Προσεγγιστικοί Αλγόριθμοι

Συνδυαστική Βελτιστοποίηση

Βασίλης Ζησιμόπουλος

Θεωριτική Πληροφορική
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Weighted vertex cover

Γραμμικό πρόγραμμα

$$\begin{aligned} \text{minimize:} \quad & \sum_{i=1}^{n} w_i x_i \\ \text{subject to:} \quad & x_i + x_j \geq 1, && \forall [u, v] \in E \\ & x_i \in \{0, 1\}, && \forall u_i \in V \end{aligned}$$

Relaxation: $x_i \in [0, 1]$

Rounding για το Vertex Cover

Algorithm 1 Vertex Cover με Rounding

1: $C \leftarrow \emptyset$
2: Λύσε το χαλαρωμένο γραμμικό πρόγραμμα για το Vertex Cover.
3: Έστω $\bar{x}$ η λύση του.
4: **for** $i = 1$ **to** n **do**
5: **if** $\bar{x_i} \geq \frac{1}{2}$ **then**
6: $C \leftarrow C \cup \{v_i\}$
7: **return** C

2-προσεγγιστικός

- z^* το κόστος της λύσης του LP
- C το κάλυμα που επιστρέφει ο αλγόριθμος 1.
- C^* το βέλτιστο κάλυμα.

2-προσεγγιστικός

- z^* το κόστος της λύσης του LP
- C το κάλυμα που επιστρέφει ο αλγόριθμος 1.
- C^* το βέλτιστο κάλυμα.

$$z^* = \sum_{i=1}^{n} w_i \bar{x}_i \geq \sum_{i:\bar{x}_i \geq 1/2} w_i \bar{x}_i \geq \sum_{i:\bar{x}_i \geq 1/2} \frac{1}{2} w_i = \frac{1}{2} w(C)$$

2-προσεγγιστικός

- z^* το κόστος της λύσης του LP
- C το κάλυμα που επιστρέφει ο αλγόριθμος 1.
- C^* το βέλτιστο κάλυμα.

$$z^* = \sum_{i=1}^{n} w_i \bar{x}_i \geq \sum_{i:\bar{x}_i \geq 1/2} w_i \bar{x}_i \geq \sum_{i:\bar{x}_i \geq 1/2} \frac{1}{2} w_i = \frac{1}{2} w(C)$$

$$z^* \geq \frac{1}{2} w(C) \Rightarrow w(C) \leq 2z^* \leq 2w(C^*) \Rightarrow \frac{w(C)}{w(C^*)} \leq 2$$

Πολυωνυμικές Παραλλαγές του Vertex Cover

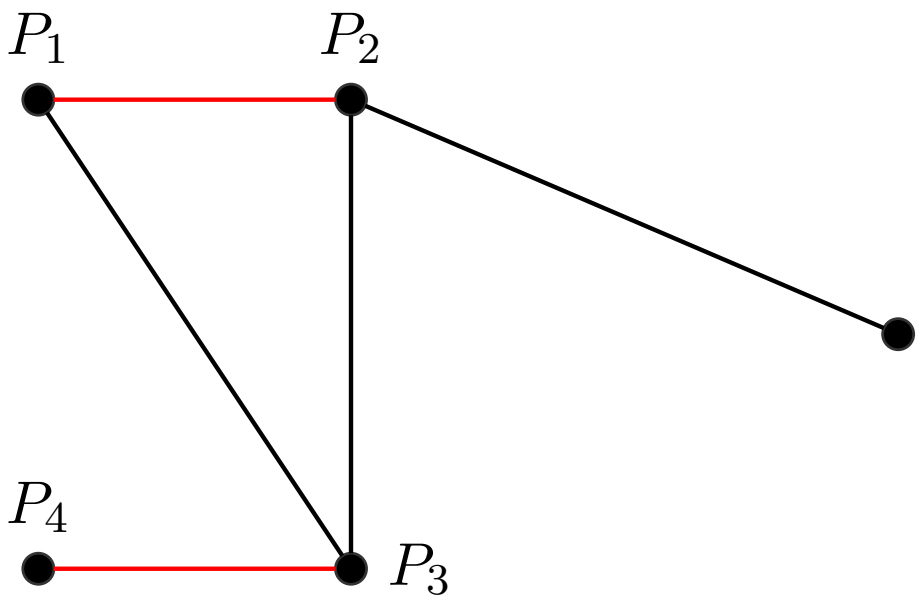
- Δέντρα/Δάση (Trees/Forests)
- Γράφοι Διαστημάτων (Interval Graphs)
- Χορδικοί Γράφοι (Chordal Graphs)
- Διμερείς Γράφοι (Bipartite Graphs)

Minimum Bin Packing

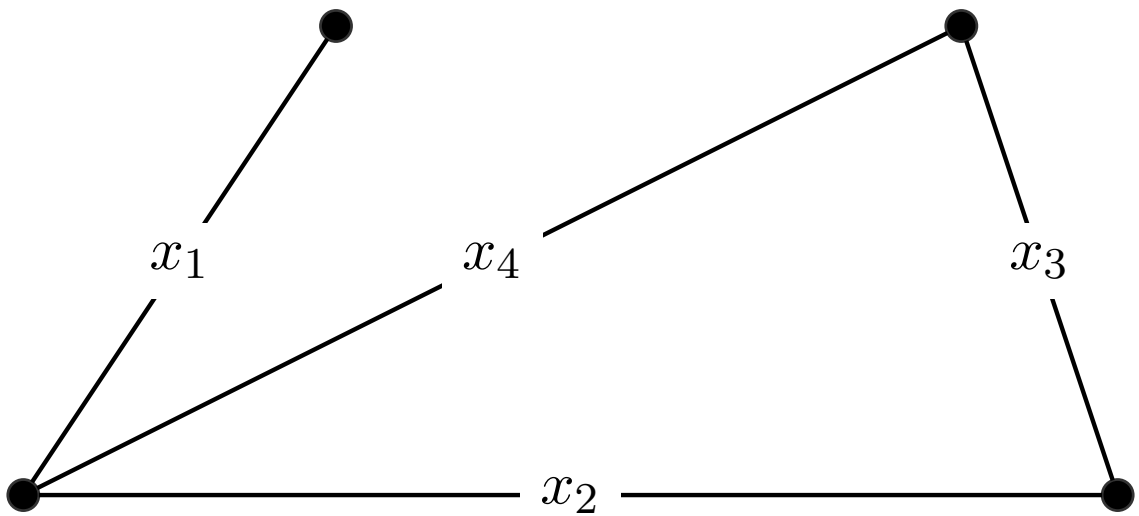
- Πεπερασμένο σύνολο από αντικείμενα.
- $t(u)$ το μέγεθος αντικειμένου $u \in U$ (ακέραιος).
- B η χωρητικότητα του κάδου.
- $\sum_{u \in U_i} t(u) \leq B, \quad 1 \leq i \leq m$

Bin Packing (Polynomial case)

- $P_1, P_2, \ldots, P_n$ αντικείμενα και άπειρο αριθμό κάδων μεγέθους B.
- $P_i > B/3$. Το πολύ 2 από κάθε P_i σε κάθε κάδο.
- max couples.



Πρόβλημα Καλύματος Ακμών (Άσκηση)



Πρόβλημα Χρονοπρογραμματισμού σε Γνωστό Πλήθος Επεξεργαστών (M-Processor Scheduling)

- m πανομοιότυποι επεξεργαστές $P_1, P_2, \ldots, P_m$.
- n ανεξάρτητες εργασίες.
- $t_1, t_2, \ldots, t_n$ χρόνοι εκτέλεσης.
- ελάχιστος χρόνος περάτωσης χωρίς προεκχώρηση (preemption).
- NP-Complete ακόμα και για $m = 2$.

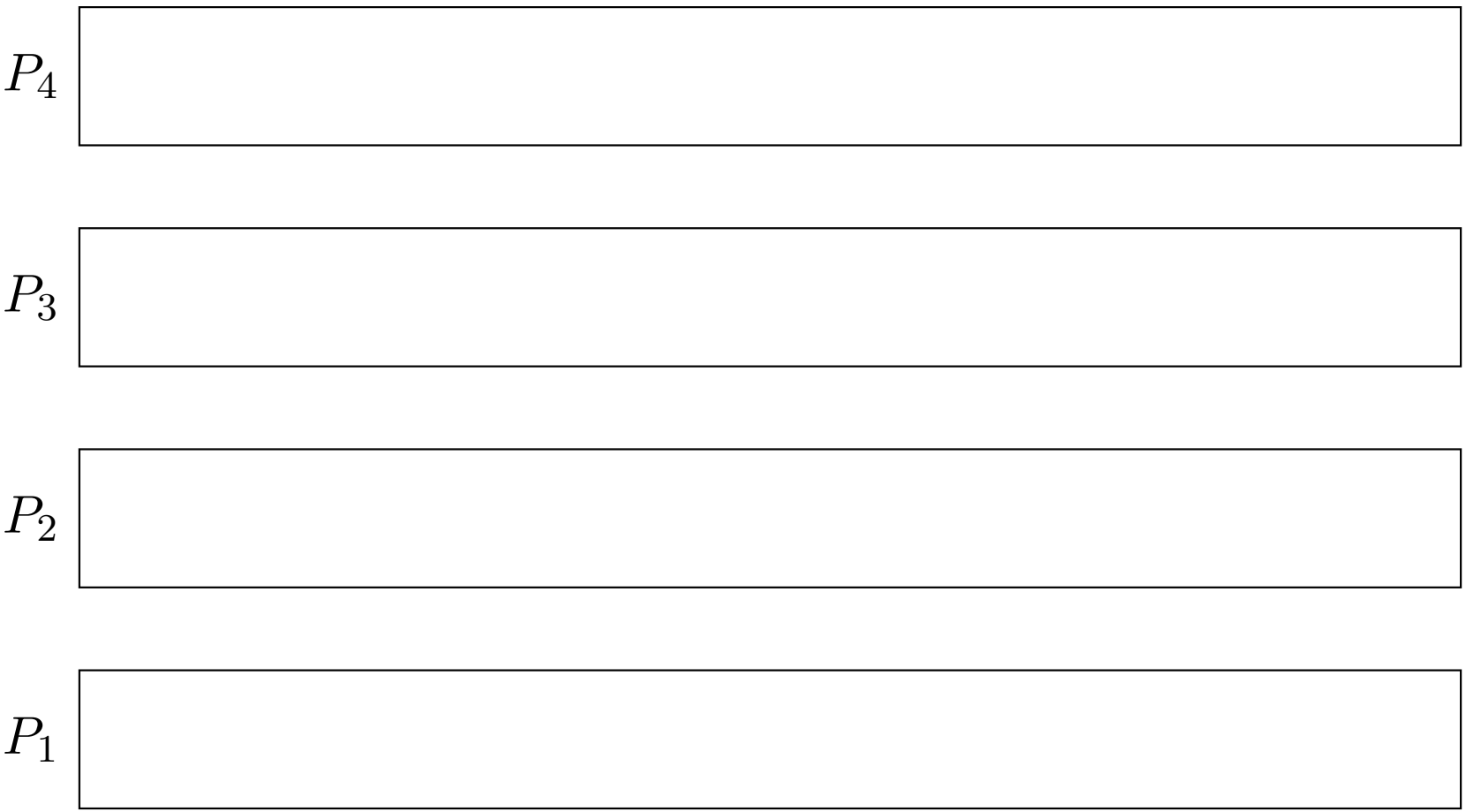
Approximation Algorithm - List Scheduling

Algorithm 2 List Scheduling

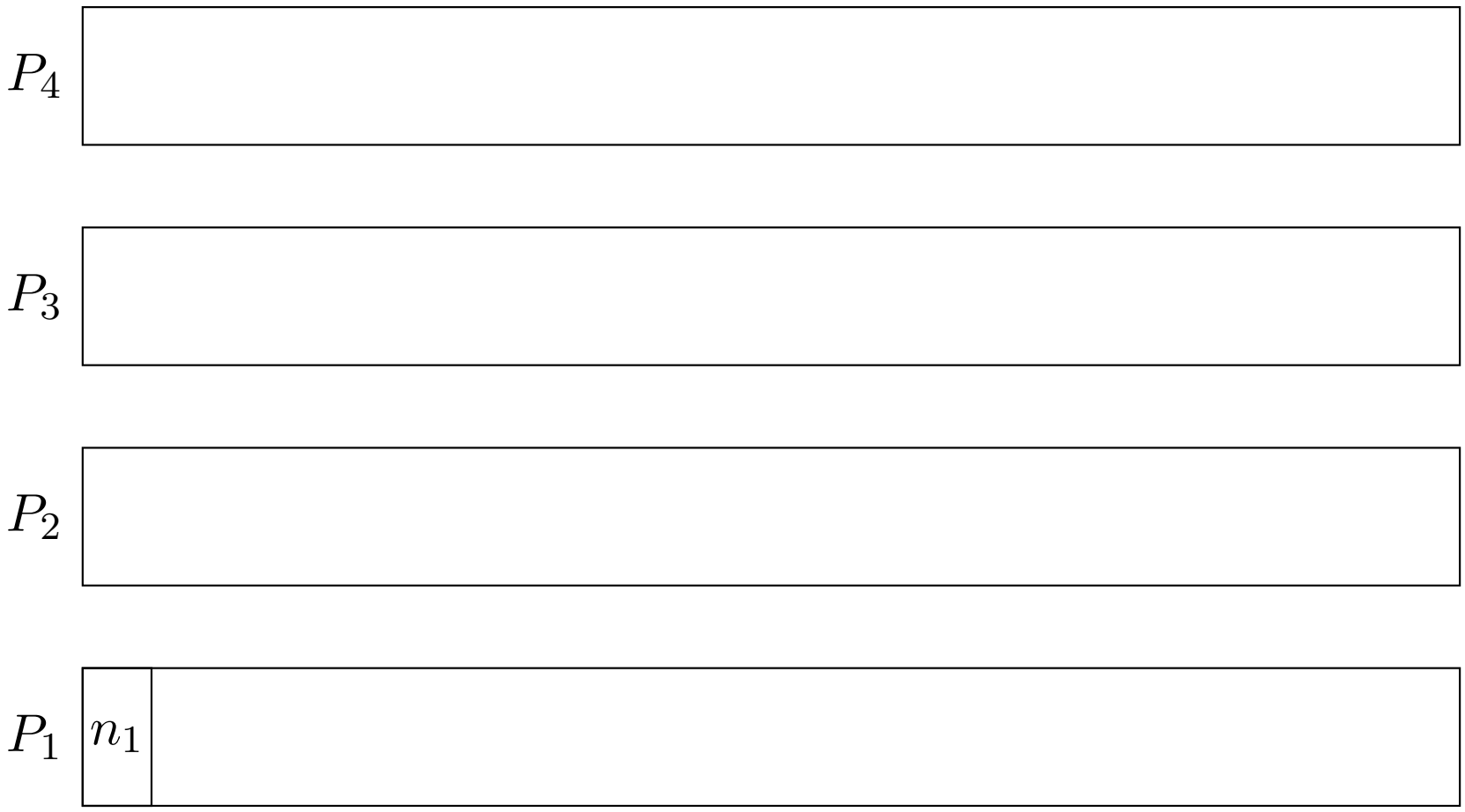
1: **for** $i = 1$ **to** n **do**
2: ανάθεσε την εργασία i στον επεξεργαστή με το μικρότερο φόρτο
3: **return** τα σύνολα εργασιών που ανατέθηκαν σε κάθε επεξεργαστή

$(2 - \frac{1}{m})$ - προσεγγιστικός.

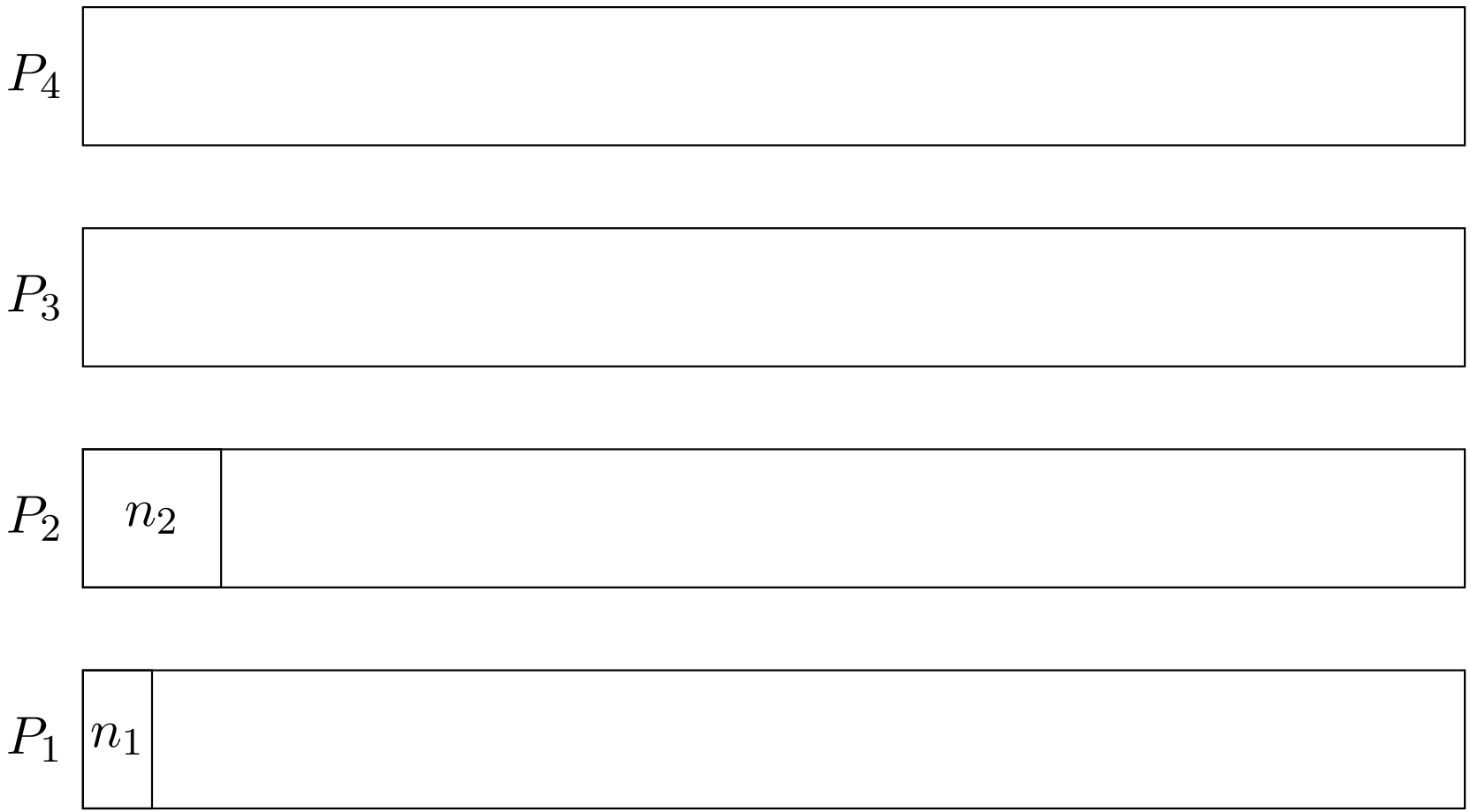
Παράδειγμα



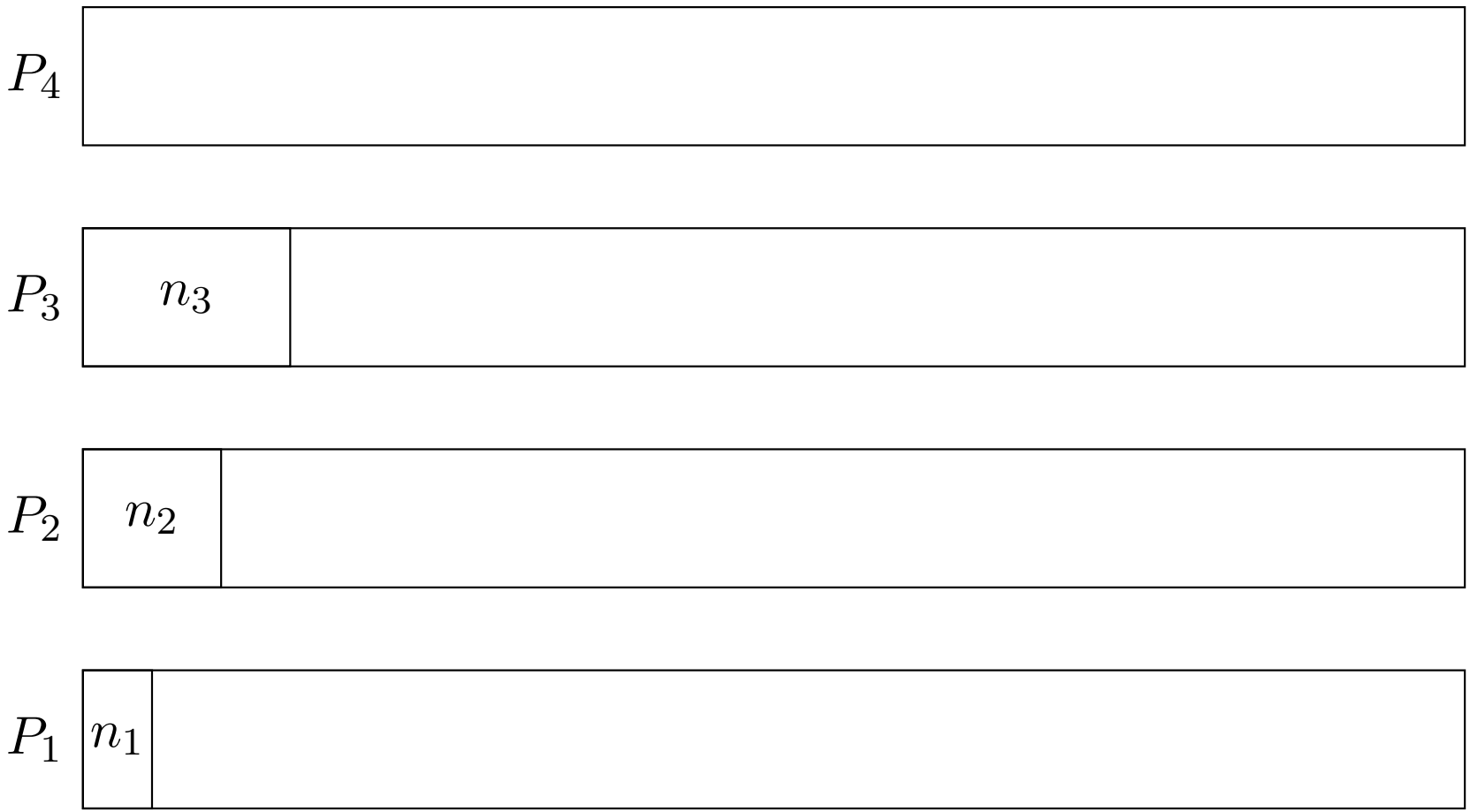
Παράδειγμα



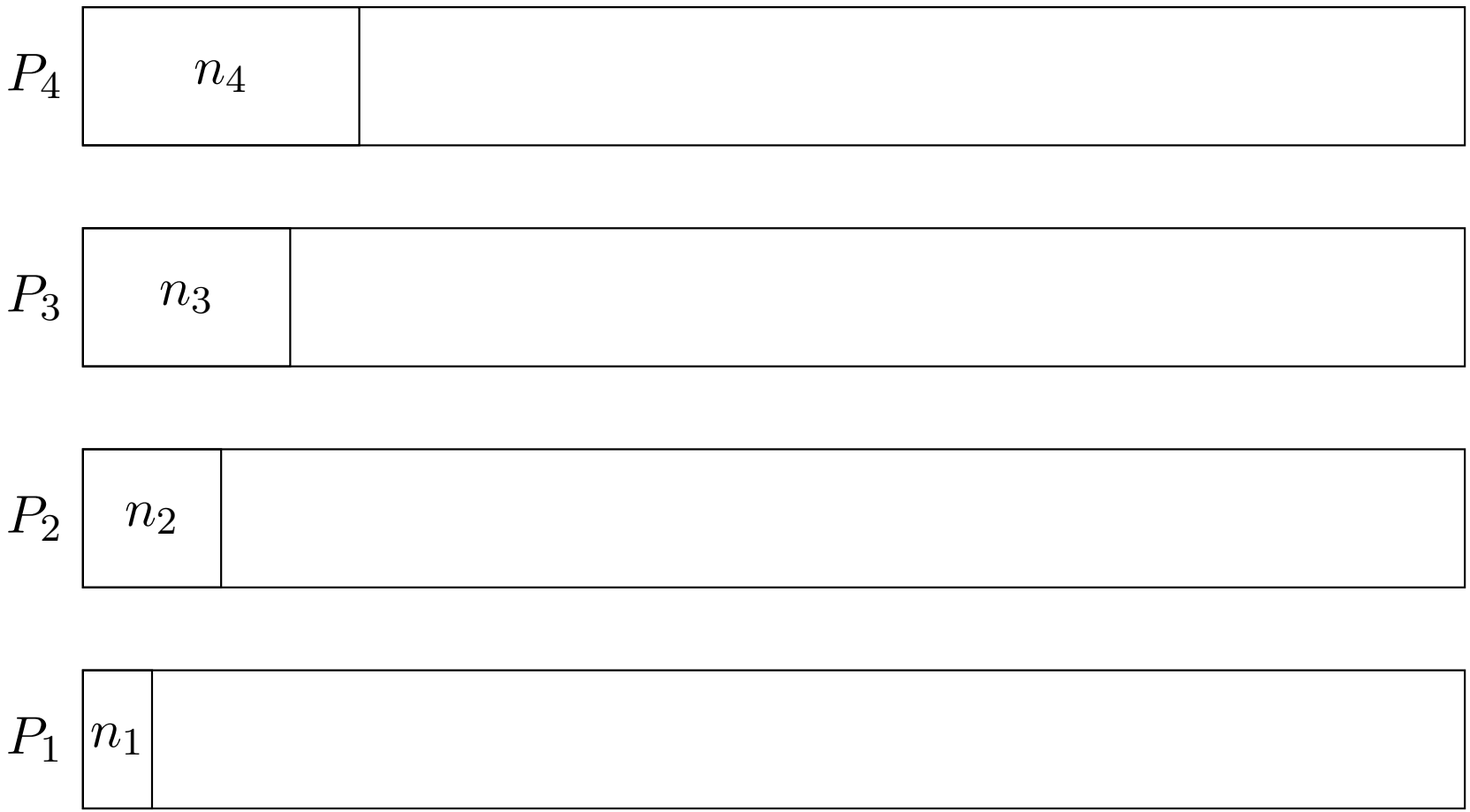
Παράδειγμα



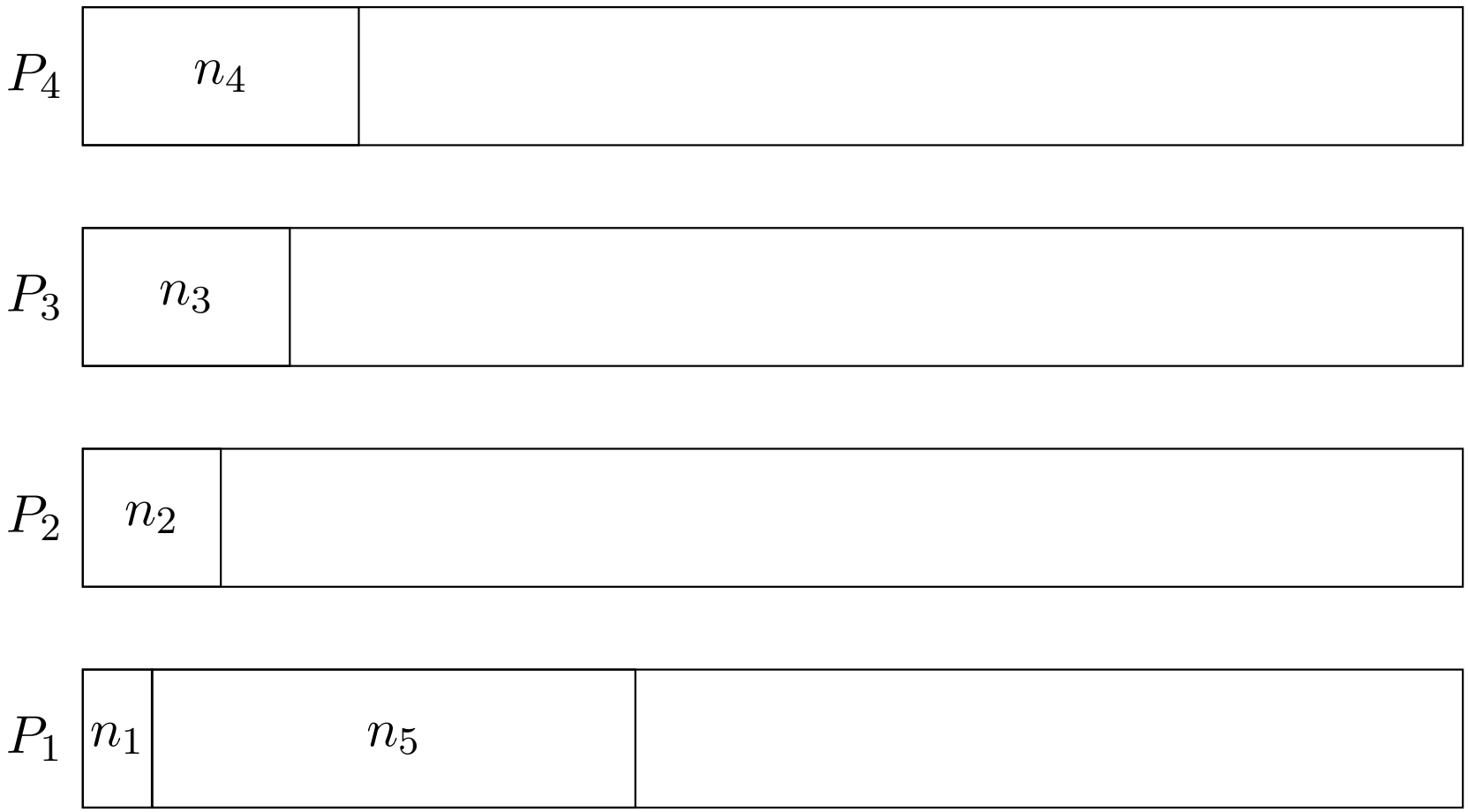
Παράδειγμα



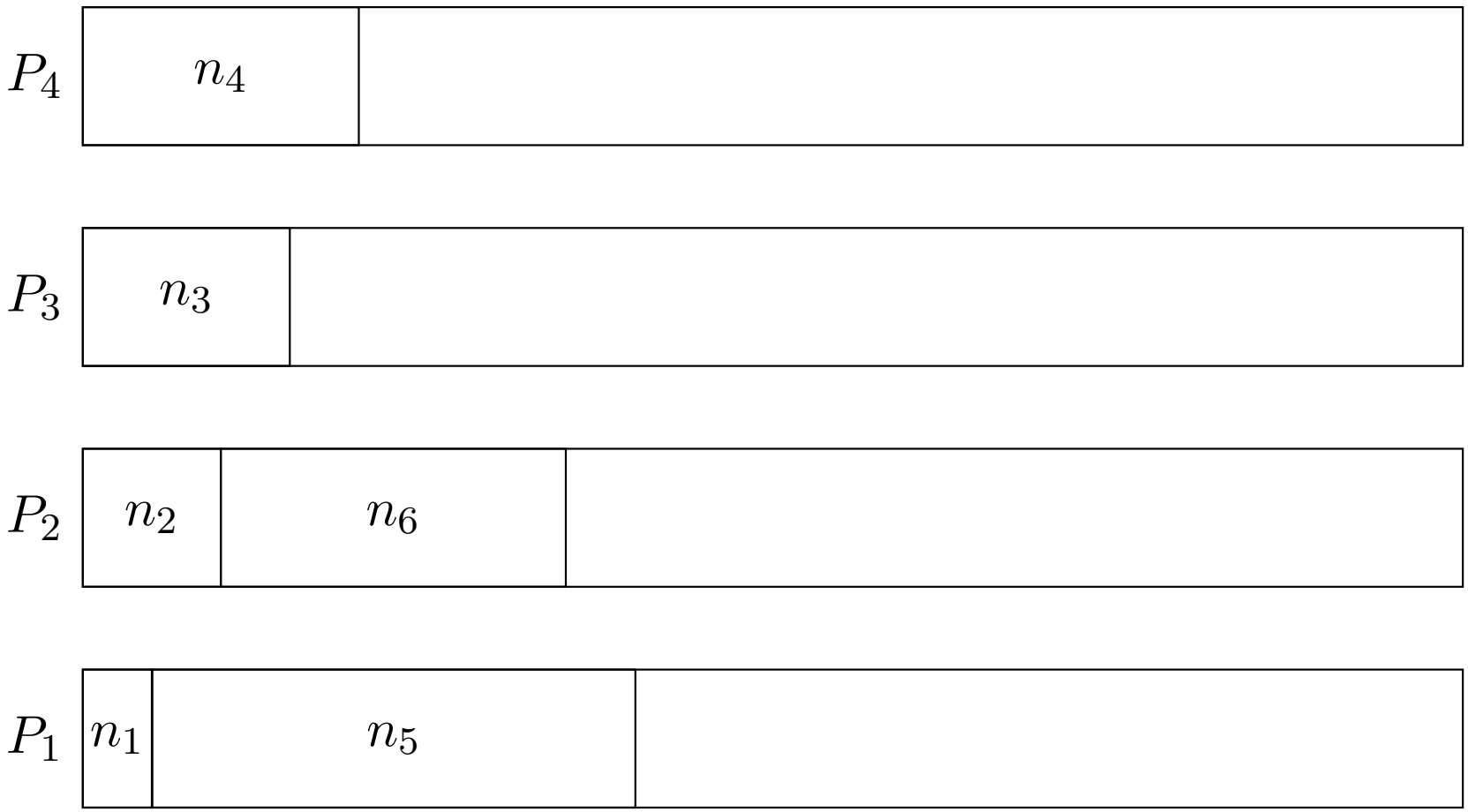
Παράδειγμα



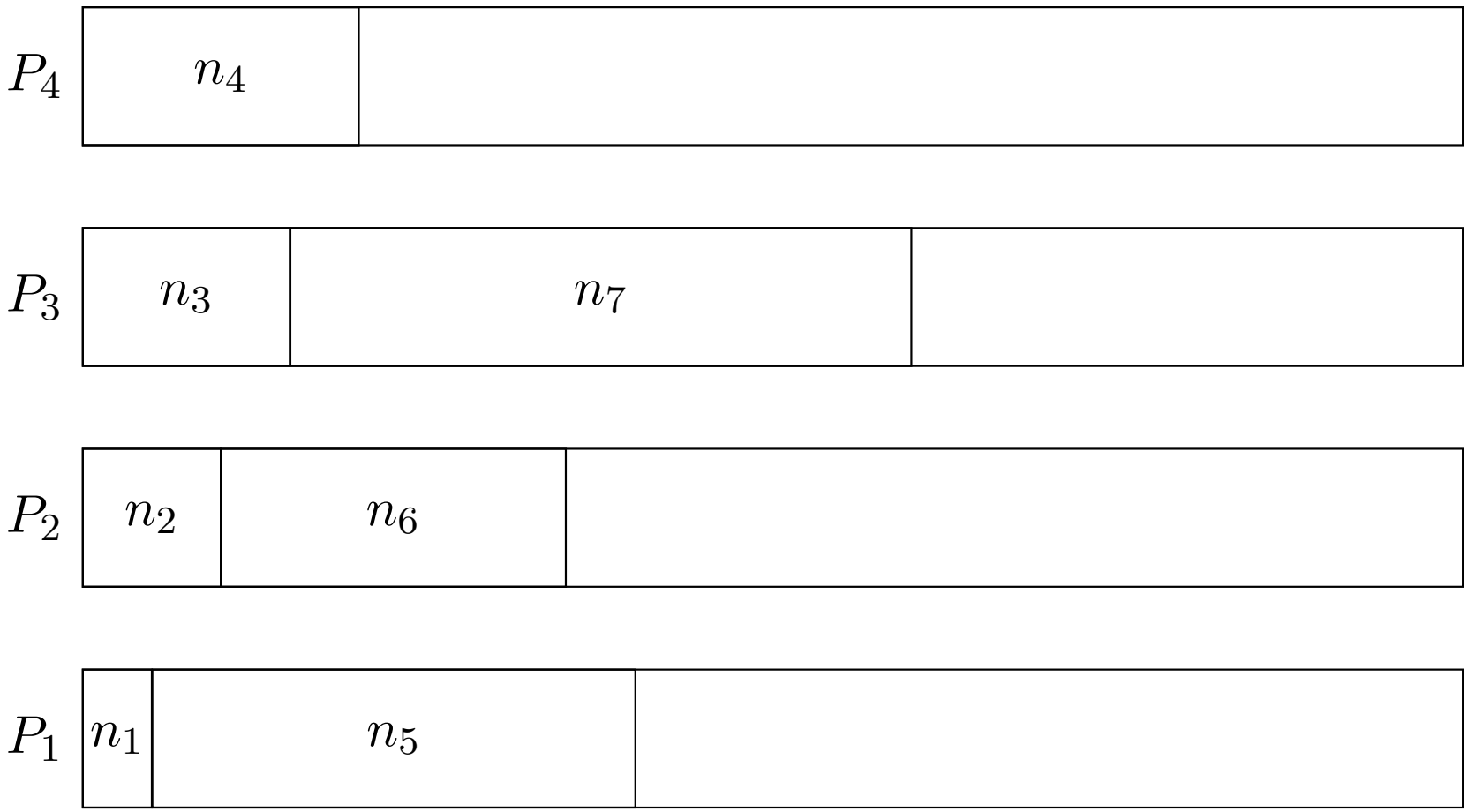
Παράδειγμα



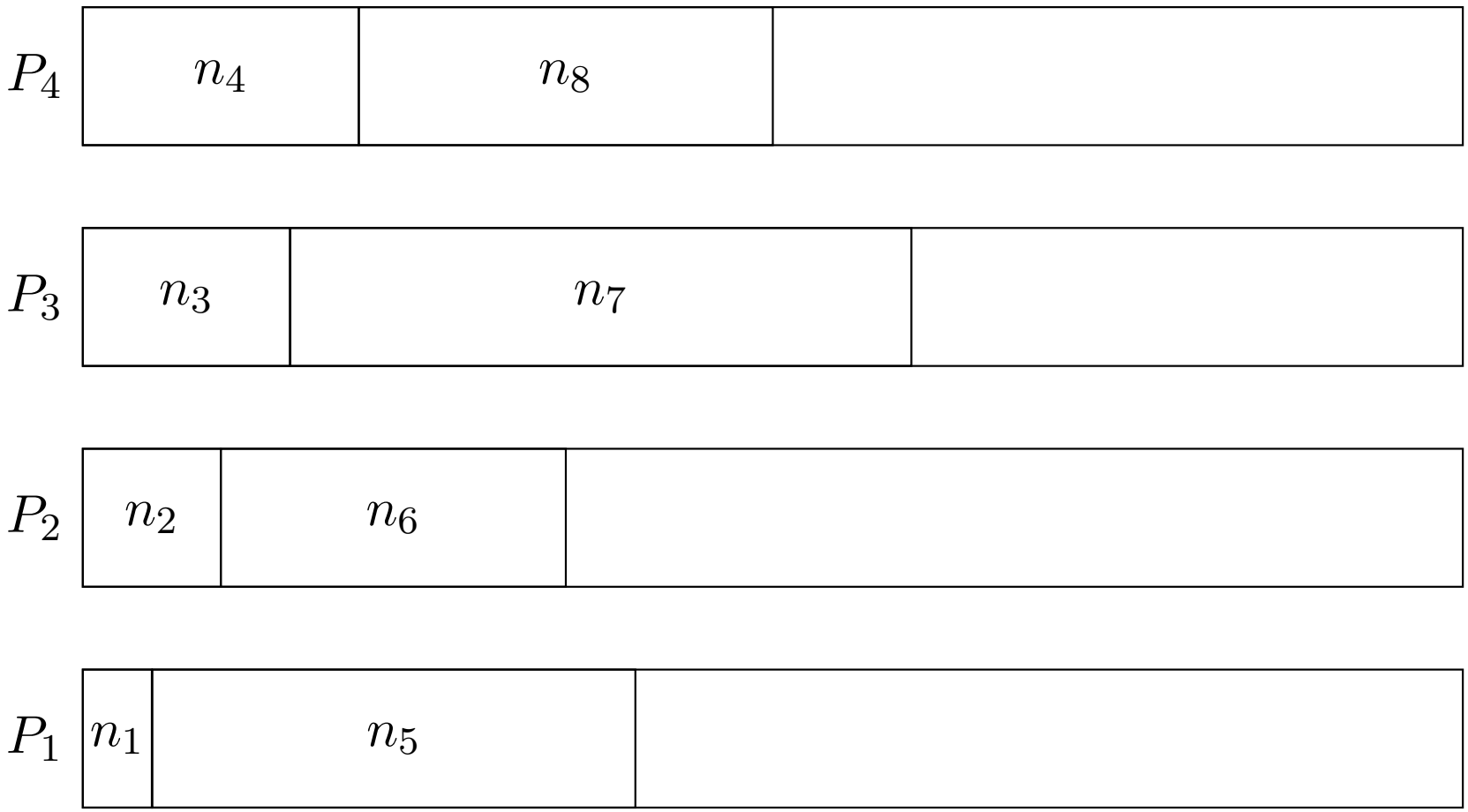
Παράδειγμα



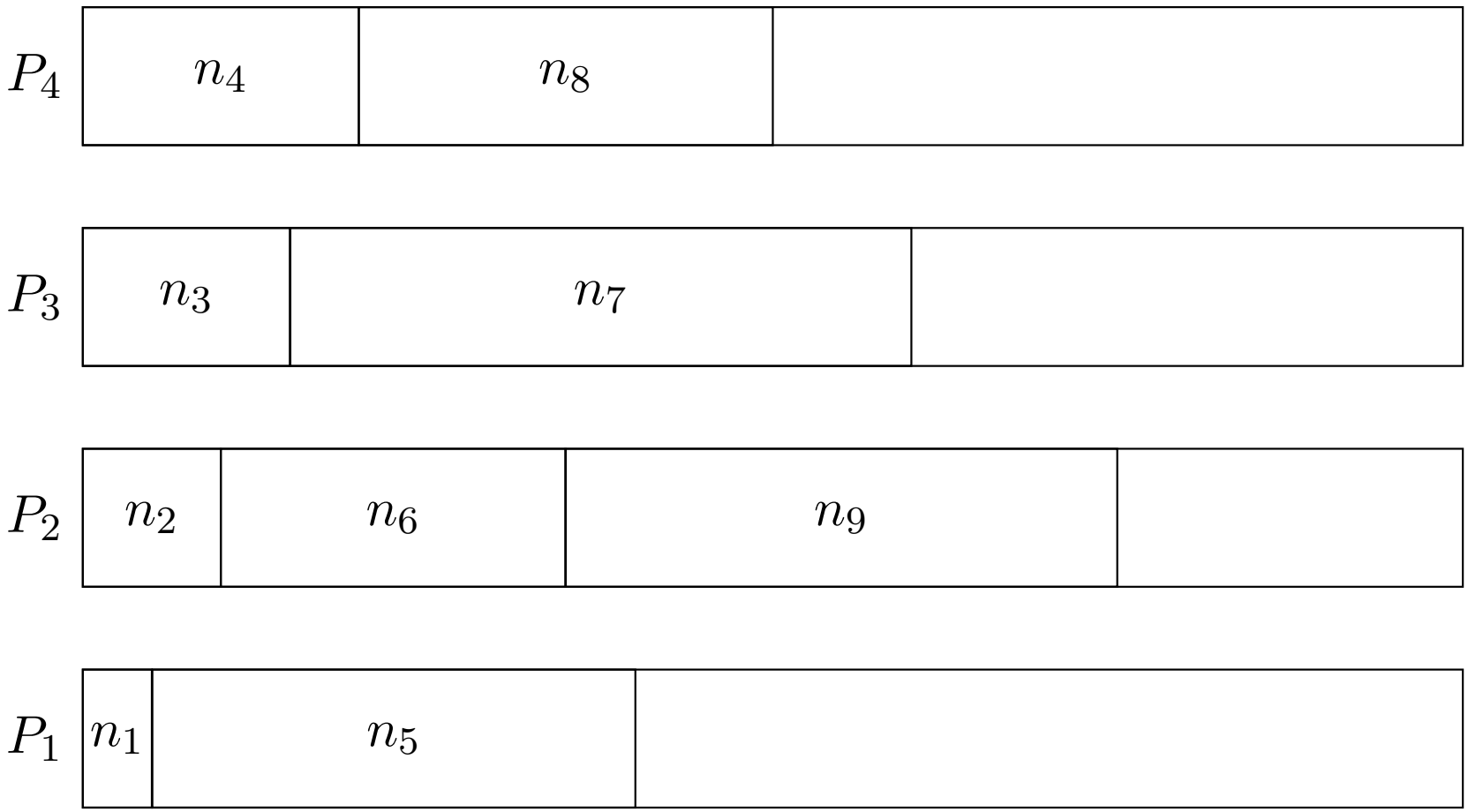
Παράδειγμα



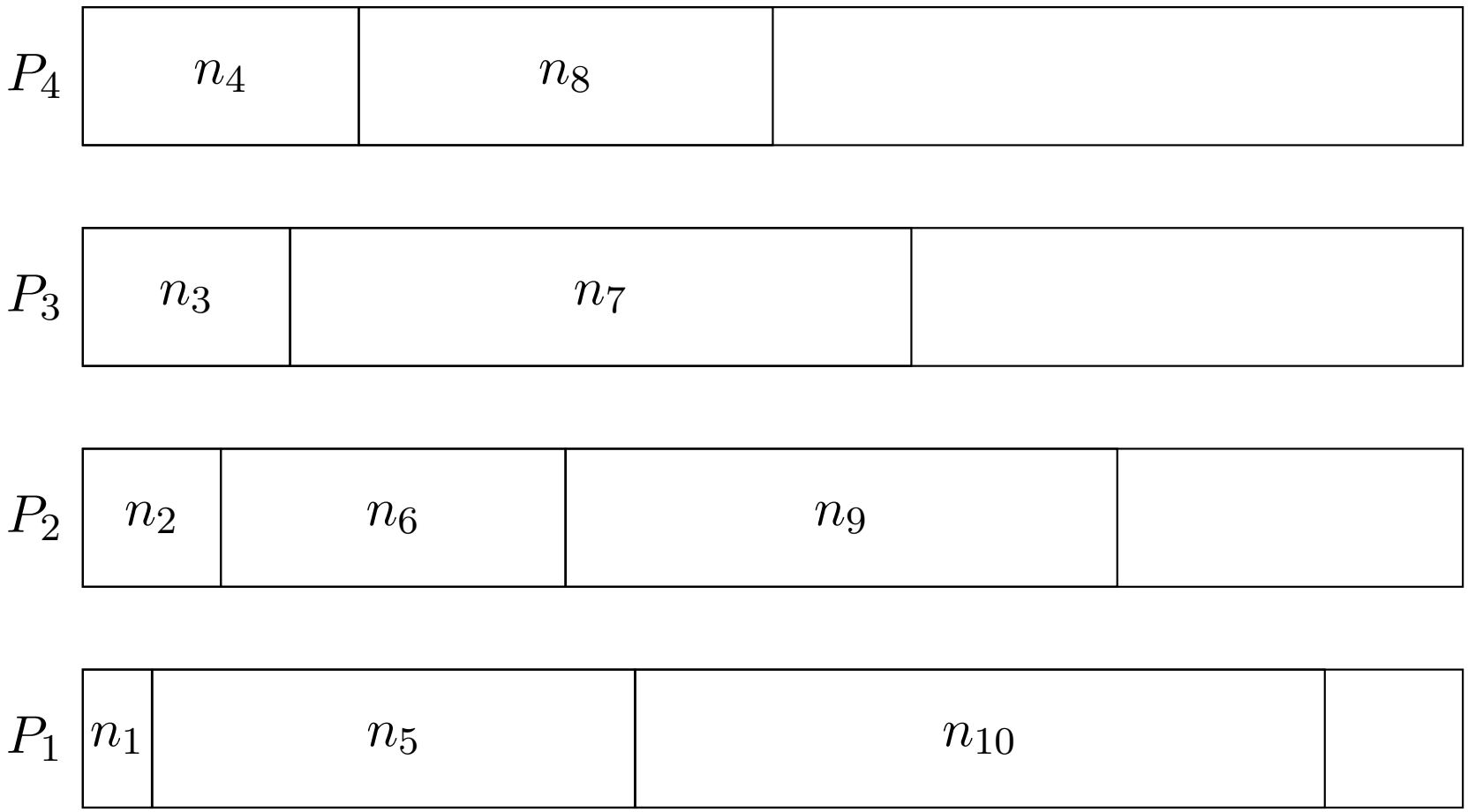
Παράδειγμα



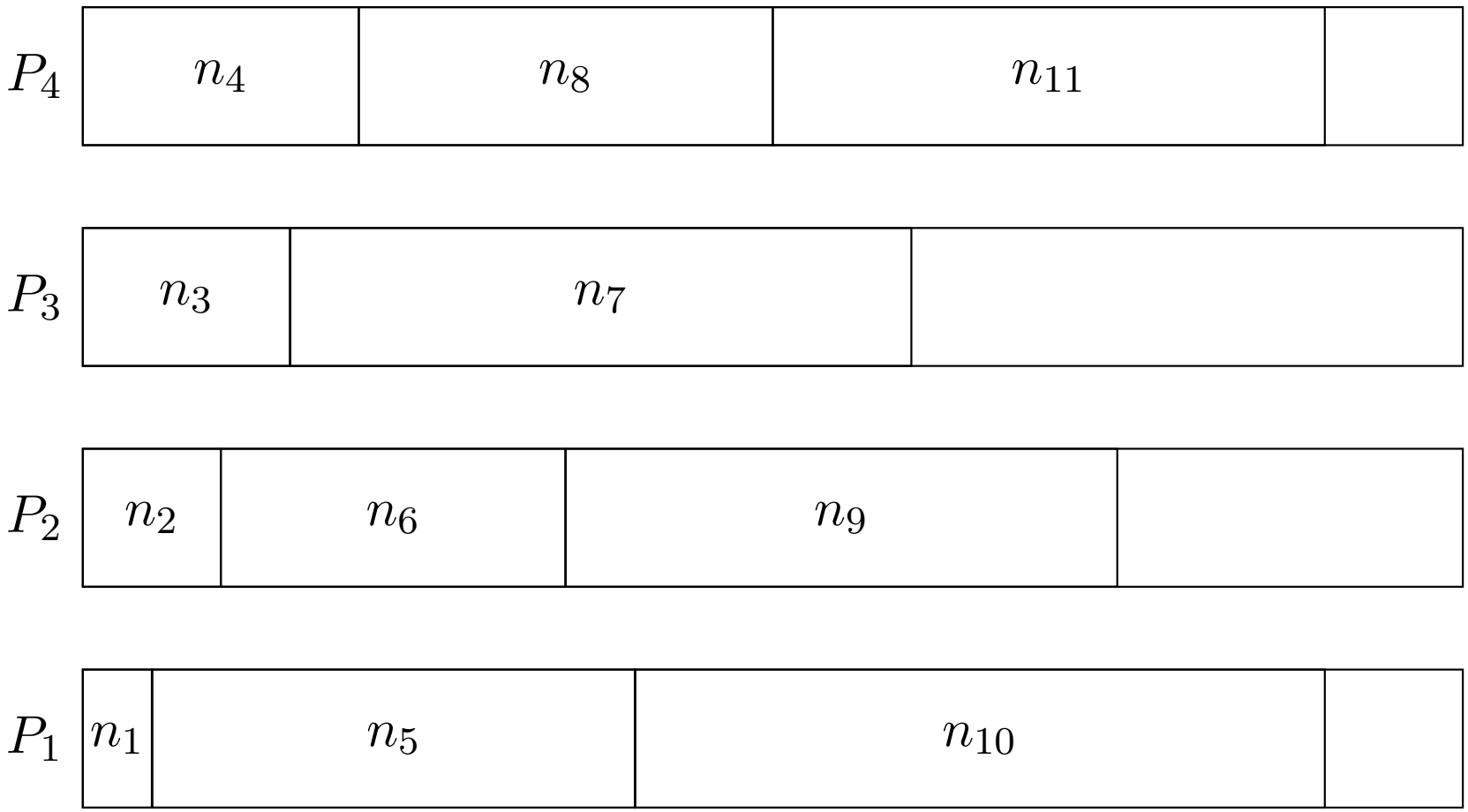
Παράδειγμα



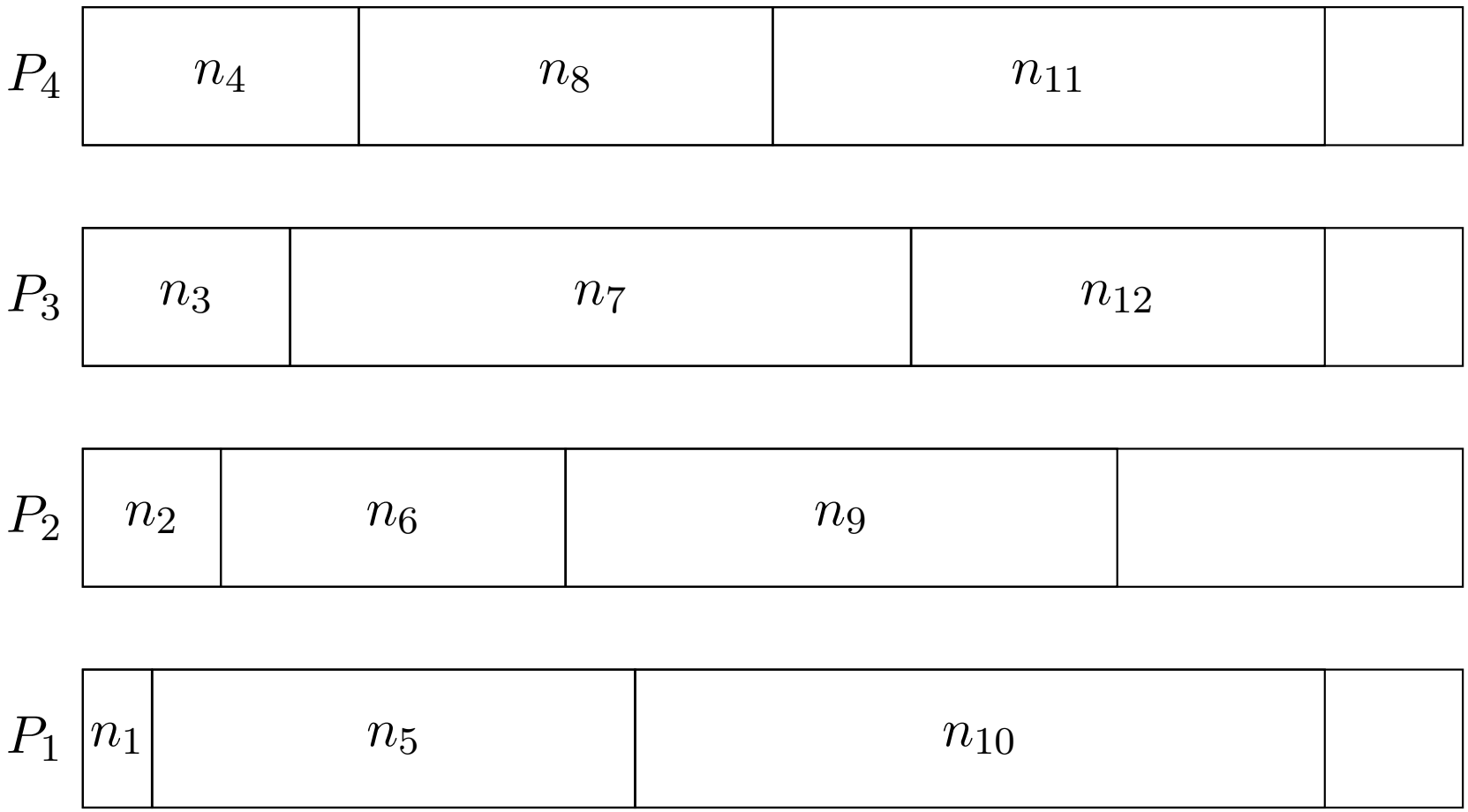
Παράδειγμα



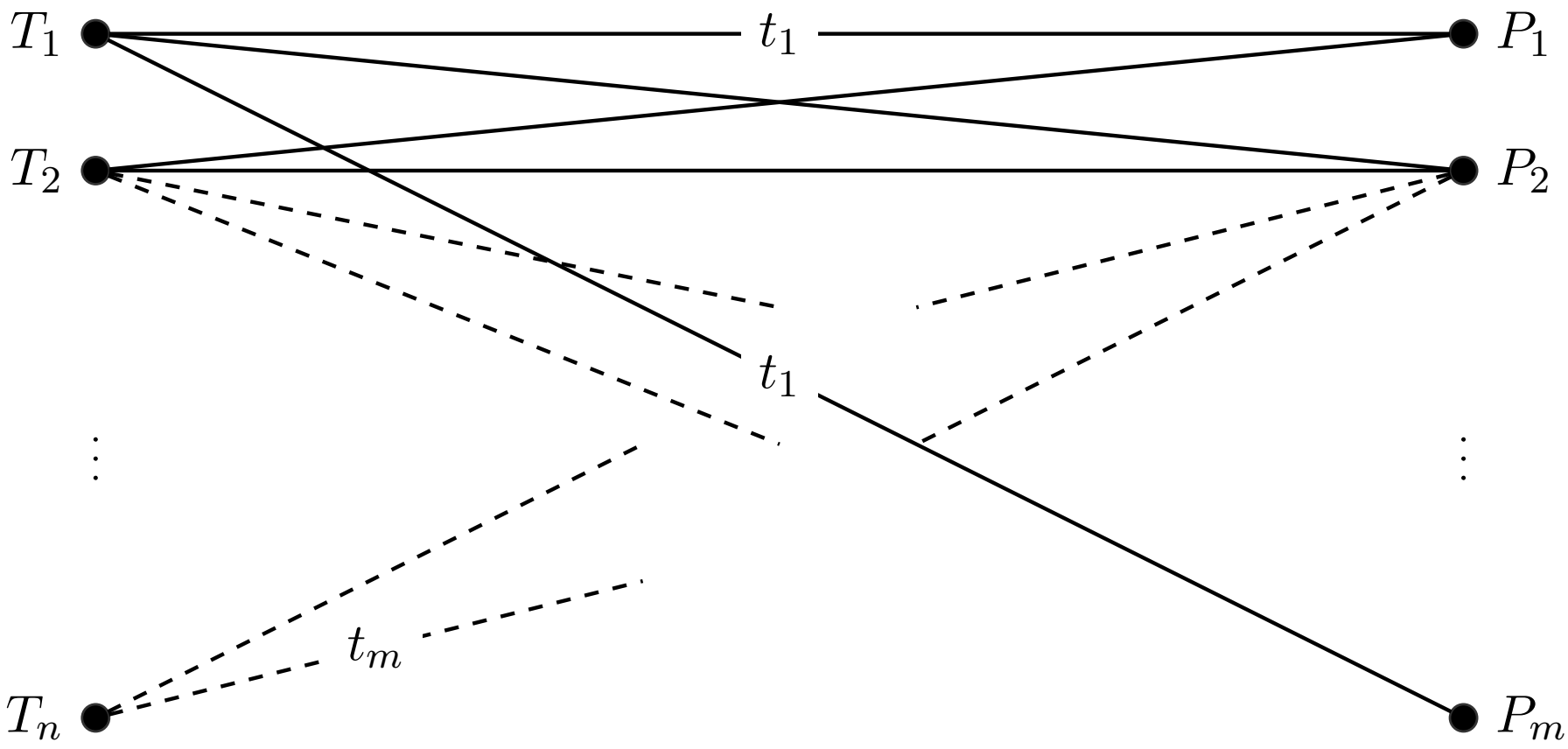
Παράδειγμα



Παράδειγμα



Identical Machines



M-Processor Scheduling

Γραμμικό πρόγραμμα

minimize: $c_{\max}$

subject to:
$$\sum_{i=1}^{n} t_i x_{ip} \leq c_{\max} \qquad \forall p = 1, \ldots, m$$
$$\sum_{p=1}^{m} x_{ip} = 1, \qquad \forall i = 1, \ldots, n$$
$$x_{ip} \in \{0, 1\}, \qquad \forall i = 1, \ldots, n \text{ and } p = 1, \ldots, m$$

List scheduling approximation ratio

- Στιγμιότυπο I.
- P_1 με τον μεγαλύτερο φόρτο.
- P_1 ολοκληρώνει σε χρόνο L.
- j η τελευταία εργασία που ανατίθεται στον P_1.
- $\forall P_k, k \neq 1,$ συνολικό φόρτο $\geq L - t_j$.
- Φόρτος του $P_1 = L - t_j$ πριν την ανάθεση της j

List scheduling approximation ratio

$$\sum_{i=1}^{n} t_i \geq m(L - t_j) + t_j \tag{1}$$

List scheduling approximation ratio

$$\sum_{i=1}^{n} t_i \geq m(L - t_j) + t_j \tag{1}$$

$$OPT(I) \geq \frac{\sum_{i=1}^{n} t_i}{m} \tag{2}$$

List scheduling approximation ratio

$$\sum_{i=1}^{n} t_i \geq m(L - t_j) + t_j \tag{1}$$

$$OPT(I) \geq \frac{\sum_{i=1}^{n} t_i}{m} \tag{2}$$

$$OPT(I) \geq (L - t_j) + \frac{t_j}{m} = A(I) - (1 - \frac{1}{m})t_j \tag{3}$$

List scheduling approximation ratio

$$\sum_{i=1}^{n} t_i \geq m(L - t_j) + t_j \tag{1}$$

$$OPT(I) \geq \frac{\sum_{i=1}^{n} t_i}{m} \tag{2}$$

$$OPT(I) \geq (L - t_j) + \frac{t_j}{m} = A(I) - (1 - \frac{1}{m})t_j \tag{3}$$

$$OPT(I) \geq t_j \tag{4}$$

List scheduling approximation ratio

$$\sum_{i=1}^{n} t_i \geq m(L - t_j) + t_j \tag{1}$$

$$OPT(I) \geq \frac{\sum_{i=1}^{n} t_i}{m} \tag{2}$$

$$OPT(I) \geq (L - t_j) + \frac{t_j}{m} = A(I) - (1 - \frac{1}{m})t_j \tag{3}$$

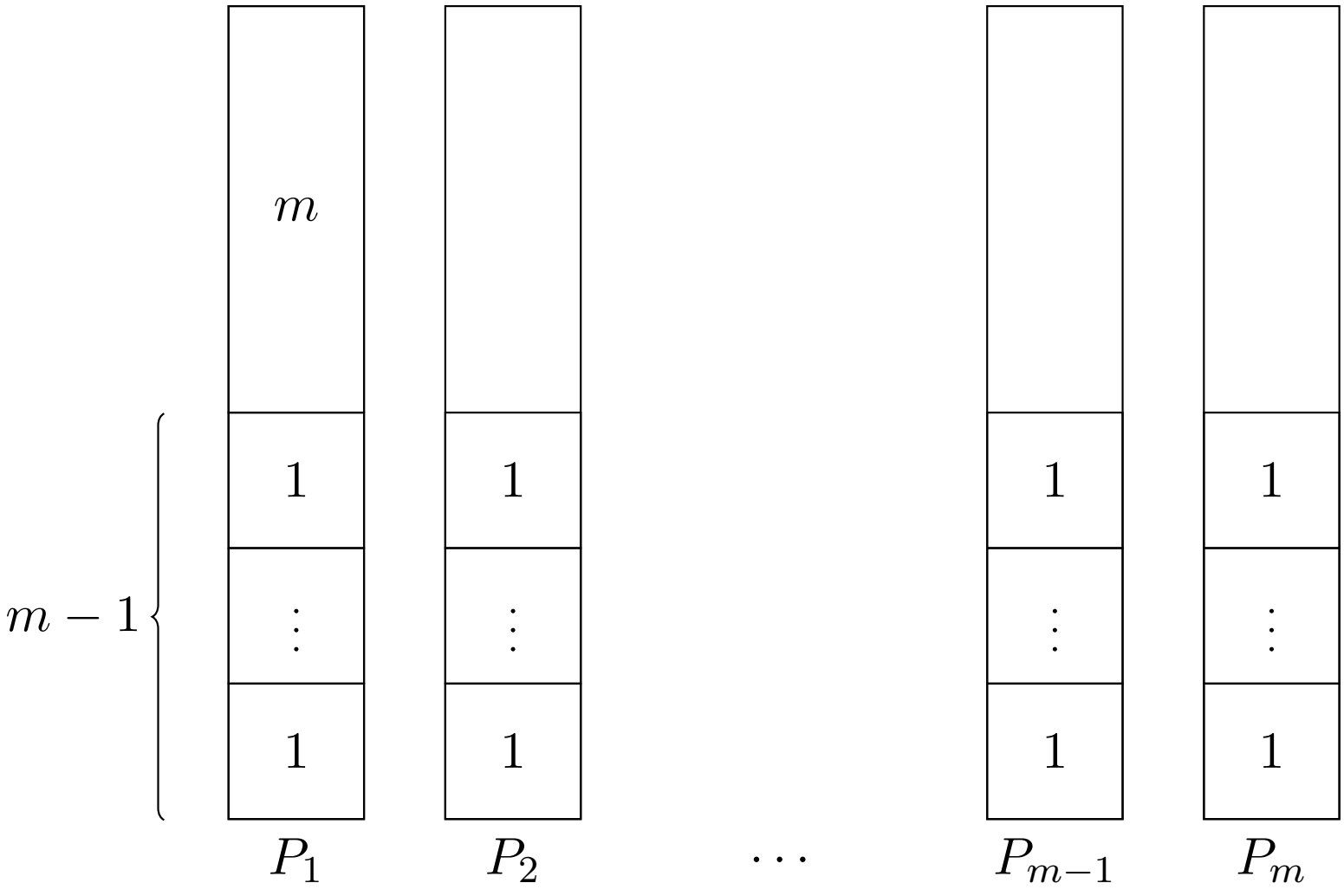
$$OPT(I) \geq t_j \tag{4}$$

$$\frac{A(I)}{OPT(I)} \leq 2 - \frac{1}{m}$$

The approximation is tight

- $n = m(m-1)+1$
- $t_1 = t_2 = \cdots = t_{n-1} = 1, t_n = m$
- $OPT(I) = m, A(I) = 2m-1$
- $\frac{A(I)}{OPT(I)} = 2 - \frac{1}{m}$

The approximation is tight



Minimum M-Processor Scheduling

- n εργασίες: $e_1, \ldots, e_n$.
- m επεξεργαστές: $P_1, \ldots, P_m$ (m σταθερό).
- Να βρεθεί η ανάθεση των εργασιών που ελαχιστοποιεί το makespan (διάρκεια εκτέλεσης χωρίς προεκχώρηση).
- Υπάρχει FPTAS. (FPTAS $\subseteq$ PTAS $\subseteq$ APX $\subseteq$ NPO)

Multifit (Coffman,Garey, Johnson)

Algorithm 3 Multifit M-Processor Scheduling

1: Διάταξε τις εργασίες σε φθίνουσα σειρά του χρόνου εκτέλεσης
2: Υπολόγισε τα κάτω και άνω φράγματα $c_{\min}$ και $c_{\max}$
3: **while** $c_{\min} == c_{\max}$ || να γίνει ορισμένο πλήθος επαναλήψεων **do**
4: $\quad c = \frac{c_{\min} + c_{\max}}{2}$
5: $\quad$ **for** $i = 1$ **to** n **do**
6: $\quad\quad$ ανάθεσε την i στον πρώτο διαθέσιμο επεξεργαστή
7: $\quad\quad$ **if** πλήθος χρησιμοποιούμενων επεξεργαστών $> m$ **then**
8: $\quad\quad\quad c_{\min} = c$
9: $\quad\quad$ **else**
10: $\quad\quad\quad c_{\max} = c$
11: **return** τα σύνολα εργασιών που ανατέθηκαν σε κάθε επεξεργαστή

Με επικοινωνία

- $T = \{T_1, \ldots, T_n\}$
- $P = \{P_1, \ldots, P_n\}$
- $G = (V, E)$
- c_{ij} (η T_i επικοινωνεί με την T_j)
- t_{ip} (χρόνος εκτέλεσης εργασίας i στον επεξεργαστή p)

Με επικοινωνία

Γραμμικό πρόγραμμα

minimize: $c_{\max}$

subject to:
$$\sum_{i=1}^{n} t_{ip} x_{ip} + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ip} (1 - x_{jp}) \leq c_{\max} \qquad \forall p = 1, \ldots, m$$

$$\sum_{p=1}^{m} x_{ip} = 1, \qquad \forall i = 1, \ldots, n$$

$$x_{ip} \in \{0, 1\}, \qquad \forall i = 1, \ldots, n \quad \forall p = 1, \ldots, m$$

Δέντρο Steiner

- $G = (V, E, M), M \subseteq V$ (Μ το σύνολο των Steiner κόμβων)
- Steiner Tree: $T = (V^{'}, E^{'}), M \subseteq T(V^{'}) \subseteq V, E^{'} \subseteq E$
- $W(E^{'})$ ελάχιστο
- $|M| = 2$ πολυωνυμικό
- $|M| = V$ πολυωνυμικό
- NP-Hard

DNH (Distance Network Heuristic)

Algorithm 4 Steiner Tree

1: Κατασκεύασε πλήρη γράφο $D_G(M) = (M, E', W)$ όπου $W(e) = d([u, v])$ και $e \in E'$ η ακμή μεταξύ u και v
2: Βρες ένα Minimum Spanning Tree T_1 στον $D_G(M)$
3: **for all** $e \in T_1$ **do**
4: αντικατέστησε την e με το συντομότερο μονοπάτι μεταξύ των άκρων της στον G. Έστω T_D ο γράφος που προκύπτει
5: Βρες ένα Minimum Spanning Tree T_2 στον υπογράφο του G που επάγεται από τον T_D
6: Κατασκεύασε το T_{DNH} από το T_2 διαγράφοντας τους κόμβους $u \in V - M$ με $d(u) = 1$
7: **return** το δέντρο T_{DNH}

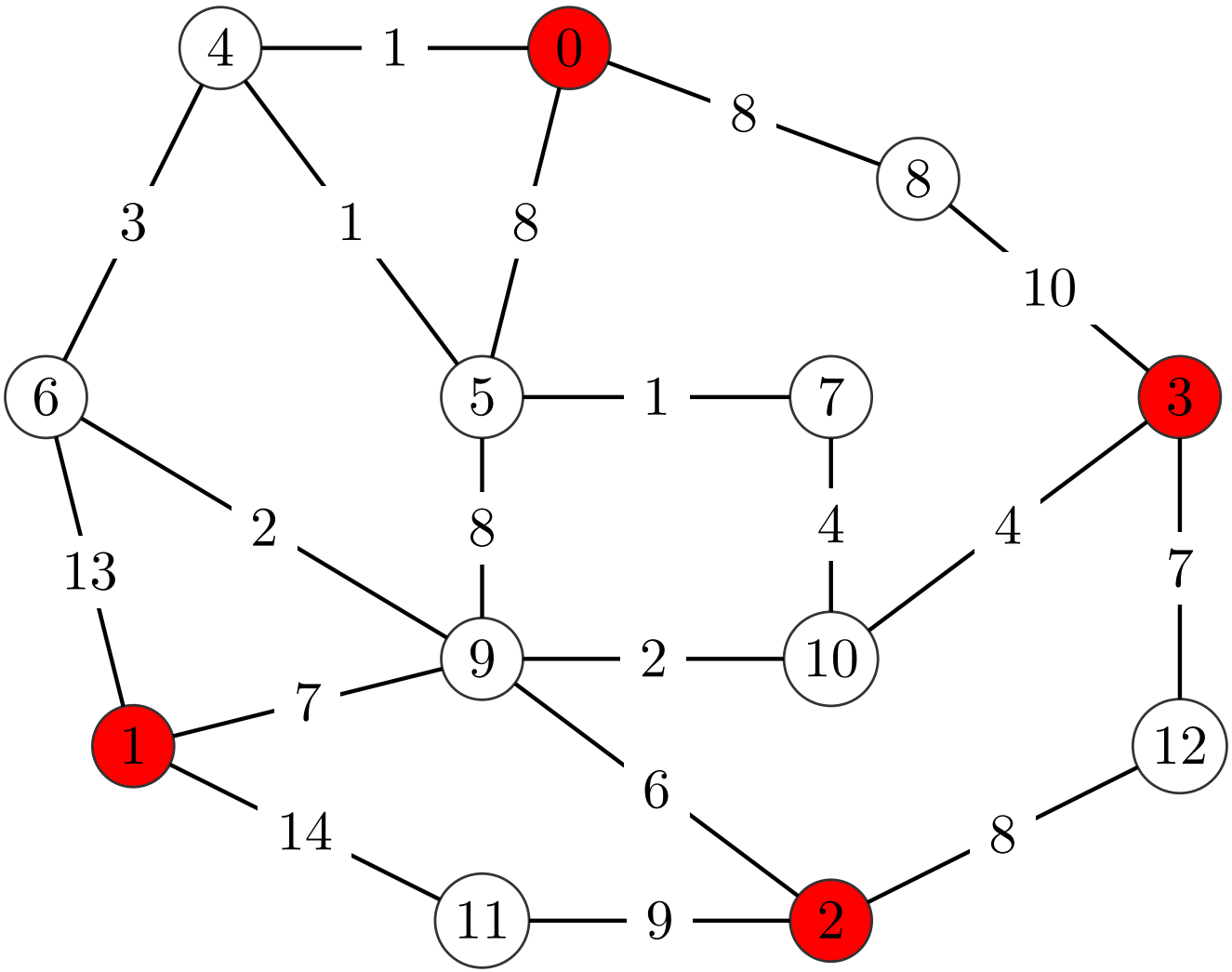
DNH (Distance Network Heuristic)

Algorithm 5 Steiner Tree

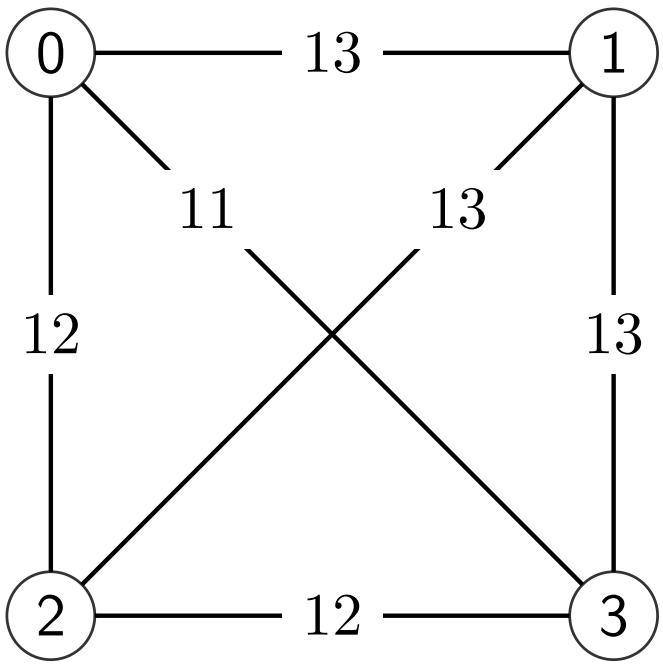
1: Κατασκεύασε πλήρη γράφο $D_G(M) = (M, E', W)$ όπου $W(e) = d([u, v])$ και $e \in E'$ η ακμή μεταξύ u και v
2: Βρες ένα Minimum Spanning Tree T_1 στον $D_G(M)$
3: **for all** $e \in T_1$ **do**
4: αντικατέστησε την e με το συντομότερο μονοπάτι μεταξύ των άκρων της στον G. Έστω T_D ο γράφος που προκύπτει
5: Βρες ένα Minimum Spanning Tree T_2 στον υπογράφο του G που επάγεται από τον T_D
6: Κατασκεύασε το T_{DNH} από το T_2 διαγράφοντας τους κόμβους $u \in V - M$ με $d(u) = 1$
7: **return** το δέντρο T_{DNH}

- Πολυπλοκότητα: $O(|M||V^2|)$, $O|M|(|E| + |V|log|V|)$ με Fibonacci heaps.
- 2-προσεγγιστικός.
- 5% στην πράξη.

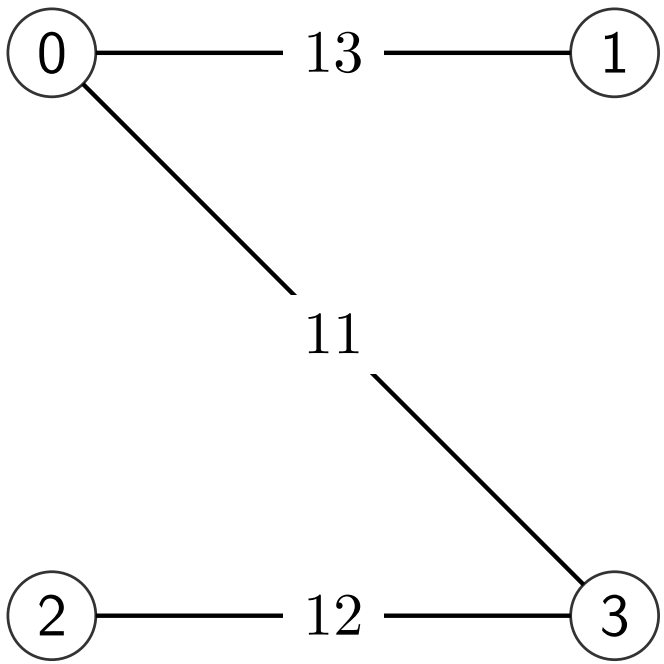
Παράδειγμα



Παράδειγμα



(a) Graph $D_G(M)$



(b) Spannign Tree T_1

Παράδειγμα

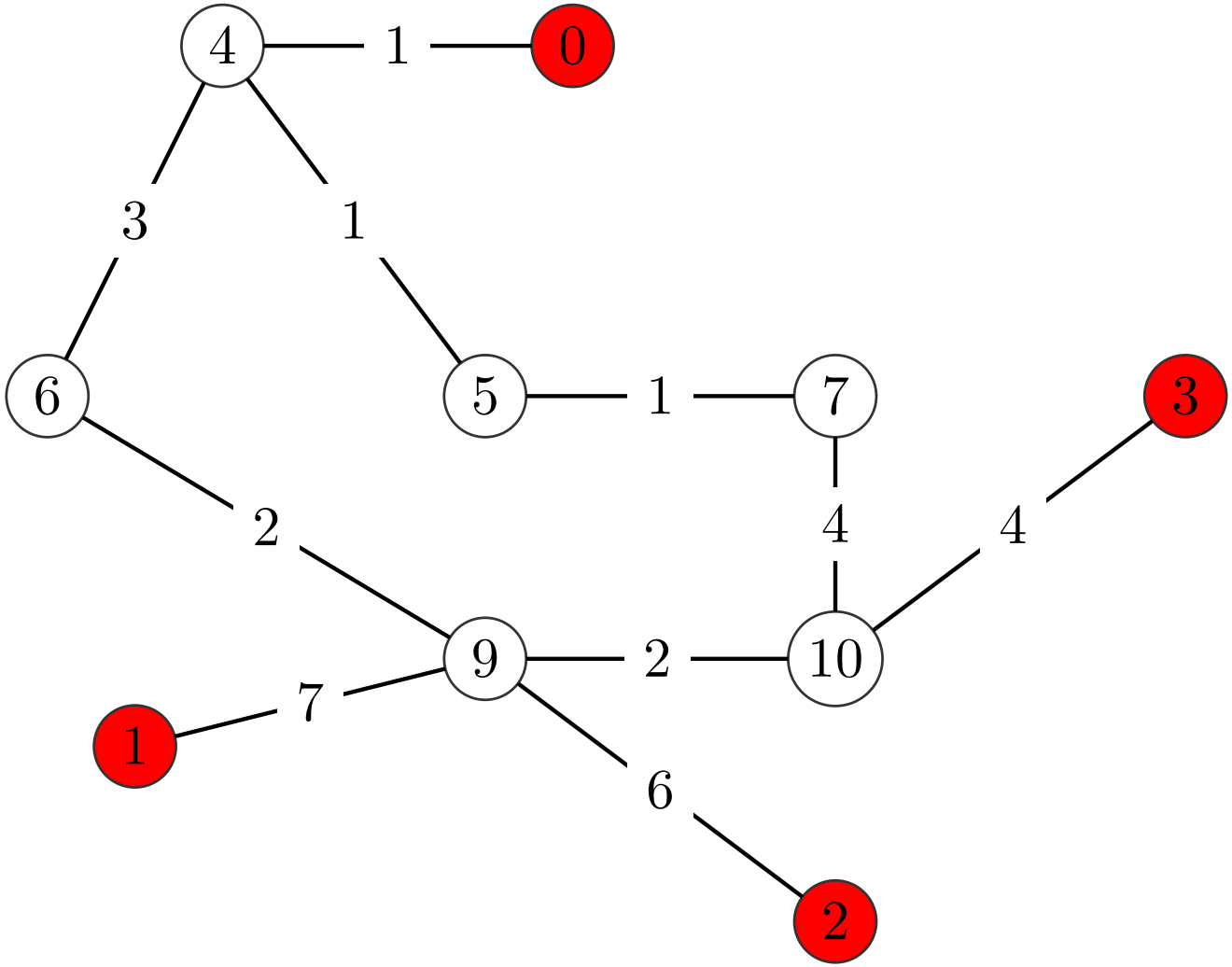


Figure: 0 4 6 9 1 / 0 4 5 7 10 3 / 3 10 9 2

Παράδειγμα

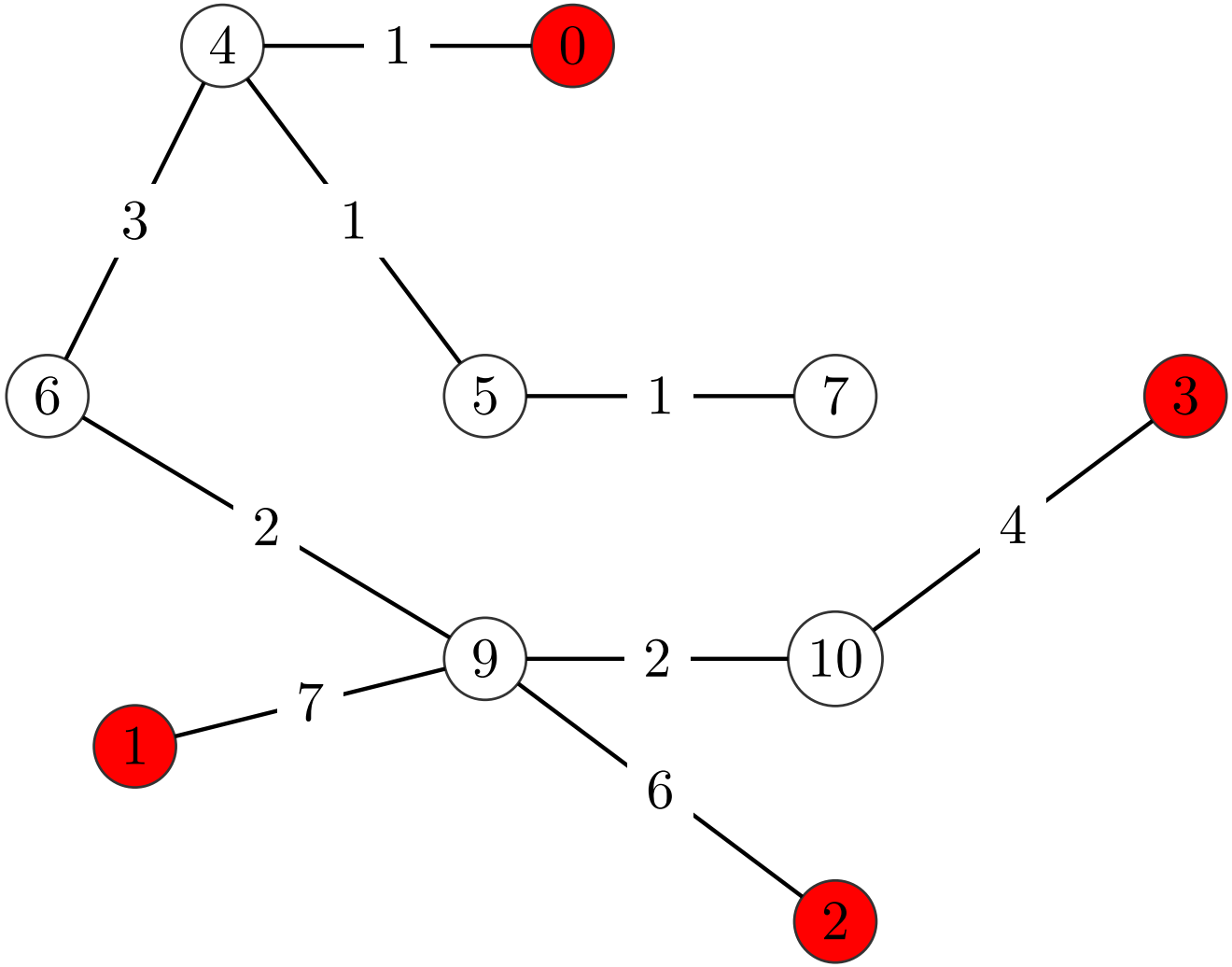


Figure: MST T_2

Παράδειγμα

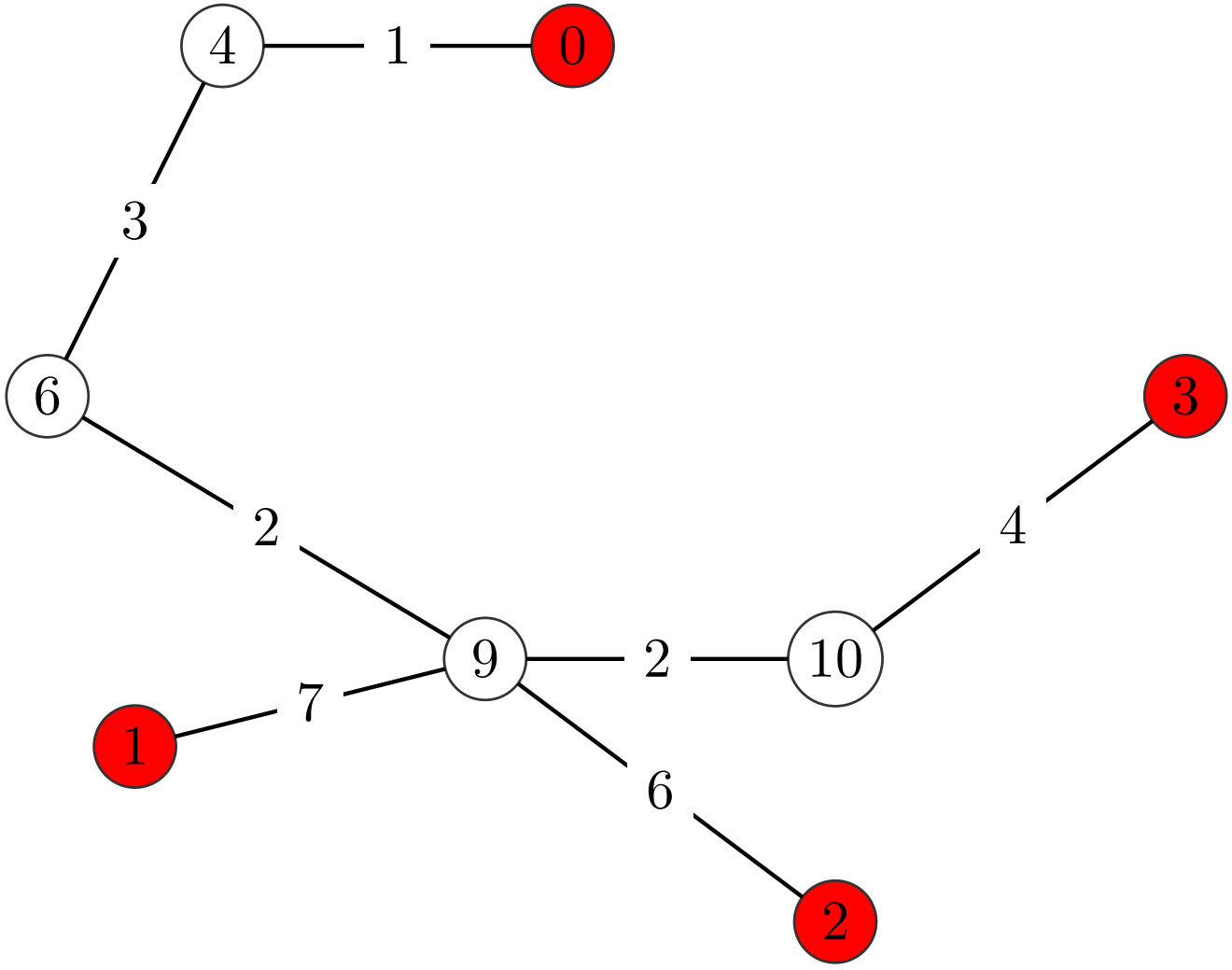


Figure: Final Tree T_{DNH}