THE UNIVERSITY *of* EDINBURGH
informatics

Advanced Robotics

Optimal control (quick intro)

Steve Tonneau
School of Informatics
University of Edinburgh

Problem presentation

$$\min_{X,U} \int_0^T l\big(x(t),u(t)\big)dt + l_T\big(x(T)\big)$$

$$\text{s.t.} \quad \dot{x}(t) = f(x(t),u(t))$$

Path cost

Terminal cost

- X and U are functions of t:

  X: $t \in \Re \rightarrow x(t) \in \Re^{\mathrm{nx}}$

  U: $t \in \Re \rightarrow u(t) \in \Re^{\mathrm{nu}}$

- The terminal time T is fixed

Recap from last time:

Problem presentation

$$\min_{X,U} \int_0^T l\big(x(t), u(t)\big)dt + l_T\big(x(T)\big)$$

$$\text{s.t.} \quad \dot{x}(t) = f(x(t),u(t))$$

Path cost

Terminal cost

- X and U are functions of t:

  $X: t \in \Re \rightarrow x(t) \in \Re^{\mathrm{nx}}$

  $U: t \in \Re \rightarrow u(t) \in \Re^{\mathrm{nu}}$

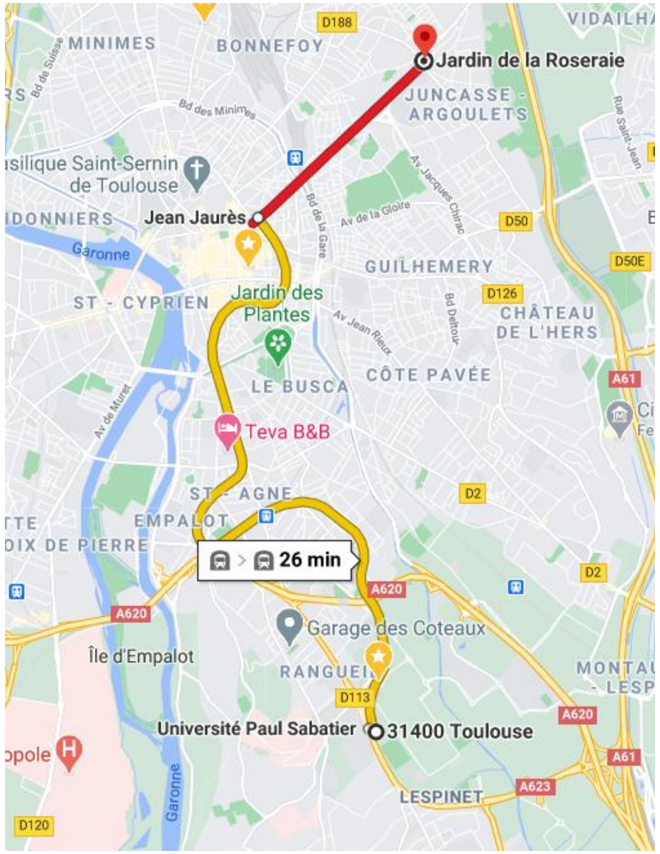
- The terminal time T is fixed

Principle of optimality

- How to find the optimal control?

- Principle of optimality:

  Subpath of optimal paths are also optimal for their own subproblem



Constructing an optimal policy

For every possible state we are in:

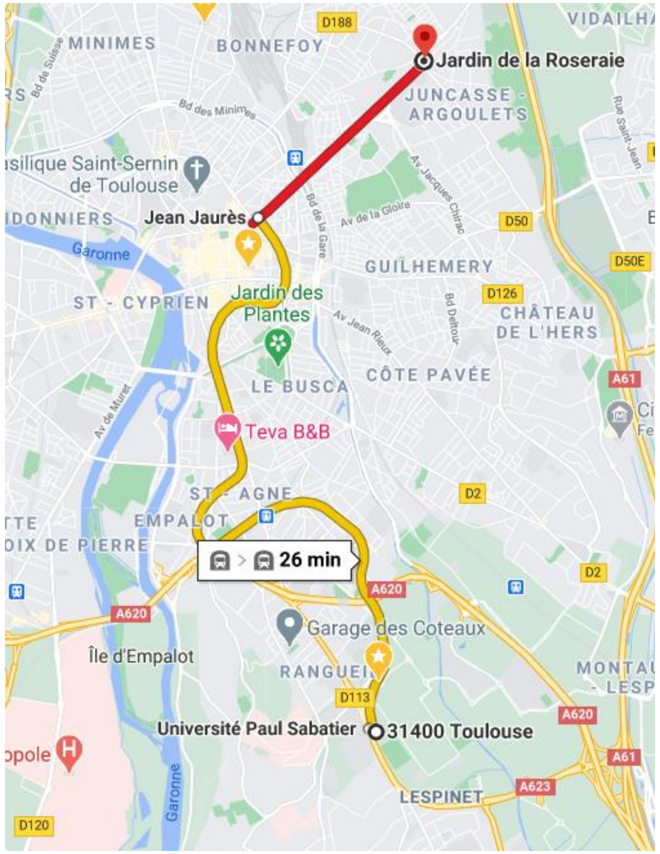
- There exists an optimal action towards the goal
  "Going to Jean Jaurès is optimal…"

- To know the action is optimal we need to know what
  next action will be optimal
  "… Because Jean Jaurès => Roseraie is optimal"

How is that helping?

If we know the cost of ALL actions from ALL states
at each state we can determine exactly what best action to take

A function that gives us an action to take for a given state is called a policy



Constructing an optimal policy

For every possible state we are in:

- ❑ There exists an optimal action towards the goal
  "Going to Jean Jaurès is optimal…"

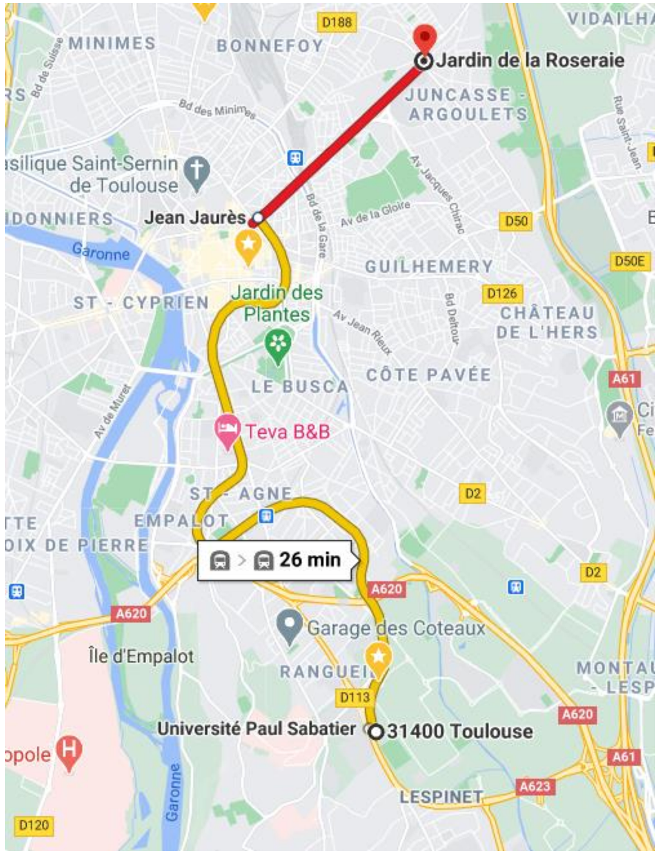
- ❑ To know the action is optimal we need to know what next action will be optimal
  "… Because Jean Jaurès => Roseraie is optimal"

How is that helping?

If we know the cost of ALL actions from ALL states
at each state we can determine exactly what best action to take

A function that gives us an action to take for a given state is called a policy

⇨ (How) to compute optimal policies for robotics problems?



Transcription of optimality principle

- Value function: cost-to-go from current state

$$V_t(x_t) = \min_{u_t, \cdots, u_{N-1}} \sum_{k=t}^{T-1} l_k(x_k, u_k) + l_T(x_T)$$

Transcription of optimality principle

- Value function: cost-to-go from current state

$$V_t(x_t) = \min_{u_t, \cdots, u_{N-1}} \sum_{k=t}^{T-1} l_k(x_k, u_k) + l_T(x_T)$$

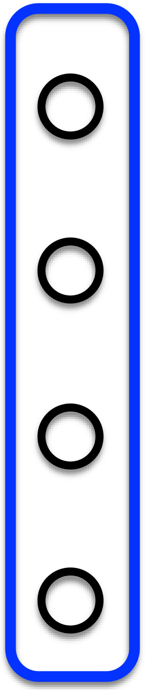
- According to Bellman's principle of optimality:

$$V_t(x_t) = \min_{u_t} l_t(x_t, u_t) + V_{t+1}(x_{t+1})$$

$$x_{t+1} = f_t(x_t, u_t)$$

Problem solved backwards, for all possible states

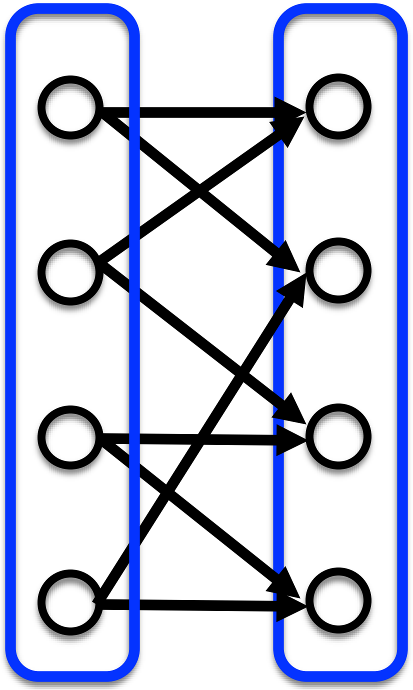
$$V_t(x_t) = \min_{u_t} l_t(x_t, u_t) + V_{t+1}(x_{t+1})$$



Final States
Stage T
$V_T(x_T)$

Problem solved backwards, for all possible states

$$V_t(x_t) = \min_{u_t} l_t(x_t, u_t) + V_{t+1}(x_{t+1})$$



Stage T-1

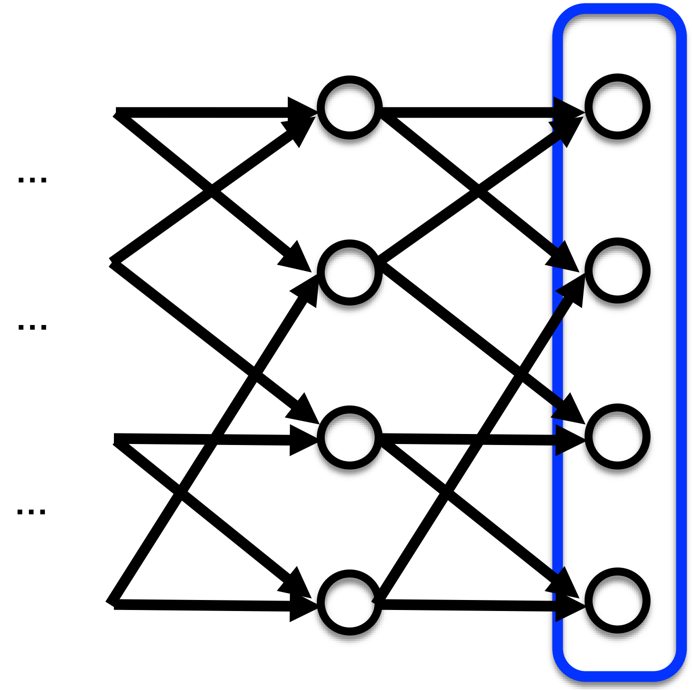
Final States
Stage T

$V_{T-1}(x_{T-1})$ $V_T(x_T)$

$\pi_{T-1}(x_{T-1})$

Problem solved backwards, for all possible states

$$V_t(x_t) = \min_{u_t} l_t(x_t, u_t) + V_{t+1}(x_{t+1})$$



Stage T-1

$V_{T-1}(x_{T-1})$

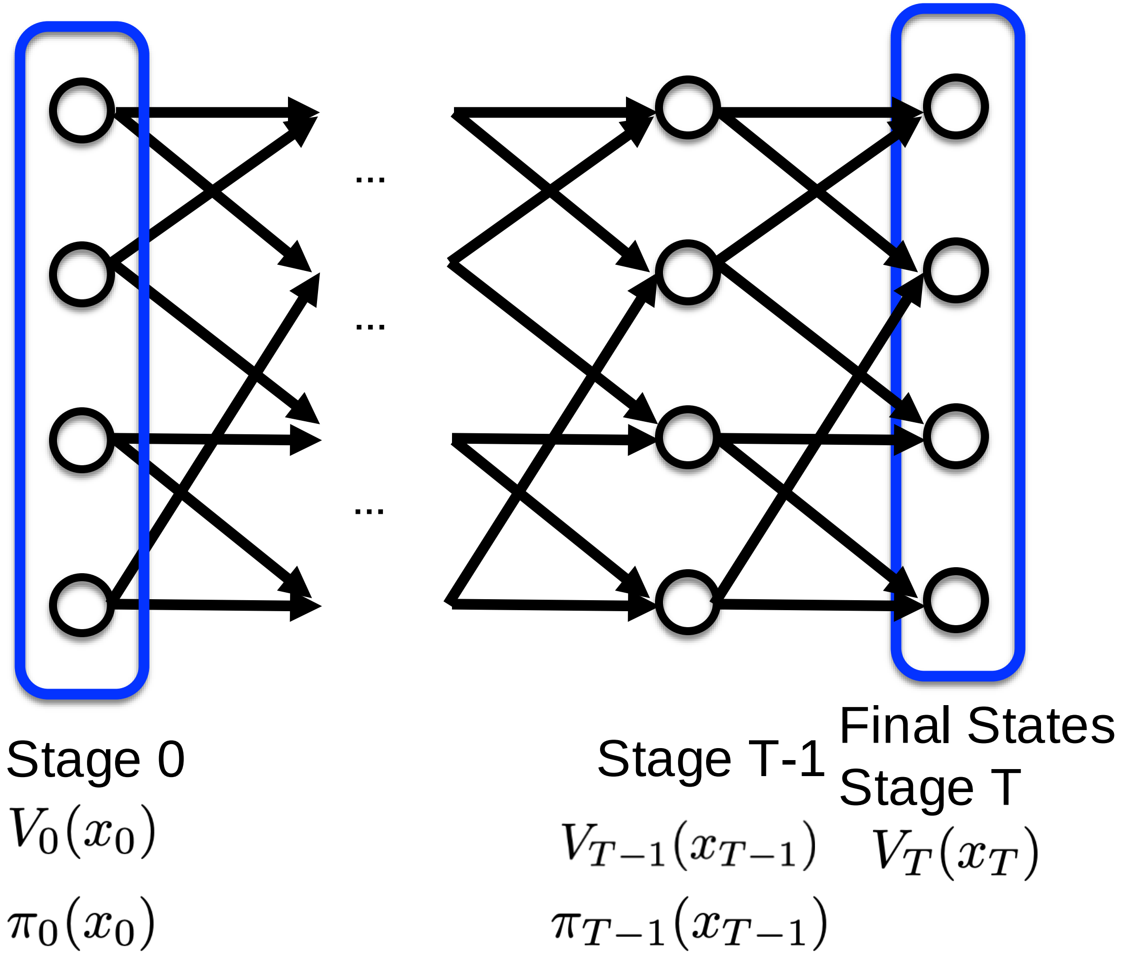
$\pi_{T-1}(x_{T-1})$

Final States
Stage T

$V_T(x_T)$

Problem solved backwards, for all possible states

$$V_t(x_t) = \min_{u_t} l_t(x_t, u_t) + V_{t+1}(x_{t+1})$$



Obvious limitations to the approach

- Curse of dimensionality:

  How to solve for all possible states in high dimensions ?

- In general, trade curse of dimensionality for local optimality

  We'll see one example with trajectory optimisation