Security of distributed Model Predictive Control under False Data injection

Rafael Accácio NOGUEIRA

2022-12-07

https://bit.ly/3g3S6X4

45 minutes !!!!

Good afternoon, thank you all for being here.I'm Rafael Accácio and I'm going to present my work on the security of distributed model predictive control under false data injection.

Context

❝ Necessity is the mother of invention ❞

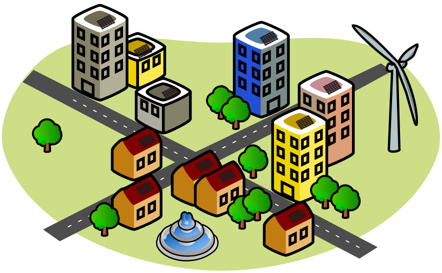
CentraleSupélec

Context

“Necessity is the mother of invention”

Context

"Necessity is the mother of invention"



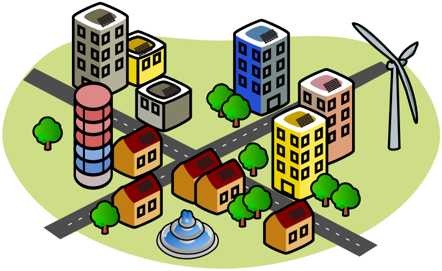
- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

  (include your problem here)

CentraleSupélec

Context

"Necessity is the mother of invention"



- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management
  (include your problem here)

CentraleSupélec

Context

"Necessity is the mother of invention"

- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

(include your problem here)

CentraleSupélec

Context

“Necessity is the mother of invention”

- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

(include your problem here)

Context

“Necessity is the mother of invention”

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

Context

"Necessity is the mother of invention"

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Context

“Necessity is the mother of invention”

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Context

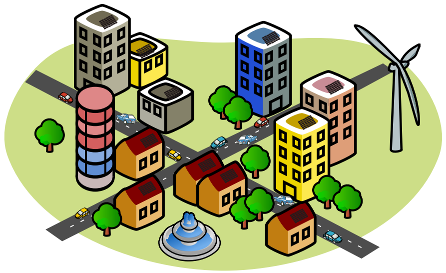
❝Necessity is the mother of invention❞

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Context

"Necessity is the mother of invention"

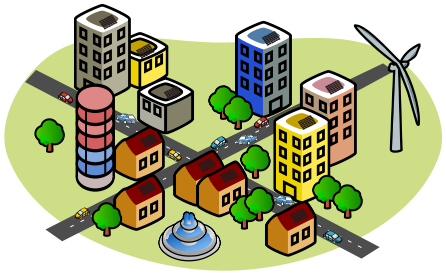


- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Context

"Necessity is the mother of invention"



- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

Context

❝Necessity is the mother of invention❞

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Context

"Necessity is the mother of invention"

- Multiple systems interacting
- Coupled by constraints
  - Technical/ Comfort
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
- Solution → MPC

CentraleSupélec

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect

For those who are not familiar with mpc. Mpc is the model based predictive controller.

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

CentraleSupélec

The objective is to find the best control sequence using predictions based on a model.

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect

When we say best,

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

we mean optimal.

CentraleSupélec

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned} \boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leq 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1, \dots, N\} \\ &\forall i \in \{1, \dots, m\} \\ &\forall j \in \{1, \dots, p\} \end{aligned}
\end{aligned}
$$

CentraleSupélec

So we need to solve an optimization problem.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
&\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} && J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
&\text{subject to} && \left.\begin{array}{l} \boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0 \end{array}\right\} \begin{array}{l} \forall \xi \in \{1, \dots, N\} \\ \forall i \in \{1, \dots, m\} \\ \forall j \in \{1, \dots, p\} \end{array}
\end{aligned}
$$

CentraleSupélec

And we have the control sequence of u as the decision variable.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k])$$

$$\text{subject to} \quad \left.\begin{array}{l} \boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leq 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0 \end{array}\right\} \begin{array}{l} \forall \xi \in \{1, \dots, N\} \\ \forall i \in \{1, \dots, m\} \\ \forall j \in \{1, \dots, p\} \end{array}$$

CentraleSupélec

which is calculated for a horizon N

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
&\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} && J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
&\text{subject to} && \left.\begin{aligned}
&\boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\
&g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leq 0 \\
&h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0
\end{aligned}\right\} \begin{aligned}
&\forall \xi \in \{1, \dots, N\} \\
&\forall i \in \{1, \dots, m\} \\
&\forall j \in \{1, \dots, p\}
\end{aligned}
\end{aligned}
$$

So, we need an objective function. For example follow a trajectory while minimizing the energy.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned} \boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1, \dots, N\} \\ &\forall i \in \{1, \dots, m\} \\ &\forall j \in \{1, \dots, p\} \end{aligned}
\end{aligned}$$

CentraleSupélec

A model of the system

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
&\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} && J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
&\text{subject to} && \left.\begin{aligned} \boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1, \dots, N\} \\ &\forall i \in \{1, \dots, m\} \\ &\forall j \in \{1, \dots, p\} \end{aligned}
\end{aligned}
$$

CentraleSupélec

with its states

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
&\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} && J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
&\text{subject to} && \left.\begin{aligned} &\boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ &g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leqslant 0 \\ &h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1, \dots, N\} \\ &\forall i \in \{1, \dots, m\} \\ &\forall j \in \{1, \dots, p\} \end{aligned}
\end{aligned}
$$

CentraleSupélec

and inputs

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned}
\boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\
g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\
h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0
\end{aligned}\right\}
\begin{aligned}
&\forall \xi \in \{1, \dots, N\} \\
&\forall i \in \{1, \dots, m\} \\
&\forall j \in \{1, \dots, p\}
\end{aligned}
\end{aligned}
$$

CentraleSupélec

But we can also integrate some constraints, such QoS or technical restrictions

Model-based Predictive Control

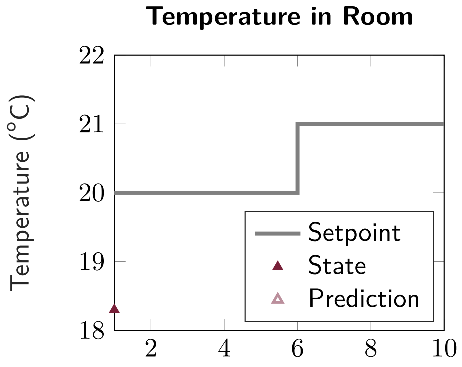
Find optimal control sequence using predictions based on a model.

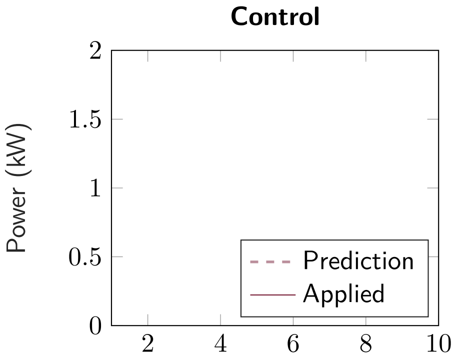
- We need an optimization problem
  - Decision variable is the control sequence (Over horizon N)
  - Objective function to optimize
  - System's Model (states and inputs)
  - Other constraints to respect (QoS, technical restrictions, ...)

$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned}
\boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\
g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\
h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0
\end{aligned}\right\}
\begin{aligned}
&\forall \xi \in \{1, \dots, N\} \\
&\forall i \in \{1, \dots, m\} \\
&\forall j \in \{1, \dots, p\}
\end{aligned}
\end{aligned}
$$

CentraleSupélec

Find optimal control sequence



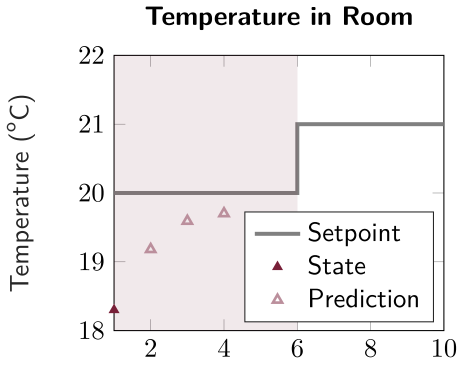


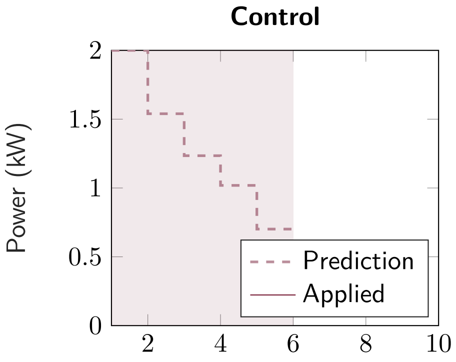
So, for example, if we may have a setpoint to follow

Model Predictive Control

In a nutshell

Find optimal control sequence



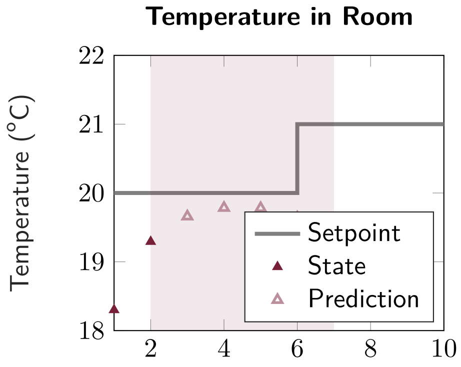


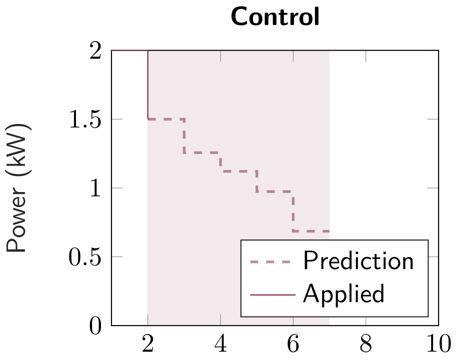
We find an optimal control sequence

Model Predictive Control

In a nutshell

Find optimal control sequence, apply first element





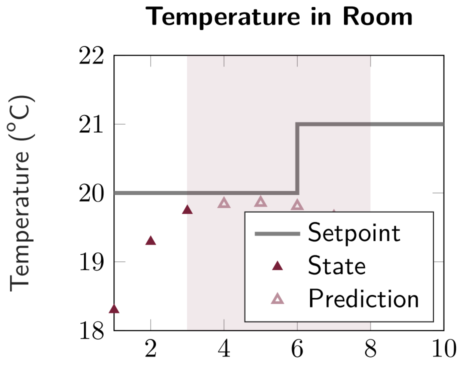
CentraleSupélec

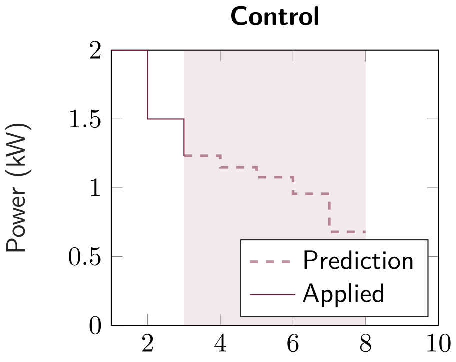
We apply only the first element

Model Predictive Control

In a nutshell

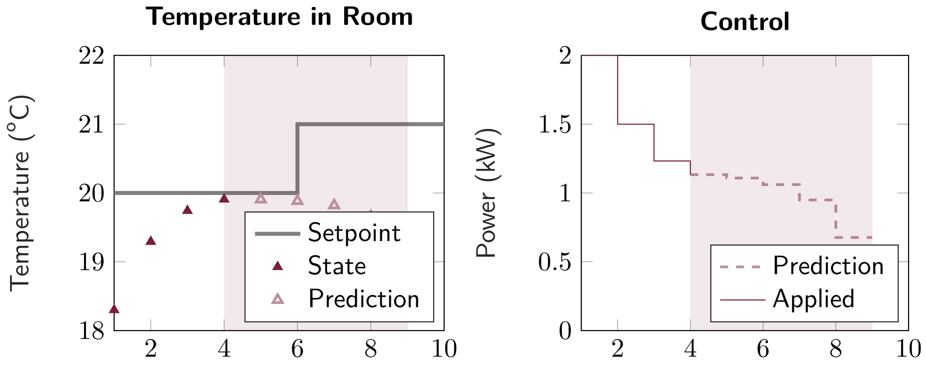
Find optimal control sequence, apply first element, rinse repeat





and then we repeat

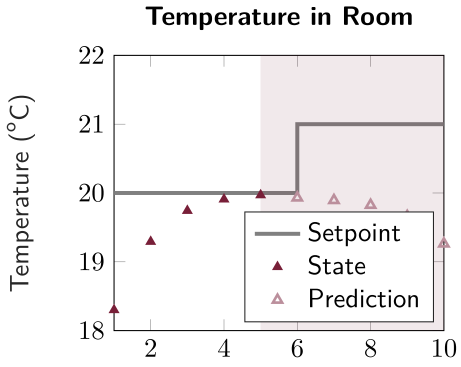
Find optimal control sequence, apply first element, rinse repeat → Receding Horizon

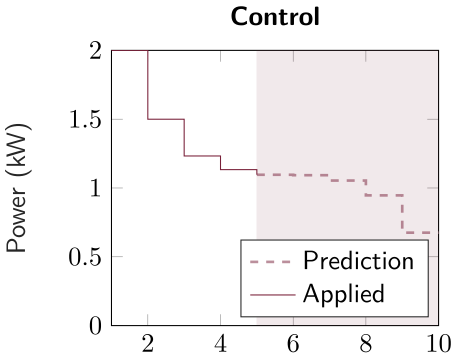


CentraleSupélec

following what we call the receding horizon strategy.However this problem is not always straightforward to solve, for some cases it can be easier.

Find optimal control sequence, apply first element, rinse repeat → Receding Horizon





Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

CentraleSupélec

- Problems
    - Complexity of calculation
    - Topology (Geographical distribution)
    - Flexibility (Add/remove parts)
    - Privacy
- Solution: Divide and Conquer (distributed MPC)
    - Break calculation
    - Make Systems Communicate

Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

CentraleSupélec

Model Predictive Control

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

Nothing is perfect

- Problems
  - Complexity of calculation
  - Topology (Geographical distribution)
  - Flexibility (Add/remove parts)
  - Privacy
- Solution: Divide and Conquer (distributed MPC)
  - Break calculation
  - Make Systems Communicate

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate
  - Many flavors to choose from

CentraleSupélec

However, the solution will depend on the horizon, the number of constraints, and sizes of input and states, increasing the complexity of the calculation

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate
  - Many flavors to choose from

A strategy to alleviate is to distribute the calculation whenever possible. And there are many ways to divide it as the book shows.

It is about communication

- We break the MPC into multiple
- Make them Communicate
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

[1]

CentraleSupélec

Here we opt for a hierarchical strategy where we use multiple MPCs and an agent to coordinate and manage the coupling aspects of the problem.

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

Distributed Model Predictive Control

It is about communication

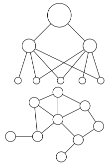
- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

1

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

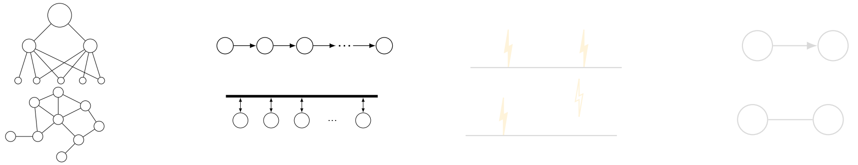


1

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

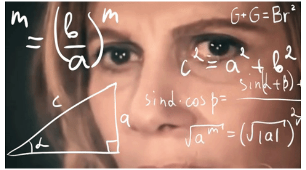


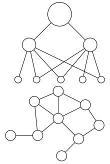
[1]

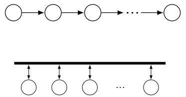
Distributed Model Predictive Control

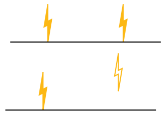
It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...





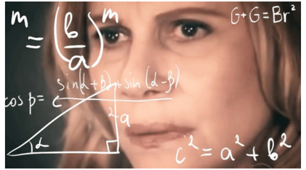


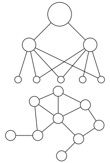


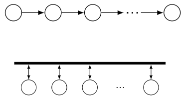
1

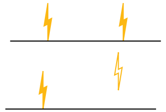
It is about communication

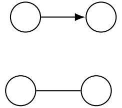
- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...









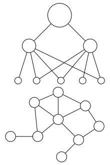


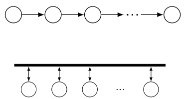
1

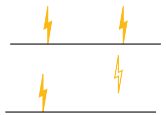
Distributed Model Predictive Control

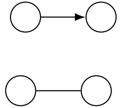
It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...









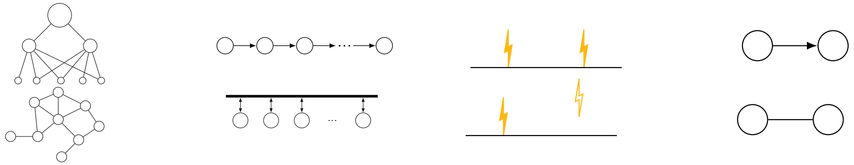
[1] Distributed Model Predictive Control made easy

CentraleSupélec

Distributed Model Predictive Control

It is about communication

- We break the MPC into multiple
- Make them Communicate , But how?
  - Many flavors to choose from[1]
    - Hierarchical/Anarchical
    - Sequential/Parallel
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...

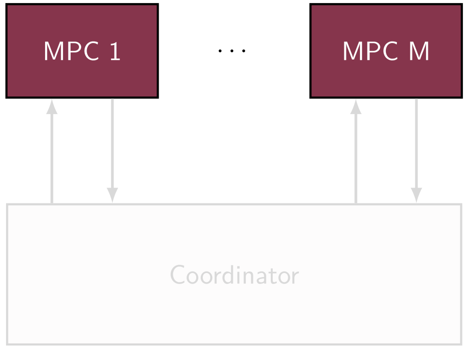


[1] Distributed Model Predictive Control made easy

CentraleSupélec

Distributed Model Predictive Control

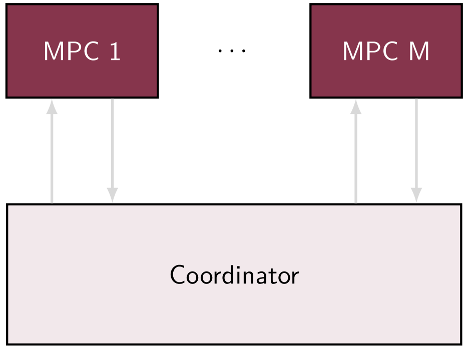
Optimization Frameworks



- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

CentraleSupélec

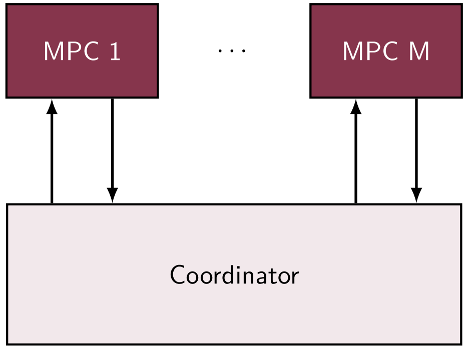


- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

Distributed Model Predictive Control

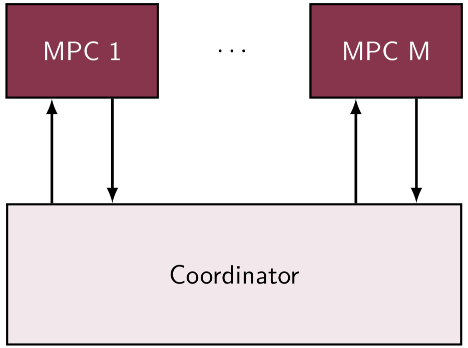
Optimization Frameworks



- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

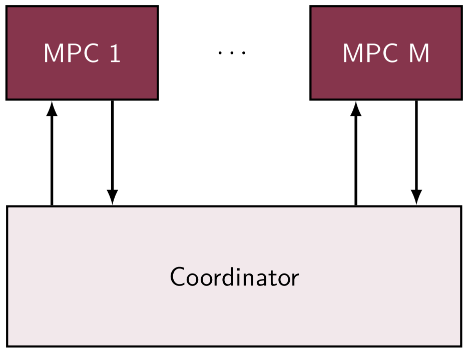
Optimization Frameworks



- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

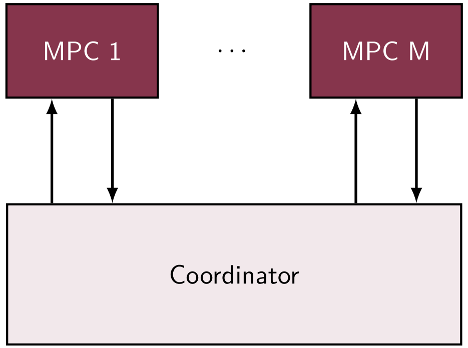
Until Convergence

CentraleSupélec



- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

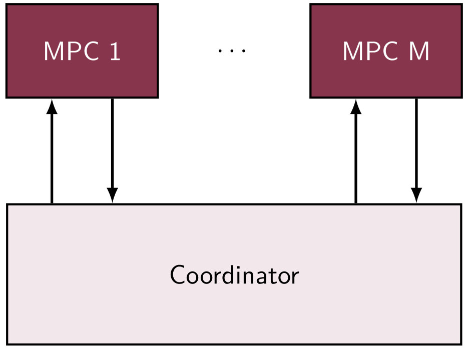


- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

Distributed Model Predictive Control

Optimization Frameworks



- Coordinator → Hierarchical
- Bidirectional
- No delay → Synchronous
- Agents solve local problems
- Variables are updated

Until Convergence

CentraleSupélec

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!

CentraleSupélec

Distributed Model Predictive Control

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!

CentraleSupélec

Distributed Model Predictive Control

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!

CentraleSupélec

Distributed Model Predictive Control

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!

CentraleSupélec

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!

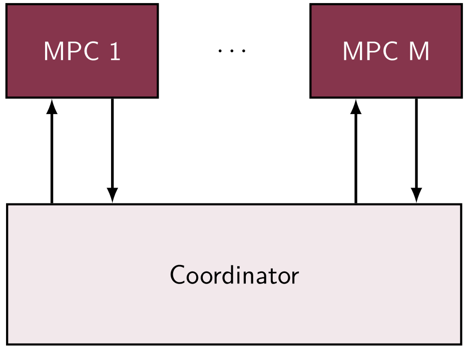
CentraleSupélec

Distributed Model Predictive Control

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

- How can an agent attack?
- What are the consequences of an attack?
- Can we mitigate the effects?

Let's have a preview!



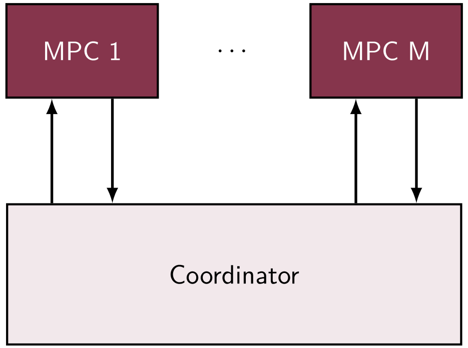
- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

CentraleSupélec

How can a non-cooperative agent attack?

Literature

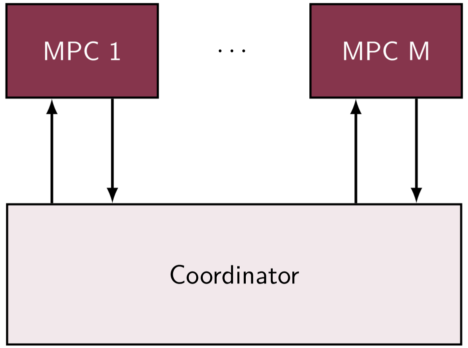


- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

How can a non-cooperative agent attack?

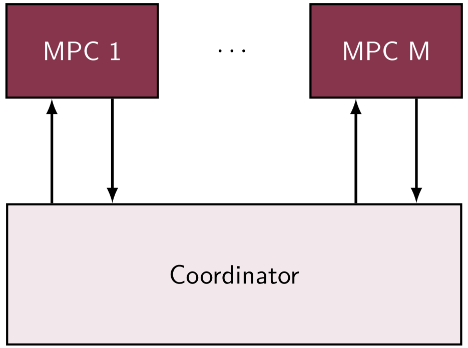
Literature



- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

Literature

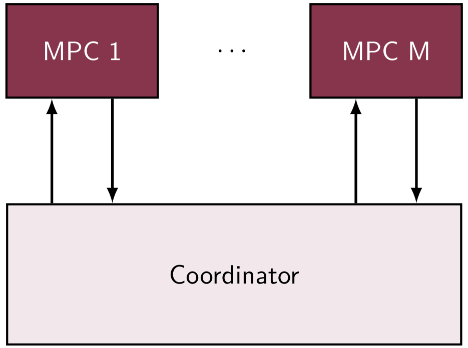


- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

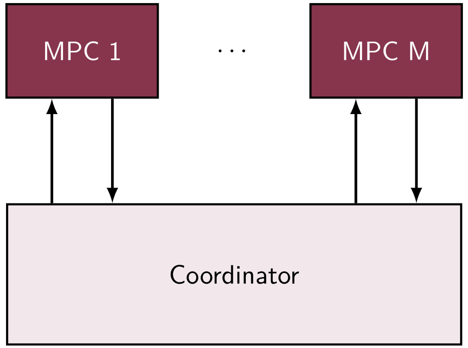
How can a non-cooperative agent attack?

Literature



- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks



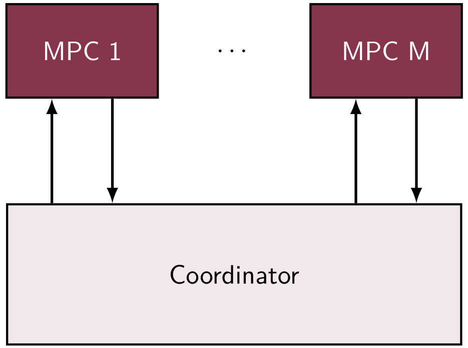
- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

CentraleSupélec

How can a non-cooperative agent attack?

Literature



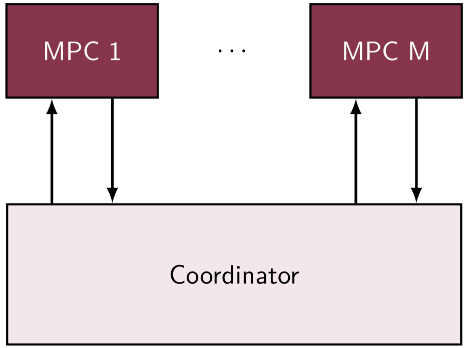
- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

(use control different from the agreed)

How can a non-cooperative agent attack?

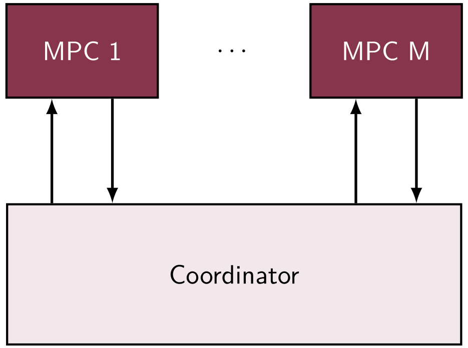
Literature



- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

How can a non-cooperative agent attack?

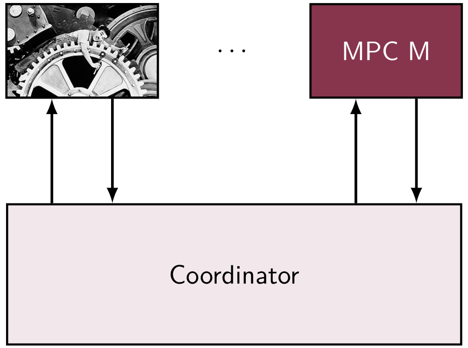


- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks

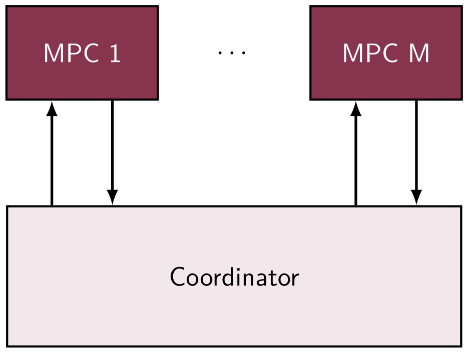
How can a non-cooperative agent attack?

Literature



- [Vel+17a; CMI18] present attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent

Deception Attacks (Internal change)

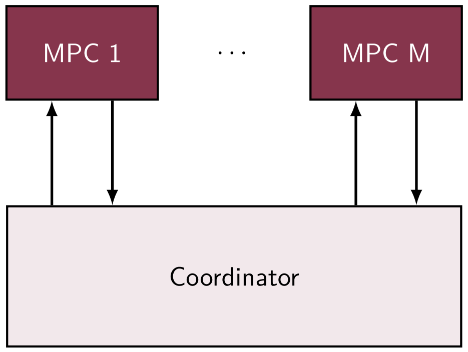


- We are in coordinator's shoes
- What matters is the interface
    - Attacker changes communication

CentraleSupélec

How can a non-cooperative agent attack?

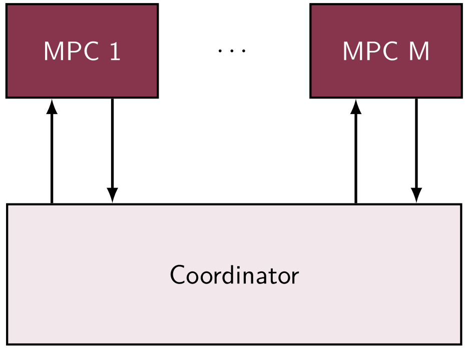
Our approach



- We are in coordinator's shoes
- What matters is the interface
  - Attacker changes communication
    - False Data Injection

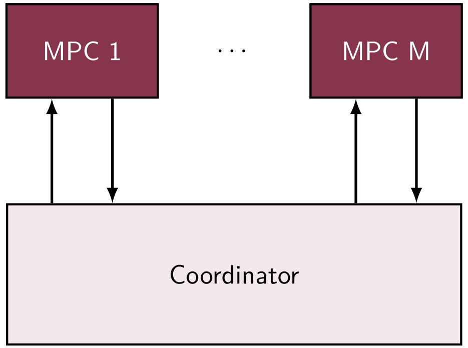
How can a non-cooperative agent attack?

Our approach



- We are in coordinator's shoes
- What matters is the interface
  - Attacker changes communication
    - False Data Injection

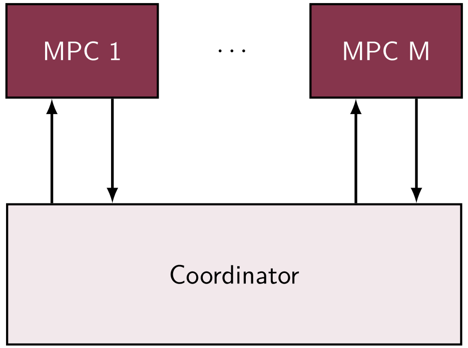
CentraleSupélec



- We are in coordinator's shoes
- What matters is the interface
  - Attacker changes communication
    - False Data Injection

How can a non-cooperative agent attack?

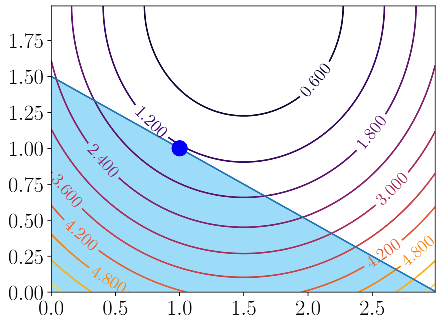
Our approach



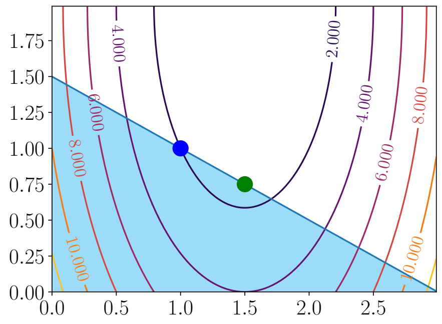
- We are in coordinator's shoes
- What matters is the interface
  - Attacker changes communication
    - False Data Injection

Consequence of an attack

- Attack modifies optimization problem
  - Optimum value is shifted



Original minimum.

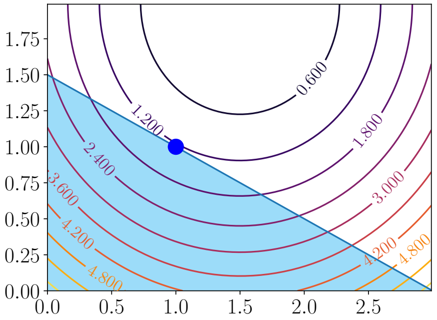


Minimum after attack.

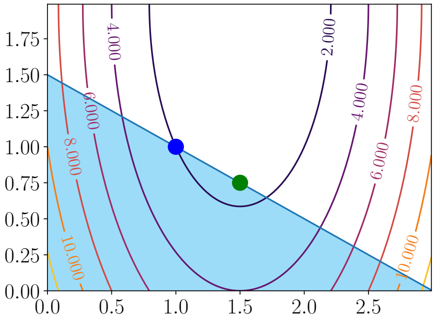
CentraleSupélec

Consequence of an attack

- Attack modifies optimization problem
  - Optimum value is shifted



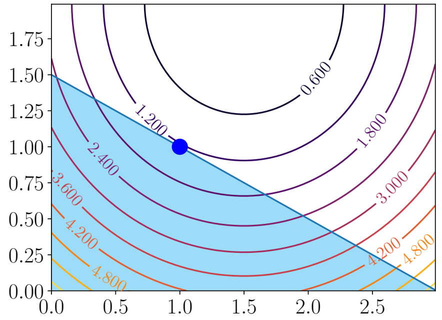
Original minimum.



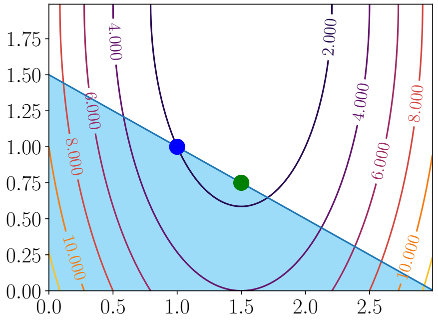
Minimum after attack.

CentraleSupélec

- Attack modifies optimization problem
  - Optimum value is shifted



Original minimum.



Minimum after attack.

Mitigating the effects

- We can recover by
  - Ignoring attacker
  - Recuperating original behavior (at least trying)

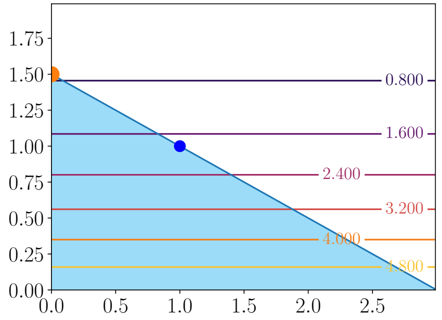
CentraleSupélec

Mitigating the effects

- We can recover by
  - Ignoring attacker
  - Recuperating original behavior (at least trying)

CentraleSupélec

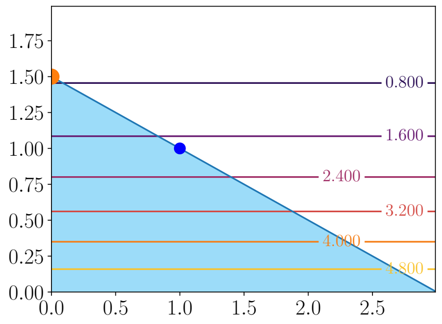
- We can recover by
  - Ignoring attacker
  - Recuperating original behavior (at least trying)



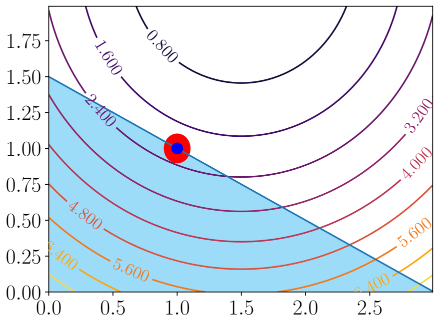
Ignore attacker.

Mitigating the effects

- We can recover by
  - Ignoring attacker
  - Recuperating original behavior (at least trying)



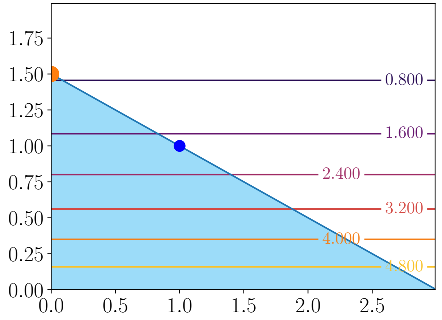
Ignore attacker.



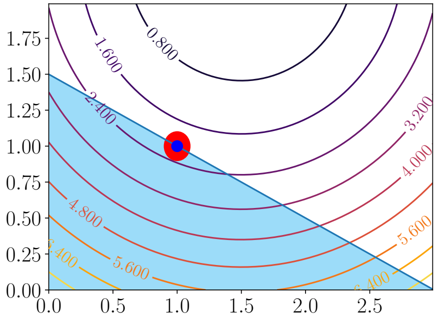
Recover original behavior.

CentraleSupélec

- We can recover by
  - Ignoring attacker
  - Recuperating original behavior (at least trying)



Ignore attacker.



Recover original behavior.

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

CentraleSupélec

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

$\Big\{$ Attack free

CentraleSupélec

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes $\begin{cases} \text{Attack free} \\ \text{When attack detected} \end{cases}$
  1. Detection/Isolation
  2. Mitigation

CentraleSupélec

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes $\begin{cases} \text{Attack free} \\ \text{When attack detected} \end{cases}$
  1. Detection/Isolation
  2. Mitigation

CentraleSupélec

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes $\begin{cases} \text{Attack free} \\ \text{When attack detected} \end{cases}$
  1. Detection/Isolation
  2. Mitigation

CentraleSupélec

Classification of mitigation techniques

- Passive (Robust) - 1 mode
- Active (Resilient) - 2 modes
  1. Detection/Isolation
  2. Mitigation

$\begin{cases} \text{Attack free} \\ \text{When attack detected} \end{cases}$

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites,

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites, We propose a resilient method for two kind of systems with increasing complexities.

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

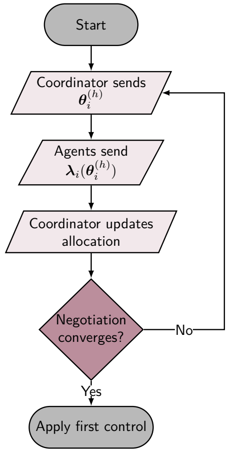
CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites, We propose a resilient method for two kind of systems with increasing complexities.

Outline

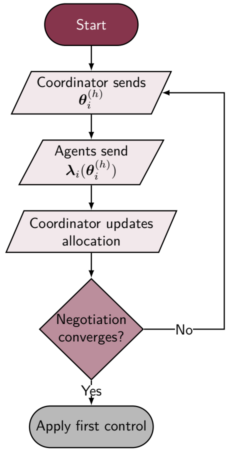
1. Vulnerabilities in distributed MPC based on Primal Decomposition
   - What is the Primal Decomposition?
   - How can an agent attack?
   - Consequences

Quantity Decomposition | Resource Allocation



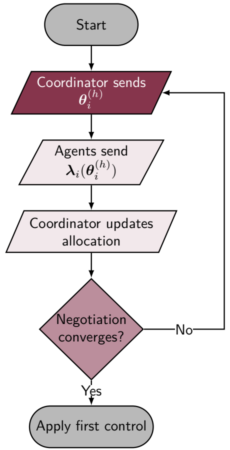
In a flowchart for a quantity decomposition based DMPC,

Quantity Decomposition | Resource Allocation



In a flowchart for a quantity decomposition based DMPC,

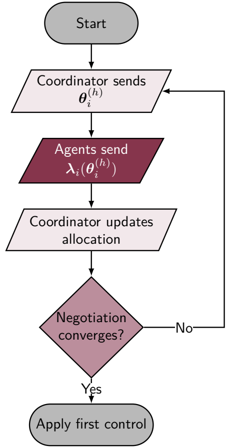
Quantity Decomposition | Resource Allocation



Allocation θ_i

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta,

Quantity Decomposition | Resource Allocation

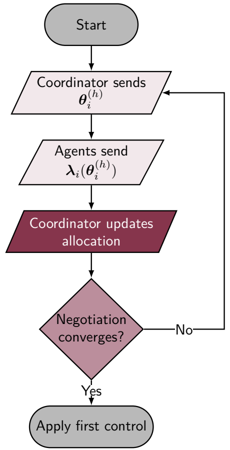


Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda,

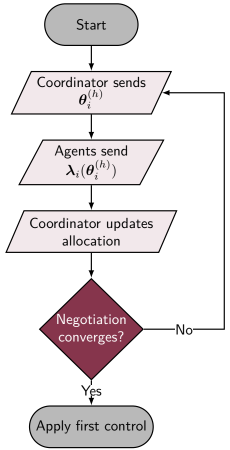
Quantity Decomposition | Resource Allocation



Allocation $\boldsymbol{\theta}_i$
Dissatisfaction $\boldsymbol{\lambda}_i$

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation.

Quantity Decomposition | Resource Allocation



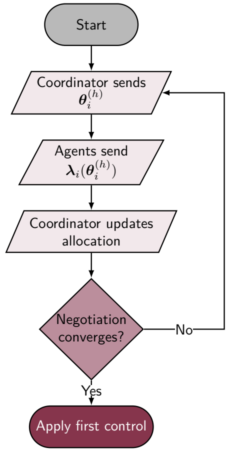
Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

CentraleSupélec

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation. If negotiation converges,

Quantity Decomposition | Resource Allocation



Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

CentraleSupélec

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation. If negotiation converges, then the negotiation ends and each agent applies the first element of the control found

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned}\underset{\boldsymbol{u}_1,\dots,\boldsymbol{u}_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i,\boldsymbol{u}_i)\\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{u}_{\mathsf{total}}\end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned}\underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i,\boldsymbol{u}_i)\\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i\end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{array}{ll} \underset{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M}{\text{minimize}} & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s.t.} & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{array}$$

For each $i \in \mathcal{M}$

$$\begin{array}{ll} \underset{\boldsymbol{u}_i}{\text{minimize}} & J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s. t.} & \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{array}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i \, : \boldsymbol{\lambda}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{\boldsymbol{u}_1,\dots,\boldsymbol{u}_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{aligned}$$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned}\underset{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i,\boldsymbol{u}_i)\\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}}\end{aligned}$$

↓ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned}\underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i,\boldsymbol{u}_i)\\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \leq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i\end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{u_1,\dots,u_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{aligned}$$

↓ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{u_1,\dots,u_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{u_1,\dots,u_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

- Objective is sum of local ones
- Constraints couple variables

$$\begin{aligned} \underset{\boldsymbol{u}_1,\dots,\boldsymbol{u}_M}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \leq \boldsymbol{u}_{\text{total}} \end{aligned}$$

$\downarrow$ For each $i \in \mathcal{M}$

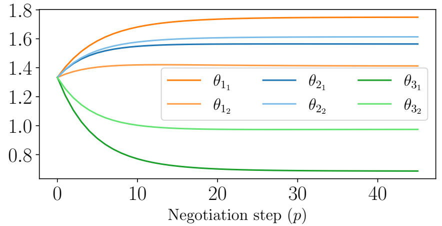
1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$
4. Allocation is updated (respecting global constraint)

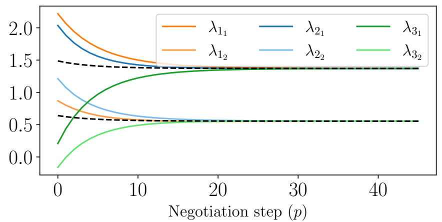
$$\begin{aligned} \underset{\boldsymbol{u}_i}{\text{minimize}} \quad & J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s. t.} \quad & \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \leq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

Quantity Decomposition | Resource Allocation

Until everybody is equally dissatisfied

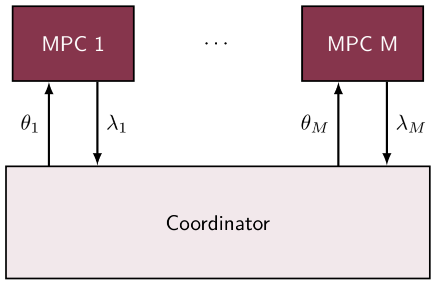




CentraleSupélec

How can a non-cooperative agent attack?

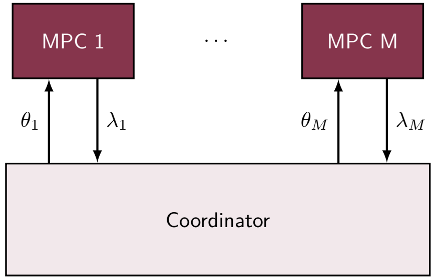
Our approach



- $\boldsymbol{\lambda}_i$ is the only interface
- $\boldsymbol{\lambda}_i$ depends on local parameters
- Malicious agent modifies $\boldsymbol{\lambda}_i$

How can a non-cooperative agent attack?

Our approach



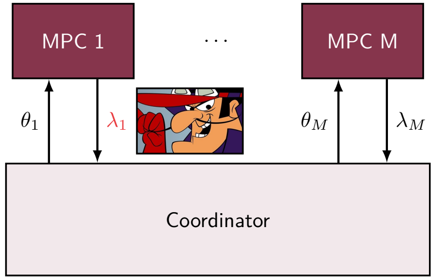
- $\boldsymbol{\lambda}_i$ is the only interface
- $\boldsymbol{\lambda}_i$ depends on local parameters
- Malicious agent modifies $\boldsymbol{\lambda}_i$

CentraleSupélec

How can an agent attack the system? Its only interface with the coordination is lambda

How can a non-cooperative agent attack?

Our approach

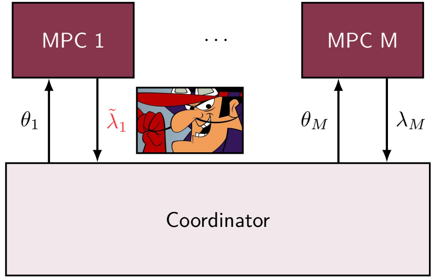


- $\boldsymbol{\lambda}_i$ is the only interface
- $\boldsymbol{\lambda}_i$ depends on local parameters
- Malicious agent modifies $\boldsymbol{\lambda}_i$

How can an agent attack the system? Its only interface with the coordination is lambdaso we suppose it attacks by sending a different $lambda_i$, modified by a function $gamma_i$ of $lambda_i$

How can a non-cooperative agent attack?

Our approach



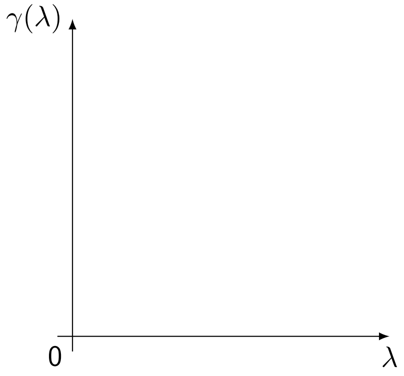
- $\boldsymbol{\lambda}_i$ is the only interface
- $\boldsymbol{\lambda}_i$ depends on local parameters
- Malicious agent modifies $\boldsymbol{\lambda}_i$

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i)$$

How can an agent attack the system? Its only interface with the coordination is lambdaso we suppose it attacks by sending a different $lambda_i$, modified by a function $gamma_i$ of $lambda_i$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

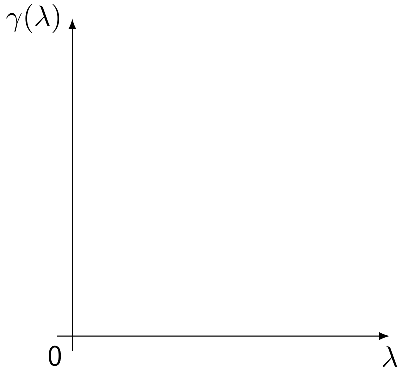
Assumptions

- *Attacker satisfied only if it really is* $\gamma(\lambda) = 0 \to \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \to \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\lambda}_i = T_i[k]\lambda_i \to \exists T_i[k]^{-1}$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

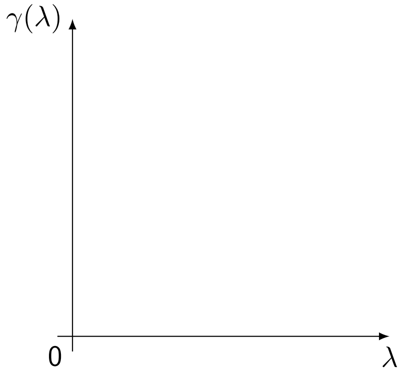
Assumptions

- *Attacker satisfied only if it really is* $\gamma(\lambda) = 0 \to \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \to \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \to \exists T_i[k]^{-1}$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

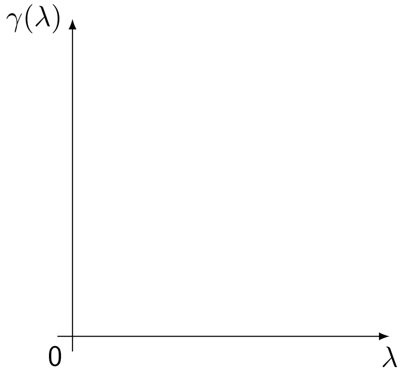
Assumptions

- *Attacker satisfied only if it really is* $\gamma(\lambda) = 0 \to \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \to \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \to \exists T_i[k]^{-1}$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

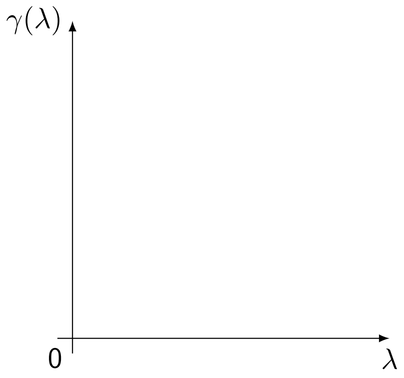
Assumptions

- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

Assumptions

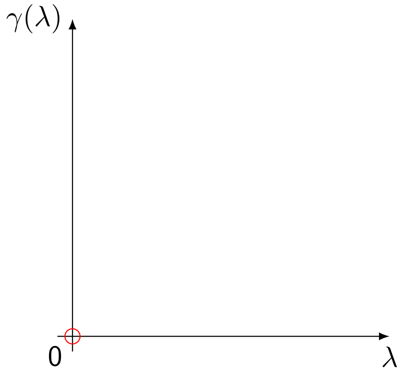
- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

CentraleSupélec

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

Assumptions

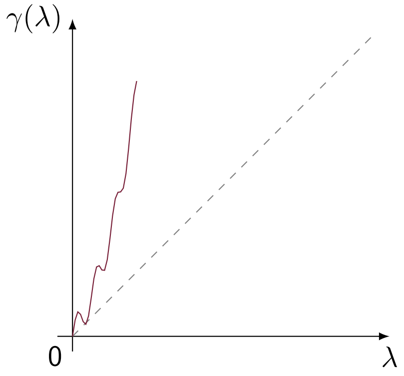
- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

CentraleSupélec

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

Assumptions

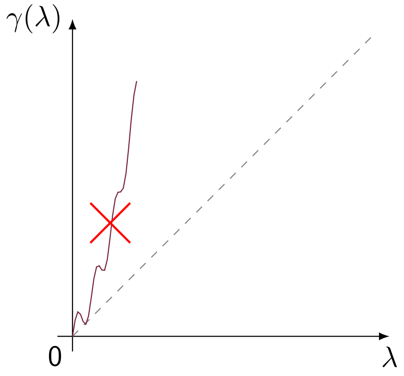
- *Attacker satisfied only if it really is*
    - $\gamma(\lambda) = 0 \to \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing*
  $\lambda_b > \lambda_a \to \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \to \exists T_i[k]^{-1}$

CentraleSupélec

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

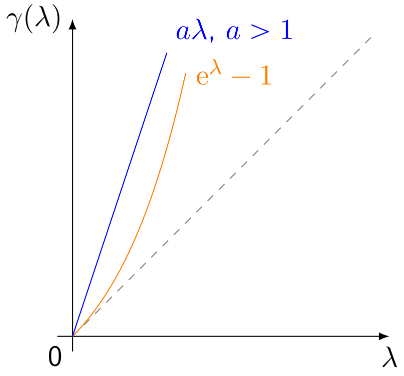
Assumptions

- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

Assumptions

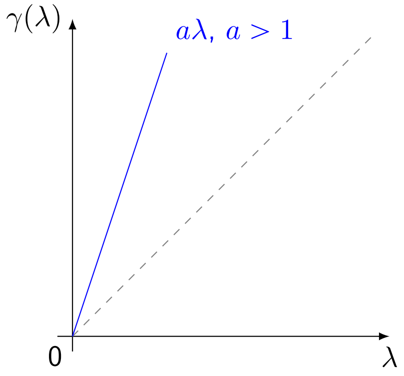
- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

CentraleSupélec

How does an agent lie?

Liar, Liar, Pants of fire



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

Assumptions

- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- *Attacker is greedy* $\gamma(\lambda) > \lambda$
- *Attack is monotonically increasing* $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

- Invertible
- If $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i \rightarrow \exists T_i[k]^{-1}$

CentraleSupélec

Example

4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\lambda}_1 = \gamma_1(\lambda_1) = \tau_1 I \lambda_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating

Example

4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

We give an example of 4 agents negotiating Agent 1 is non-cooperative

Example

4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when

Example

4 distinct agents

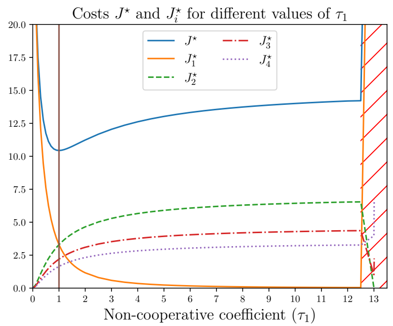
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high

Example



4 distinct agents

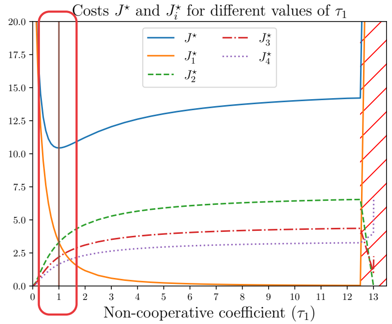
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Example



4 distinct agents

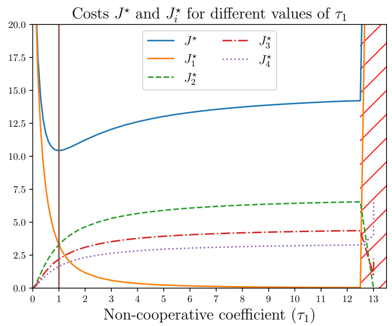
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Example



4 distinct agents

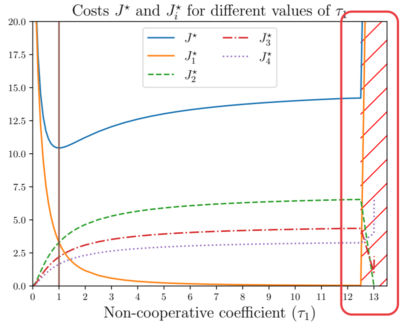
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Example



4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

- But can we mitigate these effects?
- Yes! (At least in some cases)

- But can we mitigate these effects?
- Yes! (At least in some cases)

- But can we mitigate these effects?
- Yes! (At least in some cases)

- But can we mitigate these effects?
- Yes! (At least in some cases)

CentraleSupélec

Outline

2. Resilient Primal Decomposition-based dMPC for deprived systems
   - Analyzing deprived systems
   - Building an algorithm
   - Applying mechanism

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i \boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

[2] Under some conditions

CentraleSupélec

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i \boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

[2]Under some conditions

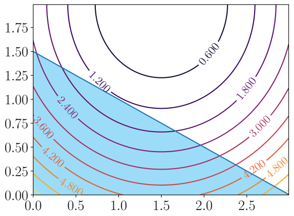
CentraleSupélec

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|^2_{H_i} + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$



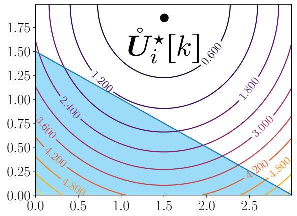
[2] Under some conditions

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$



[2] Under some conditions

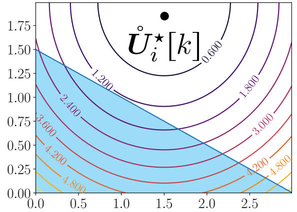
CentraleSupélec

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \tfrac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i \boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{aligned}$$



[2] Under some conditions

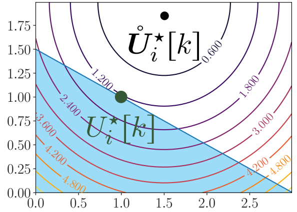
CentraleSupélec

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i \boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{aligned}$$



[2] Under some conditions ▸ see here

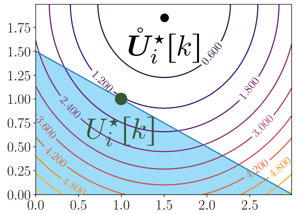
CentraleSupélec

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $\bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] \geq \boldsymbol{\theta}_i[k] \rightarrow$ Scarcity
  - Solution projected onto boundary
  - Same as with equality constraints[2]

$$
\begin{aligned}
\underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\
\text{subject to} \quad & \bar{\Gamma}_i \boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k]
\end{aligned}
$$



[2]Under some conditions see here

CentraleSupélec

Deprived Systems

But why?

- No Scarcity
  - ⟶ All constraints satisfied
  - ⟶ No coordination needed
  - ⟶ No incentive to cheat
- Scarcity
  - ⟶ Competition
  - ⟶ Consensus/Compromise
  - ⟶ Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - → All constraints satisfied
  - → No coordination needed
  - → No incentive to cheat
- Scarcity
  - → Competition
  - → Consensus/Compromise
  - → Agents may cheat

CentraleSupélec

Deprived Systems

But why?

- No Scarcity
  - → All constraints satisfied
  - → No coordination needed
  - → No incentive to cheat
- Scarcity
  - → Competition
  - → Consensus/Compromise
  - → Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - $\longrightarrow$ All constraints satisfied
  - $\longrightarrow$ No coordination needed
  - $\longrightarrow$ No incentive to cheat

- Scarcity
  - $\longrightarrow$ Competition
  - $\longrightarrow$ Consensus/Compromise
  - $\longrightarrow$ Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - $\longrightarrow$ All constraints satisfied
  - $\longrightarrow$ No coordination needed
  - $\longrightarrow$ No incentive to cheat

- Scarcity
  - $\longrightarrow$ Competition
  - $\longrightarrow$ Consensus/Compromise
  - $\longrightarrow$ Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - → All constraints satisfied
  - → No coordination needed
  - → No incentive to cheat
- Scarcity
  - → Competition
  - → Consensus/Compromise
  - → Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - → All constraints satisfied
  - → No coordination needed
  - → No incentive to cheat
- Scarcity
  - → Competition
  - → Consensus/Compromise
  - → Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - $\longrightarrow$ All constraints satisfied
  - $\longrightarrow$ No coordination needed
  - $\longrightarrow$ No incentive to cheat
- Scarcity
  - $\longrightarrow$ Competition
  - $\longrightarrow$ Consensus/Compromise
  - $\longrightarrow$ Agents may cheat

Deprived Systems

But why?

- No Scarcity
  - → All constraints satisfied
  - → No coordination needed
  - → No incentive to cheat
- Scarcity
  - → Competition
  - → Consensus/Compromise
  - → Agents may cheat

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{ \begin{array}{l} \bullet\ P_i \text{ is time invariant} \\ \bullet\ \boldsymbol{s}_i[k] \text{ is time variant} \end{array} \right.$

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{aligned}\underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{U}_i[k]\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k]\end{aligned}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator)
- P_i is time invariant
- $\boldsymbol{s}_i[k]$ is time variant

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i \boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i \boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{\right.$
- P_i is time invariant
- $\boldsymbol{s}_i[k]$ is time variant

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - s_i[k]$$

(local parameters unknown by coordinator) $\left\{ \begin{array}{l} \bullet\ P_i \text{ is time invariant} \\ \bullet\ s_i[k] \text{ is time variant} \end{array} \right.$

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$
\begin{aligned}
\underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\|\boldsymbol{U}_i[k]\|^2_{H_i} + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\
\text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k]
\end{aligned}
$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\begin{cases} \bullet\ P_i \text{ is time invariant} \\ \bullet\ \boldsymbol{s}_i[k] \text{ is time variant} \end{cases}$

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{\right.$
- P_i is time invariant
- $\boldsymbol{s}_i[k]$ is time variant

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{ \begin{array}{l} \bullet\ P_i \text{ is time invariant} \\ \bullet\ \boldsymbol{s}_i[k] \text{ is time variant} \end{array} \right.$

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{ \begin{array}{l} \bullet\ P_i \text{ is time invariant} \\ \bullet\ s_i[k] \text{ is time variant} \end{array} \right.$

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i \boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{aligned}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i \boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator)

- P_i is time invariant
- $\boldsymbol{s}_i[k]$ is time variant

CentraleSupélec

Deprived Systems

Analysis

Assumptions

- *Quadratic local problems*
- *Scarcity*

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{aligned}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{ \right.$
- P_i is time invariant
- $\boldsymbol{s}_i[k]$ is time variant

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\lambda}_i = T_i[k]\lambda_i$
  - Parameters modified

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -T_i[k]P_i\boldsymbol{\theta}_i[k] - T_i[k]\boldsymbol{s}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

CentraleSupélec

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\hat{\boldsymbol{s}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\hat{\bar{P}}_i[k]\boldsymbol{\theta}_i - \hat{\bar{\boldsymbol{s}}}_i[k]$$

- If $\left\| \hat{\bar{P}}_i[k] - \bar{P}_i \right\|_F > \epsilon_P \to$ Attack
- Ok, but how can we estimate $\hat{\bar{P}}_i[k]$?

[1] Using Recursive Least Squares for example

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\hat{\hat{s}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\hat{\hat{P}}_i[k]\boldsymbol{\theta}_i - \hat{\hat{s}}_i[k]$$

- If $\left\|\hat{\hat{P}}_i[k] - \bar{P}_i\right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\hat{\hat{P}}_i[k]$?

[1]Using Recursive Least Squares for example

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\hat{\tilde{s}}_i[k]$ such as:

$$\tilde{\lambda}_i = -\hat{\tilde{P}}_i[k]\theta_i - \hat{\tilde{s}}_i[k]$$

- If $\left\| \hat{\tilde{P}}_i[k] - \bar{P}_i \right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\hat{\tilde{P}}_i[k]$?

[1]Using Recursive Least Squares for example

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\hat{\tilde{\boldsymbol{s}}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\hat{\tilde{P}}_i[k]\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i[k]$$

- If $\left\| \hat{\tilde{P}}_i[k] - \bar{P}_i \right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\hat{\tilde{P}}_i[k]$?

[1]Using Recursive Least Squares for example

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\widehat{\tilde{\boldsymbol{s}}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\widehat{\tilde{P}}_i[k]\boldsymbol{\theta}_i - \widehat{\tilde{\boldsymbol{s}}}_i[k]$$

- If $\left\| \widehat{\tilde{P}}_i[k] - \bar{P}_i \right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\widehat{\tilde{P}}_i[k]$?

[1] Using Recursive Least Squares for example

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k
and if the estimated P tilde is different from the nominal P, then there is an attack

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{P}_i[k]$ and $\hat{\tilde{s}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\hat{\tilde{P}}_i[k]\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i[k]$$

- If $\left\| \hat{\tilde{P}}_i[k] - \bar{P}_i \right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\hat{\tilde{P}}_i[k]$?

[1] Using Recursive Least Squares for example

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k
and if the estimated P tilde is different from the nominal P, then there is an attack

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

- If we estimate[1] $\hat{\tilde{P}}_i[k]$ and $\hat{\tilde{\boldsymbol{s}}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = -\hat{\tilde{P}}_i[k]\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i[k]$$

- If $\left\|\hat{\tilde{P}}_i[k] - \bar{P}_i\right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\hat{\tilde{P}}_i[k]$?

[1] Using Recursive Least Squares for example

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k
and if the estimated P tilde is different from the nominal P, then there is an attack

Estimating $\widehat{\widetilde{P}}_i[k]$

- We need to estimate $\widehat{\widetilde{P}}_i[k]$ and $\widehat{s}_i[k]$ simultaneously
- Challenge: Estimation during negotiation fails
  - Update function couples θ_i^p and $\lambda_i^p \rightarrow$ low input excitation
- Solution: Send a random[3] sequence to increase excitation.

[3] A random signal has persistent excitation of any order ( )

Estimating $\widehat{\widetilde{P}}_i[k]$

- We need to estimate $\widehat{\widetilde{P}}_i[k]$ and $\widehat{\widetilde{\boldsymbol{s}}}_i[k]$ simultaneously
- Challenge: Estimation during negotiation fails
    - Update function couples θ_i^p and $\lambda_i^p \rightarrow$ low input excitation
- Solution: Send a random[3] sequence to increase excitation.

[3] A random signal has persistent excitation of any order ( )

Estimating $\widehat{\tilde{P}}_i[k]$

- We need to estimate $\widehat{\tilde{P}}_i[k]$ and $\widehat{\tilde{\boldsymbol{s}}}_i[k]$ simultaneously
- Challenge: Estimation during negotiation fails
  - Update function couples $\boldsymbol{\theta}_i^p$ and $\boldsymbol{\lambda}_i^p$ → low input excitation
- Solution: Send a random[3] sequence to increase excitation.

[3] A random signal has persistent excitation of any order ( )

Estimating $\widehat{\tilde{P}}_i[k]$

- We need to estimate $\widehat{\tilde{P}}_i[k]$ and $\widehat{\tilde{\boldsymbol{s}}}_i[k]$ simultaneously
- Challenge: Estimation during negotiation fails
  - Update function couples $\boldsymbol{\theta}_i^p$ and $\boldsymbol{\lambda}_i^p$ → low input excitation
- Solution: Send a random[3] sequence to increase excitation.

[3] A random signal has persistent excitation of any order ( )

CentraleSupélec

Estimating $\hat{\tilde{P}}_i[k]$

- We need to estimate $\hat{\tilde{P}}_i[k]$ and $\hat{\tilde{\boldsymbol{s}}}_i[k]$ simultaneously
- Challenge: Estimation during negotiation fails
  - Update function couples $\boldsymbol{\theta}_i^p$ and $\boldsymbol{\lambda}_i^p$ → low input excitation
- Solution: Send a random[3] sequence to increase excitation.

[3] A random signal has persistent excitation of any order (Adaptive Control)

Classification of mitigation techniques

- Active (Resilient)
  1. Detection/Isolation ✔
  2. Mitigation ?

Classification of mitigation techniques

- Active (Resilient)
  1. Detection/Isolation ✔
  2. Mitigation ?

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- We now have $\widehat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i \widehat{\tilde{P}}_i[k]^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\hat{\boldsymbol{s}}_i[k]$$

- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- We now have $\widehat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i \widehat{\tilde{P}}_i[k]^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\widehat{\tilde{\mathbf{s}}}_i[k]$$

- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- We now have $\hat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i\hat{\tilde{P}}_i[k]^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\hat{\tilde{\boldsymbol{s}}}_i[k]$$

- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- We now have $\widehat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i\widehat{\tilde{P}}_i[k]^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\widehat{\tilde{\boldsymbol{s}}}_i[k]$$

- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation
With the estimate of the inverse of T and the estimate of s tilde, we can reconstruct the original lambda

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- We now have $\widehat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i \widehat{\tilde{P}}_i[k]^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\widehat{\tilde{\boldsymbol{s}}}_i[k]$$

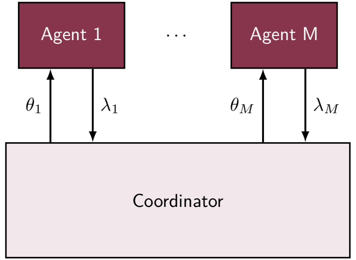
- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation
With the estimate of the inverse of T and the estimate of s tilde, we can reconstruct the original lambda

Complete Mechanism



- Supervise exchanges by inquiring the agents
- Estimate how they will behave

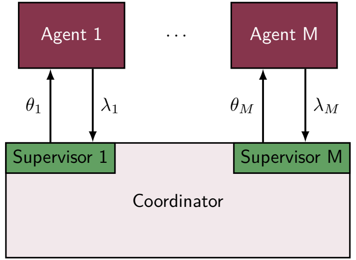
Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

CentraleSupélec

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator

Complete Mechanism



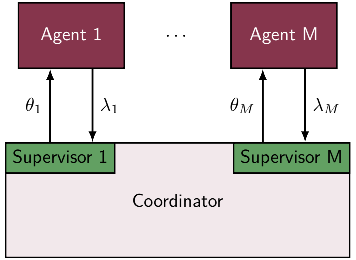
- Supervise exchanges by inquiring the agents
- Estimate how they will behave

Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases,

Complete Mechanism



- Supervise exchanges by inquiring the agents
- Estimate how they will behave

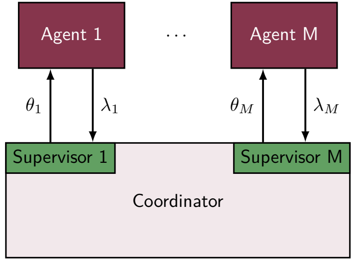
Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

CentraleSupélec

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative

Complete Mechanism



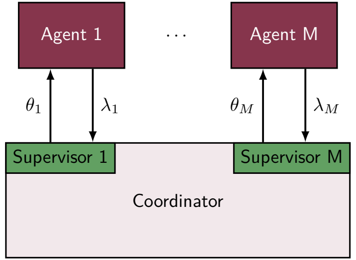
- Supervise exchanges by inquiring the agents
- Estimate how they will behave

Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative and then reconstruct the lambda i s and use in the usual negotiation

Complete Mechanism



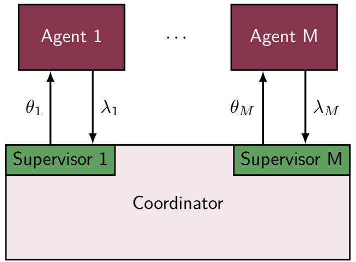
- Supervise exchanges by inquiring the agents
- Estimate how they will behave

Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative and then reconstruct the lambda i s and use in the usual negotiation

Complete Mechanism



- Supervise exchanges by inquiring the agents
- Estimate how they will behave

Two Phases

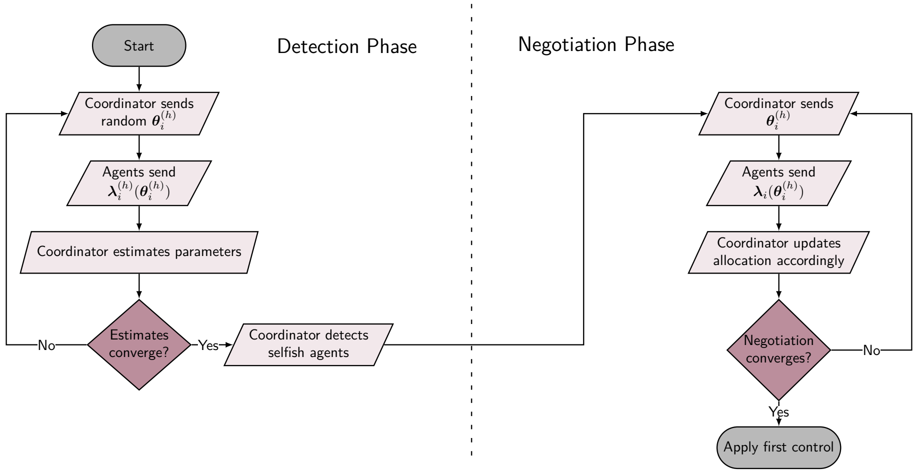
1. Detect which agents are non-cooperative
2. Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

CentraleSupélec

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative and then reconstruct the lambda i s and use in the usual negotiation

Complete algorithm

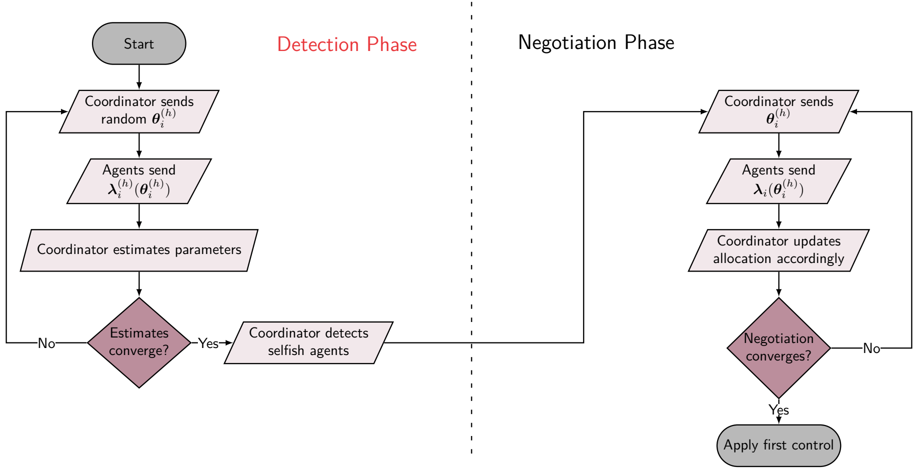
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases

Complete algorithm

RPdMPC-DS

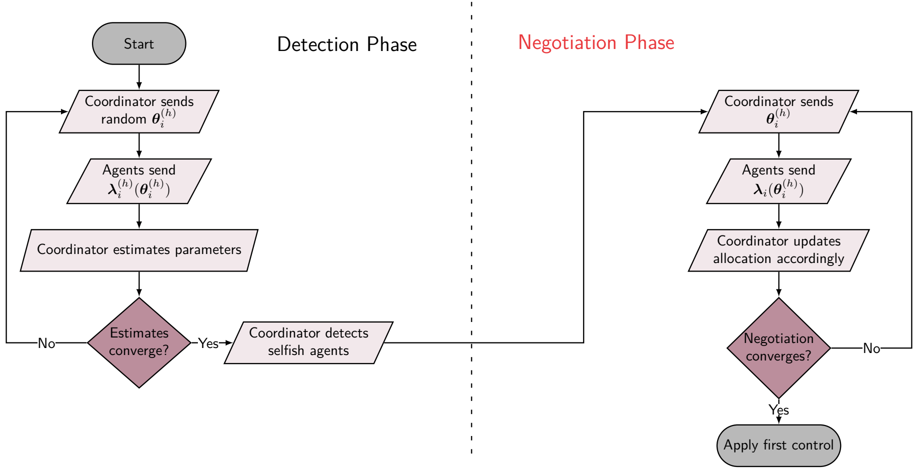


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase

Complete algorithm

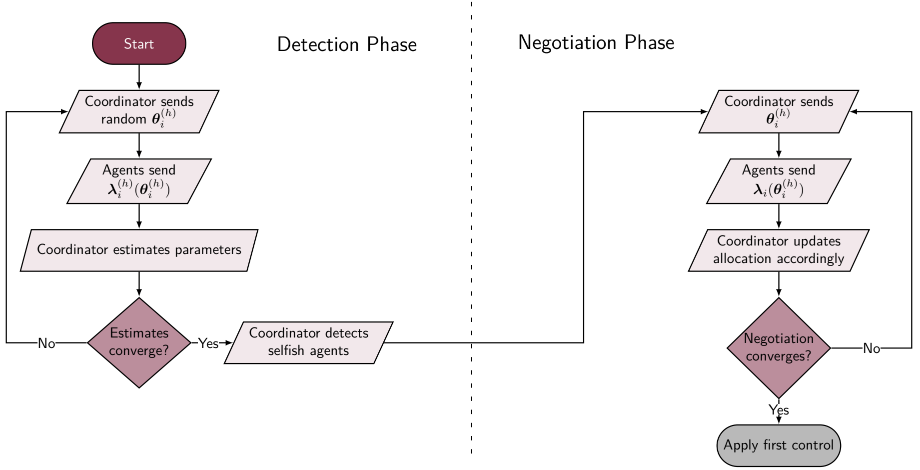
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

RPdMPC-DS

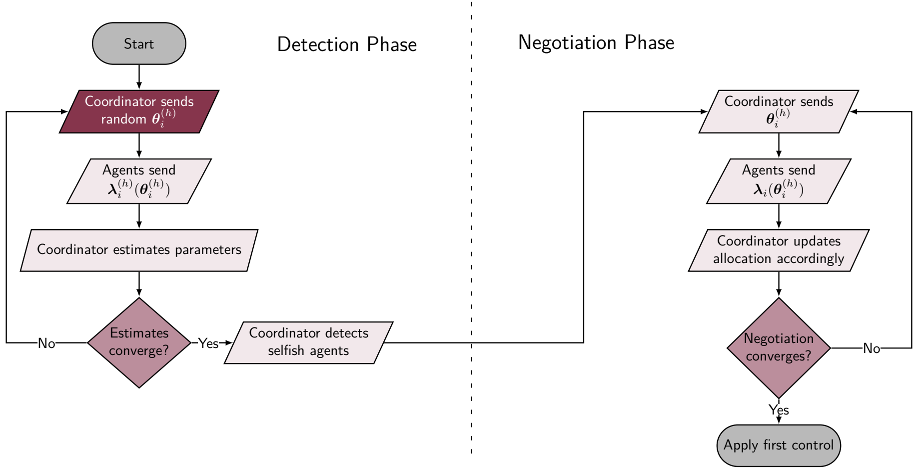


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

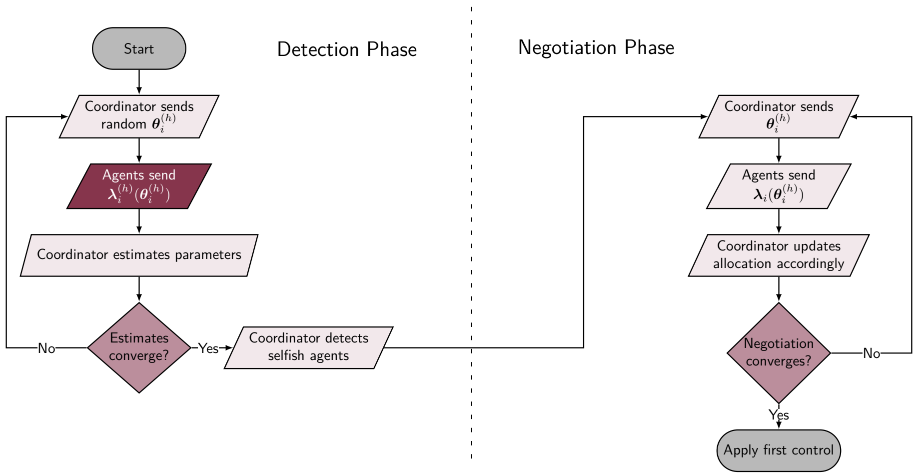
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i

Complete algorithm

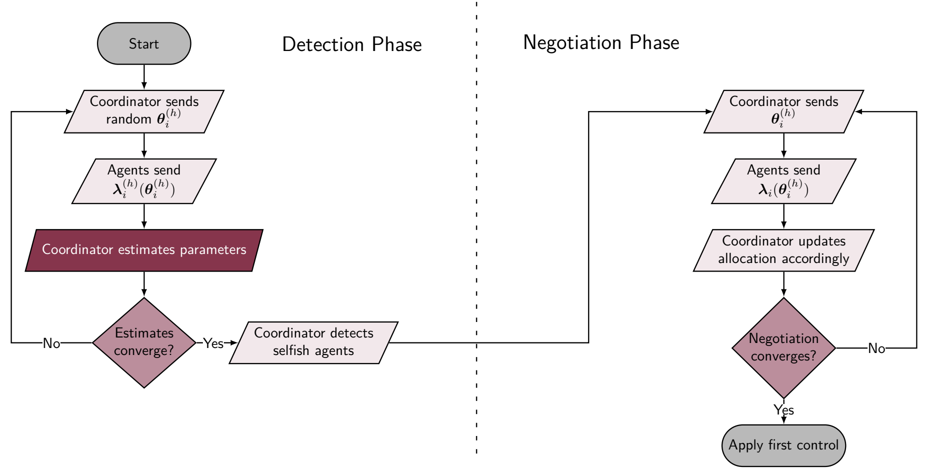
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i

Complete algorithm

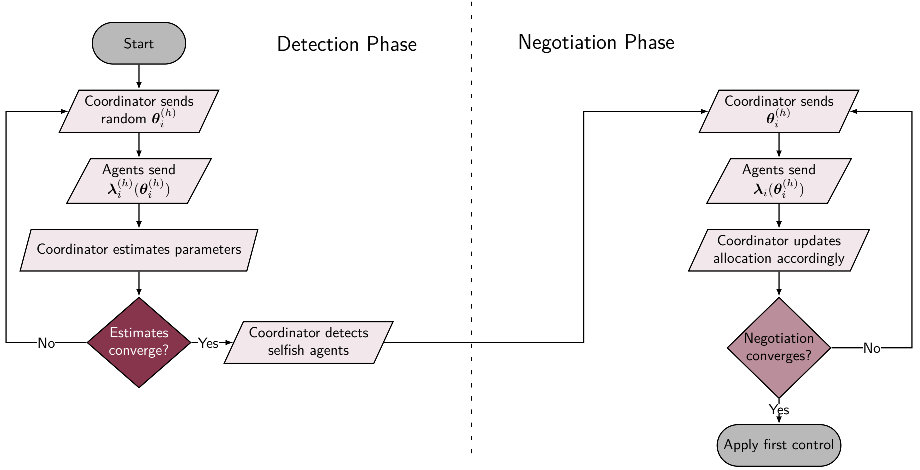
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde

Complete algorithm

RPdMPC-DS

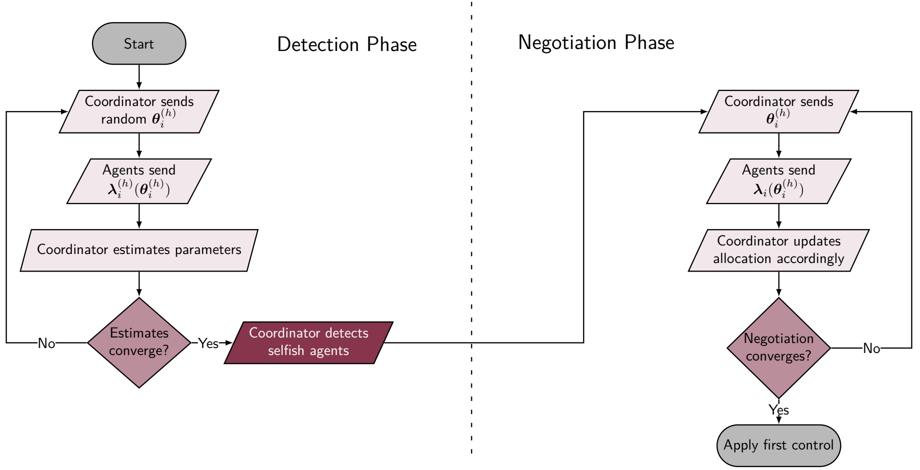


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge

Complete algorithm

RPdMPC-DS

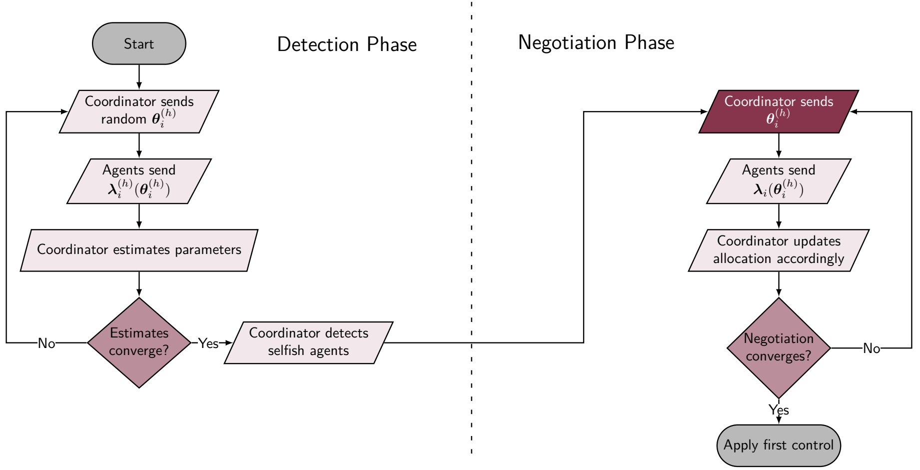


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative

Complete algorithm

RPdMPC-DS

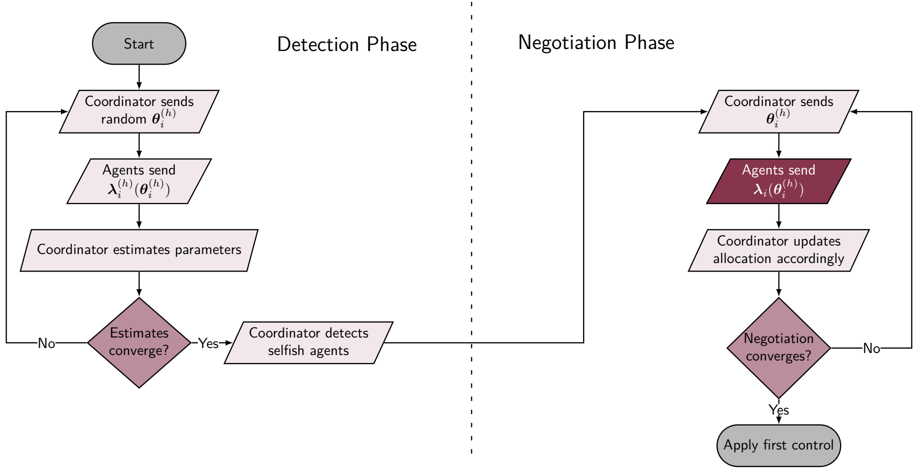


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i

Complete algorithm

RPdMPC-DS

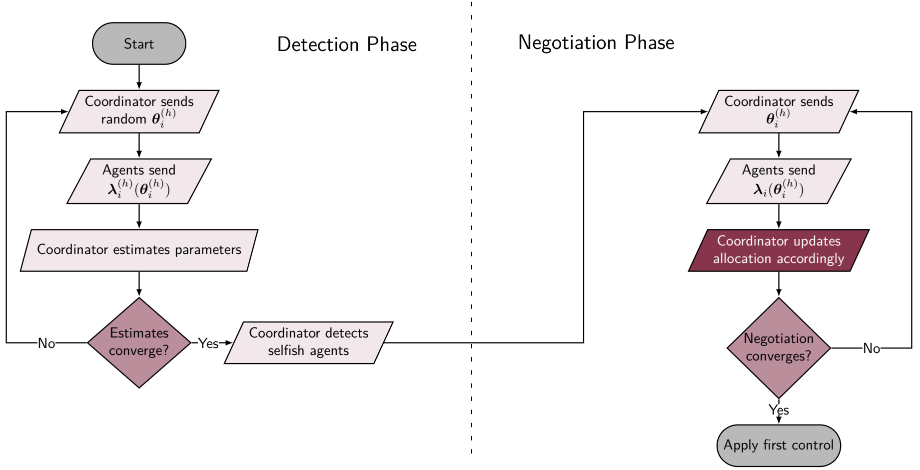


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i

Complete algorithm

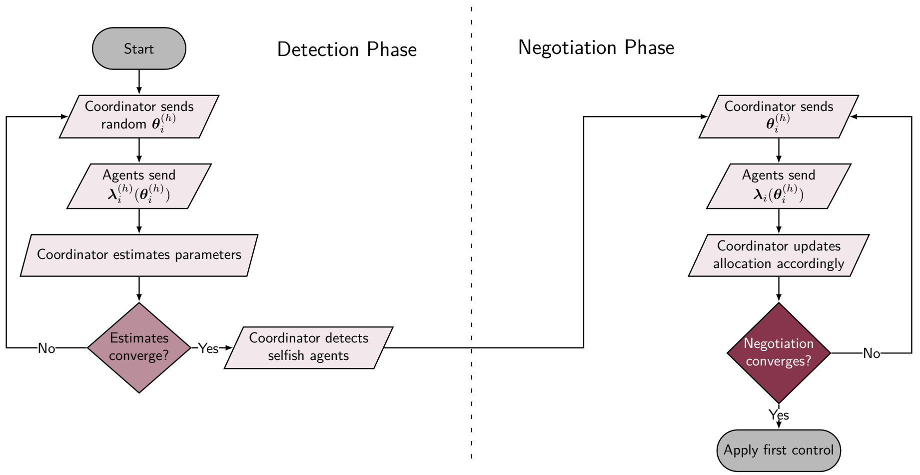
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent

Complete algorithm

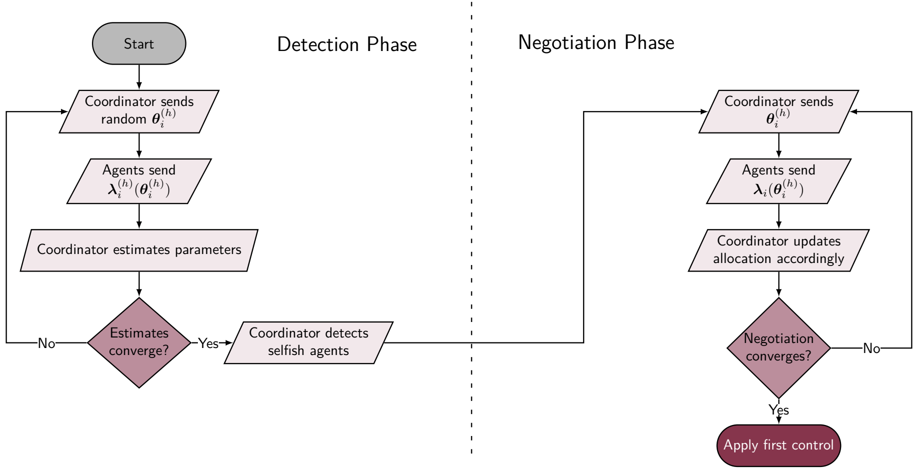
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges

Complete algorithm

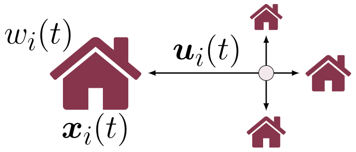
RPdMPC-DS



CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Example

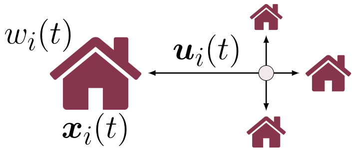


District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of $5\mathrm{h}$
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

We give an academic example of the temperature control of 4 distinct rooms under power scarcity

Example



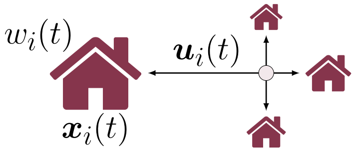
District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of 5h
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model

Example



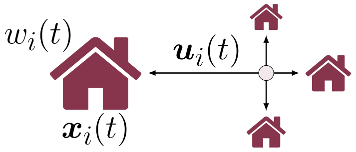
District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of 5h
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint

Example



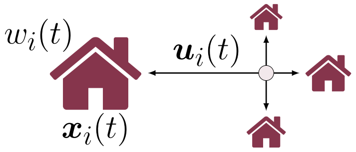
District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of $5\mathrm{h}$
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint

Example

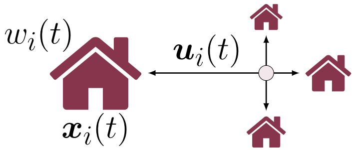


District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of 5h
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h

Example



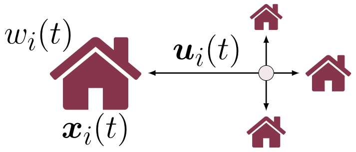
District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of 5h
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes

Example

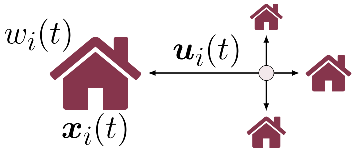


District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of $5\mathrm{h}$
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes
3 scenarios are simulated, the nominal, one where agent 1 presents non cooperative behavior from k greater than 6, and another with the selfish behavior and the secure mechanism activated

Example



District Heating Network (4 Houses)

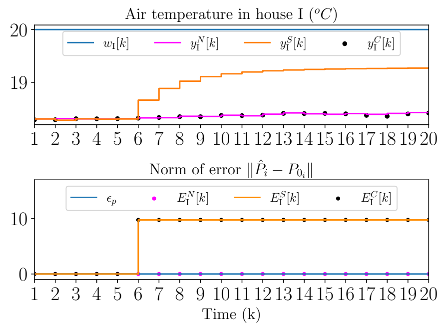
- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of $5\mathrm{h}$
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes
3 scenarios are simulated, the nominal, one where agent 1 presents non cooperative behavior from k greater than 6, and another with the selfish behavior and the secure mechanism activated

Results

Temporal



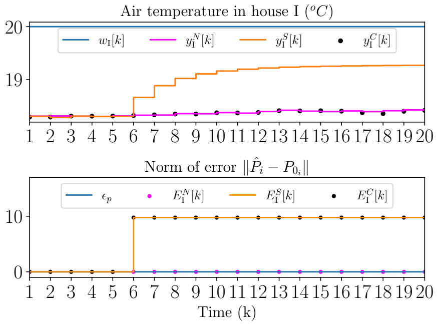
Temperature in house I.
Error $E_I(k)$.
N Nominal, S Selfish, C Corrected

CentraleSupélec

In the figure we see the air temperature and the estimation error for house 1

Results

Temporal



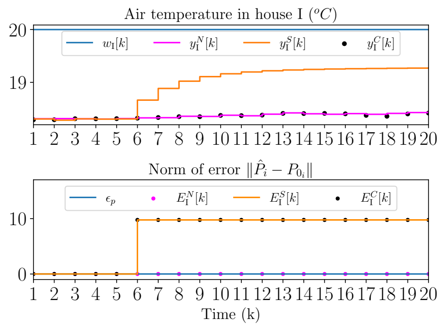
Temperature in house I.
Error $E_I(k)$.
Ⓝ Nominal, Ⓢ Selfish, Ⓒ Corrected

CentraleSupélec

The nominal behavior is in magenta, and as said it cannot reach the setpoint, in blue

Results

Temporal



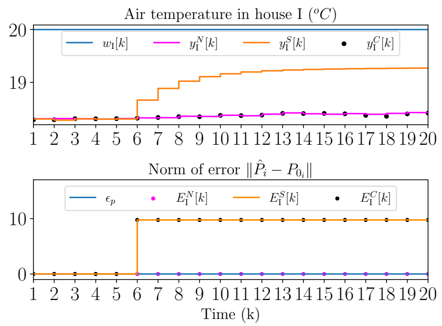
Temperature in house I.
Error $E_I(k)$.
N Nominal, S Selfish, C Corrected

- S Temperature close to reference
- C Temperature close to N

When the room presents selfish behavior (in orange), it reduces its cost and get closer to the setpoint, we see that the attack increases the error

Results

Temporal



Temperature in house I.
Error $E_I(k)$.
Ⓝ Nominal, Ⓢ Selfish, Ⓒ Corrected

- Ⓢ Temperature close to reference
- Ⓒ Temperature close to Ⓝ

Now, for the case with correction (the black dots), even if it attacks the system, the temperature is close to its nominal value

Results

Costs

Objective functions J_i (% error)

Agent	Scenario N	Scenario S	Scenario C
I	299.5	$190.8\ (-36.3)$	$301.0\ (0.0)$
II	192.4	$234.1\ (21.7)$	$191.4\ (-0.5)$
III	305.9	$359.1\ (17.4)$	$305.9\ (-0.0)$
IV	297.5	$349.9\ (17.6)$	$297.2\ (-0.1)$
Global	1095.3	$1133.9\ (3.5)$	$1095.5\ (0.0)$

CentraleSupélec

Now, if we compare the costs for each scenario we see how the cost of agent 1 decreases when it attacks, while the cost of other agents increase.

Results

Costs

Objective functions J_i (% error)

Agent	Scenario N	Scenario S	Scenario C
I	299.5	190.8 (−36.3)	301.0 (0.0)
II	192.4	234.1 (21.7)	191.4 (−0.5)
III	305.9	359.1 (17.4)	305.9 (−0.0)
IV	297.5	349.9 (17.6)	297.2 (−0.1)
Global	1095.3	1133.9 (3.5)	1095.5 (0.0)

CentraleSupélec

Now, if we compare the costs for each scenario we see how the cost of agent 1 decreases when it attacks, while the cost of other agents increase.
When the secure algorithm is activated the costs are very close to the original ones. So

Outline

CentraleSupélec

Relaxing scarcity assumption

- Let's relax the scarcity assumption
- And add some local constraints
- Similarly we have the local problems and update

$$
\begin{aligned}
\underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}[k]\|_H^2 + \boldsymbol{f}[k]^T \boldsymbol{U}[k] \\
\text{subject to} \quad & \bar{\Gamma}\boldsymbol{U}[k] \preceq \boldsymbol{U}_{\mathsf{max}} \\
& \boldsymbol{U}[k] \in \mathcal{U}
\end{aligned}
$$

$$
\begin{aligned}
\underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}_i[k]\|_{H_i}^2 + \boldsymbol{f}_i[k]^T \boldsymbol{U}_i[k] \\
\text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \\
& \boldsymbol{U}_i[k] \in \mathcal{U}_i
\end{aligned}
$$

$$
\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})
$$

Relaxing scarcity assumption

- Let's relax the scarcity assumption
- And add some local constraints
- Similarly we have the local problems and update

$$\begin{aligned} \underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}[k]\|_H^2 + \boldsymbol{f}[k]^T\boldsymbol{U}[k] \\ \text{subject to} \quad & \bar{\Gamma}\boldsymbol{U}[k] \preceq \boldsymbol{U}_{\text{max}} \\ & \boldsymbol{U}[k] \in \mathcal{U} \end{aligned}$$

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}_i[k]\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \\ & \boldsymbol{U}_i[k] \in \mathcal{U}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Relaxing scarcity assumption

- Let's relax the scarcity assumption
- And add some local constraints
- Similarly we have the local problems and update

$$
\begin{aligned}
\underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \tfrac{1}{2}\left\|\boldsymbol{U}[k]\right\|_H^2 + \boldsymbol{f}[k]^T\boldsymbol{U}[k] \\
\text{subject to} \quad & \bar{\Gamma}\boldsymbol{U}[k] \preceq \boldsymbol{U}_{\max} \\
& \boldsymbol{U}[k] \in \mathcal{U}
\end{aligned}
$$

$$
\begin{aligned}
\underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \tfrac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\
\text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \\
& \boldsymbol{U}_i[k] \in \mathcal{U}_i
\end{aligned}
$$

$$
\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})
$$

CentraleSupélec

Relaxing scarcity assumption

- Let's relax the scarcity assumption
- And add some local constraints
- Similarly we have the local problems and update

$$\begin{array}{ll} \underset{\boldsymbol{U}[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}[k]\right\|_H^2 + \boldsymbol{f}[k]^T\boldsymbol{U}[k] \\ \text{subject to} & \bar{\Gamma}\boldsymbol{U}[k] \preceq \boldsymbol{U}_{\max} \\ & \boldsymbol{U}[k] \in \mathcal{U} \end{array}$$

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \\ & \boldsymbol{U}_i[k] \in \mathcal{U}_i \end{array}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Relaxing scarcity assumption

- Let's relax the scarcity assumption
- And add some local constraints
- Similarly we have the local problems and update

$$\begin{aligned} \underset{\boldsymbol{U}[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}[k]\|_H^2 + \boldsymbol{f}[k]^T\boldsymbol{U}[k] \\ \text{subject to} \quad & \bar{\Gamma}\boldsymbol{U}[k] \preceq \boldsymbol{U}_{\max} \\ & \boldsymbol{U}[k] \in \mathcal{U} \end{aligned}$$

$$\begin{aligned} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} \quad & \tfrac{1}{2}\|\boldsymbol{U}_i[k]\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} \quad & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \\ & \boldsymbol{U}_i[k] \in \mathcal{U}_i \end{aligned}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^n_{\boldsymbol{\lambda}_i} \\ \vdots & \vdots \\ -P_i^{(2^{n_{\text{ineq}}}-1)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(2^{n_{\text{ineq}}}-1)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_{\text{ineq}}}-1}_{\boldsymbol{\lambda}_i} \end{cases}$$

Increasingly Sparse

For n_{ineq} constraints $\rightarrow 2^{n_{\text{ineq}}}$ permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^n_{\boldsymbol{\lambda}_i} \\ \quad \vdots & \quad \vdots \\ -P_i^{(2^{n_{\text{ineq}}}-1)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(2^{n_{\text{ineq}}}-1)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_{\text{ineq}}}-1}_{\boldsymbol{\lambda}_i} \end{cases}$$

Increasingly Sparse

For n_{ineq} constraints $\longrightarrow$ $2^{n_{\text{ineq}}}$ permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^n_{\boldsymbol{\lambda}_i} \\ \vdots & \vdots \\ -P_i^{\left(2^{n_\text{ineq}}-1\right)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{\left(2^{n_\text{ineq}}-1\right)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_\text{ineq}}-1}_{\boldsymbol{\lambda}_i} \end{cases}$$

Increasingly Sparse

For n_ineq constraints $\longrightarrow$ 2^{n_ineq} permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^n_{\boldsymbol{\lambda}_i} \\ \qquad\vdots & \qquad\vdots \\ -P_i^{\left(2^{n_\text{ineq}}-1\right)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{\left(2^{n_\text{ineq}}-1\right)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_\text{ineq}}-1}_{\boldsymbol{\lambda}_i} \end{cases} \Bigg\downarrow$$

Increasingly Sparse

For n_ineq constraints $\rightarrow 2^{n_\text{ineq}}$ permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^n_{\boldsymbol{\lambda}_i} \\ \vdots & \vdots \\ -P_i^{\left(2^{n_\text{ineq}}-1\right)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{\left(2^{n_\text{ineq}}-1\right)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_\text{ineq}}-1}_{\boldsymbol{\lambda}_i} \end{cases}$$

Increasingly Sparse

For n_ineq constraints $\rightarrow 2^{n_\text{ineq}}$ permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$

Instead of having

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

Now we have

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{n}_{\boldsymbol{\lambda}_i} \\ \quad\vdots & \quad\vdots \\ -P_i^{\left(2^{n_{\text{ineq}}}-1\right)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{\left(2^{n_{\text{ineq}}}-1\right)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^{2^{n_{\text{ineq}}}-1}_{\boldsymbol{\lambda}_i} \end{cases} \Bigg\downarrow$$

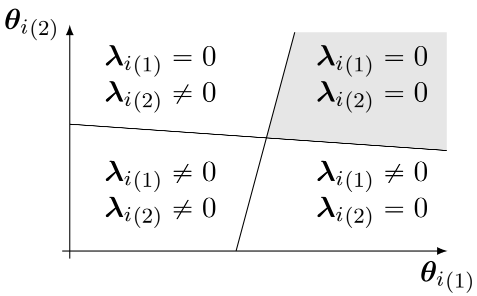
Increasingly Sparse

For n_{ineq} constraints $\rightarrow 2^{n_{\text{ineq}}}$ permutations

CentraleSupélec

Analyzing System

Solution for $\lambda_i[k]$ (Continued)



Two constraints partitioning $\boldsymbol{\theta}_i$ solution space.

Analyzing System

Negotiation

$$\mathrm{Proj}^{\mathbb{S}}(\boldsymbol{\theta}[k]^{(p)}+\rho^{(p)}\boldsymbol{\lambda}[k]^{(p)}) = \underset{\boldsymbol{x}}{\mathrm{argmin}} \left\{ \begin{array}{ll} \underset{\boldsymbol{U}[k]}{\text{minimize}} & \left\| \boldsymbol{x} - \boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)} \right\| \\ \text{subject to} & I_c^M \boldsymbol{x} \leq \boldsymbol{U}_{\max} : \boldsymbol{\mu} \end{array} \right\}$$

CentraleSupélec

Analyzing System

Negotiation (Continued)

$$\boldsymbol{\theta}[k]^{(p+1)} = \begin{cases} \boldsymbol{x}_0 + I_c^{M\,(0)} \left[-P_{\boldsymbol{\mu}}^{(0)} \boldsymbol{U}_{max} + \boldsymbol{s}_{\boldsymbol{\mu}}^{(0)}[k] \right], & \text{if } \boldsymbol{x}_0 \in \mathcal{R}_{\boldsymbol{\mu}}^{0} \\ \quad\vdots & \quad\vdots \\ \boldsymbol{x}_0, & \text{if } \boldsymbol{x}_0 \in \mathcal{R}_{\boldsymbol{\mu}}^{2^c-1} \end{cases}$$

$$\text{where } \boldsymbol{x}_0 = \boldsymbol{\theta}[k]^{(p)} + \rho^{(p)} \boldsymbol{\lambda}[k]^{(p)}$$

Analyzing System

Conclusion

$$\bar{\epsilon} = \overbrace{2^{c}}^{\text{regions in projection}} \times \underbrace{2^{n_{\text{ineq}}} \times \cdots \times 2^{n_{\text{ineq}}}}_{M\times\text{regions in each } \boldsymbol{\lambda}_i} = 2^{c+Mn_{\text{ineq}}}$$

Mitigation

- Ideal

$$\widehat{T_i[k]^{-1}} = \bar{P}_i \widehat{\widehat{P}_i[k]}^{-1}$$

- $P_i^{(0)}$ only invertible

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \widehat{\widehat{P}_i^{(0)}[k]}^{-1}$$

  - But how to force scarcity? Artificial Scarcity

Mitigation

- Ideal

$$\widehat{T_i[k]^{-1}} = \bar{P}_i \widehat{\widehat{P}_i[k]}^{-1}$$

- $P_i^{(0)}$ only invertible

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \widehat{\widehat{P}_i^{(0)}[k]}^{-1}$$

  - But how to force scarcity? Artificial Scarcity

Mitigation

- Ideal

$$\widehat{T_i[k]^{-1}} = \bar{P}_i \hat{\hat{P}}_i[k]^{-1}$$

- $P_i^{(0)}$ only invertible

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \hat{\hat{P}}_i^{(0)}[k]^{-1}$$

  - But how to force scarcity? Artificial Scarcity

Mitigation

- Ideal

$$\widehat{T_i[k]^{-1}} = \bar{P}_i \widehat{\hat{P}_i[k]}^{-1}$$

- $P_i^{(0)}$ only invertible

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \widehat{\hat{P}_i^{(0)}[k]}^{-1}$$

  - But how to force scarcity? Artificial Scarcity

CentraleSupélec

Mitigation

- Ideal

$$\widehat{T_i[k]^{-1}} = \bar{P}_i \widehat{\widehat{P}}_i[k]^{-1}$$

- $P_i^{(0)}$ only invertible

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \widehat{\widehat{P}}_i^{(0)}[k]^{-1}$$

  - But how to force scarcity? Artificial Scarcity

Artificial Scarcity

Who is it? Who is it?

- Since we control θ_i let's choose one where all constraints are active

Artificial Scarcity

Who is it? Who is it?

- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active

Artificial Scarcity

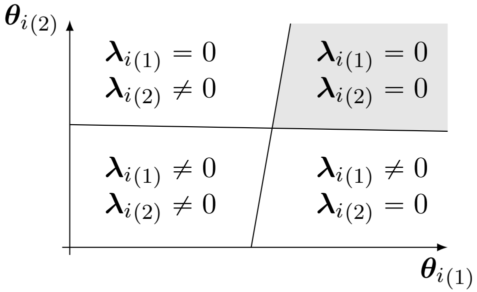
Who is it? Who is it?

- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active

Artificial Scarcity

Who is it? Who is it?

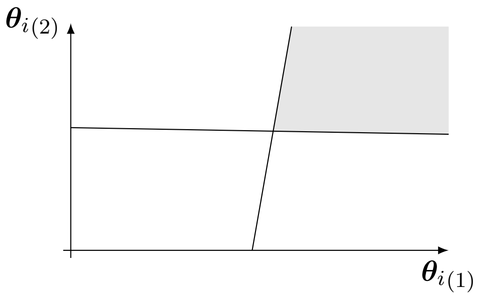
- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active



Artificial Scarcity

Who is it? Who is it?

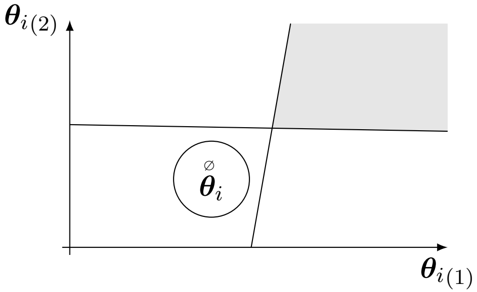
- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active



Artificial Scarcity

Who is it? Who is it?

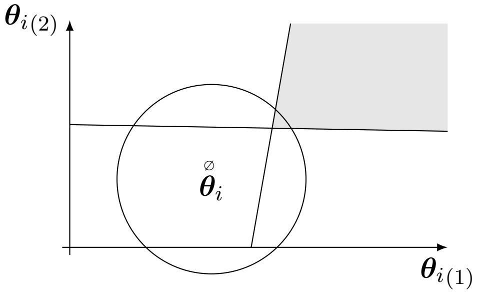
- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active



Artificial Scarcity

Who is it? Who is it?

- Since we control $\boldsymbol{\theta}_i$ let's choose one where all constraints are active



Expectation Maximization

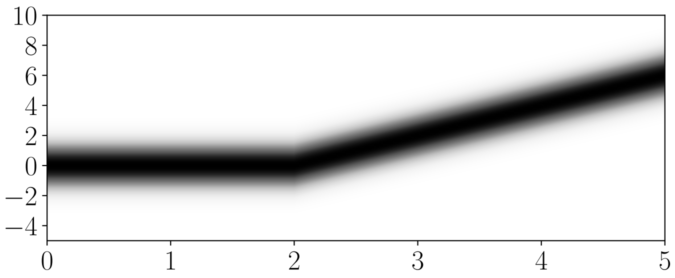


Figure 1: Gaussian Mixture for a 1D PWA function with 2 modes.

Expectation Maximization

Algorithm

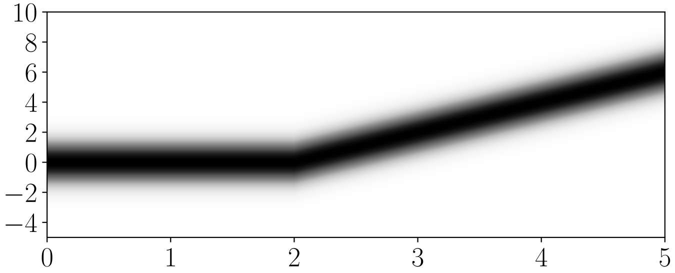


Figure 2: Gaussian Mixture for a 1D PWA function with 2 modes.

Detection

Same same

- Error $E_i^{(0)}[k] = \left\| \widehat{\bar{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F$
- Create threshold $\epsilon_{P_i^{(0)}}$
- Indicator $\eth_i \in \{0, 1\}$ detects the attack in agent i.
- $\eth_i^{(0)} = \mathbb{1}_{\{E_i^{(0)}[k] \geq \epsilon_{P_i^{(0)}}\}}$

So, let's detail the detection mechanism.

Detection

Same same

- Error $E_i^{(0)}[k] = \left\| \widehat{\widetilde{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F$
- Create threshold $\epsilon_{P_i^{(0)}}$
- Indicator $\eth_i \in \{0, 1\}$ detects the attack in agent i.
- $\eth_i^{(0)} = \mathbb{1}_{\{E_i^{(0)}[k] \geq \epsilon_{P_i^{(0)}}\}}$

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)

Detection

Same same

- Error $E_i^{(0)}[k] = \left\| \widehat{\tilde{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F$
- Create threshold $\epsilon_{P_i^{(0)}}$
- Indicator $\mathfrak{d}_i \in \{0, 1\}$ detects the attack in agent i.
- $\mathfrak{d}_i^{(0)} = \mathbb{1}_{\{E_i^{(0)}[k] \geqslant \epsilon_{P_i^{(0)}}\}}$

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p

Detection

Same same

- Error $E_i^{(0)}[k] = \left\| \widehat{\tilde{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F$
- Create threshold $\epsilon_{P_i^{(0)}}$
- Indicator $\mathfrak{d}_i \in \{0, 1\}$ detects the attack in agent i.
- $\mathfrak{d}_i^{(0)} = \mathbb{1}_{\{E_i^{(0)}[k] \geqslant \epsilon_{P_i^{(0)}}\}}$

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p
and finally we create an indicator d i for the attack of agent i

Detection

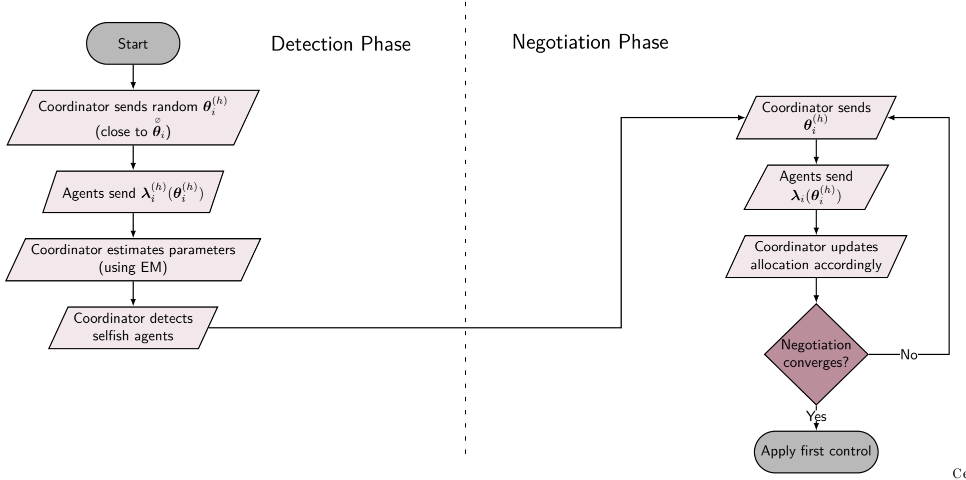
Same same

- Error $E_i^{(0)}[k] = \left\| \widehat{\tilde{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F$
- Create threshold $\epsilon_{P_i^{(0)}}$
- Indicator $\mathfrak{d}_i \in \{0, 1\}$ detects the attack in agent i.
- $\mathfrak{d}_i^{(0)} = \mathbb{1}_{\{E_i^{(0)}[k] \geqslant \epsilon_{P_i^{(0)}}\}}$

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p
and finally we create an indicator d i for the attack of agent i
It is equal to 1 if the error is greater than the threshold, indicating an attack or 0 otherwise

Complete algorithm

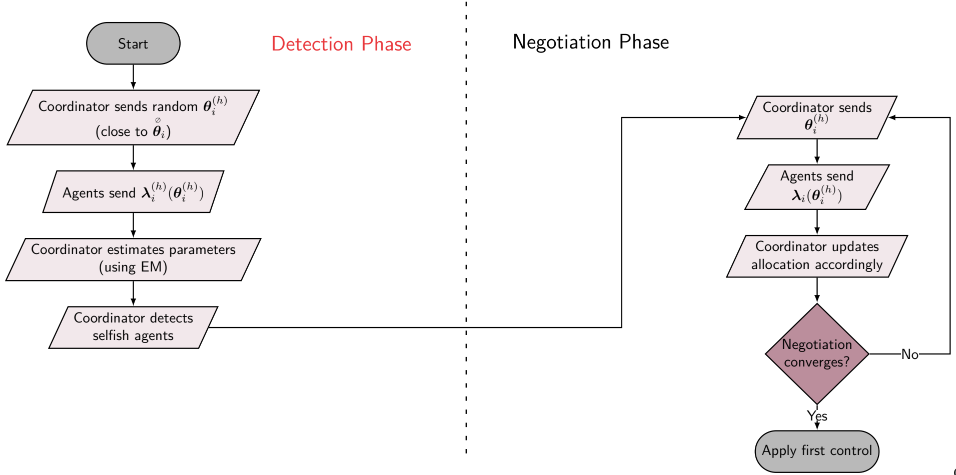
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases

Complete algorithm

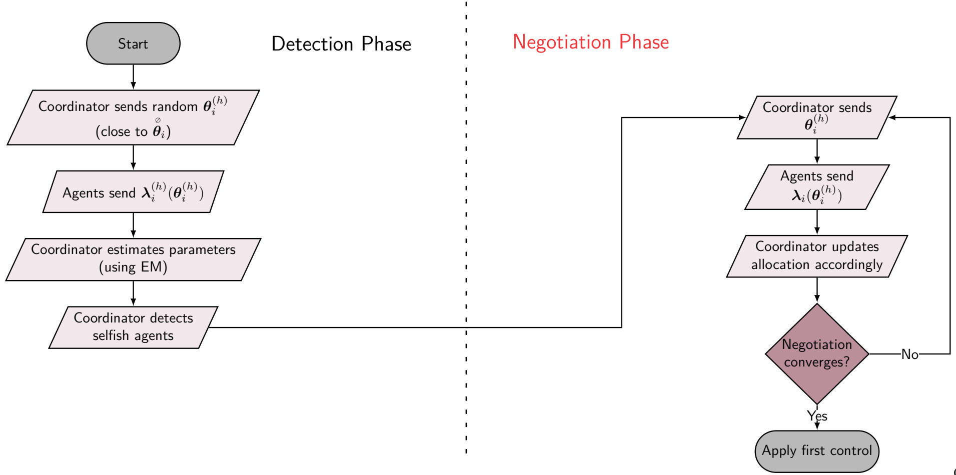
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase

Complete algorithm

RPdMPC-AS Refaire

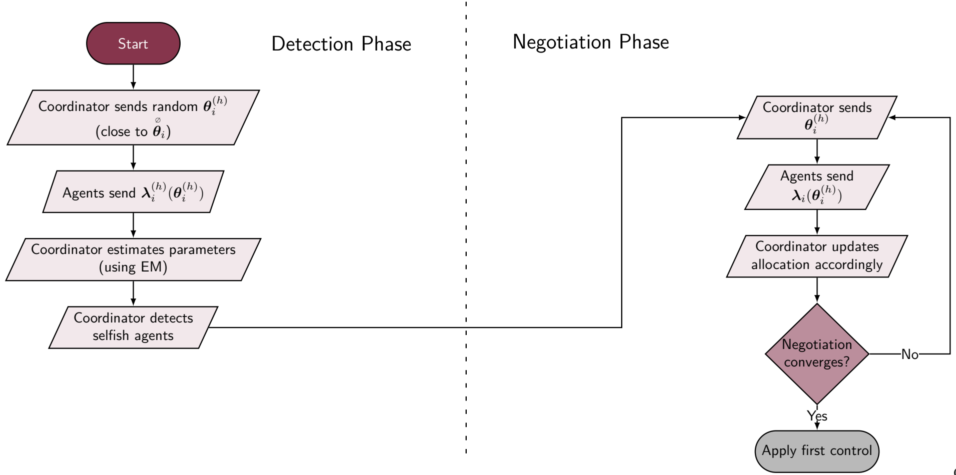


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

RPdMPC-AS Refaire

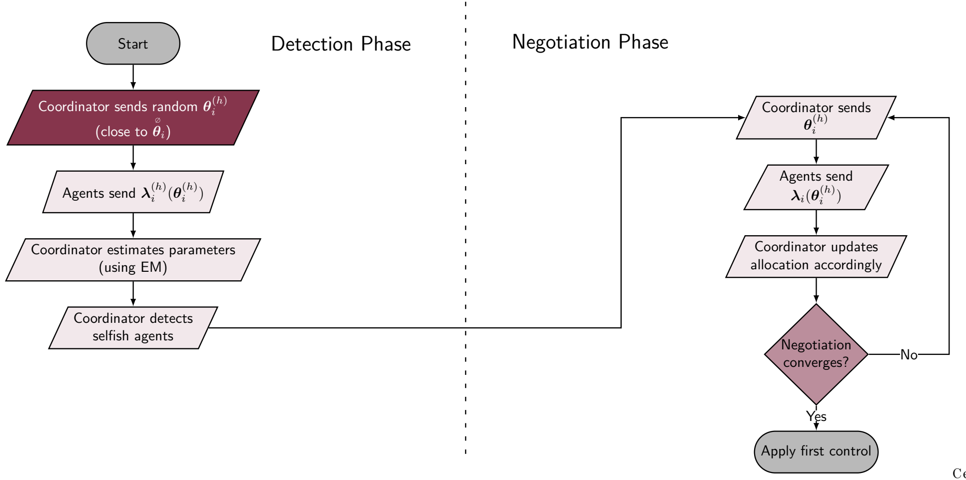


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

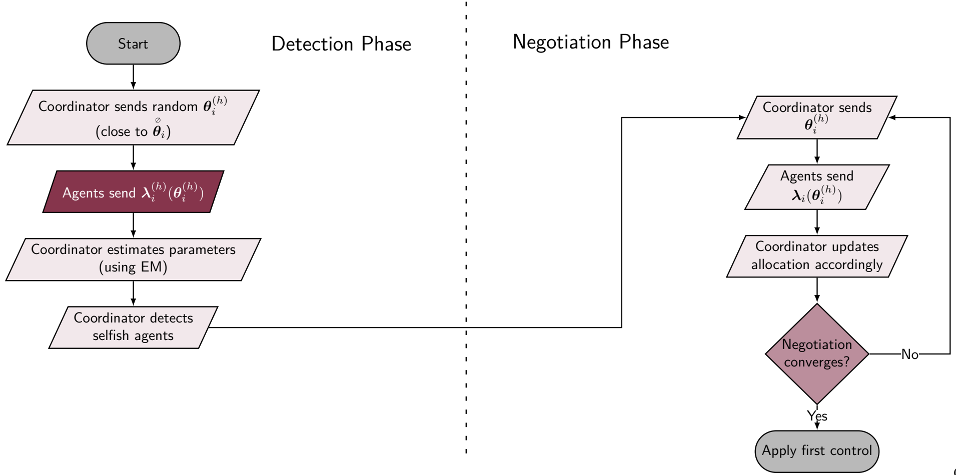
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i

Complete algorithm

RPdMPC-AS Refaire

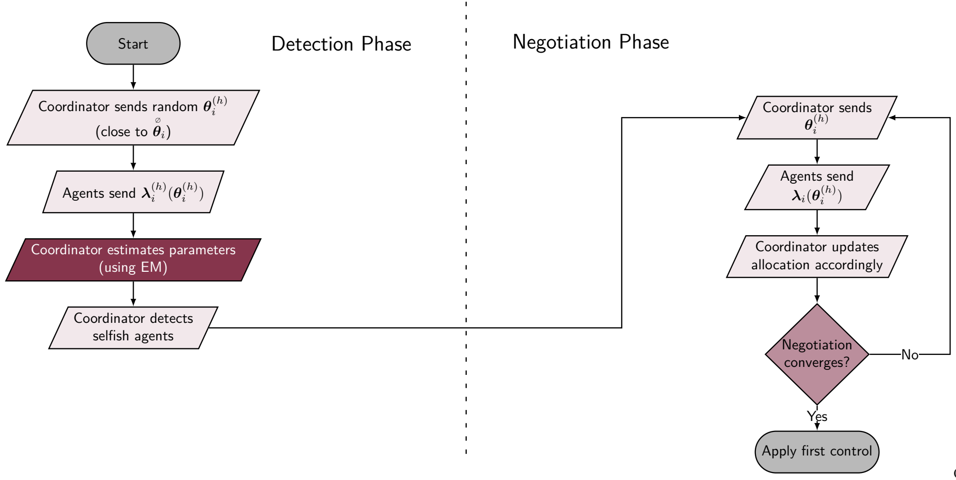


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i

Complete algorithm

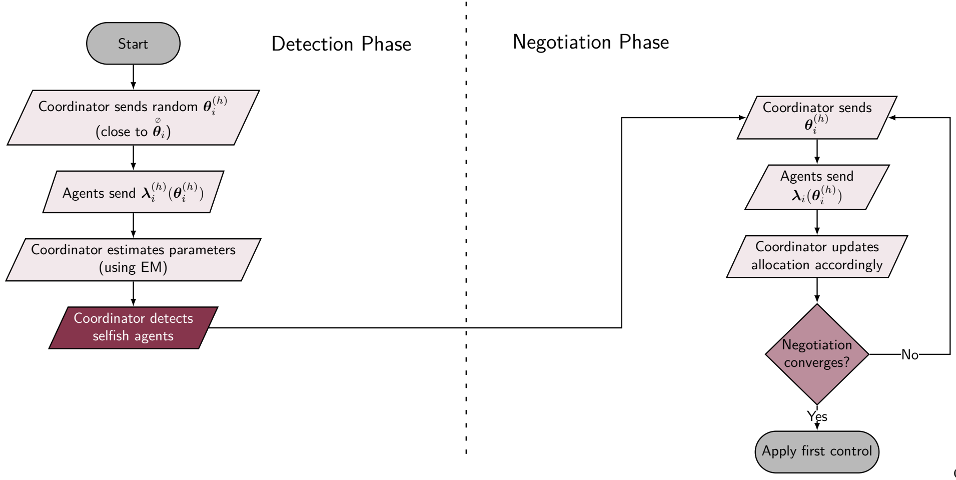
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde

Complete algorithm

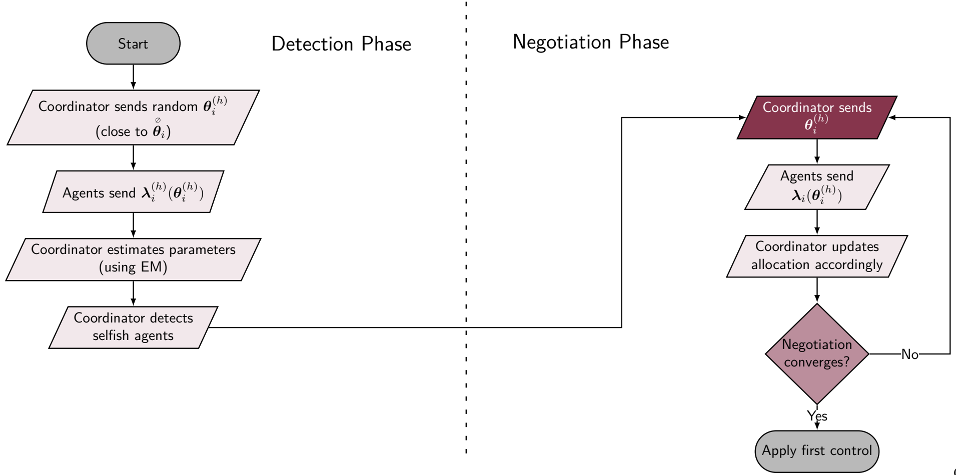
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge

Complete algorithm

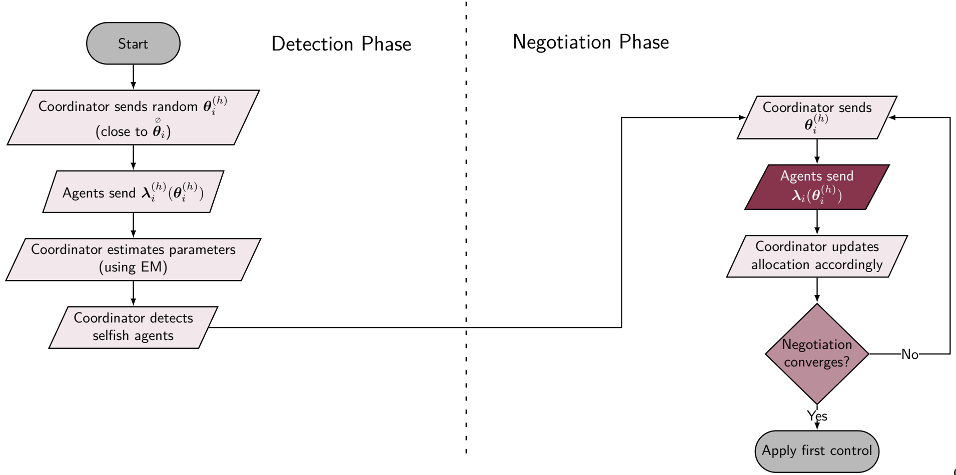
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative

Complete algorithm

RPdMPC-AS Refaire

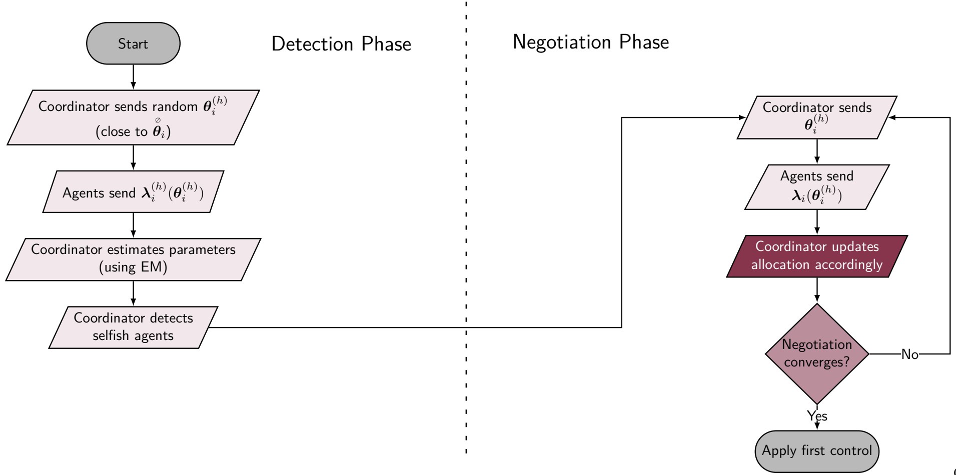


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i

Complete algorithm

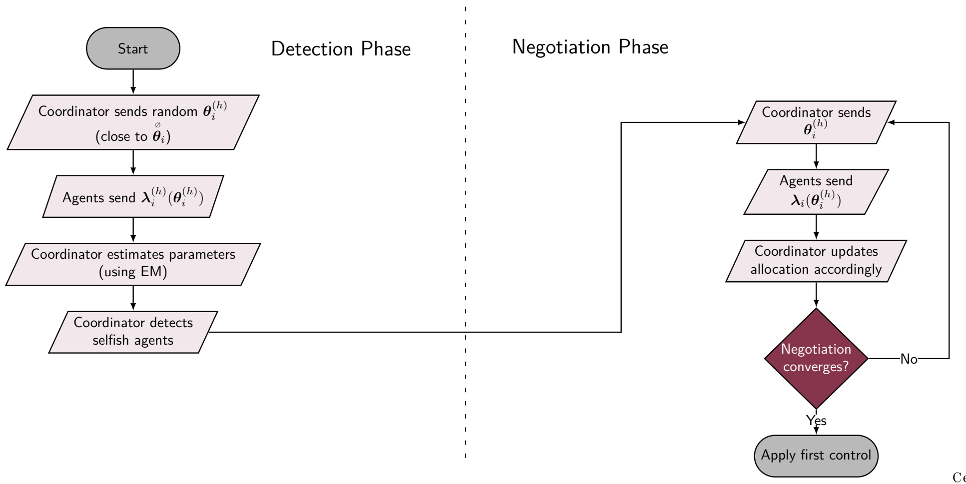
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i

Complete algorithm

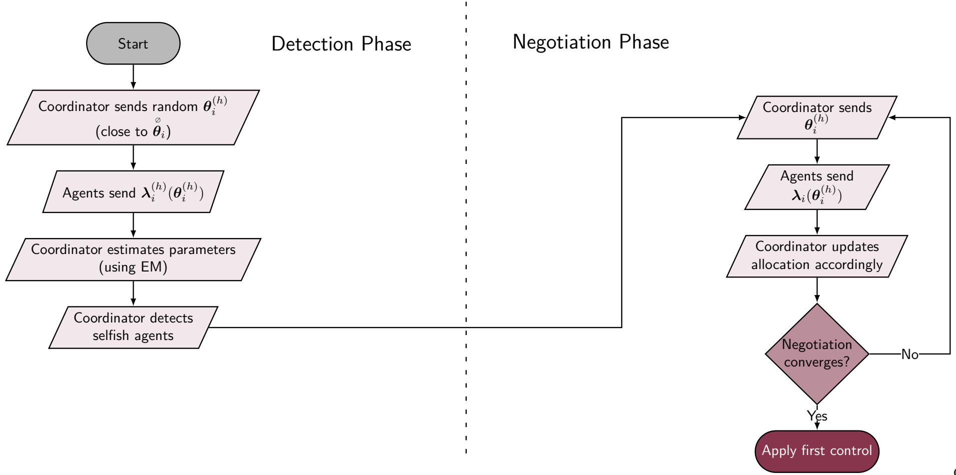
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent

Complete algorithm

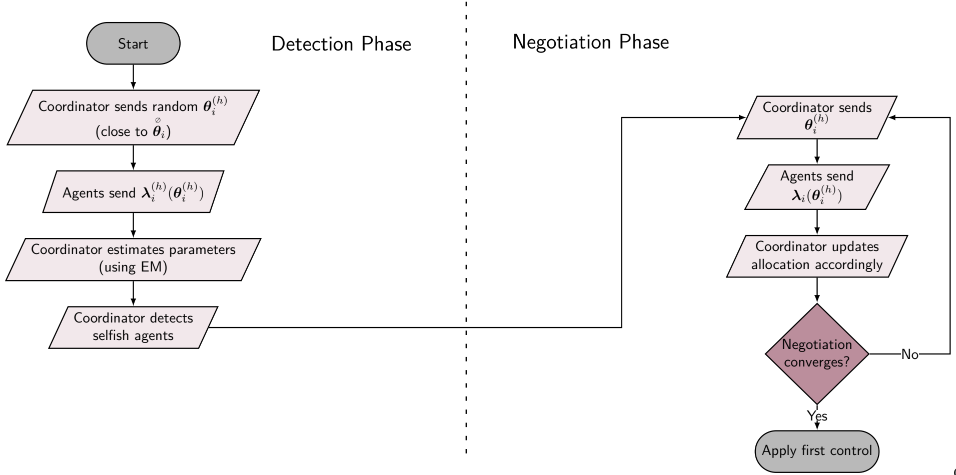
RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges

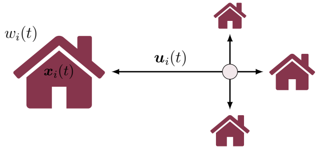
Complete algorithm

RPdMPC-AS Refaire



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Example



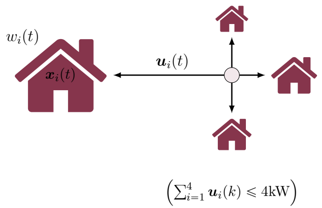
$\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\text{kW}\right)$

District Heating Network (4 Houses)

- Houses modeled using 3R-2C
- Not enough power
- Period of 5h ($T_s = 0.25\text{h}$)
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-AS)
- $T_I = \begin{bmatrix} 14.43288267 & 0. & 0. & 0. \\ 0. & 13.4590903 & 0. & 0. \\ 0. & 0. & 6.93065061 & 0. \\ 0. & 0. & 0. & 3.4447393 \end{bmatrix}$

CentraleSupélec

Example

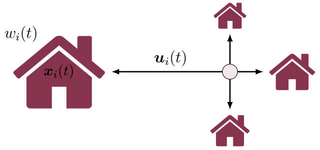


District Heating Network (4 Houses)

- Houses modeled using 3R-2C
- ~~Not enough power~~ (Change $(\boldsymbol{x}_0, \boldsymbol{w}_0)$)
- Period of 5h ($T_s = 0.25\text{h}$)
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-AS)
- $T_I = \begin{bmatrix} 14.43288267 & 0. & 0. & 0. \\ 0. & 13.4590903 & 0. & 0. \\ 0. & 0. & 6.93065061 & 0. \\ 0. & 0. & 0. & 3.4447393 \end{bmatrix}$

CentraleSupélec

Example



$\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\text{kW}\right)$

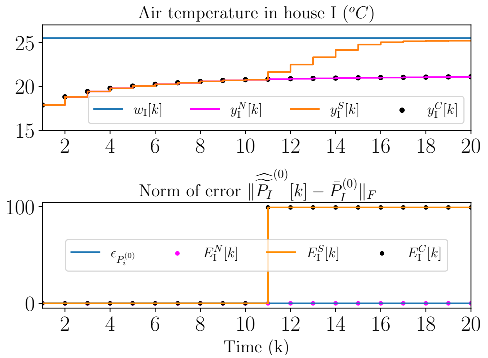
District Heating Network (4 Houses)

- Houses modeled using 3R-2C
- ~~Not enough power~~ (Change $(\boldsymbol{x}_0, \boldsymbol{w}_0)$)
- Period of 5h ($T_s = 0.25\text{h}$)
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-AS)
- $T_I = \begin{bmatrix} 14.43288267 & 0. & 0. & 0. \\ 0. & 13.4590903 & 0. & 0. \\ 0. & 0. & 6.93065061 & 0. \\ 0. & 0. & 0. & 3.4447393 \end{bmatrix}$

CentraleSupélec

Results

Temporal



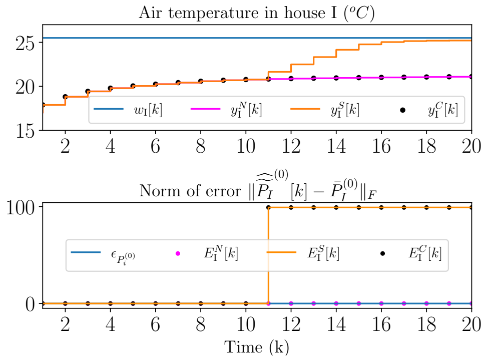
Temperature in house I and the variable $E_I(k)$ for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

CentraleSupélec

In the figure we see the air temperature and the estimation error for room 1

Results

Temporal



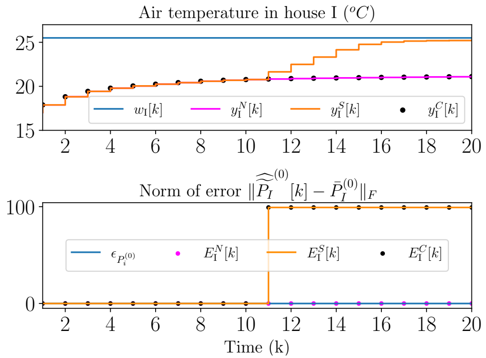
Temperature in house I and the variable $E_I(k)$ for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

CentraleSupélec

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue

Results

Temporal

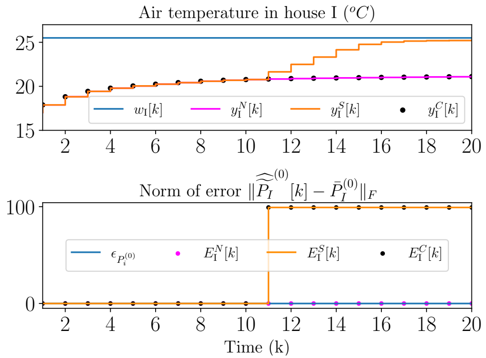


Temperature in house I and the variable $E_I(k)$ for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue
When the room presents selfish behavior (in blue), it reduces its cost and get closer to the setpoint, we see that the attack increases the error

Results

Temporal



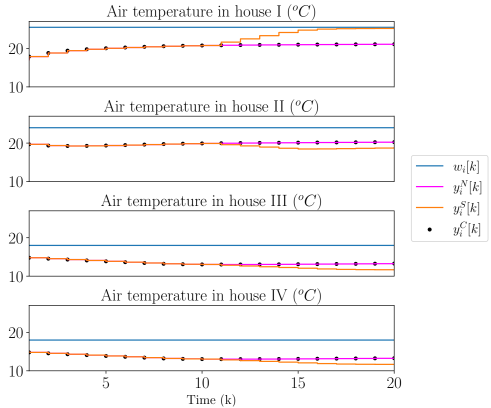
Temperature in house I and the variable $E_I(k)$ for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

CentraleSupélec

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue
When the room presents selfish behavior (in blue), it reduces its cost and get closer to the setpoint, we see that the attack increases the error
Now, for the case with correction (the red dots), even if it attacks the system, the temperature is close to its nominal value

Results

Temporal (Continued)

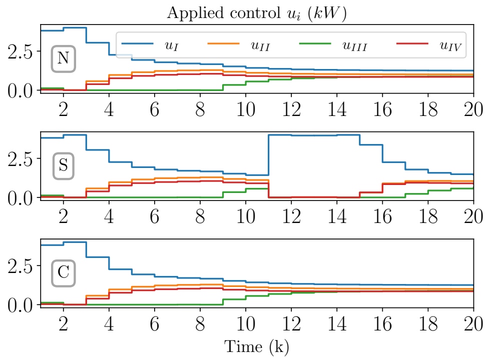


Air temperature in all houses for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

CentraleSupélec

Results

Control



Control applied in all houses for different scenarios.
N Nominal, S Selfish behavior, C Selfish + Correction

CentraleSupélec

Results

Costs

Objective functions J_i (% error).

Agent	Scenario N	Scenario S	Scenario C
I	19868.2	12618.5 (-36.5)	19868.2 (-0.0)
II	13784.5	18721.1 (35.8)	13784.5 (0.0)
III	17276.0	22324.9 (29.2)	17276.1 (0.0)
IV	10086.0	13872.4 (37.5)	10086.0 (0.0)
Global	61014.7	67536.9 (10.7)	61014.7 (-0.0)

Summary

- Vulnerabilities of Primal decomposition dMPC
- Resilient strategy for 2 kinds of systems
  - Deprived systems (where demand is greater than total resources)
  - Systems with possible artificial scarcity (sensible optimal demand)

Summary

- Vulnerabilities of Primal decomposition dMPC
- Resilient strategy for 2 kinds of systems
  - Deprived systems (where demand is greater than total resources)
  - Systems with possible artificial scarcity (sensible optimal demand)

CentraleSupélec

Summary

- Vulnerabilities of Primal decomposition dMPC
- Resilient strategy for 2 kinds of systems
  - Deprived systems (where demand is greater than total resources)
  - Systems with possible artificial scarcity (sensible optimal demand)

CentraleSupélec

Summary

- Vulnerabilities of Primal decomposition dMPC
- Resilient strategy for 2 kinds of systems
  - Deprived systems (where demand is greater than total resources)
  - Systems with possible artificial scarcity (sensible optimal demand)

CentraleSupélec

Summary

- Vulnerabilities of Primal decomposition dMPC
- Resilient strategy for 2 kinds of systems
    - Deprived systems (where demand is greater than total resources)
    - Systems with possible artificial scarcity (sensible optimal demand)

CentraleSupélec

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

CentraleSupélec

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

CentraleSupélec

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

CentraleSupélec

Future Directions

- Study of robustness + noise
- Partial reconstruction of cheating matrix
- Resilient strategy with soft constraints
- Recursive EM (or alternative)
- ...

Thank you!

Repository
https://github.com/Accacio/thesis

Contact
rafael.accacio.nogueira@gmail.com

If you want to see the simulations of this paper we have a github repository, and if you want to send me an email about this paper or this presentation you can flash the QR code in the right. Thank you!