

Άπληστοι Αλγόριθμοι

Δημήτρης Φωτάκης

Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο



Άπληστοι Αλγόριθμοι

- ... για προβλήματα **βελτιστοποίησης**:
 - Λειτουργούν σε **βήματα**.
 - Κάθε βήμα κάνει μια **αμετάκλητη** επιλογή για λύση.
 - **Άπληστη** επιλογή: αυτό που φαίνεται καλύτερο με βάση **τρέχουσα κατάσταση** και κάποιο (απλό) **κριτήριο**.
 - Ίδια στρατηγική στο υποπρόβλημα που προκύπτει.
- Πλεονεκτήματα:
 - Γρήγοροι, απλοί, και **«φυσιολογικοί»** αλγόριθμοι.
 - Εφαρμόζεται (επιτυχώς) σε πολλά και σημαντικά προβλήματα.
- Μειονεκτήματα:
 - Βέλτιστη λύση μόνο **υπό προϋποθέσεις!**
- Βέλτιστη λύση: **απόδειξη ορθότητας** (συν. επαγωγή).

Άπληστη Στρατηγική

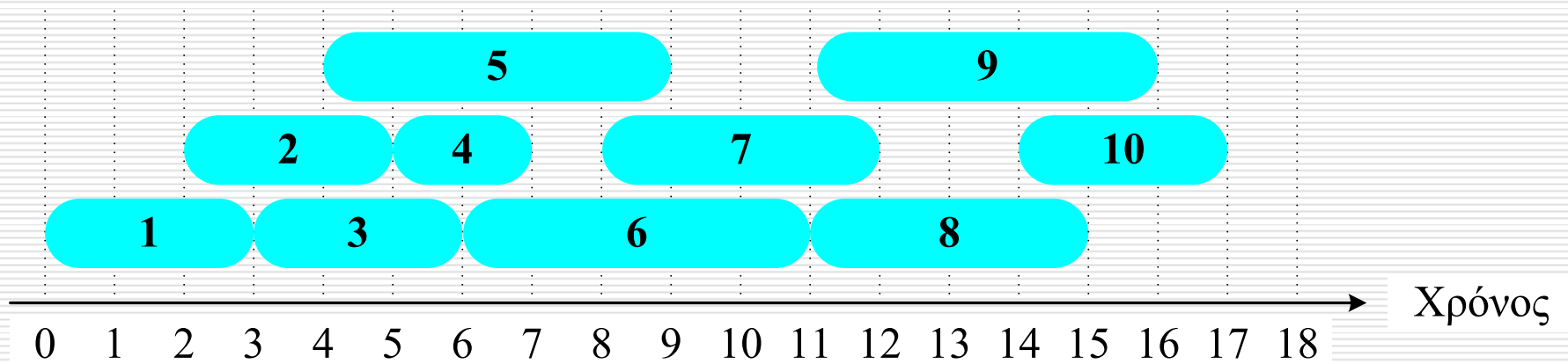
- Ταξινόμηση **συνιστωσών** με βάση κάποιο απλό κριτήριο.
- (Αμετάκλητη) επιλογή καθορίζει αν «**καλύτερη**» συνιστώσα θα συμπεριληφθεί στη λύση.
 - Επιλογή με κάποιον **απλό κανόνα**.
- **Ίδια στρατηγική** σε υποπρόβλημα που προκύπτει.
 - **Μη-προσαρμοστικός**: ίδια ταξινόμηση σε όλα τα βήματα.
 - **Προσαρμοστικός**: αλλάζει ταξινόμηση σε κάθε βήμα.
- **Χρόνος εκτέλεσης** συνήθως καθορίζεται από επιλογή «καλύτερης» συνιστώσας σε κάθε βήμα.

Επιλογή Δραστηριοτήτων

- n δραστηριότητες: αρχή και τέλος $[s_i, f_i) : f_i > s_i \geq 0$ (π.χ. μαθήματα, υπολογιστικές διεργασίες).
- Επιλογή δραστηριοτήτων χωρίς χρονικές επικαλύψεις και δρομολόγηση σε κοινό πόρο (π.χ. αίθουσα διδασκαλίας, επεξεργαστής).
- Ζητούμενο: δρομολόγηση **μέγιστου** #δραστηριοτήτων.
- Πρόβλημα συνδυαστικής βελτιστοποίησης:
 - Κάθε δρομολόγηση χωρίς επικαλύψεις: εφικτή λύση.
 - Ζητούμενο: εφικτή δρομολόγηση με **μέγιστο** #δραστηριοτήτων.

Παράδειγμα

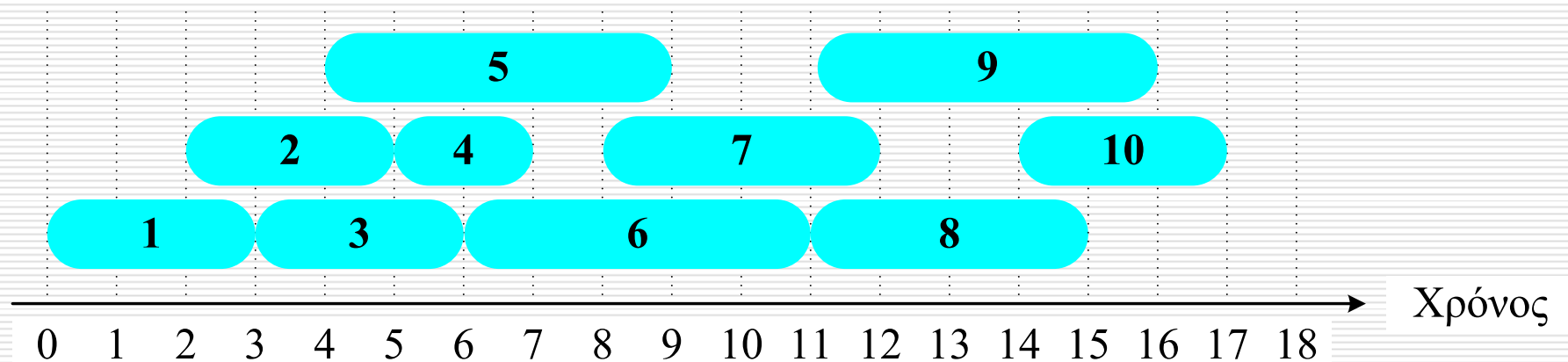
i	1	2	3	4	5	6	7	8	9	10
s_i	0	2	3	5	4	6	8	11	11	14
f_i	3	5	6	7	9	11	12	15	16	17



- Βέλτιστη λύση: **4 δραστηριότητες.**
Π.χ. {1, 3, 6, 8}, {2, 4, 7, 10}, {1, 4, 7, 10}, ...

Άπληστος Αλγόριθμος

- Κριτήριο άπληστης επιλογής;
 - Ελάχιστος **χρόνος ολοκλήρωσης**.
- Ταξινόμηση σε **αύξουσα** σειρά χρόνου **ολοκλήρωσης**.
Επόμενη δραστηριότητα:
 - **Δρομολογείται** αν είναι **εφικτό** (πόρος είναι ελεύθερος).
 - **Αγνοείται** αν δρομολόγηση δεν είναι εφικτή.



Υλοποίηση

```
greedySelection((s1, f1), ..., (sn, fn))
```

```
/* f1 ≤ f2 ≤ ... ≤ fn */
```

```
C ← {1}; j ← 1;
```

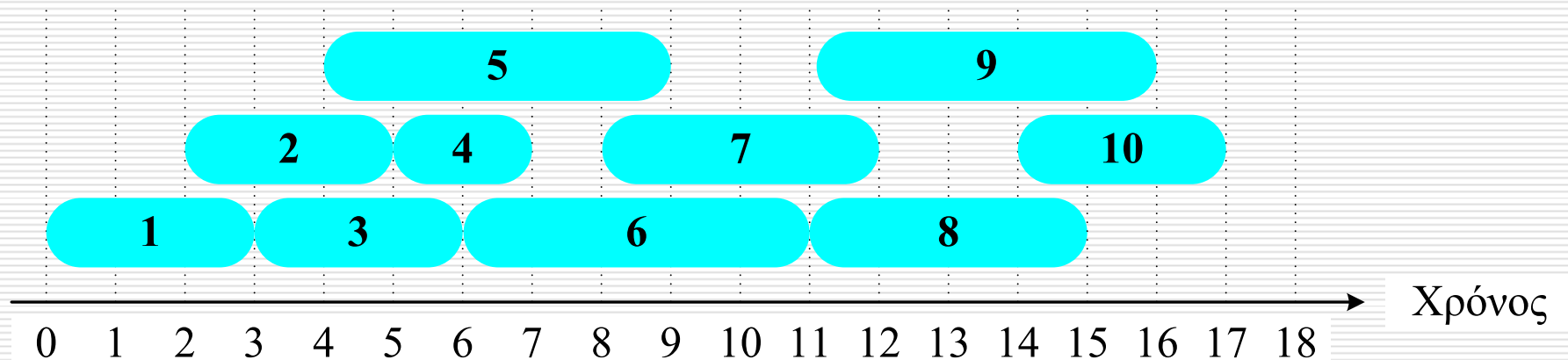
```
for i ← 2 to n do
```

```
    if si ≥ fj then
```

```
        C ← C ∪ {i}; j ← i;
```

```
return(C);
```

Χρόνος **O(n log n)**
(ταξινόμηση ως προς
χρόνο ολοκλήρωσης).



Υπολογισμός Βέλτιστης Λύσης

- Βέλτιστη λύση: **απόδειξη ορθότητας** (επαγωγή).
- Βασίζεται σε **δύο ιδιότητες** (απαραίτητες!):
 - **Αρχή βελτιστότητας** (βέλτιστες επιμέρους λύσεις):
 - Κάθε τμήμα βέλτιστης λύσης αποτελεί βέλτιστη λύση για αντίστοιχο υποπρόβλημα.
 - π.χ. κάθε τμήμα μιας συντομότερης διαδρομής είναι συντομότερη διαδρομή μεταξύ των άκρων του.
 - Χαρακτηριστικό και **δυναμικού προγραμματισμού**.
 - Ιδιότητα **άπληστης επιλογής**:
 - Υπάρχει βέλτιστη λύση που **συμφωνεί** με την άπληστη επιλογή που κάνει ο αλγόριθμος.
 - ... ή ισοδύναμα: η άπληστη επιλογή **μπορεί** να οδηγήσει σε βέλτιστη λύση.

Ορθότητα

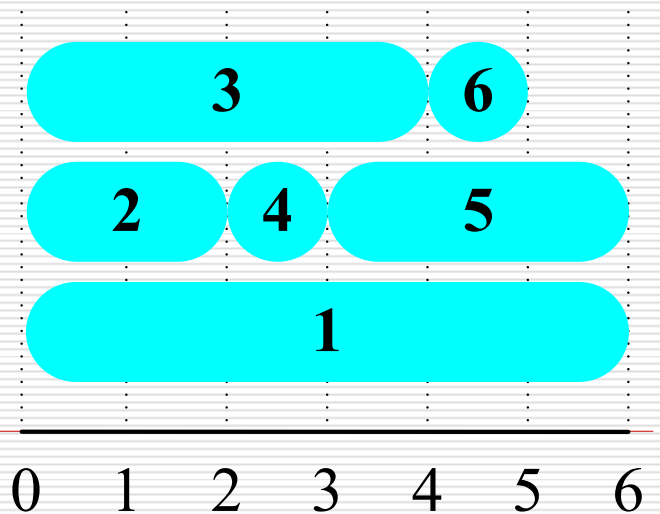
- **Επαγωγή** στον #δραστηριοτήτων.
Υποθέτουμε πάντα ότι $f_1 \leq f_2 \leq \dots \leq f_n$
- **Βάση:** αν 1 δραστ., αυτή επιλέγεται πάντα.
- Έστω αλγ. υπολογίζει **βέλτιστη λύση** για $\leq n - 1$ δραστ.
Θδο. υπολογίζει βέλτιστη λύση για **σύνολο** A με n δραστ.
 - Αλγ. επιλέγει 1 (f_1) και βέλτιστη λύση $A_1 = \{i : s_i \geq f_1\}$
 - $C^*(A)$ βέλτιστη λύση και j δραστ. $C^*(A)$ ολοκληρώνεται πρώτη.
 - $|C^*(A)| \leq 1 + |C^*(A_1)| = \#$ δραστηριοτήτων άπληστου αλγ.
 - Άπληστη επιλογή: $f_j \geq f_1 \Rightarrow (C^*(A) \setminus \{j\}) \cup \{1\}$ **βέλτιστη**.
- Άπληστος αλγόριθμος υπολογίζει **βέλτιστη λύση** για A .

Ορθότητα

- Αποδείξαμε ότι $|C^*(A)| = 1 + |C^*(A_1)|$
- Ιδιότητα **άπληστης επιλογής**:
 - (Άπληστη) επιλογή δραστηριότητας με ελάχιστο χρόνο ολοκλήρωσης οδηγεί σε συνολικά βέλτιστη λύση.
- Ιδιότητα **βέλτιστων επιμέρους λύσεων**:
 - Βέλτιστη λύση περιέχει βέλτιστη λύση για υποπρόβλημα A_1 (δραστ. που δεν επικαλύπτονται με πρώτη).

Χρωματισμός Διαστημάτων

- n διαστήματα: αρχή και τέλος $[s_i, f_i) : f_i > s_i \geq 0$
- Χρωματισμός όλων ώστε επικαλυπτόμενα διαστήματα να έχουν διαφορετικό χρώμα.
- Ζητούμενο: χρωματισμός με **ελάχιστο** #χρωμάτων.
- Άπληστος αλγόριθμος:
 - Ταξινόμηση με χρόνο έναρξης.
 - Κάθε διάστημα που αρχίζει παίρνει πρώτο διαθέσιμο χρώμα.
 - Κάθε διάστημα που τελειώνει «απελευθερώνει» το χρώμα του.
- Χρήση χρώματος $d \geq 2$ μόνο αν επικάλυψη d διαστημάτων.



Δρομολόγηση Εργασιών

- **Ένας** εξυπηρετητής (π.χ. επεξεργαστής, εκτυπωτής, ταμίας).
- Σύνολο N με n εργασίες: χρόνο εκτέλεσης $t_i > 0$ (π.χ. υπολογιστικές διεργασίες, εκτυπώσεις, συναλλαγές).
- Δρομολόγηση για **ελαχιστοποίηση συνολικού** (ισοδύναμα, μέσου) **χρόνου εξυπηρέτησης**.
 - $t_1 = 8, t_2 = 7, t_3 = 2, t_4 = 5$.
 - $1, 2, 3, 4: 8 + 15 + 17 + 22 = \mathbf{62}$.
 - $3, 4, 2, 1: 2 + 7 + 14 + 22 = \mathbf{45}$.
 - Δρομολόγηση: μετάθεση $\pi : \{1, \dots, n\} \mapsto \{1, \dots, n\}$
 - Χρόνος εξυπηρέτησης $i : c_i(\pi) = \sum_{j:\pi(j) \leq \pi(i)} t_j$
 - Συνολικός χρόνος εξυπηρέτησης: $T(\pi) = \sum_{i=1}^n c_i(\pi)$

Άπληστος Αλγόριθμος

- Δρομολόγηση σε **αύξουσα** σειρά **χρόνου** εκτέλεσης:

$$t_1 \leq t_2 \leq \dots \leq t_n$$

- Συνολικός χρόνος εξυπηρέτησης:

$$T = t_1 + (t_1 + t_2) + (t_1 + t_2 + t_3) + \dots + (t_1 + \dots + t_n)$$

$$= n t_1 + (n - 1)t_2 + (n - 2)t_3 + \dots + t_n$$

$$= \sum_{i=1}^n (n - i + 1)t_i$$

- Βέλτιστος γιατί **όσο μεγαλύτερος** χρόνος εκτέλεσης, **τόσο λιγότερες φορές** συνεισφέρει στο συνολικό χρόνο εξυπηρέτησης.

Ορθότητα: Επιχείρημα Ανταλλαγής

- Έστω π^* βέλτιστη δρομολόγηση, $T(\pi^*)$ συνολικός χρόνος.
 $\lambda^*(j)$: σειρά εργασίας j στη βέλτιστη δρομολόγηση.
- Έστω π^* διαφορετική από άπληστη:
 - k πρώτη που δρομολογείται αργότερα στην π^* : $\lambda^*(k) > k$
 - j αυτή που δρομολογείται k -οστή στην π^* : $\lambda^*(j) = k$
 - ... συμφωνούν σε $k - 1$ αρχικές: $k < j$ και $t_k \leq t_j$
 - t_k και t_j στο $T(\pi^*)$: $(n - k + 1)t_j + \dots + (n - \lambda^*(k) + 1)t_k$
 - Ανταλλαγή k και j : $(n - k + 1)t_k + \dots + (n - \lambda^*(k) + 1)t_j$
(k πηγαίνει στη θέση που έχει στην άπληστη δρομολόγηση).
 - Διαφορά: $(\lambda^*(k) - k)(t_k - t_j) \leq 0$
- Έτσι π^* γίνεται ίδια με άπληστη **χωρίς αύξηση** χρόνου.
 - **Άπληστη** δρομολόγηση είναι **βέλτιστη**.

Ιδιότητες

- Ιδιότητα **άπληστης επιλογής**:
 - Για κάθε k , βέλτιστη δρομολόγηση π^* συμφωνεί με άπληστη στη σειρά των k πρώτων εργασιών.
 - (Άπληστη) επιλογή **συντομότερης διαθέσιμης** \rightarrow βέλτιστη.
- Ιδιότητα **βέλτιστων επιμέρους λύσεων**:
 - $T = n t_1 + (n - 1)t_2 + (n - 2)t_3 + \dots + t_n$
 - Αν αγνοήσουμε t_1 , π^* παραμένει βέλτιστη για υπόλοιπες.
- Απόδειξη **ορθότητας**: (επαγωγική) εφαρμογή ιδιότητας άπληστης επιλογής.

Πρόβλημα του Περιπτερά

- Κέρματα αξίας 1, 5, και 20 λεπτών.
- Ρέστα ποσό x με **ελάχιστο** #κερμάτων.
- Αλγόριθμος:
 - Όσο περισσότερα 20λεπτα: $c_{20}(x) = \lfloor x/20 \rfloor$
 - Όσο περισσότερα 5λεπτα: $c_5(x) = \lfloor (x - 20 c_{20}(x))/5 \rfloor$
 - Υπόλοιπα 1λεπτα: $c_1(x) = x - 20 c_{20}(x) - 5 c_5(x)$
- **Βέλτιστη** λύση χρησιμοποιεί **ίδιο** #κερμάτων:
 - 20λεπτα: Δεν μπορεί περισσότερα. Βελτιώνεται αν λιγότερα.
 - Αν ίδιο #20λέπτων, τότε ίδιο #5λέπτων.
 - ... **επαγωγή** στα πλήθος διαφορετικών κερμάτων.
- Δουλεύει αλγόριθμος αν κέρματα 1, 12, και 20 λεπτών;
 - Π.χ. ρέστα 24 λεπτά.

Κλασματικό Πρόβλημα Σακιδίου

- Δίνονται n είδη και ένα **σακίδιο** μεγέθους B .
Είδος i διαθέσιμο σε ποσότητα s_i με αξία p_i : (s_i, p_i)
- Είδος i μπορεί να συμπεριληφθεί στο σακίδιο σε **οποιοδήποτε ποσοστό**.
- Ζητείται συλλογή **μέγιστης αξίας** που **χωράει** στο σακίδιο.

$$\begin{aligned} \max & \sum_{i=1}^n f_i p_i \\ \text{ώστε} & \sum_{i=1}^n f_i s_i \leq B \\ & f_i \in [0, 1] \quad \forall i \in [n] \end{aligned}$$

- Είδη: $\{ (3, 5), (2, 7), (4, 4), (6, 8), (5, 4) \}$
Μέγεθος σακιδίου: **10**.
- Βέλτιστη λύση = $\{ 1 \times (3, 5), 1 \times (2, 7), (5/6) \times (6, 8) \}$
Βέλτιστη αξία = $5 + 7 + (5/6) \times 8 = 18.3333$

Άπληστος Αλγόριθμος

- Είδη $N = \{1, \dots, n\}$, σακίδιο μεγέθους B .
Βέλτιστη λύση $F^* = (f_1^*, f_2^*, \dots, f_n^*)$
- **Βέλτιστες Επιμέρους Λύσεις.** Αγνοούμε είδος i :
 - $F_{-i}^* = (f_1^*, \dots, f_{i-1}^*, f_{i+1}^*, \dots, f_n^*)$ **βέλτιστη λύση**
για $N \setminus \{i\}$ με σακίδιο $B - f_i^* s_i$
- Είδος i : $r_i = p_i / s_i$ (αξία / μονάδα μεγέθους)
- Είδη σε φθίνουσα σειρά r_i : $r_1 \geq r_2 \geq \dots \geq r_n$
 - Όσο περισσότερο από i χωράει στο (διαθέσιμο) σακίδιο.
$$f_i = \begin{cases} 1 & \text{αν } B_{i-1} \geq s_i \\ B_{i-1} / s_i & \text{αν } B_{i-1} < s_i \end{cases}$$
 - Αναπροσαρμογή διαθέσιμου σακιδίου και επόμενο είδος.
$$B_i = B_{i-1} - f_i s_i$$

Υλοποίηση

`greedyKnapsack($B, (s_1, p_1), \dots, (s_n, p_n)$)`

for $i \leftarrow 1$ **to** n **do**

$r_i \leftarrow p_i/s_i; f_i \leftarrow 0;$

Ταξινόμηση $r_1 \geq r_2 \geq \dots \geq r_n$

for $i \leftarrow 1$ **to** n **do**

if $B \leq 0$ **then** $f_i \leftarrow 0;$

else if $B \geq s_i$ **then**

$f_i \leftarrow 1; B \leftarrow B - s_i;$

else

$f_i \leftarrow B/s_i; B \leftarrow 0;$

Χρόνος **$O(n \log n)$**
(ταξινόμηση ως προς
λόγο αξίας / μέγεθος).

Άπληστη Επιλογή

- Έστω βέλτιστη λύση $F^* = (f_1^*, f_2^*, \dots, f_n^*)$
Έστω άπληστη λύση $F = (f_1, f_2, \dots, f_n)$
- Ιδιότητα **άπληστης επιλογής**:
 - Υπάρχει βέλτιστη λύση: $f_1^* = f_1$
 - Απληστία: καμία λύση με περισσότερο από είδος 1.
 - Αν βέλτιστη $f_1^* < f_1$, αντικαθιστούμε $(f_1 - f_1^*)s_1$ μονάδες άλλου είδους (ή κενού) με είδος 1:
 - Αποδεκτή λύση γιατί $B \geq f_1 s_1$
 - Αξία **δεν** μειώνεται.
- Απόδειξη ορθότητας με επαγωγική εφαρμογή ιδιότητας **άπληστης επιλογής**.

Ορθότητα

- Επαγωγή στον #ειδών.
 - Βάση: 1 είδος. Άπληστη επιλογή: $f_1^* = f_1$
 - Επαγωγική υπόθεση: $\text{ειδών} \leq n - 1$, άπληστη = βέλτιστη.
 - Θεωρούμε n είδη.
 - Άπληστη επιλογή: $f_1^* = f_1$
 - Στιγμιότυπο με $n - 1$ είδη και σακίδιο $B - f_1 s_1$
 - Επαγωγική υπόθεση: $F_{-1} = (f_2, \dots, f_n)$ **βέλτιστη** λύση.
 - Συνολικά για n είδη:

$$f_1 p_1 + \sum_{i=2}^n f_i p_i \geq f_1^* p_1 + \sum_{i=2}^n f_i^* p_i$$

- **Άπληστος** αλγόριθμος υπολογίζει **βέλτιστη λύση**.

Άπληστη Στρατηγική

- Ταξινόμηση **συνιστωσών** με βάση κάποιο κριτήριο (π.χ. **σακίδιο**: είδη σε **φθίνουσα σειρά αξία / μέγεθος**).
- (Αμετάκλητη) επιλογή καθορίζει **αν «καλύτερη»** (βλ. «επόμενη») συνιστώσα θα συμπεριληφθεί στη λύση.
- **Ίδια στρατηγική** σε υπο-πρόβλημα που προκύπτει.
 - **Μη-προσαρμοστικός**: ίδια ταξινόμηση σε όλα τα βήματα.
 - **Προσαρμοστικός**: αλλάζει ταξινόμηση σε κάθε βήμα.
- **Χρόνος εκτέλεσης** καθορίζεται από χρόνο ταξινόμησης.
- Βέλτιστη λύση: **απόδειξη ορθότητας** (συνήθ. επαγωγή).
 - Ιδιότητα **άπληστης επιλογής**.
 - Αρχή **βελτιστότητας** (βέλτιστες επιμέρους λύσεις).



Άπληστοι αλγόριθμοι (Greedy Algorithms)

Ένας άπληστος αλγόριθμος πάντα κάνει την επιλογή που φαίνεται καλύτερη τη δεδομένη στιγμή



Άπληστοι αλγόριθμοι

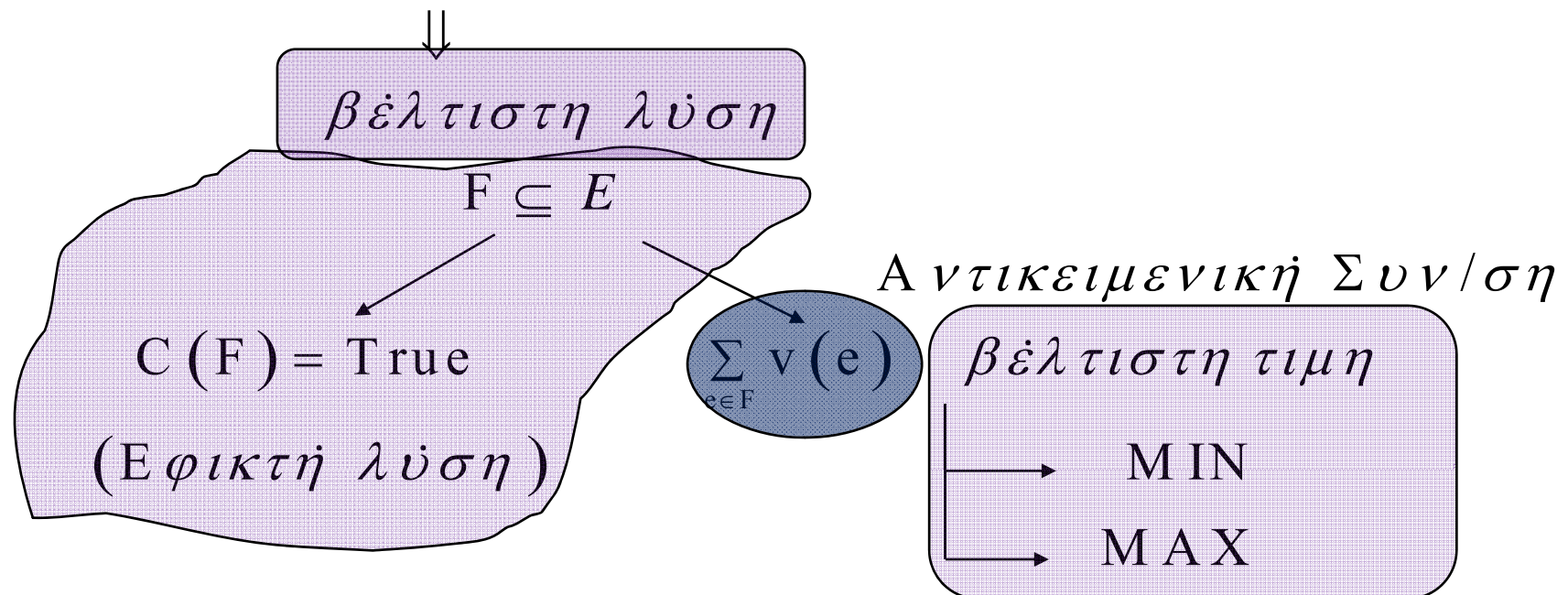
Ελπίδα: τοπικά βέλτιστη επιλογή να οδηγεί σε μια ολικά βέλτιστη λύση

ΝΑΙ: για μερικά προβλήματα

ΌΧΙ: για κάποια άλλα

Προβλήματα

- E πεπερασμένο
- $\forall e \in E \rightarrow v(e) \in \mathbb{N}$ (Αντικειμενική Συν/ση)
- $C : P(E) \rightarrow \{\text{True}, \text{False}\}$ (περιορισμοί)





Άπληστος αλγόριθμος (greedy)

⇒ Σειρά (κανόνας)

⇒ Αρχική λύση $F = \emptyset$

⇒ Τοπική Επιλογή : $F = F \cup \{e\}$

(προοδευτ. αύξηση + ικανοπ. περιορισμών)

FF, NF, BF (διαχ. μνήμης)

FIFO, LRU, OPT (paging)



Γενικός αλγόριθμος

Greedy (E)

$S := \emptyset$ {solution}

for all elements of E do

E περιλαμβάνει $x \in E$;

$E := E - \{x\}$

if $S \cup \{x\}$ feasible

then

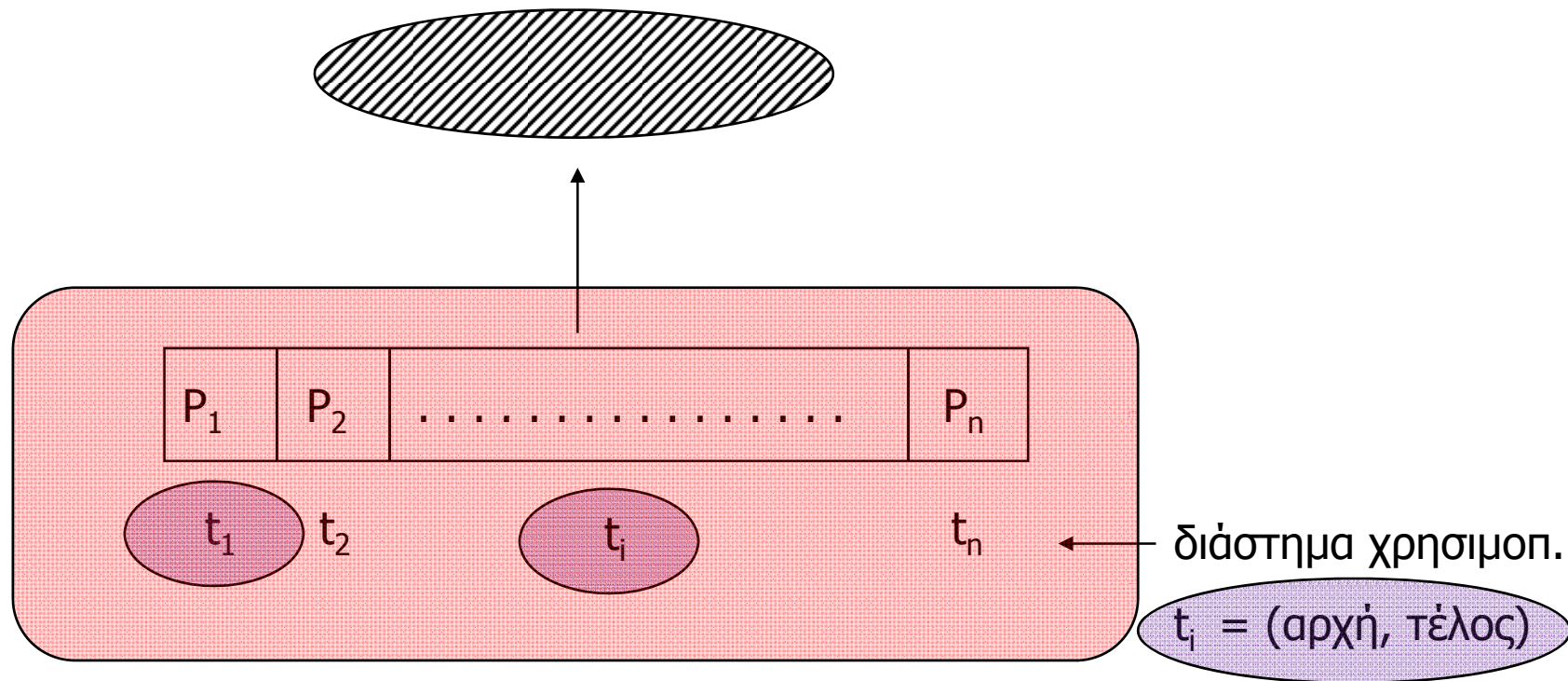
$S := S \cup \{x\}$

Ενημέρωση Αντικειμενικής Συναρτ.

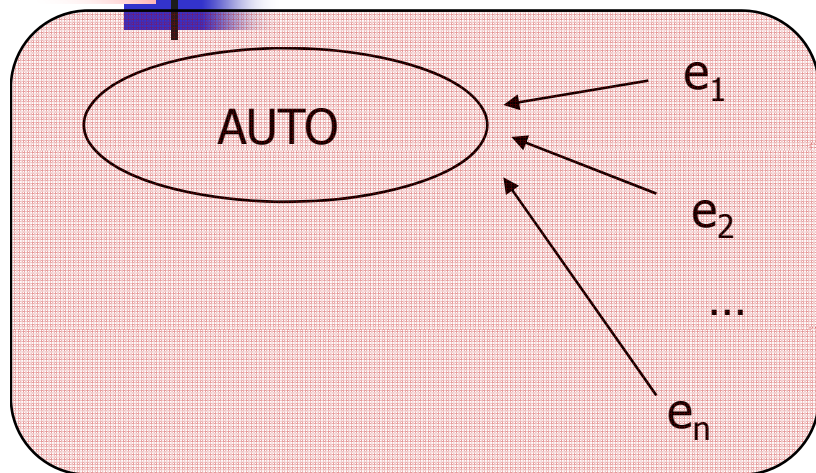
Greedy := S

End

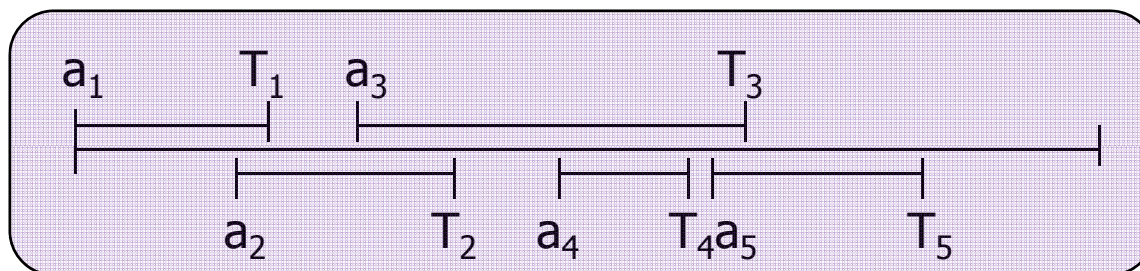
Διαχείριση πόρων



Παράδειγμα

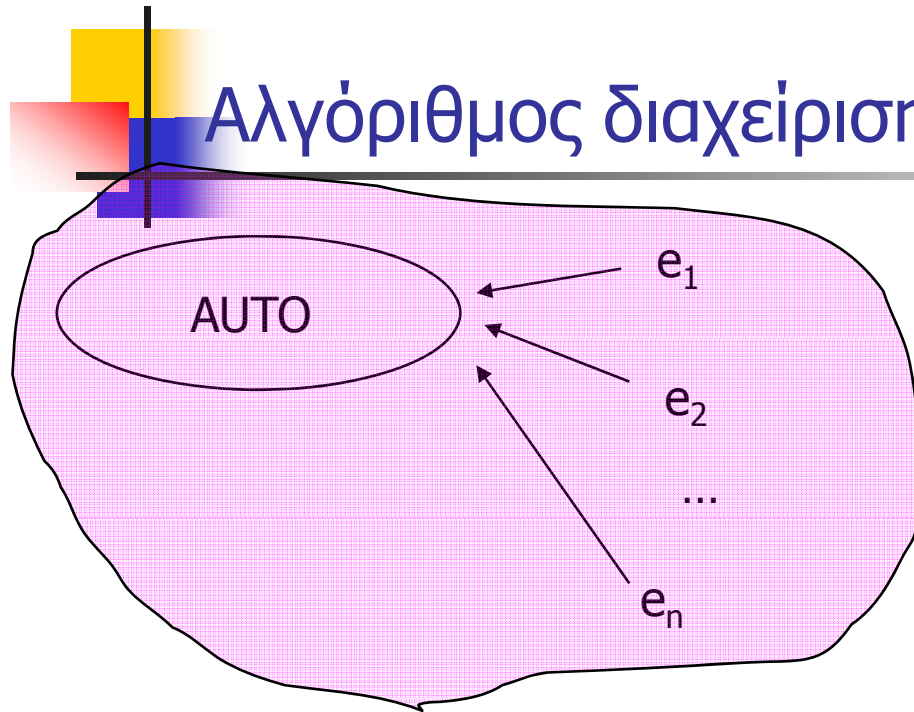


1 μόνο πελάτης σε μία
δεδομένη χρονική στιγμή

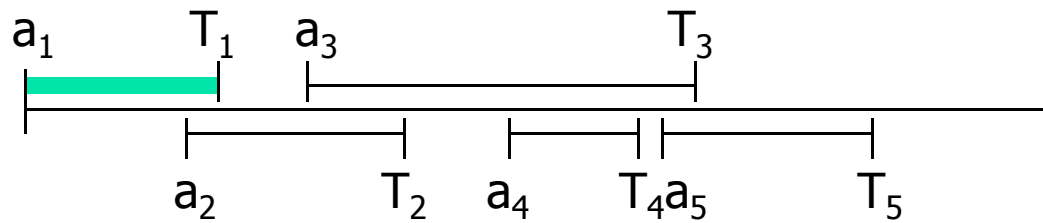


Στόχος: Ικανοποίηση μέγιστου αριθμού πελατών, $n=5$

Αλγόριθμος διαχείρισης ενός πόρου

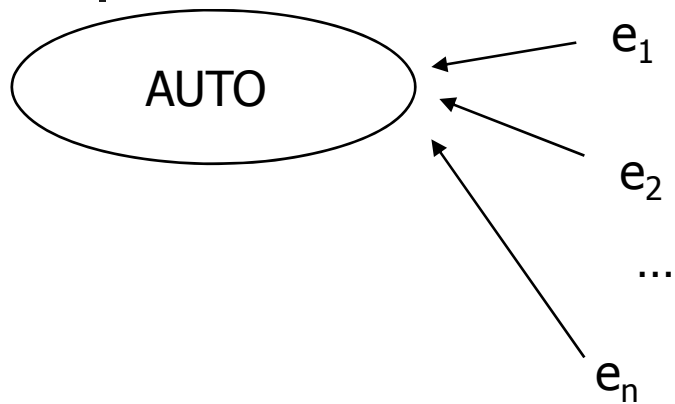


1 μόνο πελάτης σε μία δεδομένη χρονική στιγμή

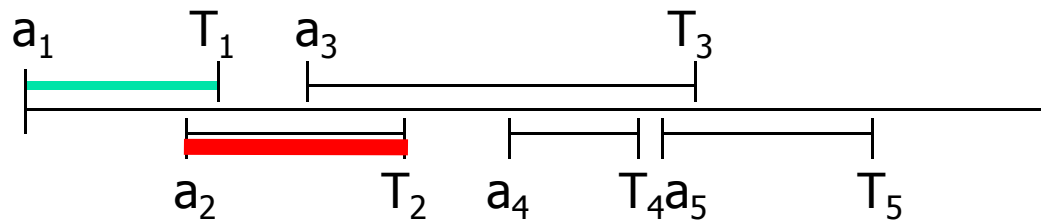


Βήμα 1

Αλγόριθμος διαχείρισης ενός πόρου

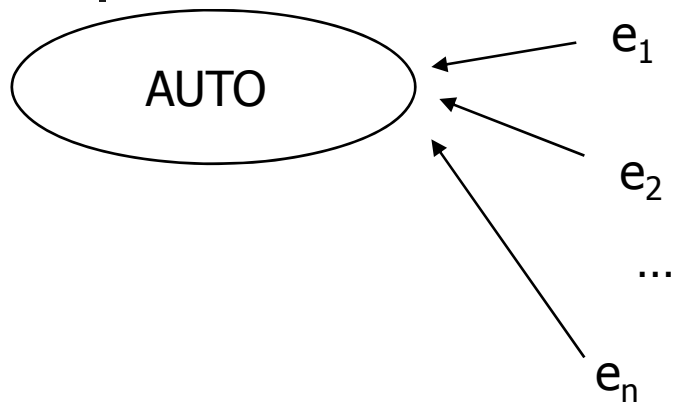


1 μόνο πελάτης σε μία
δεδομένη χρονική στιγμή

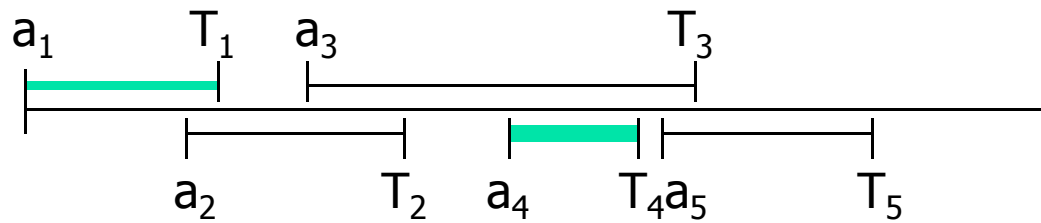


Βήμα 2 : e_2 απορρίπτεται

Αλγόριθμος διαχείρισης ενός πόρου

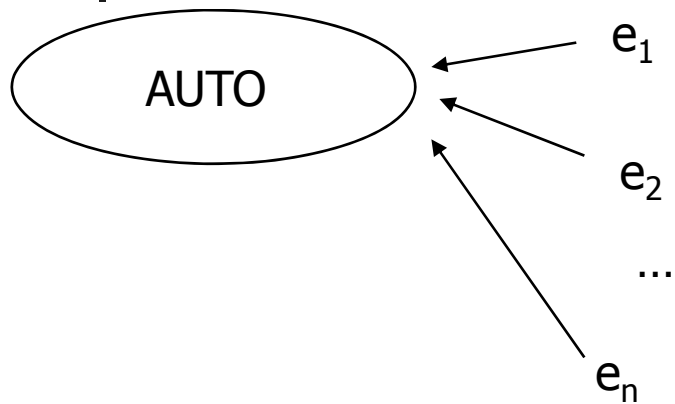


1 μόνο πελάτης σε μία
δεδομένη χρονική στιγμή

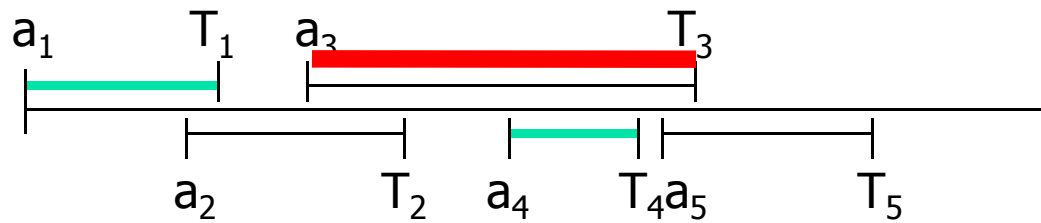


Βήμα 3

Αλγόριθμος διαχείρισης ενός πόρου

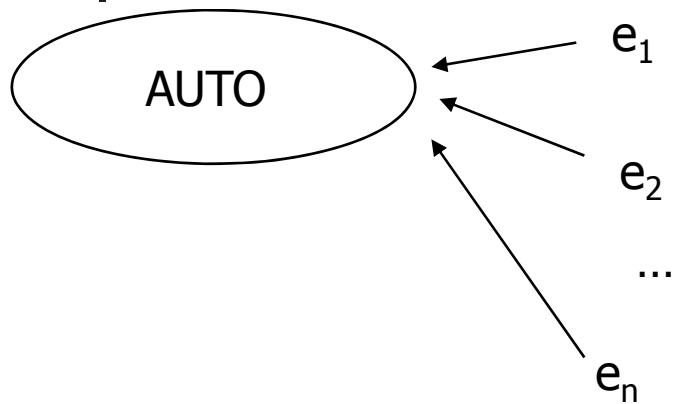


1 μόνο πελάτης σε μία
δεδομένη χρονική στιγμή

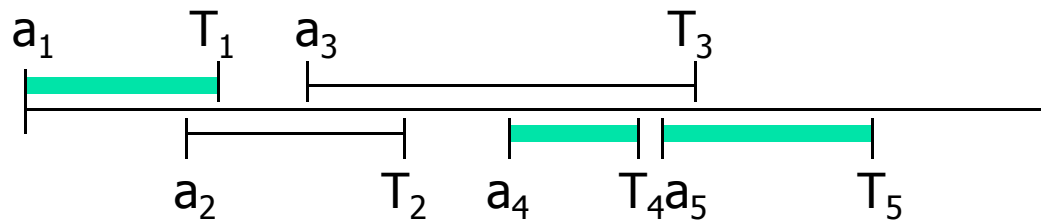


Βήμα 4 : e_3 απορρίπτεται

Αλγόριθμος διαχείρισης ενός πόρου



1 μόνο πελάτης σε μία
δεδομένη χρονική στιγμή



Βήμα 5 (n=5)



Αλγόριθμος διαχείρισης ενός πόρου

Greedy AUTO

Ταξινόμησε τα στοιχεία κατά φθίνουσα σειρά αποπεράτωσης

$F \leftarrow \emptyset$

For $i=1$ to n **do**

if e_i και last in F δεν τέμνονται **then**

$F \leftarrow F \cup \{e_i\};$

Απόδειξη βελτιστότητας

$F = \{x_1, x_2, \dots, x_k, \dots, x_p\}$ λύση Greedy

$Opt = \{y_1, y_2, \dots, y_k, \dots, y_q\}$ $q \geq p$ βέλτιστη

(Ταξίση ημερ. αύξουσα)

$x_1 = y_1, \dots, x_k = y_k, x_{k+1} \neq y_{k+1}$

F απο κατασκευή : $T(x_{k+1}) \leq T(y_{k+1})$

$Opt' = \{x_1, x_2, \dots, x_{k-1}, x_k, x_{k+1}, y_{k+2}, \dots, y_p\} \dots$

$Opt' = \{x_1, x_2, \dots, x_{k-1}, x_k, x_{k+1}, \dots, x_p\}, p = q$



Μεγιστοποίηση ολικής διάρκειας

Ολική διάρκεια ενοικίασης
AUTO : MAX ?

Greedy = ?

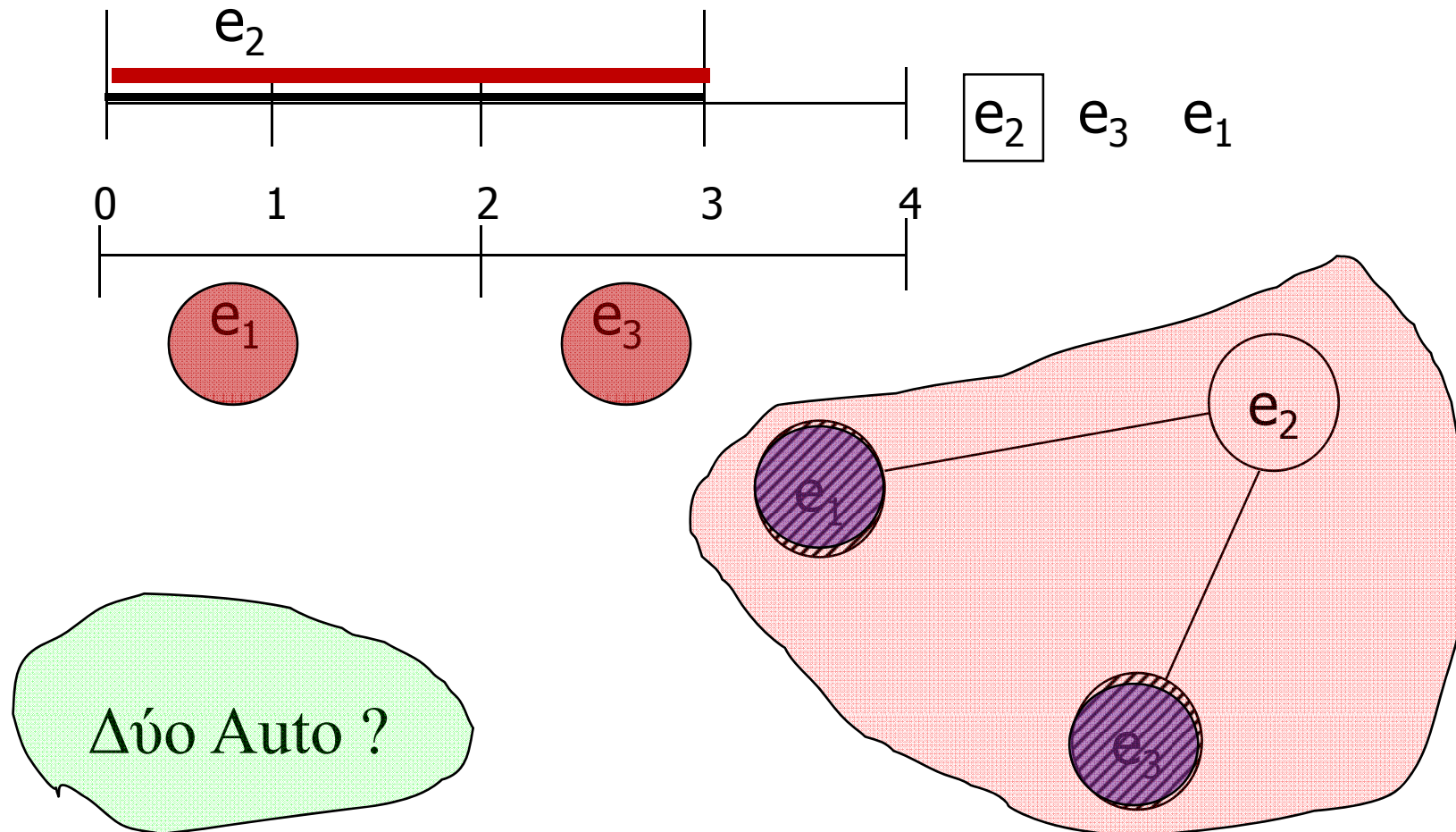
- *Προτεραιότητα : μεγάλη διάρκεια*

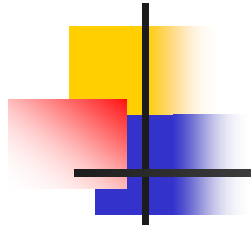


Ταξίση : διαρκείες



Μοντελοποίηση με γράφους





Level algorithms
