Resource-Aware Network Optimization in Autonomous Energy Grids

Jorge Cortés

Mechanical and Aerospace Engineering
University of California, San Diego
http://carmenere.ucsd.edu/jorge

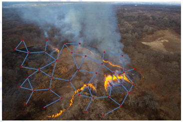
Virtual Workshop on Resilient Autonomous Energy Systems
National Renewable Energy Laboratory

September 8-9, 2021

Joint w/: Priyank Srivastava
Guido Cavraro

Complex Cyberphysical Systems

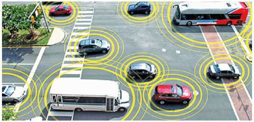
We live in an **interconnected** world



environmental sensing

industrial internet

energy grids



connected vehicles

IoT

Google Loon

Tight coupling between physical and cyber processes:

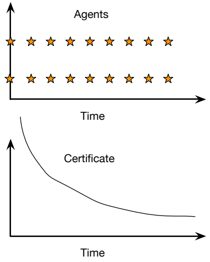
> *convergence of control, communication, computing, storage, sampling, signal processing, estimation, interaction with humans*

Need for efficient use of available resources

Resource-Aware Control and Coordination

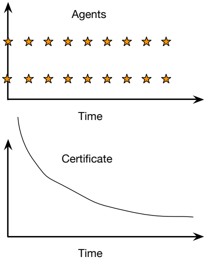
Continuous or **periodic** implementation paradigm

- costly-to-implement synchronization for information sharing, processing, decision making
- 'passive' asynchronism, fixed agent time schedules
- inefficient implementations for processor usage, communication bandwidth, energy



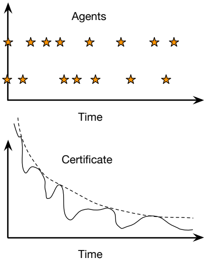
Resource-Aware Control and Coordination

Continuous or **periodic** implementation paradigm



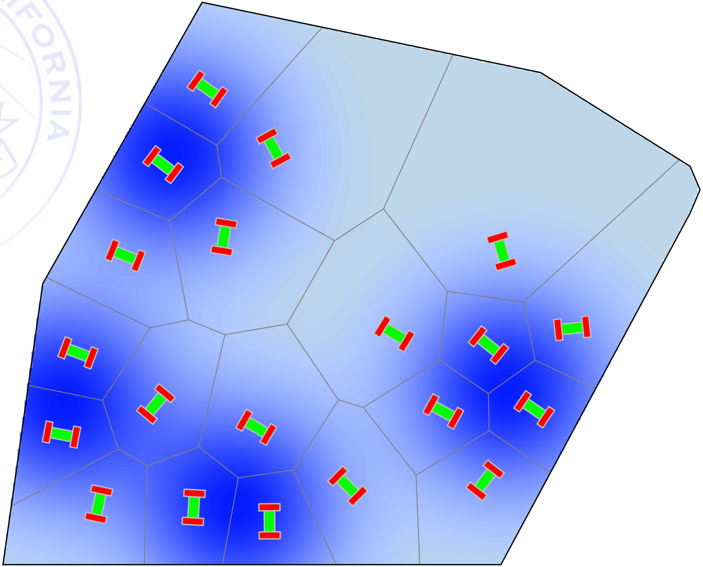
- costly-to-implement synchronization for information sharing, processing, decision making
- 'passive' asynchronism, fixed agent time schedules
- inefficient implementations for processor usage, communication bandwidth, energy

Opportunistic state-triggered paradigm



- trade-offs: comp, comm, sensing, control
- identify criteria to autonomously trigger actions based on task – 'active' asynchronism
- efficient implementations, incorporates uncertainty

A Movie is Worth a Thousand Words



How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

By ensuring $\dot{V} = \nabla V(x) \cdot f(x, k(\bar{x})) < 0$

How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x,u)$ on $\mathbb{R}^n$ with stabilization via

- controller: $k : \mathbb{R}^n \to \mathbb{R}^m$
- certificate: Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$

$$\left.\begin{array}{l} \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

$$\begin{aligned} \dot{V} &= \nabla V(x) \cdot f(x, k(\bar{x})) \\ &= \nabla V(x) \cdot f(x, k(x)) + \nabla V(x) \cdot (f(x, k(\bar{x})) - f(x, k(x))) \end{aligned}$$

How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x,u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

$$\begin{aligned} \dot{V} &= \nabla V(x) \cdot f(x, k(\bar{x})) \\ &\leq \nabla V(x) \cdot f(x, k(x)) + h(x) \|\bar{x} - x\| \leq 0 \end{aligned}$$

How to Decide When to Update?

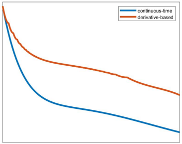
Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

- controller: $k : \mathbb{R}^n \to \mathbb{R}^m$
- certificate: Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$

$$\left.\begin{array}{l} k : \mathbb{R}^n \to \mathbb{R}^m \\ V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

Trigger criterium: $\|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x, k(x))}{h(x)}$



How to Decide When to Update?

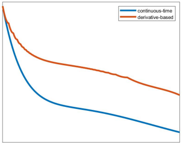
Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

- controller: $k : \mathbb{R}^n \to \mathbb{R}^m$
- certificate: Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$

$$\left.\begin{array}{l} \\ \\ \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

Trigger criterium: $\|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x, k(x))}{h(x)}$



Insights

- feasibility: rule out accumulation of trigger times

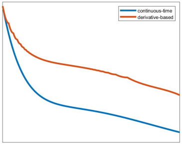
How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

$$\text{Trigger criterium: } \|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x, k(x))}{h(x)}$$



Insights

- feasibility: rule out accumulation of trigger times
- ensures stability, but how about guarantees on performance?

How to Decide When to Update?

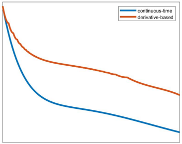
Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

- controller: $k : \mathbb{R}^n \to \mathbb{R}^m$
- certificate: Lyapunov function $V : \mathbb{R}^n \to \mathbb{R}$

$$\dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

Trigger criterium: $\|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x, k(x))}{h(x)}$



Insights

- feasibility: rule out accumulation of trigger times
- ensures stability, but how about guarantees on performance?
- trigger evaluation: computable with available information

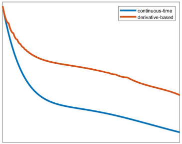
How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet \text{ controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet \text{ certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

Trigger criterium: $\|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x,k(x))}{h(x)}$



Insights

- feasibility: rule out accumulation of trigger times
- ensures stability, but how about guarantees on performance?
- trigger evaluation: computable with available information
- inherently aperiodic: active asynchronism

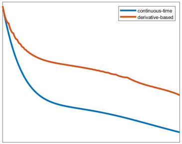
How to Decide When to Update?

Simplified setup: system $\dot{x} = f(x, u)$ on $\mathbb{R}^n$ with stabilization via

$$\left.\begin{array}{l} \bullet\ \text{controller: } k : \mathbb{R}^n \to \mathbb{R}^m \\ \bullet\ \text{certificate: Lyapunov function } V : \mathbb{R}^n \to \mathbb{R} \end{array}\right\} \quad \dot{V} = \nabla V(x) \cdot f(x, k(x)) < 0$$

Synthesis for $\dot{x} = f(x, k(\bar{x}))$? (w/ $\bar{x}$ constructed from sampled information of x)

Trigger criterium: $\|\bar{x} - x\| \leq \frac{-\nabla V(x) \cdot f(x, k(x))}{h(x)}$



Insights

- feasibility: rule out accumulation of trigger times
- ensures stability, but how about guarantees on performance?
- trigger evaluation: computable with available information
- inherently aperiodic: active asynchronism
- what is resource to be aware of?

What is the ‘Resource’ to Be Aware of?

Started off with actuator updates for stabilization, expanded to whole plant-sensor-actuator-controller model

P. Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic Control*, 52(9):1680–1685, 2007

W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada. An introduction to event-triggered and self-triggered control. In *IEEE Conf. on Decision and Control*, pages 3270–3285, Maui, HI, 2012

What is the ‘Resource’ to Be Aware of?

Started off with actuator updates for stabilization, expanded to whole plant-sensor-actuator-controller model

P. Tabuada. Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic Control*, 52(9):1680–1685, 2007

W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada. An introduction to event-triggered and self-triggered control. In *IEEE Conf. on Decision and Control*, pages 3270–3285, Maui, HI, 2012

Increasingly richer notions of **‘resource’**

- communication among individual agents to achieve collective task
- understanding stabilization under information constraints
- actuator updates for safety-critical systems
- costly recomputation of optimal policy
- re-sampling system state in accelerated optimization
- requesting information from a human

Today: Resource-Aware Network Optimization

Specific **challenges:**

- feasibility: emergence of Zeno behavior b/c of availability of partial information to agents
- trigger evaluation: solution of optimization problem is not known a priori!
- trigger evaluation: criterion for individual agents computable with locally available information

Agent-supervisor coordination strategy

> *decentralized, opportunistic, guarantees anytime feasibility and asymptotic convergence to optimizer*

Problem Formulation

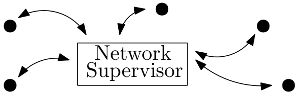
$$\min_{x \in \mathcal{X}} \quad \sum_{i=1}^{n} f_i(x_i) + g(x)$$

Problem data

- f_i: local cost function of agent i, depends on its own state
- g: coupling cost function, depends on network state
- $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$: separable constraint set

Problem Formulation

$$\min_{x \in \mathcal{X}} \underbrace{\sum_{i=1}^{n} f_i(x_i)}_{\text{agents}} + \underbrace{g(x)}_{\text{supervisor}}$$

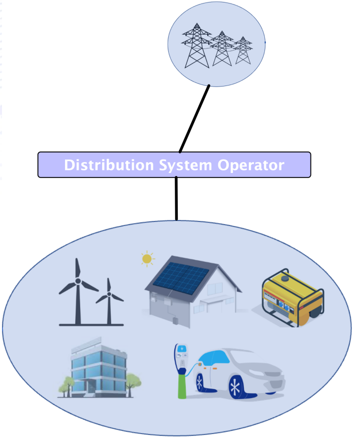


Problem data

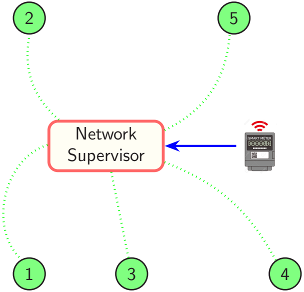
- f_i: local cost function of agent i, depends on its own state
- g: coupling cost function, depends on network state
- $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$: separable constraint set

Communication: agents rely on supervisor to obtain information about coupling cost

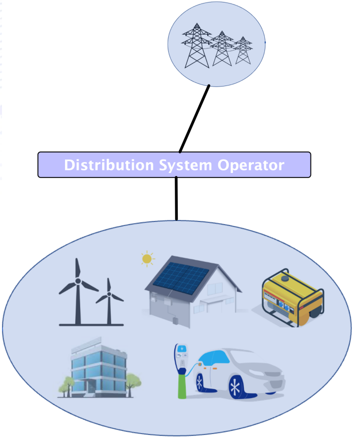
Prosumer-Based Distribution Network



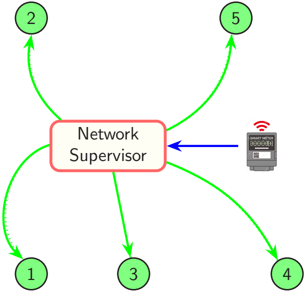
- prosumers: internal generation cost
- substation: power exchange cost w/ grid



Prosumer-Based Distribution Network



- prosumers: internal generation cost
- substation: power exchange cost w/ grid



Event-Triggered Agent-Supervisor Coordination

Network optimization

$$\min_{x \in \mathcal{X}} \quad \sum_{i=1}^{n} f_i(x_i) + g(x)$$

- Without coupling cost g, each agent simply would solve $\min_{x_i \in \mathcal{X}_i} f_i(x_i)$
- Presence of g couples agents' decisions, requiring continuous or periodic agent-supervisor communication

Goal: endow agents with locally evaluable criterion to opportunistically decide when to request information

Unconstrained Case

Gradient descent

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x)}_{\text{supervisor}}, \quad i \in \{1, \ldots, n\}$$

Unconstrained Case

Gradient descent

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x)}_{\text{supervisor}}, \quad i \in \{1, \ldots, n\}$$

Requires supervisor to provide information continuously!

Unconstrained Case

Gradient descent

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x)}_{\text{supervisor}}, \quad i \in \{1, \ldots, n\}$$

Requires supervisor to provide information continuously!

Opportunistic implementation

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x(t_k))}_{\text{supervisor}}$$

Requires supervisor to provide information at triggering times $\{t_k\}_{k=0}^{\infty}$

Unconstrained Case

Gradient descent

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x)}_{\text{supervisor}}, \quad i \in \{1, \ldots, n\}$$

Requires supervisor to provide information continuously!

Opportunistic implementation

$$\dot{x}_i = -\underbrace{\nabla_i f_i(x_i)}_{\text{agent}} - \underbrace{\nabla_i g(x(t_k))}_{\text{supervisor}}$$

Requires supervisor to provide information at triggering times $\{t_k\}_{k=0}^{\infty}$

How to determine $\{t_k\}_{k=0}^{\infty}$ in a decentralized fashion and ensure convergence?

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t_{k+1}^i = \min\{t > t_k \mid L_g|x_i - x_i(t_k)| = \sigma|\nabla_i f_i(x_i) + \nabla_i g(x(t_k))|\}$$

(L_g Lipschitz constant of ∇g and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1,\ldots,n\}} t_{k+1}^i$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \dots, n\}$ evaluates

$$t^i_{k+1} = \min \{ t > t_k \mid L_g |x_i - x_i(t_k)| = \sigma |\nabla_i f_i(x_i) + \nabla_i g(x(t_k))| \}$$

(L_g Lipschitz constant of ∇g and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1, \dots, n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \dots, n\}$

Objective Function is Motonically Decreasing

$V(x) = f(x) + g(x) - f(x^*) - g(x^*)$ is monotonically decreasing along $\cup_{k=0}^{\infty} [t_k, t_{k+1}]$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t^i_{k+1} = \min\{t > t_k \mid L_g |x_i - x_i(t_k)| = \sigma |\nabla_i f_i(x_i) + \nabla_i g(x(t_k))|\}$$

(L_g Lipschitz constant of ∇g and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1, \ldots, n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Objective Function is Motonically Decreasing

$V(x) = f(x) + g(x) - f(x^*) - g(x^*)$ is monotonically decreasing along $\cup_{k=0}^{\infty} [t_k, t_{k+1}]$

Proof follows from $\dot{V} \leq -\|\dot{x}\|^2 \left(1 - \frac{\|\nabla g(x) - \nabla g(x(t_k))\|}{\|\dot{x}\|}\right)$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t^i_{k+1} = \min \{ t > t_k \mid L_g |x_i - x_i(t_k)| = \sigma |\nabla_i f_i(x_i) + \nabla_i g(x(t_k))| \}$$

(L_g Lipschitz constant of ∇g and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1, \ldots, n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Minimum Inter-Event Time and Convergence to Optimizer

There exists $\tau > 0$ such that $t_{k+1} - t_k \geq \tau$ for all k, and any trajectory of opportunistic implementation converges to optimizer

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t^i_{k+1} = \min\{t > t_k \mid L_g|x_i - x_i(t_k)| = \sigma|\nabla_i f_i(x_i) + \nabla_i g(x(t_k))|\}$$

(L_g Lipschitz constant of ∇g and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1,\ldots,n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Minimum Inter-Event Time and Convergence to Optimizer

There exists $\tau > 0$ such that $t_{k+1} - t_k \geq \tau$ for all k, and any trajectory of opportunistic implementation converges to optimizer

Proof relies on lower-bounding time it takes state estimation error to reach triggering threshold

Constrained Case

Optimization via Continuous Projected Dynamics

Continuous projected dynamics

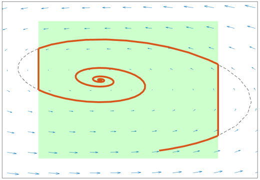
$$\dot{x} = \Pi_{\mathcal{X}}(x - \lambda(\nabla f(x) + \nabla g(x))) - x$$

where $\lambda > 0$ is design parameter

Continuous projected dynamics

$$\dot{x} = \Pi_{\mathcal{X}}(x - \lambda(\nabla f(x) + \nabla g(x))) - x$$

where $\lambda > 0$ is design parameter



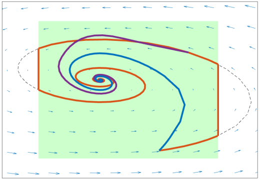
Constrained Case

Optimization via Continuous Projected Dynamics

Continuous projected dynamics

$$\dot{x} = \Pi_{\mathcal{X}}(x - \lambda(\nabla f(x) + \nabla g(x))) - x$$

where $\lambda > 0$ is design parameter



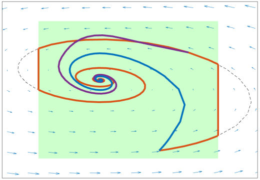
Constrained Case

Optimization via Continuous Projected Dynamics

Continuous projected dynamics

$$\dot{x} = \Pi_{\mathcal{X}}(x - \lambda(\nabla f(x) + \nabla g(x))) - x$$

where $\lambda > 0$ is design parameter



If $f + g$ has Lipschitz gradient,

- feasible set is forward invariant and attractive
- dynamics converges to optimizer

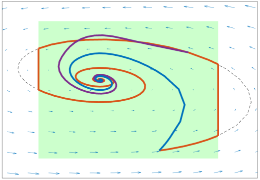
Constrained Case

Optimization via Continuous Projected Dynamics

Continuous projected dynamics

$$\dot{x} = \Pi_{\mathcal{X}}(x - \lambda(\nabla f(x) + \nabla g(x))) - x$$

where $\lambda > 0$ is design parameter



If $f + g$ has Lipschitz gradient,

- feasible set is forward invariant and attractive
- dynamics converges to optimizer

Opportunistic implementation

$$\dot{x}_i = \Pi_{\mathcal{X}_i}(x_i - \lambda(\nabla_{x_i} f_i(x_i) + \underbrace{\nabla_{x_i} g(x(t_k))}_{\text{supervisor}})) - x_i$$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \dots, n\}$ evaluates

$$t^i_{k+1} = \min \left\{ t > t_k \mid \lambda \bar{L}_g |x_i - x_i^k| = \sigma |\Pi_{\mathcal{X}_i}(x_i - \lambda(\nabla_{x_i} f_i(x_i) + \nabla_{x_i} g(x(t_k)))) - x_i| \right\}$$

($\bar{L}_g$ Lipschitz constant of ∇g over $\mathcal{X}$ and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1, \dots, n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \dots, n\}$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t_{k+1}^i = \min \left\{ t > t_k \mid \lambda \bar{L}_g |x_i - x_i^k| = \sigma |\Pi_{\mathcal{X}_i}(x_i - \lambda(\nabla_{x_i} f_i(x_i) + \nabla_{x_i} g(x(t_k)))) - x_i| \right\}$$

($\bar{L}_g$ Lipschitz constant of ∇g over $\mathcal{X}$ and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1,\ldots,n\}} t_{k+1}^i$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Objective Function is Motonically Decreasing

$V(x) = f(x) + g(x) - f(x^*) - g(x^*)$ is monotonically decreasing along $\cup_{k=0}^{\infty} [t_k, t_{k+1}]$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \ldots, n\}$ evaluates

$$t^i_{k+1} = \min \left\{ t > t_k \mid \lambda \bar{L}_g |x_i - x_i^k| = \sigma |\Pi_{\mathcal{X}_i}(x_i - \lambda(\nabla_{x_i} f_i(x_i) + \nabla_{x_i} g(x(t_k)))) - x_i| \right\}$$

($\bar{L}_g$ Lipschitz constant of ∇g over $\mathcal{X}$ and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1,\ldots,n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \ldots, n\}$

Minimum Inter-Event Time and Convergence to Optimizer

For $\lambda < 1/H$, there exists $\tau > 0$ such that $t_{k+1} - t_k \geq \tau$ for all k, and any trajectory of opportunistic implementation converges to optimizer

$H = \max_{i \in \{1,\ldots,n\}} \max_{x_i \in \mathcal{X}_i} \nabla^2_{x_i} f_i(x_i)$

Decentralized Event-Triggered Coordination

1. each agent $i \in \{1, \dots, n\}$ evaluates

$$t^i_{k+1} = \min \left\{ t > t_k \mid \lambda \bar{L}_g |x_i - x_i^k| = \sigma |\Pi_{\mathcal{X}_i}(x_i - \lambda(\nabla_{x_i} f_i(x_i) + \nabla_{x_i} g(x(t_k)))) - x_i| \right\}$$

($\bar{L}_g$ Lipschitz constant of ∇g over $\mathcal{X}$ and $\sigma \in (0, 1)$ design parameter)

2. whoever finds smallest time, determines triggering time

$$t_{k+1} = \min_{i \in \{1, \dots, n\}} t^i_{k+1}$$

3. supervisor provides information to compute $\nabla_i g(x(t_{k+1}))$ to each $i \in \{1, \dots, n\}$

Minimum Inter-Event Time and Convergence to Optimizer

For $\lambda < 1/H$, there exists $\tau > 0$ such that $t_{k+1} - t_k \geq \tau$ for all k, and any trajectory of opportunistic implementation converges to optimizer

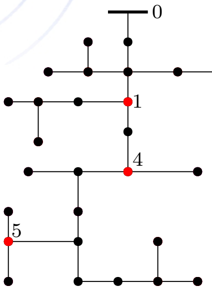
$H = \max_{i \in \{1, \dots, n\}} \max_{x_i \in \mathcal{X}_i} \nabla^2_{x_i} f_i(x_i)$

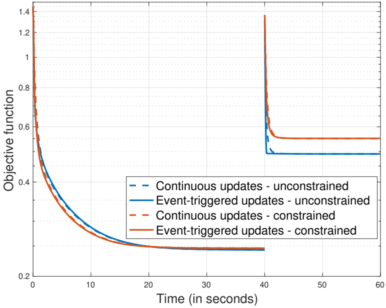
Proof uses nonsmooth analysis to lower-bound time it takes state estimation error to reach threshold

Simulations on IEEE 37-Bus Test Feeder

Minimizing generation cost in prosumer-based distribution network

$$\min_{x \in \mathcal{X}} \quad \sum_{i=1}^{5} f_i(x_i) + f_0(-\mathbf{1}^\top x + c)$$



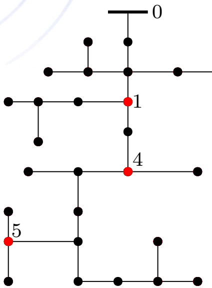


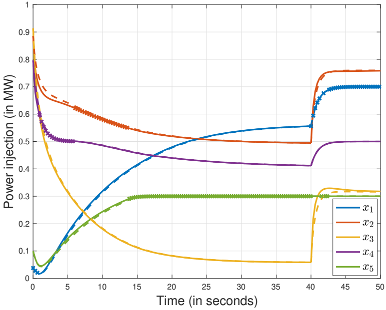
red nodes are generators, node 0 is supervisor

Simulations on IEEE 37-Bus Test Feeder

Minimizing generation cost in prosumer-based distribution network

$$\min_{x \in \mathcal{X}} \quad \sum_{i=1}^{5} f_i(x_i) + f_0(-\mathbf{1}^\top x + c)$$





red nodes are generators, node 0 is supervisor

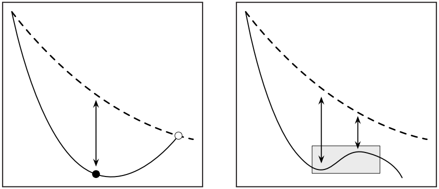
Summary

Resource-aware control and coordination

rich paradigm for real-time implementation of cyberphysical systems

Outlook

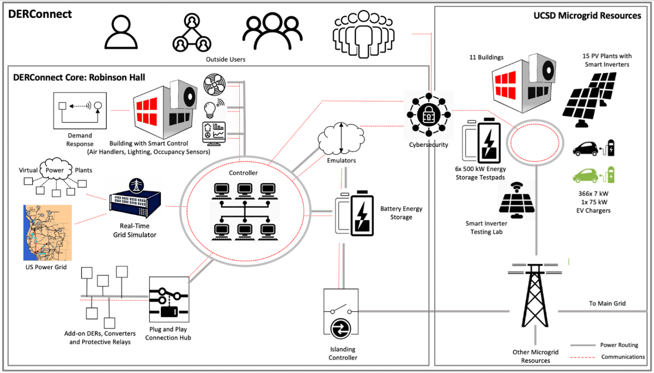
- trigger design: performance vs efficiency vs implementability
- trigger evaluation: synthesis of distributed triggers, adaptive budgets
- distributed asynchronism
- resource understood broadly
- elaborate uses of sampling information



P. Srivastava, G. Cavraro, and J. Cortés. Agent-supervisor coordination for decentralized event-triggered optimization. *IEEE Control Systems Letters*, 2021. Submitted

DERConnect

NSF User Facility for Control of Distributed Energy Resources



Testbed for distributed controls

- 2,5K controlled devices
- 30K metered devices
- 2M simulated nodes
- remote access nationally

Beyond: cybersecurity, building operations, human-cyber-physical systems

Help Us Shape It

Call for **feedback** from research community:

- types of tests you envision?
- capabilities in terms of communication, sensing, control, and actuation?
- ability to determine test conditions&parameters, templates, tools, libraries
- resolution, granularity, data access and availability during&after tests

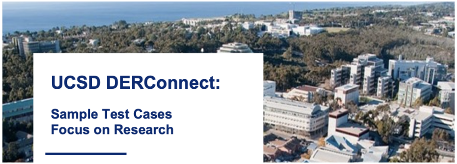
Help Us Shape It

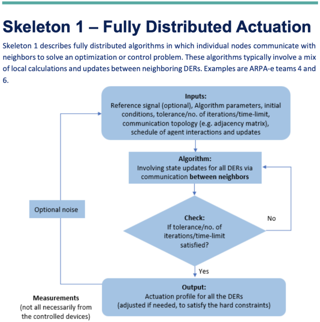
Call for **feedback** from research community:

- types of tests you envision?
- capabilities in terms of communication, sensing, control, and actuation?
- ability to determine test conditions&parameters, templates, tools, libraries
- resolution, granularity, data access and availability during&after tests

Opportunity to shape capabilities&functionality of DERConnect

Current **live document** outlines 4 high-level skeletons of envisioned test cases




Skeleton 1 – Fully Distributed Actuation

Skeleton 1 describes fully distributed algorithms in which individual nodes communicate with neighbors to solve an optimization or control problem. These algorithms typically involve a mix of local calculations and updates between neighboring DERs. Examples are ARPA-e teams 4 and 6.

https://sites.google.com/ucsd.edu/derconnect