Optimal Trajectories in Discrete Space with Acceleration Constraints

Arnaud Casteigts[1] Matteo De Francesco[1] Pierre Leone[1]

[1]Department of Computer Science, University of Geneva

ALGO Seminar

February 27th, 2026

Outline

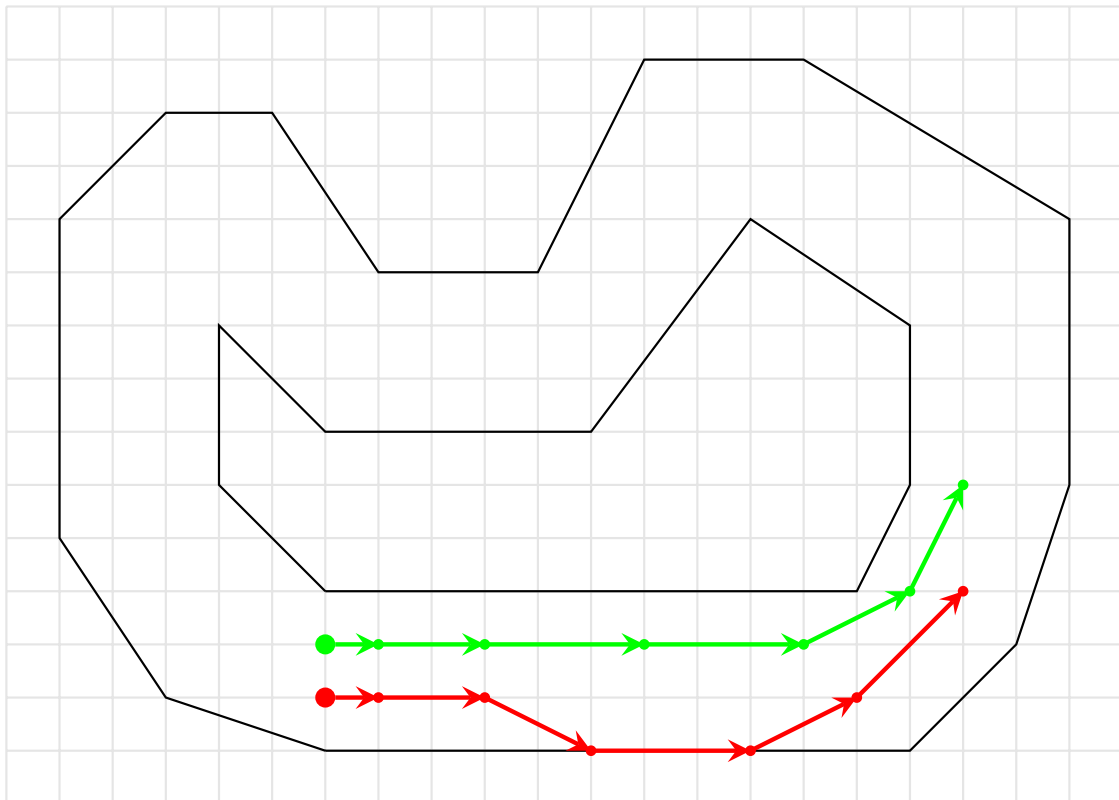
Quick background

VectorRacer a discrete model embedding physics of movements in a clever way[1].

Quick background

VectorRacer a discrete model embedding physics of movements in a clever way[1].

Originally, a paper-and-pencil game among two players in a *racetrack*:



Motivation and contributions

Motion planning in an open space without track limits (for example, drones delivering packages)

Motivation and contributions

Motion planning in an open space without track limits (for example, drones delivering packages)

We revisit the techniques of Bekos et al. [2]:

$\rightarrow$ Improve and extend the constant time procedure to recover a succint representation of all the *feasible branching times* among two configurations in *any* dimension d;

Motivation and contributions

Motion planning in an open space without track limits (for example, drones delivering packages)

We revisit the techniques of Bekos et al. [2]:

$\rightarrow$ Improve and extend the constant time procedure to recover a succint representation of all the *feasible branching times* among two configurations in *any* dimension d;

Furthermore,

- We provide a constant time (and space) strategy to characterize an explicit trajectory among two configurations;
- We prove that the speed in an open space setting is not trivially bounded;

Model

A d-dimensional generalization of Gardner's model in $\mathbb{Z}^d$.

Model

A d-dimensional generalization of Gardner's model in $\mathbb{Z}^d$.

A vehicle at time t is represented by a *configuration* $c_t = (p_t, v_t)$:

- $p_t \in \mathbb{Z}^d$ is the position of the vehicle;
- $v_t \in \mathbb{Z}^d$ is its velocity;

Model

A d-dimensional generalization of Gardner's model in $\mathbb{Z}^d$.

A vehicle at time t is represented by a *configuration* $c_t = (p_t, v_t)$:

- $p_t \in \mathbb{Z}^d$ is the position of the vehicle;
- $v_t \in \mathbb{Z}^d$ is its velocity;

The velocity corresponds to the last move made by the vehicle, $v_t = p_t - p_{t-1}$.

Model

A d-dimensional generalization of Gardner's model in $\mathbb{Z}^d$.

A vehicle at time t is represented by a *configuration* $c_t = (p_t, v_t)$:

- $p_t \in \mathbb{Z}^d$ is the position of the vehicle;
- $v_t \in \mathbb{Z}^d$ is its velocity;

The velocity corresponds to the last move made by the vehicle, $v_t = p_t - p_{t-1}$.

Constraints on the acceleration: $v_{t+1} - v_t \in \{-1, 0, 1\}^d$ for all t; two consecutive velocity vectors can at most differ by one unit in each dimension.

Model

A d-dimensional generalization of Gardner's model in $\mathbb{Z}^d$.

A vehicle at time t is represented by a *configuration* $c_t = (p_t, v_t)$:

- $p_t \in \mathbb{Z}^d$ is the position of the vehicle;
- $v_t \in \mathbb{Z}^d$ is its velocity;

The velocity corresponds to the last move made by the vehicle, $v_t = p_t - p_{t-1}$.

Constraints on the acceleration: $v_{t+1} - v_t \in \{-1, 0, 1\}^d$ for all t; two consecutive velocity vectors can at most differ by one unit in each dimension.

A *trajectory* is a sequence of configurations $c_0, \ldots, c_\ell$ where every two consecutive configurations satisfy the above conditions.

Branching Cost

Problem: Branching Cost

In: Two configurations $c = (p, v)$ and $c' = (p', v')$ in $\mathbb{Z}^d \times \mathbb{Z}^d$

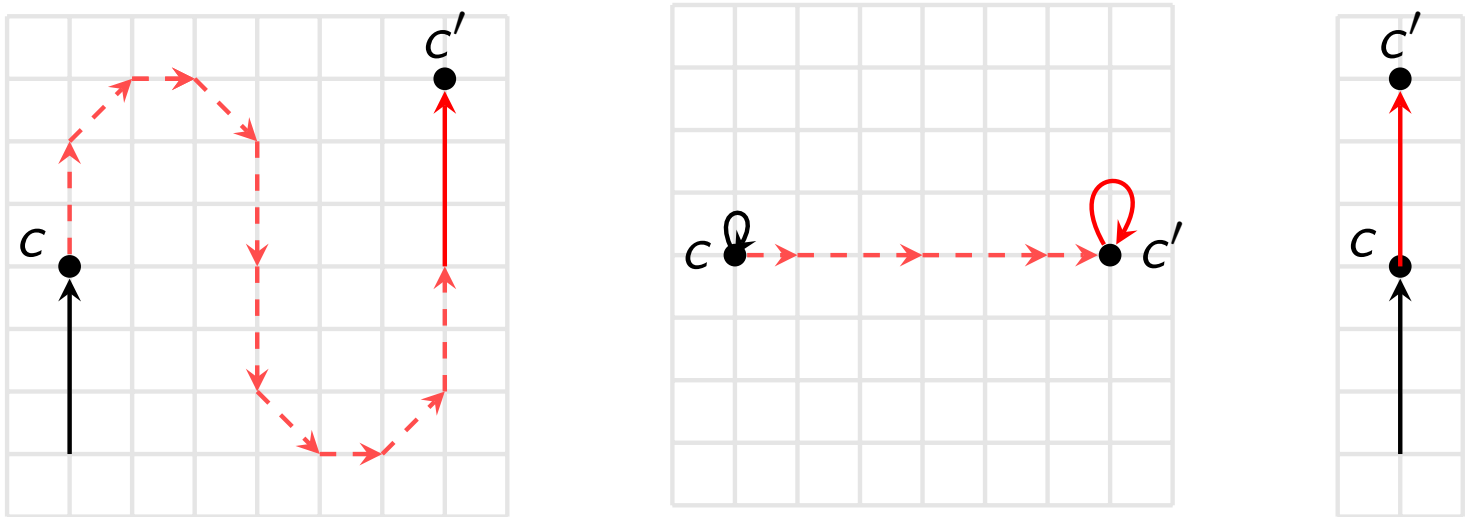
Out: What is the shortest length of a trajectory from c to c'?

Branching Cost

Braching two d-dimensional configuration $\equiv$ branching several instances in the single dimension $+$ an appropriate combination strategy.

Branching Cost

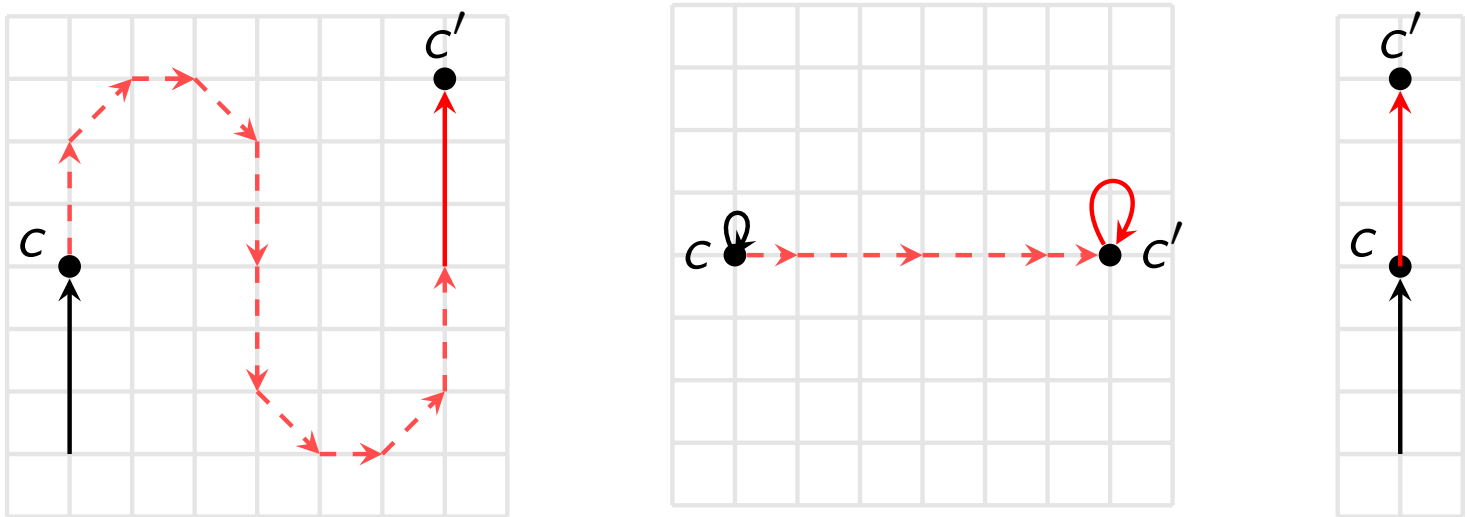
Braching two d-dimensional configuration $\equiv$ branching several instances in the single dimension $+$ an appropriate combination strategy.



- Branching c to c' requires at least 11 timesteps;

Branching Cost

Braching two d-dimensional configuration $\equiv$ branching several instances in the single dimension + an appropriate combination strategy.



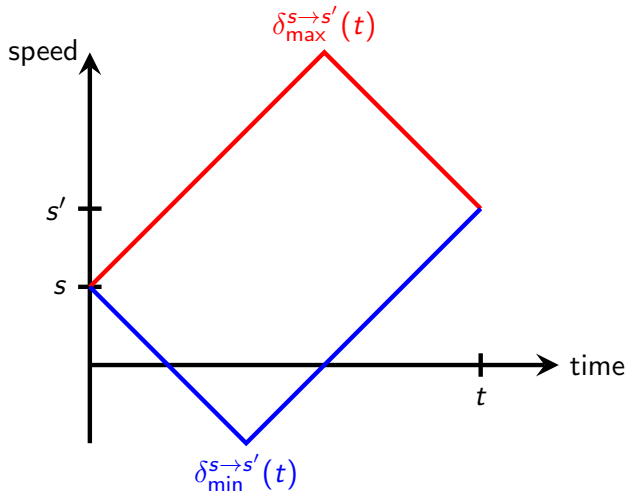
- Branching c to c' requires at least 11 timesteps;
- However, the minimum time in x is 5 timesteps, and 1 timestep for y

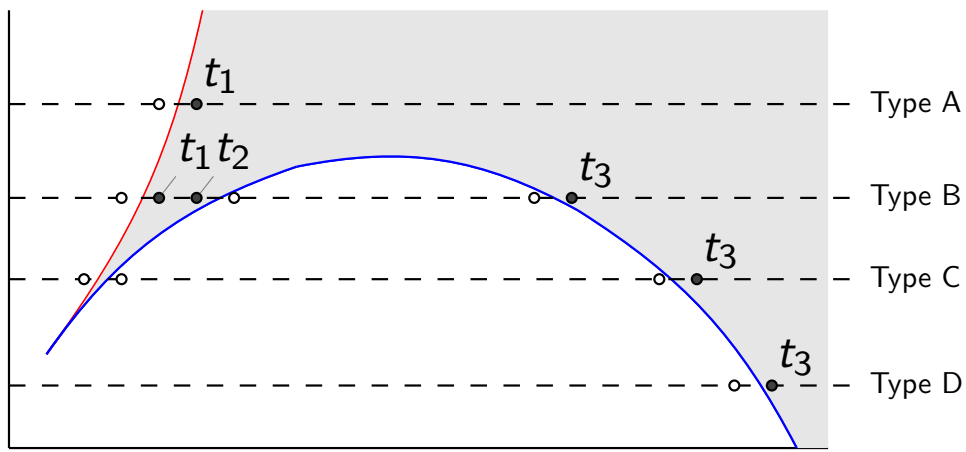
Branching Cost d-dimensional

Given $c = (x, s)$ and $c' = (x', s')$ (one-dimensional), $\mathcal{I}(c, c')$ returns the set of feasible time for branching c to c'.

Branching Cost d-dimensional

Given $c = (x, s)$ and $c' = (x', s')$ (one-dimensional), $\mathcal{I}(c, c')$ returns the set of feasible time for branching c to c'.
$\mathcal{I}(c, c')$ can always be described either as a single interval $\mathcal{I} = [t_1, \infty)$ or as two intervals $\mathcal{I} = [t_1, t_2] \cup [t_3, \infty)$.





Branching Cost d-dimensional

Given $c = ((x_1, x_2, \cdots, x_d), (s_1, s_2, \cdots, s_d))$ and
$c' = ((x'_1, x'_2, \cdots, x'_d), (s'_1, s'_2, \cdots, s'_d))$

Branching Cost d-dimensional

Given $c = ((x_1, x_2, \cdots, x_d), (s_1, s_2, \cdots, s_d))$ and $c' = ((x'_1, x'_2, \cdots, x'_d), (s'_1, s'_2, \cdots, s'_d))$

$\rightarrow$ we retrieve the whole feasible interval as

$$\min \mathcal{I} = \bigcap_{i \in d} \mathcal{I}((x_i, s_i), (x'_i, s'_i))$$

Branching Cost d-dimensional

Given $c = ((x_1, x_2, \cdots, x_d), (s_1, s_2, \cdots, s_d))$ and $c' = ((x'_1, x'_2, \cdots, x'_d), (s'_1, s'_2, \cdots, s'_d))$

$\rightarrow$ we retrieve the whole feasible interval as

$$\min \mathcal{I} = \bigcap_{i \in d} \mathcal{I}((x_i, s_i), (x'_i, s'_i))$$

In $O(d \log d)$ time

Branching Trajectory

Problem: Branching Trajectory

In: Two configurations $c = (p, v)$ and $c' = (p', v')$ in $\mathbb{Z}^d \times \mathbb{Z}^d$

Out: A shortest trajectory $c_0, c_1, \ldots, c_\ell$ from c to c'.

Branching Trajectory

Problem: Branching Trajectory

In: Two configurations $c = (p, v)$ and $c' = (p', v')$ in $\mathbb{Z}^d \times \mathbb{Z}^d$

Out: A shortest trajectory $c_0, c_1, \ldots, c_\ell$ from c to c'.

Using the previous procedure, $\mathcal{I} = (c, c')$ yields a feasible interval.

Branching Trajectory

Problem: Branching Trajectory

In: Two configurations $c = (p, v)$ and $c' = (p', v')$ in $\mathbb{Z}^d \times \mathbb{Z}^d$
Out: A shortest trajectory $c_0, c_1, \ldots, c_\ell$ from c to c'.

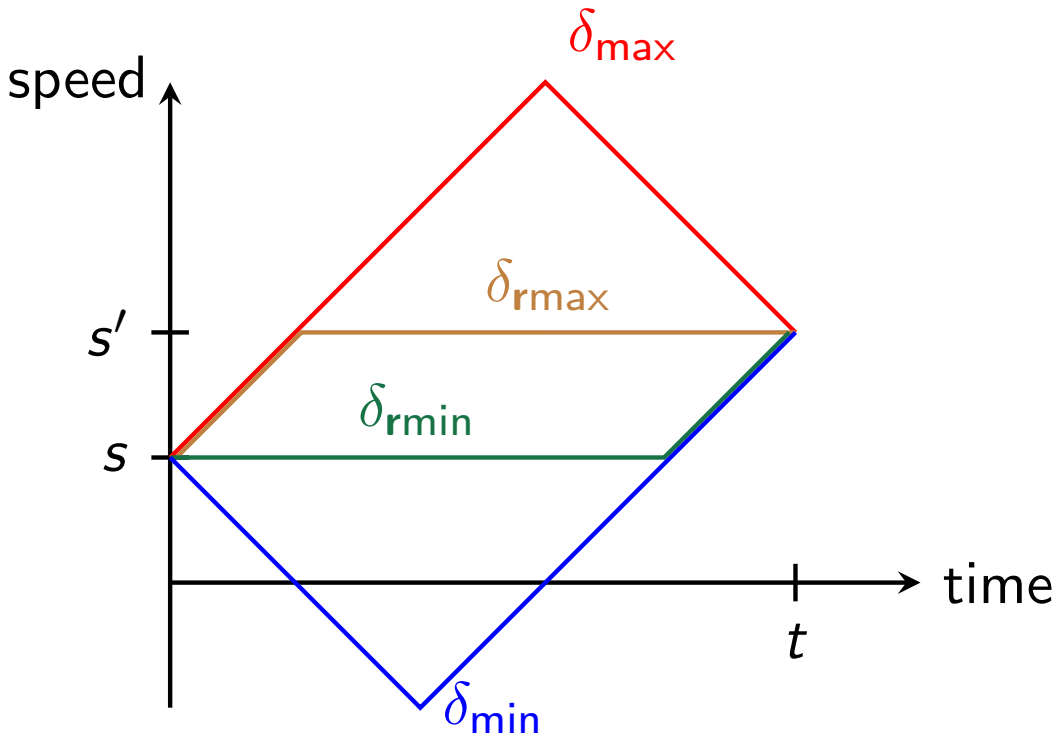
Using the previous procedure, $\mathcal{I} = (c, c')$ yields a feasible interval.

Pick a specific $t \in \mathcal{I}$. How to compute a shortest trajectory in t moves?

Again, we separate the computation in each dimension:

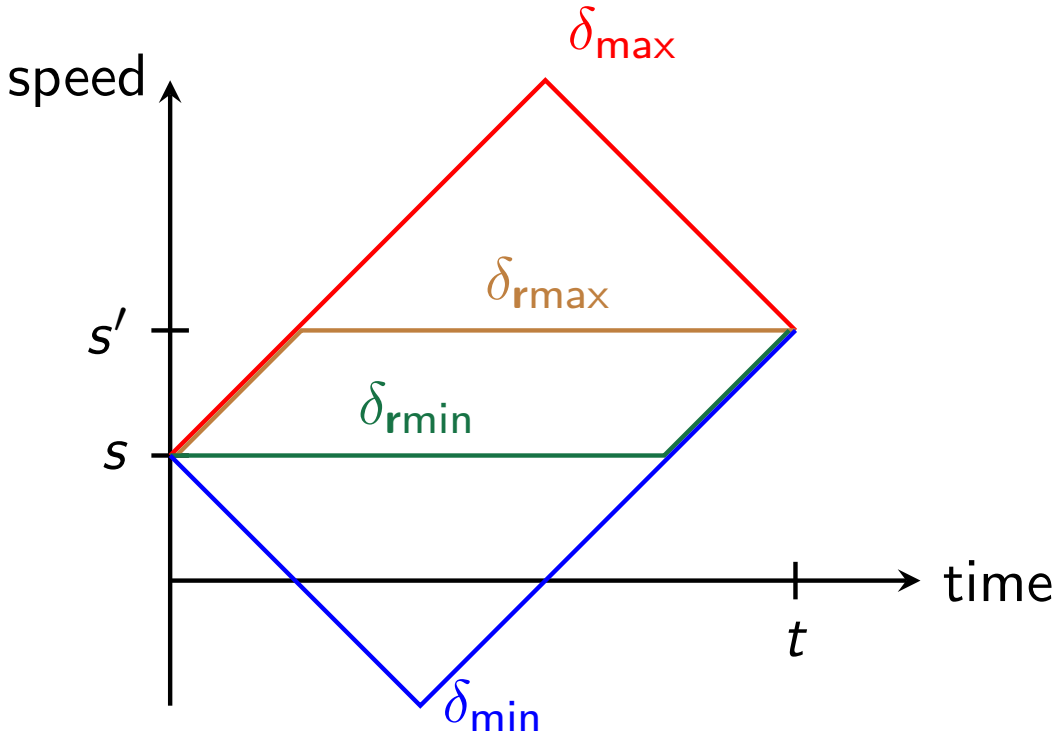
Branching Trajectory

Given t, s and s', we can calculate in constant time 4 quantities



Branching Trajectory

Given t, s and s', we can calculate in constant time 4 quantities



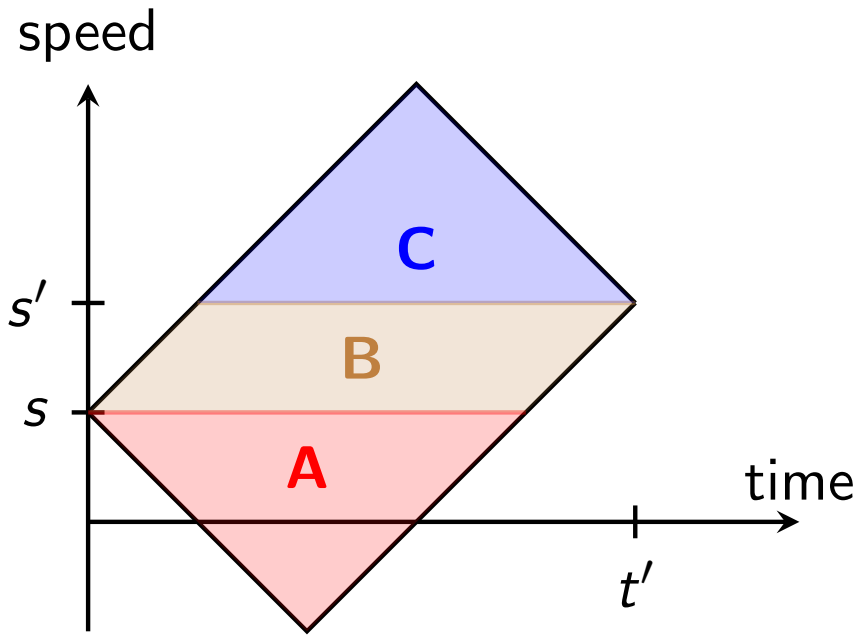
red trajectory: maximum distance covered from s and s' in t steps.

blue trajectory: minimum distance covered from s and s' in t steps.

brown and green: two additional reference distances.

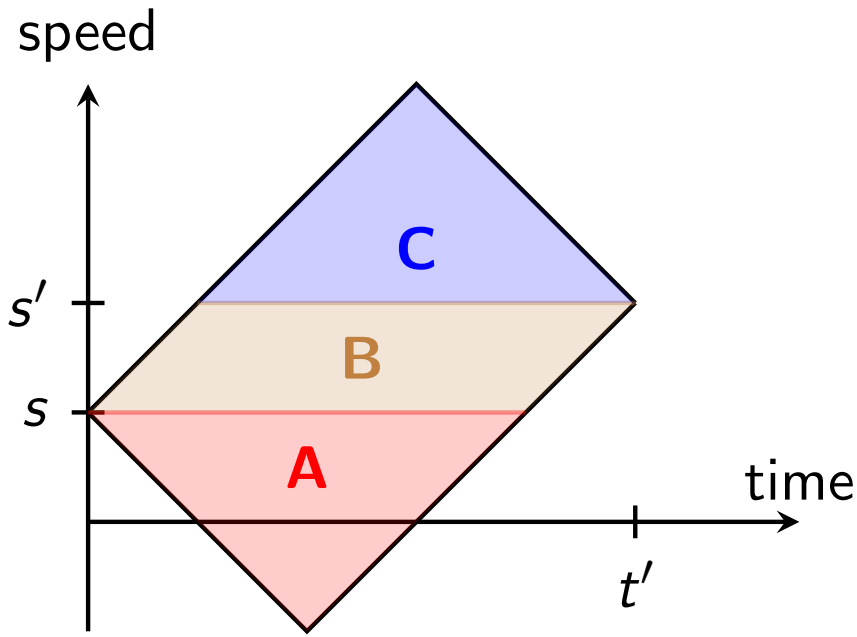
Branching Trajectory

Since t is feasible, given the distance to cover $\delta = x' - x$, we can partition the *distances* space:



Branching Trajectory

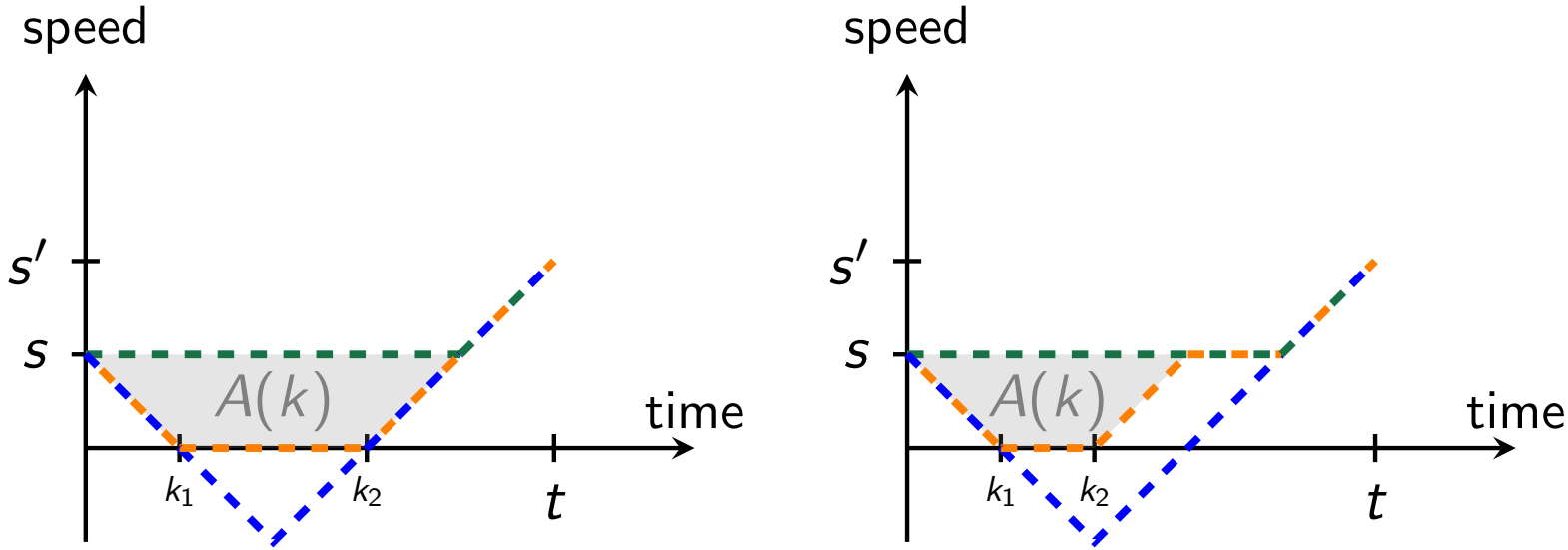
Since t is feasible, given the distance to cover $\delta = x' - x$, we can partition the *distances* space:



- $\delta_{\mathbf{r}\text{max}} < \delta \leq \delta_{\text{max}}$ (region **C**);
- $\delta_{\mathbf{r}\text{min}} < \delta \leq \delta_{\mathbf{r}\text{max}}$ (region **B**);
- $\delta_{\text{min}} \leq \delta \leq \delta_{\mathbf{r}\text{min}}$ (region **A**);

Branching Trajectory

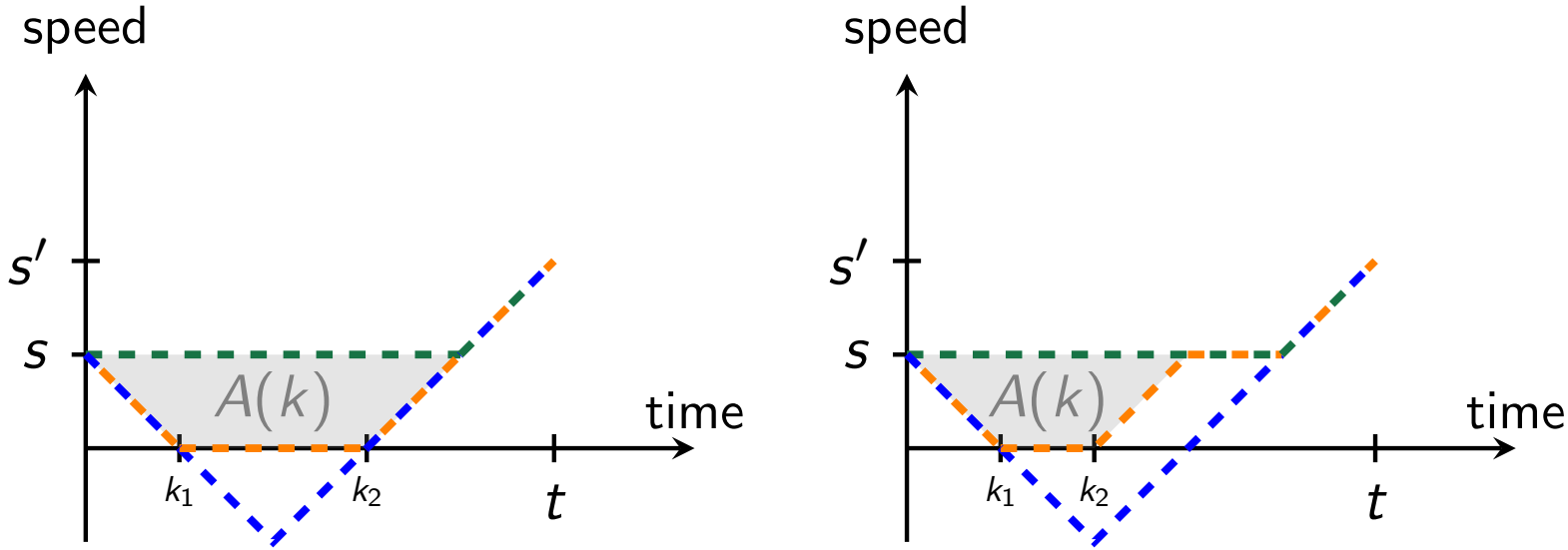
Suppose we are in region **A**: then a feasible δ trajectory has one of the following form



We can express $A(k)$ in constant time and solve it to find k_1 and k_2.

Branching Trajectory

Suppose we are in region **A**: then a feasible δ trajectory has one of the following form



We can express $A(k)$ in constant time and solve it to find k_1 and k_2.

Using checkpoints k_1 and k_2, we can describe a feasible trajectory in constant time and constant space.

$$(-, k_1)(0, k_2 - k_1)(+, \ell - k_2)$$

$$(-, k_1)(0, k_2 - k_1)(+, 1)(0, \ell - k_2 - k_1 - s' + s)(+, s' - s + k_1 - 1)$$

Multipoint Trajectory

Problem: Multipoint Trajectory

In: a sequence $x_0, x_1, \ldots, x_n$ of points in Euclidean space $\mathbb{Z}^d$
Out: a trajectory $\mathcal{T} = c_0, c_1, \ldots, c_\ell$ with $c_i = (p_i, v_i)$ such that $v_0 = v_\ell = \vec{0}$ and $\mathcal{T}$ visits the points $x_0, x_1, \ldots, x_n$ in this order. Furthermore, ℓ is minimum over all such trajectories.

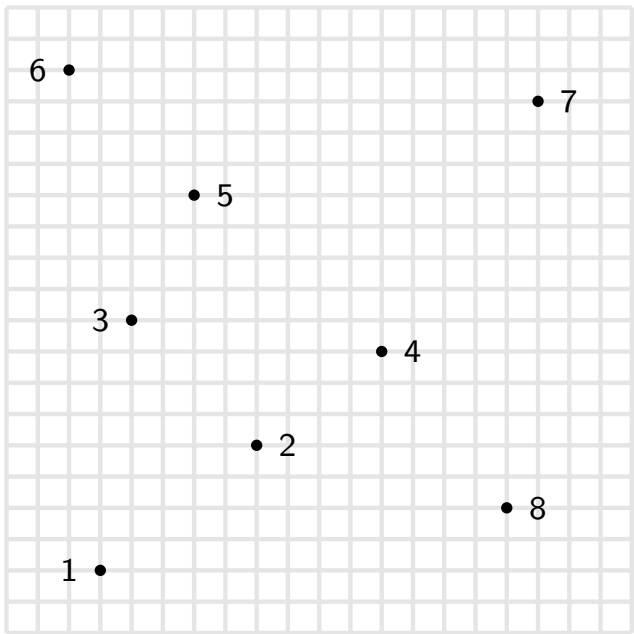
Multipoint Trajectory

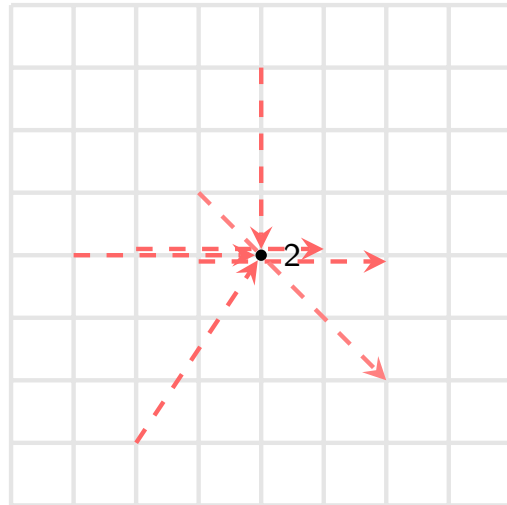
Problem: Multipoint Trajectory

In: a sequence $x_0, x_1, \ldots, x_n$ of points in Euclidean space $\mathbb{Z}^d$
Out: a trajectory $\mathcal{T} = c_0, c_1, \ldots, c_\ell$ with $c_i = (p_i, v_i)$ such that $v_0 = v_\ell = \vec{0}$ and $\mathcal{T}$ visits the points $x_0, x_1, \ldots, x_n$ in this order. Furthermore, ℓ is minimum over all such trajectories.

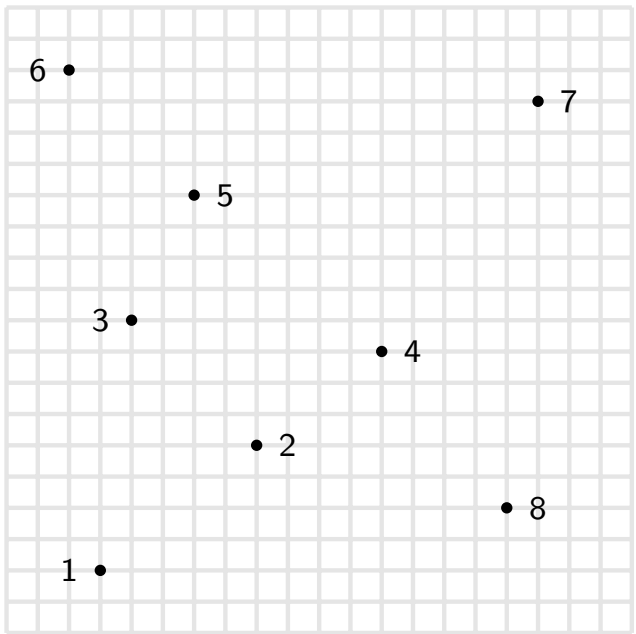
A configuration $c = (p, v)$ *visits* a point x if x lies on a segment between the points $p - v$ and p (the point must be traversed by the vector, not necessarily be located at its endpoint).

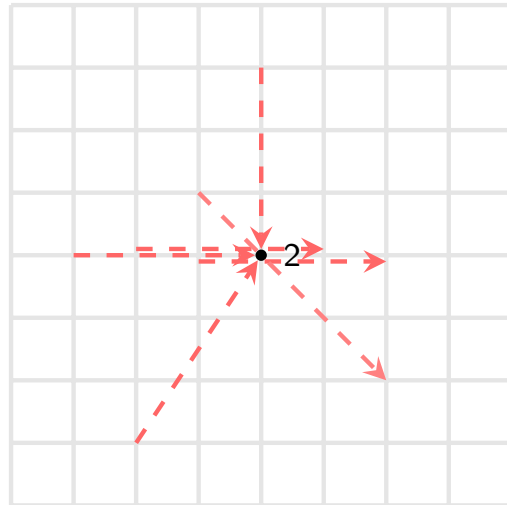
Multipoint Trajectory





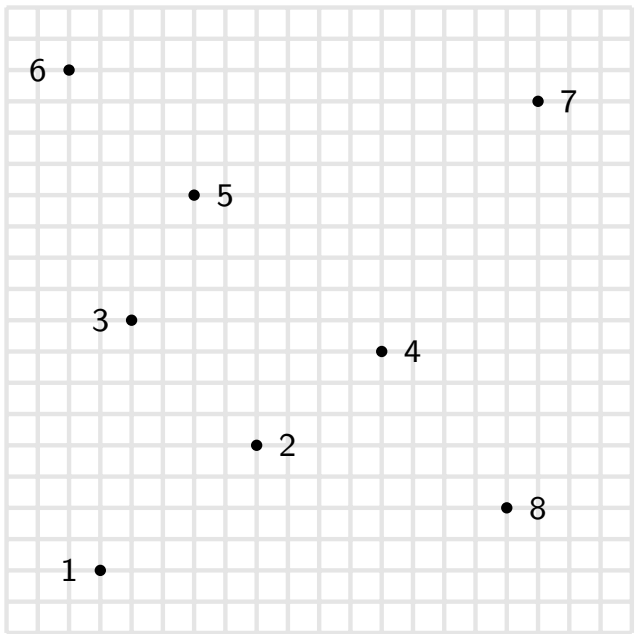
Multipoint Trajectory

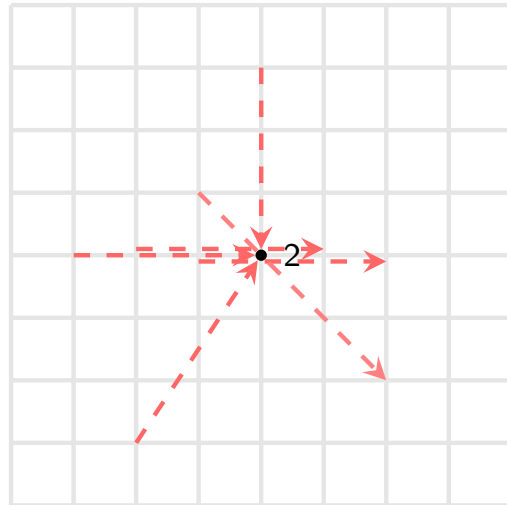




Dynamic programming algorithm, considering for every point all the configurations that can *visit* the point.

Multipoint Trajectory

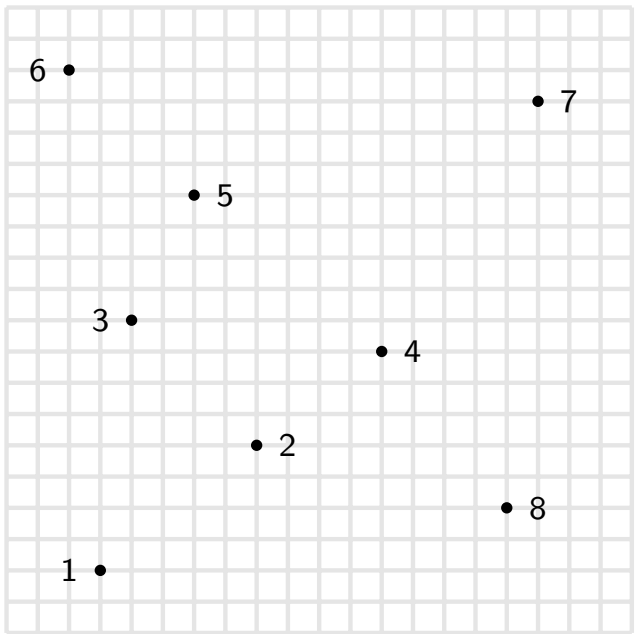


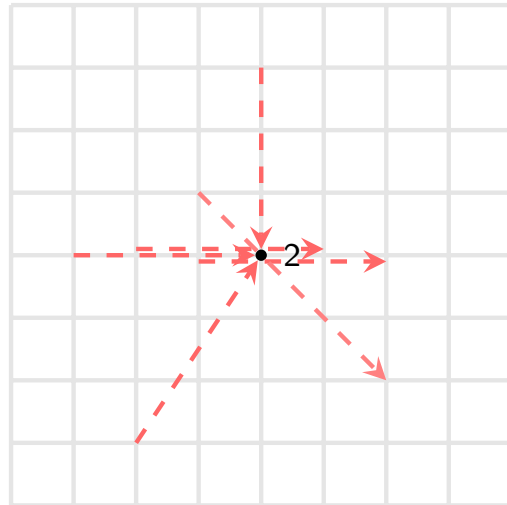


Dynamic programming algorithm, considering for every point all the configurations that can *visit* the point.

Complexity linear in the points, but quadratically in the number of configurations per point $O(n|\mathcal{C}|^2)$.

Multipoint Trajectory





Dynamic programming algorithm, considering for every point all the configurations that can *visit* the point.

Complexity linear in the points, but quadratically in the number of configurations per point $O(n|\mathcal{C}|^2)$.

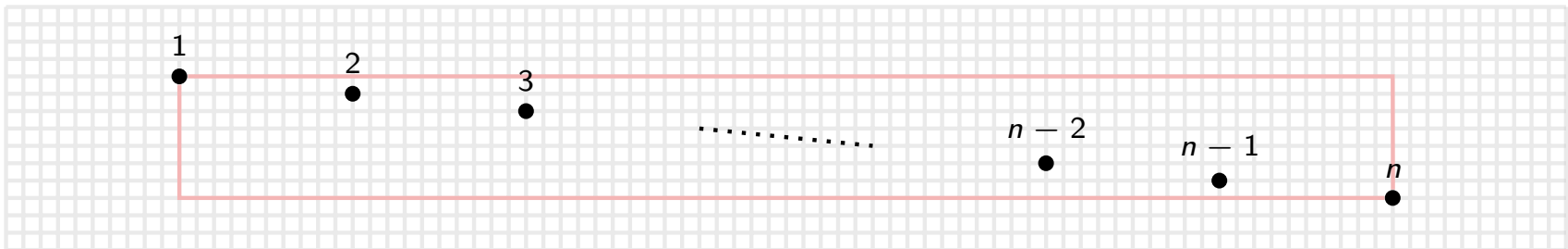
The number of configurations grows with the speed: how to choose *locally maximum configurations speed* for every point?

Bounding speed is not local

Given the following instance:

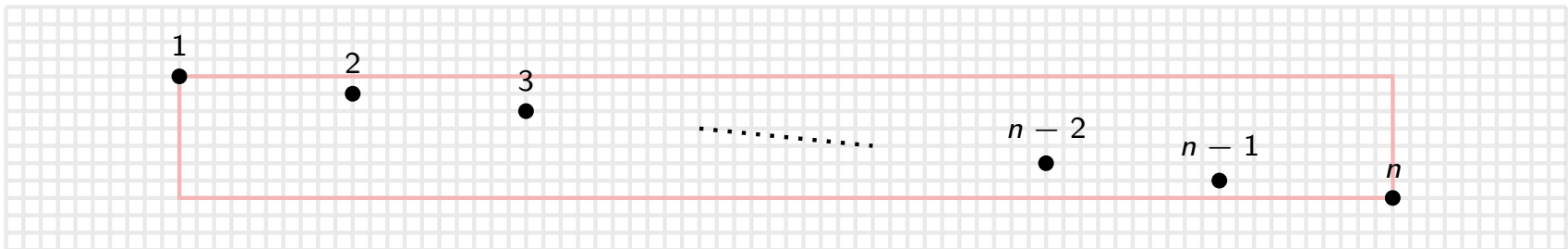
Bounding speed is not local

Given the following instance:



Bounding speed is not local

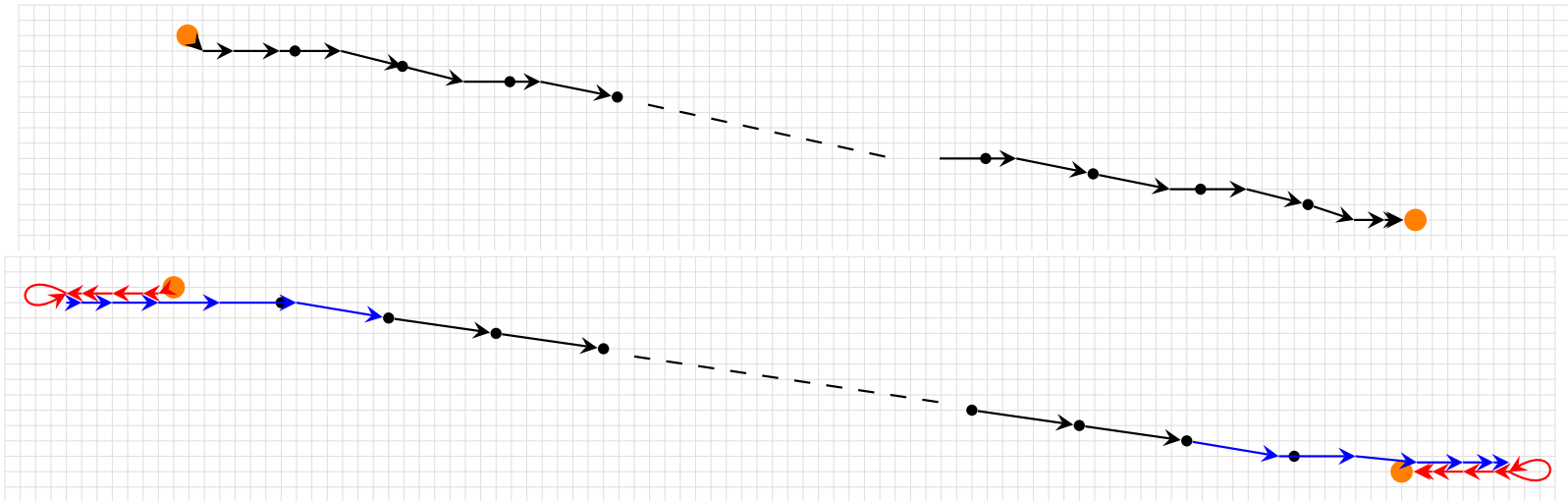
Given the following instance:



In red, *minimum bounding box* enclosing all the points.

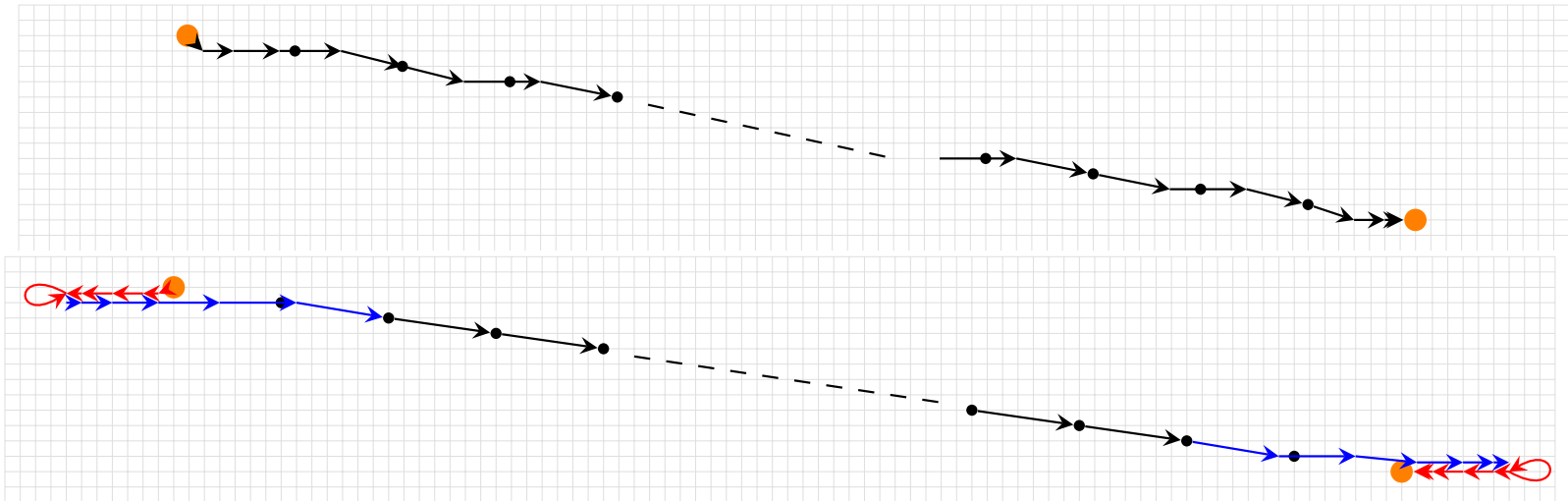
Every point at distance δ (x-axis) and 1 (y-axis) from one another.

Bounding speed is not local



We distinguish two kind of trajectories:

Bounding speed is not local



We distinguish two kind of trajectories:

- **NTU (Non-TUrning)** if it does not contain configurations with negative speed in the x dimension.
- **TU (TUrning)** if it allows for configurations with negative speed in the x dimension.

Bounding speed is not local

We denote with c^*_{NTU} the optimal cost of the best **NTU** trajectory (similarly for the others);

Bounding speed is not local

We denote with c^*_{NTU} the optimal cost of the best **NTU** trajectory (similarly for the others);

Denote also with $TU(\delta)$ the family of **TU** trajectory where configurations can reach at most δ speed in absolute value;

Bounding speed is not local

We denote with c^*_{NTU} the optimal cost of the best **NTU** trajectory (similarly for the others);

Denote also with $TU(\delta)$ the family of **TU** trajectory where configurations can reach at most δ speed in absolute value;

We analytically derived the following result:

Bounding speed is not local

Lemma 1 For all $\delta \geq 7$, any NTU trajectory cannot reach a x-speed of δ.

Bounding speed is not local

Lemma 1 For all $\delta \geq 7$, any NTU trajectory cannot reach a x-speed of δ.

Theorem 1 For any fixed $\delta \geq 7$ and constant $k \in \mathbb{Z}$, there exists a $n_0 = O(k^2\delta^2)$ such that $\forall n \geq n_0$, there exists an TU($k\delta$) with a better cost than an NTU.

Conjecture on the speed

The *advantage* bound vanishes for $k = O(\sqrt{n})$.

Conjecture on the speed

The *advantage* bound vanishes for $k = O(\sqrt{n})$.

Experiments on random instances show the same phenomena.

Conjecture on the speed

The *advantage* bound vanishes for $k = O(\sqrt{n})$.

Experiments on random instances show the same phenomena.

Conjecture 1 Let L be the maximum distance between two cities in any of the dimensions. Then, there exists an optimal trajectory that never exceeds speed $\sqrt{L}$ in any of the dimensions.

Experiments with/without conjecture

We implemented a Dynamic Programming (DP) algorithm in different settings:

Experiments with/without conjecture

We implemented a Dynamic Programming (DP) algorithm in different settings:

- varying the size of L;
- varying the size of n;
- using a (large and safe) conservative bound on the speed;
- using the conjectured bound on the speed;

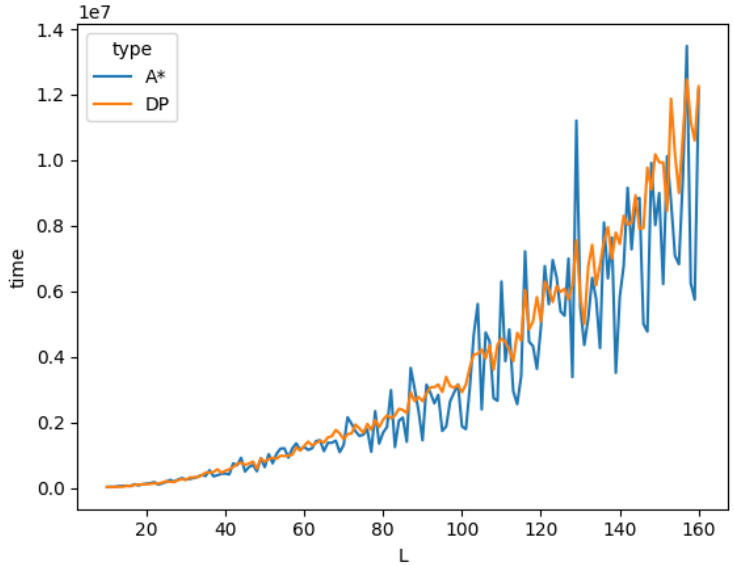
Experiments with/without conjecture

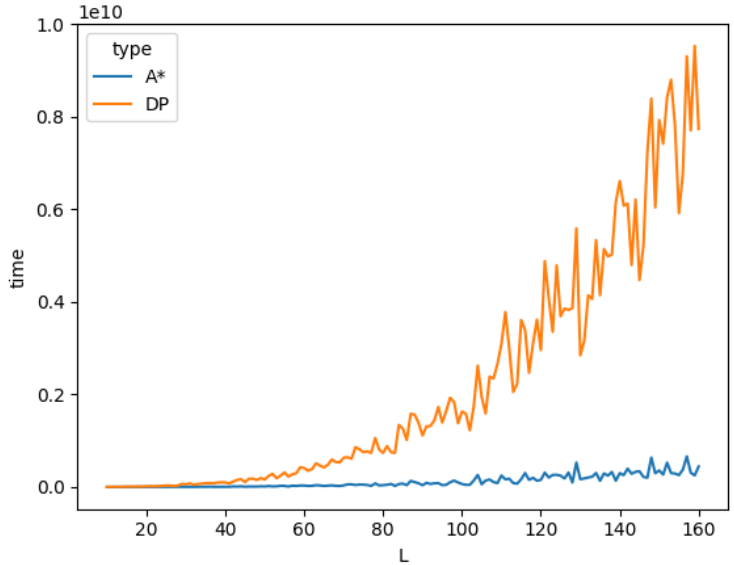
We implemented a Dynamic Programming (DP) algorithm in different settings:

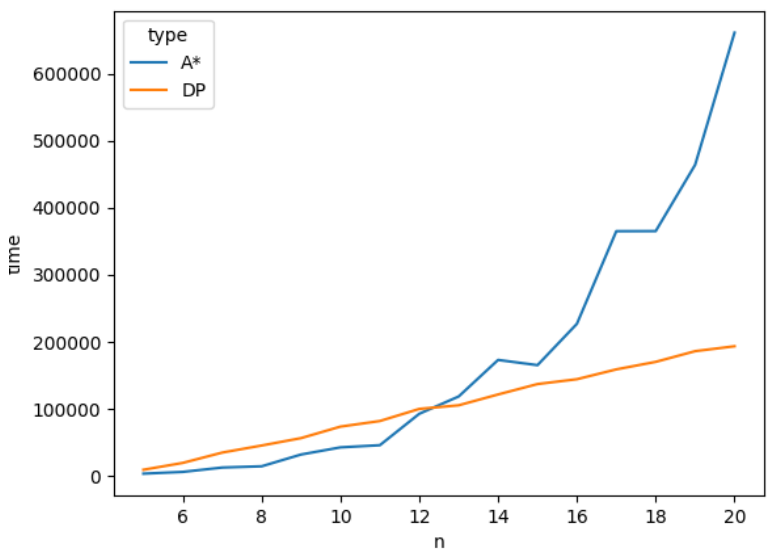
- varying the size of L;
- varying the size of n;
- using a (large and safe) conservative bound on the speed;
- using the conjectured bound on the speed;

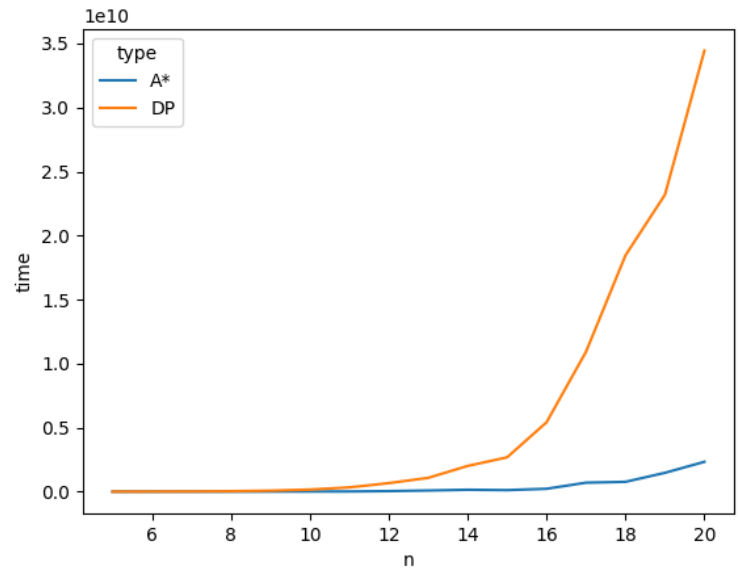
Comparing our DP with A^* we obtained

Experiments with/without conjecture









Open questions

The particular instance we built satisfy our conjecture (up to constant factor). Does the same holds for an arbitrary arrangements of points?

Open questions

The particular instance we built satisfy our conjecture (up to constant factor). Does the same holds for an arbitrary arrangements of points?

Are there families of instances where more (possibly locally) conservative bounds can be proven?

Thank you for the attention!

Questions?

Bibliography I

[1] M. Gardner, "Sim, chomp and race track-new games for intellect," *Scientific American*, vol. 228, no. 1, pp. 108–115, 1973.

[2] M. A. Bekos, T. Bruckdorfer, H. Förster, M. Kaufmann, S. Poschenrieder, and T. Stüber, "Algorithms and insights for racetrack," *Theoretical Computer Science*, 2018.