Network abstract linear programming and application to formation control

Giuseppe Notarstefano

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

http://cor.unile.it
www.dei.unipd.it/∼notarste

Work supervised by: Francesco Bullo (UCSB)
Work carried out during the PhD program at the University of Padova

Motivations and contribution

Objective: solve optimization problems over networks in a distributed way.

Contribution: identify a class of optimization problems (over networks),

provide distributed algorithms to solve them,

apply to robotic and sensor networks.

- Abstract linear programming: definition and main properties

- Network abstract linear programming: distributed algorithms

- Application 1: formation control

- Application 2: sensor selection

Outline

- Abstract linear programming: definition and main properties
- Network abstract linear programming: distributed algorithms
- Application 1: formation control
- Application 2: sensor selection

Linear programming

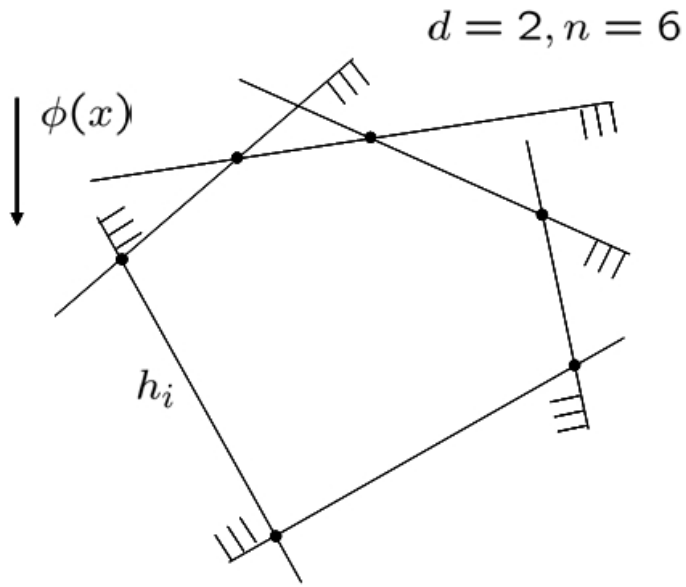
Linear programming: minimize a linear function in d variables
subject to n linear inequalities (interested in $\mathbf{d} << \mathbf{n}$);

- $x \in \mathbb{R}^d$, $f \in \mathbb{R}^d$, $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$,

$$\text{minimize } f^T x$$

$$\text{subj. to } Ax \preceq b$$

- linear objective and constraints
- convex problem



Linear programming

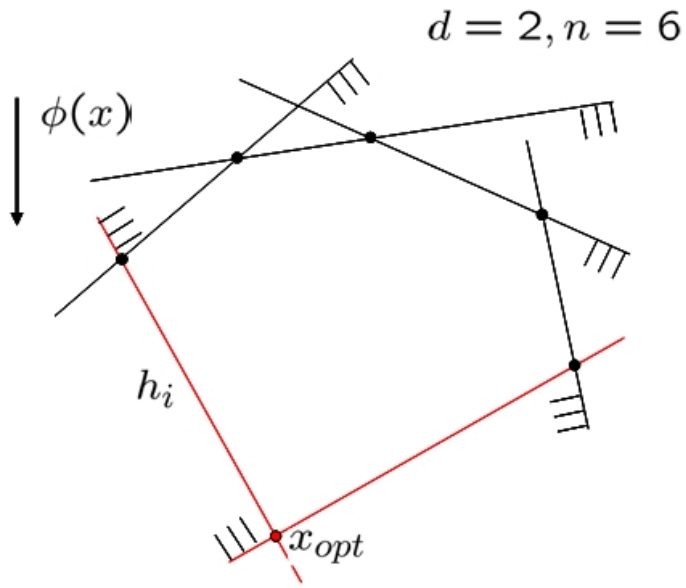
Linear programming: minimize a linear function in d variables
subject to n linear inequalities (interested in $\mathbf{d} << \mathbf{n}$);

- $x \in \mathbb{R}^d$, $f \in \mathbb{R}^d$, $A \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$,

$$\text{minimize } f^T x$$

$$\text{subj. to } Ax \preceq b$$

- linear objective and constraints
- convex problem



The solution is completely characterized by d constraints

Abstract framework

Abstract linear programming: abstract framework that captures the main features of linear programming.

Consider the optimization problem specified by the pair (H, ω)

- H is a finite set of constraints,
- $\omega(G)$ is the *value function* (minimum value attainable by the objective function subject to $G \subset H$)

Abstract framework (axioms)

Axioms

Monotonicity. For any F, G, with $F \subset G \subset H$

$$\omega(F) \leq \omega(G)$$

Locality. For any $F \subset G \subset H$ with $\omega(F) = \omega(G)$ and any $h \in H$, then

$$\omega(G) < \omega(G \cup \{h\}) \Rightarrow \omega(F) < \omega(F \cup \{h\})$$

(h is *violated* by F and G)

Abstract framework (axioms)

Axioms

Monotonicity. For any F, G, with $F \subset G \subset H$

$$\omega(F) \leq \omega(G)$$

Locality. For any $F \subset G \subset H$ with $\omega(F) = \omega(G)$ and any $h \in H$, then

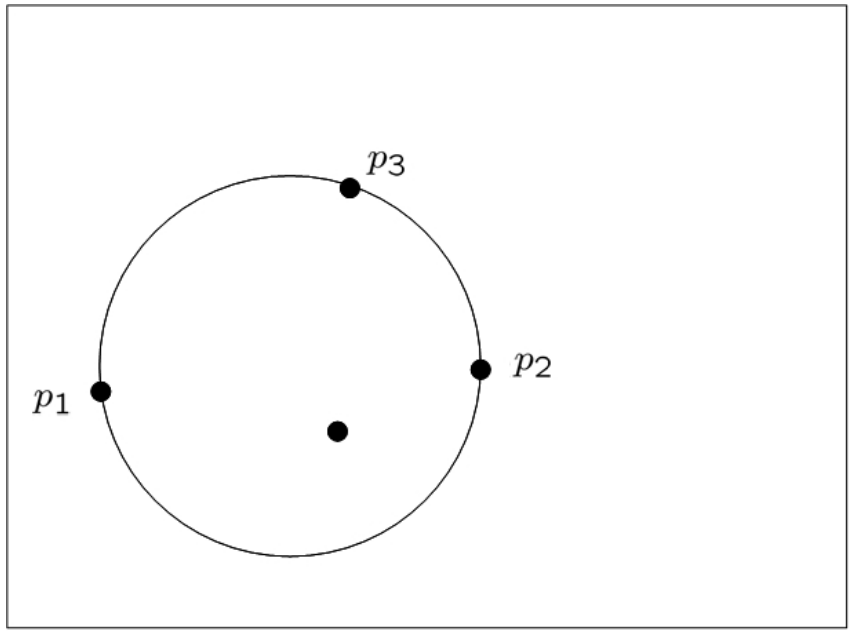
$$\omega(G) < \omega(G \cup \{h\}) \Rightarrow \omega(F) < \omega(F \cup \{h\})$$

(h is *violated* by F and G)

References: *Agarwal, Sharir*, ACM-CS '98; *Matousek, Sharir, Welzl*, ALG '96; *Gartner, Welzl*, STACS '96.

Smallest enclosing ball

Compute the smallest ball enclosing a set of points



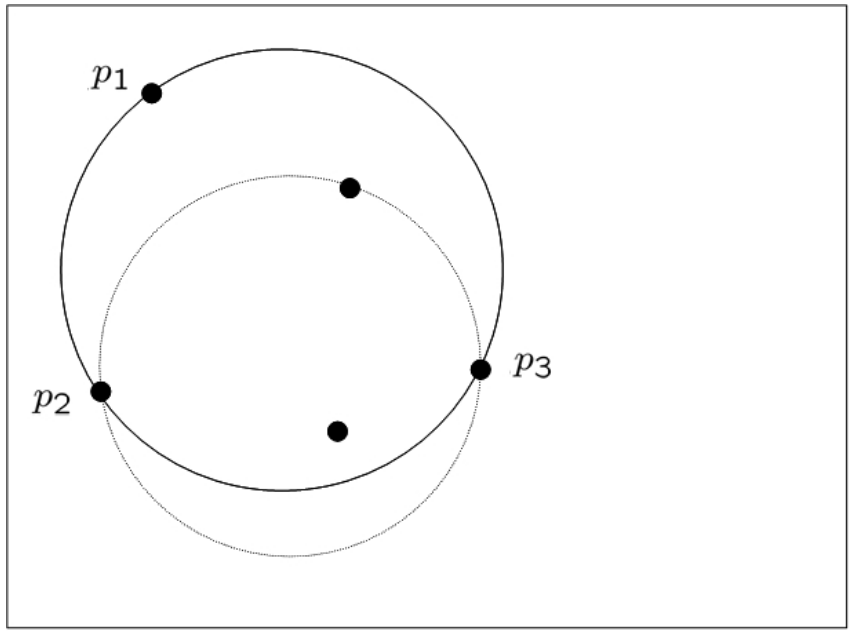
For any $F \subset G$

Monotonicity: $\omega(F) \leq \omega(G)$

Locality: $\omega(G) < \omega(G \cup \{h\}) \Rightarrow \omega(F) < \omega(F \cup \{h\})$

Smallest enclosing ball

Compute the smallest ball enclosing a set of points



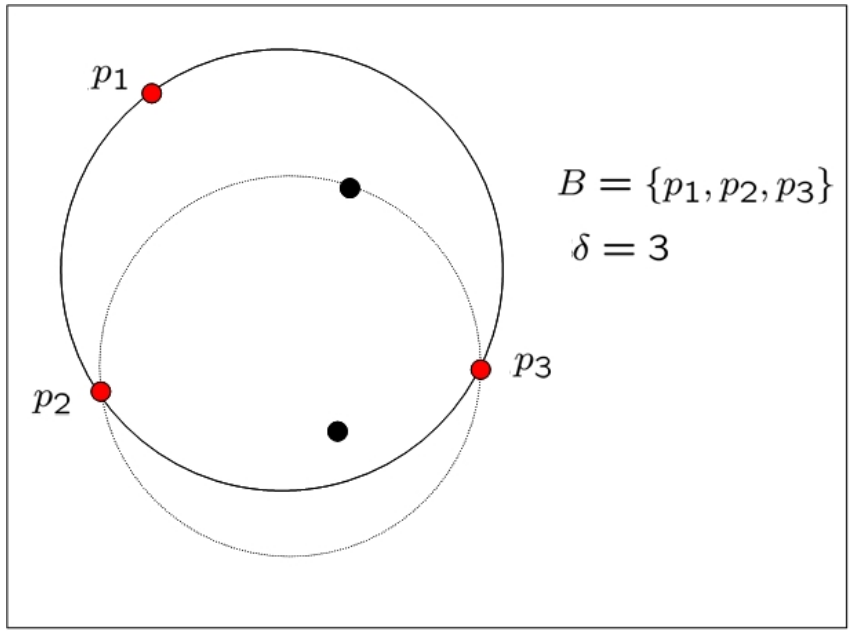
For any $F \subset G$

Monotonicity: $\omega(F) \leq \omega(G)$

Locality: $\omega(G) < \omega(G \cup \{h\}) \Rightarrow \omega(F) < \omega(F \cup \{h\})$

Smallest enclosing ball

Compute the smallest ball enclosing a set of points



For any $F \subset G$

Monotonicity: $\omega(F) \leq \omega(G)$

Locality: $\omega(G) < \omega(G \cup \{h\}) \Rightarrow \omega(F) < \omega(F \cup \{h\})$

Abstract framework (useful definitions)

Basis B of G: minimal subset of constraints $B \subset G \subset H$, such that

$$\omega(B) = \omega(G)$$

Combinatorial dimension δ: maximum cardinality of any basis B

Primitive operations

Violation test: for a constrain $h \in H$ and a basis B, tests if h is violated by B.

$\texttt{Viol}(B, h)$

Basis computation: for a constraint h and a basis B, computes a basis of $B \cup \{h\}$.

$\texttt{Basis}(B \cup \{h\},)$

Abstract framework (useful definitions)

Basis B of G: minimal subset of constraints $B \subset G \subset H$, such that

$$\omega(B) = \omega(G)$$

Combinatorial dimension δ: maximum cardinality of any basis B

Primitive operations

Violation test: for a constrain $h \in H$ and a basis B, tests if h is violated by B.

$\texttt{Viol}(B, h)$

Basis computation: for a constraint h and a basis B, computes a basis of $B \cup \{h\}$.

$\texttt{Basis}(B \cup \{h\},)$

Remark: Any basis B of H characterizes the solution completely!

Examples (Geometric Optimization)

- Linear programming

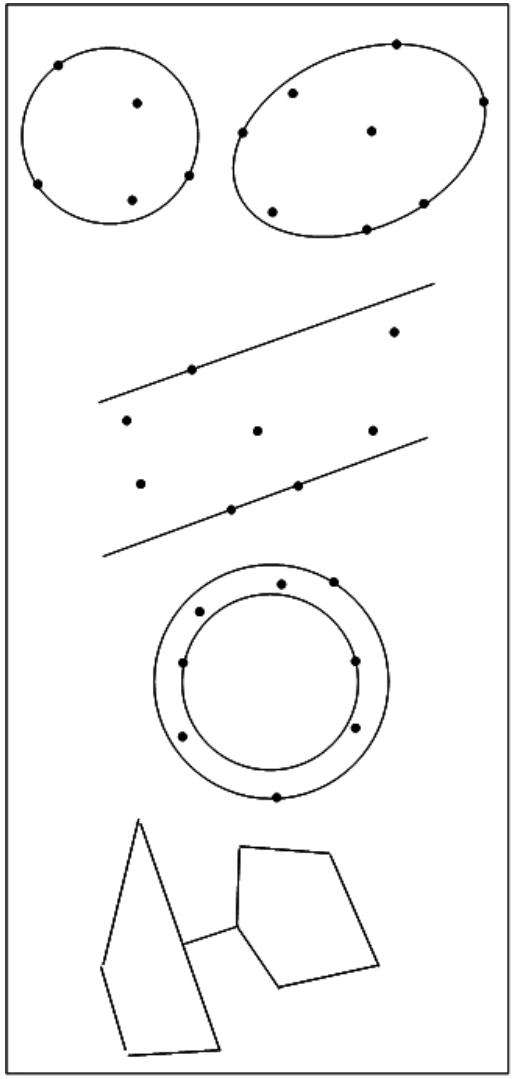
- Smallest enclosing ball, ellipsoid and orthotope

- Smallest enclosing stripe (generic points)

- Smallest enclosing annulus

- Shortest distance between polytopes

- ...



Outline

- Abstract linear programming: definition and main properties
- Network abstract linear programming: distributed algorithms
- Application 1: formation control
- Application 2: sensor selection

Network modeling

Network. Collection of "computing elements" located
at nodes of a (directed) network graph.
N. A. Lynch - Distributed algorithms

Communication graph $\mathcal{G} = (I, E_{\text{cmm}})$

· $I = \{1, \ldots, n\}$, identifier of the computing elements

· $E_{\text{cmm}} : \mathbb{N}_0 \to 2^{I \times I}$, communication edge map

· $E_{\text{cmm}}(t) = \{(i, j) \in I \times I \mid \text{process } i \text{ can communicate to } j \text{ at time } t\}$

Network modeling

Network. Collection of "computing elements" located
at nodes of a (directed) network graph.
N. A. Lynch - Distributed algorithms

Communication graph $\mathcal{G} = (I, E_{\text{cmm}})$

· $I = \{1, \ldots, n\}$, identifier of the computing elements

· $E_{\text{cmm}} : \mathbb{N}_0 \to 2^{I \times I}$, communication edge map

· $E_{\text{cmm}}(t) = \{(i, j) \in I \times I \mid \text{process } i \text{ can communicate to } j \text{ at time } t\}$

Assumption: $\mathcal{G}$ jointly strongly connected.

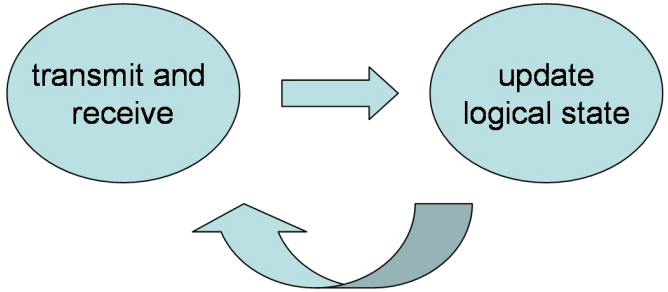
Distributed algorithm

Sets

- W, set of "logical" states $w^{[i]} \in \mathbb{R}^d$
- $W_0 \subset W$, subset of allowable initial values
- M, message alphabet, collection of messages $y_j^{[i]} \in M$

Maps

- message generation function
  $y_j^{[i]}(t) = \mathrm{msg}\Big(w^{[j]}(t)\Big), \quad (i,j) \in E_{\mathrm{cmm}}(t)$
- state transition function
  $w^{[i]}(t+1) = \mathrm{stf}\Big(w^{[i]}(t), y^{[i]}(t)\Big)$



Network abstract linear program

Network abstract linear program $(\mathcal{G}, (H, \omega), \mathcal{B})$

(i) $\mathcal{G} = (I, E_{\text{cmm}})$, communication digraph;

(ii) (H, ω), abstract linear program;

(iii) $\mathcal{B} : H \to I$, surjective map called *constraint distribution map*.

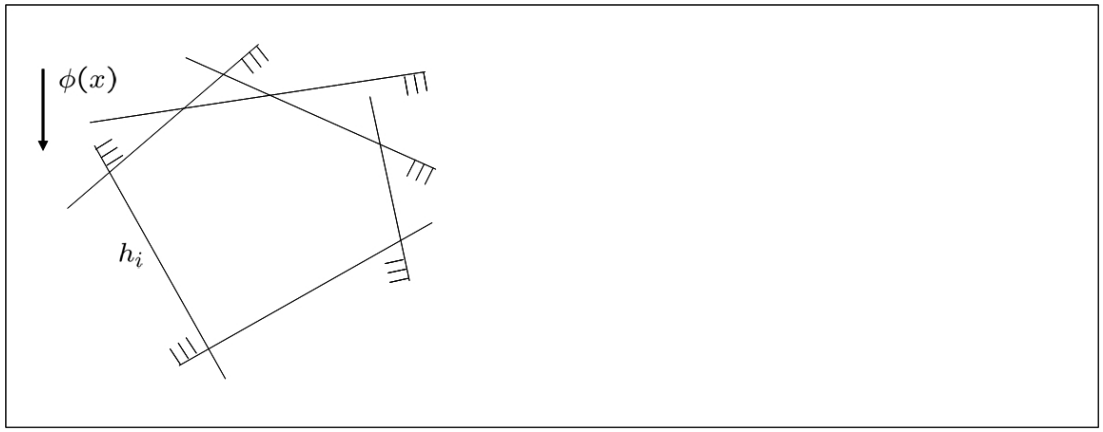
Network abstract linear program

Network abstract linear program $(\mathcal{G}, (H, \omega), \mathcal{B})$

(i) $\mathcal{G} = (I, E_{\text{cmm}})$, communication digraph;

(ii) (H, ω), abstract linear program;

(iii) $\mathcal{B} : H \to I$, surjective map called *constraint distribution map*.



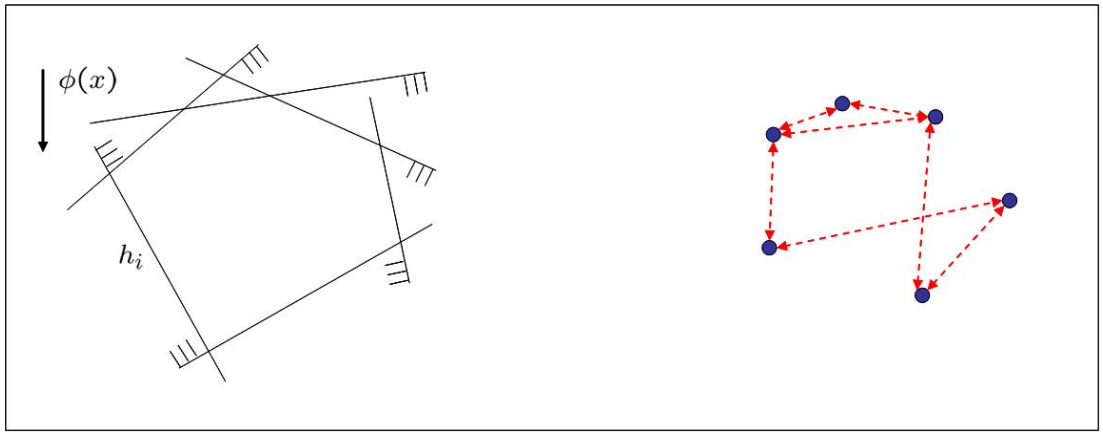
Network abstract linear program

Network abstract linear program $(\mathcal{G}, (H, \omega), \mathcal{B})$

(i) $\mathcal{G} = (I, E_{\text{cmm}})$, communication digraph;

(ii) (H, ω), abstract linear program;

(iii) $\mathcal{B} : H \to I$, surjective map called *constraint distribution map*.



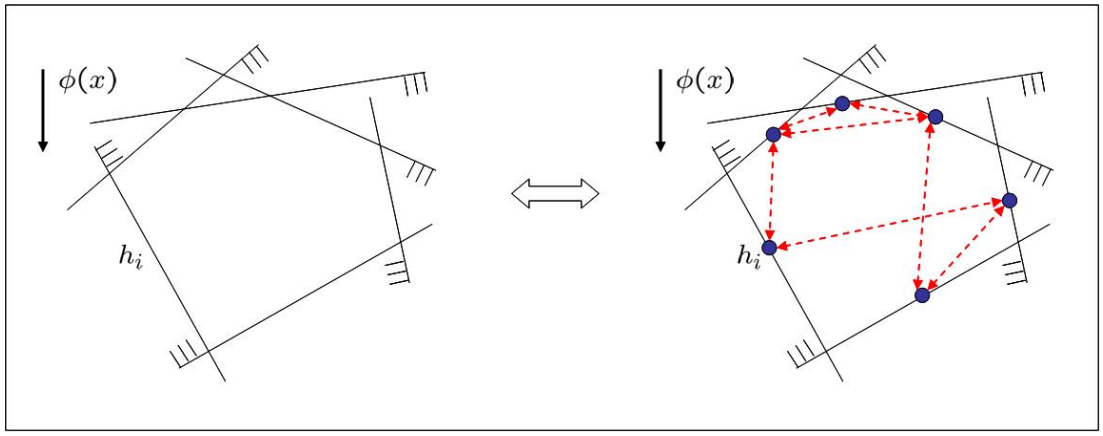
Network abstract linear program

Network abstract linear program $(\mathcal{G}, (H, \omega), \mathcal{B})$

(i) $\mathcal{G} = (I, E_{\text{cmm}})$, communication digraph;

(ii) (H, ω), abstract linear program;

(iii) $\mathcal{B} : H \to I$, surjective map called *constraint distribution map*.



Solution of network abstract linear program

Each process computes the basis B that solves $(\mathcal{G}, (H, \omega), \mathcal{B})$, that is:

- $w^{[i]}(0) = h_i$
- $\exists\ T$ such that $\omega(w^{[i]}(T)) = \omega(B) = \omega(H)$

(Consensus problem)

Desired

- Size of allocated memory ($\dim(W)$) does NOT depend on n
- Computation time at each communication round does NOT depend on n
- T (Time Complexity) depends "nicely" on n (we would like $O(n)$).

Distributed algorithm (*FloodBasis*)

FloodBasis [Informal description]
Each agent j initializes its logical state $w^{[j]}$ to its constraint, then

(i) it acquires from its neighbors $\mathcal{N}(j)$ their current logical state;

(ii) it computes the basis of its logical state and its neighbors' logical state.

(iii) it updates its logical state $w^{[j]}$ and message $y^{[j]}$ with the basis obtained in (ii) and its initial constraint (that it maintains in memory);

Distributed algorithm (*FloodBasis*)

FloodBasis [Informal description]
Each agent j initializes its logical state $w^{[j]}$ to its constraint, then

(i) it acquires from its neighbors $\mathcal{N}(j)$ their current logical state;

(ii) it computes the basis of its logical state and its neighbors' logical state;

(iii) it updates its logical state $w^{[j]}$ and message $y^{[j]}$ with the basis obtained in (ii) and its initial constraint (that it maintains in memory).

Remark: each node must have bounded in-degree.

Correctness of *FloodBasis*

Proposition 1 (Correctness of *FloodBasis*)**.** Given

- time-dependent network with communication digraph $\mathcal{G} = (I, E_{\text{cmm}})$;
- $(\mathcal{G}, (H, \omega), \mathcal{B})$, a network abstract linear program;
- $\mathcal{G}$ jointly strongly connected.

Then the *FloodBasis* algorithm solves $(\mathcal{G}, (H, \omega), \mathcal{B})$. □

(CDC '07)

Correctness of *FloodBasis* (sketch of proof)

Sketch of proof. :

- for each i, $\omega(B^{[i]})$ is nondecreasing;
- the domain set of ω has finite cardinality;
- if the algorithm stops, then $\omega(B^{[i]}) = \omega(B)$ for each $i \in \{1, \ldots, n\}$.

□

Halting condition

Proposition 2 (Halting condition)**.** Given

- time-independent network with strongly connected digraph $\mathcal{G}$
- the *FloodBasis* algorithm is running.

Then each process can halt the algorithm execution if the value of its basis has not changed after $2\,\mathrm{diam}(\mathcal{G}) + 1$ communication rounds. □

(CDC '07)

Remarks

Generalizations of *FloodBasis*

- time-dependent graph, arbitrary memory, arbitrary in-degree
- time-independent graph, bounded memory, arbitrary in-degree

Time complexity

- rough bound $O(n^{\delta+1})$
- conjecture: $O(n)$ expected time

Outline

- Abstract linear programming: definition and main properties
- Network abstract linear programming: distributed algorithms
- Application 1: formation control
- Application 2: sensor selection

Robotic network

Robotic network

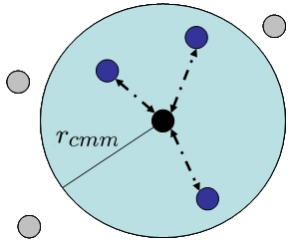
- control systems (physical agents)

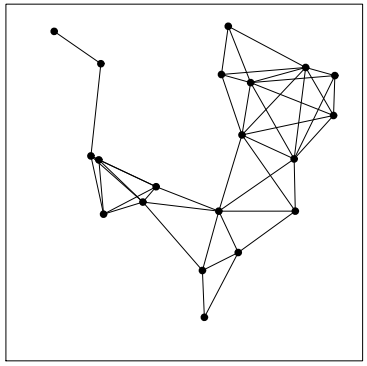
$$p^{[i]}(t+1) = p^{[i]}(t) + u^{[i]}(t), \qquad p^{[i]} \in \mathbb{R}^d, u^{[i]} \in B(0, r_{\text{ctr}})$$

- communication graph $\mathcal{G}_{\text{disk}}(r_{\text{cmm}}) \cdot (\{p_1, ..., p_n\}) = (I, \mathcal{E}_{\text{disk}})$

$$\mathcal{E}_{\text{disk}}(I) = \{(i,j) \mid \|p_i - p_j\| \leq r_{\text{cmm}}\}$$

(On Synchronous Robotic Networks - Martinez, Bullo, Cortes, Frazzoli)





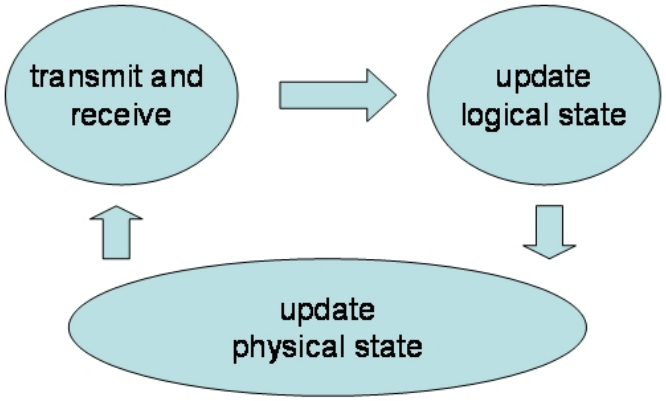
Control and communication law

Sets

- W, set of "logical" states $w^{[i]} \in \mathbb{R}^d$
- $W_0 \subset W$, subset of allowable initial values
- M, message alphabet, collection of messages $y_j^{[i]} \in M$

Maps

- message generation function
  $y_j^{[i]}(t) = \mathrm{msg}\Big(w^{[j]}(t)\Big), \quad (i,j) \in E$
- state transition function
  $w^{[i]}(t+1) = \mathrm{stf}\Big(w^{[i]}(t), y^{[i]}(t)\Big)$
- control function
  $u^{[i]}(t+1) = \mathrm{ctl}\Big(w^{[i]}(t+1), x^{[i]}(t)\Big)$



(On Synchronous Robotic Networks - Martinez, Bullo, Cortes, Frazzoli)

Minimum time formation control

General setting

Given a robotic network, find a control and communication law such that the agents converge in minimum time to a prescribed shape.

Point, line and circle formation

Consider $d = 2$, converge in minimum time to a point (rendezvous), a line, a circle.

Centralized solution

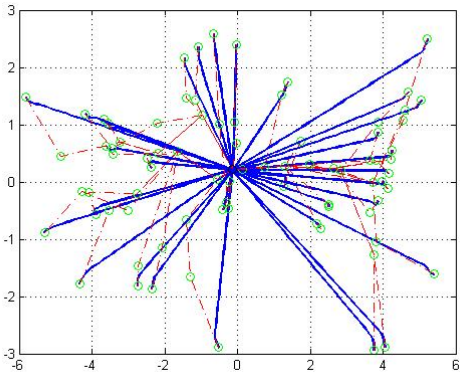
Equivalent to finding

- Smallest enclosing ball $\longrightarrow$ point (rendezvous) — ALP
- Smallest enclosing stripe $\longrightarrow$ line — ALP if non generic points
- Minimum width enclosing annulus $\longrightarrow$ circle — ALP if smallest area annulus

Move toward estimate

- Naive idea: compute centralized solution by means of *FloodBasis*, then move toward the optimal target;
- Alternative (uncorrect): compute centralized solution by means of *FloodBasis* and start moving toward the "estimate" of the optimal target;
- Move-toward-estimate (correct): compute centralized solution by means of *FloodBasis* and start moving toward the "estimate" of the optimal target, while maintaining connectivity.

Example: rendezvous



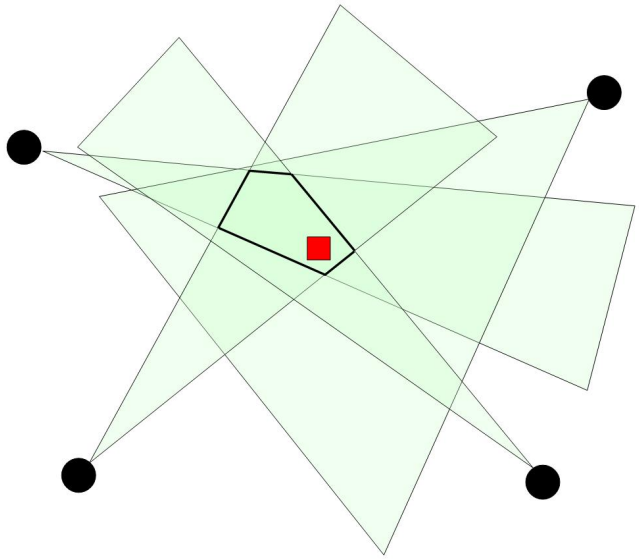
Outline

- Abstract linear programming: definition and main properties
- Network abstract linear programming: distributed algorithms
- Application 1: formation control
- Application 2: sensor selection

Sensor selection problem

- set of sensors $S = \{s^{[1]}, ..., s^{[n]}\}$ in the plane;
- target to be detected located at $x \in \mathbb{R}^2$;
- each sensor detects a (possibly unbounded) region s.t.
  - · contains the target;
  - · intersection of a finite number of half-planes.

Problem (k-SSP):
find the $k << n$ sensors that provide the "best" estimation of the target position (according to a utility function)

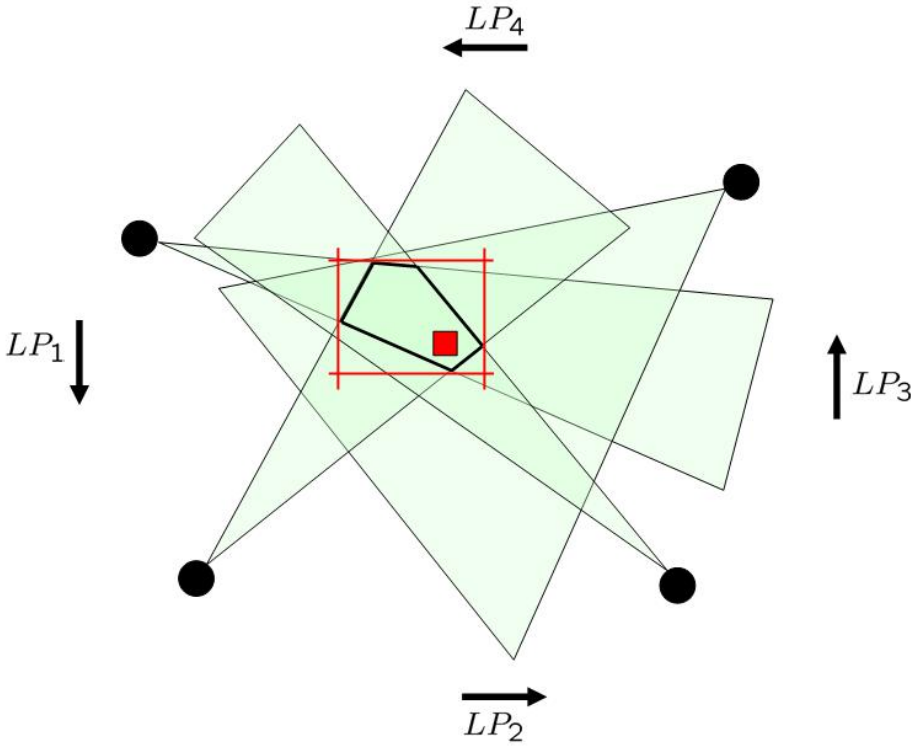


problem setting as in *V. Isler, R. Bajcsy* - TASE '06

Network ALP for k-SSP

α-approximation of k-SSP
find k sensors s.t. the error in estimating the position of the target
is within a factor α of the error of the optimal.

Solution: solve 4 parallel NALP



Conclusions

Summary

- Identified a class of optimization problems
- Distributed algorithm over network
- Application to motion coordination (distributed geometric optimization)
- Application to sensor networks

Perspectives

- Time complexity of *FloodBasis* (at least for specific cases)
- Other (hopefully) important applications for NALP
- Find other classes of optimization problems