Constraint-exchange methods for distributed optimization in asynchronous networks

Giuseppe Notarstefano

Control Optimization and Robotics group
Università del Salento, Lecce (Italy)
giuseppe.notarstefano@unisalento.it

http://cor.unisalento.it/notarstefano

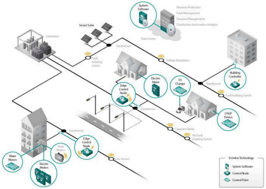
Joint work with M. Bürger, F. Allgöwer (Univ. Stuttgart), F. Bullo (UCSB)

Why distributed optimization?

Cyber-physical network systems are revolutionizing everyday life

smart communicating devices
embedded computation
sensing/control

- smart phones, watches, glasses, ...
- car-2-x systems
- smart grids
- mobile robots
- ...

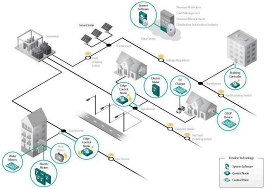


Why distributed optimization?

Cyber-physical network systems are revolutionizing everyday life

smart communicating devices
embedded computation
sensing/control

- smart phones, watches, glasses, ...
- car-2-x systems
- smart grids
- mobile robots
- ...



Estimation, Learning, Decision & Control problems are often optimization problems!

Available methods (incomplete review)

- *Sub-gradient methods*

  [Tsitsiklis & al., TAC '86], [Nedic & Ozdaglar, TAC '09], [Johansson & al, SJO '09], [Nedic & al., TAC '10], ...

- *Dual decomposition & ADMM*

  [Low & Lapsley, '99], [Schizas & al. '08], [Boyd & al. '10], [Wei & Ozdaglar, CDC '12] ...

- *Newton-like methods*

  [Zargham & al., ACC '11], [Zanella & al., CDC '11], ...

- *Constraint exchange methods*

  [Notarstefano & Bullo, TAC '11], [Bürger & al., Automatica '12], ...

  [Bürger, Notarstefano & Allgöwer, TAC 14]

Available methods (incomplete review)

- *Sub-gradient methods*

  [Tsitsiklis & al., TAC '86], [Nedic & Ozdaglar, TAC '09], [Johansson & al, SJO '09], [Nedic & al., TAC '10], ...

- *Dual decomposition & ADMM*

  [Low & Lapsley, '99], [Schizas & al. '08], [Boyd & al. '10], [Wei & Ozdaglar, CDC '12] ...

- *Newton-like methods*

  [Zargham & al., ACC '11], [Zanella & al., CDC '11], ...

- *Constraint exchange methods*

  [Notarstefano & Bullo, TAC '11], [Bürger & al., Automatica '12], ...

  [Bürger, Notarstefano & Allgöwer, TAC 14]

Available methods (incomplete review)

- *Sub-gradient methods*

  [Tsitsiklis & al., TAC '86], [Nedic & Ozdaglar, TAC '09], [Johansson & al, SJO '09], [Nedic & al., TAC '10], ...

- *Dual decomposition & ADMM*

  [Low & Lapsley, '99], [Schizas & al. '08], [Boyd & al. '10], [Wei & Ozdaglar, CDC '12] ...

- *Newton-like methods*

  [Zargham & al., ACC '11], [Zanella & al., CDC '11], ...

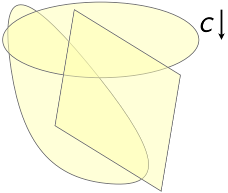
- *Constraint exchange methods*

  [Notarstefano & Bullo, TAC '11], [Bürger & al., Automatica '12], ...
  [Bürger, Notarstefano & Allgöwer, TAC 14] *Today's talk!*

Distributed optimization set-up

General Convex Optimization Problem:

$$\max c^T z$$
$$z \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$

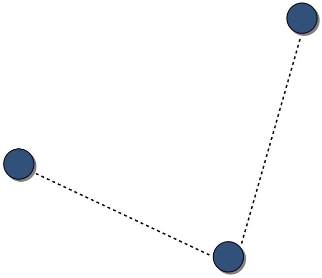


- $\mathcal{Z}_i \subset \mathbb{R}^d$ are *general convex sets*.
- Linear objective function is *most general!*

Distributed optimization set-up

General Convex Optimization Problem:

$$\max c^T z$$
$$z \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$



- $\mathcal{Z}_i \subset \mathbb{R}^d$ are *general convex sets*.
- Linear objective function is *most general!*

Network of n Agents (processors, spatially distributed)

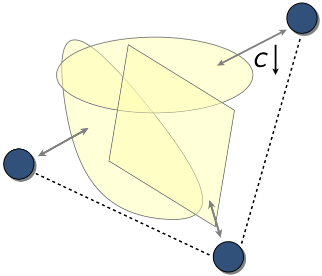
- Nodes have *local* memory, computation and communication.
- Communication may be *asynchronous* and *directed*.

Distributed optimization set-up

General Convex Optimization Problem:

$$\max c^T z$$

$$z \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$



- $\mathcal{Z}_i \subset \mathbb{R}^d$ are *general convex sets*.
- Linear objective function is *most general!*

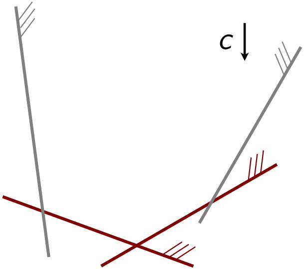
Network of n Agents (processors, spatially distributed)

- Each agent i *knows c and $\mathcal{Z}_i$*.
- *Goal:* agree on a global minimizer.

Distributed linear program

Standard LP: d variables & n constraints

$$\max c^T z$$
$$a_i^T z \leq b_i \quad i \in \{1, \ldots, n\}$$

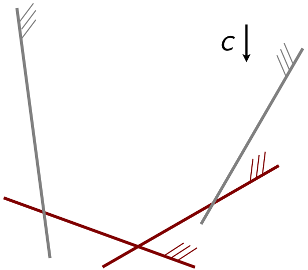


- set of linear constraints (halfspace) $\mathcal{Z}_i = \{z \in \mathbb{R}^d \mid a_i^T z \leq b_i\}$
- solution uniquely determined by at most d *active constraints*

Distributed linear program

Standard LP: d variables & n constraints

$$\max c^T z$$
$$a_i^T z \le b_i \quad i \in \{1, \ldots, n\}$$



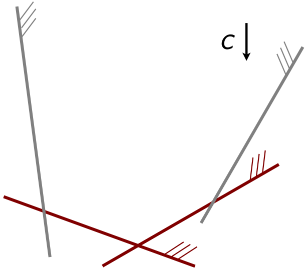
- set of linear constraints (halfspace) $\mathcal{Z}_i = \{z \in \mathbb{R}^d \mid a_i^T z \le b_i\}$
- solution uniquely determined by at most d *active constraints*

Basis: minimal set of active constraints, B, $|B| \le d$.

Distributed linear program

Standard LP: d variables & n constraints

$$\max c^T z$$
$$a_i^T z \leq b_i \quad i \in \{1, \ldots, n\}$$



- set of linear constraints (halfspace) $\mathcal{Z}_i = \{z \in \mathbb{R}^d \mid a_i^T z \leq b_i\}$
- solution uniquely determined by at most d *active constraints*

GOAL

design a distributed algorithm
such that the nodes agree on the optimal basis

Constraints Consensus algorithm for LP

processor state: a set of constraints $B^{[i]}$ — initialized $B^{[i]} := \emptyset$

message generation: transmit the set of constraints $B^{[i]}$

state update rule:

1. collect all constraints

$$H_{tmp}^{[i]} := B^{[i]} \cup \left(\cup_{\text{for all in-neighbors } j} B^{[j]} \right) \cup \{ \, (a_i, b_i) \, \}$$

2. solve local LP

$$\begin{aligned} \max \; & c^T z \\ & a_k^T z \leq b_k \qquad \text{for all } (a_k, b_k) \in H_{tmp}^{[i]} \end{aligned}$$

3. update $B^{[i]}$ as *basis* (active constraints) of local LP

Distributed convex programming

General Convex Optimization Problem:

$$\max c^T z$$
$$z \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$

Distributed convex programming

General Convex Optimization Problem:

$$\max c^T z$$
$$z \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$

Constraints consensus works for general convex programs!

Constraints Consensus algorithm

processor state: a set of constraints $B^{[i]}$ — initialized $B^{[i]} := \emptyset$

message generation: transmit the set of constraints $B^{[i]}$

state update rule:

1. collect all constraints

$$H_{tmp}^{[i]} := B^{[i]} \cup \left(\cup_{\text{for all in-neighbor } j} B^{[j]} \right) \cup \{\mathcal{Z}_i\}$$

2. solve local *Convex Program (CP)*

$$\max c^T z$$
$$z \in \bigcap_{\{\mathcal{Z}_j\} \subset H_{tmp}^{[i]}} \mathcal{Z}_j$$

3. update $B^{[i]}$ as *basis* (active constraints) of local CP

End of the story?

Constraints consensus works for a larger class of problems defined in an abstract framework: *abstract programs or LP-type problems*

G. Notarstefano and F. Bullo. Distributed Abstract Optimization via Constraints Consensus: Theory and Applications. IEEE TAC, 56(10):2247-2261, Oct 2011.

End of the story?

Constraints consensus works for a larger class of problems defined in an abstract framework: *abstract programs or LP-type problems*

G. Notarstefano and F. Bullo. Distributed Abstract Optimization via Constraints Consensus: Theory and Applications. IEEE TAC, 56(10):2247-2261, Oct 2011.

Problem: there can be optimization problems in which $\mathcal{Z}_i$

- is not defined analytically
- contains infinitely many constraints (semi-infinite programming).

End of the story?

Constraints consensus works for a larger class of problems defined in an abstract framework: *abstract programs or LP-type problems*

G. Notarstefano and F. Bullo. Distributed Abstract Optimization via Constraints Consensus: Theory and Applications. IEEE TAC, 56(10):2247-2261, Oct 2011.

Problem: there can be optimization problems in which $\mathcal{Z}_i$

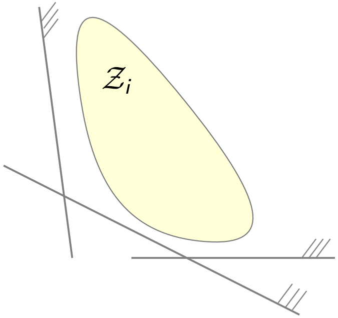
- is not defined analytically
- contains infinitely many constraints (semi-infinite programming).

What does it mean to exchange $\mathcal{Z}_i$?

Main Idea: Polyhedral Approximation

Polyhedral Approximation of $\mathcal{Z}_i$:

- approximate with a
  set of half-spaces,
- refine the approximation iteratively.
- advantage:
  approximation easy to obtain!



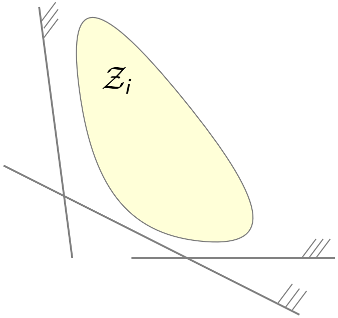
Algorithm main feature

Generate half-planes to iteratively *outer-approximate* optimal point

Main Idea: Polyhedral Approximation

Polyhedral Approximation of $\mathcal{Z}_i$:

- approximate with a
  set of half-spaces,
- refine the approximation iteratively.
- advantage:
  approximation easy to obtain!



Algorithm main feature

Generate half-planes to iteratively *outer-approximate* optimal point

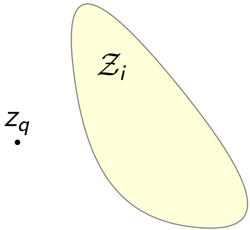
Node capability: Cutting-Plane Oracle

Cutting-Plane Oracle $\mathrm{ORC}(z_q, \mathcal{Z}_i)$:

Given a *query point z_q*,

- either assert $z_q \in \mathcal{Z}_i$, or
- return a separating hyperplane (cutting-plane)

$$a(z_q)^T z \leq b(z_q),\ \forall z \in \mathcal{Z}_i \quad \text{and} \quad a(z_q)^T z_q - b(z_q) = s(z_q) > 0.$$



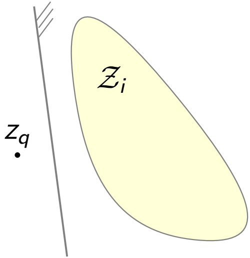
Assumption: $z_q \to \bar{z}$ and $s(z_q) \to 0$ implies $\bar{z} \in \mathcal{Z}_i$.

Node capability: Cutting-Plane Oracle

Cutting-Plane Oracle $\mathrm{ORC}(z_q, \mathcal{Z}_i)$:

Given a *query point z_q*,

- either assert $z_q \in \mathcal{Z}_i$, or
- return a separating hyperplane (cutting-plane)



$$a(z_q)^T z \le b(z_q),\ \forall z \in \mathcal{Z}_i \quad \text{and} \quad a(z_q)^T z_q - b(z_q) = s(z_q) > 0.$$

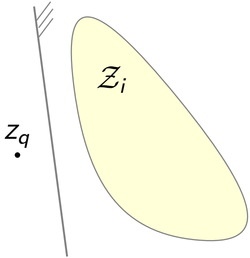
Assumption: $z_q \to \bar{z}$ and $s(z_q) \to 0$ implies $\bar{z} \in \mathcal{Z}_i$.

Node capability: Cutting-Plane Oracle

Cutting-Plane Oracle $\mathrm{ORC}(z_q, \mathcal{Z}_i)$:

Given a *query point* z_q,

- either assert $z_q \in \mathcal{Z}_i$, or
- return a separating hyperplane (cutting-plane)



$$a(z_q)^T z \leq b(z_q), \ \forall z \in \mathcal{Z}_i \quad \text{and} \quad a(z_q)^T z_q - b(z_q) = s(z_q) > 0.$$

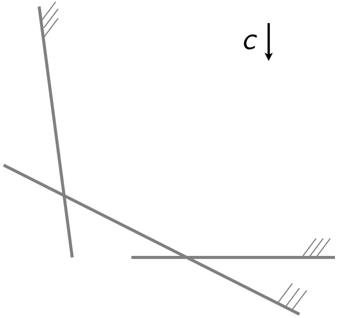
Assumption: $z_q \to \bar{z}$ and $s(z_q) \to 0$ implies $\bar{z} \in \mathcal{Z}_i$.

Node capability: LP solver

Approximate Linear Program:

$$\max c^T z$$
$$A_H^T z \leq b_H,$$

with H a collection of cutting-planes.

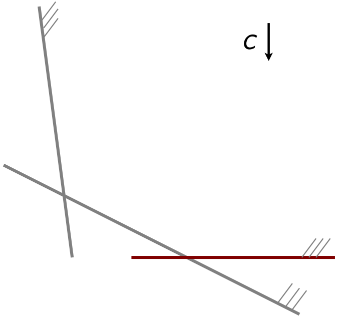


Node capability: LP solver

Approximate Linear Program:

$$\max c^T z$$
$$A_H^T z \leq b_H,$$

with H a collection of cutting-planes.



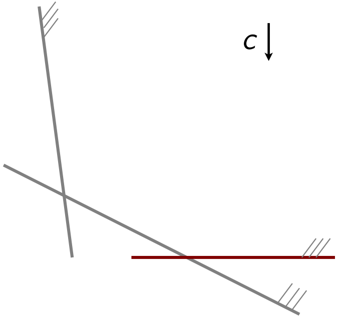
Basis: Minimal set of active constraints, $B \subset H$, $|B| \leq d$.

Node capability: LP solver

Approximate Linear Program:

$$\max c^T z$$
$$A_H^T z \le b_H,$$

with H a collection of cutting-planes.



Basis: Minimal set of active constraints, $B \subset H$, $|B| \le d$.

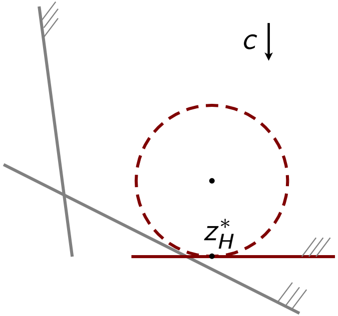
Problem: *non-uniqueness of optimal solution*

Node capability: LP solver

Approximate Linear Program:

$$\max c^T z$$
$$A_H^T z \leq b_H,$$

with H a collection of cutting-planes.



Basis: Minimal set of active constraints, $B \subset H$, $|B| \leq d$.

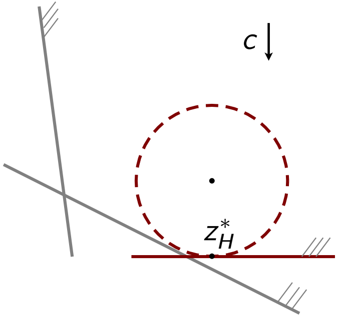
Problem: *non-uniqueness of optimal solution*

Minimal 2-norm solution gives a unique z_H^*

Approximate Linear Program:

$$\max c^T z$$
$$A_H^T z \leq b_H,$$

with H a collection of cutting-planes.



Minimal 2-norm solution: There exists $\bar{\epsilon}$ such that for all $\epsilon \in (0, \bar{\epsilon}]$.

z_H^* is the unique maximizer over $\mathcal{H}$ of

$$J_\epsilon(z) = c^T z - \frac{\epsilon}{2}\|z\|^2.$$

Cutting-Plane Consensus Algorithm

Cutting-Plane Consensus:

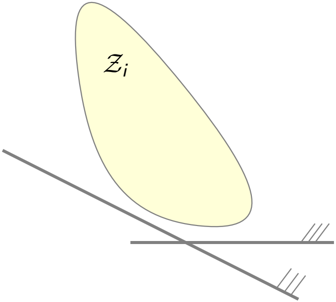
Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

Cutting-Plane Consensus Algorithm

Cutting-Plane Consensus:

Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

Initialization: $\mathcal{B}_0^{[i]} \supset \mathcal{Z}_i$ *and* $\max_{z \in \mathcal{B}_0^{[i]}} c^T z < \infty$.

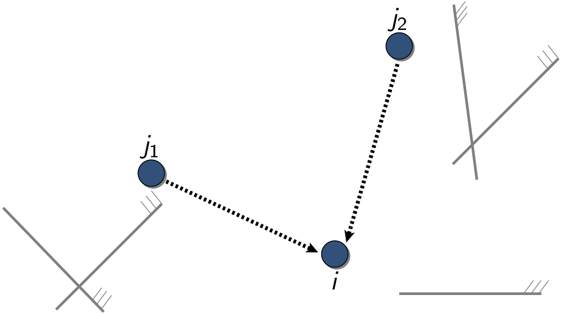


Cutting-Plane Consensus:

Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

Repeat:

(S1) *Receive the basis of in-neighbors* $Y^{[i]}(t) = \bigcup_{j \in \mathcal{N}_I(i,t)} B^{[j]}(t)$, $H^{[i]}_{tmp}(t) = B^{[i]}(t) \cup Y^{[i]}(t)$;



Cutting-Plane Consensus Algorithm

Cutting-Plane Consensus:

Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

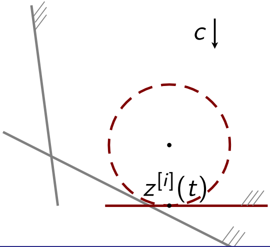
Repeat:

(S2) *Compute*

1. *a query point $z^{[i]}(t)$ as the 2-norm solution of*

$$\max_{z \in H_{tmp}^{[i]}(t)} \quad c^T z$$

2. *a minimal set of active constraints $B_{tmp}^{[i]}(t)$;*



Cutting-Plane Consensus Algorithm

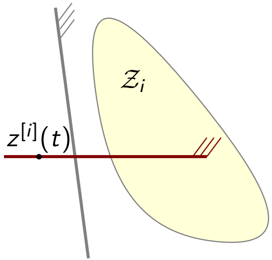
Cutting-Plane Consensus:

Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

Repeat:

(S3) *Call cutting-plane oracle*

$$h(z^{[i]}(t)) = \mathrm{ORC}(z^{[i]}(t), \mathcal{Z}_i);$$



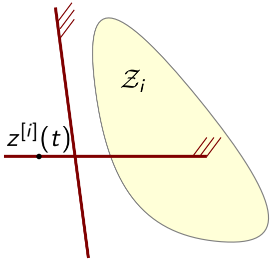
Cutting-Plane Consensus Algorithm

Cutting-Plane Consensus:

Processor i stores a basis of cutting-planes $B^{[i]}(t)$.

Repeat:

(S4) *Update $B^{[i]}(t+1)$ as minimal basis of $B^{[i]}_{tmp}(t) \cup h(z^{[i]}(t))$.*



Cutting-Plane Consensus Algorithm

Cutting-Plane Consensus:

Processor i stores a *basis of cutting-planes* $B^{[i]}(t)$.

Initialization: $\mathcal{B}_0^{[i]} \supset \mathcal{Z}_i$ and $\max_{z \in \mathcal{B}_0^{[i]}} c^T z < \infty$.

Repeat:

(S1) *Receive* the basis of in-neighbors $Y^{[i]}(t) = \bigcup_{j \in \mathcal{N}_I(i,t)} B^{[j]}(t)$, $H_{tmp}^{[i]}(t) = B^{[i]}(t) \cup Y^{[i]}(t)$;

(S2) *Compute* a query point $z^{[i]}(t)$ as 2-norm solution of $\max_{z \in H_{tmp}^{[i]}(t)} c^T z$, and a minimal set of active constraints $B_{tmp}^{[i]}(t)$;

(S3) *Call* cutting-plane oracle

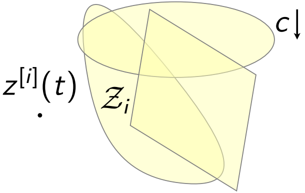
$$h(z^{[i]}(t)) = \mathrm{ORC}(z^{[i]}(t), \mathcal{Z}_i);$$

(S4) *Update* $B^{[i]}(t+1)$ as minimal basis of $B_{tmp}^{[i]}(t) \cup h(z^{[i]}(t))$.

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

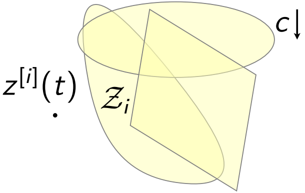
Details in

M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

Details in

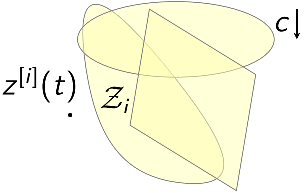
M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Algorithm analysis

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

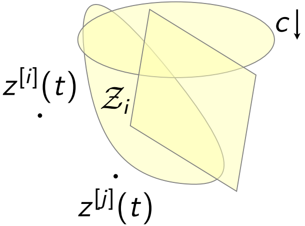
Details in

M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

Details in

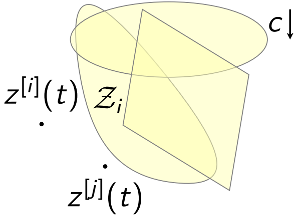
M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Algorithm analysis

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

Details in

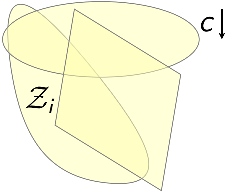
M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Algorithm analysis

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

Details in

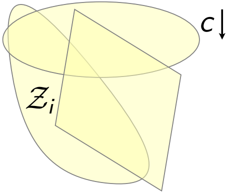
M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Algorithm analysis

Correctness proof idea:

Query points $z^{[i]}(t)$

- converge to set $\mathcal{Z}_i$,
- converge to each other $\|z^{[i]}(t) - z^{[j]}(t)\| \to 0$,
- converge to optimal solution.



Key analysis feature

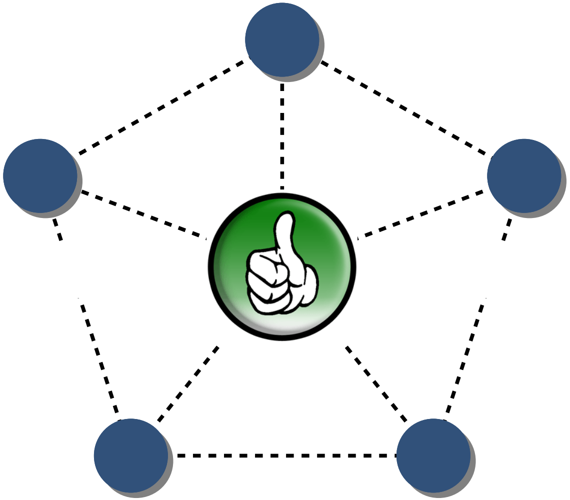
Minimal 2-norm solution is crucial!

- Convergence (proof) depends heavily on it: use $J_\epsilon(z^{[i]}(t))$.
- It allows to drop old cutting-planes.

Details in

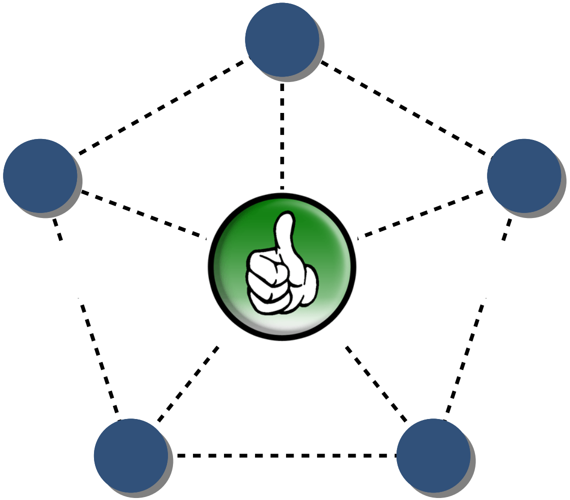
M. Bürger, G. Notarstefano, F. Allgöwer, IEEE TAC, Feb. 2014.

Algorithm strengths



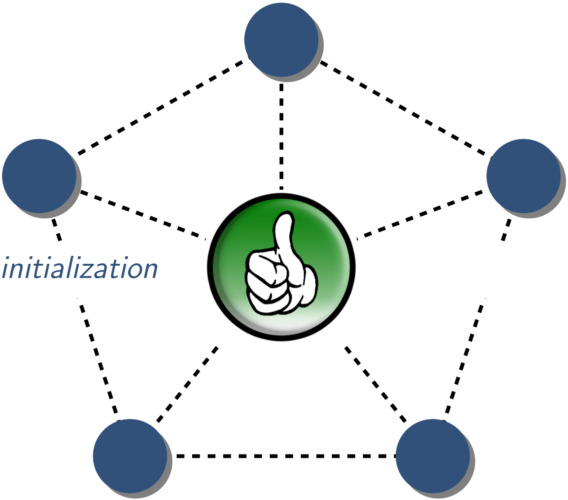
Local computations: $\mathcal{O}(d|\mathcal{N}_I(i)|)$

Algorithm strengths



Any (non-smooth) convex program *Local computations:* $\mathcal{O}(d|\mathcal{N}_I(i)|)$

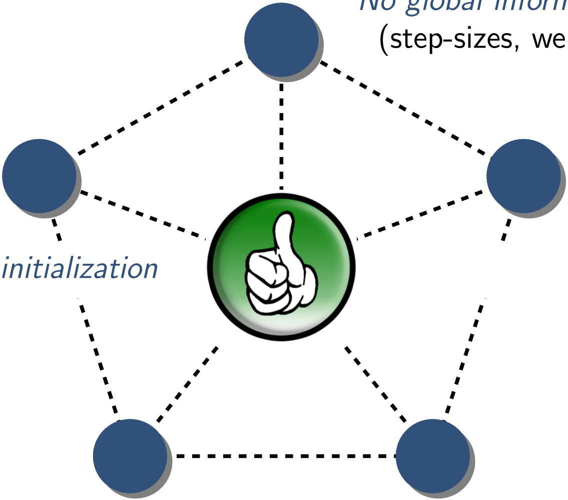
Algorithm strengths



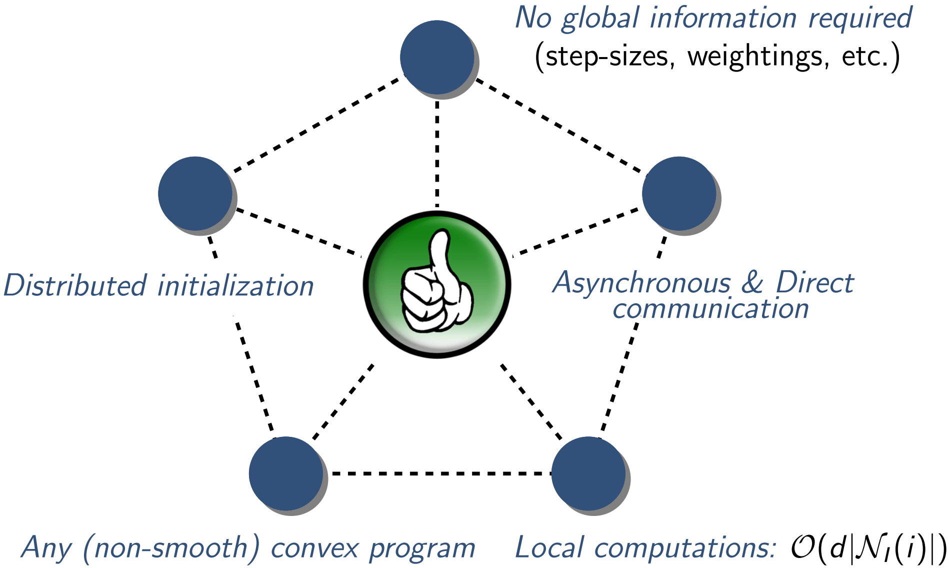
Distributed initialization

Any (non-smooth) convex program

Local computations: $\mathcal{O}(d|\mathcal{N}_I(i)|)$


No global information required
(step-sizes, weightings, etc.)
Distributed initialization
Any (non-smooth) convex program
Local computations: $\mathcal{O}(d|\mathcal{N}_I(i)|)$

Algorithm strengths



Concrete optimization set-ups

How to specify Cutting Plane Consensus for concrete problem representations?

- Nominal inequality constraints
- Uncertain (or semi-infinite) constraints
- Dual problem representations

Nominal Inequality Constraints

$$\max c^T z$$
$$f_i(z) \leq 0, \; i = 1, \ldots, n$$

also $f_i(z) = \max_l \; \{f_{i1}(z), f_{i2}(z), \ldots, f_{iL}(z)\}$.

Nominal Inequality Constraints

$$\max c^T z$$
$$f_i(z) \leq 0, \;\; i = 1, \ldots, n$$

also $f_i(z) = \max_l \; \{f_{i1}(z), f_{i2}(z), \ldots, f_{iL}(z)\}$.

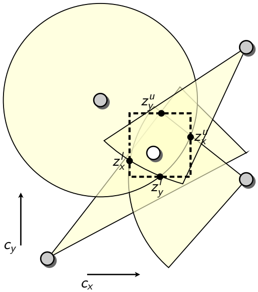
Cutting-Plane Oracle: For some query-point z_q,

$$f_i(z_q) + g_i^T (z - z_q) \leq 0,$$

for some $g_i \in \partial f_i(z_q)$ (sub-differential), is a cutting-plane.

Nominal Inequality Constraints

Application: Position Estimation in Wireless Sensor Networks



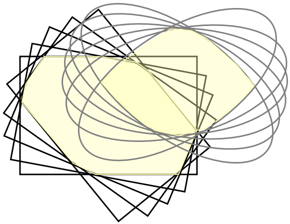
Distributed algorithm: Compute smallest bounding box!
Central problem originally formulated in:

L. Doherty, K. Pister, and L. E. Ghaoui, "Convex position estimation in wireless sensor networks," in INFOCOM 2001.

Robust Distributed Optimization

$$\max_{z \in \mathbb{R}^d} c^\top z$$

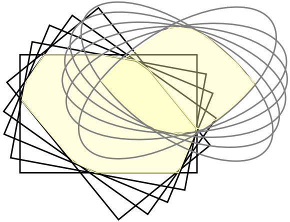
$$f_i(z, \theta_i) \leq 0, \text{ for all } \theta_i \in \Omega_i, \qquad i = 1, \ldots, n$$



Robust Distributed Optimization

$$\max_{z \in \mathbb{R}^d} c^\top z$$

$$f_i(z, \theta_i) \leq 0, \text{ for all } \theta_i \in \Omega_i, \qquad i = 1, \dots, n$$



- Each processor knows a constraint with *parametric uncertainties*.
- Agreement on optimizer of a *linear objective*.

Robust Distributed Optimization

$$\max_{z \in \mathbb{R}^d} c^\top z$$

$$f_i(z, \theta_i) \leq 0, \text{ for all } \theta_i \in \Omega_i, \qquad i = 1, \ldots, n$$

Cutting-Plane Oracle: given query point z_q. Let θ_q^* be maximizer of

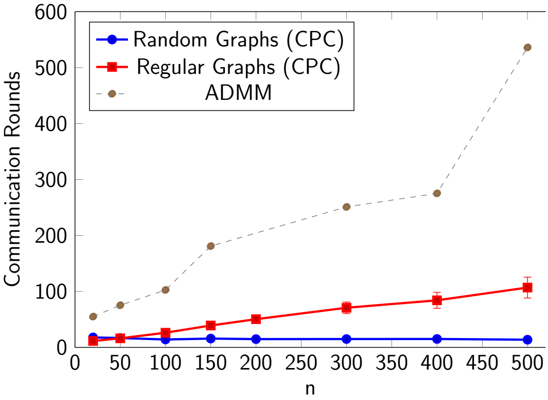
$$\max_{\theta} \; f_i(z_q, \theta), \quad \theta \in \Omega_i$$

If optimal value is greater than zero, a cutting-plane is

$$f_i(z_q, \theta_q^*) + g_i^\top (z - z_q) \leq 0, \quad g_i \in \partial f_i(z_q, \theta_q^*).$$

Robust linear programming with ellipsoidal uncertainty:

$$\min \ c^T z, \qquad a_i^T z \leq b_i, \ a_i \in \mathcal{A}_i = \{a_i = \bar{a}_i + P_i u, \|u\|_2 \leq 2\}$$



Comparison with ADMM: *better time complexity*.

Separable cost and dual decomposition

Almost Separable Optimization Problems:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \quad & \sum_{i=1}^{n} G_i x_i = \mathbf{h} \\ & x_i \in \mathcal{X}_i. \end{aligned}$$

Separable cost and dual decomposition

Almost Separable Optimization Problems:

$$\begin{aligned} \min \ & \sum_{i=1}^{n} f_i(x_i) \\ \text{s.t.} \ & \sum_{i=1}^{n} G_i x_i = \mathbf{h} \\ & x_i \in \mathcal{X}_i. \end{aligned}$$

$$\max_{\pi,u} \ -\pi^T \mathbf{h} + \sum_{i=1}^{n} u_i,$$

$$(\pi, u) \in \bigcap_{i=1}^{n} \mathcal{Z}_i$$

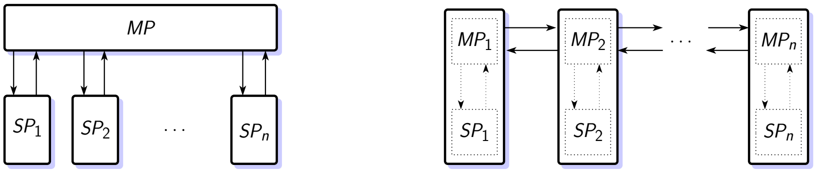
$$\mathcal{Z}_i := \Big\{ (\pi, u) : u_i \leq \min_{x_i} f_i(x_i) + \pi^T G_i x_i, \quad x_i \in \mathcal{X}_i \Big\}.$$

The *dual problem* is contained in our general setup!

Separable cost and dual decomposition

Cutting-Plane Consensus is a fully distributed version of the

dual non-linear Dantzig-Wolfe Decomposition!



Classical Decomposition! *Novel Distributed Structure!*

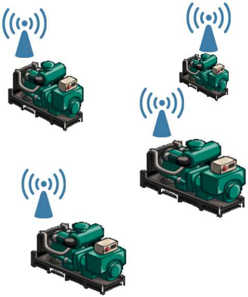
see Distributed Decision Making Via Two-Stage Distributed Simplex, CDC 2011.

Micro-grid control problem

Almost Separable Optimization:

$$\min \sum_{i=1}^{n} f_i(x_i)$$
$$\text{s.t.} \sum_{i=1}^{n} G_i x_i = \mathbf{h}$$
$$x_i \in \mathcal{X}_i.$$

Application: Micro-Grid Problem



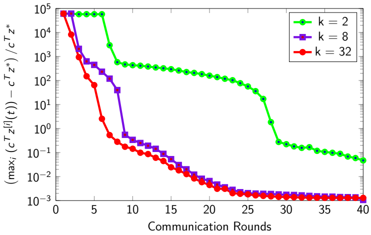
- Distributed Power Generation
- Energy Storage Devices
- Controllable Loads
- Power trading with main grid

Micro-grid control problem

Almost Separable Optimization:

$$\min \sum_{i=1}^{n} f_i(x_i)$$
$$\text{s.t.} \sum_{i=1}^{n} G_i x_i = \mathbf{h}$$
$$x_i \in \mathcal{X}_i.$$

Application: Micro-Grid Problem



- Efficient solution for problems with *100 controlled units.*
- Fully distributed solution, *without any coordination!*

Conclusions

- distributed optimization algorithms based on constraint exchange
- main strengths
  - asynchronous & direct communication
  - handle also non-smooth convex programs
  - no global parameters to set
  - completion time scales “nicely” with network size
- applicable to several concrete control and estimation problems
- suited for distributed MPC (under investigation, see ECC ’13)