Laboratório de Otimização e Combinatória

Instituto de Computação

UNICAMP

Online Combinatorial Optimization Problems

DTC/LOCo

Mário César San Felice

Postdoc at IME-USP

April 29, 2016

Combinatorial Optimization Problems

Maximization or minimization problems in which, for each input there is a set of feasible solutions and, for each solution there is a cost associated with it.

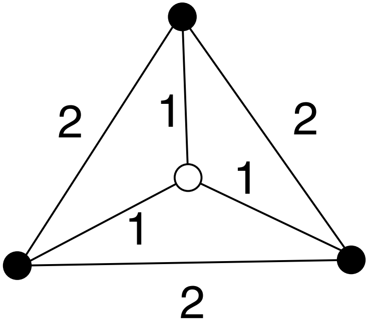
Combinatorial Optimization Problems

Maximization or minimization problems in which, for each input there is a set of feasible solutions and, for each solution there is a cost associated with it.

As an example, lets take the Steiner Tree Problem.

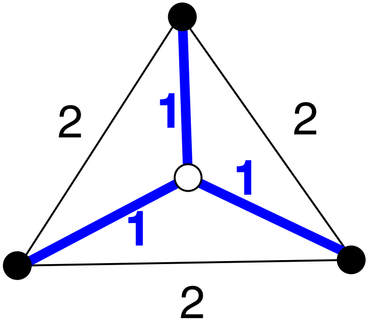
Steiner Tree Problem

Minimization problem whose input is a graph with costs on the edges, a set of terminal nodes and a set of Steiner nodes. A feasible solution is a tree that connects all terminal nodes, and its cost is the sum of the edge costs in the tree.



Steiner Tree Problem

Minimization problem whose input is a graph with costs on the edges, a set of terminal nodes and a set of Steiner nodes. A feasible solution is a tree that connects all terminal nodes, and its cost is the sum of the edge costs in the tree.



Online Problems

Problems in which the parts of the input arrive one at a time and each part need to be served before the next one arrives. Also, no decision made to serve a part may be changed in the future.

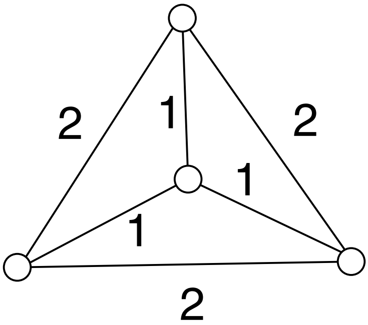
Online Problems

Problems in which the parts of the input arrive one at a time and each part need to be served before the next one arrives. Also, no decision made to serve a part may be changed in the future.

As an example, lets take the Online Steiner Tree problem.

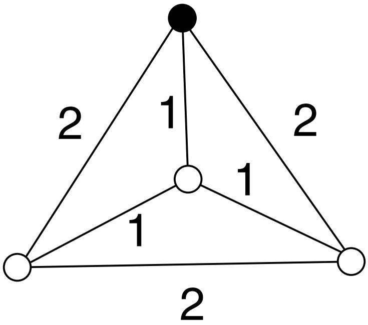
Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



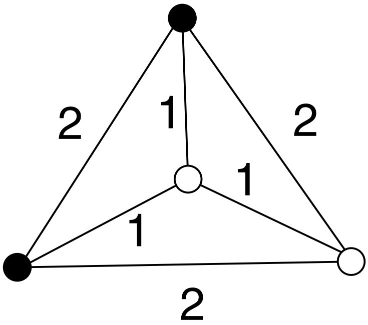
Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



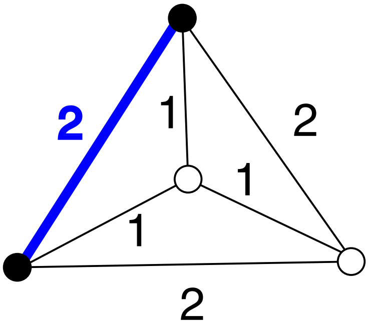
Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



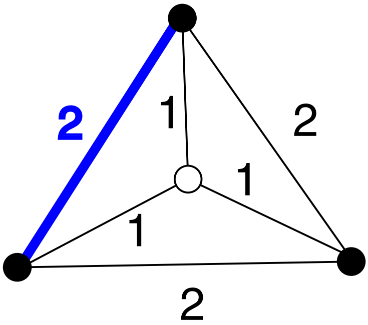
Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



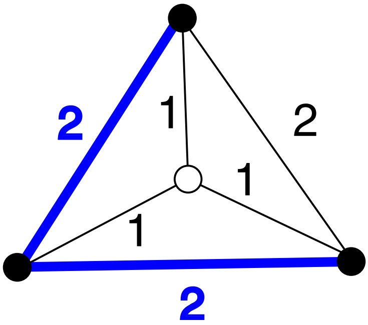
Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



Online Steiner Tree Problem

This problem is defined similarly to the Steiner Tree problem, except that the terminal nodes arrive one at a time. At all times the terminals must be connected by a tree and no edge used may be removed in the future.



Competitive Analysis

Worst case technique used to analyse online algorithms.

Competitive Analysis

Worst case technique used to analyse online algorithms.

We say that an online algorithm ALG is c-competitive if, for every input I, we have

$$\mathrm{ALG}(I) \leq c\,\mathrm{OPT}(I) \ .$$

Competitive Analysis

Worst case technique used to analyse online algorithms.

We say that an online algorithm ALG is c-competitive if, for every input I, we have

$$\mathrm{ALG}(I) \leq c\,\mathrm{OPT}(I) \quad .$$

As an example, lets take a greedy online algorithm for the Online Steiner Tree problem.

Greedy Online Steiner Tree Algorithm

Algorithm 1: OST Algorithm.

Input: (G, d)
$T \leftarrow (\emptyset, \emptyset)$;
while *a new terminal j arrives* **do**
    $T \leftarrow T \cup \{\mathrm{path}(j, V(T))\}$;
end
return T;

Greedy Online Steiner Tree Algorithm

Algorithm 1: OST Algorithm.

Input: (G, d)
$T \leftarrow (\emptyset, \emptyset)$;
while *a new terminal j arrives* **do**
  $T \leftarrow T \cup \{\mathrm{path}(j, V(T))\}$;
end
return T;

The OST Algorithm is $\mathrm{O}(\log n)$-competitive. Also, it is known a $\Omega(\log n)$ lower bound to the competitive ratio of any algorithm for this problem [Imase and Waxman 1991].

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

- Resource management: scheduling, packing and load balancing problems.

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

- Resource management: scheduling, packing and load balancing problems.
- Dynamic data structures: list access problem.

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

- Resource management: scheduling, packing and load balancing problems.
- Dynamic data structures: list access problem.
- Memory management: paging problem.

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

- Resource management: scheduling, packing and load balancing problems.
- Dynamic data structures: list access problem.
- Memory management: paging problem.
- Sustainability: ski-rental and parking permit problems.

Areas of Interest

Online problems allow us to capture the uncertainty related to input data that arrives over time. This characteristic is common in several problems from operations research and computer science:

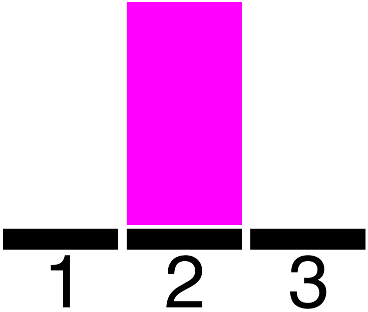
- Resource management: scheduling, packing and load balancing problems.
- Dynamic data structures: list access problem.
- Memory management: paging problem.
- Sustainability: ski-rental and parking permit problems.
- Network design: online versions of Steiner tree and facility location problems.

Online Load Balancing problem

Minimization problem whose input is a number of machines and a list of tasks with sizes. A feasible solution is an assignment of each task to a machine, and its cost is the maximum load between the machines.

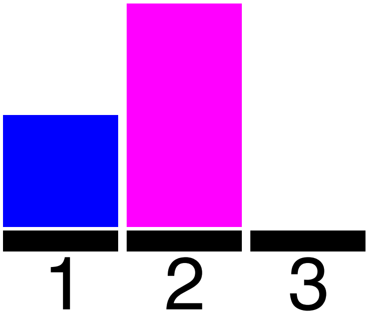
Online Load Balancing problem

Minimization problem whose input is a number of machines and a list of tasks with sizes. A feasible solution is an assignment of each task to a machine, and its cost is the maximum load between the machines.



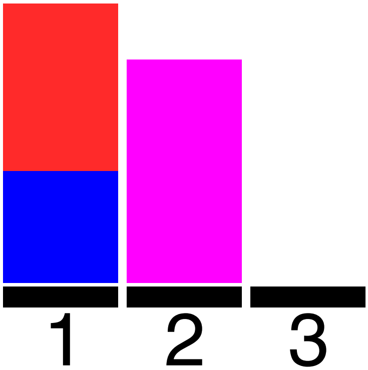
Online Load Balancing problem

Minimization problem whose input is a number of machines and a list of tasks with sizes. A feasible solution is an assignment of each task to a machine, and its cost is the maximum load between the machines.



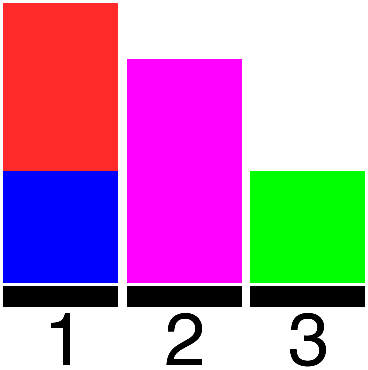
Online Load Balancing problem

Minimization problem whose input is a number of machines and a list of tasks with sizes. A feasible solution is an assignment of each task to a machine, and its cost is the maximum load between the machines.



Online Load Balancing problem

Minimization problem whose input is a number of machines and a list of tasks with sizes. A feasible solution is an assignment of each task to a machine, and its cost is the maximum load between the machines.



Greed Online Load Balancing Algorithm

Algorithm 2: OLB Algorithm.

Input: M

For each machine $i = 1, \ldots, M$ set its load $l(i)$ to 0;

$i^* \leftarrow 1$;

while *a new task j arrives* **do**

  $a(j) \leftarrow i^*$;

  $l(i^*) \leftarrow l(i^*) + s(j)$;

  choose machine with minimum load as new i^*;

end

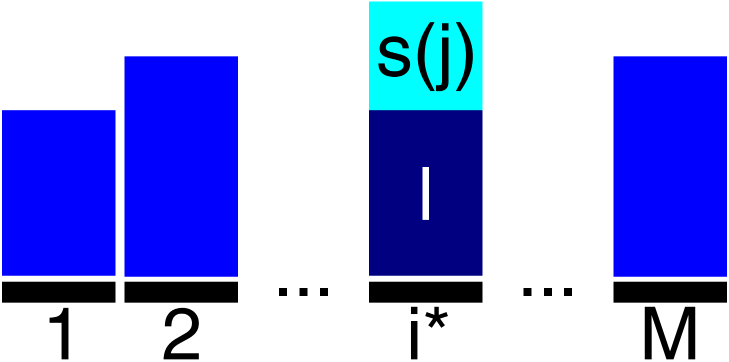
return a;

OLB Algorithm is $\left(2 - \frac{1}{M}\right)$-competitive

Let i^* be the machine with the maximum load, j be the last task assigned to i^*, and $l(i^*) = l + s(j)$.

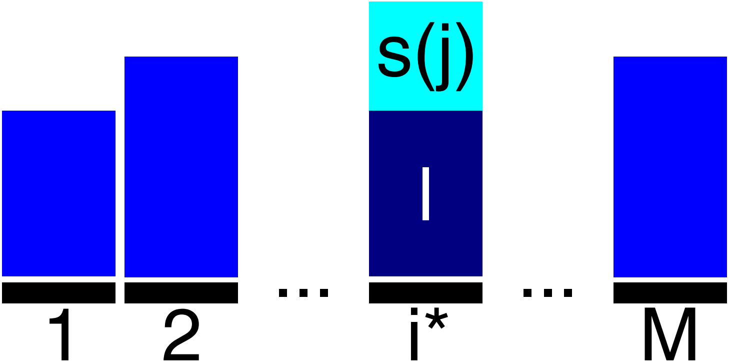
OLB Algorithm is $\left(2 - \frac{1}{M}\right)$-competitive

Let i^* be the machine with the maximum load, j be the last task assigned to i^*, and $l(i^*) = l + s(j)$.



OLB Algorithm is $(2 - \frac{1}{M})$-competitive

Let i^* be the machine with the maximum load, j be the last task assigned to i^*, and $l(i^*) = l + s(j)$.



We have $\mathrm{OPT} \geq s(j)$ and $\mathrm{OPT} \geq l + \frac{s(j)}{M}$.

OLB Algorithm is $\left(2 - \frac{1}{M}\right)$-competitive

Since $\mathrm{OPT} \geq s(j)$ and $\mathrm{OPT} \geq l + \frac{s(j)}{M}$, we have

OLB Algorithm is $\left(2 - \frac{1}{M}\right)$-competitive

Since $\mathrm{OPT} \geq s(j)$ and $\mathrm{OPT} \geq l + \frac{s(j)}{M}$, we have

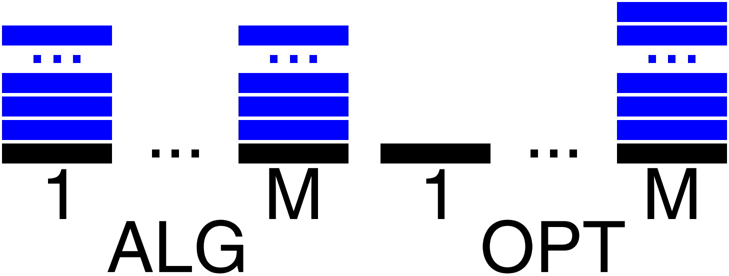
$$\begin{aligned}
\mathrm{ALG} &= l + s(j) \\
&\leq \mathrm{OPT} - \frac{s(j)}{M} + s(j) \\
&\leq \mathrm{OPT} + \left(1 - \frac{1}{M}\right)\mathrm{OPT} \\
&= \left(2 - \frac{1}{M}\right)\mathrm{OPT}\ .
\end{aligned}$$

Lower Bound for OLB Algorithm

Consider a list with $M(M-1)$ tasks of size 1 followed by a task of size M.

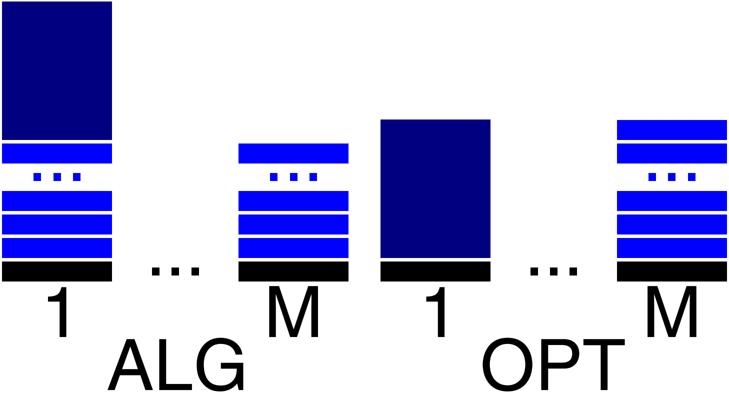
Lower Bound for OLB Algorithm

Consider a list with $M(M-1)$ tasks of size 1 followed by a task of size M.



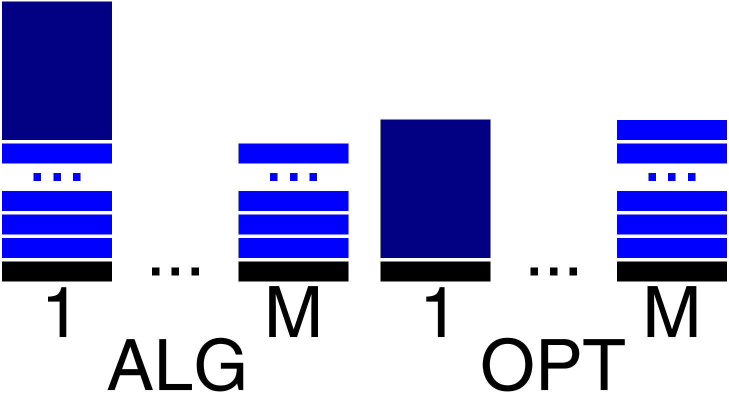
Lower Bound for OLB Algorithm

Consider a list with $M(M-1)$ tasks of size 1 followed by a task of size M.



Lower Bound for OLB Algorithm

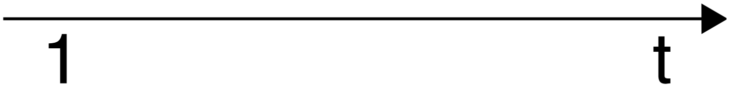
Consider a list with $M(M-1)$ tasks of size 1 followed by a task of size M.



We have $\mathrm{ALG} = 2M - 1$ and $\mathrm{OPT} = M$.

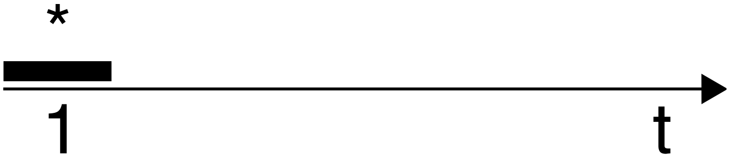
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.

Ski Rental Problem

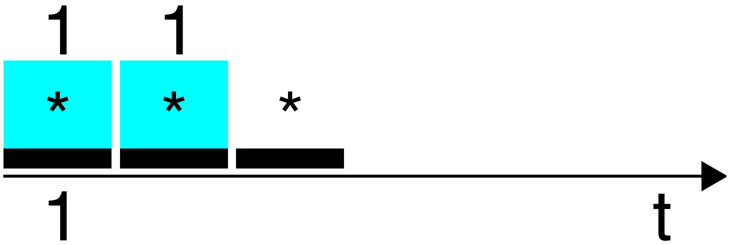
Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.

Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.

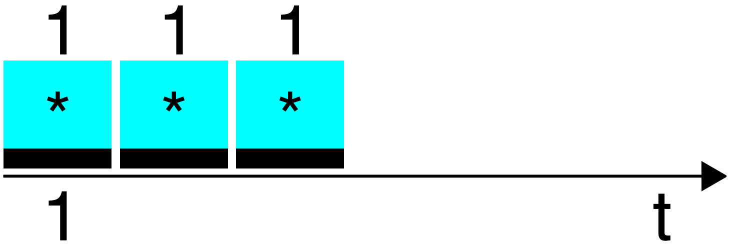
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



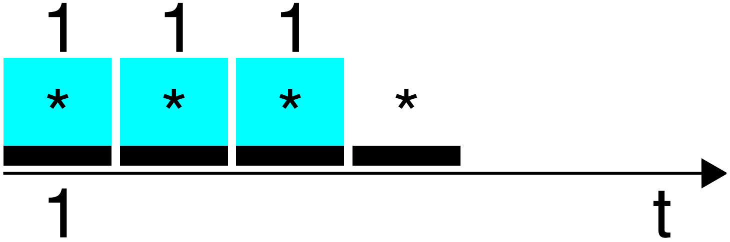
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



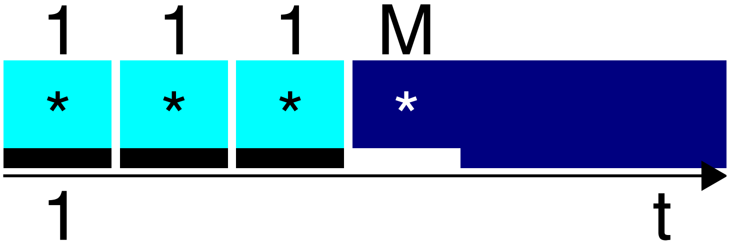
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



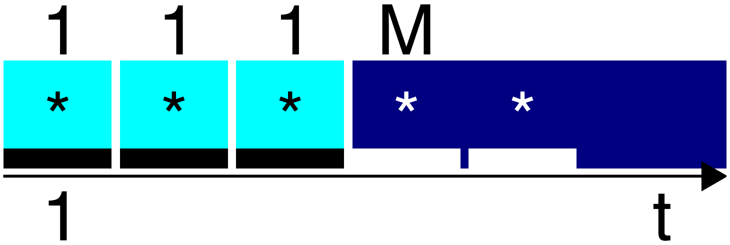
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



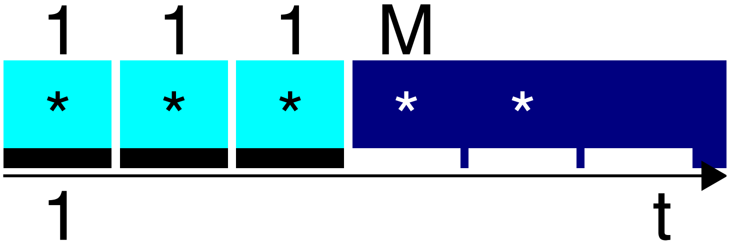
Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



Ski Rental Problem

Minimization problem whose input is the price M to buy skies and a list informing when the snow melt. A feasible solution is a list informing when we rent or buy skies, and its cost is 1 for each renting day plus M if you buy.



Ski Rental Application and Generalization

Ski rental algorithms may be used to save energy by deciding when to turn off parts of a system, like cores in a processor or computers in a cluster.

Ski Rental Application and Generalization

Ski rental algorithms may be used to save energy by deciding when to turn off parts of a system, like cores in a processor or computers in a cluster.

Also, it may be generalized to the Parking Permit Problem [Meyerson 2005].

Ski Rental Algorithm

Algorithm 3: Intuitive SR Algorithm.

Input: M

Set day j and total renting cost r to 0;

while *a new snow day happens* **do**

  if $r + 1 < M$ **then**

    Rent skies at day j and $r \leftarrow r + 1$;

  else

    Buy skies if still don't have them;

  end

  $j \leftarrow j + 1$;

end

Ski Rental Algorithm

Algorithm 3: Intuitive SR Algorithm.

Input: M

Set day j and total renting cost r to 0;

while *a new snow day happens* **do**

  if $r + 1 < M$ **then**

    Rent skies at day j and $r \leftarrow r + 1$;

  else

    Buy skies if still don't have them;

  end

  $j \leftarrow j + 1$;

end

This algorithm is 2-competitive. Why?

Ski Rental LP Formulations

Ski Rental LP Formulations

Linear programming relaxation

$$
\begin{aligned}
\min \quad & Mx + \textstyle\sum_{j=1}^{n} y_j \\
\text{s.t.} \quad & x + y_j \geq 1 \qquad \text{for } j = 1, \ldots, n, \\
& x \geq 0, y_j \geq 0 \quad \text{for } j = 1, \ldots, n,
\end{aligned}
$$

Ski Rental LP Formulations

Linear programming relaxation

$$\begin{aligned} \min \quad & Mx + \textstyle\sum_{j=1}^{n} y_j & \\ \text{s.t.} \quad & x + y_j \geq 1 & \text{for } j = 1, \ldots, n, \\ & x \geq 0, y_j \geq 0 & \text{for } j = 1, \ldots, n, \end{aligned}$$

and its dual

$$\begin{aligned} \max \quad & \textstyle\sum_{j=1}^{n} \alpha_j & \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{n} \alpha_j \leq M & \text{for } j = 1, \ldots, n, \\ & \alpha_j \leq 1 & \text{for } j = 1, \ldots, n. \\ & \alpha_j \geq 0 & \text{for } j = 1, \ldots, n. \end{aligned}$$

Primal-Dual Ski Rental Algorithm

Algorithm 4: Primal-Dual SR Algorithm.

Input: M

Set day j' to 0;

while *a new snow day happens* **do**

increase $\alpha_{j'}$ until one of the following happens:

(a) $\alpha_{j'} = 1$; /* rent skies setting $y_j = 1$ */

(b) $M = \alpha_{j'} + \sum_{j=1}^{j'-1} \alpha_j$; /* buy skies setting $x = 1$ */

$j' \leftarrow j' + 1$;

end

Does it remember the previous algorithm?

Primal-Dual SR Algorithm is 2-Competitive

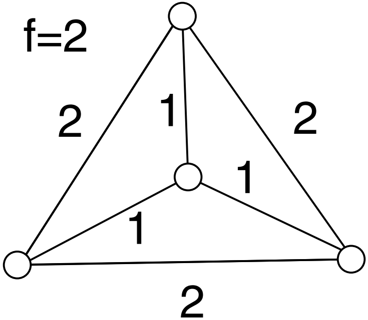
Recalling that the cost of any dual feasible solution is at most the cost of OPT, we have

Primal-Dual SR Algorithm is 2-Competitive

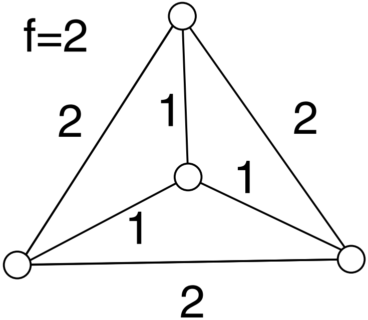
Recalling that the cost of any dual feasible solution is at most the cost of OPT, we have

$$\begin{aligned} ALG &= Mx + \sum_{j=1}^{n} y_j \\ &\leq \sum_{j=1}^{n} \alpha_j + \sum_{j=1}^{n} \alpha_j \\ &\leq 2\mathrm{OPT} \ . \end{aligned}$$

Online Facility Location Problem

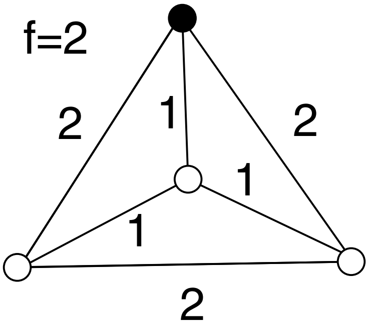


Online Facility Location Problem



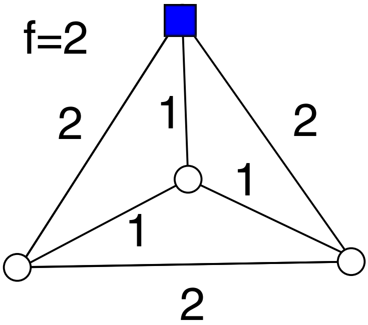
$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

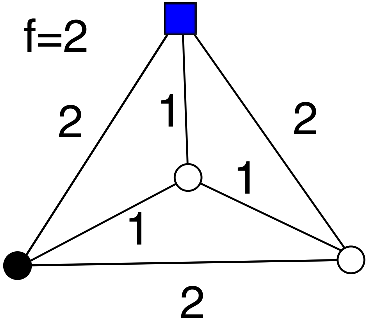
Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2$

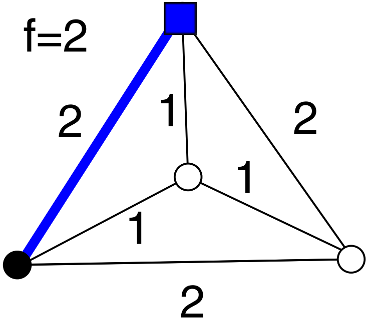
Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2$

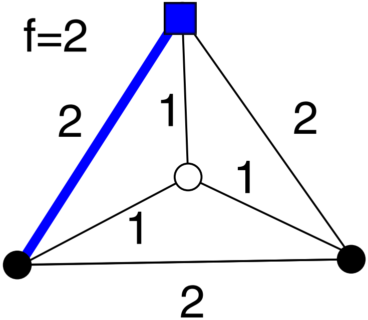
Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2 + 2$

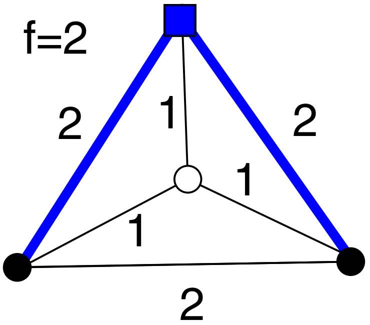
Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2 + 2$

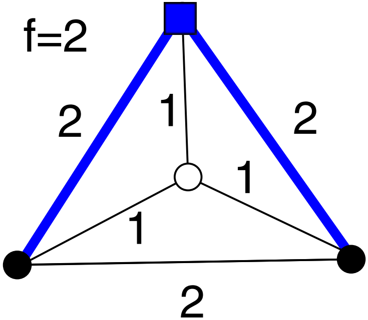
Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2 + 2 + 2$

Online Facility Location Problem



$$\min \sum_{i \in F^a} f(i) + \sum_{j \in D} d(j, a(j))$$

Total cost $= 2 + 2 + 2 = 6$.

Online Facility Location LP Formulation

Online Facility Location LP Formulation

Linear programming relaxation

$$\begin{array}{lll}
\min & \sum_{i\in F} f(i)y_i + \sum_{j\in D}\sum_{i\in F} d(j,i)x_{ji} & \\
\text{s.t.} & x_{ji} \leq y_i & \text{for } j \in D \text{ and } i \in F, \\
& \sum_{i\in F} x_{ji} \geq 1 & \text{for } j \in D, \\
& y_i \geq 0, x_{ji} \geq 0 & \text{for } j \in D \text{ and } i \in F,
\end{array}$$

Online Facility Location LP Formulation

Linear programming relaxation

$$\begin{array}{lll} \min & \sum_{i\in F} f(i)y_i + \sum_{j\in D}\sum_{i\in F} d(j,i)x_{ji} & \\ \text{s.t.} & x_{ji} \le y_i & \text{for } j \in D \text{ and } i \in F, \\ & \sum_{i\in F} x_{ji} \ge 1 & \text{for } j \in D, \\ & y_i \ge 0, x_{ji} \ge 0 & \text{for } j \in D \text{ and } i \in F, \end{array}$$

and its dual

$$\begin{array}{lll} \max & \sum_{j\in D} \alpha_j & \\ \text{s.t.} & \sum_{j\in D}(\alpha_j - d(j,i))^+ \le f(i) & \text{for } i \in F, \\ & \alpha_j \ge 0 & \text{for } j \in D. \end{array}$$

Online Facility Location Algorithm

Algorithm 5: OFL Algorithm.

Input: (G, d, f, F)
$F^a \leftarrow \emptyset$; $D \leftarrow \emptyset$;
while *a new client j' arrives* **do**
  increase $\alpha_{j'}$ until one of the following happens:
  (a) $\alpha_{j'} = d(j', i)$ for some $i \in F^a$; /* connect only */
  (b) $f(i) = (\alpha_{j'} - d(j', i)) + \sum_{j \in D}(d(j, F^a) - d(j, i))^+$ for some $i \in F \setminus F^a$; /* open and connect */
  $F^a \leftarrow F^a \cup \{i\}$; $D \leftarrow D \cup \{j'\}$; $a(j') \leftarrow i$;
end
return (F^a, a);

Online Facility Location Results

Online Facility Location Results

The OFL has competitive ratio $\Theta\left(\frac{\log n}{\log \log n}\right)$ [Fotakis 2008].

Online Facility Location Results

The OFL has competitive ratio $\Theta\left(\frac{\log n}{\log\log n}\right)$ [Fotakis 2008].

There are randomized and deterministic $\mathrm{O}(\log n)$-competitive algorithms known for it [Meyerson 2001, Fotakis 2007].

Online Facility Location Results

The OFL has competitive ratio $\Theta\left(\frac{\log n}{\log\log n}\right)$ [Fotakis 2008].

There are randomized and deterministic $\mathrm{O}(\log n)$-competitive algorithms known for it [Meyerson 2001, Fotakis 2007].

[Nagarajan and Williamson 2013] give a dual-fitting analysis for the algorithm by [Fotakis 2007].

Acknowledgements

Thank you!

Questions?