Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines

Minimum Bin Packing

Minimum Graph Coloring
Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines

Minimum Bin Packing

Minimum Graph Coloring
Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines
Minimum Bin Packing
Minimum Graph Coloring

Problem

Problem definition

**INSTANCE:** set $T$ of jobs, number $p$ of machines, time $l_j$ for executing $j$-th job

**SOLUTION:** a $p$-schedule for $T$, i.e., a function $f : T \rightarrow [1, \ldots, p]$

**COST:** *makespan*, defined as

$$\max_{i \in [1, \ldots, p]} \sum_{j \in T : f(j) = i} l_j$$
Sequential algorithms for partitioning problems

Want to partition a set $I$ of items (in our case the set $T$ of jobs) according to some criterion (in our case, minimum makespan)

1. Items ordered according to some criterion
2. Each item sequentially allocated to existing or new partition according to this order

**List Scheduling (LS) algorithm**

1. Consider any sorting of the jobs (possibly the on-line arrival order)
2. When serving the $j$-th job, consider, for every $i$, the current load on the $i$-th machine, defined as $A_i(j-1) = \sum_{1 \leq k \leq j-1: f(k) = i} l_k$
3. Allocate the $j$-th job to any machine $i$ such that $A_i(j-1)$ is minimum
Considered any instance $x$ of the Machine Scheduling problem, algorithm LS computes a solution such that:

$$m_{LS}(x) \leq \left(2 - \frac{1}{p}\right) m^*(x)$$

Proof

Required. See [1, Section 2.2]
Is this analysis tight?

Yes, below an example...

There exist instances of the Machine Scheduling problem for which the performance of LS is \( \left( 2 - \frac{1}{p} \right) \) times worse than the optimal performance.

Picture taken from [1, Section 2.2]
A better performing algorithm

Longest Processing Time first heuristic (LPT)

1. Jobs ordered according to decreasing size
2. When serving the $j$-th job, consider, for every $i$, the current load on the $i$-th machine, defined as
   \[ A_i(j - 1) = \sum_{1 \leq k \leq j-1 : f(k) = i} l_k \]
3. Allocate the $j$-th job to any machine $i$ such that $A_i(j - 1)$ is minimum

Theorem

Considered any instance $x$ of the Machine Scheduling problem, algorithm LPT computes a solution such that:

\[ m_{LS}(x) \leq \left( \frac{4}{3} - \frac{1}{3p} \right) m^*(x) \]

Proof

Required. See [1, Section 2.2]
Question

Differences between LS and LPT

Are there settings in which LS can be used and LPT not?
Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines

Minimum Bin Packing

Minimum Graph Coloring
Problem definition

**INSTANCE:** set of $n$ items, $i$-th item has size $a_i \in (0, 1)$

**SOLUTION:** a partition $\{B_1, \ldots, B_k\}$ such that

$$\sum_{i \in B_j} a_i \leq 1, \quad \forall j = 1, \ldots, k$$

**COST:** cardinality of the partition, i.e., $k$
A first algorithm

Next Fit (NF)

1. First item of size $a_1$ placed into bin $B_1$
2. For the $i$-th item ($i > 1$): let $B_j$ the last used bin when NF considers the $i$-th item; NF assigns the item to $B_j$ if this bin has enough space, otherwise it assigns the item to a new bin $B_{j+1}$

Theorem

Considered any instance $x$ of the Bin Packing problem, algorithm NF computes a solution such that:

$$m_{NF}(x) \leq 2m^*(x)$$

Proof

Required. See [1, Section 2.2]
Is this analysis tight?

Yes, below an example...

There exist instances of the Bin Packing problem for which the performance of NF is 2 times worse than the optimal performance.

Picture taken from [1, Section 2.2]
First Fit (Decreasing)

Obvious weakness of NF: only last bin considered

1. (Sort items according to non-increasing size)
2. For the $i$-th item ($i > 1$): assign to the first used bin that has enough space, otherwise open a new bin

If no sorting performed (step 1) then we have First Fit

**Theorem**

*Considered any instance $x$ of the Bin Packing problem, algorithm FFD computes a solution such that:*

$$m_{FFD}(x) \leq 1.5m^*(x) + 1$$

**Proof**

*Required. See [1, Section 2.2]*
How far is FFD from optimum?

An almost tight example...

There exist instances of the **Bin Packing** problem for which the performance of FFD is $11/9$ times away from optimum.

Picture taken from [1, Section 2.2]
Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines
Minimum Bin Packing
Minimum Graph Coloring

**Question**

**Differences between NF and FFD**

Are there settings in which NF can be used and FFD not?
Sequential algorithms for partitioning problems

L. Becchetti

1. Minimum Scheduling on Identical Machines
2. Minimum Bin Packing
3. Minimum Graph Coloring
Problem definition

**INSTANCE:** graph $G = (V, E)$

**SOLUTION:** assignment $f : V \rightarrow \{1, \ldots, K\}$ of $K$ colors to the vertices such that $\forall (u, v) \in E : f(u) \neq f(v)$

**COST:** number of colors used, i.e., $K$
Sequential algorithms for partitioning problems

L. Becchetti

Minimum Scheduling on Identical Machines

Minimum Bin Packing

Minimum Graph Coloring

**Generic sequential algorithm**

*Initially*: define an order over the vertices, obtaining some sequence \( \{v_1, \ldots, v_n\} \)

1. \( v_1 \) colored with color 1
2. For vertex \( v_i \): try to color \( v_i \) using one of the previously used color 1, \ldots, \( k \) (choose the lowest feasible color); if not possible, color \( v_i \) with new color \( k + 1 \)

**Two possible sorting strategies**

- Decreasing Degree
- Smallest Last
Performance of Sequential Coloring

- Assume vertices are considered in the order $v_1, \ldots, v_n$
- Let $G_i$ be graph induced by the vertices $v_1, \ldots, v_i$ ($G_n = G$)
- Let $k_i$ no. colors used by the sequential algorithm to color vertices in $G_i$ ($k_n$ is overall no. colors used for $G$)
- Let $d_i(v)$ denote $v$’s degree in $G_i$

**Theorem**

$$k_n \leq 1 + \max_{1 \leq i \leq n} \min\{d_n(v_i), i - 1\}$$

**Proof**

*Required.* See [1, Section 2.2]
Corollary

For any ordering of the vertices, sequential coloring uses at most \( \Delta + 1 \) colors, where \( \Delta = \max_{1 \leq i \leq n} d(v) \)

Decreasing Degree

Sort vertices according to non-increasing degree
May perform poorly...

Consider the sequence \( \{x_1, y_1, \ldots, x_n, y_n\} \). Picture taken from [1, Section 2.2]
Smallest Last (SL) heuristic

We again produce an ordering \( \{v_1, \ldots, v_n\} \) of the vertices

- \( v_n \) is the smallest degree vertex in \( G \)
- Inductively, \( v_i \) is the minimum degree vertex in the subgraph induced by \( V - \{v_{i+1}, \ldots, v_n\} \)

Performance of SL

SL not good for all graphs (can be \( \Omega(n) \) approximate) but...

Theorem

\textit{SL colors any planar graph using at most 6 colors}

Proof

\textit{Required.} Uses Euler's Theorem (See [1, Section 2.2])
Euler’s Theorem for planar graphs

**Theorem**

*In a finite, connected planar graph, if \( n, m \) and \( f \) are respectively the number of vertices, edges and faces (including the outer face), then \( n - m + f = 2 \)

**Consequences**

In any planar graph, the lowest degree vertex has at most 5 neighbours

**Corollary**

*There is a polynomial coloring algorithm \( A \) such that, if \( G \) is any planar graph: \( m_{SL}(G) \leq 2m^*(G) \)
Giorgio Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann.

*Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.*