# Almost-Optimal Strategies in Priced Timed Games

Patricia Bouyer[1], Kim G. Larsen[2],
Nicolas Markey[1], and Jacob Illum Rasmussen[2]

[1] Lab. Specification et Verification, ENS Cachan & CNRS, France
[2] Dept of Computer Science, Aalborg University, Denmark

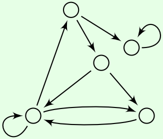
December 15, 2006

# Verification & Model-Checking

system:

property:


⇒

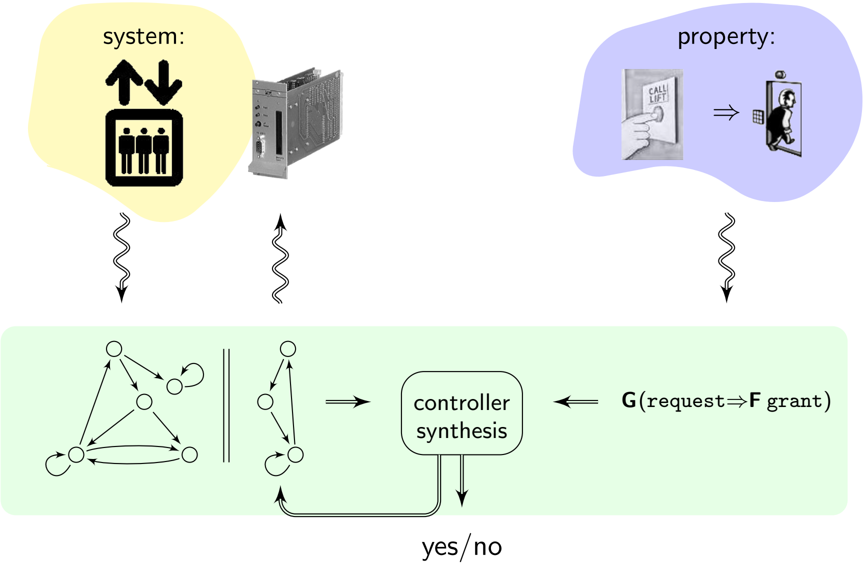


model-checking algorithm

$\mathbf{G}(\texttt{request} \Rightarrow \mathbf{F}\,\texttt{grant})$

yes/no

# Verification & Control



# Adding timing requirements

- Need for timed models:
  - the behaviour of most systems depends on time;
  - (faithful) modelling has to take time into account;

$\rightsquigarrow$ timed automata, timed Petri nets, timed process algebras, ...

- Need for time in specification:
  - again, the behaviour of most systems depends on time;
  - untimed specifications are not enough (e.g., *bounded* response property);

$\rightsquigarrow$ TCTL, MTL, TPTL, timed $\mu$-calculus, ...

# Time is not always sufficient

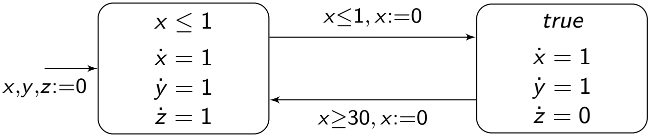
In some cases, we don't want to measure time, but rather energy consumption, price to pay for reaching some goal, ...

# Time is not always sufficient

In some cases, we don't want to measure time, but rather energy consumption, price to pay for reaching some goal, ...

- hybrid automata: timed automata augmented with variables whose derivative is not constant.

$\rightsquigarrow$ examples: leaking gas burner, water-level monitor, ...




**Theorem (HKPV97)**

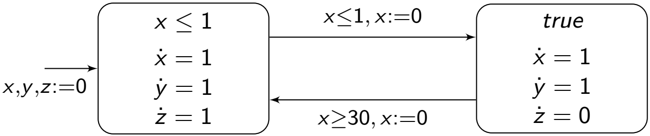
*Reachability is undecidable (even for timed automata where one single "clock" has two derivatives).*

# Time is not always sufficient

In some cases, we don't want to measure time, but rather energy consumption, price to pay for reaching some goal, ...

- hybrid automata: timed automata augmented with variables whose derivative is not constant.

$\leadsto$ examples: leaking gas burner, water-level monitor, ...




- priced timed automata: similar to hybrid automata, but the behavior only depends on clock variables.

# Related work on priced timed automata

- **Basic properties**
  - Optimal reachability [ATP01,BFH+01,LBB+01,BBBR06]
  - Mean-cost optimality [BBL04]
- **Control games**
  - Properties, and restricted decidability results [ABM04,BCFL04,BCFL05]
  - Undecidability for timed game automata with more than three clocks [BBR05,BBM06]
  - Decidability for timed automata with one clock [BLMR06]
- **Model-checking of WCTL**
  - Undecidability for timed automata with more than three clocks [BBR04,BBM06]
  - Decidability for timed automata with one clock [BLM07]

# Outline of the talk

1. Introduction
2. Definitions and examples
3. Existence of optimal strategies in 1PTGAs is decidable
4. (Pseudo-)algorithm for computing the optimal cost
5. Conclusion

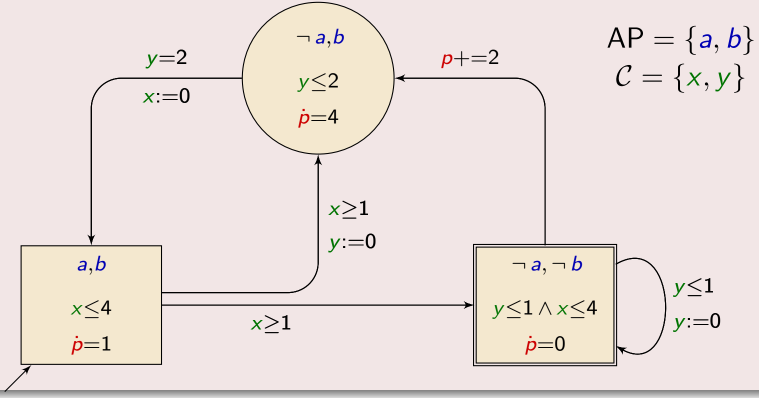
# Outline of the talk

# Priced timed game automata

## Definition (ALP01,BFH+01)

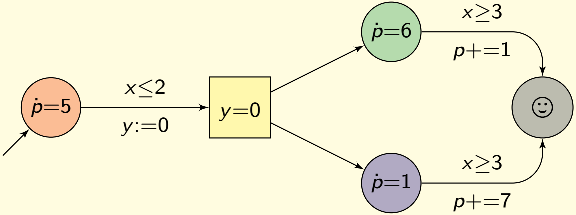
A *priced timed game automaton* is a timed automaton with costs where states are partitionned into *controllable* and *uncontrollable* ones:

$$\mathcal{G} = \langle Q_c \cup Q_u, Q_0, \mathrm{AP}, \ell, \delta, \mathcal{C}, \mathrm{G}, \mathrm{R}, \mathrm{I}, Q_{\mathrm{urg}}, \mathrm{P} \rangle$$



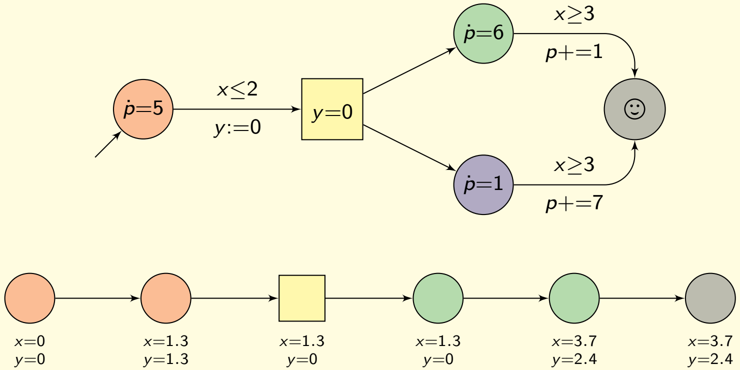
# Example

## Example



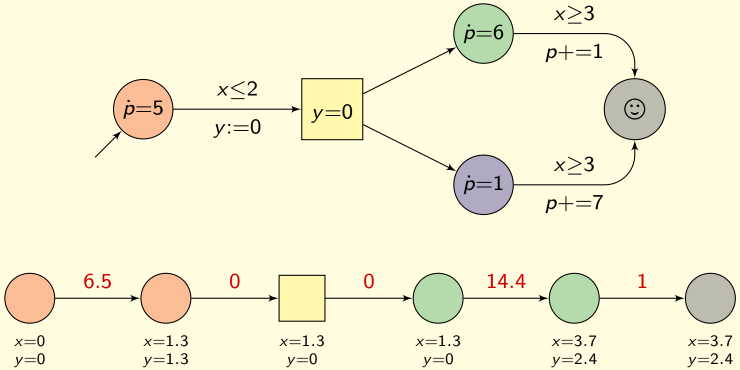
# Example

## Example



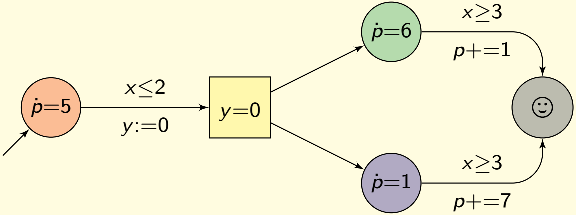
# Example

## Example



# Example

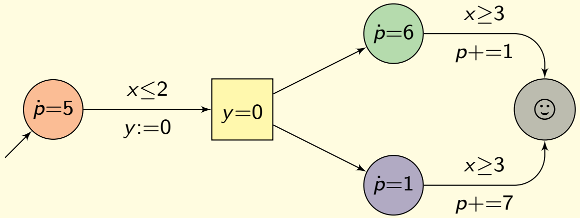
## Example




Minimal cost for reaching ☺:

# Example

## Example

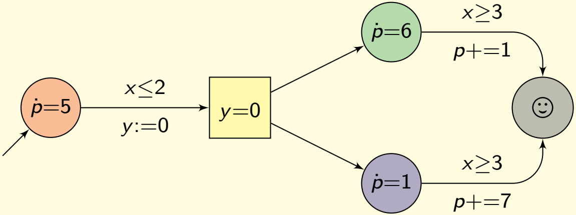



Minimal cost for reaching ☺:

$$5t + 6(3 - t) + 1$$

# Example

## Example

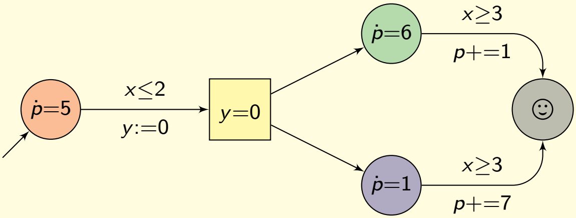



Minimal cost for reaching ☺:

$$5t + 6(3 - t) + 1,\ 5t + (3 - t) + 7$$

# Example

## Example

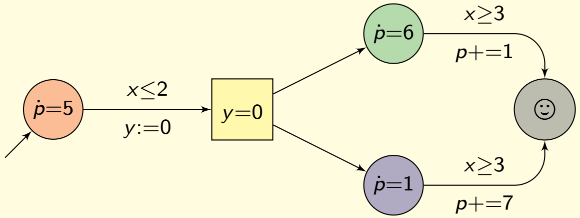



Minimal cost for reaching ☺:

$$\max\left(5t + 6(3 - t) + 1,\ 5t + (3 - t) + 7\right)$$

# Example

## Example

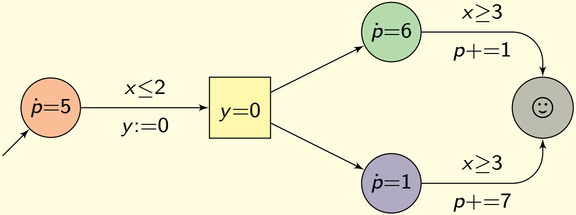



Minimal cost for reaching ☺:

$$\inf_{0 \leq t \leq 2} \max \left(5t + 6(3 - t) + 1,\ 5t + (3 - t) + 7\right)$$

# Example

## Example




Minimal cost for reaching ☺:

$$\inf_{0\le t\le 2} \max\left(5t + 6(3-t) + 1,\ 5t + (3-t) + 7\right) = 17.2$$

(when $t = 1.8$)

# Strategies

## Definitions

- Run$(A, B)$ is the set of trajectories from some state in $A$ to some state in $B$;
- a strategy is a function

$$\sigma\colon \mathrm{Run}(Q \times \mathbb{R}^{+\mathcal{C}}, Q_c \times \mathbb{R}^{+\mathcal{C}}) \longrightarrow \delta \cup \mathbb{R}^{+}_{>0}$$

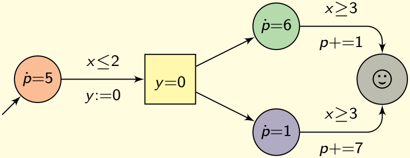
# Strategies

## Definitions

- Run$(A, B)$ is the set of trajectories from some state in $A$ to some state in $B$;
- a strategy is a function

$$\sigma : \text{Run}(Q \times \mathbb{R}^{+\mathcal{C}}, Q_c \times \mathbb{R}^{+\mathcal{C}}) \to \delta \cup \mathbb{R}^{+}_{>0}$$

## Example




Example of a strategy $\sigma$:

- in ◯, wait until $x = 2$;
- in ◯, wait until $x = 3$;
- in ◯, wait until $x = 4$;

# Strategies

## Definitions

- a run $\rho = ((q_i, v_i))_{i \in \mathbb{Z}^+}$ is compatible with a strategy $\sigma$ from step $i_0$ if, for each $i \geq i_0$ s.t. $q_i \in Q_c$,
  - if $\sigma(\rho_{\leq i}) = e \in \delta$ and $v_i \models \mathrm{I}(q_i)$ and $v_i \models \mathrm{G}(e)$, then $e = (q_i, q_{i+1})$ and $v_{i+1} = v_i[\mathrm{R}(e) \leftarrow 0]$.
  - if $\sigma(\rho_{\leq i}) = r \in \mathbb{R}^+_{>0}$ and, for all $t \in [0, r]$, $v_i + t \models \mathrm{I}(q_i)$, then $q_{i+1} = q_i$ and $v_{i+1} = v_i + r$.
- a strategy $\sigma$ is winning (for some reachability objective $W \subseteq Q$) after some finite prefix $\rho_0$ if any "prolongation" of $\rho_0$ that is compatible with $\sigma$ after $\rho_0$, reaches a location in $W$.

# Strategies

## Definitions

- the cost of a winning strategy $\sigma$ from $\rho_0$ is

$$\mathrm{Cost}(\sigma, \rho_0) = \sup\{\mathrm{cost}(\rho) \mid \rho \text{ compatible execution after } \rho_0\}$$

(assuming that the trajectory stops as soon as it enters any location in $W$).

# Strategies

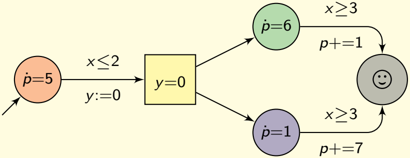
## Definitions

- the cost of a winning strategy $\sigma$ from $\rho_0$ is

$$\mathrm{Cost}(\sigma, \rho_0) = \sup\{\mathrm{cost}(\rho) \mid \rho \text{ compatible execution after } \rho_0\}$$

(assuming that the trajectory stops as soon as it enters any location in $W$).

## Example




Consider strategy $\sigma$:

- in , wait until $x = 2$;
- in , wait until $x = 3$;
- in , wait until $x = 4$;

$$\mathrm{Cost}(\sigma, (\bigcirc, x = 0)) = \sup(17, 19) = 19.$$

# Bad news!

**Theorem (BBR05,BBM06)**

*The existence of a strategy with cost less than or equal to a given value is undecidable on PTGAs.*

# Bad news!

**Theorem (BBR05,BBM06)**

*The existence of a strategy with cost less than or equal to a given value is undecidable on PTGAs.*

*Idea of the proof.* Encoding of a two-counter machine. □

The reduction can be achieved involving only three clocks.

# Bad news!

**Theorem (BBR05,BBM06)**

*The existence of a strategy with cost less than or equal to a given value is undecidable on PTGAs.*

*Idea of the proof.* Encoding of a two-counter machine. □

The reduction can be achieved involving only three clocks.

What happens with only one clock?

# Outline of the talk

# Strategies

## Definitions

- $\mathsf{Run}(A, B)$ is the set of trajectories from some state in $A$ to some state in $B$;
- a strategy is a function

$$\sigma\colon \mathsf{Run}(Q \times \mathbb{R}^+, Q_c \times \mathbb{R}^+) \to \delta \cup \mathbb{R}^+_{>0}$$

- a strategy is memoryless if it only depends on the present state:

$$\sigma\colon Q_c \times \mathbb{R}^+ \to \delta \cup \mathbb{R}^+_{>0}$$

# Strategies

## Definitions

- the cost of a winning strategy $\sigma$ from $\rho_0$ is

$$\mathrm{Cost}(\sigma, \rho_0) = \sup\{\mathrm{cost}(\rho) \mid \rho \text{ compatible execution after } \rho_0\}$$

  (assuming that the trajectory stops as soon as it enters any location in $W$).

- the optimal cost of winning from some state $s$ is

$$\mathrm{OptCost}(s) = \inf\{\mathrm{Cost}(\sigma, s) \mid \sigma \text{ winning strategy}\}$$

- a strategy $\sigma$ is $\varepsilon$-optimal in state $s$ if

$$\mathrm{OptCost}(s) \leq \mathrm{Cost}(\sigma, \rho_0) \leq \mathrm{OptCost}(s) + \varepsilon$$

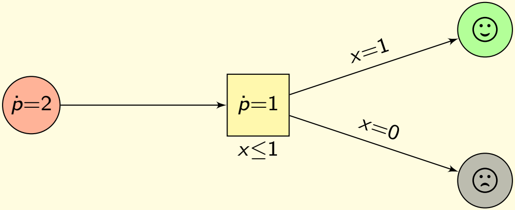
- a strategy is optimal if it is 0-optimal.

# Memorylessness and optimality

**Fact**

*In our PTGAs, optimal strategies do not always exist.*

**Example**




In this example, only $\varepsilon$-optimal strategies exist, for any $\varepsilon > 0$.

# Memorylessness and optimality

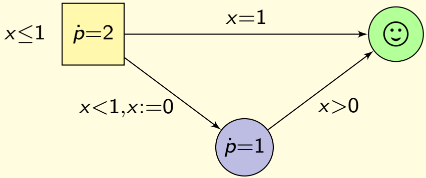
**Fact**

*In our PTGAs, optimal strategies do not always exist.*

**Fact**

*When optimal strategies exist, they might require some memory.*

**Example**




An optimal strategy depends on the date at which the blue state is entered.

# Memorylessness and optimality

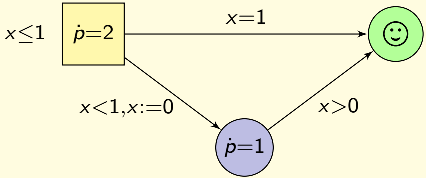
**Fact**

*In our PTGAs, optimal strategies do not always exist.*

**Fact**

*When optimal strategies exist, they might require some memory.*

**Example**




An optimal strategy depends on the date at which the blue state is entered. But there is a memoryless $\varepsilon$-optimal strategy.

# Decidability of 1PTGAs

**Definition**

Given $\varepsilon > 0$ and $N \in \mathbb{Z}^+$, a strategy $\sigma$ is $(\varepsilon, N)$ acceptable if

- $\sigma$ is $\varepsilon$-optimal and memoryless,
- there is a partition $(I_n)_{n \leq N}$ of $[0, M]$ (where $M$ is the maximal constant of the guards and invariants of the game) s.t., for any $q \in Q_c$, $x \mapsto \sigma(q, x)$ is constant on each $I_n$.

# Decidability of 1PTGAs

## Definition

Given $\varepsilon > 0$ and $N \in \mathbb{Z}^+$, a strategy $\sigma$ is $(\varepsilon, N)$ acceptable if

- $\sigma$ is $\varepsilon$-optimal and memoryless,
- there is a partition $(I_n)_{n \leq N}$ of $[0, M]$ (where $M$ is the maximal constant of the guards and invariants of the game) s.t., for any $q \in Q_c$, $x \mapsto \sigma(q, x)$ is constant on each $I_n$.

## Main Theorem

- For every location, the optimal cost is computable and is piecewise affine.
- There exists $N \in \mathbb{Z}^+$ s.t., for any $\varepsilon > 0$, we can effectively compute an $(\varepsilon, N)$-acceptable (thus, almost-optimal and memoryless) strategy.

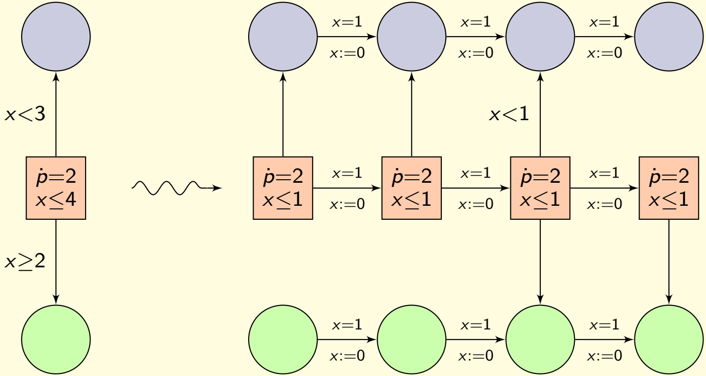
# Simplifying the problem

We restrict to TGAs with maximal constant 1 (in clock constraints)

# Simplifying the problem

We restrict to TGAs with maximal constant 1 (in clock constraints)

## Example



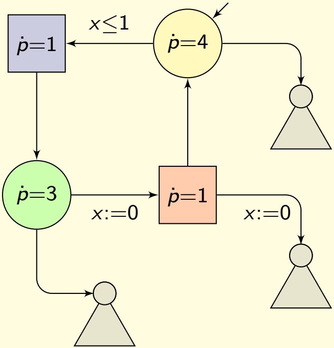
# Simplifying the problem

We restrict to strongly-connected TGAs without resets.

# Simplifying the problem

We restrict to strongly-connected TGAs without resets.

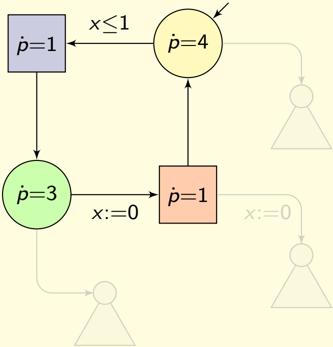
## Example



# Simplifying the problem

We restrict to strongly-connected TGAs without resets.

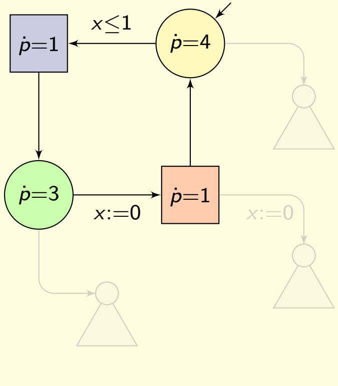
## Example



# Simplifying the problem

We restrict to strongly-connected TGAs without resets.

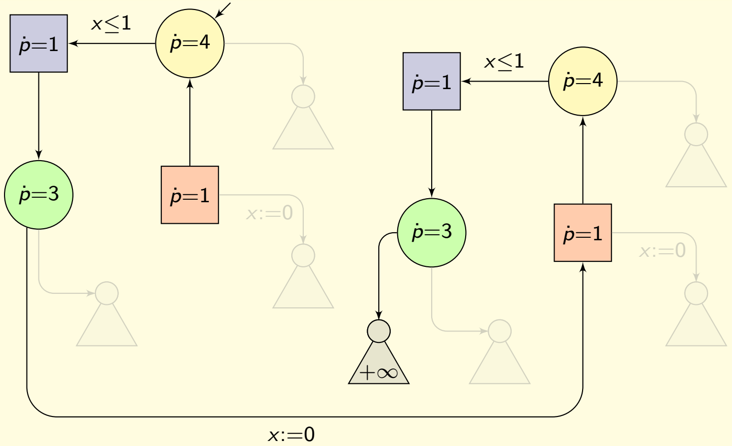
## Example



# Simplifying the problem

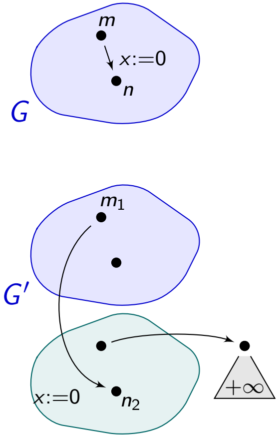
We restrict to strongly-connected TGAs without resets.

## Example



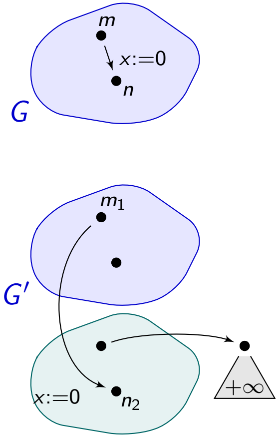
# Simplifying the problem

We restrict to strongly-connected TGAs without resets.



# Simplifying the problem

We restrict to strongly-connected TGAs without resets.

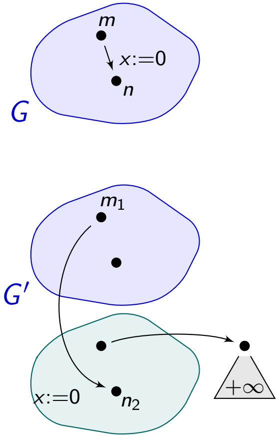



**Theorem**

$$\mathrm{OptCost}_G(q, x) = \mathrm{OptCost}_{G'}(q_1, x).$$

# Simplifying the problem

We restrict to strongly-connected TGAs without resets.




**Theorem**

$$\mathrm{OptCost}_G(q, x) = \mathrm{OptCost}_{G'}(q_1, x).$$

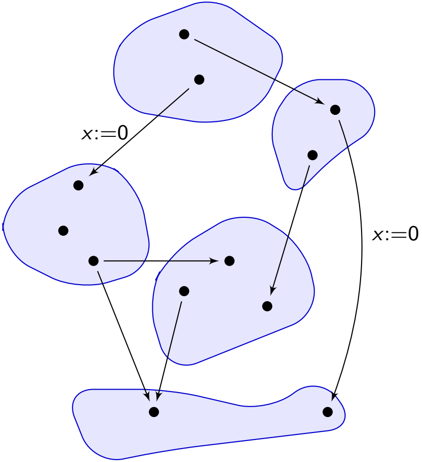
**Theorem**

If $\sigma'$ is $(\varepsilon', N')$-acceptable in $G'$, then

$$\sigma(q,x) = \begin{cases} \sigma'(q_2, x) & \\ \quad \text{if } \mathrm{Cost}(q_2, x) \le \mathrm{Cost}(q_1, x) & \\ \sigma'(q_1, x) \quad \text{otherwise} & \end{cases}$$

is $(2\varepsilon', N')$-acceptable in $G$.

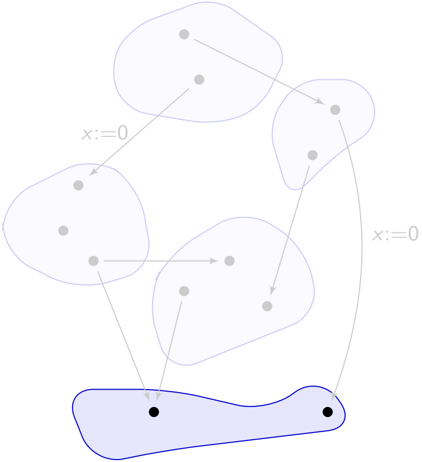
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

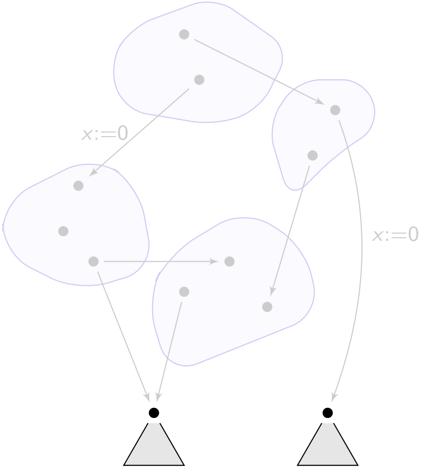
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

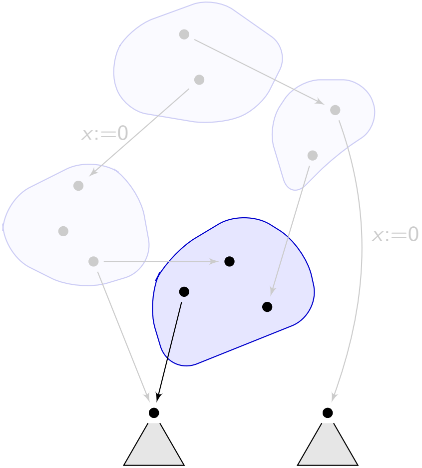
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

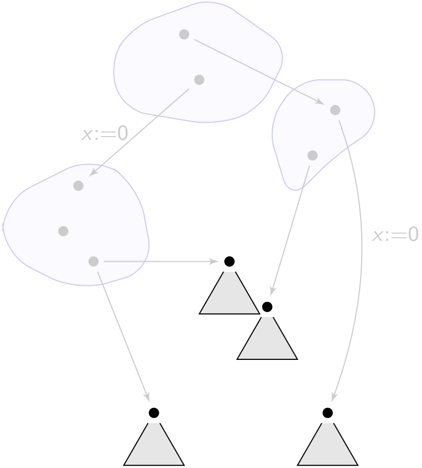
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

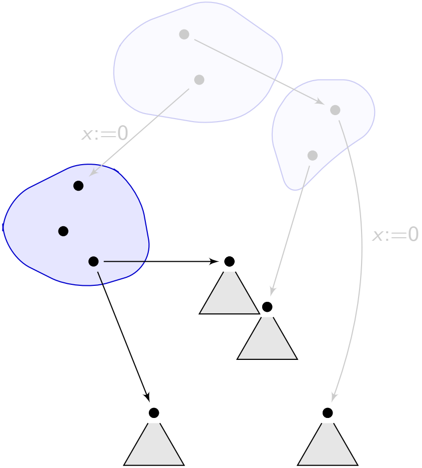
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

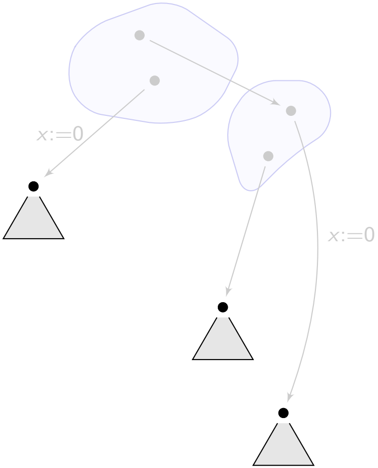
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

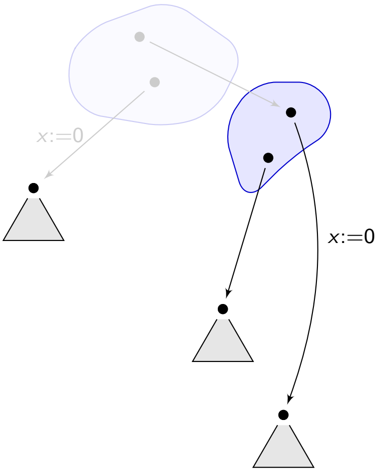
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

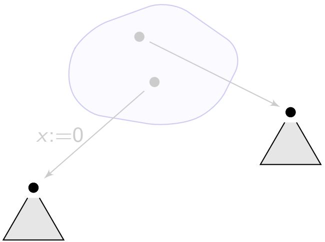
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

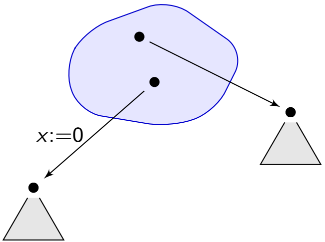
# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

# Simplifying the problem




Reduced to

- strongly-connected PTGAs
- clock is bounded by 1
- no resetting transitions.

# Main theorem with outside cost-functions

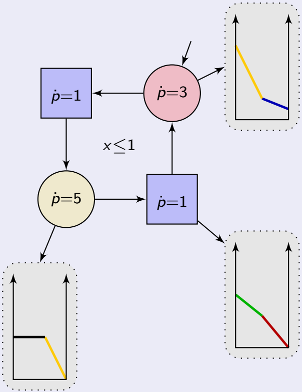
## Theorem

*Let $G$ be a strongly-connected non-resetting 1PTGA with outside cost-functions.*

- $\mathsf{OptCost}_G$ *is computable;*
- *in each location, function*

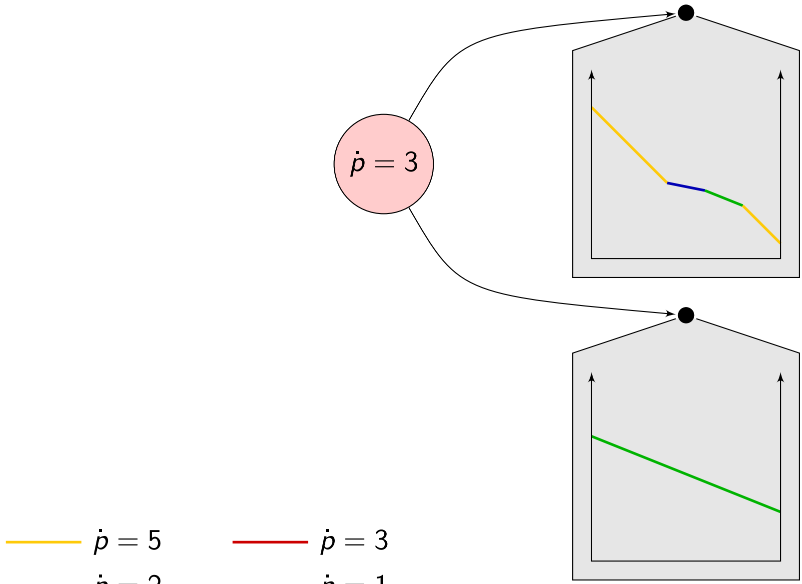
$$x \mapsto \mathsf{OptCost}_G(q, x)$$

  *is decreasing, piecewise affine and continuous. Its finitely many segments either have slope $-c$ where $c$ is the price of some locations, or are fragments of the outside cost-functions;*

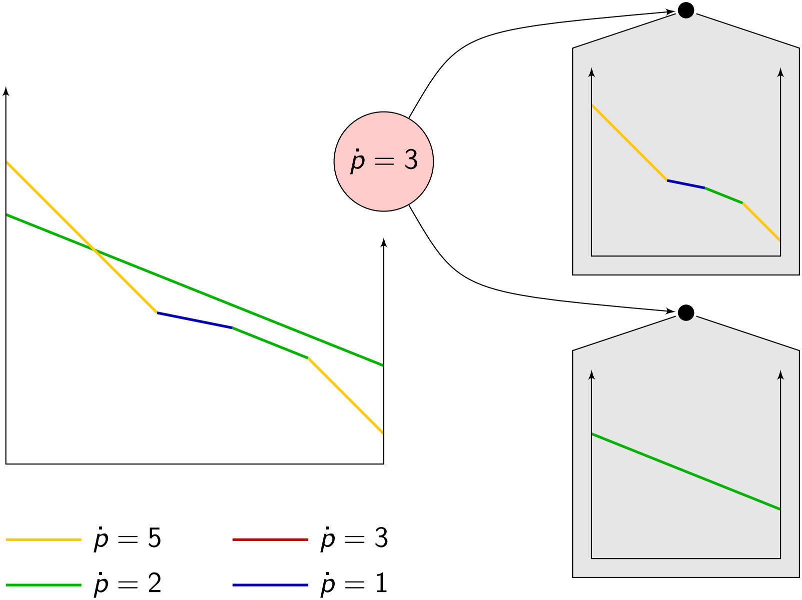



- *There exists $N \in \mathbb{Z}^+$ s.t., for any $\varepsilon > 0$, we can compute an $(\varepsilon, N)$-acceptable strategy $\sigma$.*

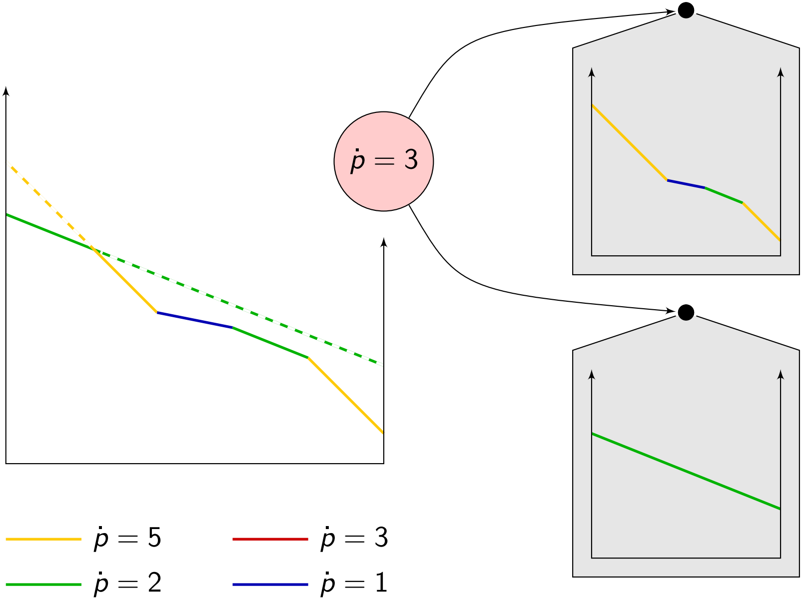
# Operations on cost functions: controllable locations



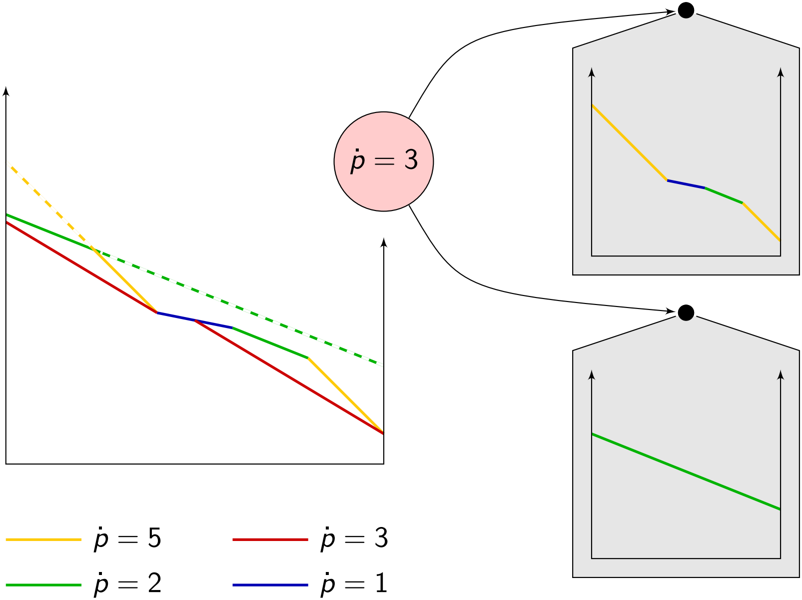
# Operations on cost functions: controllable locations



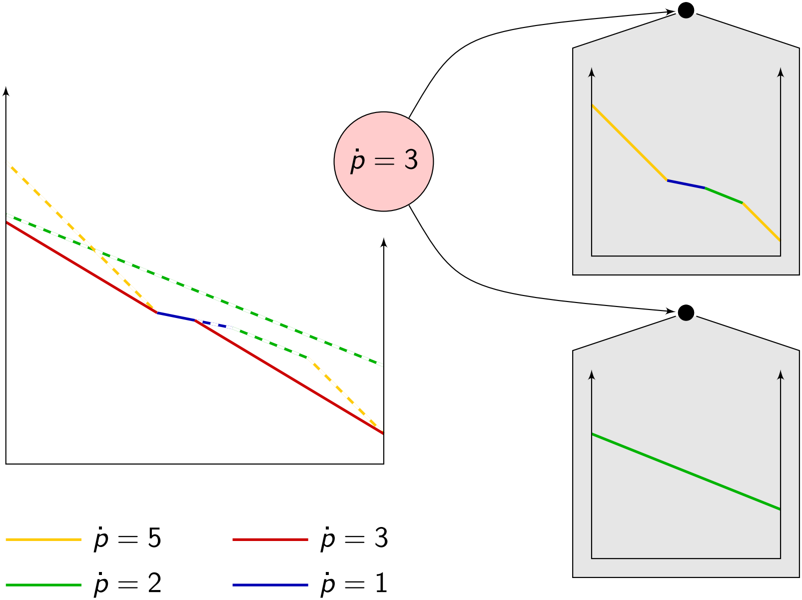
# Operations on cost functions: controllable locations



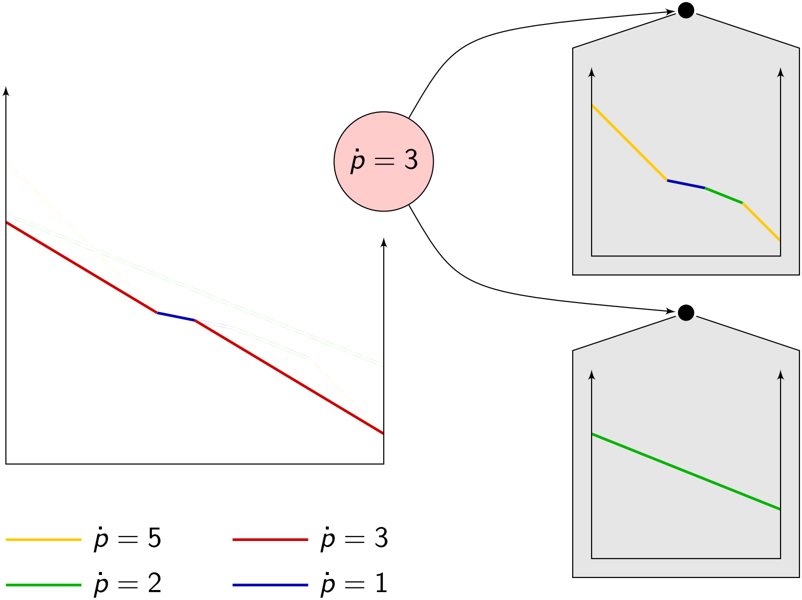
# Operations on cost functions: controllable locations



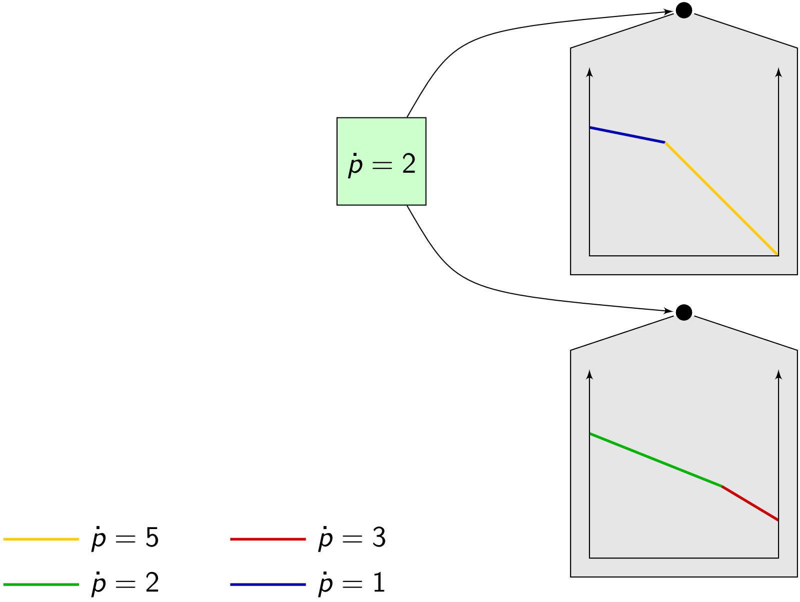
# Operations on cost functions: controllable locations



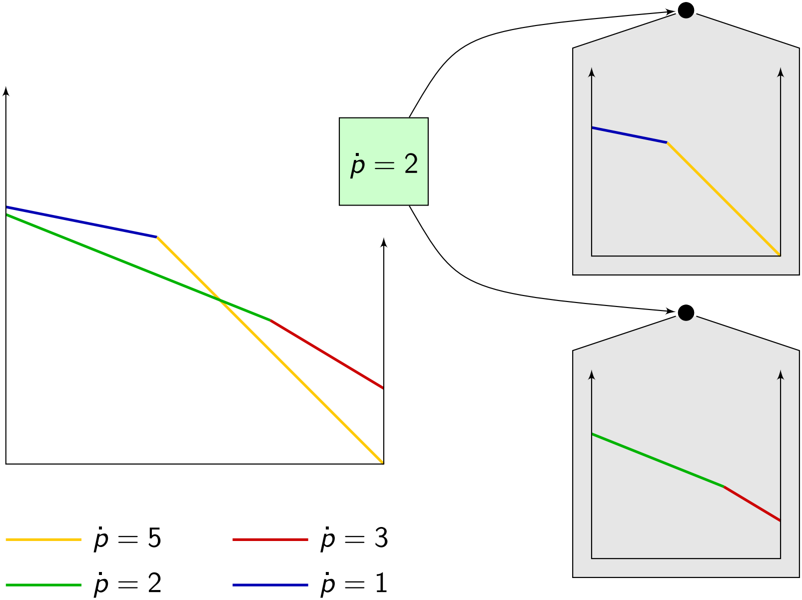
# Operations on cost functions: controllable locations



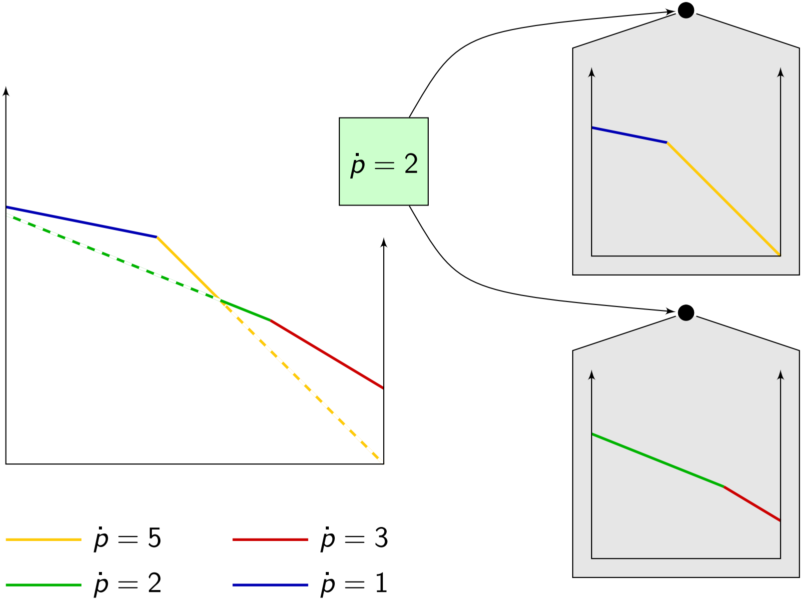
# Operations on cost functions: uncontrollable locations



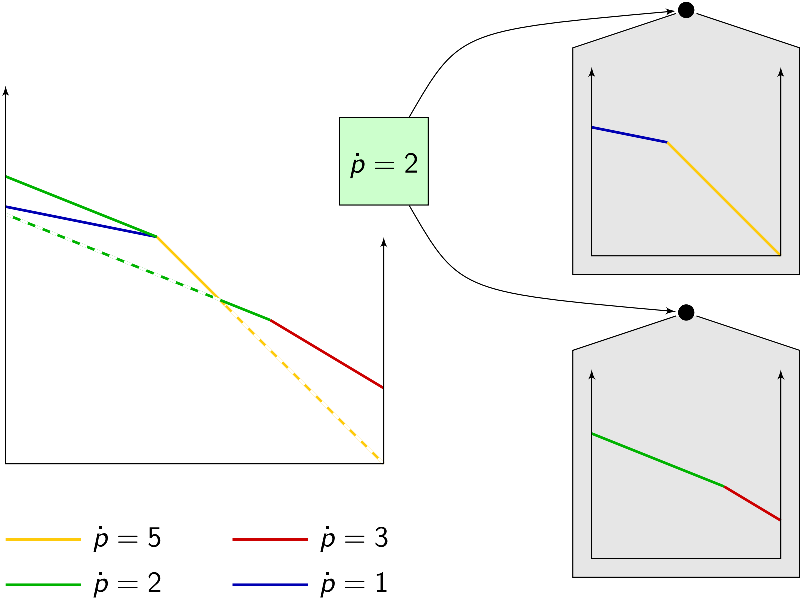
# Operations on cost functions: uncontrollable locations



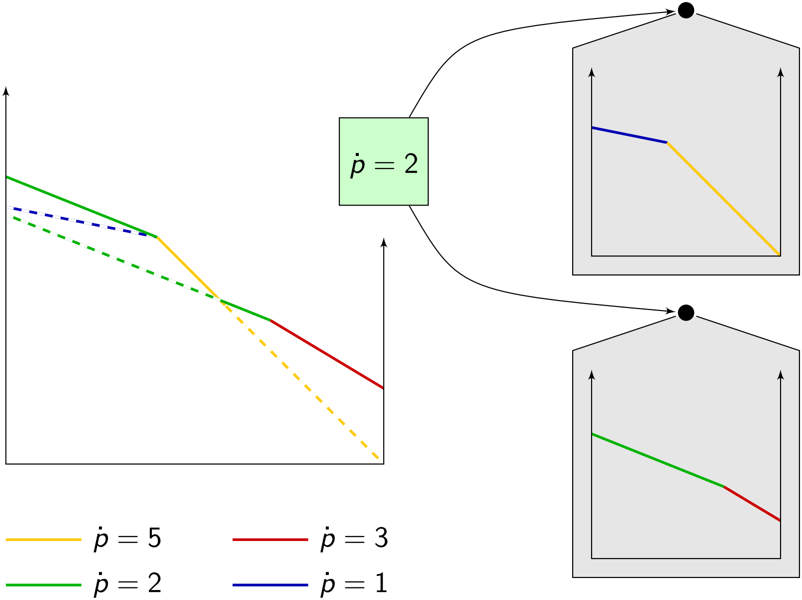
# Operations on cost functions: uncontrollable locations



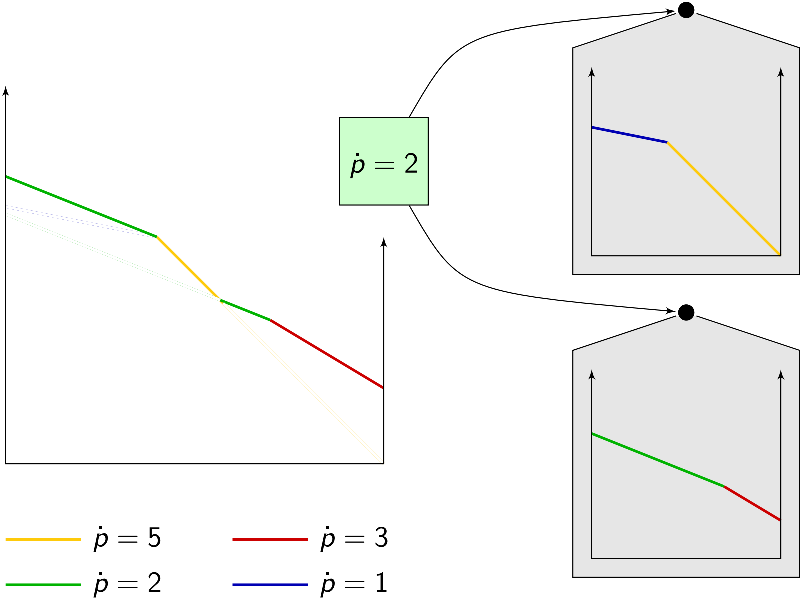
# Operations on cost functions: uncontrollable locations



# Operations on cost functions: uncontrollable locations



# Operations on cost functions: uncontrollable locations



# Inductive proof

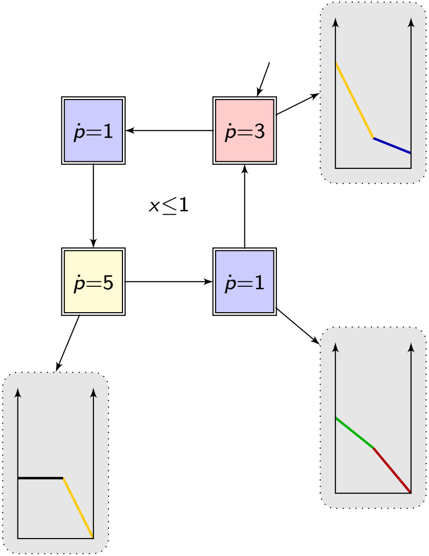
## Ideas of the proof

Induction on the number of non-urgent locations in the SCC

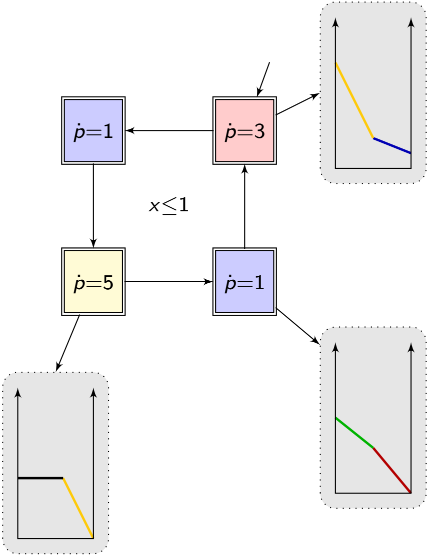
- base cases:
  - all locations are urgent (thus uncontrollable);
  - there is only one location, which is controllable (thus non-urgent).
- induction step:
  we consider one of the non-urgent locations having minimal cost rate:
  - if it is controllable, we create two SCCs having one less non-urgent location;
  - if it is uncontrollable, we make it urgent and add an extra outside cost function to which it can go.

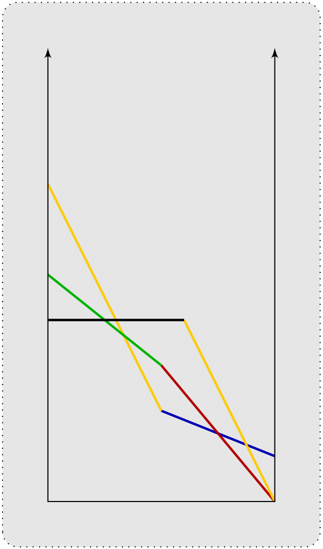
Skip proof

# Inductive proof – base cases

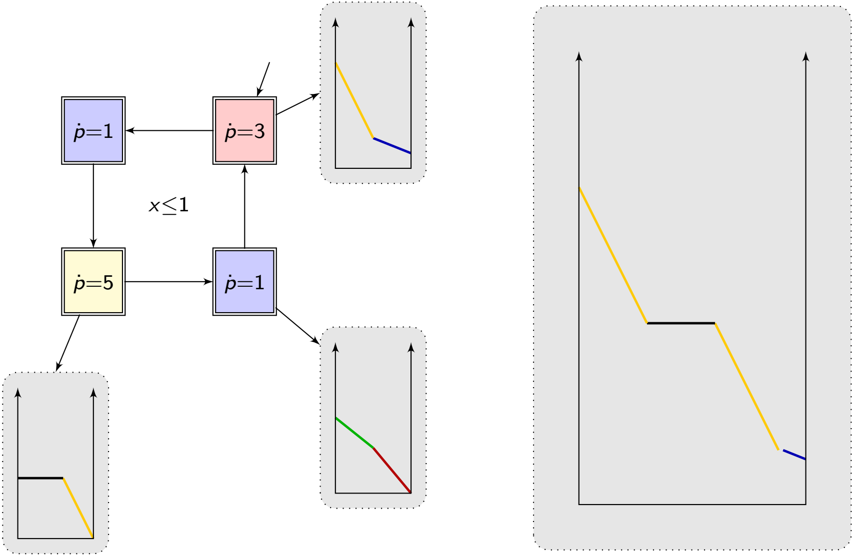


# Inductive proof – base cases

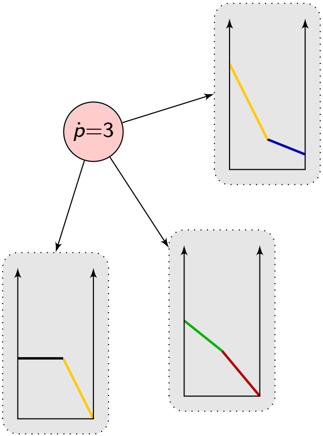


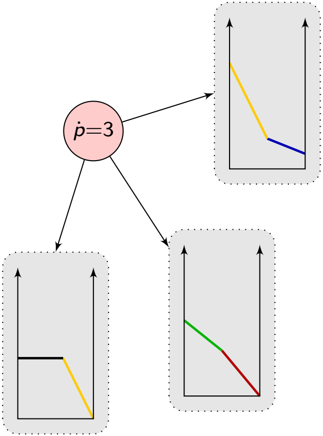
# Inductive proof – base cases

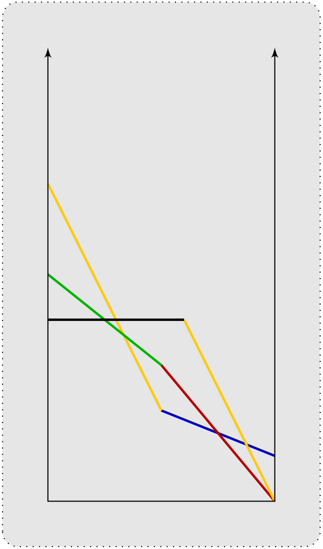


# Inductive proof – base cases

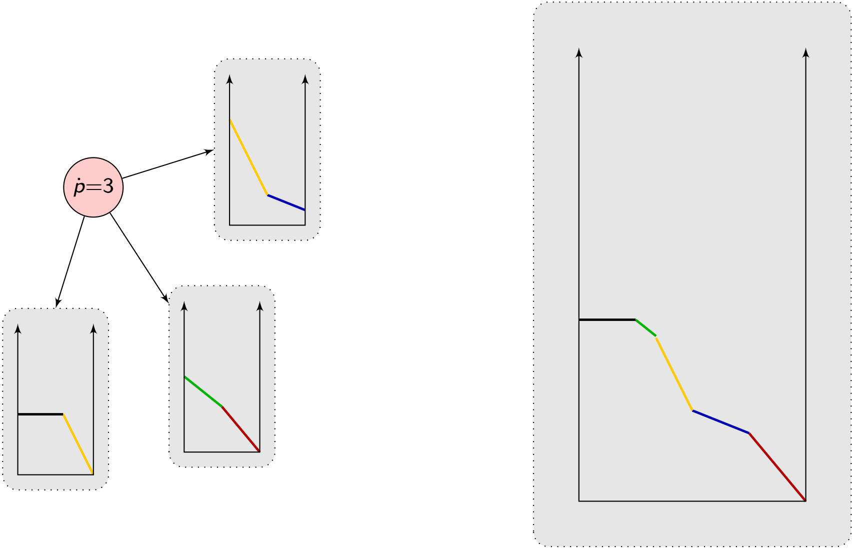


# Inductive proof – base cases

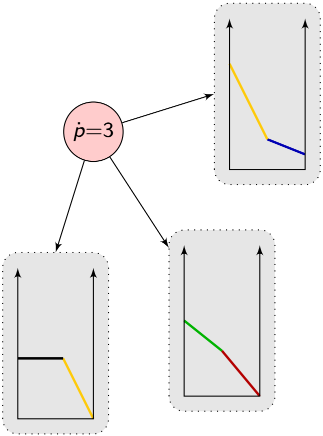


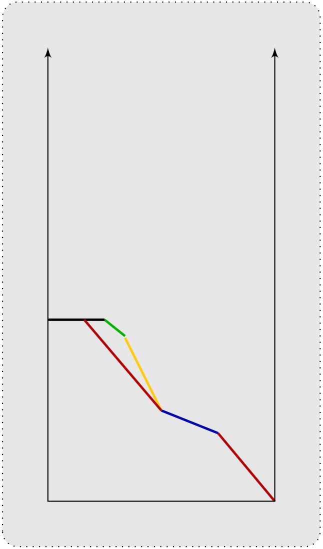



# Inductive proof – base cases

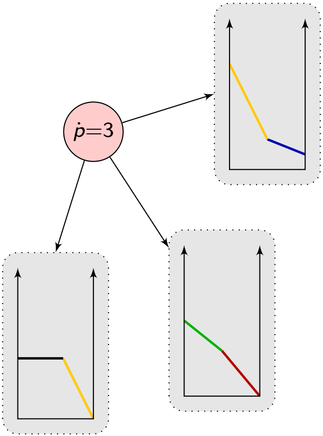


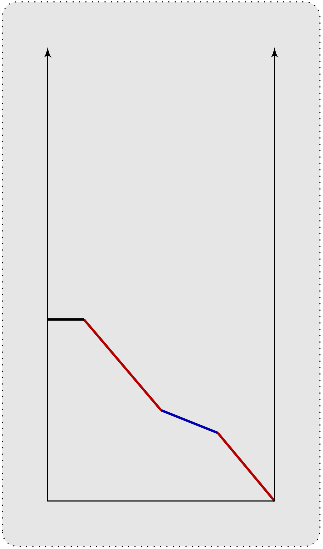
# Inductive proof – base cases



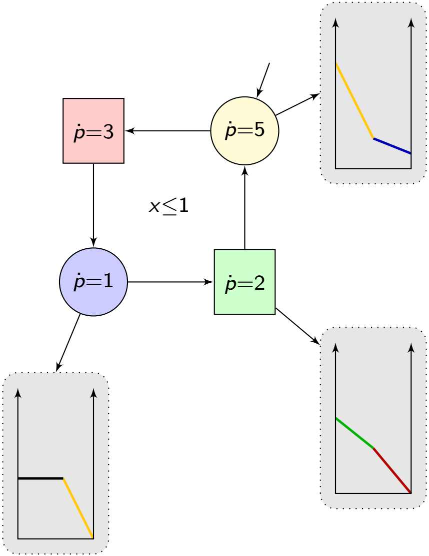
# Inductive proof – base cases



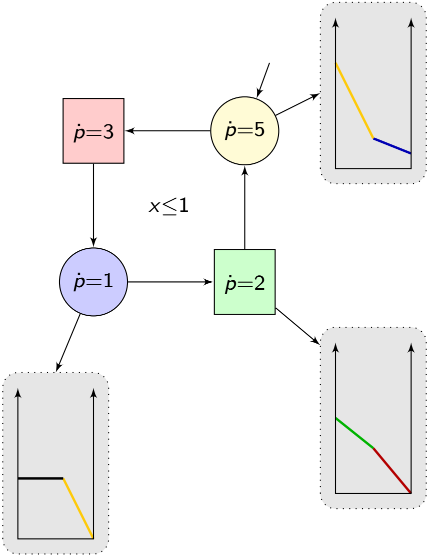
# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:



# Inductive proof – inductive cases

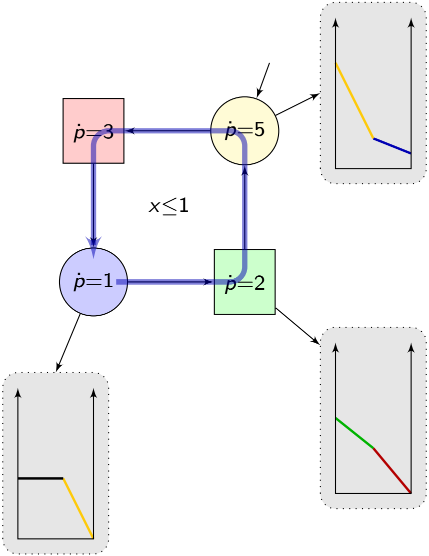
- When $q_{\min}$ is controllable:




Let $\sigma$ be a winning strategy.

# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:




Let $\sigma$ be a winning strategy.

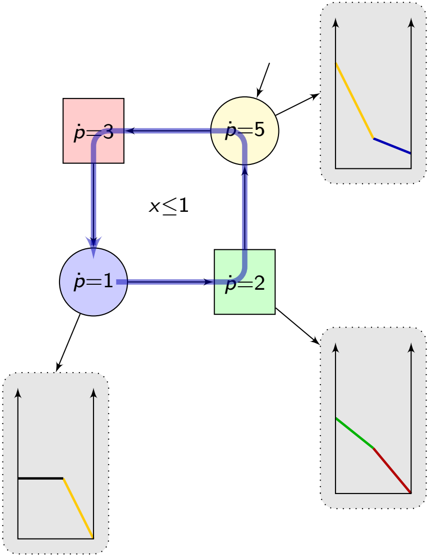
Assume there exists an outcome of $\sigma$ s.t.:

$$(q_{\min}, u) \rightarrow^* (q_{\min}, v) \rightarrow^* \text{win}$$

with $0 \leq u < v \leq 1$.

# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:




Let $\sigma$ be a winning strategy.

Assume there exists an outcome of $\sigma$ s.t.:

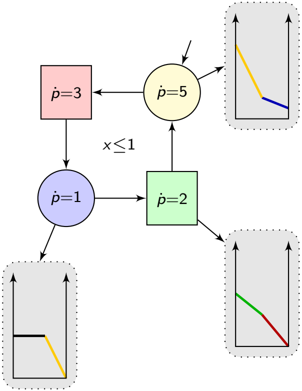
$$(q_{\min}, u) \rightarrow^* (q_{\min}, v) \rightarrow^* \text{win}$$

with $0 \leq u < v \leq 1$.

Then $\sigma$ is not optimal: waiting in $q_{\min}$ would have been cheaper.

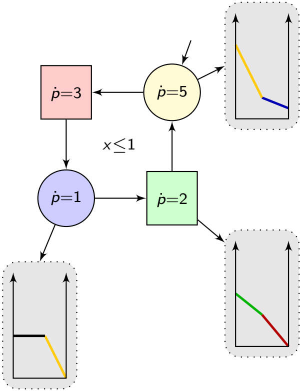
# Inductive proof – inductive cases

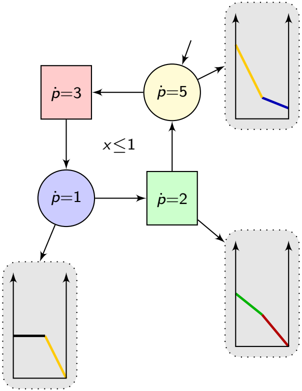
- When $q_{\min}$ is controllable:



# Inductive proof – inductive cases

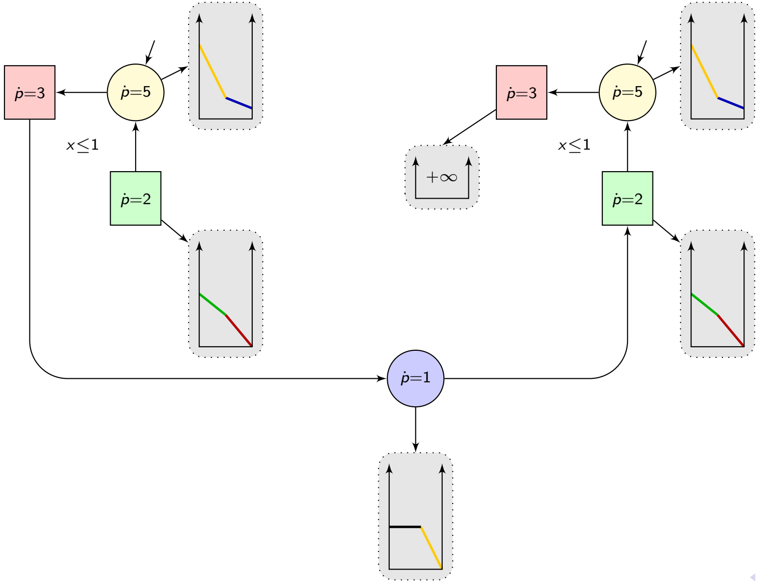
- When $q_{\min}$ is controllable:



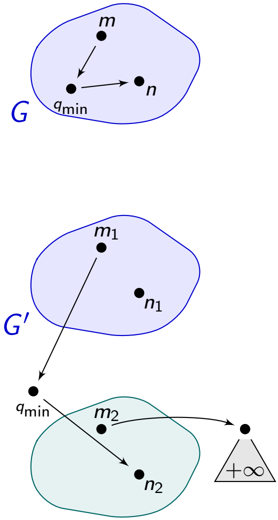
# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:



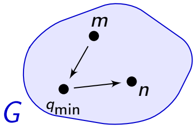
# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:



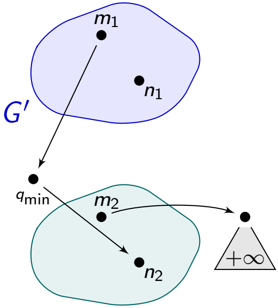
# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:



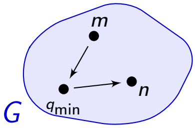

**Theorem**

$$\mathrm{OptCost}_{G'}(q_1, x) = \mathrm{OptCost}_G(q, x).$$



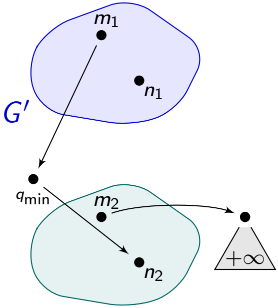
# Inductive proof – inductive cases

- When $q_{\min}$ is controllable:




**Theorem**

$$\mathrm{OptCost}_{G'}(q_1, x) = \mathrm{OptCost}_G(q, x).$$




**Theorem**

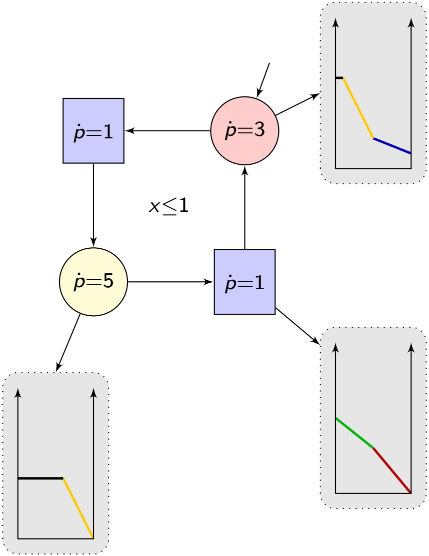
Let $\sigma'$ be an $(\varepsilon', N')$-acceptable strategy for $G'$. Let

$$\sigma(q, x) = \begin{cases} \sigma'(q_2, x) & \text{if } \mathrm{Cost}_{G'}(q_2,x) \leq \mathrm{OptCost}_{G'}(q_{\min},x) \\ \sigma'(q_1, x) & \text{otherwise} \end{cases}$$

Then $\sigma$ is $(3\varepsilon', N)$-acceptable in $G$, for some $N$ independant of $\varepsilon'$.

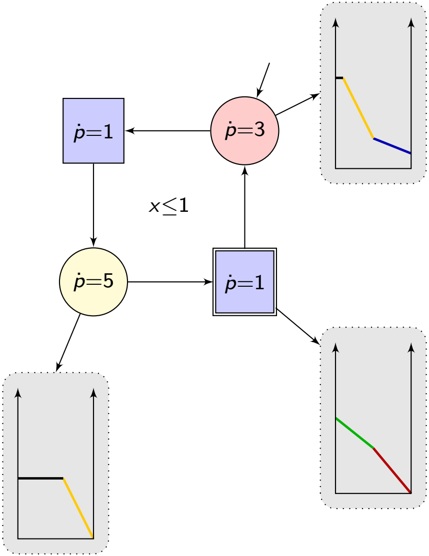
# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:



# Inductive proof – inductive cases

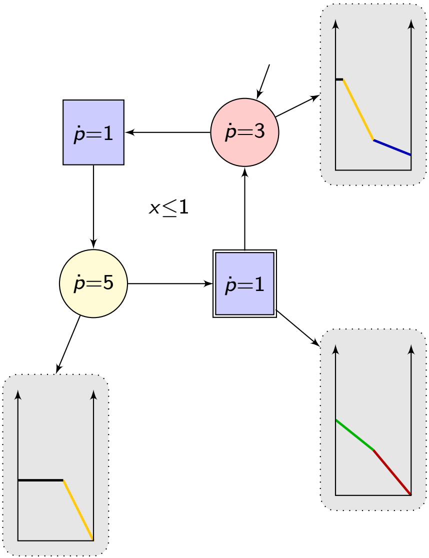
- When $q_{\min}$ is uncontrollable:



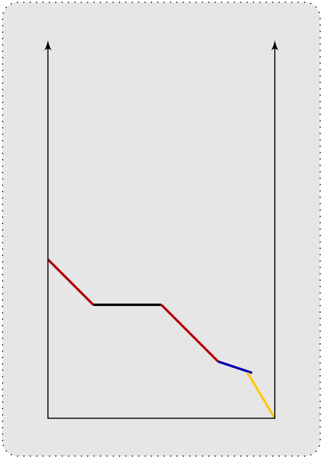

Make $q_{\min}$ urgent and apply I.H.:

# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:

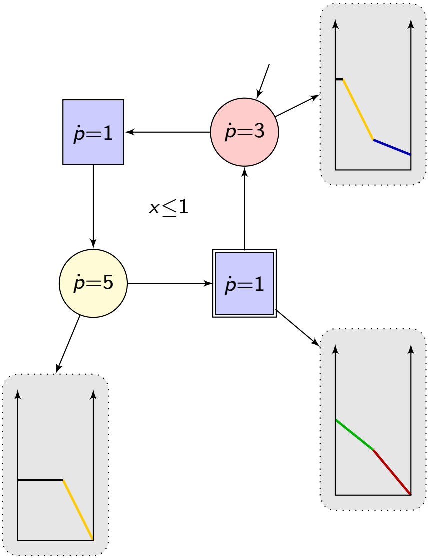



Make $q_{\min}$ urgent and apply I.H.:

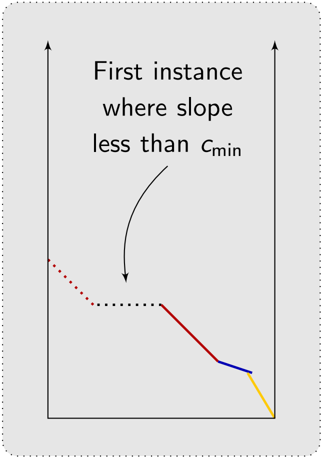


# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:

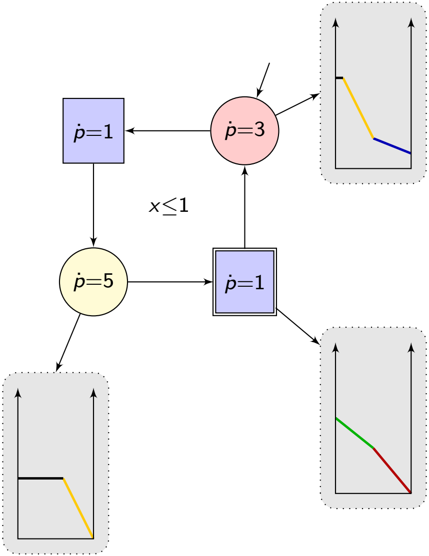



Make $q_{\min}$ urgent and apply I.H.:

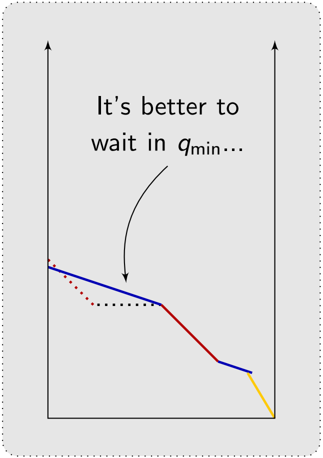


# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:



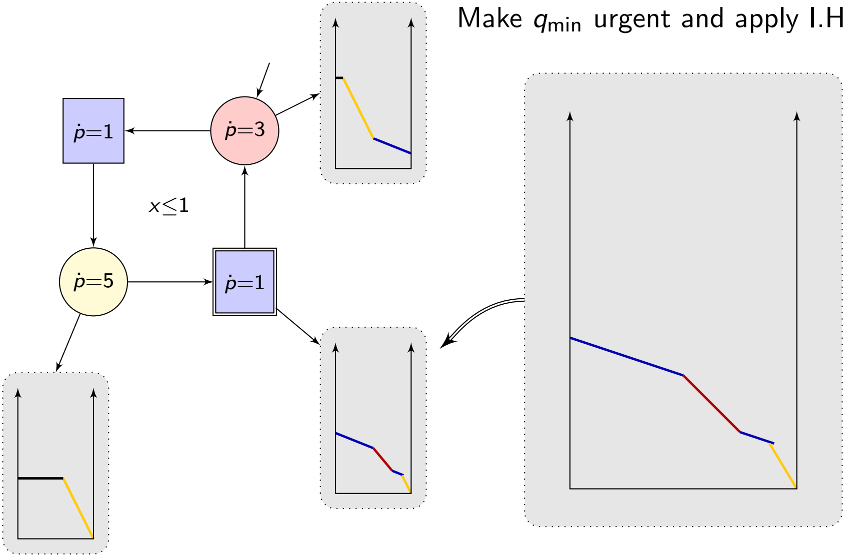

Make $q_{\min}$ urgent and apply I.H.:



# Inductive proof – inductive cases

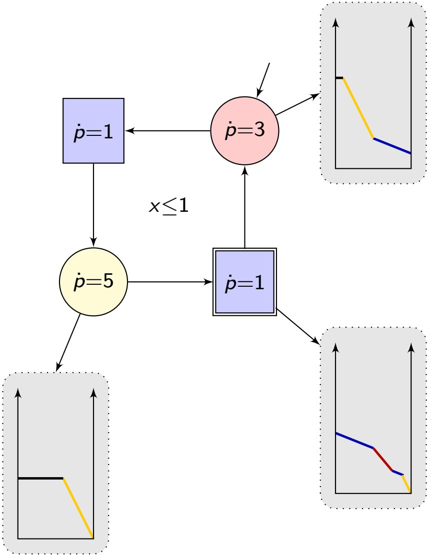
- When $q_{\min}$ is uncontrollable:

Make $q_{\min}$ urgent and apply I.H.:

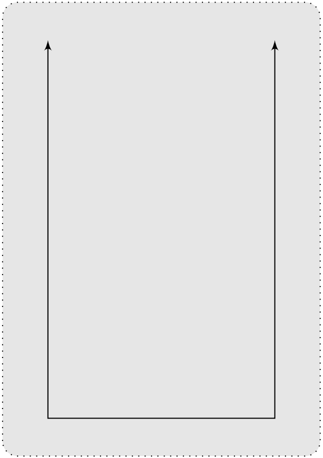


# Inductive proof – inductive cases

- When $q_{\text{min}}$ is uncontrollable:

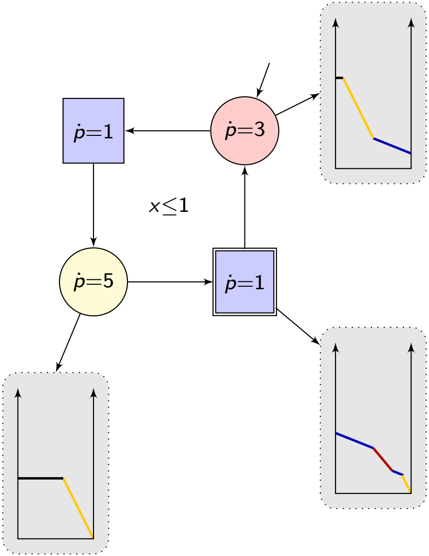



Apply I.H. again:

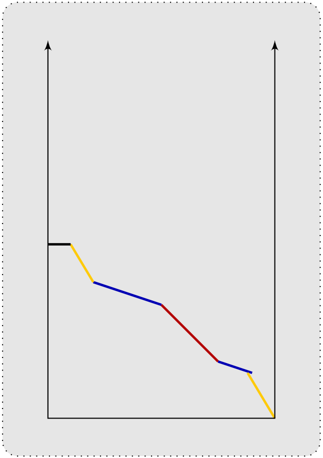


# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:

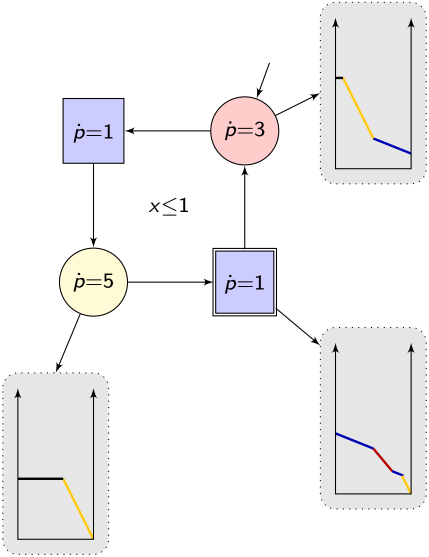



Apply I.H. again:

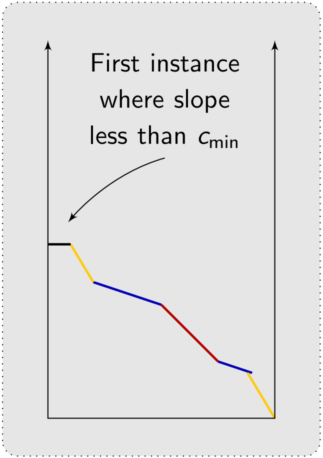


# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:

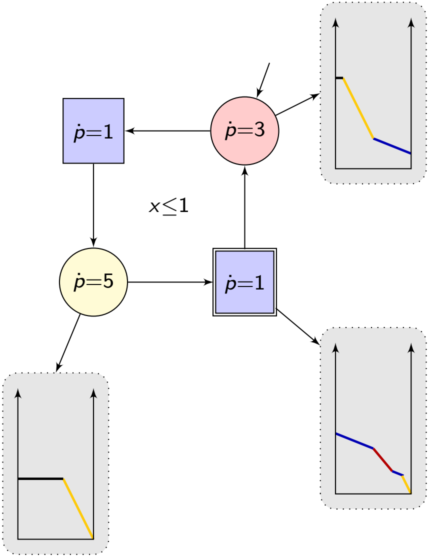



Apply I.H. again:

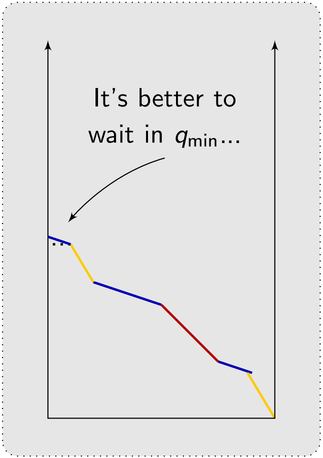


# Inductive proof – inductive cases

- When $q_{\min}$ is uncontrollable:

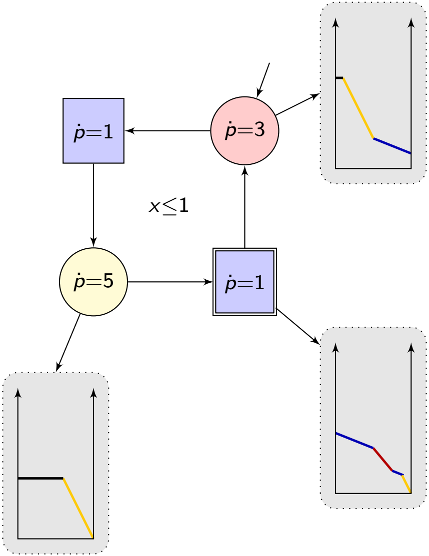



Apply I.H. again:



# Inductive proof – inductive cases

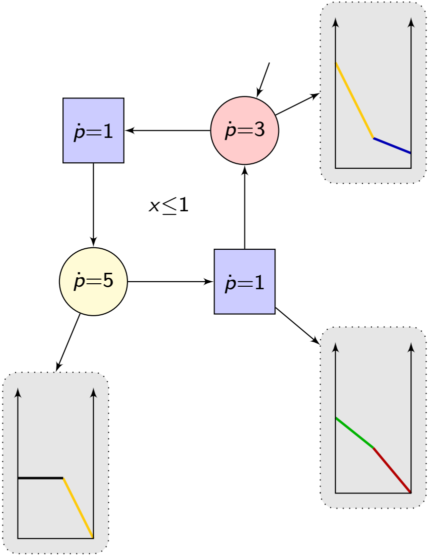
- When $q_{\min}$ is uncontrollable:



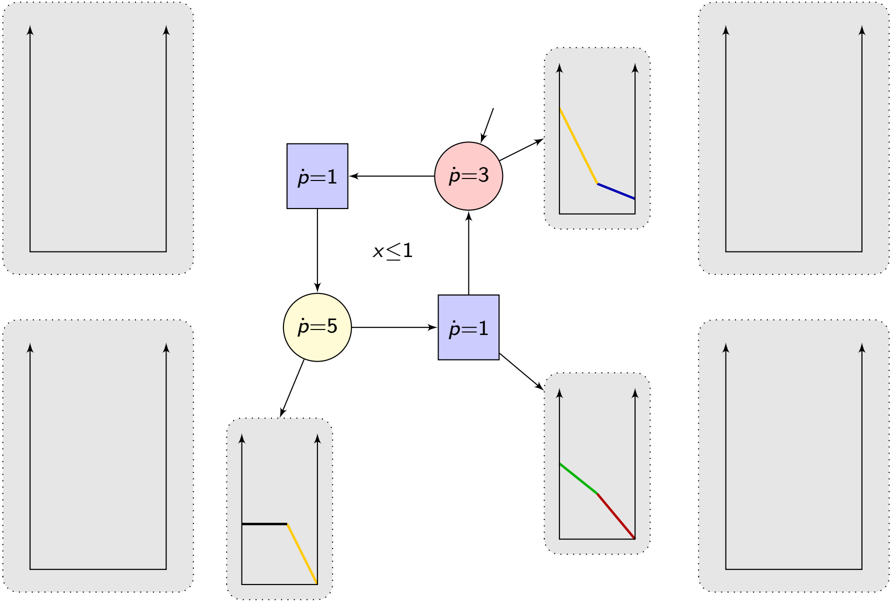

This procedure terminates because fragments having slope strictly less than $c_{\min}$ are fragments of outside functions.

# Outline of the talk

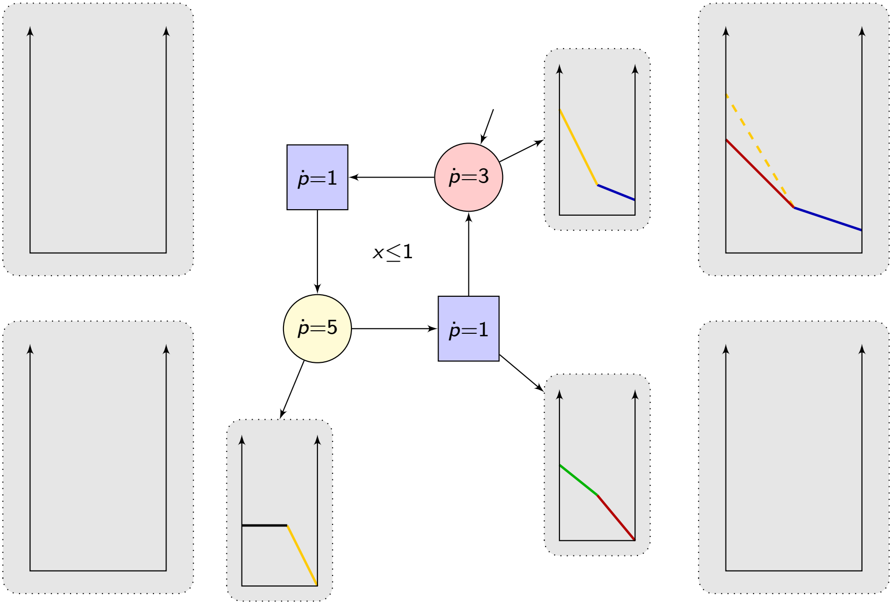
# Iterative pseudo-algorithm of [BCFL04]



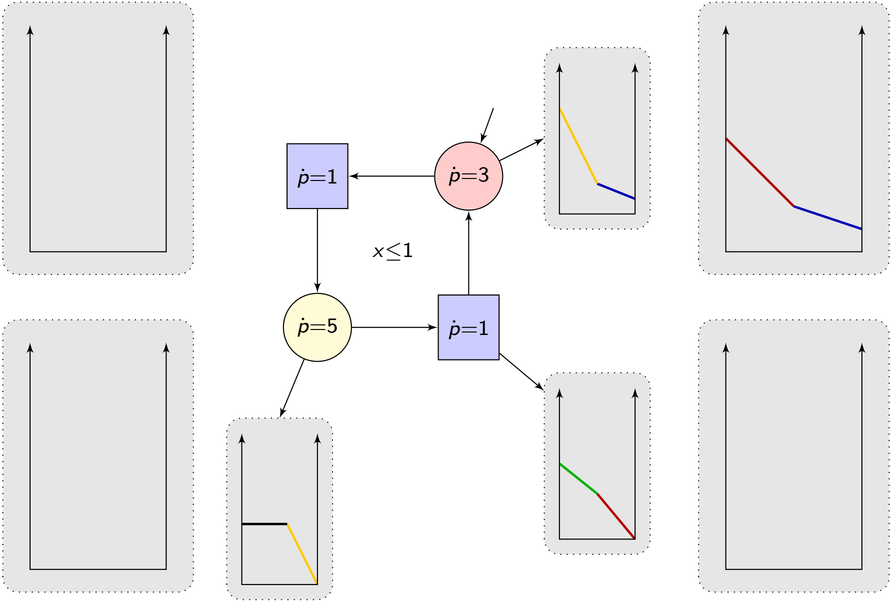
# Iterative pseudo-algorithm of [BCFL04]



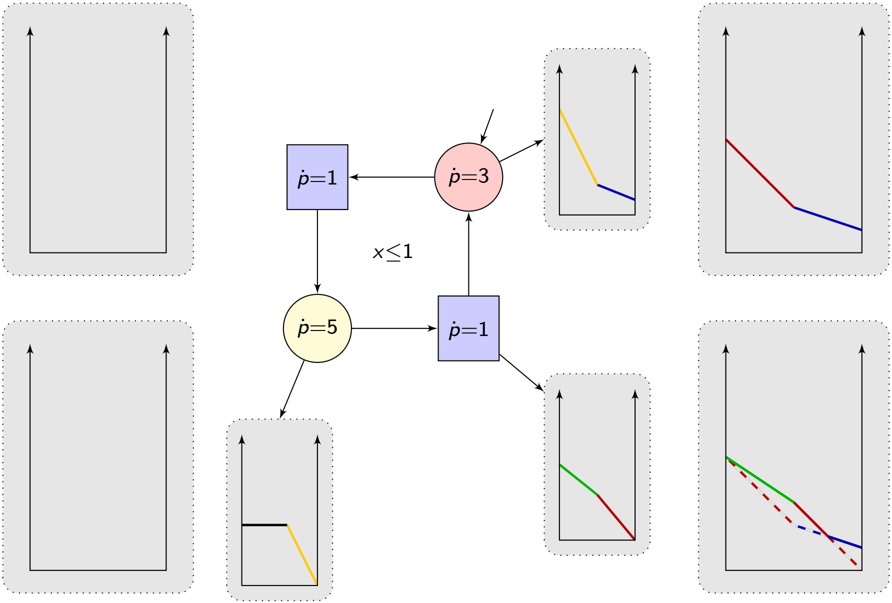
# Iterative pseudo-algorithm of [BCFL04]



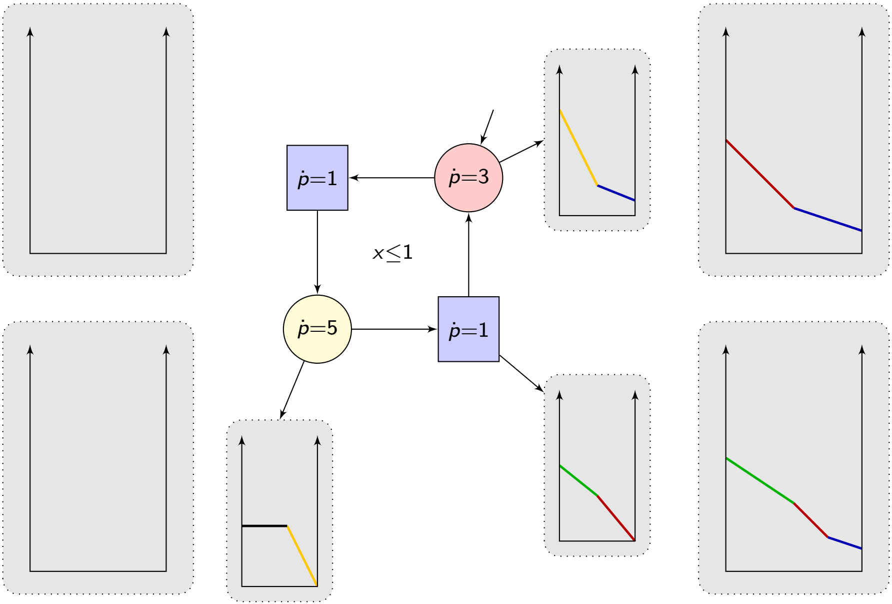
# Iterative pseudo-algorithm of [BCFL04]



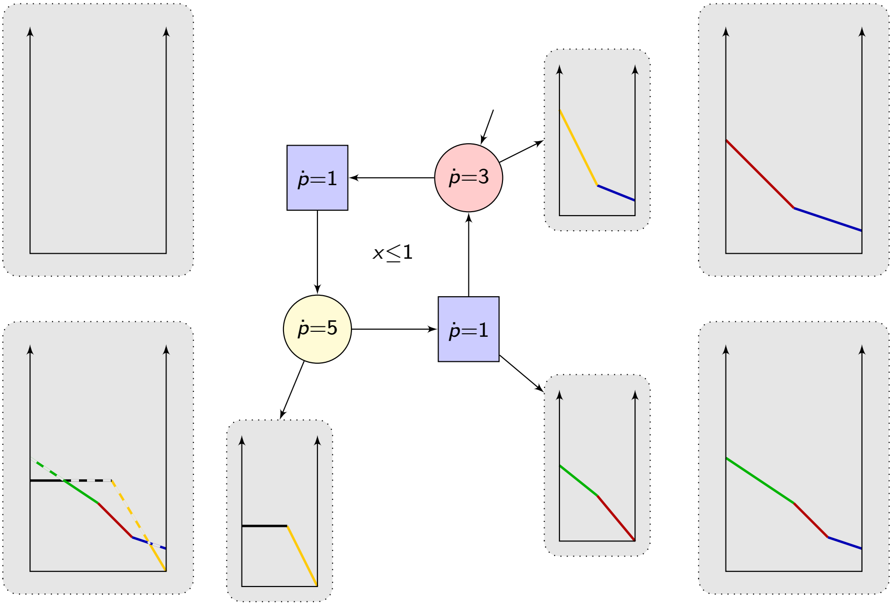
# Iterative pseudo-algorithm of [BCFL04]



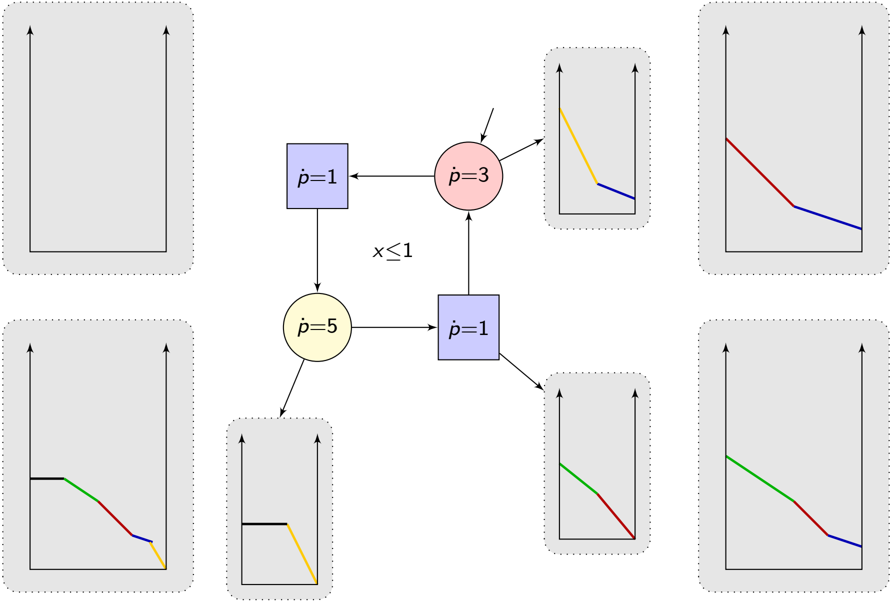
# Iterative pseudo-algorithm of [BCFL04]



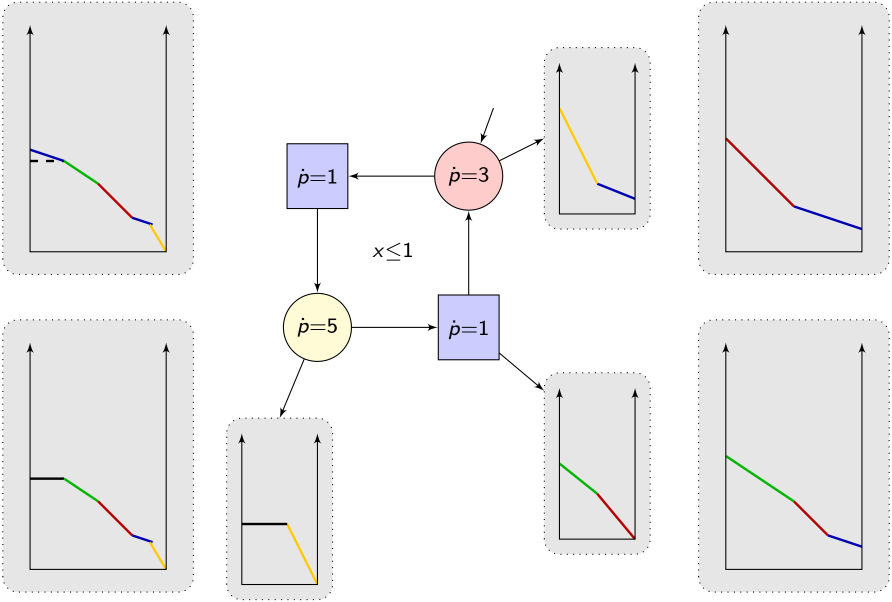
# Iterative pseudo-algorithm of [BCFL04]



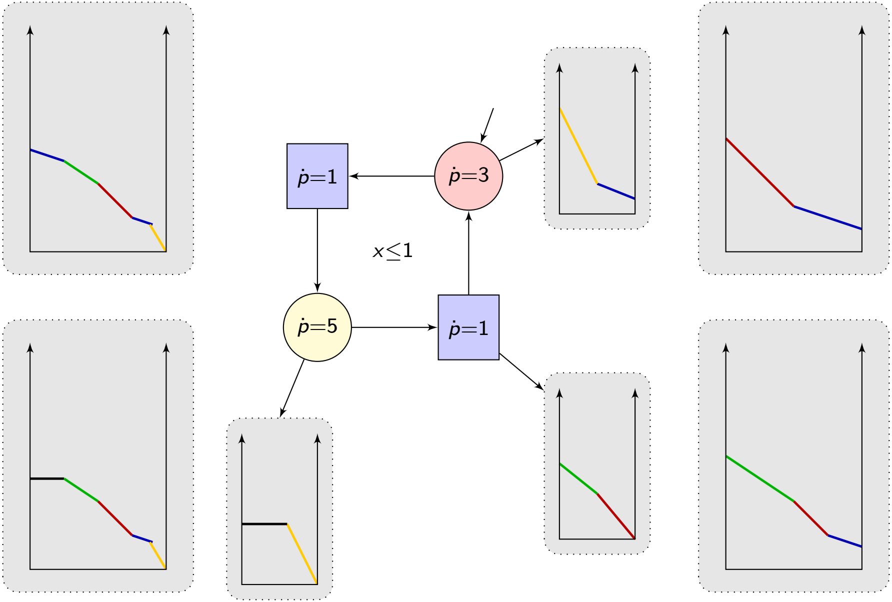
# Iterative pseudo-algorithm of [BCFL04]



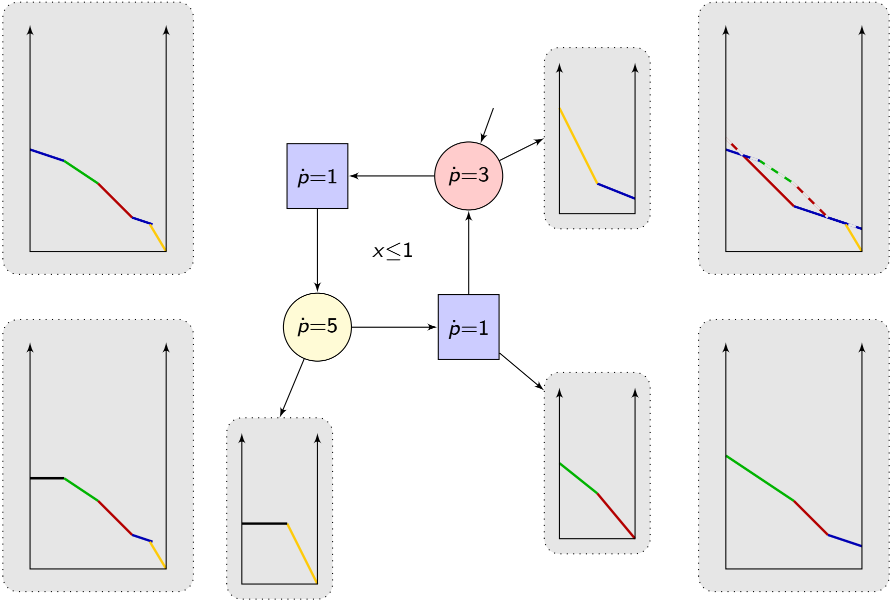
# Iterative pseudo-algorithm of [BCFL04]



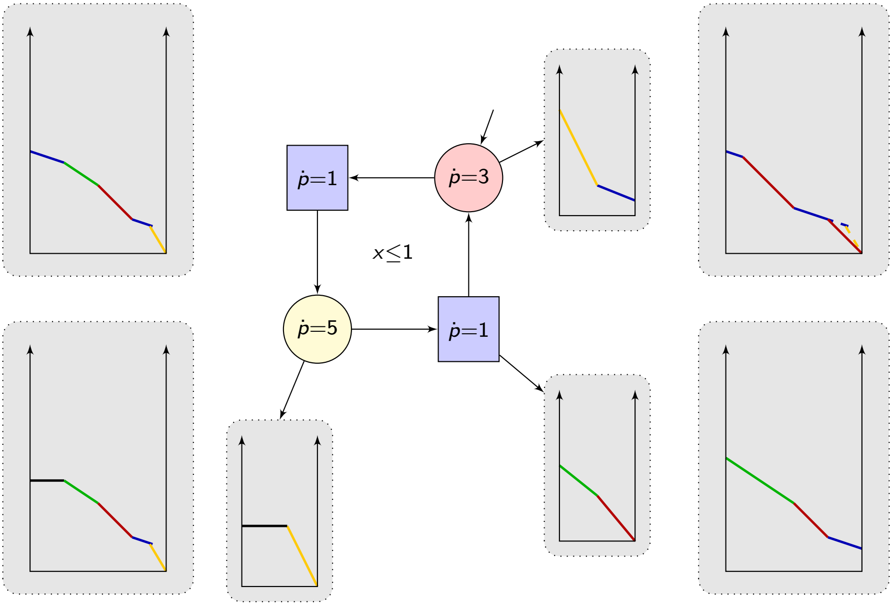
# Iterative pseudo-algorithm of [BCFL04]



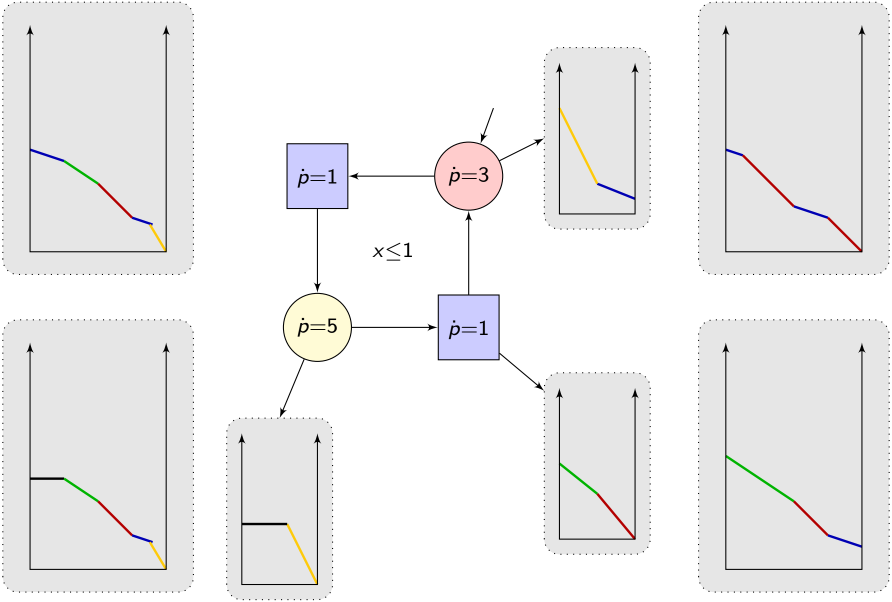
# Iterative pseudo-algorithm of [BCFL04]



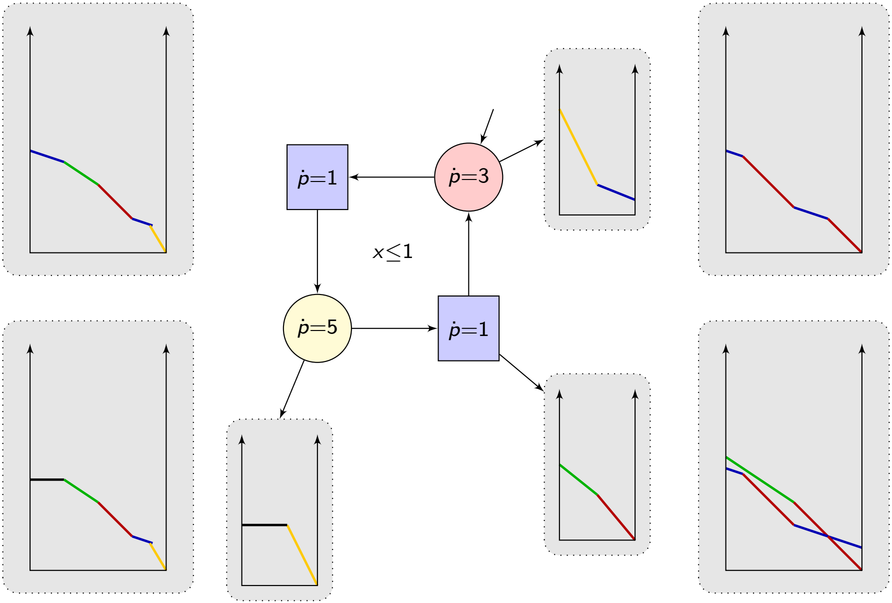
# Iterative pseudo-algorithm of [BCFL04]



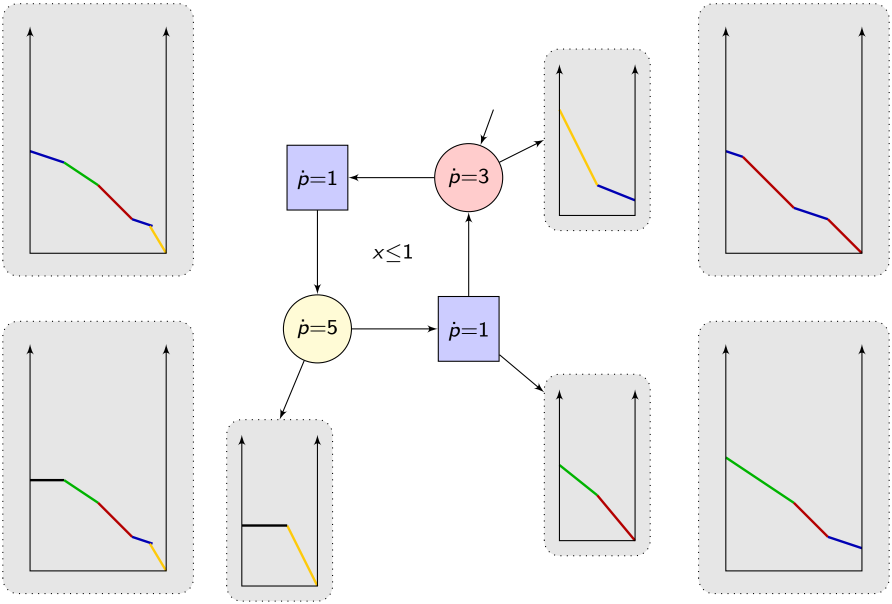
# Iterative pseudo-algorithm of [BCFL04]



# Iterative pseudo-algorithm of [BCFL04]



# Iterative pseudo-algorithm of [BCFL04]



# Iterative pseudo-algorithm of [BCFL04]

**Theorem**

*This algorithm terminates on 1PTGAs.*

# Iterative pseudo-algorithm of [BCFL04]

**Theorem**

*This algorithm terminates on 1PTGAs.*

*Proof.*

- The cost functions computed at round $i$ represent the cost of winning in at most $i$ steps.
- Since there exists $N \in \mathbb{Z}^+$ s.t., for any $\varepsilon > 0$, there exists an $(\varepsilon, N)$-acceptable strategy, we know that there exists $\varepsilon$-optimal strategies that are guaranteed to win in at most $N \times |Q|$ steps.

# Outline of the talk

# Conclusion and Perspectives

- Summary of our works:
  - Adding costs to timed automata provides a natural way for modeling resource consumption.

  - unfortunately, costs are expensive!

  ⤳ Undecidable for three-clock automata;
  ⤳ Complex algorithms for one-clock automata;
  ⤳ Convergence of the pseudo-algorithm of [BCFL04].

- Perspectives:
  - Complexity gap: our algorithm runs in 3EXPTIME, while our best lower bound is PTIME;
  - What happens in two-clock Priced Timed Automata?
  - Priced ATL model-checking: mixing games and WCTL;
  - Multi-constrained objectives.