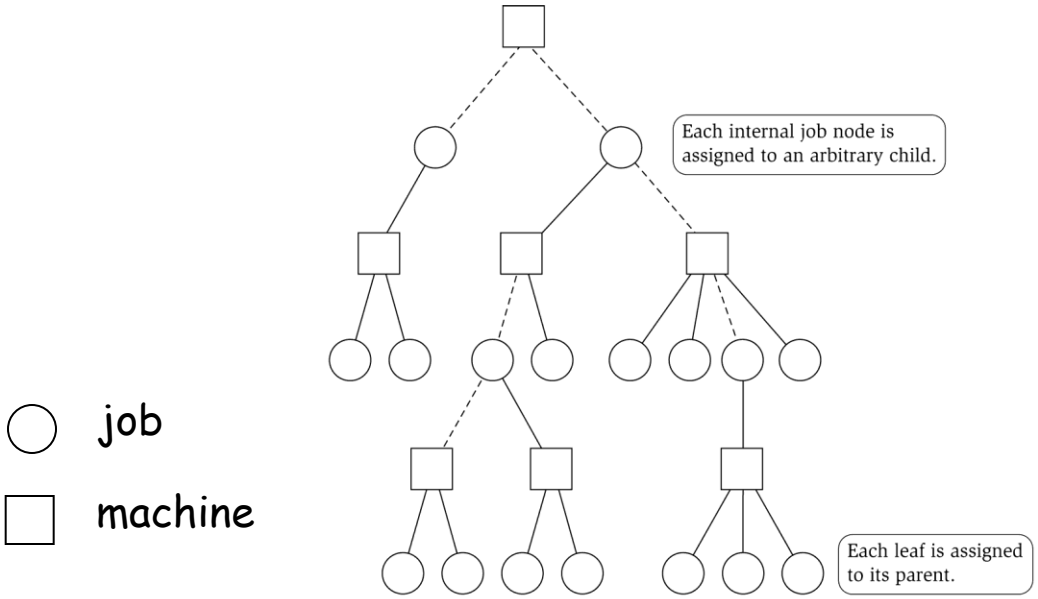
Generalized Load Balancing: Analysis

Lemma 5. If job j is a leaf node and machine i = parent(j), then $x_{ij} = t_j$.

Pf. Since i is a leaf, $x_{ij} = 0$ for all $j \neq$ parent(i). LP constraint guarantees $\Sigma_i x_{ij} = t_j$. ▪

Lemma 6. At most one non-leaf job is assigned to a machine.

Pf. The only possible non-leaf job assigned to machine i is parent(i). ▪



Generalized Load Balancing: Analysis

Theorem. Rounded solution is a 2-approximation.

Pf.

- Let J(i) be the jobs assigned to machine i.
- By Lemma 6, the load L_i on machine i has two components:

- leaf nodes

Lemma 5 ↓ LP ↓ Lemma 1 (LP is a relaxation) ↓

$$\sum_{\substack{j \in J(i) \\ j \text{ is a leaf}}} t_j = \sum_{\substack{j \in J(i) \\ j \text{ is a leaf}}} x_{ij} \leq \sum_{j \in J} x_{ij} \leq L \leq L^*$$

↑ optimal value of LP

- parent(i)

Lemma 2 ↓

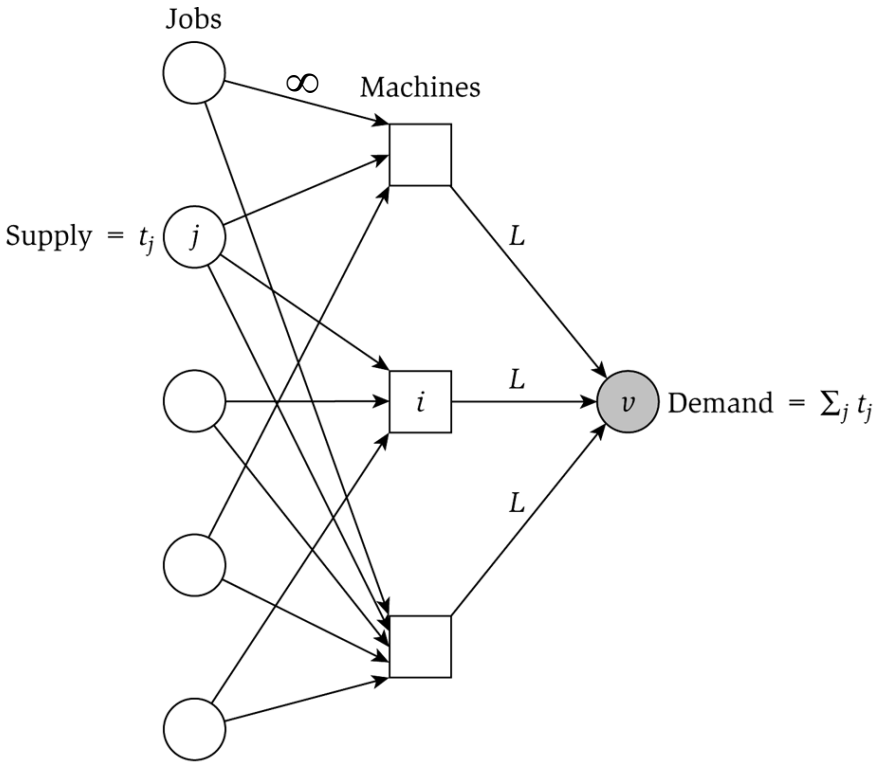
$$t_{\text{parent}(i)} \leq L^*$$

- Thus, the overall load $L_i \leq 2L^*$. ▪

Generalized Load Balancing: Flow Formulation

Flow formulation of LP.

$$\begin{aligned}
\sum_i x_{ij} &= t_j && \text{for all } j \in J \\
\sum_j x_{ij} &\le L && \text{for all } i \in M \\
x_{ij} &\ge 0 && \text{for all } j \in J \text{ and } i \in M_j \\
x_{ij} &= 0 && \text{for all } j \in J \text{ and } i \notin M_j
\end{aligned}$$



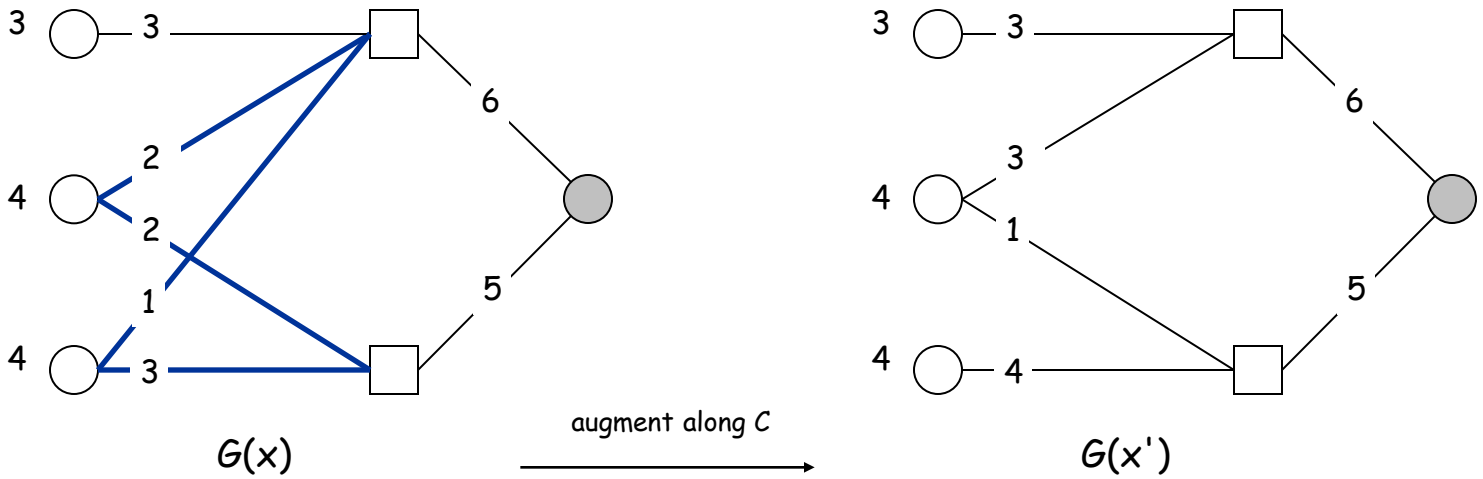
Observation. Solution to feasible flow problem with value L are in one-to-one correspondence with LP solutions of value L.

Generalized Load Balancing: Structure of Solution

Lemma 3. Let (x, L) be solution to LP. Let G(x) be the graph with an edge from machine i to job j if $x_{ij} > 0$. We can find another solution (x', L) such that G(x') is acyclic.

Pf. Let C be a cycle in G(x).

- Augment flow along the cycle C. ← flow conservation maintained
- At least one edge from C is removed (and none are added).
- Repeat until G(x') is acyclic.



Conclusions

Running time. The bottleneck operation in our 2-approximation is solving one LP with mn + 1 variables.

Remark. Can solve LP using flow techniques on a graph with m+n+1 nodes: given L, find feasible flow if it exists. Binary search to find L*.

Extensions: unrelated parallel machines. [Lenstra-Shmoys-Tardos 1990]

- Job j takes t_{ij} time if processed on machine i.
- 2-approximation algorithm via LP rounding.
- No 3/2-approximation algorithm unless P = NP.

11.8 Knapsack Problem

Polynomial Time Approximation Scheme

PTAS. $(1 + \varepsilon)$-approximation algorithm for any constant $\varepsilon > 0$.

- Load balancing. [Hochbaum-Shmoys 1987]
- Euclidean TSP. [Arora 1996]

Consequence. PTAS produces arbitrarily high quality solution, but trades off accuracy for time.

This section. PTAS for knapsack problem via rounding and scaling.

Knapsack Problem

Knapsack problem.

- Given n objects and a "knapsack."
- Item i has value $v_i > 0$ and weighs $w_i > 0$. ⟵ we'll assume $w_i \leq W$
- Knapsack can carry weight up to W.
- Goal: fill knapsack so as to maximize total value.

Ex: { 3, 4 } has value 40.

W = 11

Item	Value	Weight
1	1	1
2	6	2
3	18	5
4	22	6
5	28	7

Knapsack is NP-Complete

KNAPSACK: Given a finite set X, nonnegative weights w_i, nonnegative values v_i, a weight limit W, and a target value V, is there a subset $S \subseteq X$ such that:

$$\sum_{i \in S} w_i \leq W$$
$$\sum_{i \in S} v_i \geq V$$

SUBSET-SUM: Given a finite set X, nonnegative values u_i, and an integer U, is there a subset $S \subseteq X$ whose elements sum to exactly U?

Claim. SUBSET-SUM $\leq_P$ KNAPSACK.

Pf. Given instance $(u_1, \ldots, u_n, U)$ of SUBSET-SUM, create KNAPSACK instance:

$$v_i = w_i = u_i \qquad \sum_{i \in S} u_i \leq U$$
$$V = W = U \qquad \sum_{i \in S} u_i \geq U$$

Knapsack Problem: Dynamic Programming 1

Def. OPT(i, w) = max value subset of items 1,..., i with weight limit w.

- Case 1: OPT does not select item i.
  - OPT selects best of 1, ..., i–1 using up to weight limit w
- Case 2: OPT selects item i.
  - new weight limit = $w - w_i$
  - OPT selects best of 1, ..., i–1 using up to weight limit $w - w_i$

$$OPT(i,w) = \begin{cases} 0 & \text{if } i = 0 \\ OPT(i-1,w) & \text{if } w_i > w \\ \max\{\, OPT(i-1,w),\ \ v_i + OPT(i-1, w-w_i)\,\} & \text{otherwise} \end{cases}$$

Running time. O(n W).

- W = weight limit.
- Not polynomial in input size!

Knapsack Problem: Dynamic Programming II

Def. OPT(i, v) = min weight subset of items 1, …, i that yields value exactly v.

- Case 1: OPT does not select item i.
  - OPT selects best of 1, …, i-1 that achieves exactly value v
- Case 2: OPT selects item i.
  - consumes weight w_i, new value needed = v – v_i
  - OPT selects best of 1, …, i-1 that achieves exactly value v

$$OPT(i,v) = \begin{cases} 0 & \text{if } v = 0 \\ \infty & \text{if } i = 0,\ v > 0 \\ OPT(i-1,v) & \text{if } v_i > v \\ \min\{OPT(i-1,v),\ \ w_i + OPT(i-1, v-v_i)\} & \text{otherwise} \end{cases}$$

$V^* \le n\, v_{max}$

Running time. $O(n V^*) = O(n^2 v_{max})$.

- V^* = optimal value = maximum v such that $OPT(n, v) \le W$.
- Not polynomial in input size!

Knapsack: FPTAS

Intuition for approximation algorithm.

- Round all values up to lie in smaller range.
- Run dynamic programming algorithm on rounded instance.
- Return optimal items in rounded instance.

Item	Value	Weight
1	934,221	1
2	5,956,342	2
3	17,810,013	5
4	21,217,800	6
5	27,343,199	7

W = 11

original instance

Item	Value	Weight
1	1	1
2	6	2
3	18	5
4	22	6
5	28	7

W = 11

rounded instance

Knapsack: FPTAS

Knapsack FPTAS. Round up all values: $\overline{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil \theta, \quad \hat{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil$

- v_{max} = largest value in original instance
- ε = precision parameter
- θ = scaling factor = $\varepsilon\, v_{max} / n$

Observation. Optimal solution to problems with $\overline{v}$ or $\hat{v}$ are equivalent.

Intuition. $\overline{v}$ close to v so optimal solution using $\overline{v}$ is nearly optimal; $\hat{v}$ small and integral so dynamic programming algorithm is fast.

Running time. $O(n^3 / \varepsilon)$.

- Dynamic program II running time is $O(n^2 \hat{v}_{max})$, where

$$\hat{v}_{max} = \left\lceil \frac{v_{max}}{\theta} \right\rceil = \left\lceil \frac{n}{\varepsilon} \right\rceil$$

Knapsack: FPTAS

Knapsack FPTAS. Round up all values: $\overline{v}_i = \left\lceil \frac{v_i}{\theta} \right\rceil \theta$

Theorem. If S is solution found by our algorithm and S* is any other feasible solution then $(1+\varepsilon)\sum_{i \in S} v_i \geq \sum_{i \in S^*} v_i$

Pf. Let S* be any feasible solution satisfying weight constraint.

$$\begin{aligned}
\sum_{i \in S^*} v_i &\leq \sum_{i \in S^*} \overline{v}_i && \text{always round up} \\
&\leq \sum_{i \in S} \overline{v}_i && \text{solve rounded instance optimally} \\
&\leq \sum_{i \in S} (v_i + \theta) && \text{never round up by more than } \theta \\
&\leq \sum_{i \in S} v_i + n\theta && |S| \leq n \\
&\leq (1+\varepsilon)\sum_{i \in S} v_i && n\theta = \varepsilon v_{max},\ v_{max} \leq \Sigma_{i \in S} v_i
\end{aligned}$$

DP alg can take v_{max}

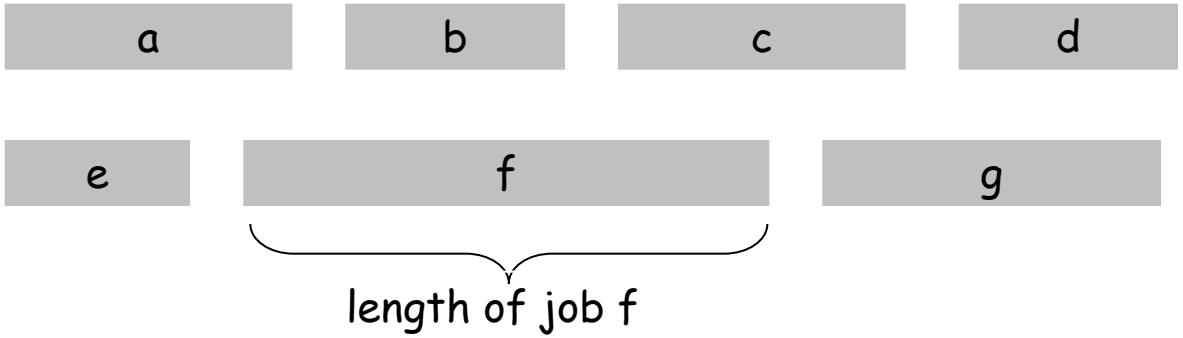
Extra Slides

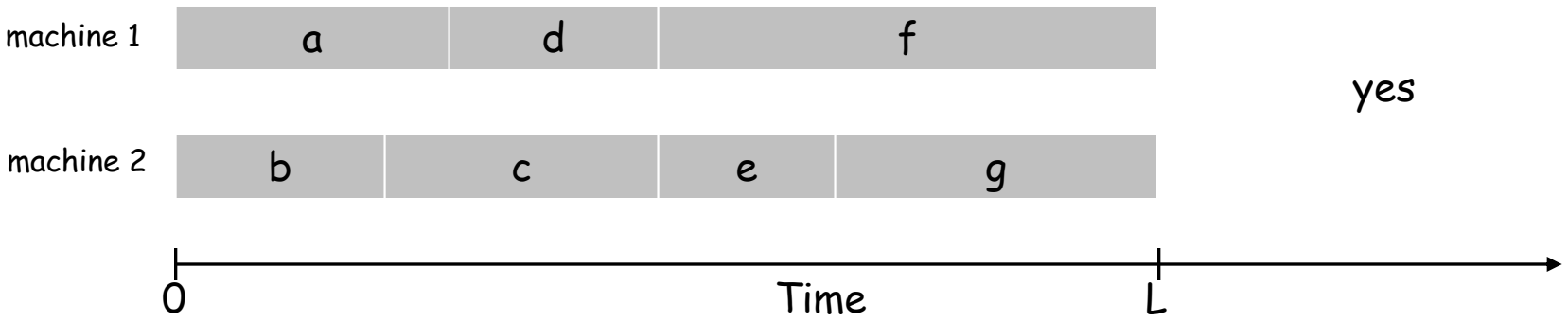
Load Balancing on 2 Machines

Claim. Load balancing is hard even if only 2 machines.

Pf. NUMBER-PARTITIONING $\leq_p$ LOAD-BALANCE.

NP-complete by Exercise 8.26





yes

Center Selection: Hardness of Approximation

Theorem. Unless P = NP, there is no ρ-approximation algorithm for metric k-center problem for any $\rho < 2$.

Pf. We show how we could use a $(2 - \varepsilon)$ approximation algorithm for k-center to solve DOMINATING-SET in poly-time.

- Let $G = (V, E)$, k be an instance of DOMINATING-SET. ⟵ see Exercise 8.29
- Construct instance G' of k-center with sites V and distances
  - $d(u, v) = 2$ if $(u, v) \in E$
  - $d(u, v) = 1$ if $(u, v) \notin E$
- Note that G' satisfies the triangle inequality.
- Claim: G has dominating set of size k iff there exists k centers C^* with $r(C^*) = 1$.
- Thus, if G has a dominating set of size k, a $(2 - \varepsilon)$-approximation algorithm on G' must find a solution C^* with $r(C^*) = 1$ since it cannot use any edge of distance 2.

Ανοικτά Ακαδημαϊκά Μαθήματα

Πανεπιστήμιο Ιωαννίνων

Τέλος Ενότητας

Χρηματοδότηση

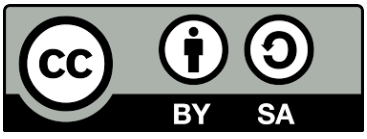
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Ιωαννίνων**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημειώματα

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά Δημιουργού - Παρόμοια Διανομή, Διεθνής Έκδοση 4.0 [1] ή μεταγενέστερη.



[1] https://creativecommons.org/licenses/by-sa/4.0/.