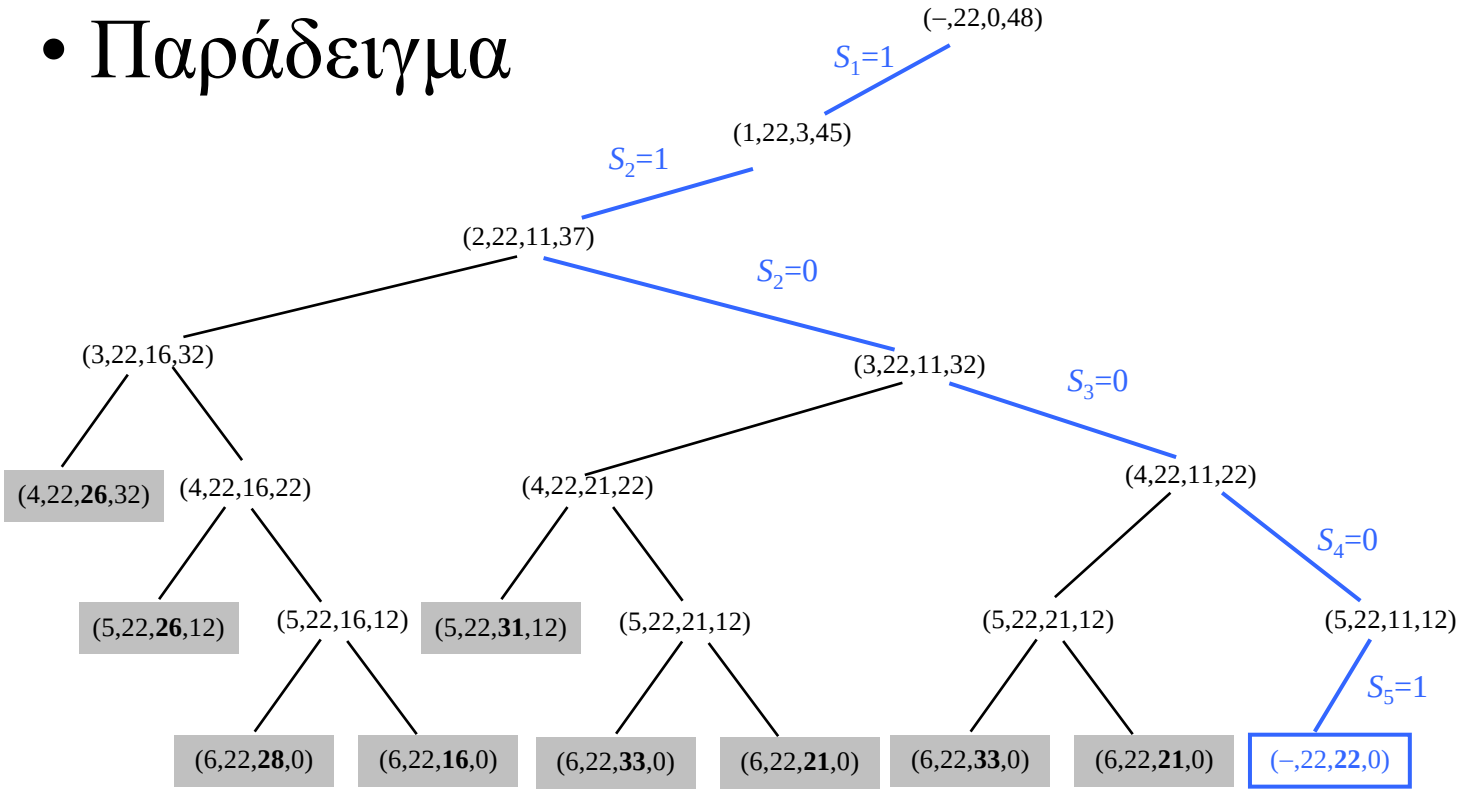
```
1.  queue q = new queue(c);
2.  list l;
3.  while (!q.isEmpty()){
4.    s = q.get();  // επόμενος σχηματισμός
5.    l = split(s)  // διακλάδωση στα επιμέρους υποπροβλήματα
6.    for (x = l.getNextIterator(); x != null; x = x.getNext()}
7.      switch (test(x)) {
8.         case IS_SOLUTION          :  return x; break;
9.         case EXISTS_SOLUTION  :  q.ins(x);   break; // τοποθέτηση για εξέταση
10.       case EMPTY                    :  break;
11.     }
12. }
13. return NO_SOLUTION_EXISTS;
```

Άθροισμα Υποσυνόλου

- Σχηματισμός
  - Λογικό διάνυσμα διάνυσμα $(x_1,x_2,\ldots,x_i,x_{i+1},\ldots,x_n)$
- Διέλευση
  - DFS
- Έλεγχοι
  - $\Sigma_{j\le i+1} x_j = t$: επιτυχίας
  - $\Sigma_{j\le i+1} x_j > t$: αποτυχίας εάν συμπεριληφθεί το x_{i+1}
  - $\Sigma_{j\le i+1} x_j + \Sigma_{j>i+1} x_j < t$: το υπολειπόμενο υποπρόβλημα είναι κενό

Άθροισμα Υποσυνόλου (συν.)

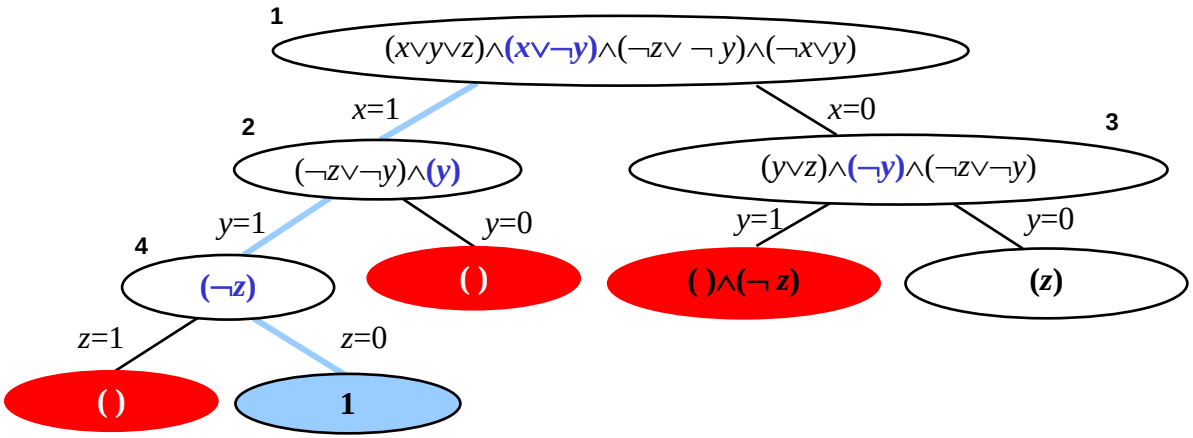
- Παράδειγμα



SAT

- Διέλευση
  - επιλογή υποπροβλήματος με μικρότερη υποπρόταση
- Δημιουργία υποπροβλημάτων
  - επιλογή μιας μεταβλητής της μικρότερης υποπροτάσεως και διάσπαση, αναθέτοντας 0 και 1
- Έλεγχος
  - Κενές υποπροτάσεις: αναληθή ανάθεση
  - Αντιφάσεις
  - Εμφάνιση της τιμής αληθείας ‘1’

Παράδειγμα



Διακλάδωση & Οριοθέτηση

- Εφαρμόζεται στα προβλήματα βελτιστοποίησης
- Όταν βρεθεί μία λύση, δεν τερματίζουμε την αναζήτηση, αλλά συνεχίζουμε γιατί θέλουμε την βέλτιστη
- Επιπροσθέτως, χρησιμοποιούνται συναρτήσεις εκτιμήσεως της ωφέλειας που θα προκύψει από συνέχιση της θεωρήσεως

Διακλάδωση & Οριοθέτηση (συν.)

Algorithm branchAndBound(configuration c)

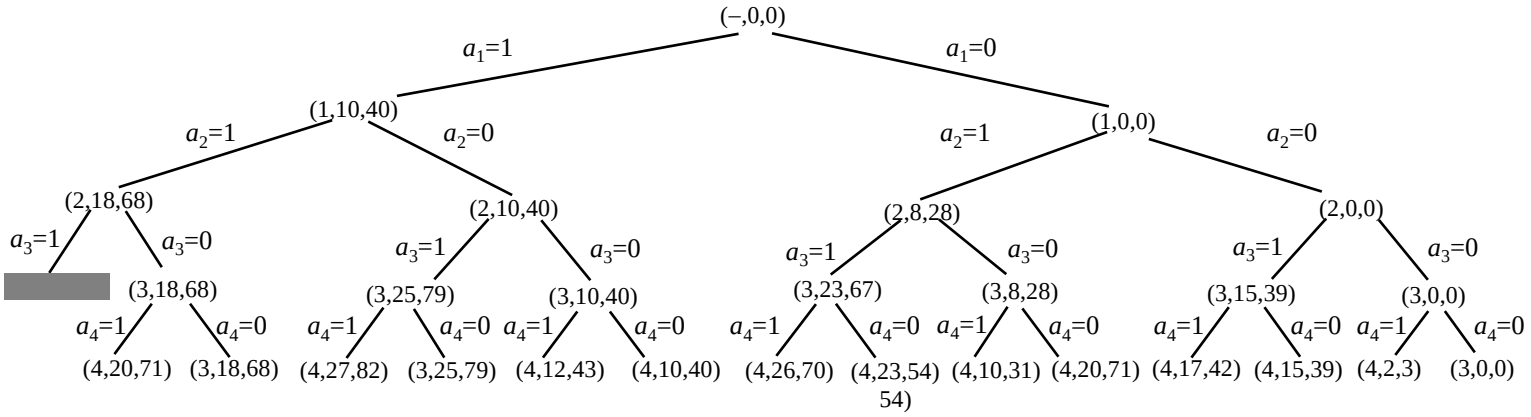
```
1.  queue q = new queue(c);
2.  list l;
3.  curBest =  -INFTY;
3.  while (!q.isEmpty()){
4.    s = q.get();  // επιλογή σχηματισμού
5.    l = split(s)  // λίστα διακλαδώσεως
6.    for (x = l.First(); x != null; x = x.getNext()}
7.      switch (test(x)) {
8.         case IS_SOLUTION          :  curBest = (cost(x) > curBest) ? cost(x) : curBest); break;
9.         case EXISTS_SOLUTION :  if (potentialGain(x) > curBest) q.ins(x); break;
10.        case EMPTY                    :  break;
11.     }
12. }
13. return (curBest == -INFTY ? NO_SOLUTION_EXISTS : curBest);
```

Σακκίδιο 0/1

- Σχηματισμός
  - άνυσμα αντικειμένων $(a_1, a_2, \ldots, a_i, a_{i+1}, \ldots, a_n)$
- Συνάρτηση εκτίμησης ωφέλειας
  - κλασματικό σακκίδιο στο εναπομείναν πρόβλημα

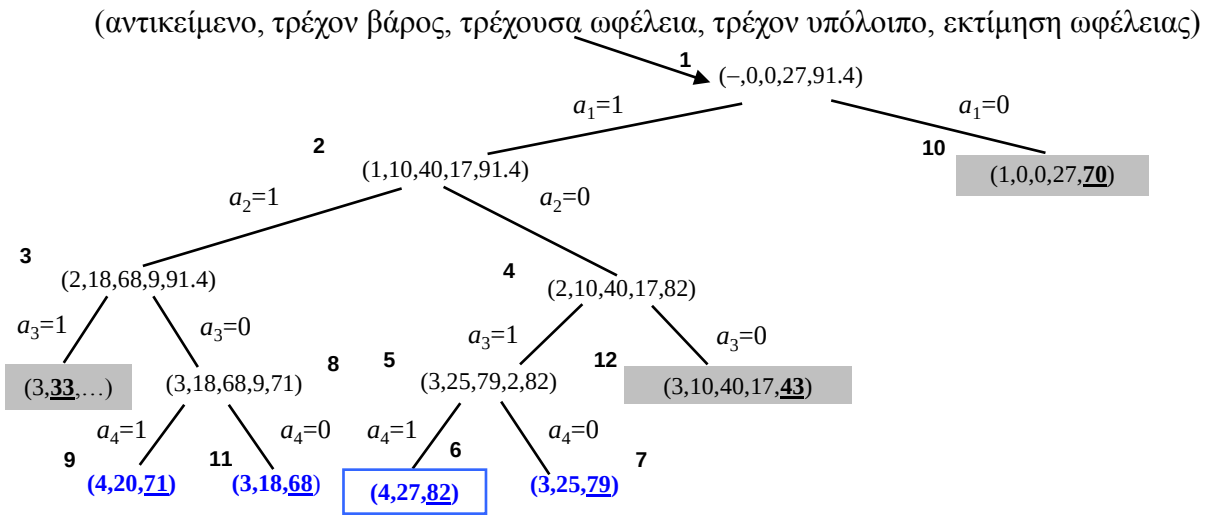
Παράδειγμα

- Απλή Οπισθοδρόμηση



Παράδειγμα (συν.)

- Διακλάδωση και Οριοθέτηση



- Τα αντικείμενα είναι διατεταγμένα κατά φθίνοντα λόγο ωφέλειας
- Έχει αναφερθεί μέχρι και 95% επιτάχυνση