

Άπληστοι Αλγόριθμοι

Επιμέλεια διαφανειών: **Δ. Φωτάκης**

Τροποποιήσεις: Α. Παγουρτζής

Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο



Άπληστοι Αλγόριθμοι

- ... για προβλήματα **βελτιστοποίησης**:
 - Λειτουργούν σε **βήματα**.
 - Σε κάθε βήμα, **αμετάκλητη** επιλογή για **τμήμα** της λύσης.
 - **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!**
- Ορθότητα: **εφικτότητα + βελτιστότητα** (συνήθ. επαγωγή).

Άπληστη Στρατηγική

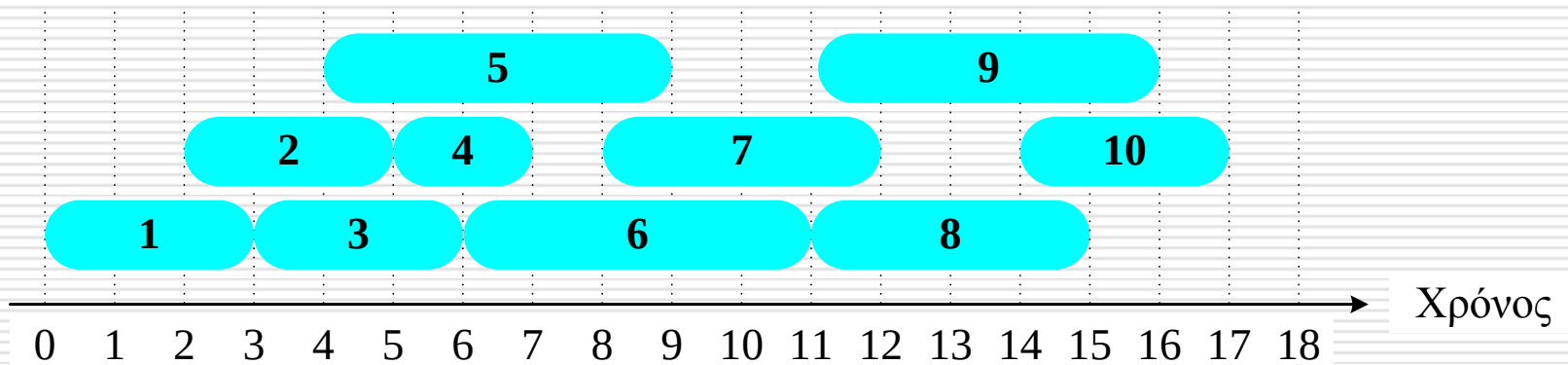
- Σύγκριση **συνιστωσών λύσης** με βάση (απλό) κριτήριο.
- (Αμετάκλητη) επιλογή καθορίζει σε κάθε βήμα **ποια** συνιστώσα είναι **«καλύτερη»** για να συμπεριληφθεί στην τελική λύση.
 - Επιλογή με κάποιον (συνήθως) **απλό κανόνα**.
- **Ίδια στρατηγική** σε υποπρόβλημα που προκύπτει.
 - **Μη-προσαρμοστική**: ίδιο κριτήριο σε όλα τα βήματα.
 - **Προσαρμοστική**: αλλάζει κριτήριο σε κάθε βήμα.
- **Χρόνος εκτέλεσης** συνήθως καθορίζεται από **χρόνο επιλογής «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).
- Ζητείται: **επιλογή** δραστηριοτήτων **χωρίς χρονικές επικαλύψεις** και δρομολόγηση **σε κοινό πόρο** (π.χ. αίθουσα διδασκαλίας, επεξεργαστής).
- Στόχος: δρομολόγηση **μέγιστου** #δραστηριοτήτων.
- Πρόβλημα **συνδυαστικής βελτιστοποίησης**:
 - Κάθε δρομολόγηση χωρίς επικαλύψεις: εφικτή λύση.
 - Στόχος: εφικτή δρομολόγηση με **μέγιστο** #δραστηριοτήτων.

Παράδειγμα

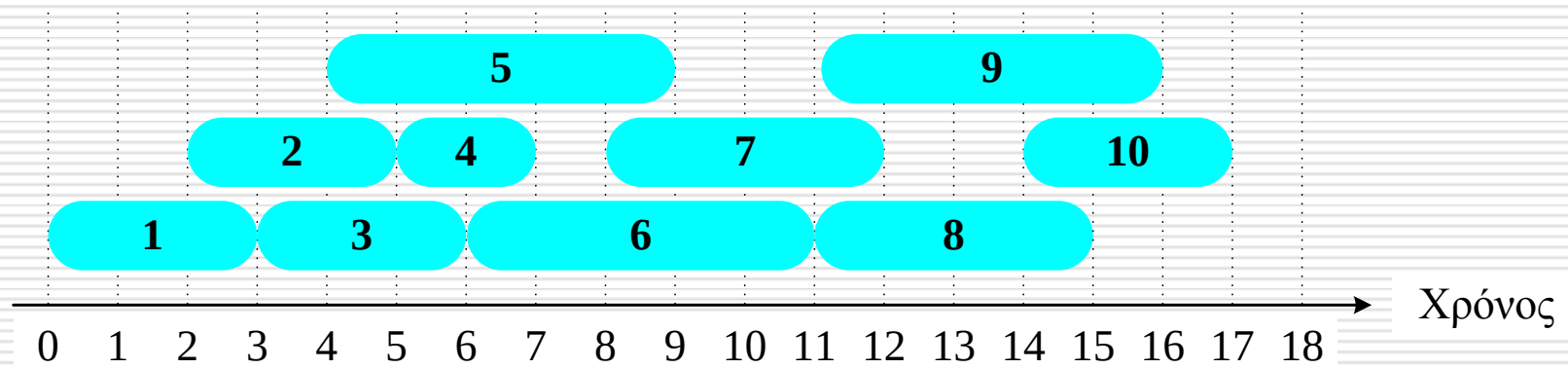
i	1	2	3	4	5	6	7	8	9	10
s_i	0	2	3	5	4	6	8	11	11	14
f_i	3	5	6	7	9	11	12	15	16	17



- 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 ≤ ... ≤ f_n */*

`C ← {1}; j ← 1;`

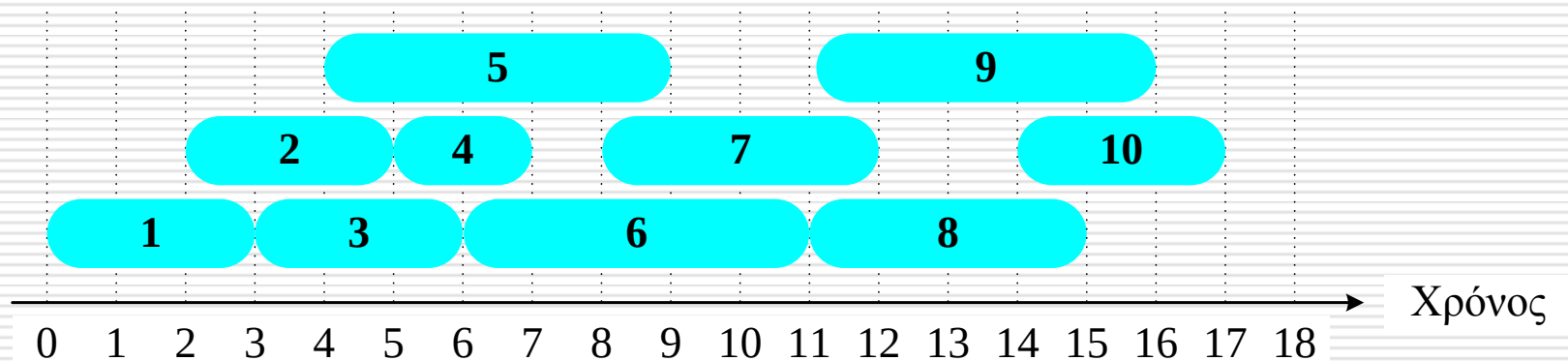
for `i ← 2 to n do`

if `si ≥ fj then`

`C ← C ∪ {i}; j ← i;`

return(`C`);

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



Απόδειξη ορθότητας λύσης

- **Απόδειξη ορθότητας**: απόδειξη **εφικτότητας** (συνήθ. εύκολο) και **βελτιστότητας** (συνήθ. με επαγωγή).
- Βελτιστότητα βασίζεται σε **δύο ιδιότητες** (απαραίτητες!):
 - **Αρχή βελτιστότητας** υπο-λύσεων (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.
 - ... ισοδ.: άπληστη επιλογή **οδηγεί** σε κάποια βέλτιστη λύση.

Απόδειξη βελτιστότητας

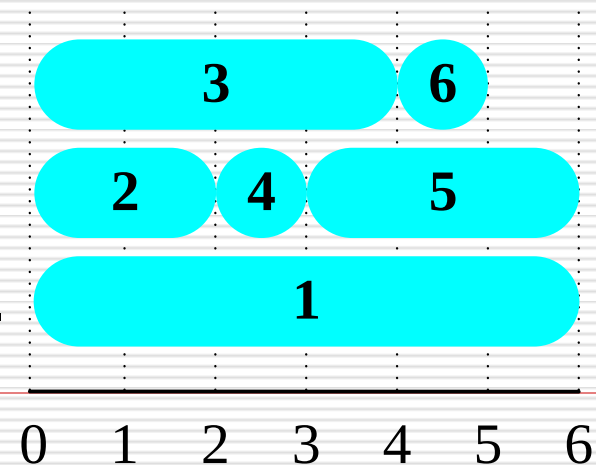
- **Επαγωγή** στον #δραστηριοτήτων.
Υποθέτουμε πάντα ότι $f_1 \leq f_2 \leq \dots \leq f_n$
- **Βάση:** αν 1 δραστ., αυτή επιλέγεται πάντα.
- Έστω αλγ. υπολογίζει **βέλτιστη λύση** για $\leq n - 1$ δραστ.
Θδο. υπολογίζει βέλτιστη λύση για **σύνολο A με n δραστ.**
 - Άπληστος αλγόριθμος **επιλέγει δραστ.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

Ορθότητα

- Αποδείξαμε ότι $|C^*(A)| = 1 + |C^*(A_1)|$
- Ιδιότητα **ἀπληστης 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).
- Εναλλακτική απόδειξη με **επιχείρημα ανταλλαγής** (και επαγωγή):
 - $f_j \geq f_1 \Rightarrow (C^*(A) \setminus \{j\}) \cup \{1\}$ είναι επίσης βέλτιστη λύση.

Χρωματισμός Διαστημάτων

- n διαστήματα: αρχή και τέλος $[s_i, f_i) : f_i > s_i \geq 0$
- Ζητείται: χρωματισμός όλων ώστε επικαλυπτόμενα διαστήματα να έχουν διαφορετικό χρώμα.
- Στόχος: χρωματισμός με **ελάχιστο** #χρωμάτων.
- Άπληστος αλγόριθμος:
 - Εξέταση σημείων ευθείας σε αύξουσα σειρά.
 - Κάθε διάστημα που τελειώνει «απελευθερώνει» το χρώμα του.
 - Κάθε διάστημα που αρχίζει παίρνει πρώτο διαθέσιμο χρώμα.
- Χρήση χρώματος $d \geq 2$ μόνο αν επικάλυψη d διαστ/των ταυτόchr.



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:

$$\begin{aligned} T &= t_1 + (t_1 + t_2) + (t_1 + t_2 + t_3) + \dots + (t_1 + \dots + t_n) \\ &= n t_1 + (n - 1) t_2 + (n - 2) t_3 + \dots + t_n \\ &= \sum_{i=1}^n (n - i + 1) t_i \end{aligned}$$

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

Βελτιστότητα: 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^* : $\lambda^*(j) = k$
 - συμφωνία σε $k - 1$ αρχικές, ισχύει $k < j$ 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 , υπάρχει βέλτιστη δρομολόγηση π^* που συμφωνεί με άπληστη στη σειρά των k πρώτων εργασιών.
 - (Άπληστη) επιλογή συντομότερης διαθέσιμης εργασίας $\rightarrow$ βέλτιστη λύση.
- Ιδιότητα **βελτιστότητας υπο-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 **βελτιστότητας**: (επαγωγική) εφαρμογή ιδιότητας άπληστης επιλογής.

Πρόβλημα του Περιπτερά

- Κέρματα αξίας 1, 5, και 20 λεπτών.
- Στόχος: 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**:
 - Θδο 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 βελτιστότητας: με **επαγωγική** εφαρμογή ιδιότητας **άπληστης επιλογής**.

Βελτιστότητα

□ Επαγωγή σε #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

- Σύγκριση **συνιστωσών λύσης** με βάση κάποιο κριτήριο (π.χ. **σακίδιο**: είδη σε **φθίνουσα σειρά αξία / μέγεθος**).
- (Αμετάκλητη) επιλογή καθορίζει **ποια** είναι «**καλύτερη**» συνιστώσα για να συμπεριληφθεί στη λύση.
- **Ίδια στρατηγική** σε υπο-πρόβλημα που προκύπτει.
 - **Μη-προσαρμοστική**: ίδιο κριτήριο σε όλα τα βήματα.
 - **Προσαρμοστική**: αλλάζει κριτήριο σε κάθε βήμα.
- **Χρόνος εκτέλεσης** καθορίζεται από **χρόνο επιλογής καλύτερης επόμενης συνιστώσας**.
- Απόδειξη **ορθότητας (βελτιστότητας)**: συνήθ. επαγωγή.
 - Ιδιότητα **άπληστης επιλογής**.
 - Ιδιότητα **βελτιστότητας υπο-λύσεων**.