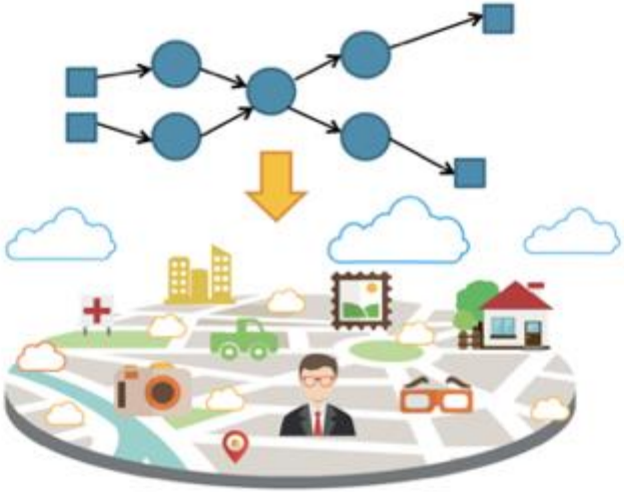
Towards Concurrent Reconfigurations

Farah Ait Salah,
Frédéric Desprez, Laurent Lefevre, Christian Perez
Avalon/Corse

Contexte and Motivation

Challenges

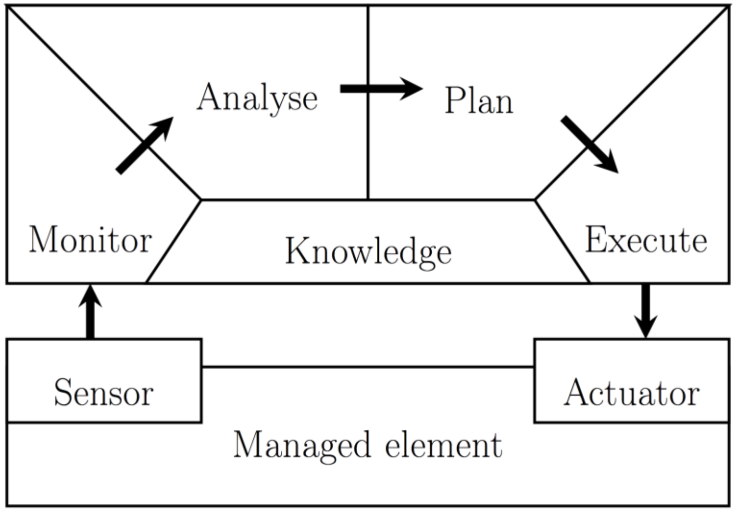
- **Service deployment** in geo-distributed and large scale infrastructure
  - like Fog environment
- Adapt service management to frequent changes of the systems
- Define a deployment plan able to handle these changes
- **Automatic provisioning, deployment and reconfiguration** of applications



MAPE-K Loop

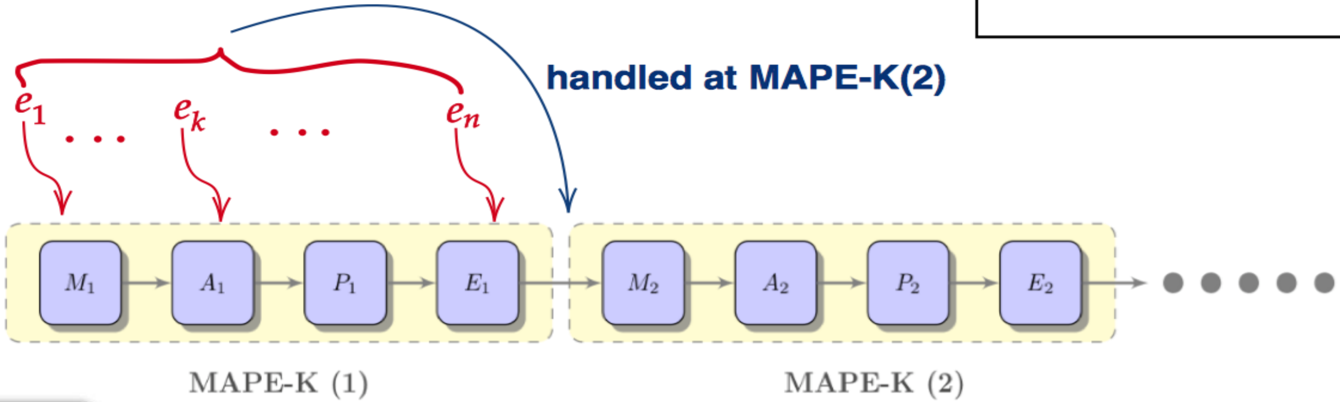
Reconfiguration

- Usually based on the use of the MAPE-K autonomic loop
  - Monitoring, Analyzing, Planning, Executing, and Knowledge phases



Limits

- Sequential loop phases (steps) in MAPE-K
- How to deal with
  - Long plan calculation for optimal placement?
  - Long execution (VMI transfer, etc.)?



Towards Concurrent MAPE-K Loops

Goal

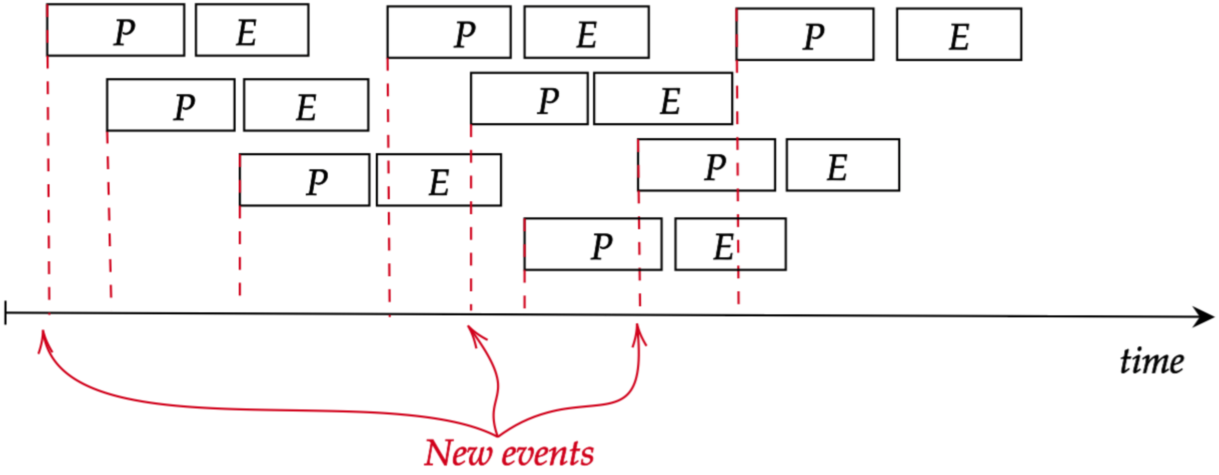
- Provide responsive and scalable solutions for autonomic service placement and reconfiguration

Approach

- Revisit the sequentiality of the MAPE-K loop
    - Introduce concurrent planning and execution steps

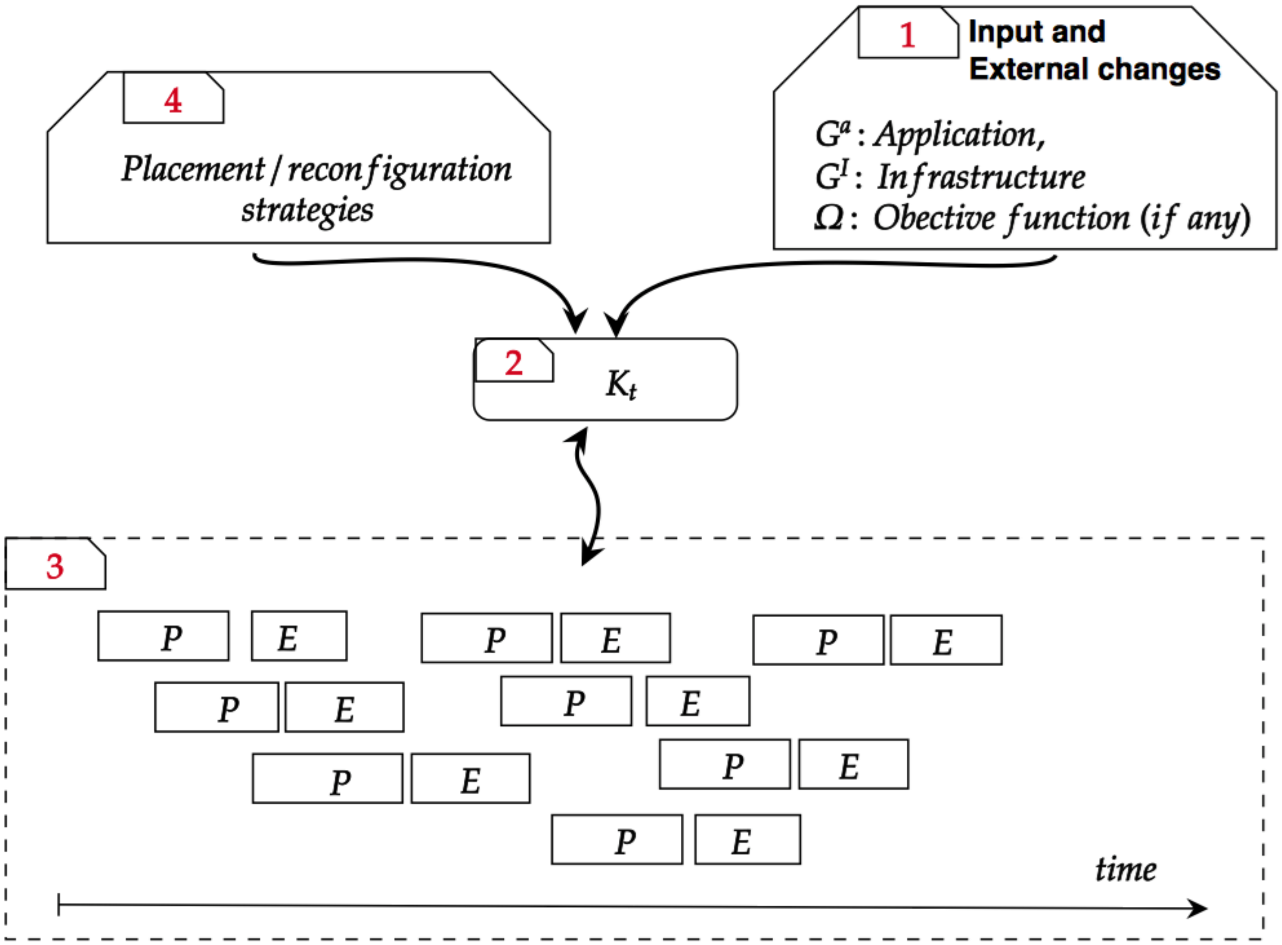
Many challenges

- Feasibility?
    - Concurrent planning?
    - Concurrent execution?
- Benefits/drawbacks?



Modeling Concurrent Planning and Execution

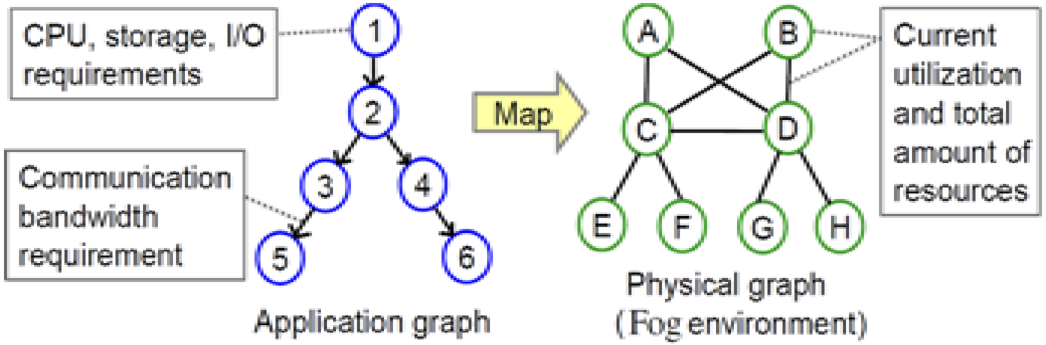
A model to describe the relationships between P, E, and K



Modeling Concurrent Planning and Executor 1: Inputs & External Changes

Infrastructure

- A directed graph $G^I = <V, E>$
- V: set of nodes (server)
  - CPU and RAM capacities
- E: set of edges (connections)
  - Latency and bandwidth capacity



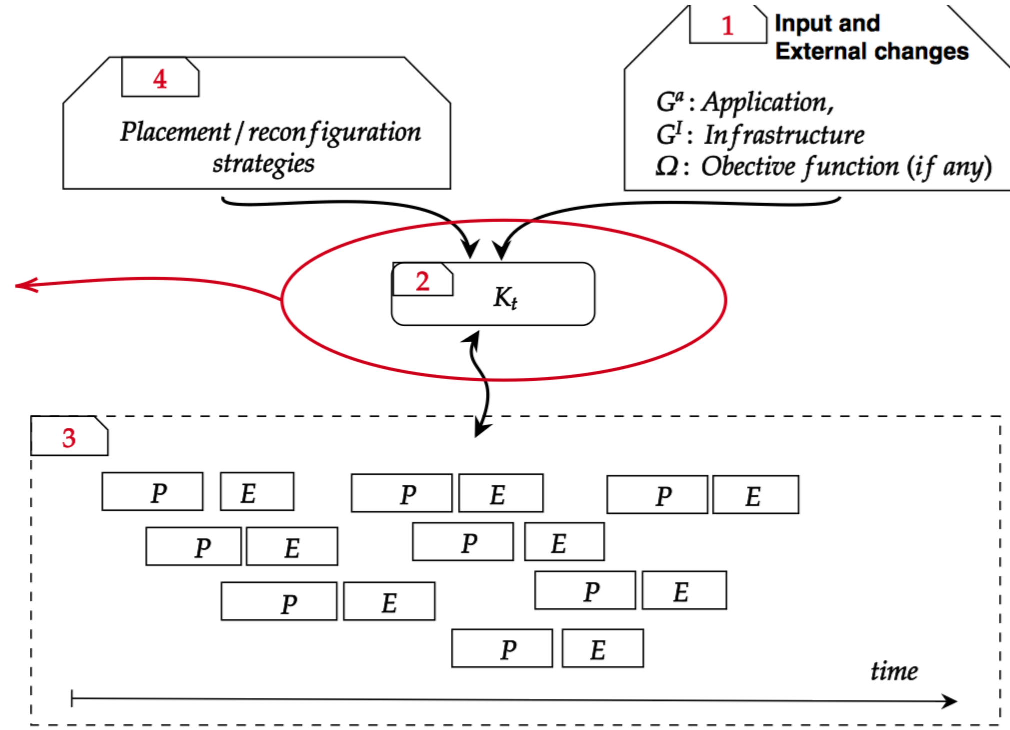
Application

- An application G^a is an ordered set of components
- A components requires CPU/RAM to work, can send/receive data (bandwidth, latency)
- Some components are fixed (ex., camera)
- Component lifecycle: undeployed, deploying, deployed, migrating, (undeploying)

External changes

- Diff={Add, Remove, Update}
  - on the infrastructure graph
  - on the application graph
  - on the objective function (if any)

Modeling Concurrent Planning and Executor 2: The Knowledge Model



The Knowledge K

- $K_t = \{G_t{}^I, G_t{}^a, M_t, LC_t, \Omega_t, *\}$

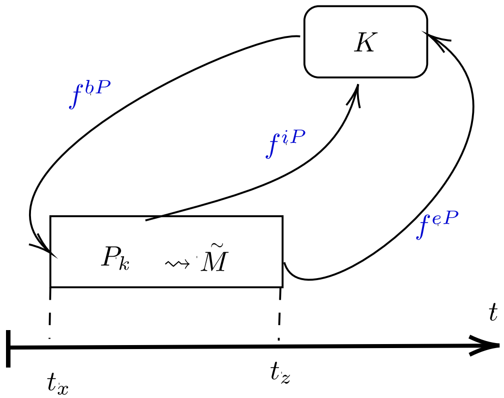
where

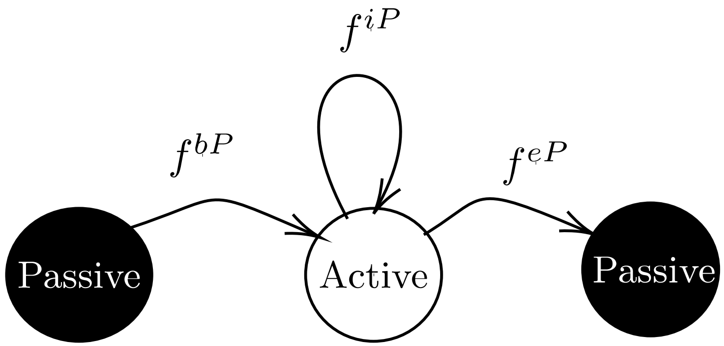
- M_t: component mapping at time t
- LC_t: the life-cycle states of components at time t
- Ω_t: the objective function at time t
- $*$: any other information that can be interesting

Modeling Concurrent Planning and Executor 3: Inputs and Outputs of each Operation

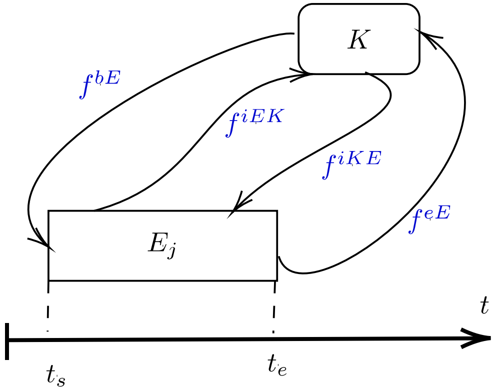
Modeling the functions related to operation P and E

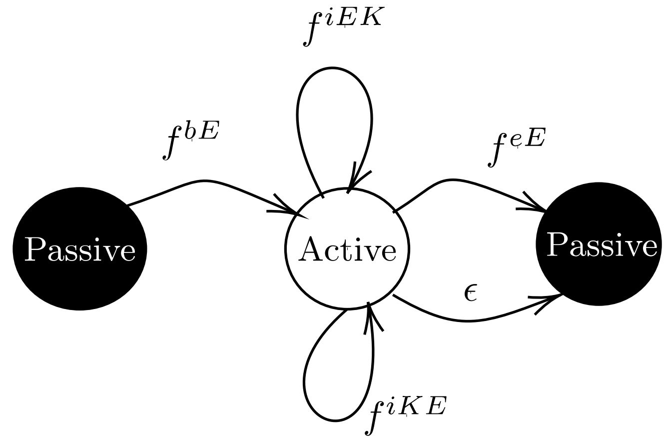
- Functions related to operation P





- Functions related to operation E

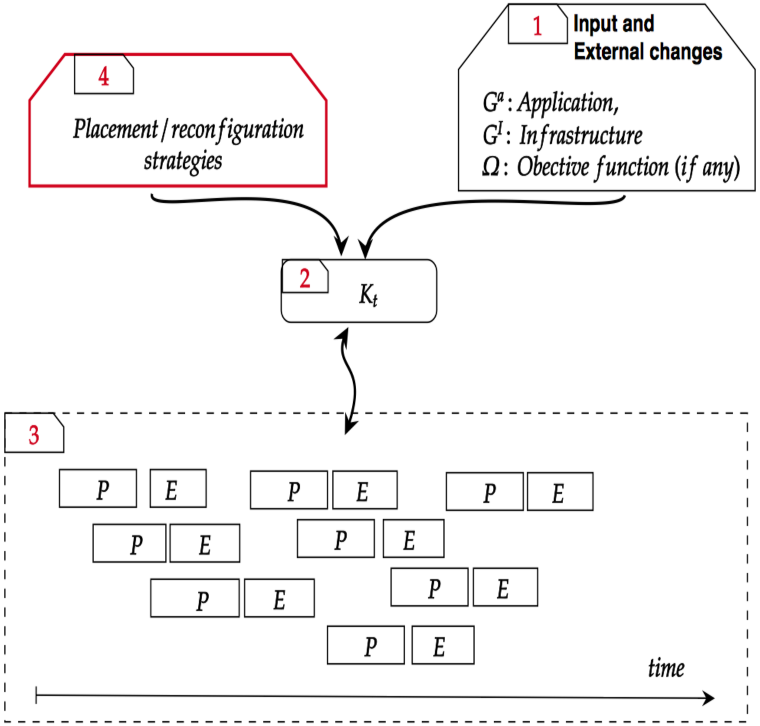




Modeling Concurrent Planning and Executor 4: Global Strategy

Challenge

- Provide an reconfiguration strategy on how to manage concurrent P and E
    - How many P? E?
    - How to merge results from several P?
    - How to manage concurrent execution?



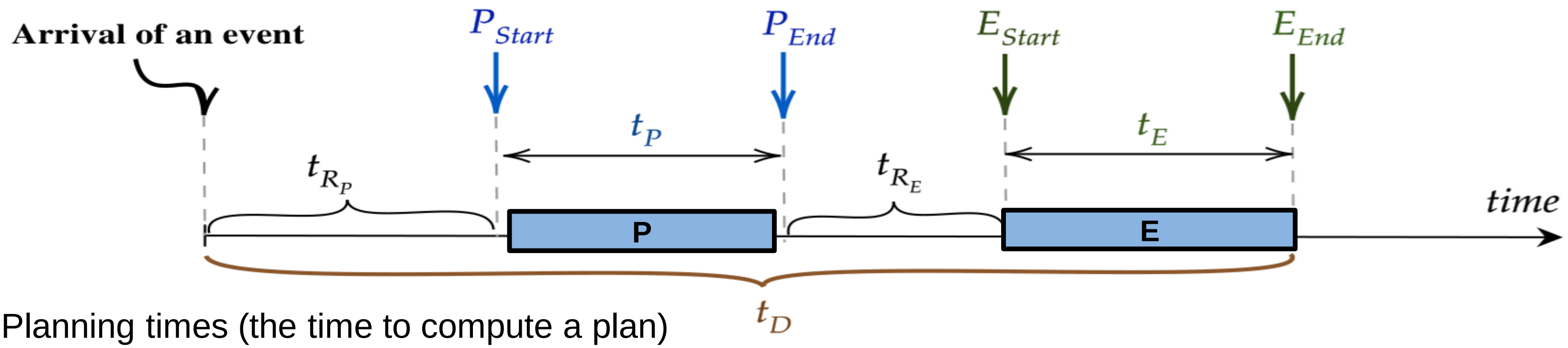
Our methodology

- Define basic and advances strategies
- Study their advantages and disadvantages
- Propose a set of strategy according to the considered use-cases and needs

Metrics

Considered metrics

- t_{R_E}: Responsiveness time for the planning phase (i.e., $t_{R_E} = t_{Arr}$- P_{Start})
- t_I: downtime (i.e., $\sum_t 1_{\{state(comp)\neq\ Dd\}}$)



where

- t_P : Planning times (the time to compute a plan)
- k_P : number of concurrent planning phases launched
- t_E : Execution times (the time to execute the provided plan)
- k_P : number of concurrent execution phases launched
- Π : event period (i.e., a new event happens every Π time unit)
- τ_m : migration rate
- W : window time (i.e., wait at-least (t time) and call operation P (resp. E))

Restricting the Space of Study

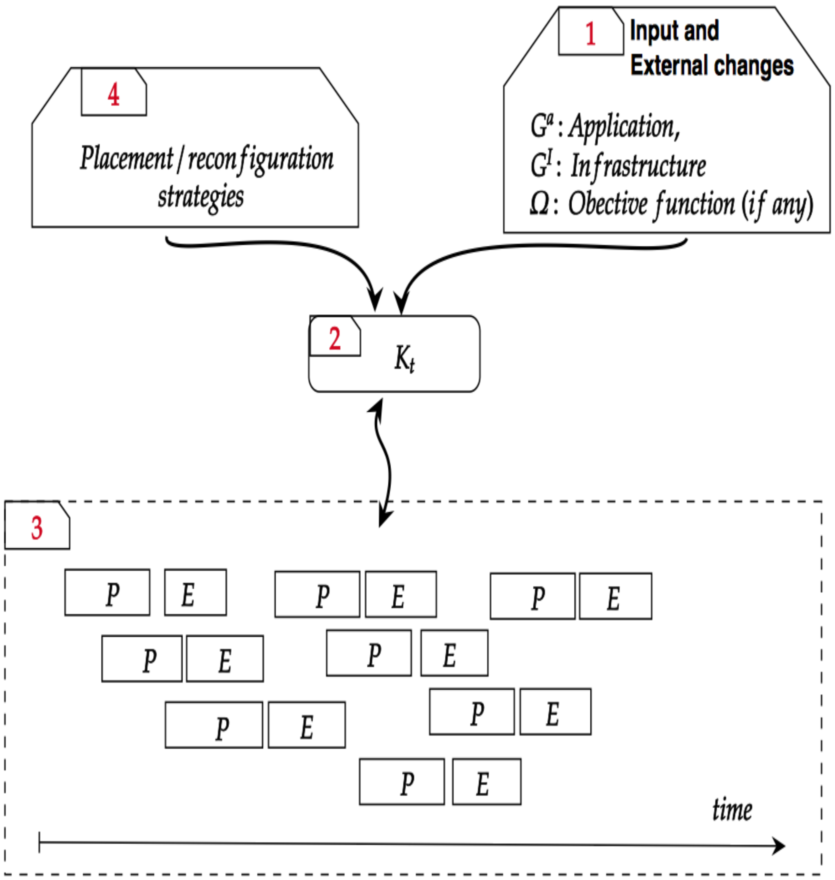
Assumptions

- Event based process
- No interference
- Event arrive at fixed rate (Π: event period)
- All the planning times are homogeneous and are uniform (equal to t_P)
- All the execution times are homogeneous and are uniform (equal to t_E)
- k_E is not bounded (unlimited E)
- $t_{R_E} = 0$ (no delay between a P and its E)
- $\tau_m = 0$ (no migration)

Simulator

Features

- Discrete time simulator
  - Implemented in Python
- The events are supposed to arrive within a given time sequence
  - appearance of a new component
  - removal a component
  - appearance of a host node
  - removal a host node
  - etc.



First Analytical Results

Responsiveness time (t_R)

		MAPE-K loop		PEconc model	
		$\forall\, \Pi > t_G$	$\forall\, \Pi \leq t_G$	$\forall\, \Pi$, and $\forall\, k_P \geq \lceil \frac{t_P}{\Pi} \rceil$	$\forall\, \Pi$, and $\forall\, k_P < \lceil \frac{t_P}{\Pi} \rceil$
t_R	Min	0	0	0	0
	Max	0	t_G	0	t_P
	Mean	0	$\frac{t_G}{2}$	0	$\frac{t_P}{2}$

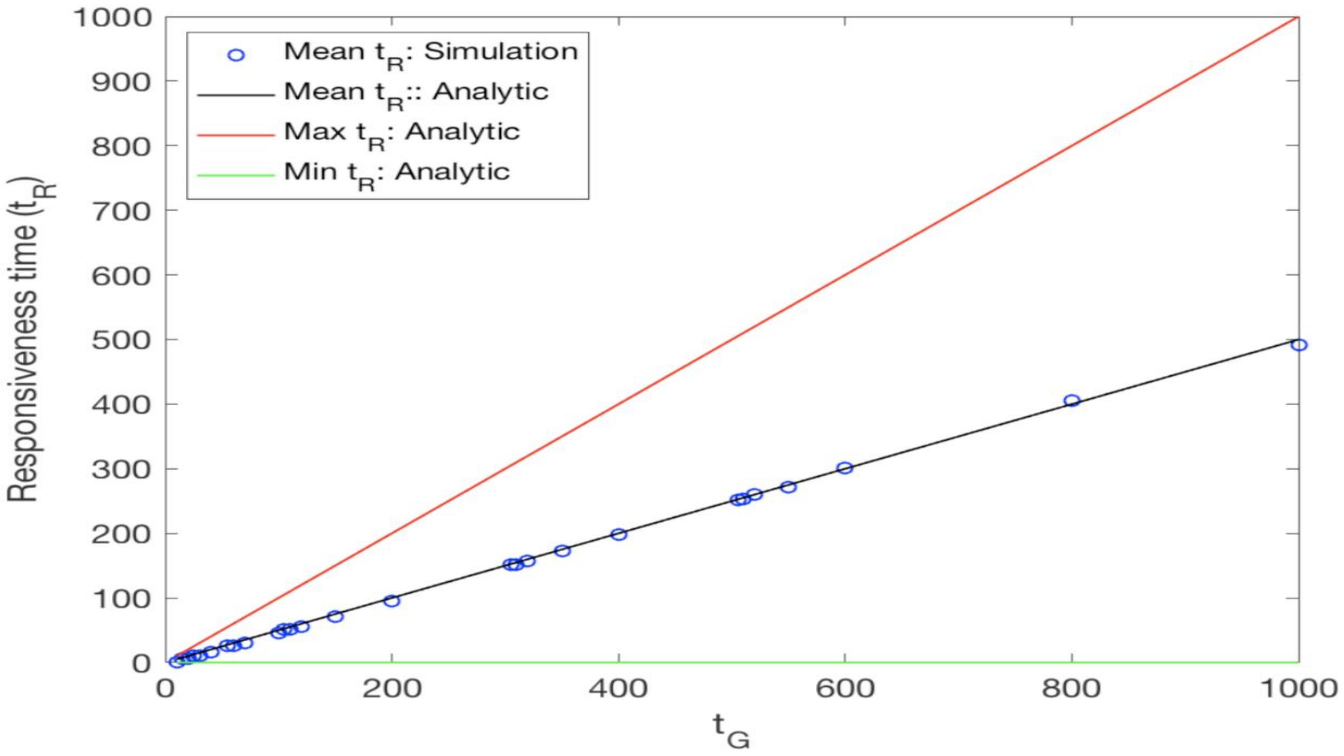
- $t_G = t_P + t_{R_E} + t_E$: time of single plan and execute cycle
- $\lfloor \frac{t_P}{\Pi} \rfloor$corresponds to the number of events that we have during the period t_P

Downtime

- $t_I = t_R + t_G$, for $\tau_m = 0$

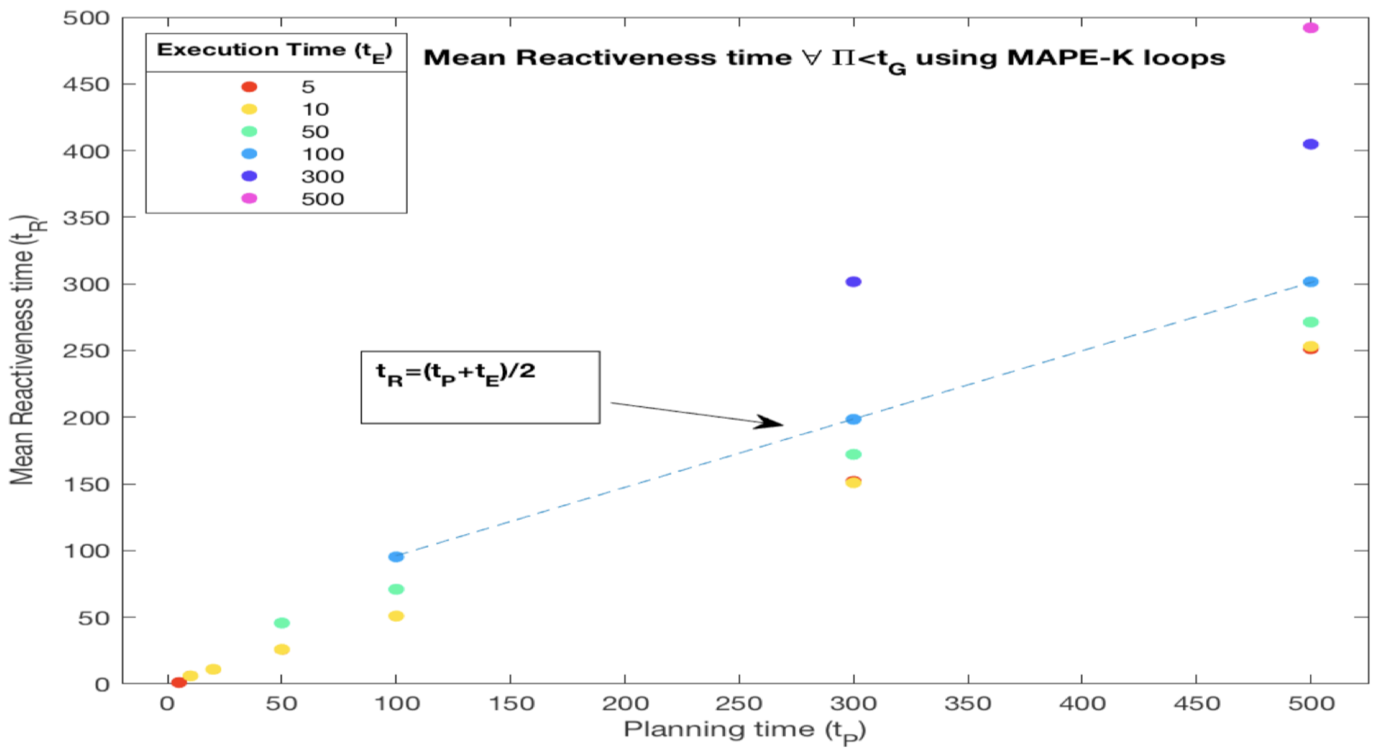
First Results: Simulation vs Analytical

Responsiveness time vs. t_G for reconfiguration system based on MAPE-K loop



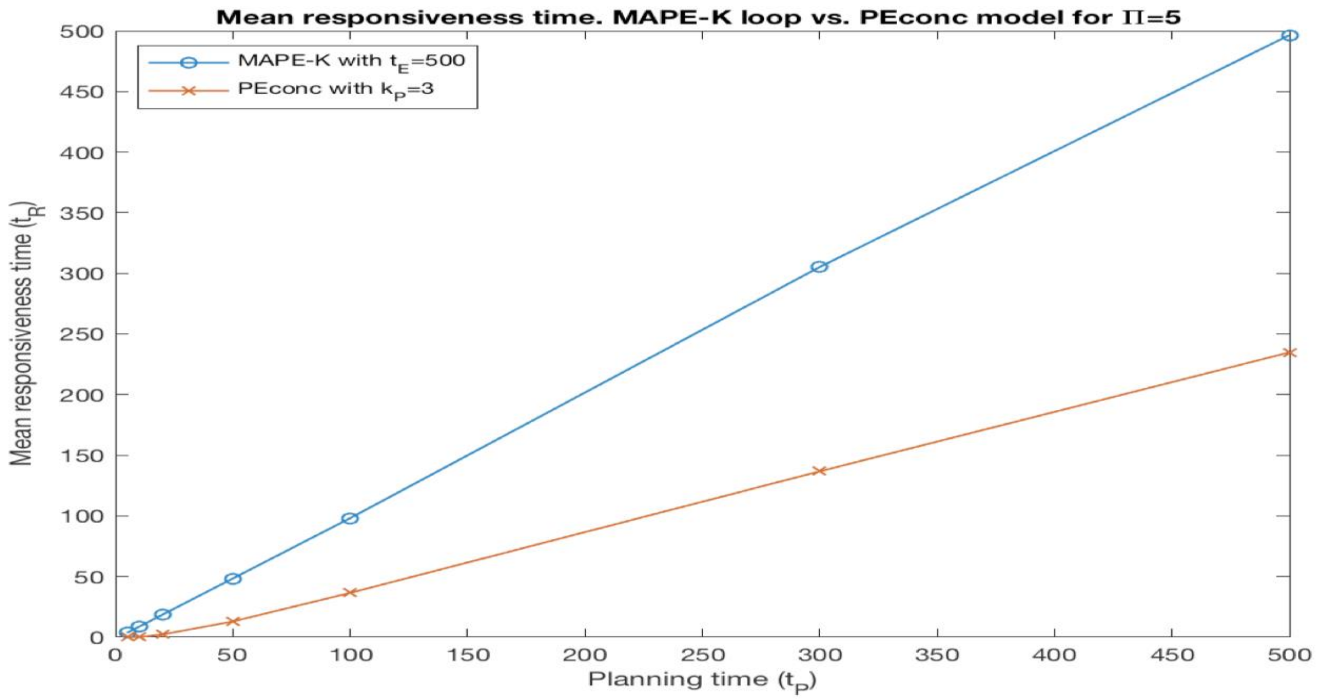
First Results: Simulation

Responsiveness time vs. planning time and execution time for reconfiguration system based on MAPE-K loop



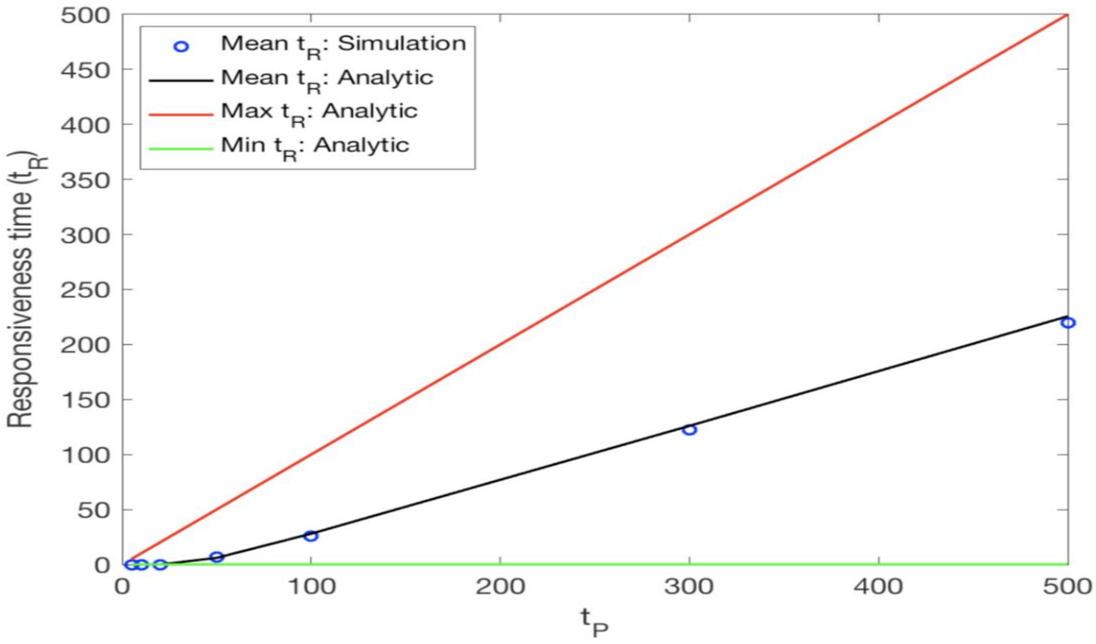
First Results: Simulation

Mean responsiveness time vs. Planning time MAPE-K vs PEconc



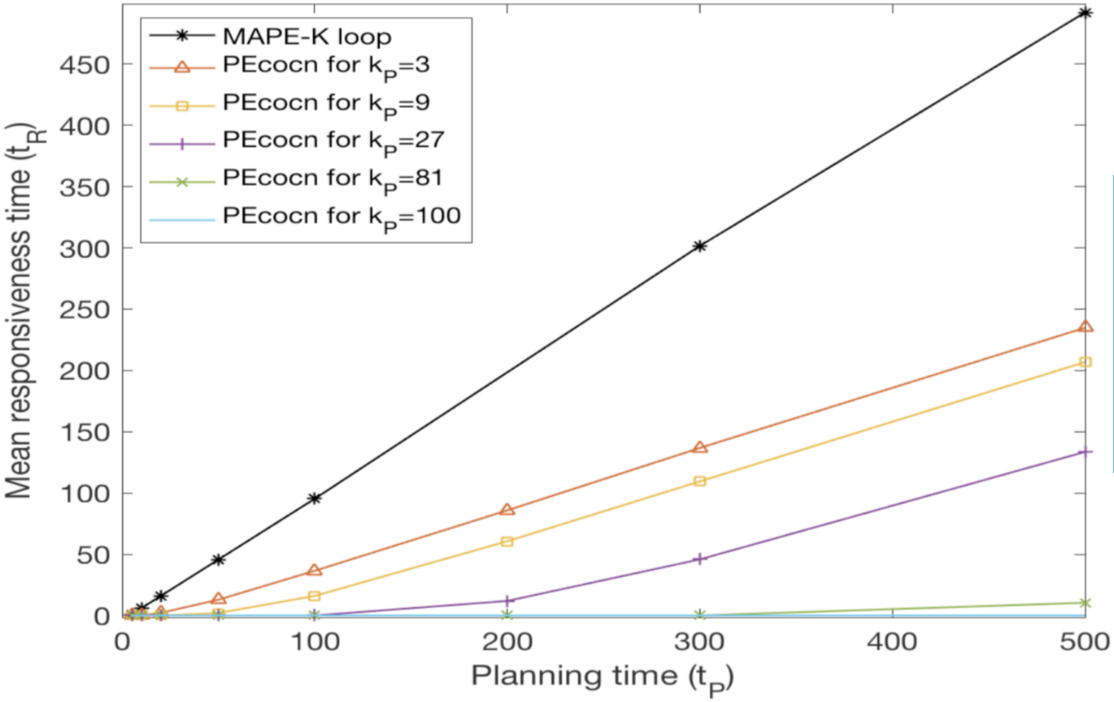
First Results: Simulation vs Analytical

Mean responsiveness time vs. Planning time for PEconc



First Results: Simulation

Responsiveness time vs. Planning time and k_P for reconfiguration system based on PEconc model



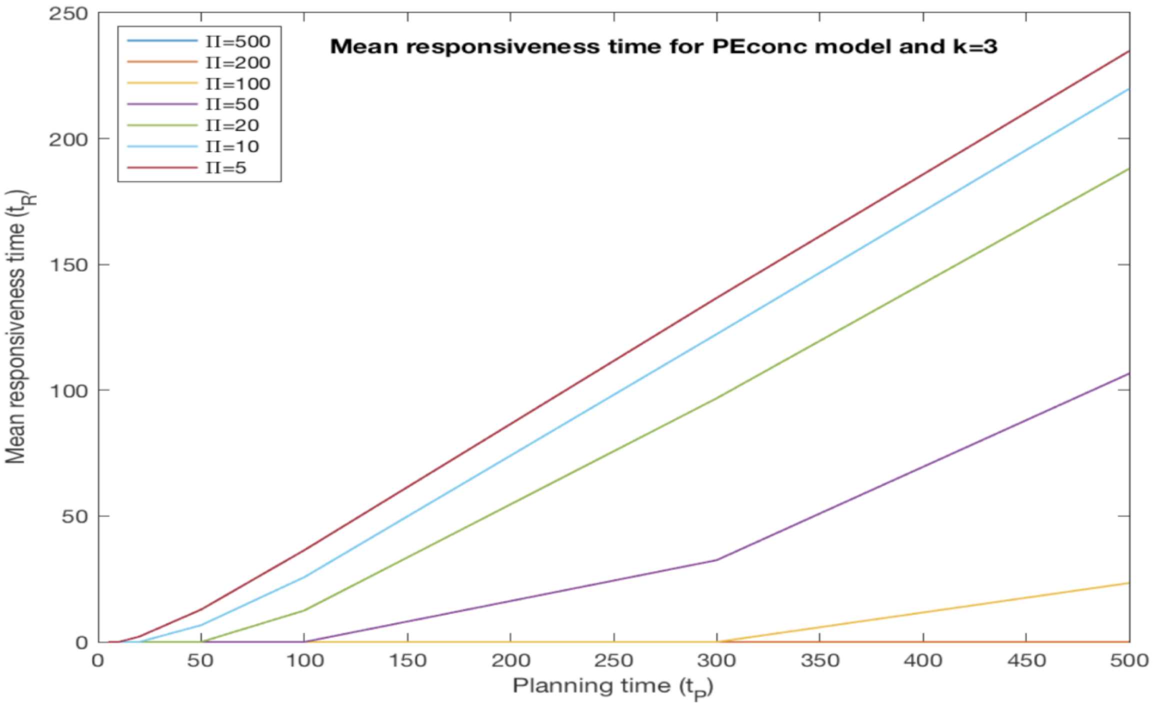
$$Mean(t_{R,PEconc}) \leq \frac{t_P}{2}$$

$$\ll \qquad \forall k_P > 1$$

$$Mean(t_{R,MAPE}) \approx \frac{t_G}{2}$$

First Results: Simulation

Responsiveness time vs. Planning time and Π for reconfiguration system based on PEconc model



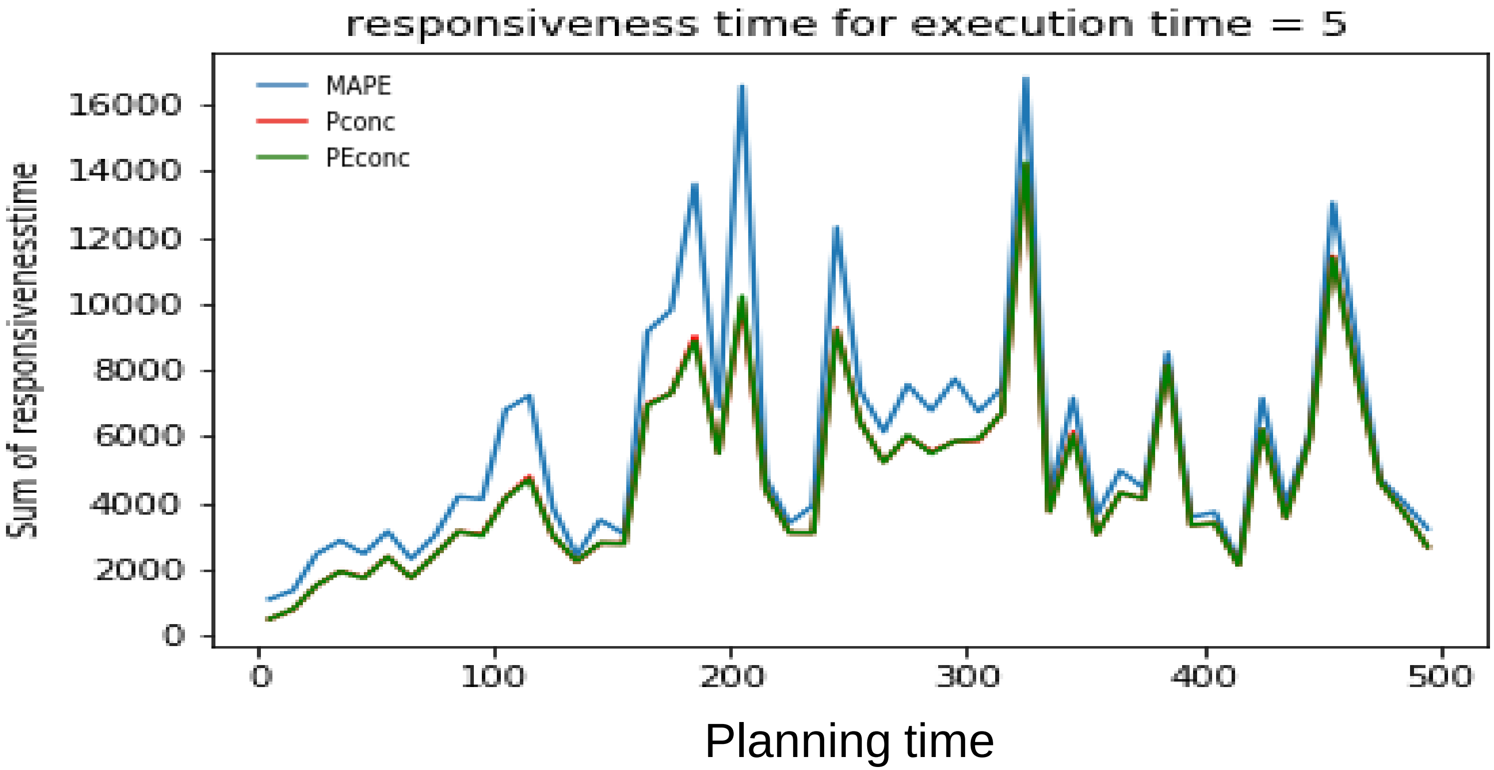
$$Mean(t_{R,PEconc}) \leq \frac{t_P}{2}$$

$$\ll \qquad \forall \Pi$$

$$Mean(t_{R,MAPE}) \approx \frac{t_G}{2}$$

First Results: Simulation

Results based on random *Events* in { *AddComp*, *RemoveComp*, *AddHost*, *RemoveHost* }



Conclusion

Challenges

- How to deal with potential long planning and/or execution step in MAPE-K loop?
- Large scale system (need of decentralized approach?)

Current work

- Study the feasibility, advantages and drawbacks of concurrent planning and execution
  - PEconc model to describe interactions
  - Analytical studies for simple cases, to obtain bounds and trends
  - Simulator for more real cases
- First results promising

Future work

- Study how to decompose mapping problems and related strategies
  - Reuse existing algorithms
  - Impact on performance and cost (resource usage)
- Improve Concerto to become a concurrent execution model and engine
- Study the need of decentralized concurrent MAPE-K