Approximation Algorithms

Ruben Hoeksma (University of Bremen)

April 23, 2018

Local search algorithms

Example I: Revisiting scheduling on identical parallel machines

Local search

Greedy algorithms

Construct a solution from nothing by repeatedly making "greedy choices".

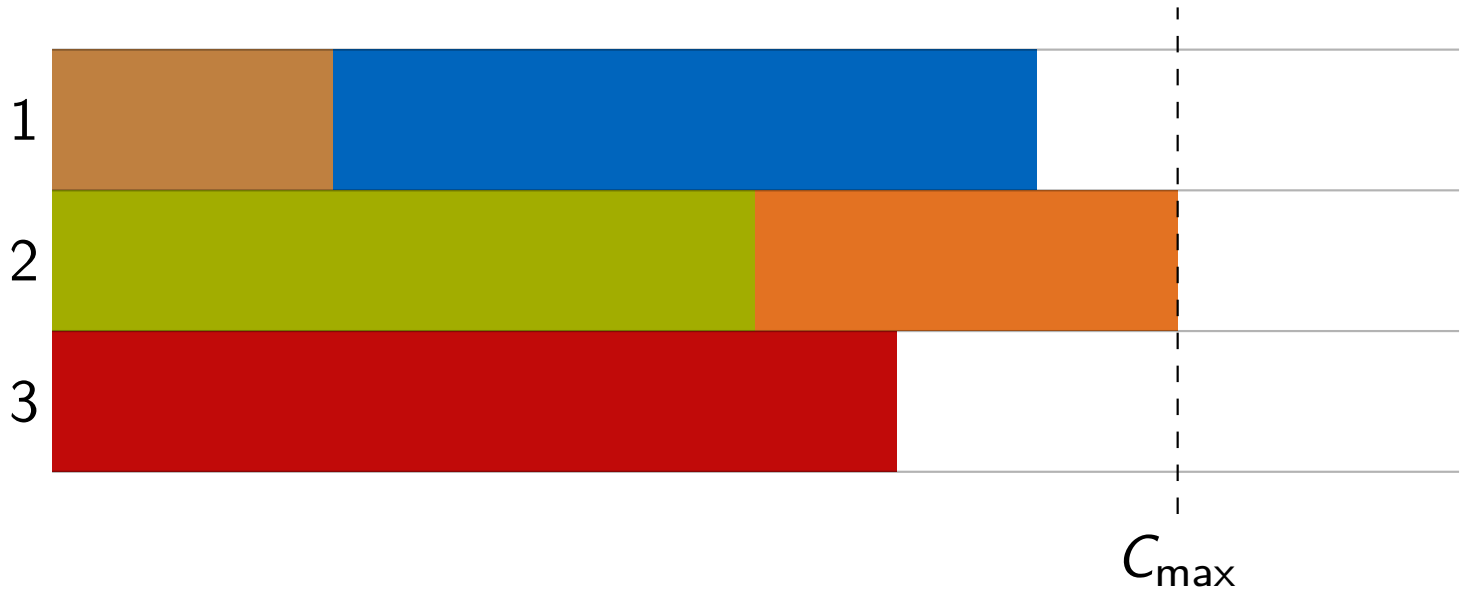
Local search algorithms

Start with any feasible solution and repeatedly improve that solution by searching for a "nearby" (local) better solution.

Scheduling on parallel machines

Input: m identical machines,
n jobs with processing times $p_1, \ldots, p_n$

Task: assign each job $j \in [n]$ to a machine $i \in [m]$
minimizing the maximum load of any machine
$C_{\max} := \max_{i \in [m]} \sum_{j \,:\, j \to i} p_j$



Theorem

(Greedy) List Scheduling is a 2-approximation for $P||C_{\max}$.

Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.

Theorem

Local search is a 2-approximation for $P||C_{\max}$.

Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.

Theorem

Local search is a 2-approximation for $P||C_{\max}$.

Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.

Theorem

Local search is a 2-approximation for $P||C_{\max}$.

Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.

Theorem

Local search is a 2-approximation for $P||C_{\max}$.

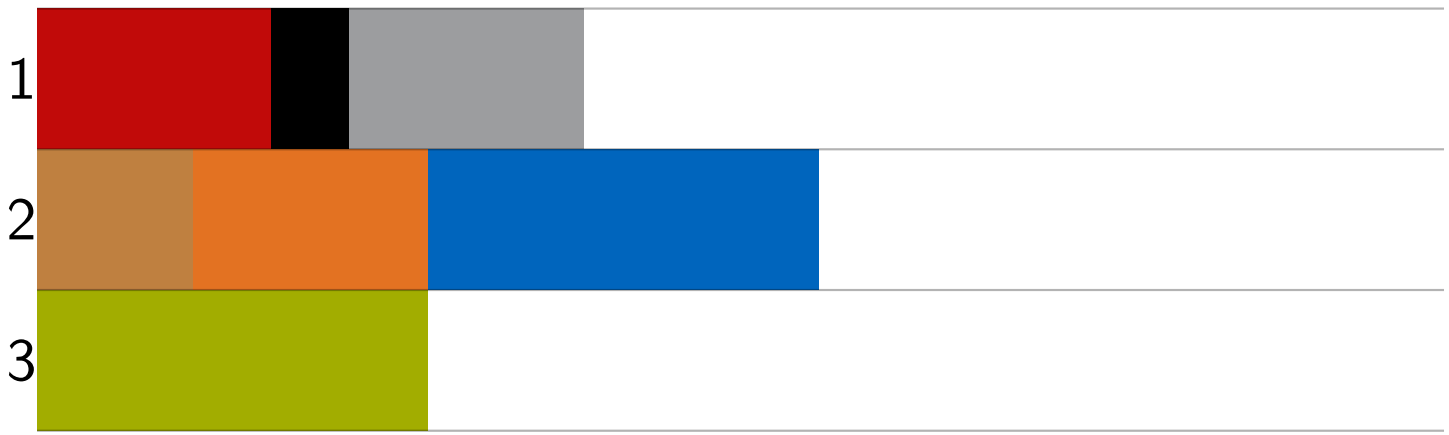
Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.



Theorem

Local search is a 2-approximation for $P||C_{\max}$.

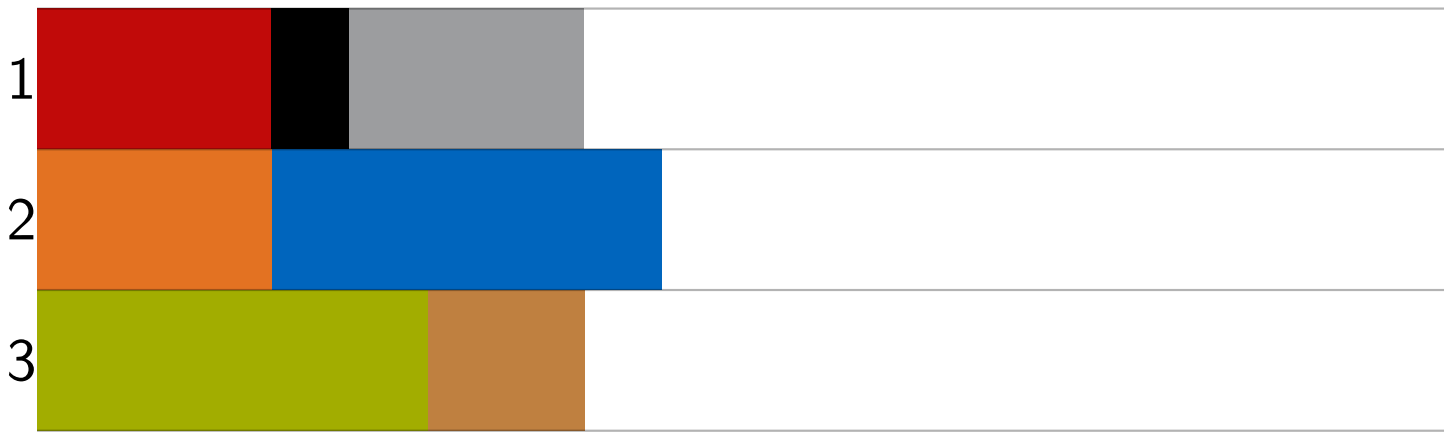
Local search

Algorithm:

1. Init. any feasible solution.
2. While $\exists j \in [n], i \in [m]$ s.t.

$$p_j + \sum_{k\,:\,k \to i} p_k < \sum_{k\,:\,k \to M_j} p_k$$

Assign j to machine i.



Theorem

Local search is a 2-approximation for $P||C_{\max}$.

Local search

Theorem

Local search is a 2-approximation for $P||C_{\max}$.

Proof

Slightly different local moves.

- Move only the last job on a machine.
- Move only to the machine with the smallest load.

Local search algorithms

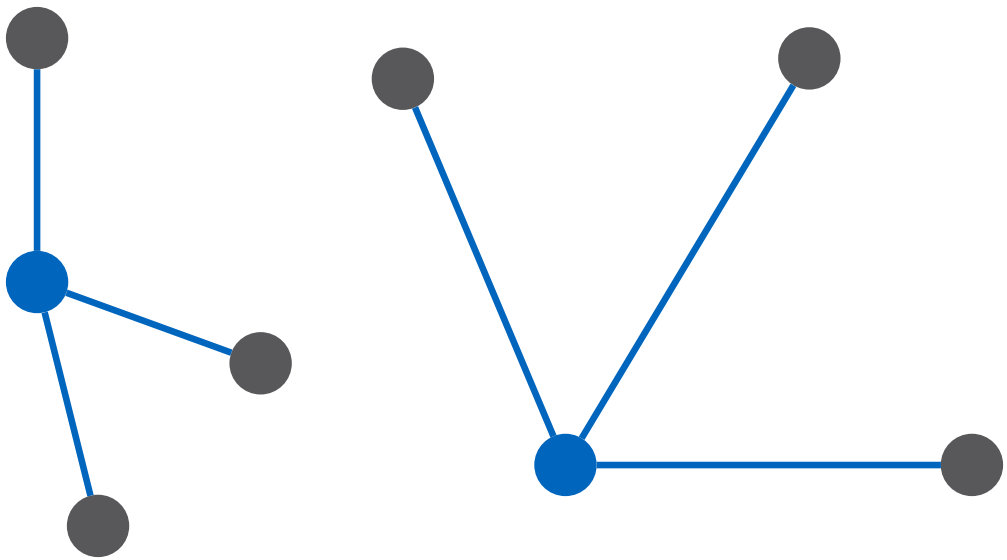
Example II: k-Median problem

The k-Median problem

Input: metric space V with distances d_{ij} for $i, j \in V$ and some $k \in \mathbb{N}$ (metric: $d_{ii} = 0, d_{ij} = d_{ji}$, and triangle ineq. $d_{ij} + d_{jk} \geq d_{ik}$)

Task: find k facilities in V, i.e., find $S \subseteq V$ with $|S| \leq k$.

Objective: minimize $\sum_{i \in V} d(i, S)$ with $d(i, S) = \min_{s \in S} d(i, s)$



Local search

Algorithm:

1. Init. any set S of k facilities.
2. While $\exists j \in V, k \in S$ s.t.

$$\sum_{i \in V} d(i, S + j - k) < \sum_{i \in V} d(i, S)$$

Swap j and k.

Theorem

Local search is a 5-approximation for k-Median.

Proof idea

S: Local optimum

S^*: Optimal solution

In S, find k swaps that imply the bound.

Mapping $\sigma : S^* \to S$: Each $i^* \in S^*$ to closest $i \in S$