New Techniques and Algorithms for Multiobjective and Lexicographic Goal-Based Shortest Path Problems

PhD Dissertation

Francisco J. Pulido Arrebola

Supervised by: Dr. Lawrence Mandow

Dept. Lenguajes y Ciencias de la Computación
E.T.S. Ingeniería Informática
Universidad de Málaga

July 7, 2015

Outline

Outline

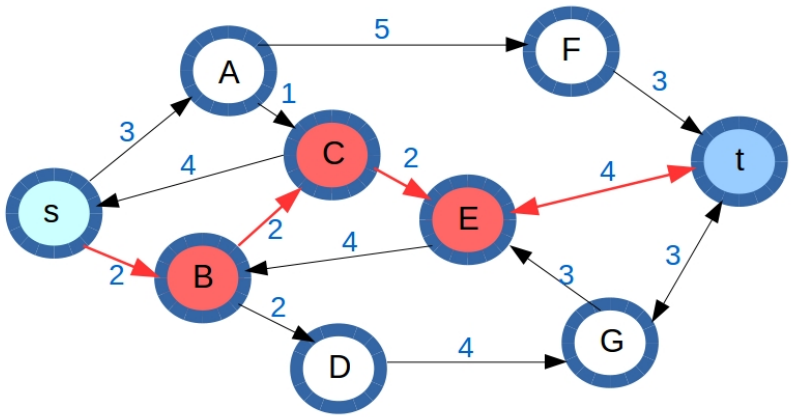
Outline

The Shortest Path Problem (SPP)

Definition

The SPP consists in finding a minimal cost path between two nodes in a graph, where the graph is composed of arcs labeled with the cost of the transition between nodes.

- Solved by Dijkstra in 1959 and by Hart et al. (A^*) in 1968 exploiting specific problem knowledge.



The Shortest Path Problem (SPP)

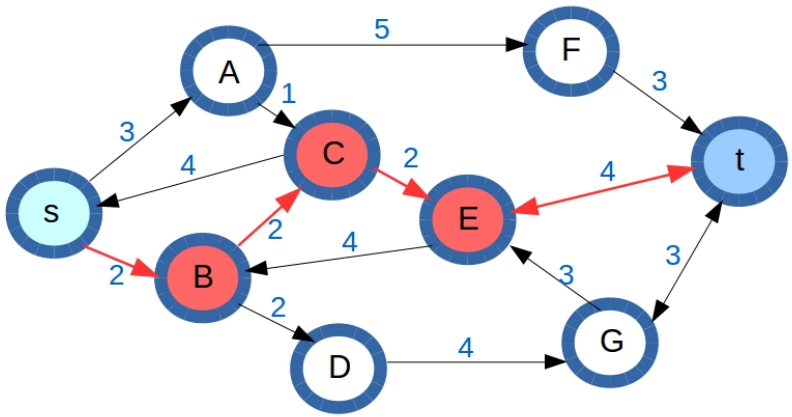
Definition

The SPP consists in finding a minimal cost path between two nodes in a graph, where the graph is composed of arcs labeled with the cost of the transition between nodes.

- Solved by Dijkstra in 1959 and by Hart et al. (A^*) in 1968 exploiting specific problem knowledge.

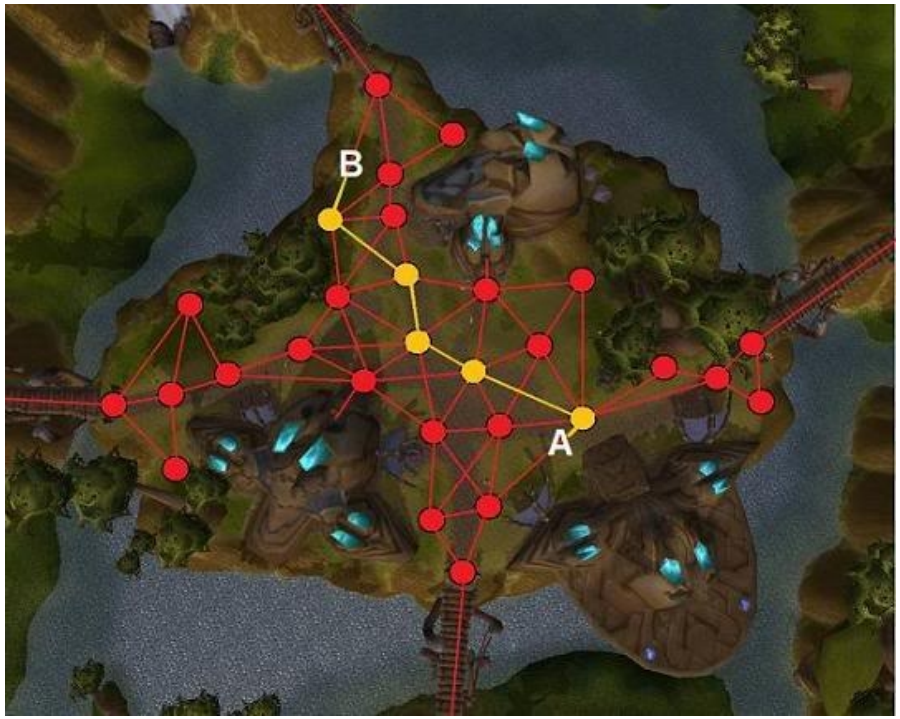
Research fields

- Artificial Intelligence
- Operational Research



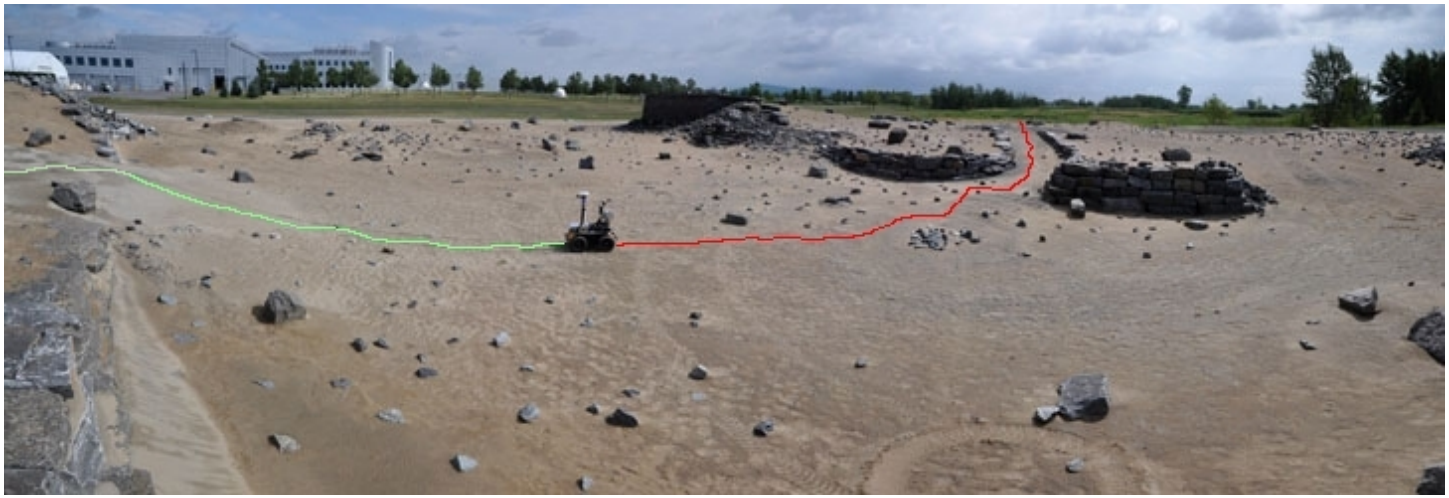
Applications to the SPP

- Path planning in game maps.
  - The game map is represented as a graph.
  - The characters movement through the map is solved as a SPP.



Applications to the SPP

- Motion planning in mobile robots.
  - The terrain is represented as a graph.
  - The robot must find a path avoiding obstacles.



Applications to the SPP

- Route planning in road maps.
  - Real road networks are graphs.
  - Arcs represent road junctions.
  - Arcs are typically labeled with the distance between junctions.

Multicriteria Search Problem (MSP)

Definition

MSP is the natural extension of SPP to the multicriteria case, where arcs are labeled with vectors instead of scalar values.

SPP	$\longrightarrow$	single-objective optimization problem
Decision problems	$\longrightarrow$	multiple conflicting criteria

Human decisions

Humans make choices according to their personal preferences, which involve simultaneously multiple, and often, conflicting criteria.

Human decisions

Humans make choices according to their personal preferences, which involve simultaneously multiple, and often, conflicting criteria.

New car?

- Price
- Speed
- Brand
- Color

Human decisions

Humans make choices according to their personal preferences, which involve simultaneously multiple, and often, conflicting criteria.

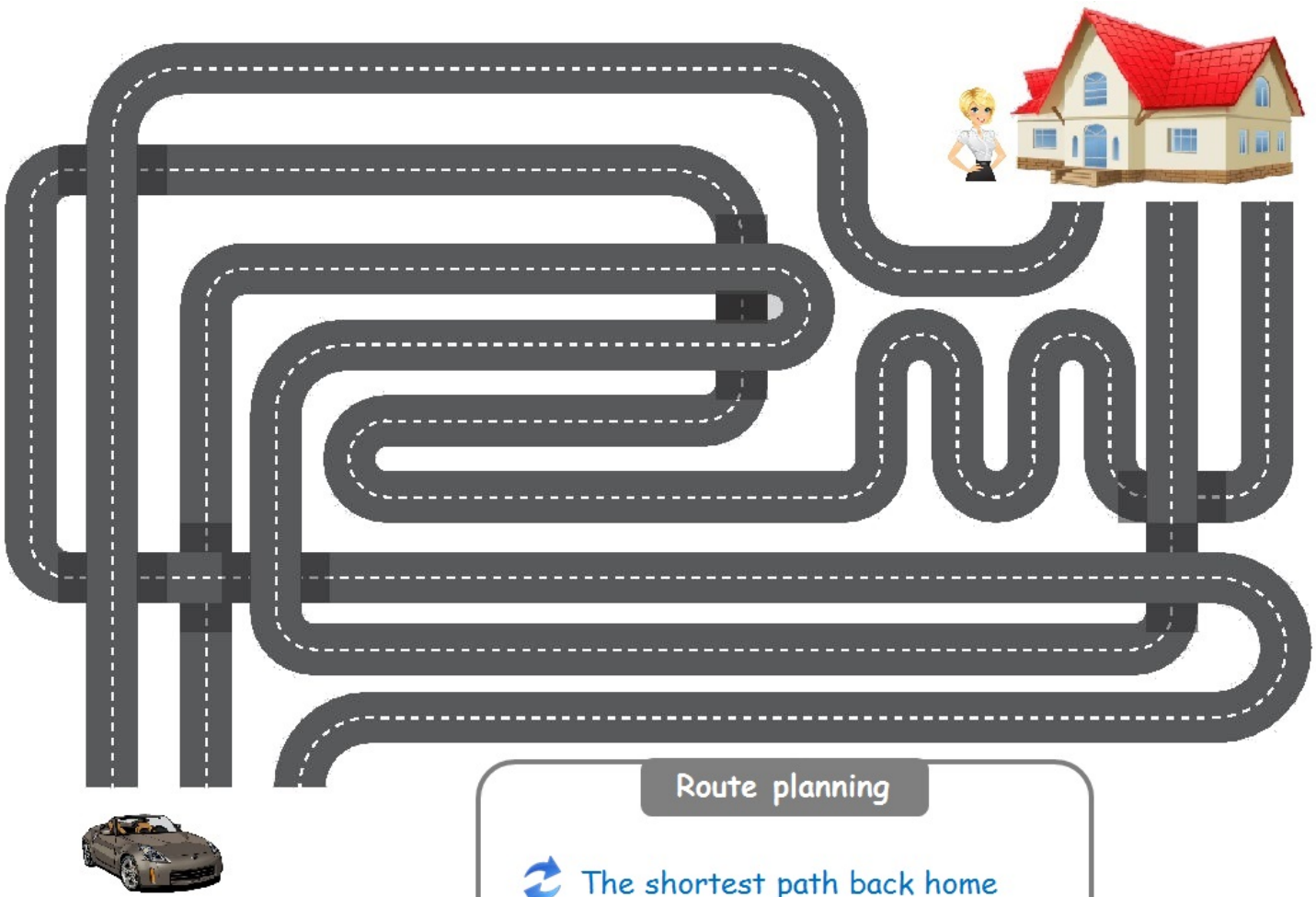
New car?

- Price
- Speed
- Brand
- Color

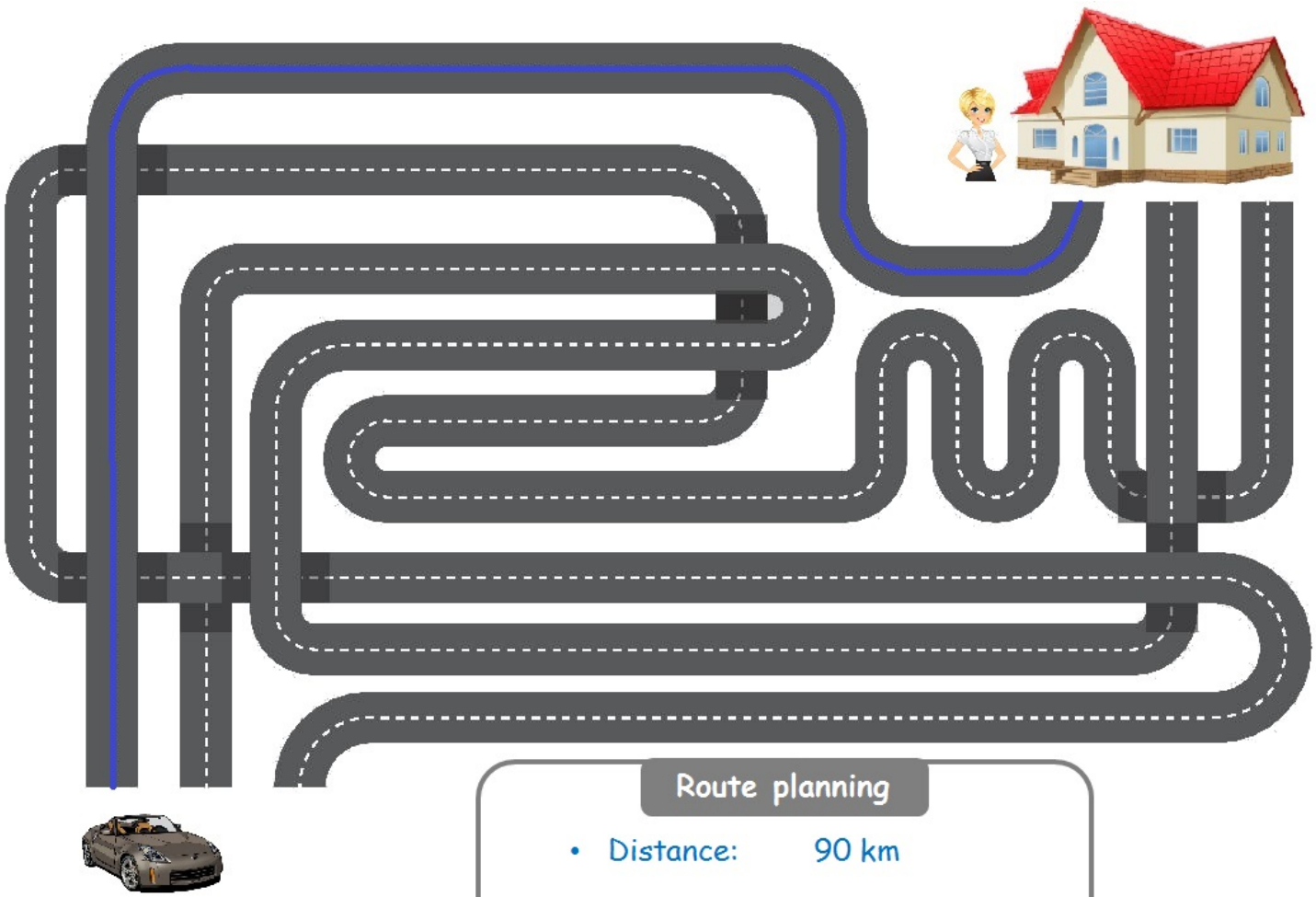
A route home?

- Distance?

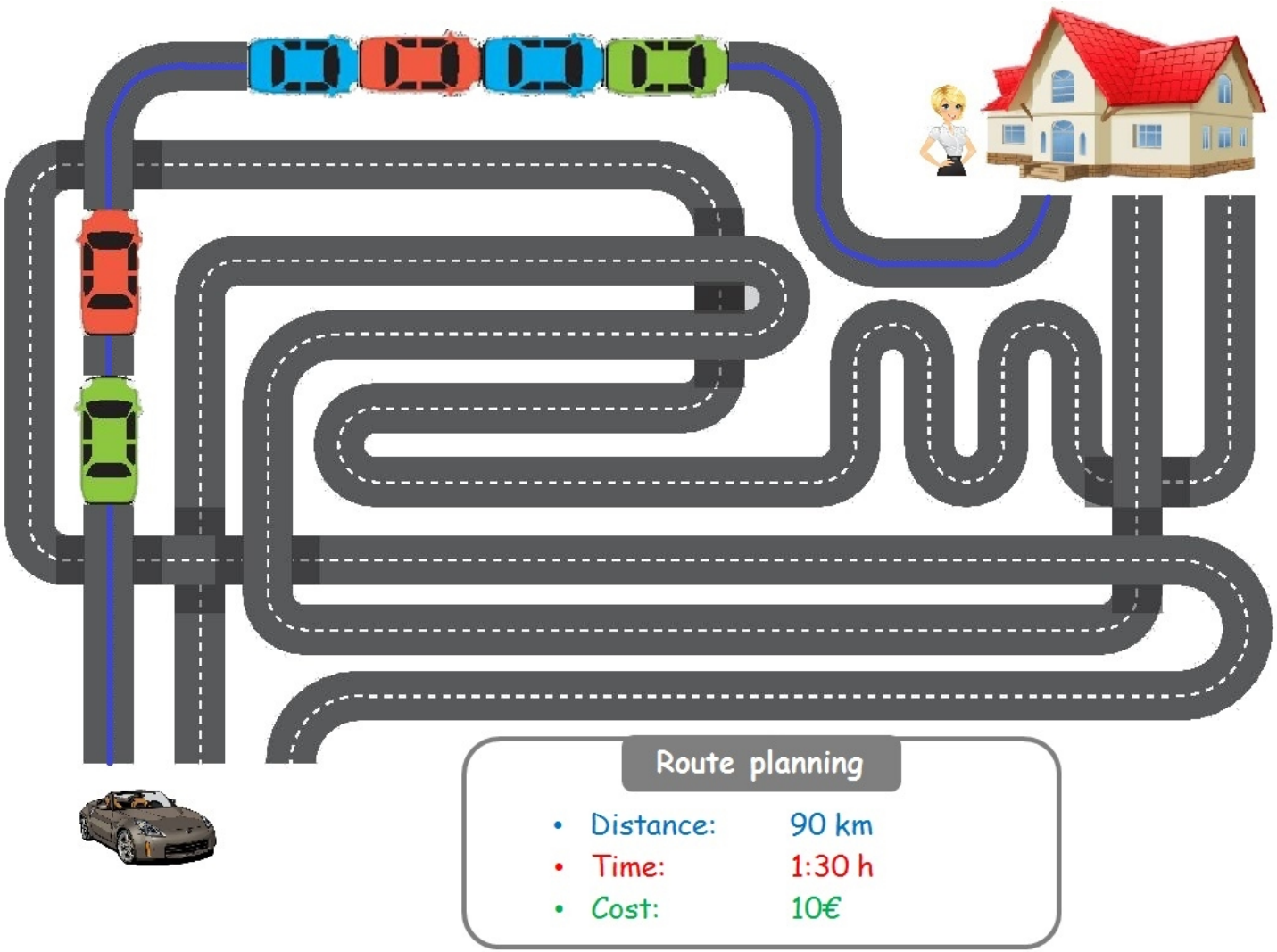
Route planning in road maps



Route planning in road maps

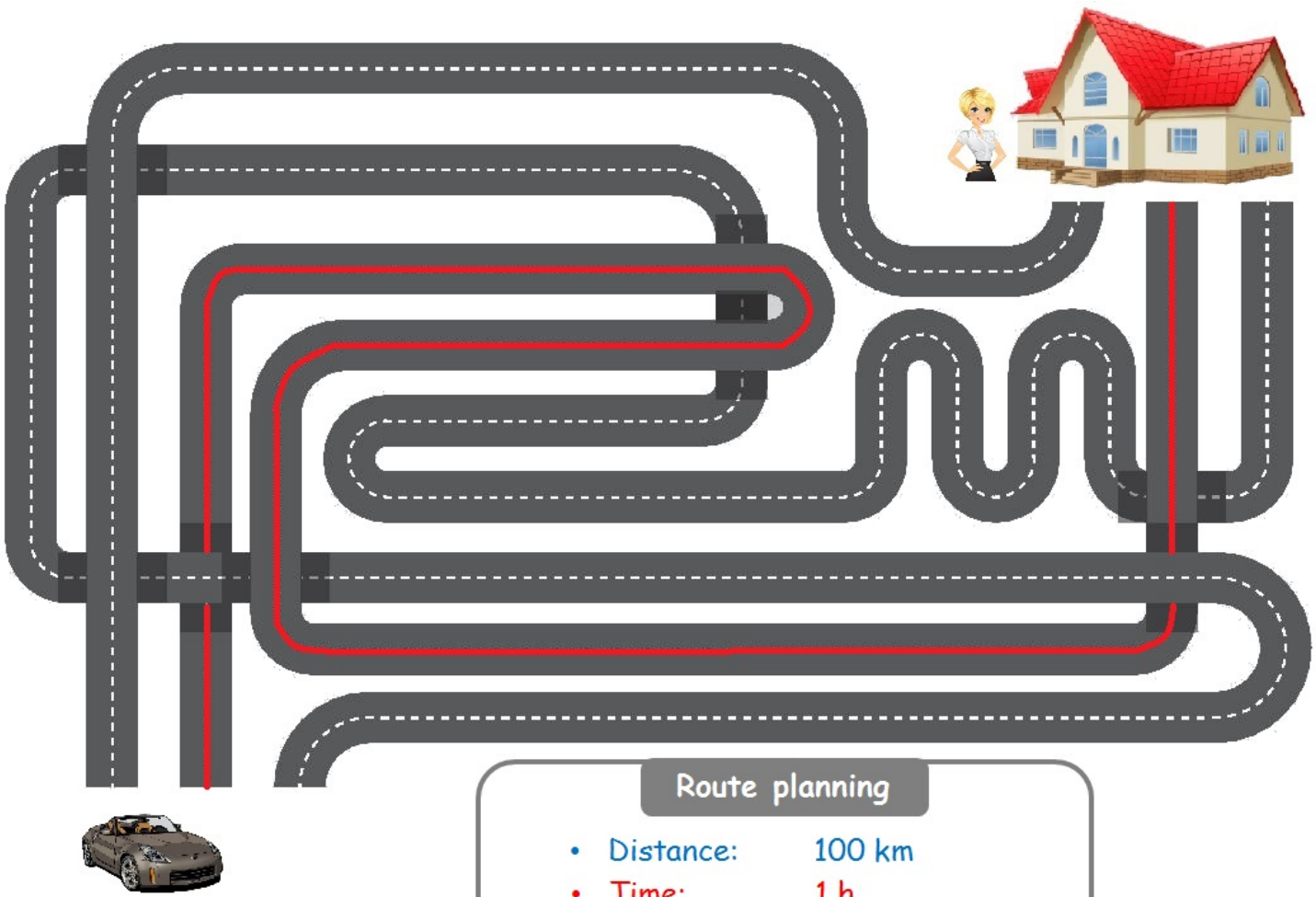


Route planning in road maps

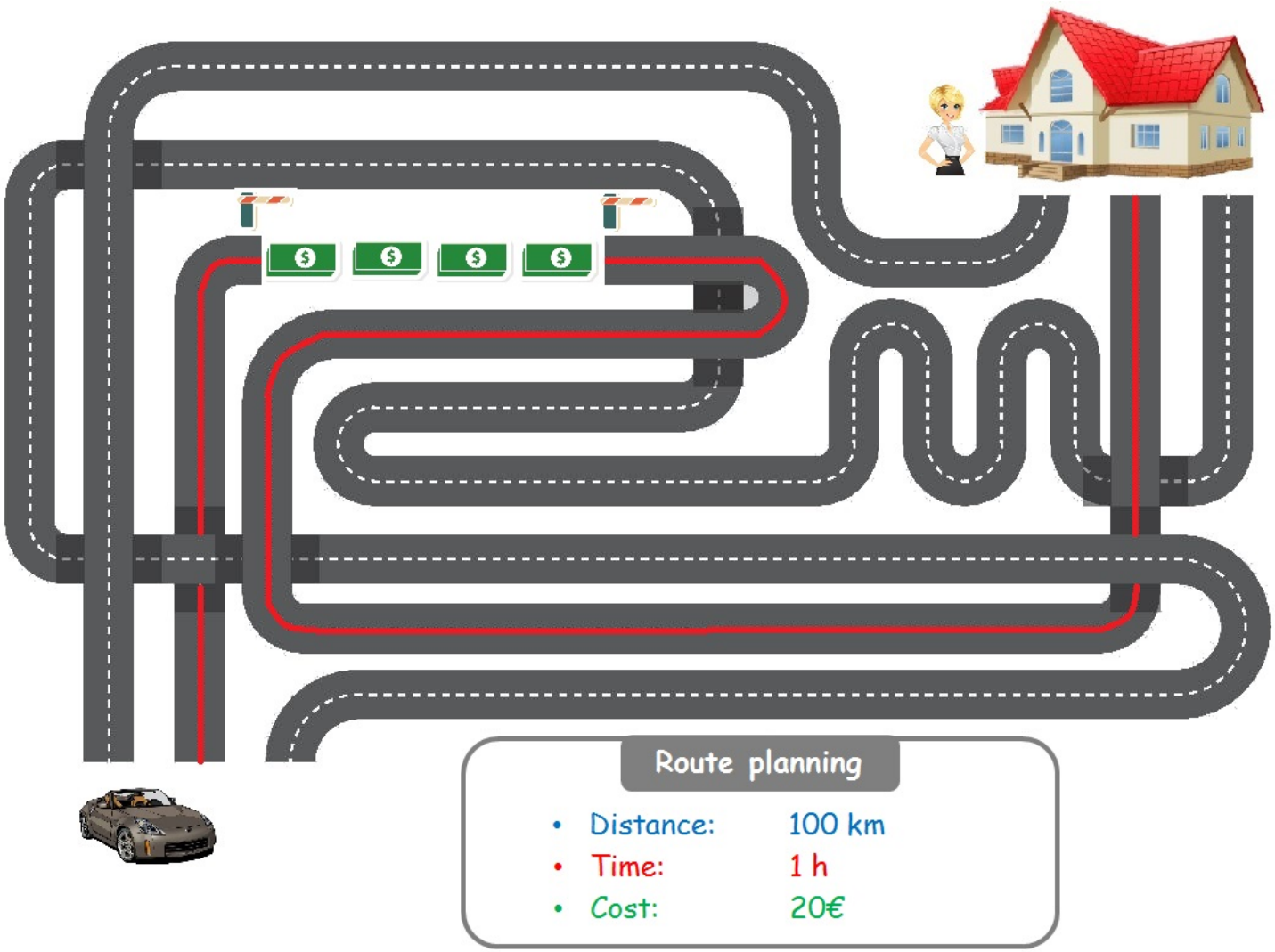


Route planning in road maps

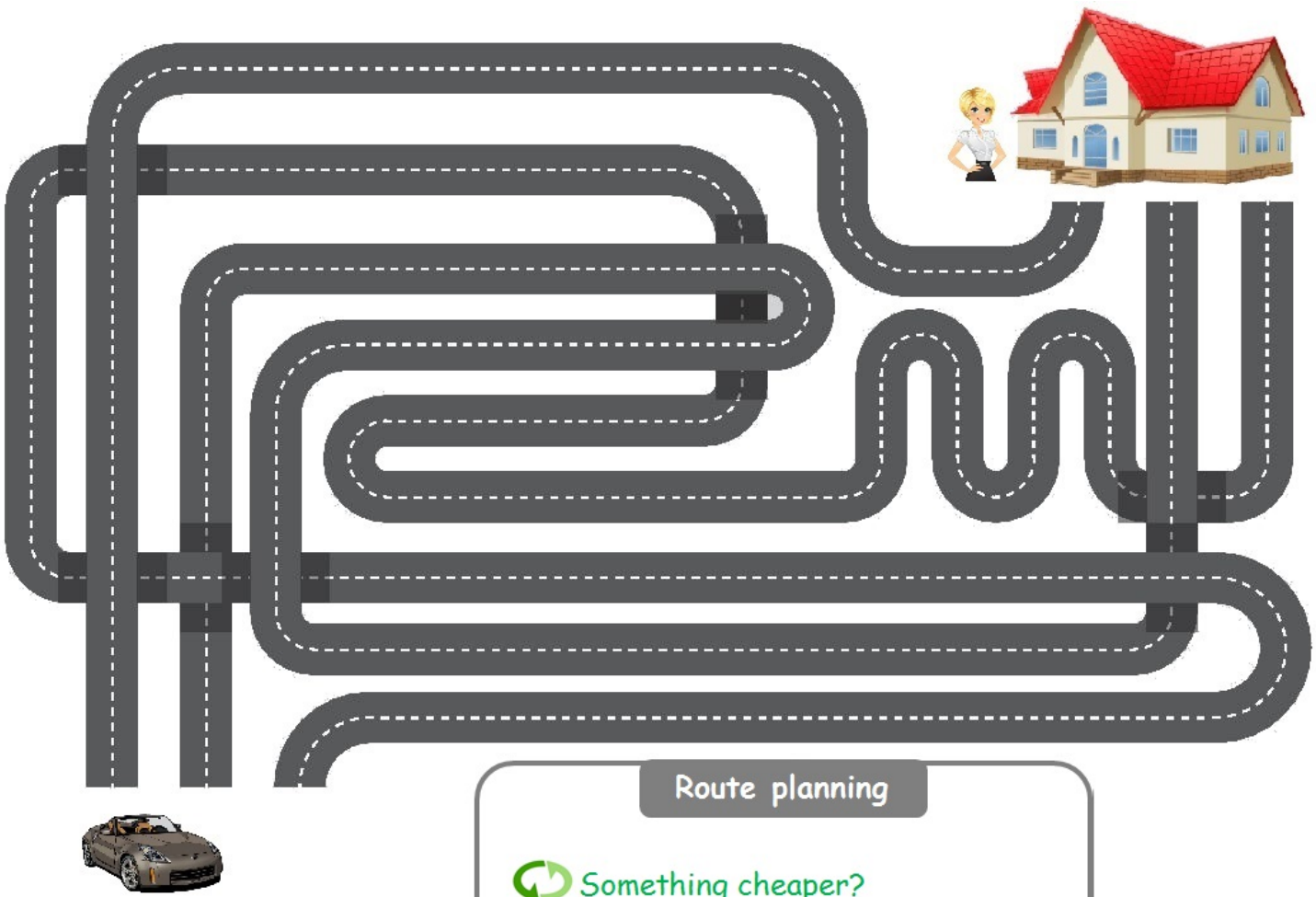
Route planning in road maps



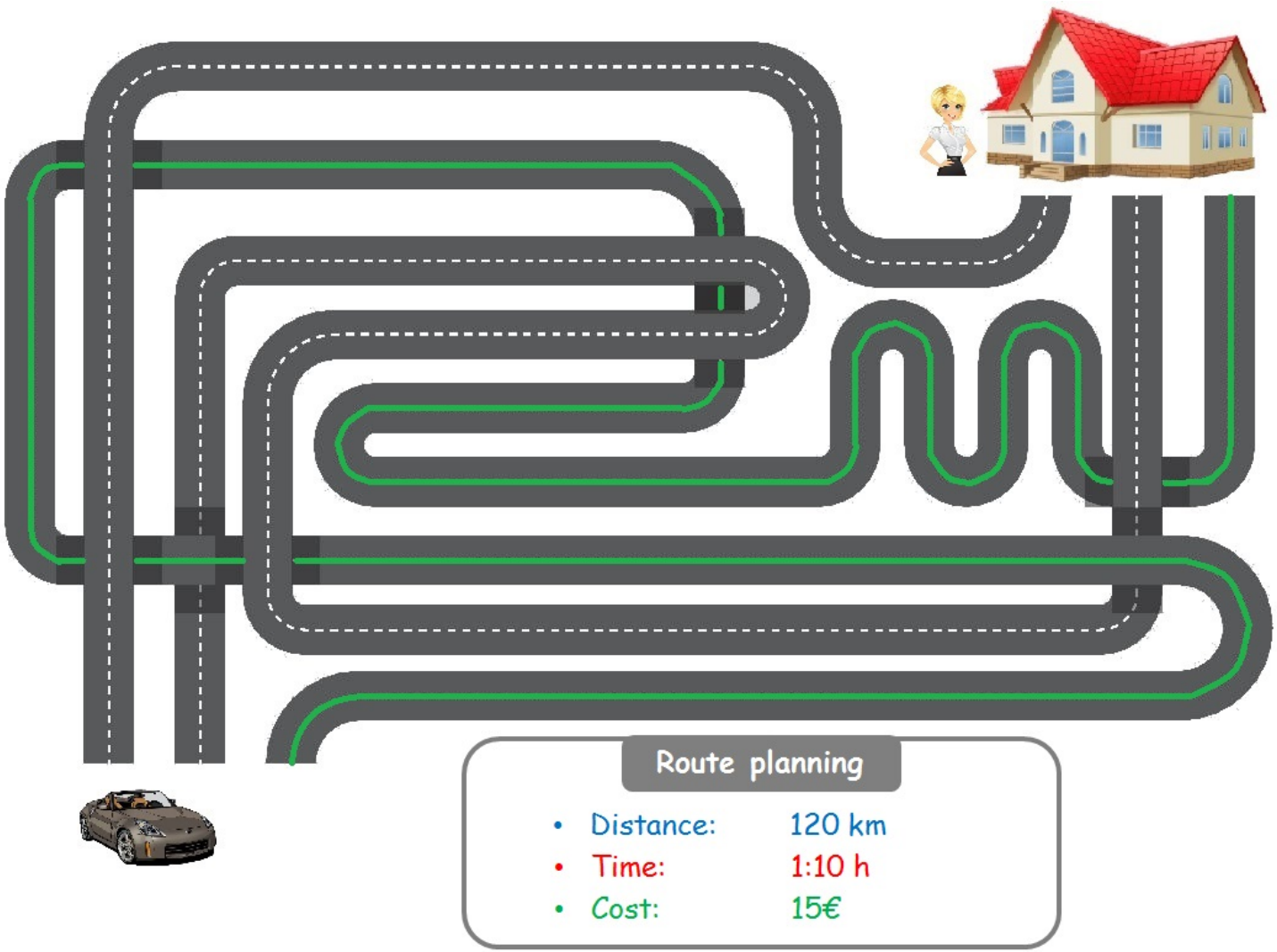
Route planning in road maps



Route planning in road maps



Route planning in road maps



Multicriteria Search Problem (MSP)

Definition

Dominance or Pareto preference $\prec$:

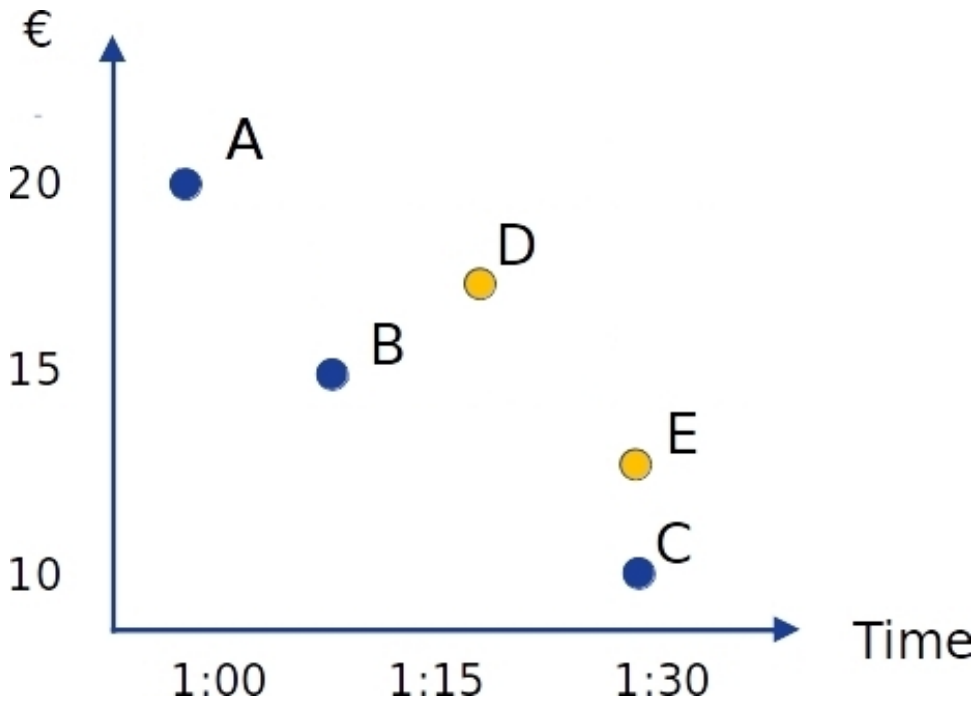
$$\vec{y} \prec \vec{y'} \quad \Leftrightarrow \quad \forall i \ (1 \leq \ i \leq \ q) \quad y_i \leq y'_i \ \wedge \ \vec{y} \neq \vec{y'}$$

where $\vec{y} = (y_1, y_2, ..., y_q)$ represents a solution path in the cost space.

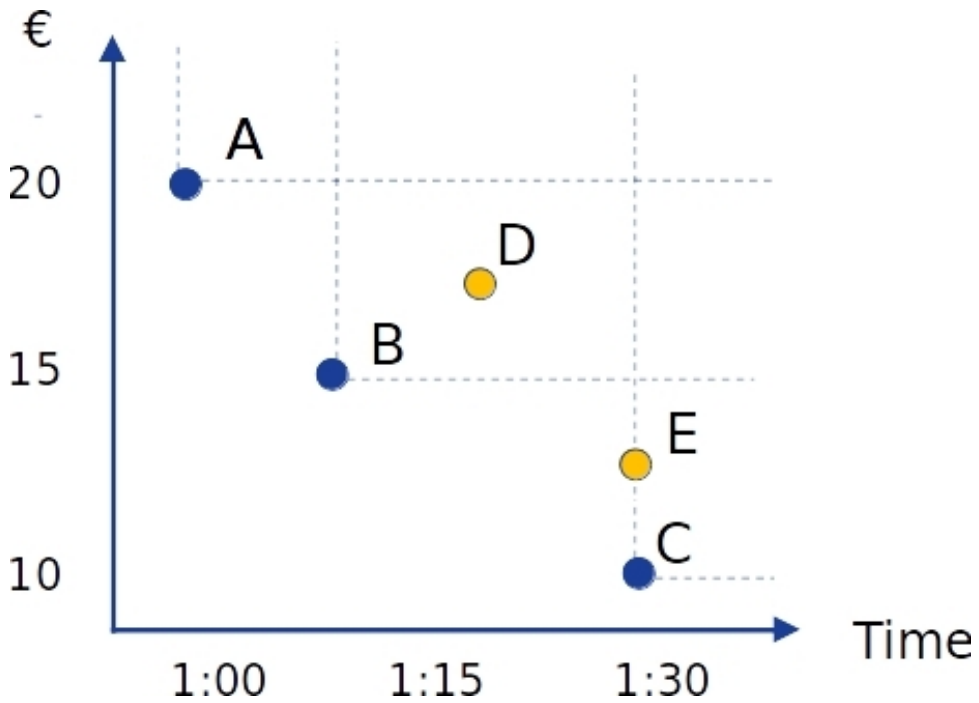
Example

Route cost representation: $\vec{y} = (y_1, y_2, y_3) \ \rightarrow \ \vec{y} = (120, 70, 15)$

Pareto optimality



Pareto optimality

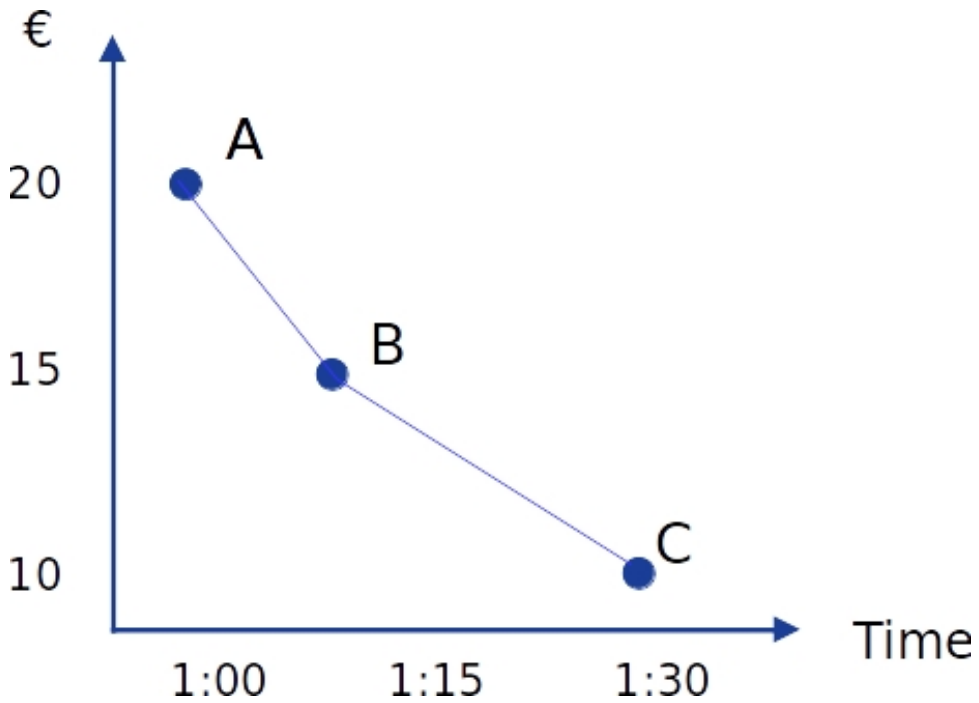


Multicriteria Search Problem (MSP)

Definition

The solution to a MSP, without any knowledge of the user preferences, is the set of non-dominated solution paths, or Pareto frontier.

Pareto optimality



Goal Programming

Definition

Goal Programming (GP) is one of the most successful techniques to solve multicriteria problems. It is also a natural way to express user preferences as targets or aspiration levels: $y_i \leq t_i$.

Definition

Lexicographic Goal Programming (LGP) ranks all goals in order of importance, dividing them into lexicographic levels of importance.

Definition

The deviation from a goal in a minimization problem represents a positive distance between the i-th goal and its aspiration level:
$d_i = \max(0, y_i - t_i)$.

Goal Programming

Example

Three criteria (economic cost (euros), travel time (minutes), and distance (km)) grouped in two priority levels where level 1 is infinitely more important than level 2. The importance of the criteria within a level is defined by weights as:

$$\begin{array}{lll} \text{Level 1:} & \textit{cost} \leq 15, & w_1 = 0.5 \\ & \textit{time} \leq 75, & w_2 = 0.5 \\ \text{Level 2:} & \textit{distance} \leq 120, & w_3 = 1. \end{array}$$

Example

Let's assume a solution path with cost $\vec{y} = (10, 60, 150)$,
its deviation from those goals is $\vec{d} = (0,\ 30) \rightarrow$
$(\max(0, (10 - 15) \times 0.5) + \max(0, (60 - 75) \times 0.5),\ (150 - 120) \times 1)$

Goal Programming

Example

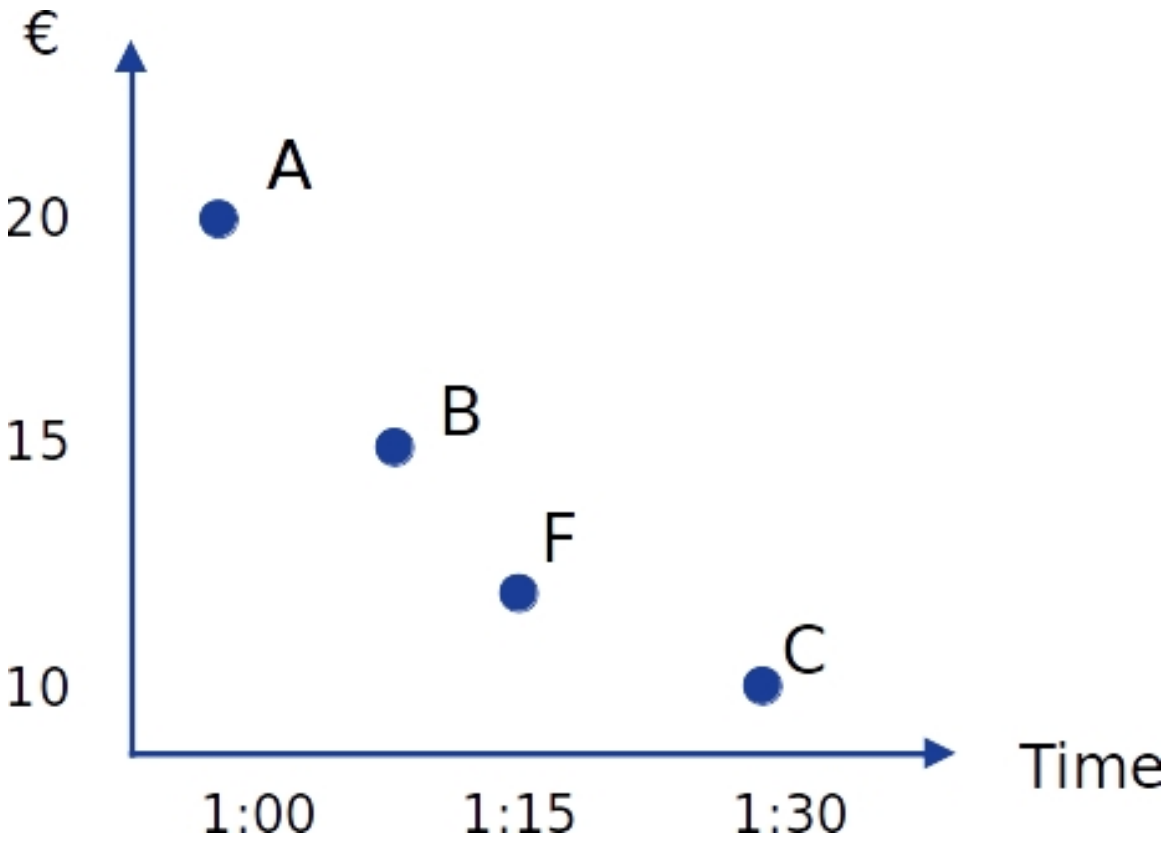
Three criteria (economic cost (euros), travel time (minutes), and distance (km)) grouped in two priority levels where level 1 is infinitely more important than level 2. The importance of the criteria within a level is defined by weights as:

$$\begin{array}{lll} \text{Level 1:} & \textit{cost} \leq 15, & w_1 = 0.5 \\ & \textit{time} \leq 75, & w_2 = 0.5 \\ \text{Level 2:} & \textit{distance} \leq 120, & w_3 = 1. \end{array}$$

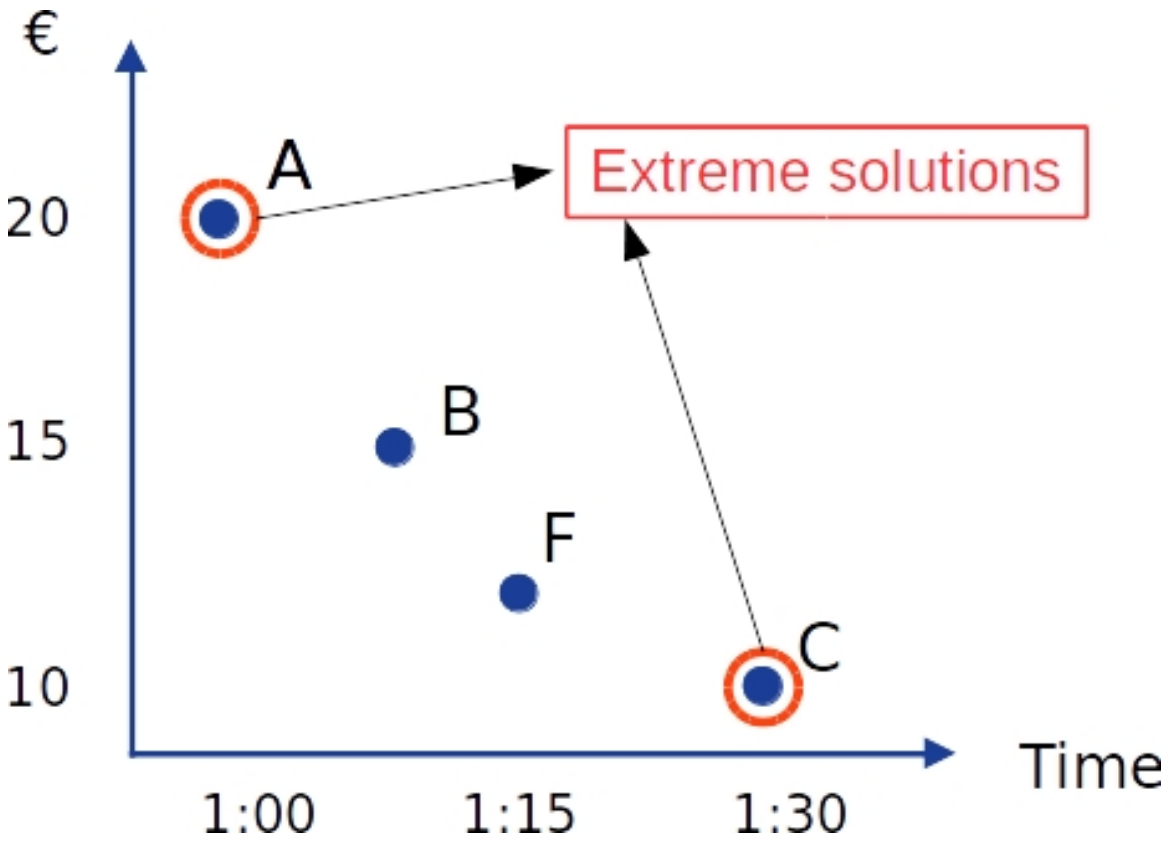
Example

Let's assume a solution path with cost $\vec{y} = (10, 60, 150)$,
its deviation from those goals is $\vec{d} = (0, 30) \rightarrow$
$(\max(0, (10 - 15) \times 0.5) + \max(0, (60 - 75) \times 0.5), (150 - 120) \times 1)$

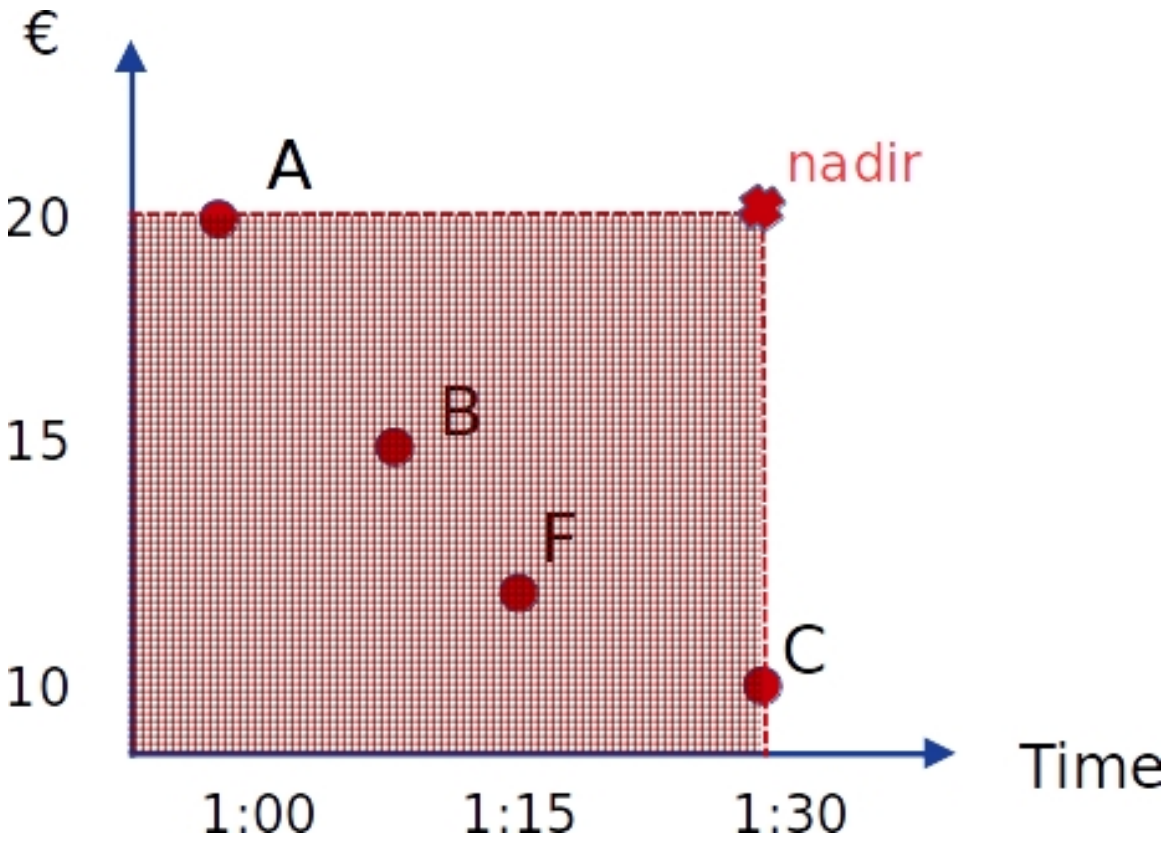
Lexicographic goals



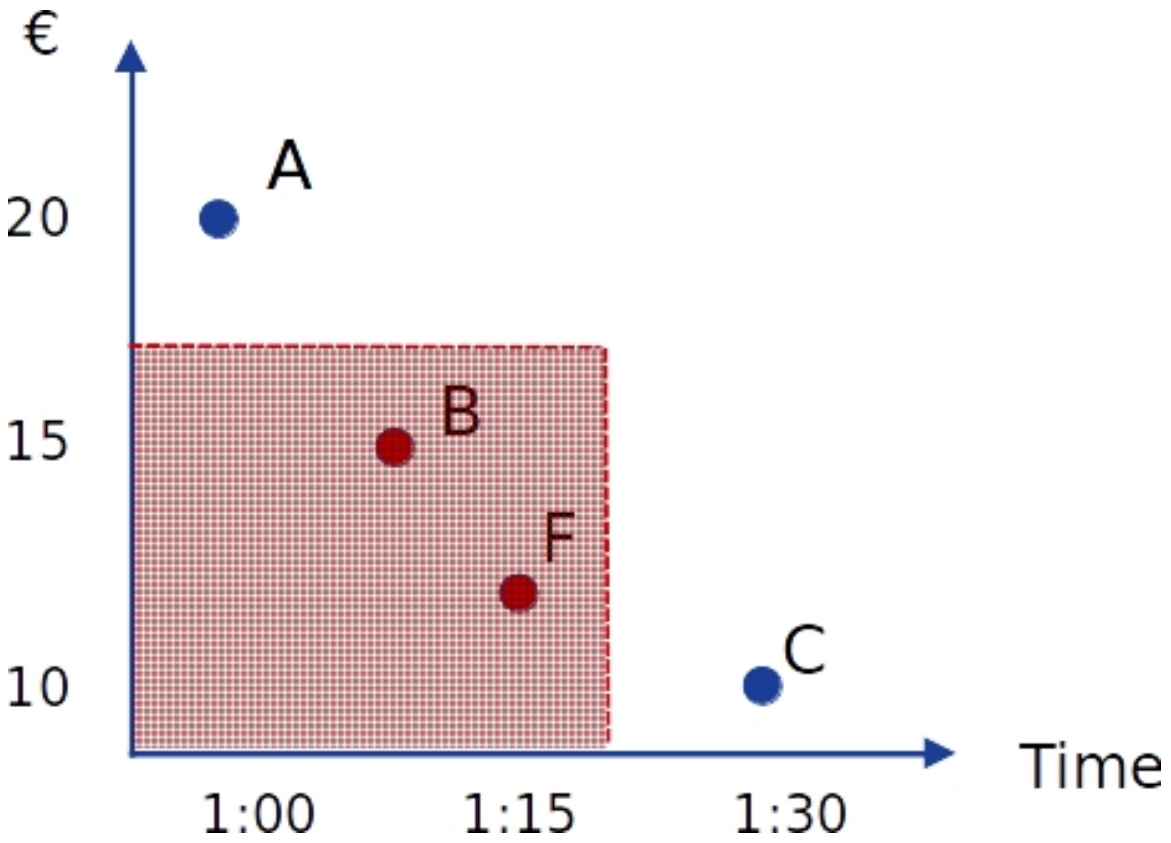
Lexicographic goals



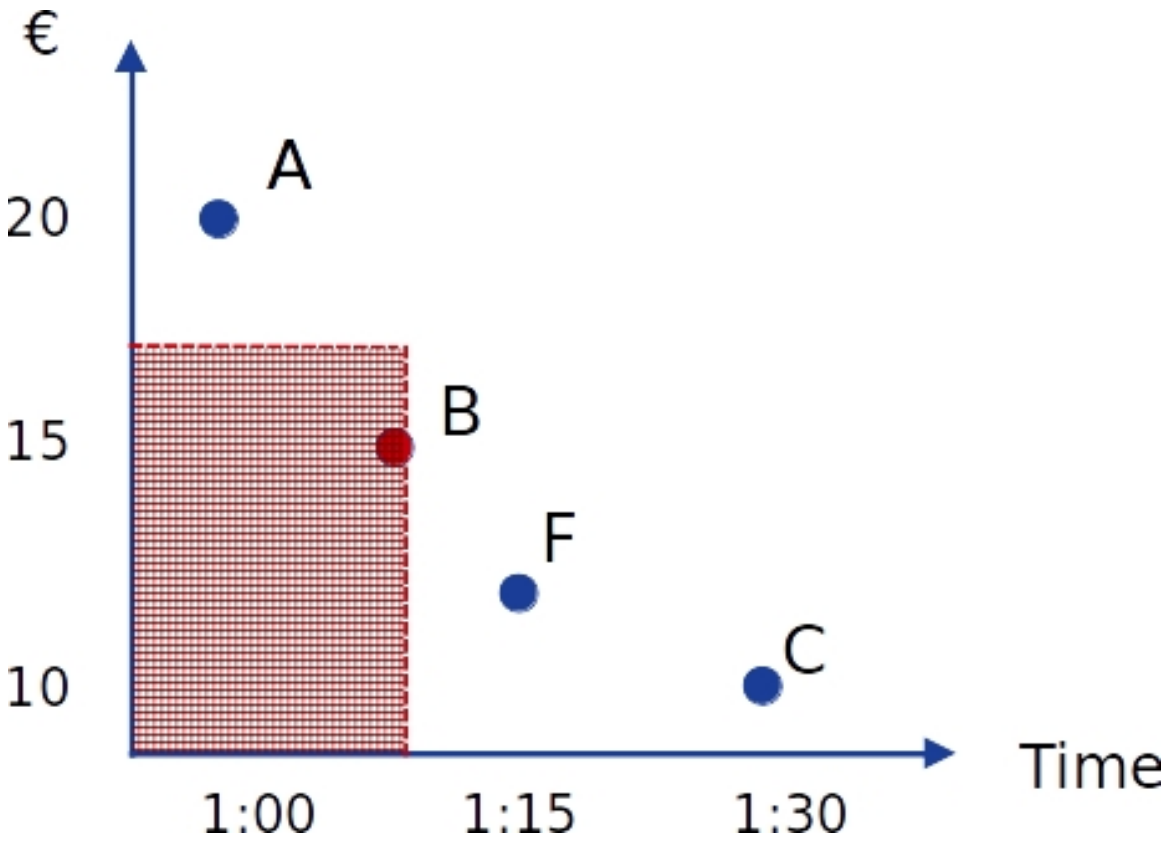
Lexicographic goals



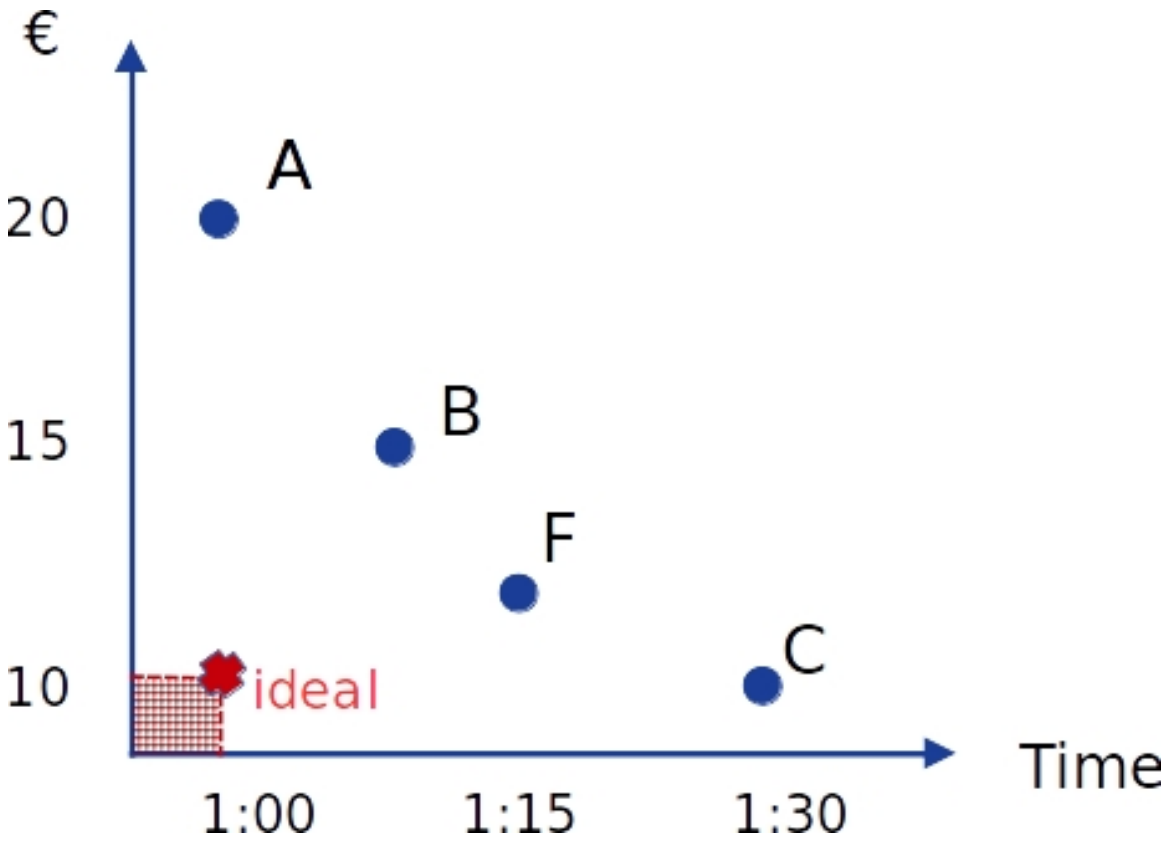
Lexicographic goals



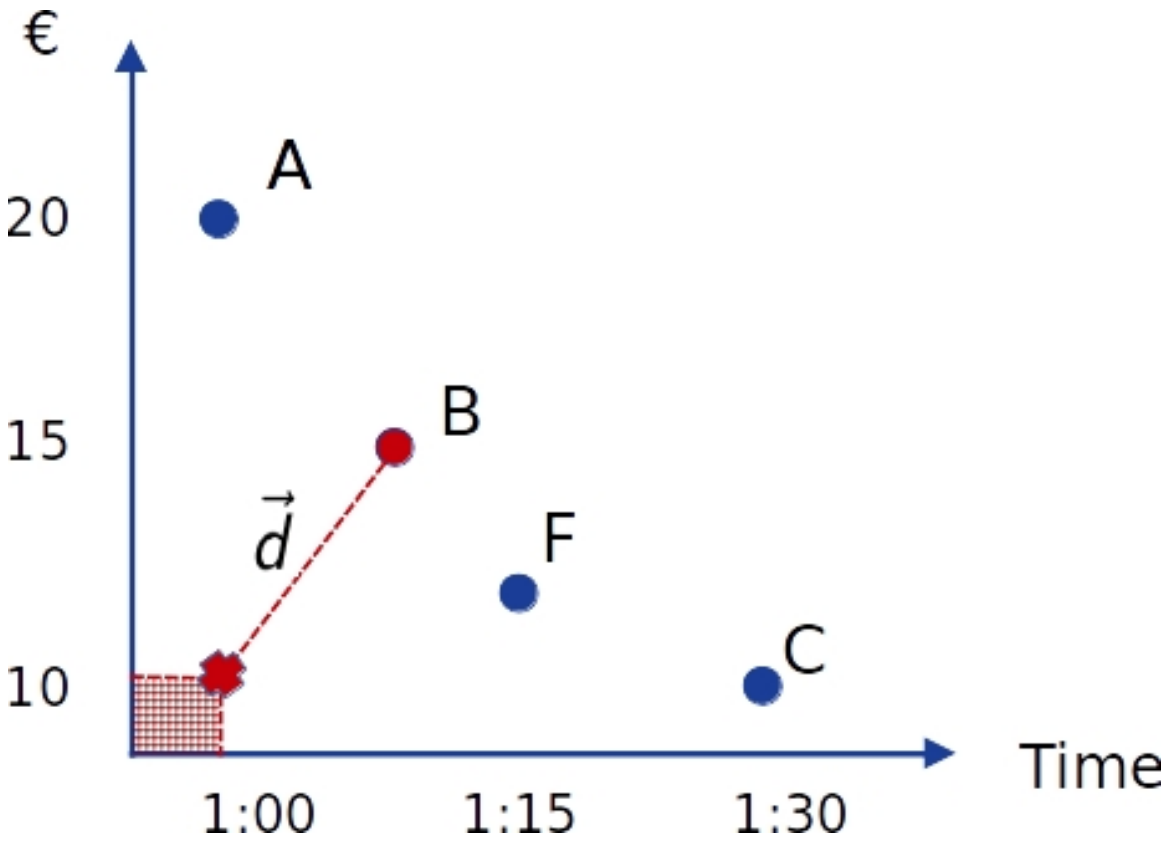
Lexicographic goals



Lexicographic goals



Lexicographic goals



Definition

We define **lexicographic goal preferences** ($\prec_G$) as a partial order relation,

$$\vec{y} \prec_G \vec{y'} \quad \Leftrightarrow \quad \vec{d}(\vec{y}) \prec_L \vec{d}(\vec{y'}) \quad \vee \quad (\vec{d}(\vec{y}) = \vec{d}(\vec{y'}) \quad \wedge \quad \vec{y} \prec \vec{y'})$$

MSP with lexicographic goals

Definition

The solution to a MSP with lexicographic goals is the set of non-dominated solution paths that satisfy the goals, or the subset that minimizes deviation if these cannot be fully satisfied.

Outline

Research goals

Tackle the MSP with lexicographic goals from two algorithmic perspectives:

Research goals

Tackle the MSP with lexicographic goals from two algorithmic perspectives:

A posteriori Preferences of the decision maker are provided after the execution of the algorithm.

A priori Preferences of the decision maker are provided before the execution of the algorithm.

Research goals

Tackle the MSP with lexicographic goals from two algorithmic perspectives:

A posteriori Preferences of the decision maker are provided after the execution of the algorithm.

A priori Preferences of the decision maker are provided before the execution of the algorithm.

- **A posteriori** algorithms obtain the whole Pareto frontier and then, extract the goal-optimal solutions.
- **A priori** algorithms will return only goal-optimal solutions.

Outline

Properties of multicriteria algorithms

Definition

An algorithm is admissible if it guarantees to find the optimal solution to the problem.

Properties of multicriteria algorithms

Definition

An algorithm is admissible if it guarantees to find the optimal solution to the problem.

Definition

We will measure the efficiency of algorithms according to the number of explored labels to find the solution.

Outline

I. Introduction

II. State of the art
- A posteriori algorithms
- A priori algorithms

III. Algorithmic contributions

IV. Empirical analyses

V. Conclusions & future work

A posteriori algorithms

Extensions of A^* to the multiobjective case to calculate the Pareto frontier:

MOA*	(Stewart & White, 1991)
TC	(Tung & Chew, 1992)
NAMOA*	(Mandow & Pérez de la Cruz, 2005)

All can be used with heuristic functions as lower bounds

A posteriori algorithms

Extensions of A^* to the multiobjective case to calculate the Pareto frontier:

MOA*	(Stewart & White, 1991) $\rightarrow$ pathological behaviour
TC	(Tung & Chew, 1992) $\rightarrow$ less efficient than NAMOA*
NAMOA*	(Mandow & Pérez de la Cruz, 2005)

All can be used with heuristic functions as lower bounds

NAMOA* properties

Analogously to A^*:

- Admissible when provided with a consistent heuristic function.
- It explores an optimal number of labels in its class.
- It improves its efficiency with more informed lower bounds.

NAMOA*

Relevant features of NAMOA*:

- Label selection policy
- Two sets of labels for each node n: $G_{op}(n)$ and $G_{cl}(n)$.
- A set COSTS of solutions.
- Discarding rules of dominated paths

Label selection policy

NAMOA* can be used with any path selection policy that assures the best label according to that policy is a non-dominated label.

Label selection policy

NAMOA* can be used with any path selection policy that assures the best label according to that policy is a non-dominated label.

- Lexicographic order
- Linear aggregation order

Label selection policy

NAMOA* can be used with any path selection policy that assures the best label according to that policy is a non-dominated label.

- Lexicographic order

$$\vec{y} \prec_L \vec{y'} \quad \Leftrightarrow \quad \exists j \ (1 \leq \ i \leq \ q) \ y_j < y'_j \ \wedge \ \forall i < j \ \ y_i = y'_i.$$

- Linear aggregation order

Label selection policy

NAMOA* can be used with any path selection policy that assures the best label according to that policy is a non-dominated label.

- Lexicographic order

$$\vec{y} \prec_L \vec{y'} \quad \Leftrightarrow \quad \exists j \ (1 \leq \ i \leq \ q) \ y_j < y'_j \ \wedge \ \forall i < j \ \ y_i = y'_i.$$

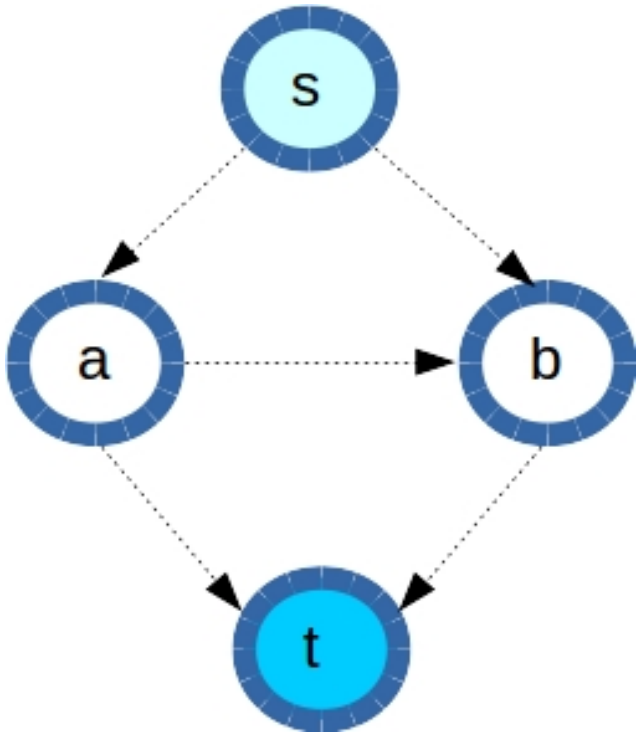
- Linear aggregation order

$$\vec{y} \prec_{lin} \vec{y'} \ \Leftrightarrow \ \sum_i y_i \ < \ \sum_i y'_i \ , \ 1 \ \leq \ i \ \leq \ q$$

Discarding dominated paths on NAMOA*

NAMOA* discards early in the search those paths that will not lead to non-dominated solutions.

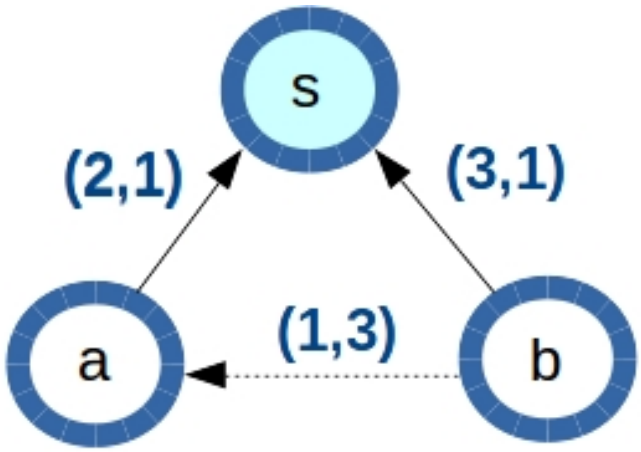
- Op-pruning
- Cl-pruning
- Filtering



Discarding dominated paths on NAMOA*

NAMOA* discards early in the search those paths that will not lead to non-dominated solutions.

- Op-pruning
- Cl-pruning
- Filtering



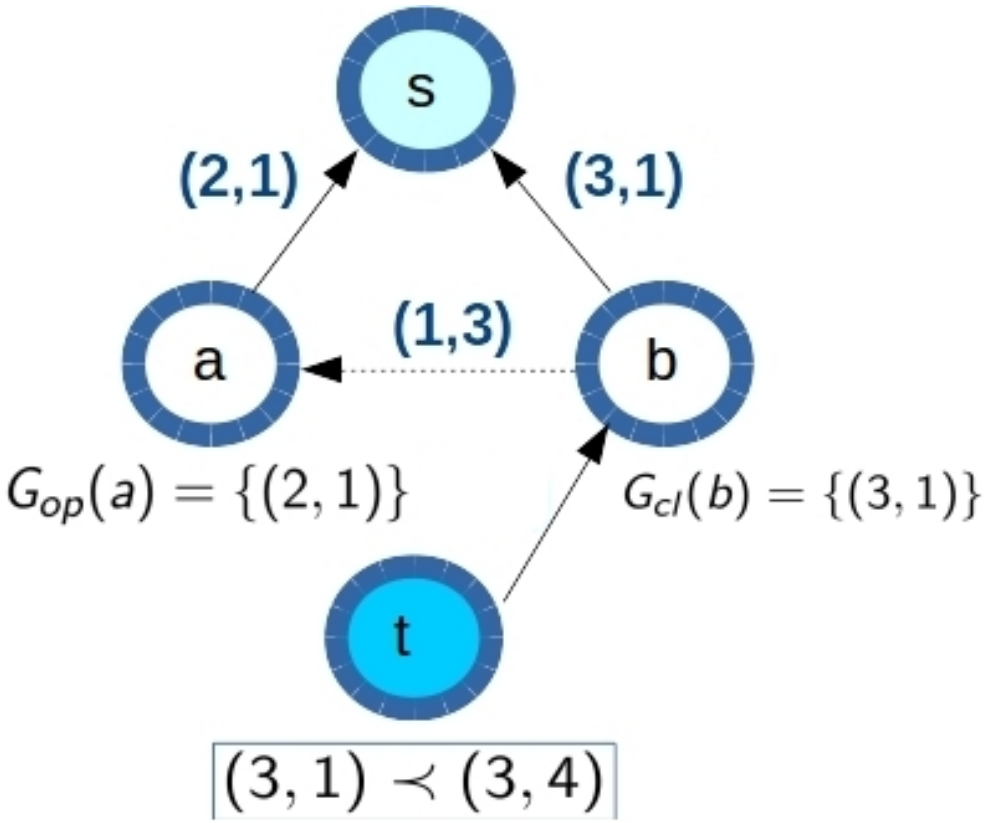
$G_{op}(a) = \{(2,1)\}$ $G_{op}(b) = \{(3,1)\}$

$(3,1) \prec (3,4)$

Discarding dominated paths on NAMOA*

NAMOA* discards early in the search those paths that will not lead to non-dominated solutions.

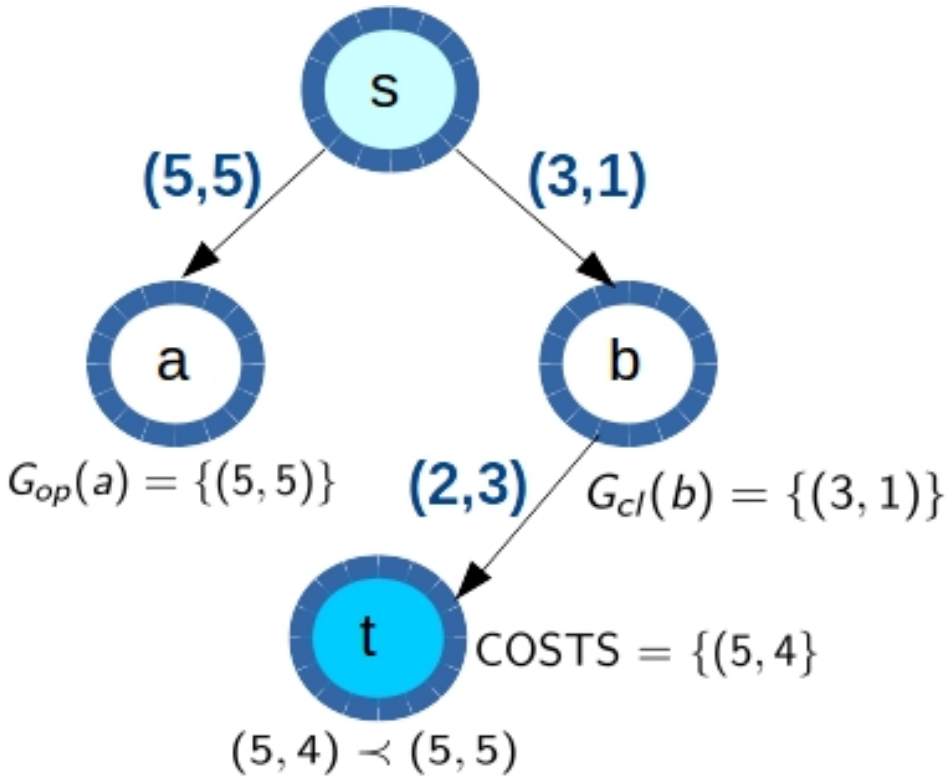
- Op-pruning
- Cl-pruning
- Filtering



Discarding dominated paths on NAMOA*

NAMOA* discards early in the search those paths that will not lead to non-dominated solutions.

- Op-pruning
- Cl-pruning
- Filtering



Outline

A priori algorithms

Is there any specifically devised algorithm for lexicographic goals?

METAL-A* (Mandow & Pérez de la Cruz, 2001)
Based on MOA* $\longrightarrow$ pathological behaviour

A priori algorithms

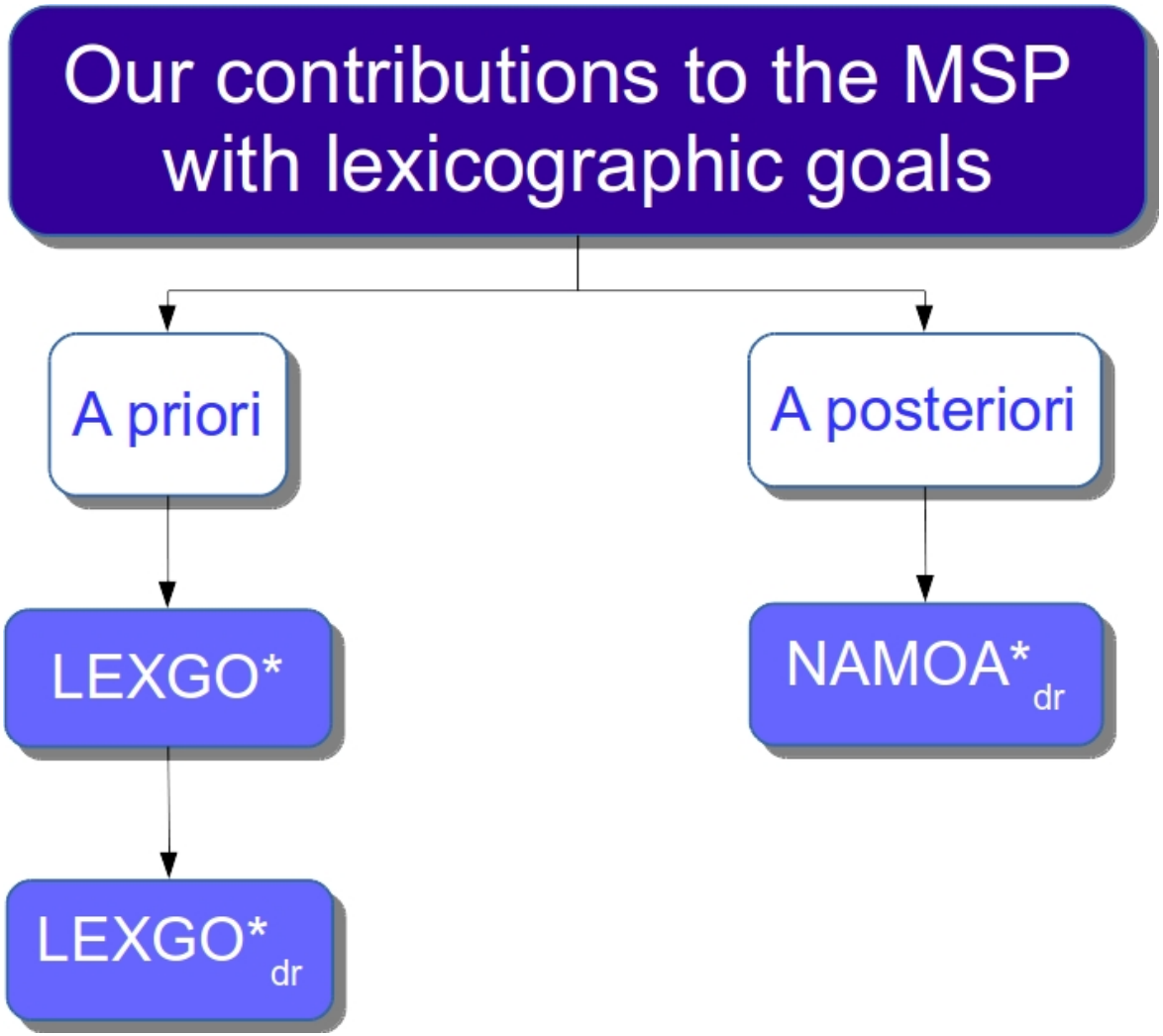
Is there any specifically devised algorithm for lexicographic goals?

METAL-A* (Mandow & Pérez de la Cruz, 2001)
Based on MOA* $\longrightarrow$ pathological behaviour

$\longrightarrow$ Then, let's propose a new approach based on NAMOA*

Outline

Algorithmic contributions



Outline

I. Introduction

II. State of the art

III. Algorithmic contributions

- LEXGO*
- T-discarding
- NAMOA$^*_{dr}$
- LEXGO$^*_{dr}$

IV. Empirical analyses

V. Conclusions & future work

Lexicographic goal preferences

Important!

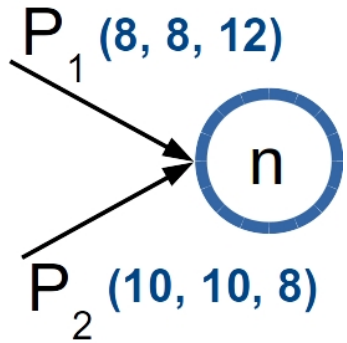
Bellman's Principle of Optimality, **an optimal path is made up of suboptimal paths**, holds for multiobjective search problems, but it does not hold for lexicographic goal preferences!

Lexicographic goal preferences

Example

Two paths P_1 and P_2 reach some node n with costs $\vec{g}(P_1) = (8, 8, 12)$ and $\vec{g}(P_2) = (10, 10, 8)$, respectively. The user goals are defined as follows:

$$\begin{array}{lll} \text{Level 1:} & g_1 \leq 10, & w_1 = 0.5 \\ & g_2 \leq 10, & w_2 = 0.5 \\ \text{Level 2:} & g_3 \leq 10, & w_3 = 1. \end{array}$$



Example

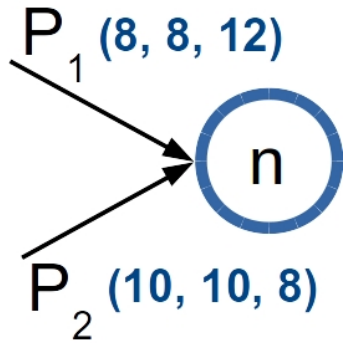
The deviation from goals for P_1 and P_2 is $\vec{d}(P_1) = (0, 2)$ and $\vec{d}(P_2) = (0, 0)$, respectively. Thus, $P_2 \prec_G P_1$.

Lexicographic goal preferences

Example

Two paths P_1 and P_2 reach some node n with costs $\vec{g}(P_1) = (8, 8, 12)$ and $\vec{g}(P_2) = (10, 10, 8)$, respectively. The user goals are defined as follows:

$$\begin{array}{lll} \text{Level 1:} & g_1 \leq 10, & w_1 = 0.5 \\ & g_2 \leq 10, & w_2 = 0.5 \\ \text{Level 2:} & g_3 \leq 10, & w_3 = 1. \end{array}$$



Example

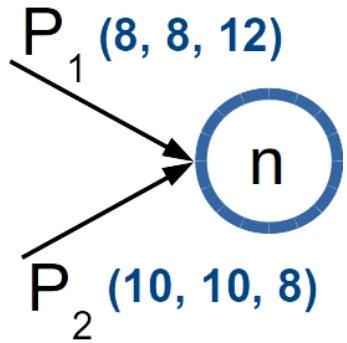
The deviation from goals for P_1 and P_2 is $\vec{d}(P_1) = (0, 2)$ and $\vec{d}(P_2) = (0, 0)$, respectively. Thus, $P_2 \prec_G P_1$.

Lexicographic goal preferences

Example

Two paths P_1 and P_2 reach some node n with costs $\vec{g}(P_1) = (8, 8, 12)$ and $\vec{g}(P_2) = (10, 10, 8)$, respectively. The user goals are defined as follows:

$$\begin{array}{lll} \text{Level 1:} & g_1 \leq 10, & w_1 = 0.5 \\ & g_2 \leq 10, & w_2 = 0.5 \\ \text{Level 2:} & g_3 \leq 10, & w_3 = 1. \end{array}$$



Example

The deviation from goals for P_1 and P_2 is $\vec{d}(P_1) = (0, 2)$ and $\vec{d}(P_2) = (0, 0)$, respectively. Thus, $P_2 \prec_G P_1$.

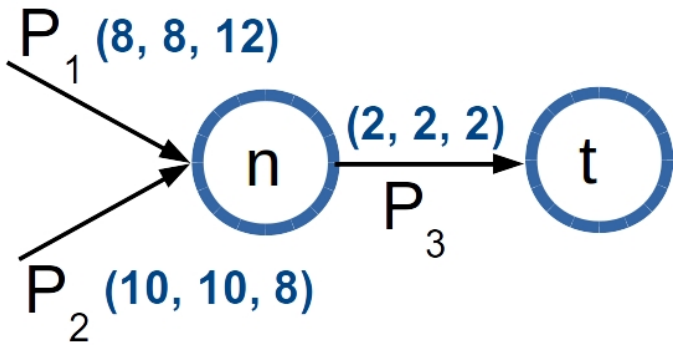
Lexicographic goal preferences

Example

Let's now consider there is only one additional path P_3 to the destination node t, such as $\vec{g}(P_3)) = (2, 2, 2)$.
Therefore,

- $\vec{g}(P_1P_3)) = (10, 10, 14)$ and $\vec{g}(P_2P_3)) = (12, 12, 10)$.
- $\vec{d}(P_1P_3) = (0, 4)$ and $\vec{d}(P_2P_3) = (2, 0)$.
- Now, $P_1P_3 \prec_G P_2P_3$

We have pruned an optimal path!



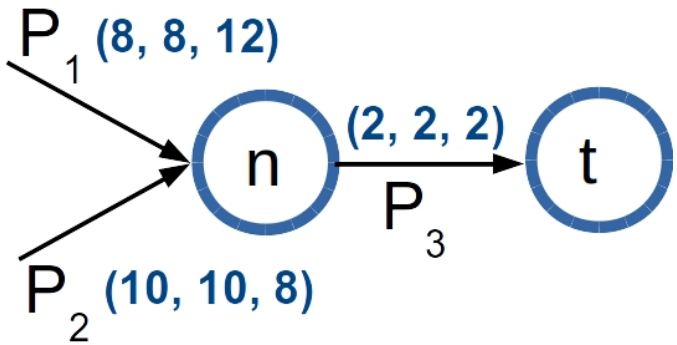
Lexicographic goal preferences

Example

Let's now consider there is only one additional path P_3 to the destination node t, such as $\vec{g}(P_3)) = (2, 2, 2)$.
Therefore,

- $\vec{g}(P_1P_3)) = (10, 10, 14)$ and $\vec{g}(P_2P_3)) = (12, 12, 10)$.
- $\vec{d}(P_1P_3) = (0, 4)$ and $\vec{d}(P_2P_3) = (2, 0)$.
- Now, $P_1P_3 \prec_G P_2P_3$

We have pruned an optimal path!



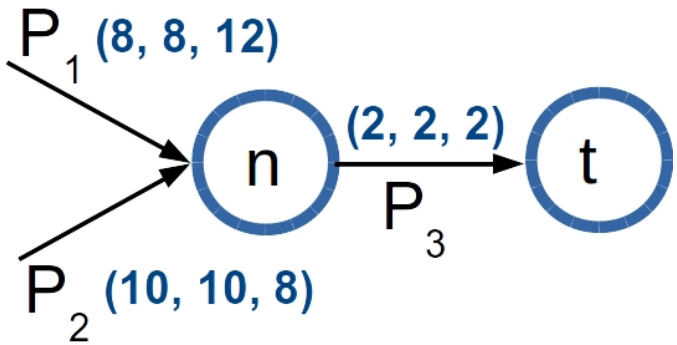
Lexicographic goal preferences

Example

Let's now consider there is only one additional path P_3 to the destination node t, such as $\vec{g}(P_3)) = (2, 2, 2)$.
Therefore,

- $\vec{g}(P_1P_3)) = (10, 10, 14)$ and $\vec{g}(P_2P_3)) = (12, 12, 10)$.
- $\vec{d}(P_1P_3) = (0, 4)$ and $\vec{d}(P_2P_3) = (2, 0)$.
- Now, $P_1P_3 \prec_G P_2P_3$

We have pruned an optimal path!

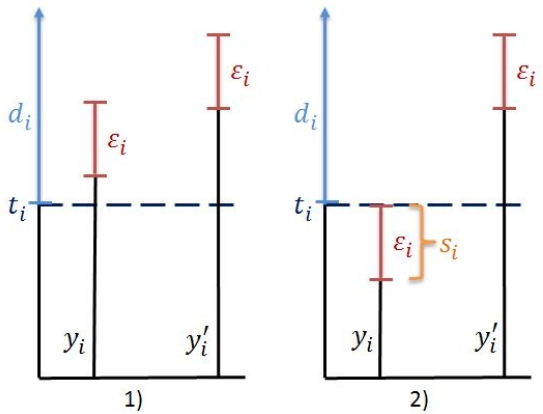


Lexicographic goal preferences

Example

Let's now assume paths P_1 and P_2 have costs $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, respectively. The user goals are now *different* and defined as follows:

$$\begin{aligned} \text{Level 1:} \quad & g_1 \leq 10, \quad w_1 = 1 \\ \text{Level 2:} \quad & g_2 \leq 10, \quad w_2 = 0.5 \\ & g_3 \leq 10, \quad w_3 = 0.5. \end{aligned}$$



Example

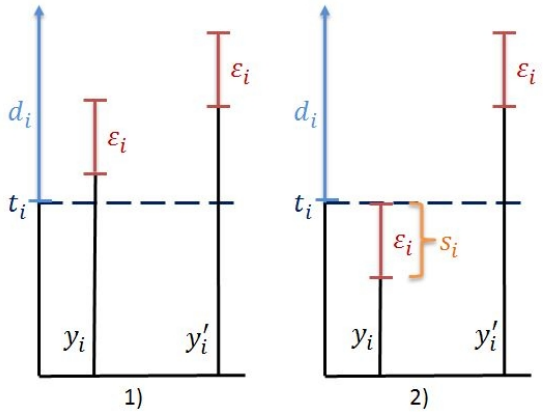
The deviation from goals for P_1 and P_2 is now $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$, respectively. $P_2 \prec_G P_1$, but, can P_1 be safely pruned?

Lexicographic goal preferences

Example

Let's now assume paths P_1 and P_2 have costs $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, respectively. The user goals are now *different* and defined as follows:

$$\begin{aligned} \text{Level 1:} \quad & g_1 \leq 10, \quad w_1 = 1 \\ \text{Level 2:} \quad & g_2 \leq 10, \quad w_2 = 0.5 \\ & g_3 \leq 10, \quad w_3 = 0.5. \end{aligned}$$



Example

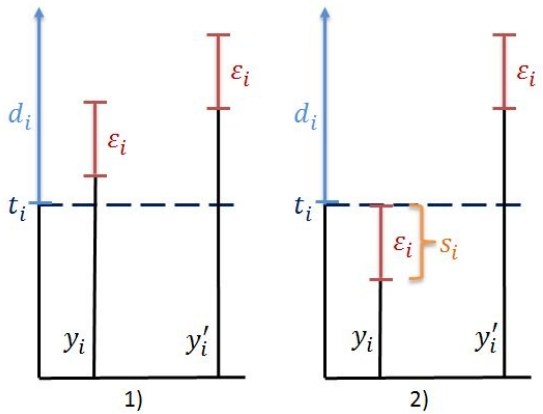
The deviation from goals for P_1 and P_2 is now $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$, respectively. $P_2 \prec_G P_1$, but, can P_1 be safely pruned?

Lexicographic goal preferences

Example

Let's now assume paths P_1 and P_2 have costs $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, respectively. The user goals are now *different* and defined as follows:

$$\begin{aligned} \text{Level 1:} \quad & g_1 \leq 10, \quad w_1 = 1 \\ \text{Level 2:} \quad & g_2 \leq 10, \quad w_2 = 0.5 \\ & g_3 \leq 10, \quad w_3 = 0.5. \end{aligned}$$



Example

The deviation from goals for P_1 and P_2 is now $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$, respectively. $P_2 \prec_G P_1$, but, can P_1 be safely pruned?

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1.$
- $s_3(P_2) = (10 - 6) \times 0.5 = 2.$
- $\delta_2(P_1, P_2) = 2 - 1 = 1,$

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1.$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1.$
- $s_3(P_2) = (10 - 6) \times 0.5 = 2.$
- $\delta_2(P_1, P_2) = 2 - 1 = 1,$

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1.$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$,
where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1.$
- $s_3(P_2) = (10 - 6) \times 0.5 = 2.$
- $\delta_2(P_1, P_2) = 2 - 1 = 1,$

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1.$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1$.
- $s_3(P_2) = (10 - 6) \times 0.5 = 2$.
- $\delta_2(P_1, P_2) = 2 - 1 = 1$,

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1$.

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1.$
- $s_3(P_2) = (10 - 6) \times 0.5 = 2.$
- $\delta_2(P_1, P_2) = 2 - 1 = 1,$

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1.$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

Slack variable

$$s_k = \max(0, t_k - y_k)$$

Cross slacks

$$\delta_j(\vec{y}, \vec{y'}) = \sum_{k \in I_j} w_k \times \max(0, s'_k - s_k)$$

Example

Given $\vec{g}(P_1) = (12, 8, 16)$ and $\vec{g}(P_2) = (12, 12, 6)$, where $\vec{d}(P_1) = (2, 3)$ and $\vec{d}(P_2) = (2, 1)$:

- $s_2(P_1) = (10 - 8) \times 0.5 = 1.$
- $s_3(P_2) = (10 - 6) \times 0.5 = 2.$
- $\delta_2(P_1, P_2) = 2 - 1 = 1,$

Finally, $d_2(P_2) - d_2(P_1) = 3 - 1 = 2 > 1.$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

$$\vec{y} \prec_P \vec{y'} \quad \Leftrightarrow \quad \forall i < j \;\; (d_i(\vec{y}) = d_i(\vec{y'}) \quad \wedge \quad \delta_i(\vec{y}, \vec{y'}) = 0)) \quad \wedge$$
$$\exists j \; (d_j(\vec{y}) < d_j(\vec{y'}) \quad \wedge \quad \delta_j(\vec{y}, \vec{y'}) < d_j(\vec{y'}) - d_j(\vec{y})$$

Deviation-based filtering

$$\vec{d_B} \prec_L \vec{d}$$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

$$\vec{y} \prec_P \vec{y'} \quad \Leftrightarrow \quad \forall i < j \;\; (d_i(\vec{y}) = d_i(\vec{y'}) \quad \wedge \quad \delta_i(\vec{y}, \vec{y'}) = 0)) \quad \wedge$$
$$\exists j \; (d_j(\vec{y}) < d_j(\vec{y'}) \quad \wedge \quad \delta_j(\vec{y}, \vec{y'}) < d_j(\vec{y'}) - d_j(\vec{y})$$

Deviation-based filtering

$$\vec{d_B} \prec_L \vec{d}$$

Discarding labels by deviation

Deviation-based pruning ($\prec_P$)

$$\vec{y} \prec_P \vec{y'} \quad \Leftrightarrow \quad \forall i < j \;\; (d_i(\vec{y}) = d_i(\vec{y'}) \quad \wedge \quad \delta_i(\vec{y}, \vec{y'}) = 0)) \quad \wedge$$
$$\exists j \; (d_j(\vec{y}) < d_j(\vec{y'}) \quad \wedge \quad \delta_j(\vec{y}, \vec{y'}) < d_j(\vec{y'}) - d_j(\vec{y})$$

Deviation-based filtering

$$\vec{d_B} \prec_L \vec{d}$$

LEXGO*

In addition to the pruning and filtering rules of NAMOA*, LEXGO* also uses its own rules to prune and filter paths by deviation.

LEXGO*

LEXGO* shares NAMOA* formal properties

- The new deviation-based pruning and filtering rules do not discard goal-optimal solutions.
- LEXGO* always returns the set of all goal-optimal solutions.
- LEXGO* expands a subset of the labels expanded by NAMOA*.

Outline

I. Introduction

II. State of the art

III. Algorithmic contributions

- LEXGO*
- T-discarding
- NAMOA*dr
- LEXGO*dr

IV. Empirical analyses

V. Conclusions & future work

T-discarding

Limiting factor in MSP

Runtime, rather than memory, is the limiting factor in multicriteria search algorithms performance, as well as in the size of problems that can be practically solved.

Limiting factor in MSP (II)

The majority of the algorithm runtime can be attributed to dominance checks against G_{op}, G_{cl} and COSTS to discard dominated paths.

T-discarding

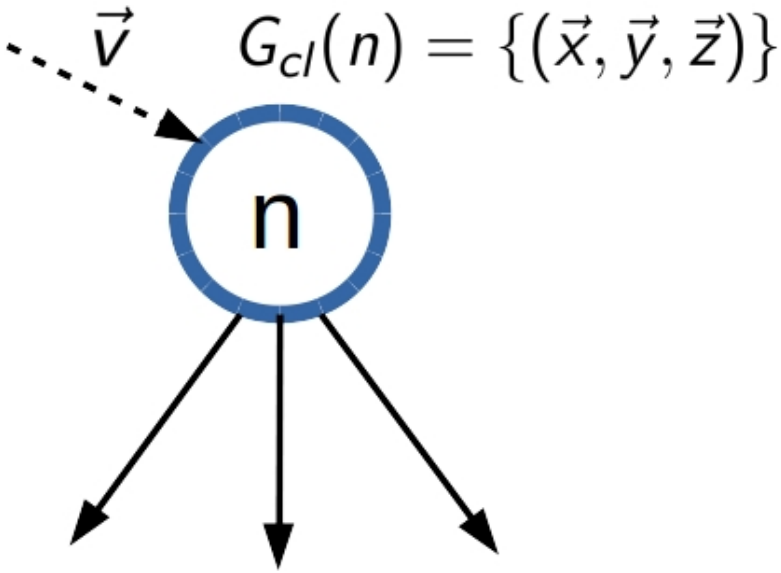
Let's suppose a typical scenario where a new path with cost $\vec{v}$ reaches some node n. This node has already three closed paths with costs $(\vec{x}, \vec{y}, \vec{z})$. Let's now check if $\vec{v}$ is dominated by any vector in $G_{cl}(n)$:

Example

- $\vec{x} \prec \vec{v}$? $\rightarrow$ $x_1 \leq v_1$, $x_2 \leq v_2$ ...
- $\vec{y} \prec \vec{v}$? $\rightarrow$ $y_1 \leq v_1$, $y_2 \leq v_2$ ...
- $\vec{z} \prec \vec{v}$? $\rightarrow$ $z_1 \leq v_1$, $z_2 \leq v_2$ ...

Worst-case scenario:

- 3 vector comparisons
- 9 scalar comparisons



T-discarding

Definition

T-discarding (*truncated discarding*) is a dimensionality reduction technique to speed up dominance checks.

T-discarding

Assumptions

To apply t-discarding an algorithm must have:

- Lexicographic order as label selection policy.
- One consistent heuristic function as lower bound.

T-discarding

Assumptions

To apply t-discarding an algorithm must have:

- Lexicographic order as label selection policy.
- One consistent heuristic function as lower bound.

Example

When lexicographic order is used to select open paths:

- $\rightarrow (3, 5, 7)$
- $\rightarrow (4, 6, 4)$
- $\rightarrow (4, 6, 7)$
- $\rightarrow (6, 2, 4)$

T-discarding

Assumptions

To apply t-discarding an algorithm must have:

- Lexicographic order as label selection policy.
- One consistent heuristic function as lower bound.

Example

When lexicographic order is used to select open paths:

- $\rightarrow$ (3, -, -)
- $\rightarrow$ (4, -, -)
- $\rightarrow$ (4, -, -)
- $\rightarrow$ (6, -, -)

T-discarding

Assumptions

To apply t-discarding an algorithm must have:

- Lexicographic order as label selection policy.
- One consistent heuristic function as lower bound.

Property

Assuming the last selected open path had cost $\vec{c} = c_1, c_2, ..., c_q$, the next path selected to be open with cost $\vec{c'} = c'_1, c'_2, ..., c'_q$ must have $c'_1 \geq c_1$!

T-discarding

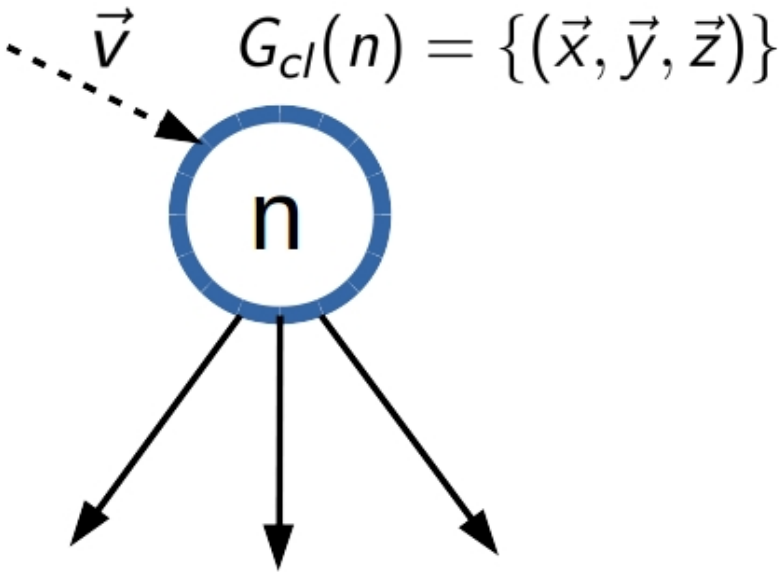
Let's suppose a typical scenario where a new path with cost $\vec{v}$ reaches some node n. This node has already three closed paths with costs $(\vec{x}, \vec{y}, \vec{z})$. Let's now check if $\vec{v}$ is dominated by any vector in $G_{cl}(n)$:

Example

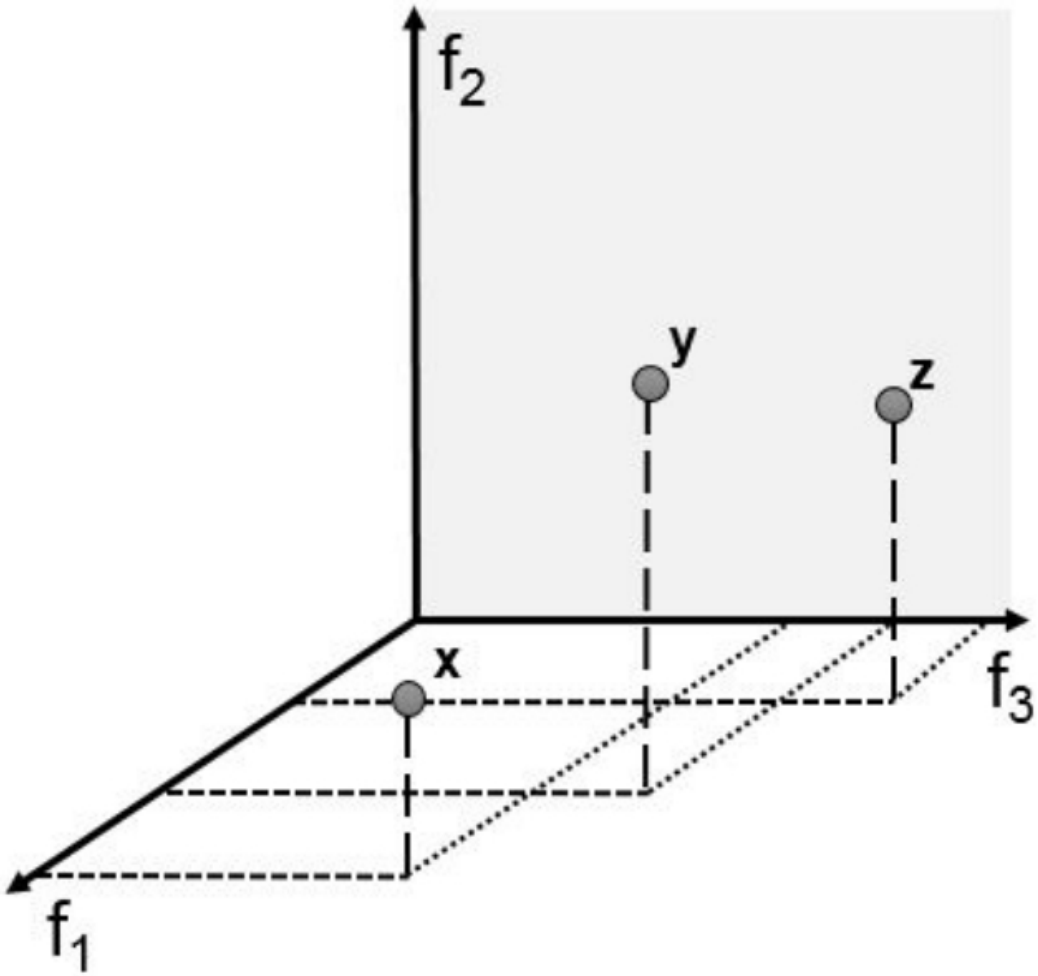
- $\vec{x} \prec \vec{v}$? $\rightarrow$ $x_1 \leq v_1$, $x_2 \leq v_2$ ...
- $\vec{y} \prec \vec{v}$? $\rightarrow$ $y_1 \leq v_1$, $y_2 \leq v_2$ ...
- $\vec{z} \prec \vec{v}$? $\rightarrow$ $z_1 \leq v_1$, $z_2 \leq v_2$ ...

Worst-case scenario:

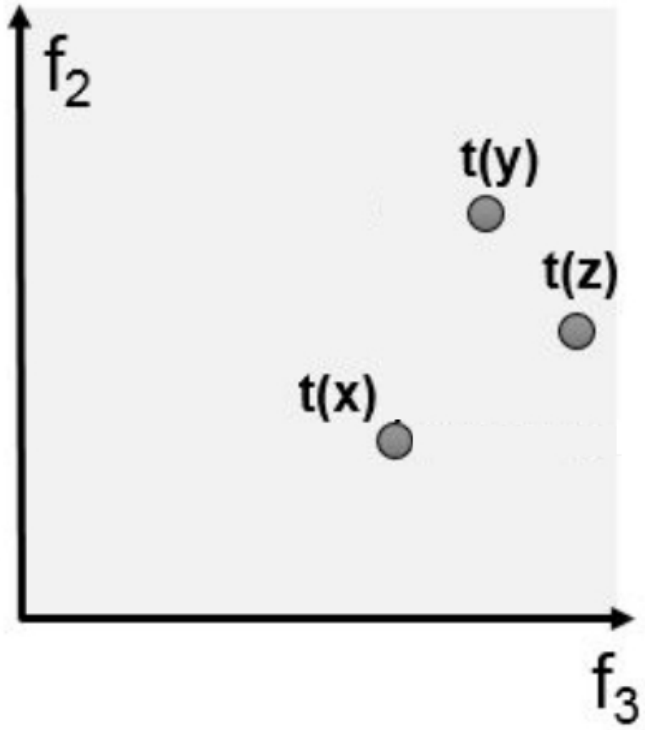
- 3 vector comparisons
- 9 scalar comparisons



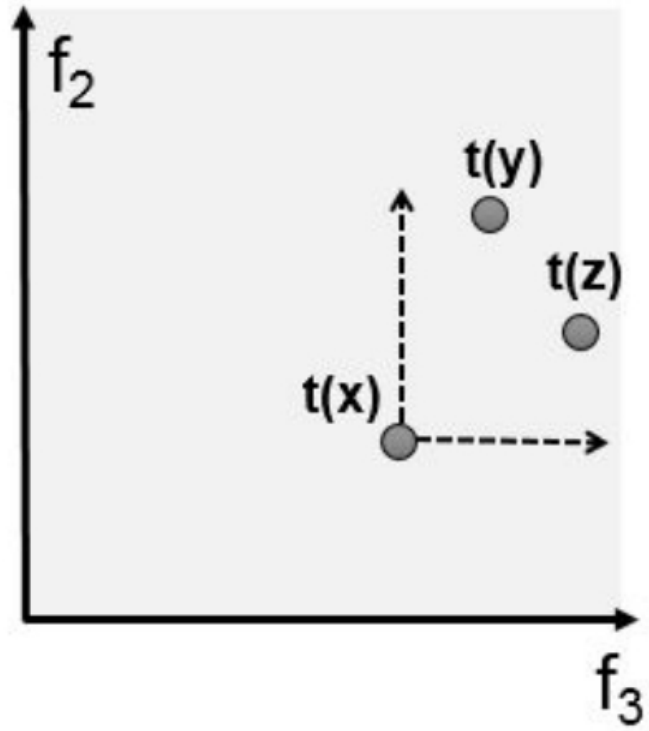
T-discarding



T-discarding



T-discarding



T-discarding

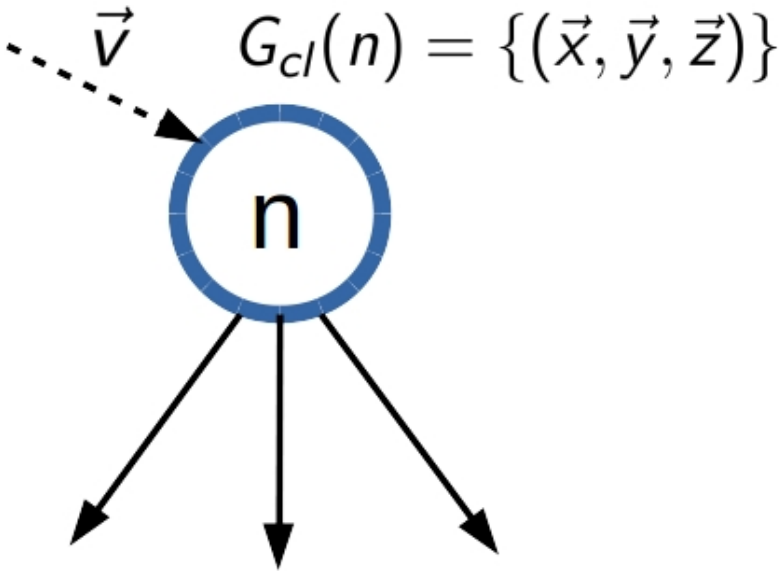
Let's suppose a typical scenario where a new path with cost $\vec{v}$ reaches some node n. This node has already three closed paths with costs $(\vec{x}, \vec{y}, \vec{z})$. Let's now check if $\vec{v}$ is dominated by any vector in $G_{cl}(n)$:

Example

- $\vec{x} \prec \vec{v}$? $\rightarrow$ $x_1 \leq v_1$, $x_2 \leq v_2$ ...
- $\vec{y} \prec \vec{v}$? $\rightarrow$ $y_1 \leq v_1$, $y_2 \leq v_2$ ...
- $\vec{z} \prec \vec{v}$? $\rightarrow$ $z_1 \leq v_1$, $z_2 \leq v_2$ ...

Worst-case scenario:

- 3 vector comparisons
- 9 scalar comparisons



T-discarding

Is a vector $\vec{v}$ dominated by any vector in the previous set $(\vec{x}, \vec{y}, \vec{z})$?

Standard discarding

$\vec{v}$ must be compared to $(\vec{x}, \vec{y}, \vec{z})$ to check if it is dominated.

T-discarding

The truncated vector $t(\vec{v})$ must be compared only to $t(\vec{x})$ to check if it is dominated.

T-discarding

Benefits of t-discarding

- Dominance checks can be done with vectors reduced in one dimension.
- The size of sets $T(G_{cl}(n))$ and $T(COSTS)$ is also reduced.

Important!

T-discarding only implies checking dominance against different sets:

- Very simple implementation.
- Any label discarded by standard pruning $\Longleftrightarrow$ is discarded by t-discarding.
- Does not affect any formal property of the algorithm.

Outline

I. Introduction

II. State of the art

III. Algorithmic contributions

- LEXGO*
- T-discarding
- NAMOA*$_{dr}$
- LEXGO*$_{dr}$

IV. Empirical analyses

V. Conclusions & future work

The applicability of t-discarding to NAMOA* is full and straightforward:

Cl-pruning and filtering are now performed against:

- $T(G_{cl}(n))$
- T(COSTS)

Outline

I. Introduction

II. State of the art

III. Algorithmic contributions

- LEXGO*
- T-discarding
- NAMOA$^*_{\text{dr}}$
- LEXGO$^*_{\text{dr}}$

IV. Empirical analyses

V. Conclusions & future work

The applicability of t-discarding to $LEXGO^*$ is partial:

Cl-pruning and filtering are now performed against:

- $T(G_{cl}(n))$
- T(COSTS)

but only when $\vec{d} = 0$.

Summary

	NAMOA*	LEXGO*	NAMOA*$_{dr}$	LEXGO*$_{dr}$
Op-pruning	$G_{op}(n)$	$G_{op}(n)$	$G_{op}(n)$	$G_{op}(n)$
Cl-pruning	$G_{cl}(n)$	$G_{cl}(n)$	$T(G_{cl}(n))$	$T(G_{cl}(n))$*
Filtering	COSTS	COSTS	T(COSTS)	T(COSTS)*

Outline

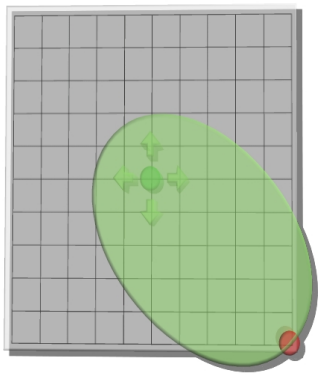
Empirical analyses

- Are these new algorithms effective?
- Will LEXGO* become faster when goals get more restrictive?
- Have we achieved improvements over NAMOA*? To what extent? Why?

Empirical analyses

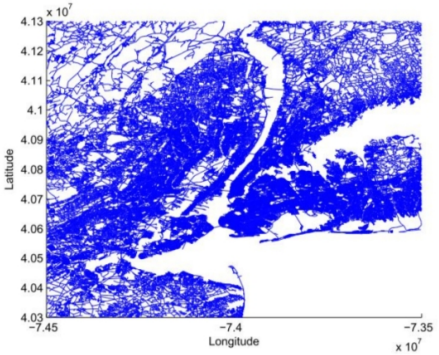
Random grids

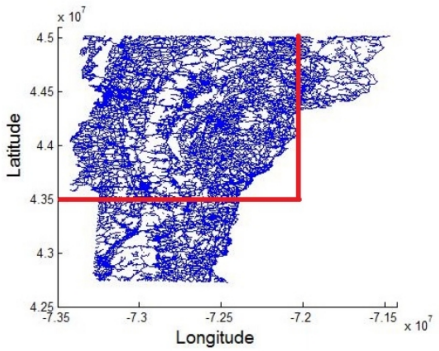
- 100×100
- Solution depth from 20 to 100
- Random costs in range [1,10]



Realistic Road maps

- 20 random problems over New York City & Vermont State from 9th DIMACS Challenge
- Three integer costs
  - Distance
  - Travel time
  - Economic cost





Outline

A priori

LEXGO* vs NAMOA*

LEXGO* vs NAMOA*

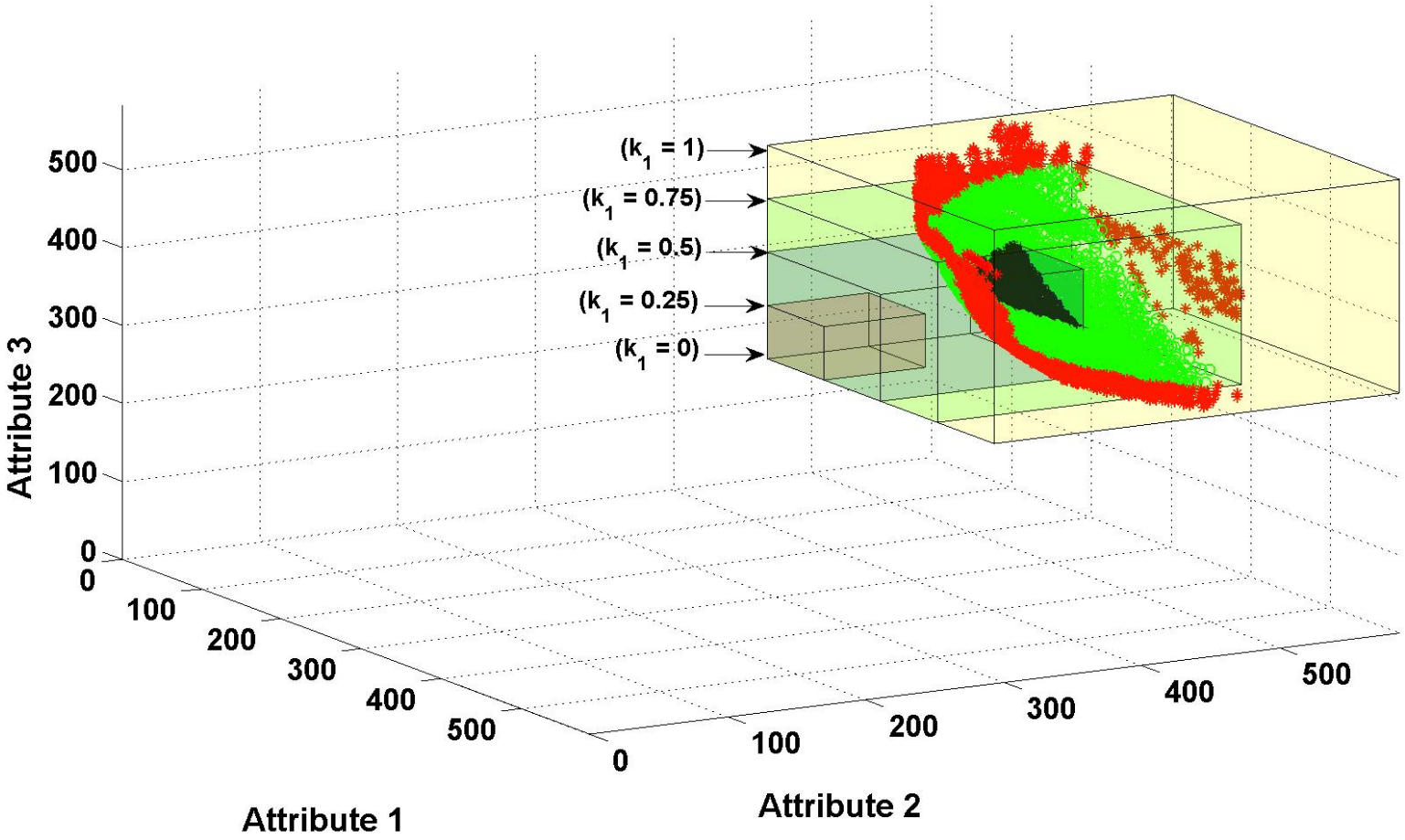


Figure: 3D Pareto frontier according to goal satisfiability.

LEXGO* vs NAMOA*

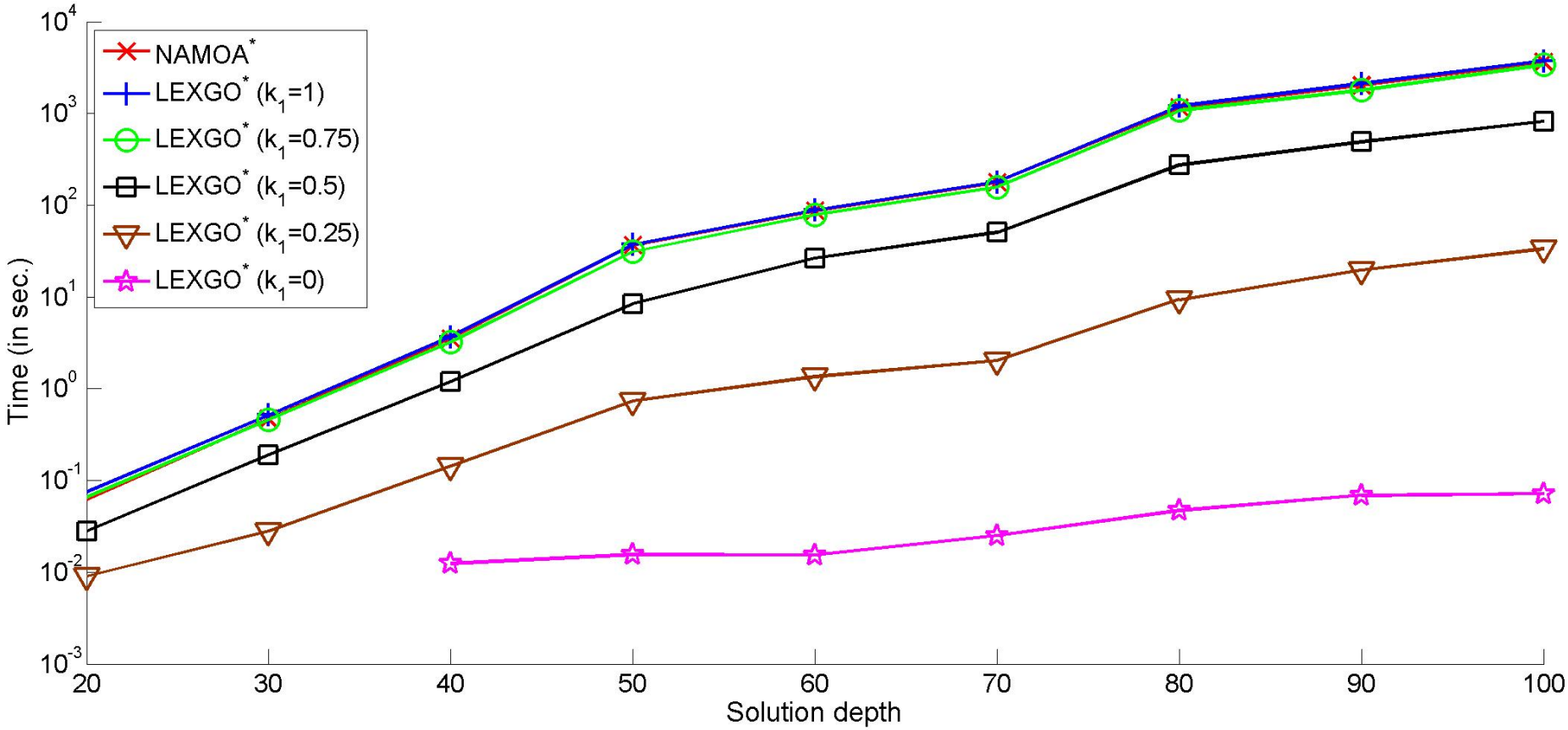


Figure: Runtimes of NAMOA* and LEXGO* in seconds (logarithmic scale).

LEXGO* vs NAMOA*

Table: Runtimes for $d = 100$ experiments.

	LEXGO*$_{\text{lex}}$					
NAMOA*$_{\text{lex}}$	1	0.75	0.5	0.25	0	k_1
Runtime (s)	%	%	%	%	%	
3,662.9	102.7	92.3	22.3	0.9	0.001	

A priori

LEXGO*	vs	NAMOA*
$\text{LEXGO}^*_{\text{dr}}$	vs	LEXGO*

$\text{LEXGO}^*_{\text{dr}}$ vs LEXGO^*

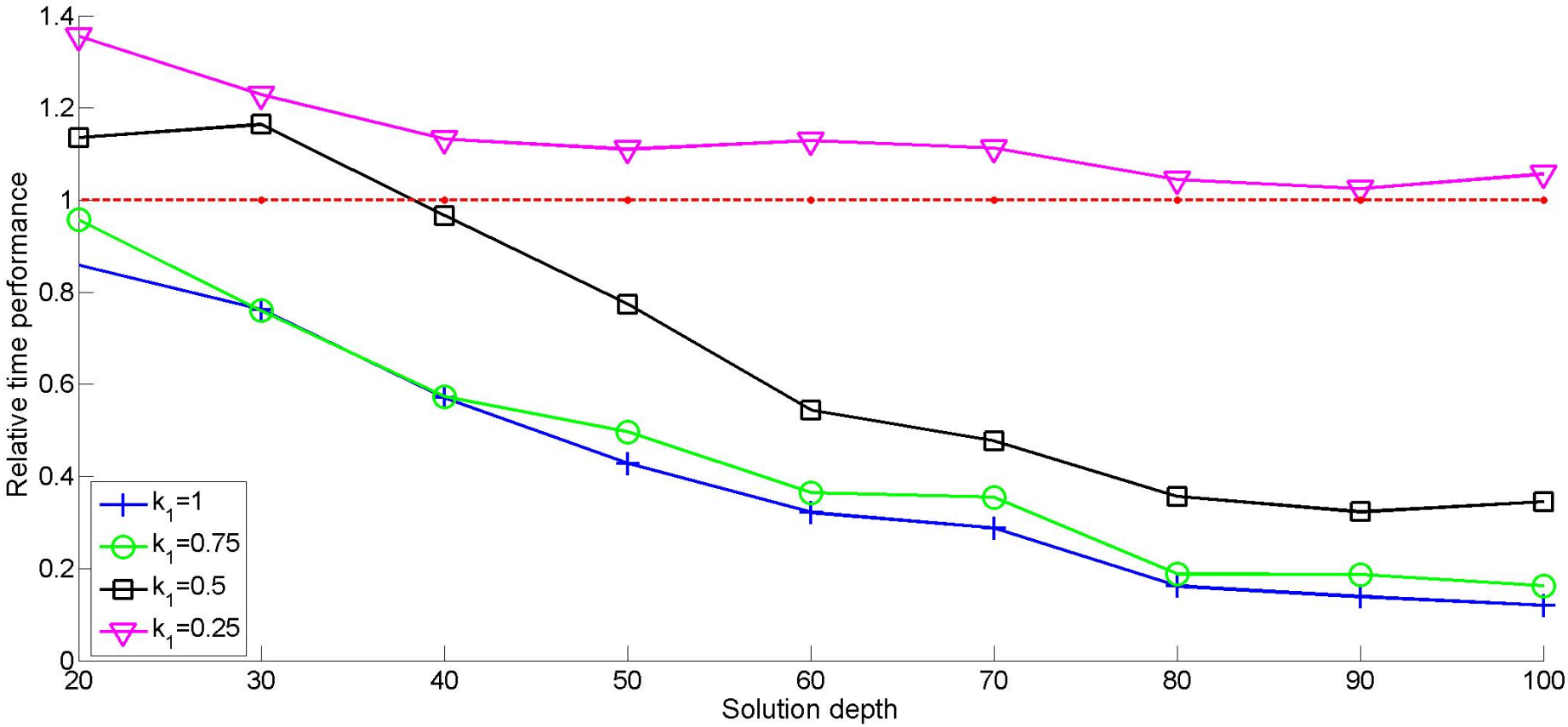


Figure: Relative runtime performance of $\text{LEXGO}^*_{\text{dr}}$ over LEXGO^*.

$\text{LEXGO}^*_{\text{dr}}$ vs LEXGO^*

Table: Runtimes in seconds for $d = 100$ experiments.

k_1	$\text{LEXGO}^*_{\text{lex}}$	$\text{LEXGO}^*_{\text{dr}}$	Speedup
1	3,763.78	254.40	14.81
0.75	3,381.33	300.86	11.27
0.5	819.28	282.49	2.90
0.25	33.47	35.37	0.94
0	0.07	0.10	0.7

A posteriori

$NAMOA^*_{dr}$ vs $NAMOA^*$

$NAMOA^*_{dr}$ vs $NAMOA^*$

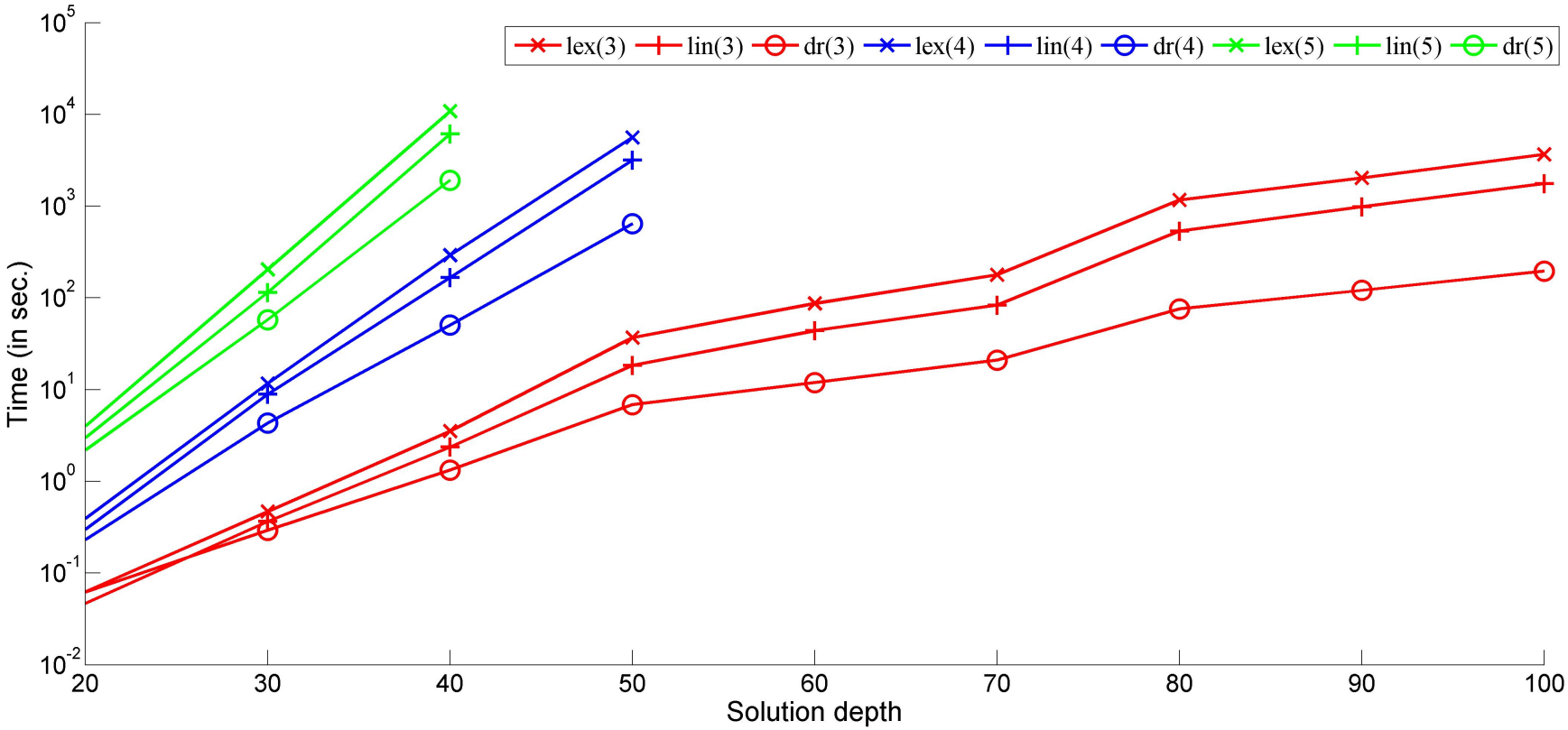


Figure: Runtimes for $NAMOA^*_{lex}$, $NAMOA^*_{lin}$, and $NAMOA^*_{dr}$ with $q = \{3, 4, 5\}$

A posteriori vs A priori

$\text{NAMOA}^*_{\text{dr}}$ vs $\text{LEXGO}^*_{\text{dr}}$

$\text{LEXGO}^*_{\text{dr}}$ vs $\text{NAMOA}^*_{\text{dr}}$

Table: Runtimes of $\text{LEXGO}^*_{\text{dr}}$ and $\text{NAMOA}^*_{\text{dr}}$.

	$\text{LEXGO}^*_{\text{dr}}$				
$\text{NAMOA}^*_{\text{dr}}$ Runtime (s)	$(k_1 = 1)$	$(k_1 = 0.75)$	$(k_1 = 0.5)$	$(k_1 = 0.25)$	$(k_1 = 0)$
100%	129.7%	153.4%	144%	18%	0.04%

Outline

A priori

LEXGO* vs NAMOA*

LEXGO* vs NAMOA*

Table: Experiments for Vermont State map.

	LEXGO*$_{\text{lex}}$					
NAMOA*$_{\text{lex}}$	1	0.75	0.5	0.25	0	k_1
Avg. runtime (s)	%	%	%	%	%	
5,048.47	100.39	74.67	29.06	3.51	0.01	

A priori

LEXGO*	vs	NAMOA*
$\text{LEXGO}^*_{\text{dr}}$	vs	LEXGO*

$\text{LEXGO}^*_{\text{dr}}$ vs LEXGO^*

Table: Runtimes in seconds for Vermont State map.

	1	0.75	0.5	0.25	0	k_1
$\text{LEXGO}^*_{\text{lex}}$	5,068.00	3,769.51	1,467.00	177.33	0.03	
$\text{LEXGO}^*_{\text{dr}}$	94.90	83.59	57.38	157.02	0.03	

A posteriori

$NAMOA^*_{dr}$ vs $NAMOA^*$

$NAMOA^*_{dr}$ vs $NAMOA^*$

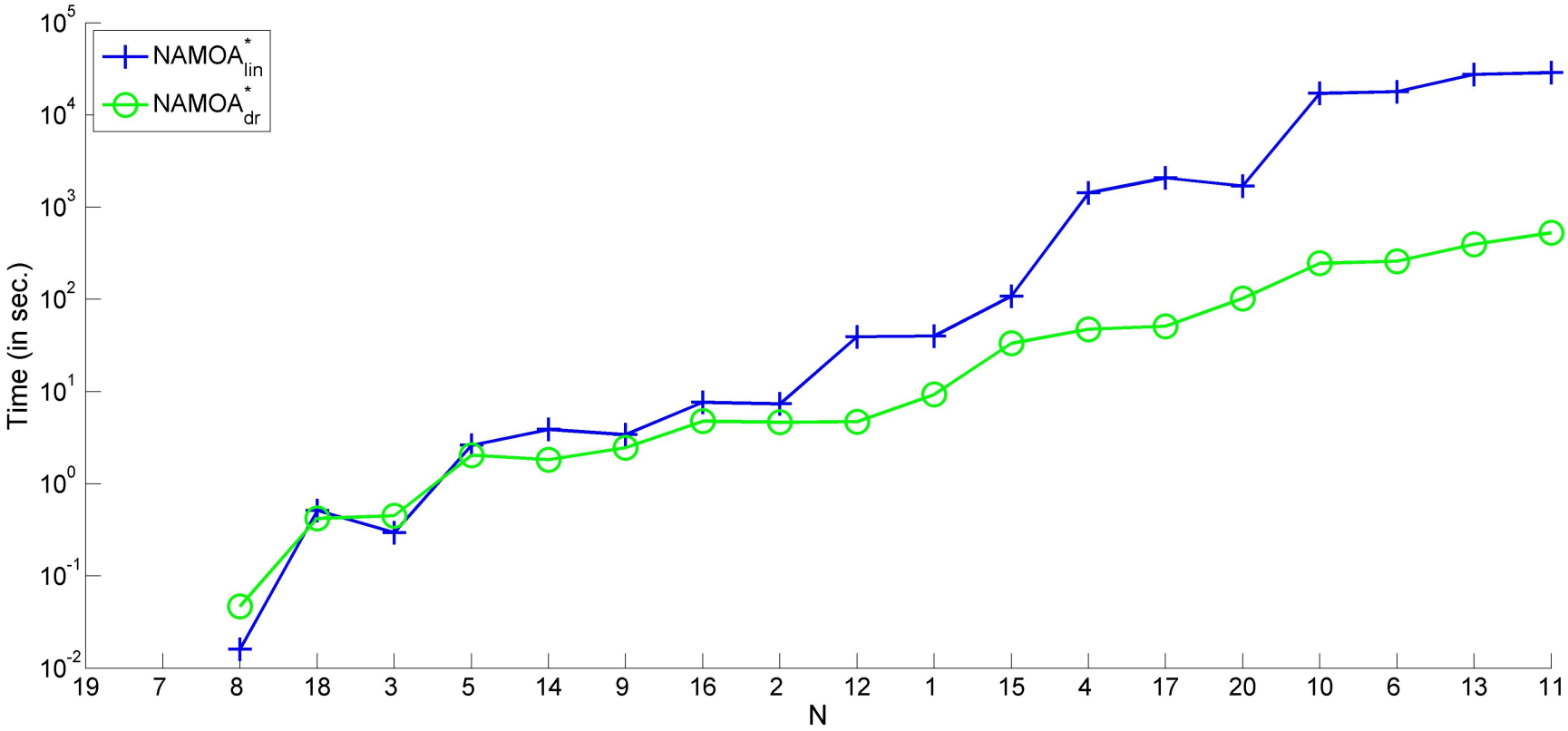


Figure: Runtimes of $NAMOA^*_{lin}$ and $NAMOA^*_{dr}$ for Vermont map problems.

A posteriori vs A priori

$\text{NAMOA}^*_{\text{dr}}$ vs $\text{LEXGO}^*_{\text{dr}}$

$LEXGO^*_{dr}$ vs $NAMOA^*_{dr}$

Table: Runtimes in seconds of $LEXGO^*_{dr}$ and $NAMOA^*_{dr}$ in Vermont State map problems.

	$LEXGO^*_{dr}$					
$NAMOA^*_{dr}$	1	0.75	0.5	0.25	0	k_1
84.66	94.89	83.59	57.38	157.02	0.03	

Why do the new algorithms have such a runtime performance?

LEXGO* vs NAMOA*

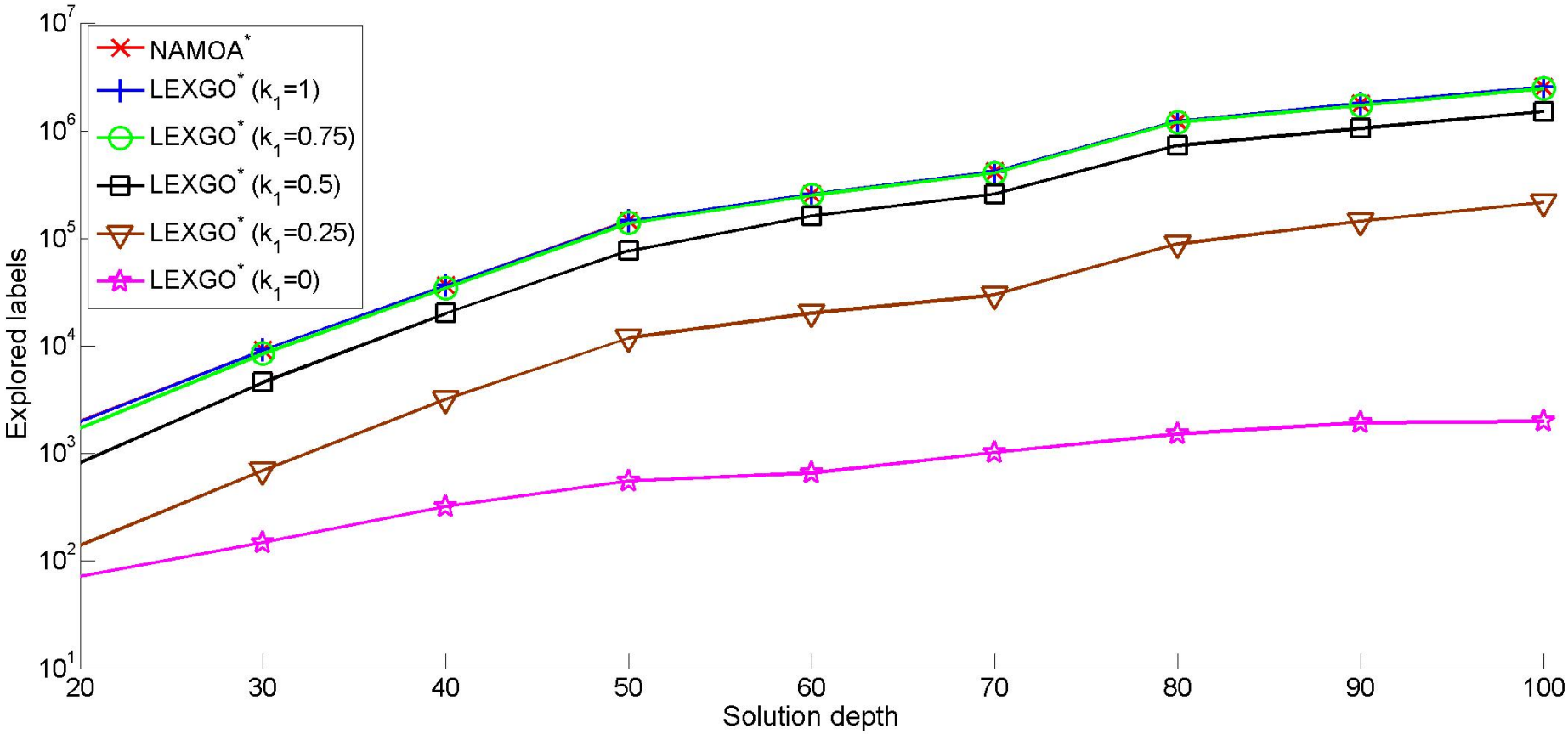


Figure: Explored labels by NAMOA* and LEXGO* in grids.

Deviation-based pruning

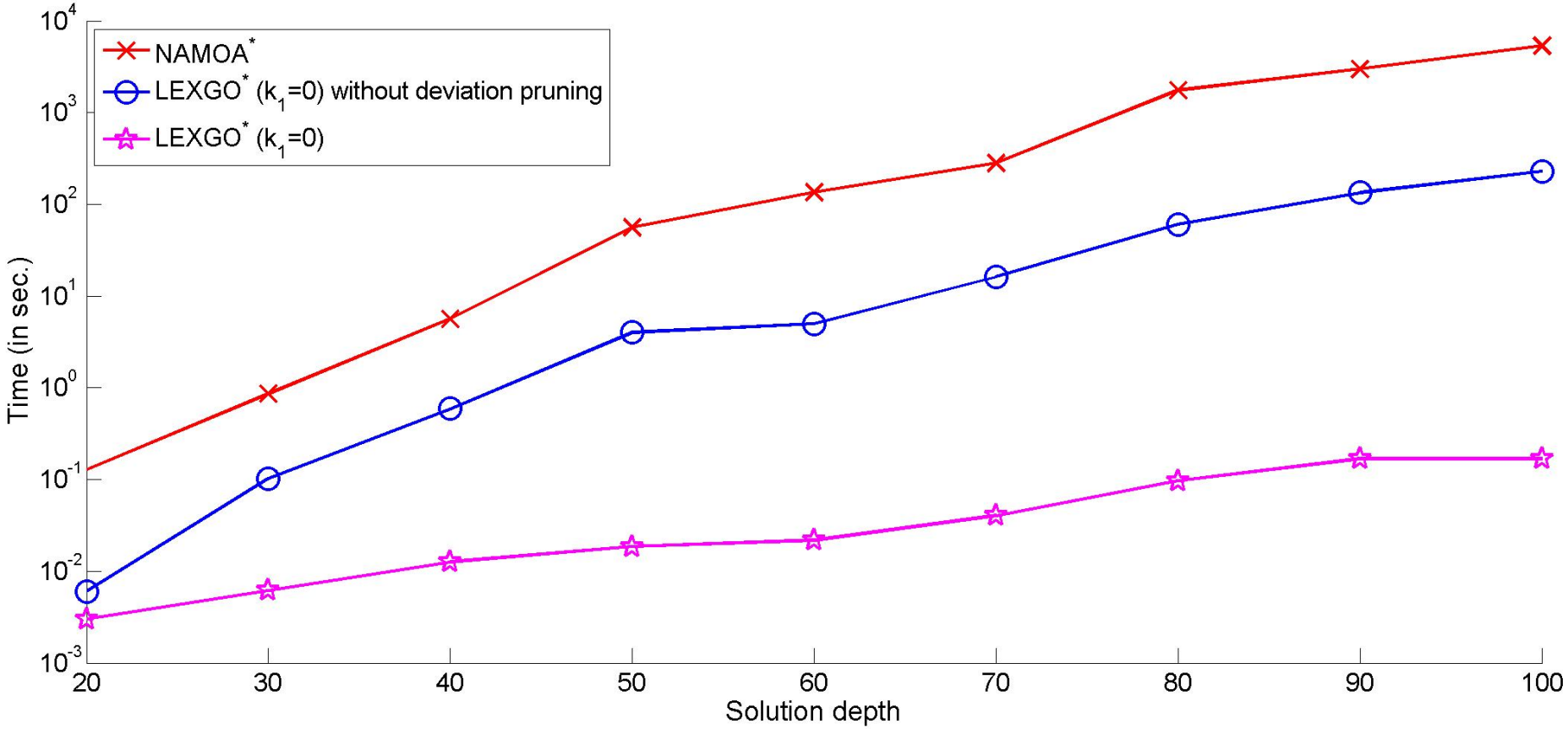


Figure: Impact of deviation pruning in $k_1 = 0$ experiments.

$NAMOA^*_{dr}$ vs $NAMOA^*$

Table: Set sizes for Vermont State map.

Map	$(\frac{\sum T(G_{cl})}{\sum G_{cl}})\%$	$(\frac{\sum T(C^*)}{\sum C^*})\%$
VT_{cut}	3.43	2.45

$NAMOA^*_{dr}$ vs $NAMOA^*$

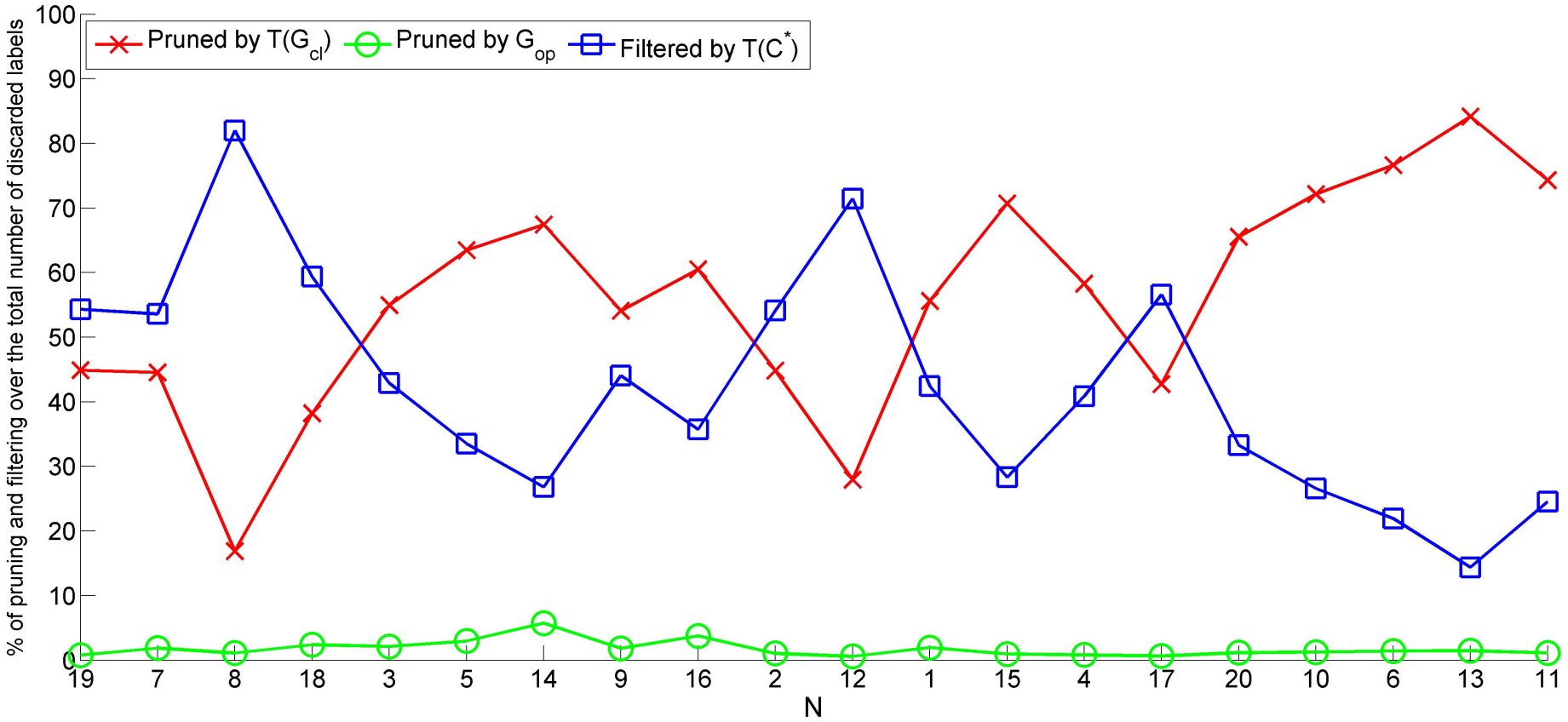


Figure: Discarded labels in Vermont problems.

New York experiments

Now, we can even solve more problems!

Problem #	$NAMOA^*_{dr}$	$NAMOA^*_{lin}$
#11	30 min.	31 hours
#6	3 hours	25 days

Outline

Outline

Conclusions

1. We have tackled the MSP with lexicographic preferences from two approaches: a priori and a posteriori.
2. On the a priori approach, we have contributed two new algorithms: LEXGO* and LEXGO$^*_{dr}$.
3. On the a posteriori approach: NAMOA$^*_{dr}$ based on NAMOA*.
4. All algorithmic approaches have been formally proved to be admissible.
5. The contributions of this thesis outperform the state of the art in an order of magnitude for medium size problems and two orders of magnitude for the hardest problems.
6. NAMOA$^*_{dr}$ LEXGO$^*_{dr}$ represent the new state of the art in MSP with lexicographic goals.

Outline

Future work

1. Investigate more formal proofs about LEXGO*, concerning the optimality in its class of algorithms and the expansion of labels when using more informed lower bounds.
2. Extend LEXGO* to be used with other GP models or other formulas to measure the deviation from goals.
3. Research the applicability of t-discarding to other multiobjective or multicriteria search algorithms with lower bounds.
4. Bring multicriteria search algorithms to real route planning applications.

Thank you for your attention !!!

Gracias por su atención !!!

New Techniques and Algorithms for Multiobjective and Lexicographic Goal-Based Shortest Path Problems

PhD Dissertation

Francisco J. Pulido Arrebola

Supervised by: Dr. Lawrence Mandow

Dept. Lenguajes y Ciencias de la Computación
E.T.S. Ingeniería Informática
Universidad de Málaga

July 7, 2015