Distributed connectivity of mobile robotic networks

Job Talk

July 24, 2009

Mike Schuresko

advised by Jorge Cortés

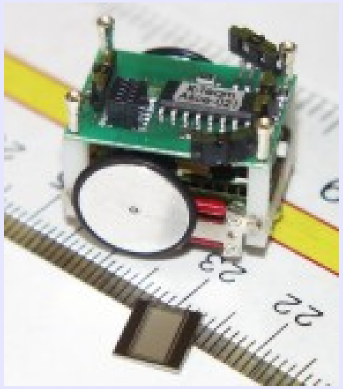
Applied Mathematics and Statistics
University of California, Santa Cruz

Mechanical and Aerospace Engineering
University of California, San Diego

Outline

Distributed control of swarms

Schooling fish


Tiny robot, courtesy (CS), see (CAS00)

(i) Large number of robots with limited communication.

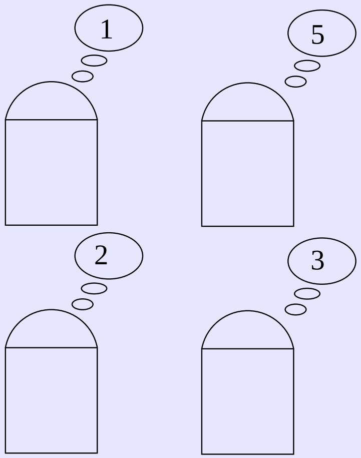
(ii) Control algorithm and communication law on each robot.

Goal is to write control algorithm and communication law for individual robots such that the whole swarm achieves some collective task.

Applications

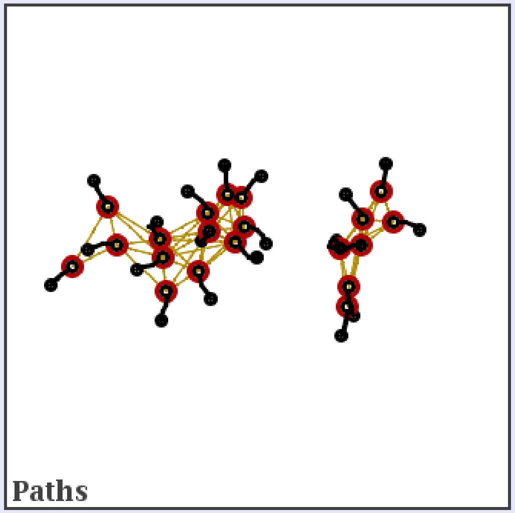
- Programmable matter
- Exploration
  - Mars
  - Environmental
  - interior/cave
  - urban search/rescue
- (ad-hoc) Cell phones with directional antennas
- Service stochastic events over an area.
- Mobile infrastructure
  - highway cleanup deploy likely repair needs
  - lightweight drones optimize ad-hoc net coverage
- Manufacturing plant = network of many robots.

Sample Tasks

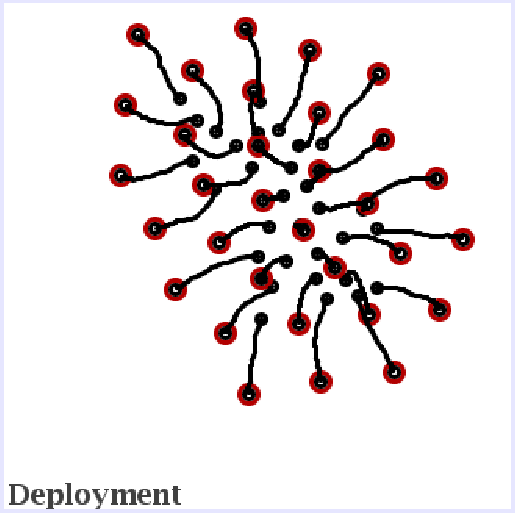


Consensus

Flocking

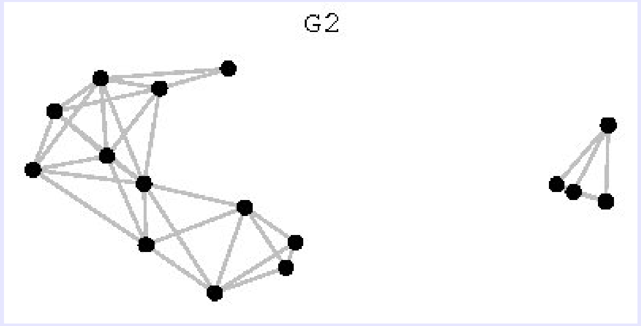


Rendezvous

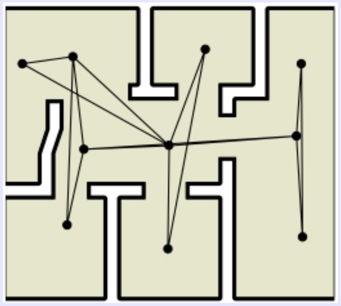


Deployment

Spatially-induced graphs



R-disk communication graph



Visibility graph

We model communication networks with spatially induced graphs

(i) Set of robot positions induce ***graph,*** $G = (V, E)$.

(ii) Edge between robots i and j indicates communication is possible between i and j.

(iii) Mapping between set of positions and graph should be invariant under permutation of robot identities

Here we show the ***r-disk graph***. We like to pick graphs which are reasonable, but crude, approximations of how wireless networks might actually behave.

Robotic Network Model

Network model equivalent to (MBCF07a).

Each robot runs a discrete time ***communications law***. At particular time slices, robots communicate with neighbors over proximity graph, and modify values stored in ***logic variables***.

Each robot, i, runs a continuous time ***control law*** which controls the motion of robot i based on i's position state and logic variables. Robots are fully actuated. In our case, they live in $\mathbb{R}^2$.

Outline

(i) ***Intro***

  (a) Cooperative control

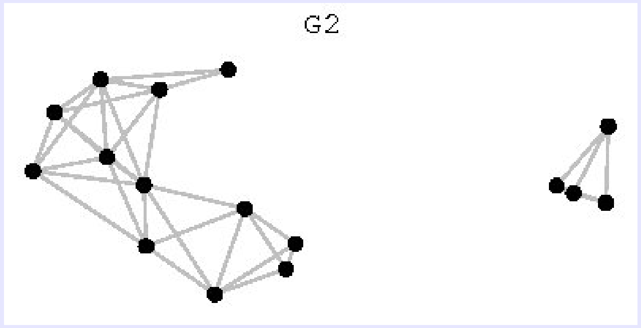
  (b) ***Connectivity problem***

  (c) Literature review

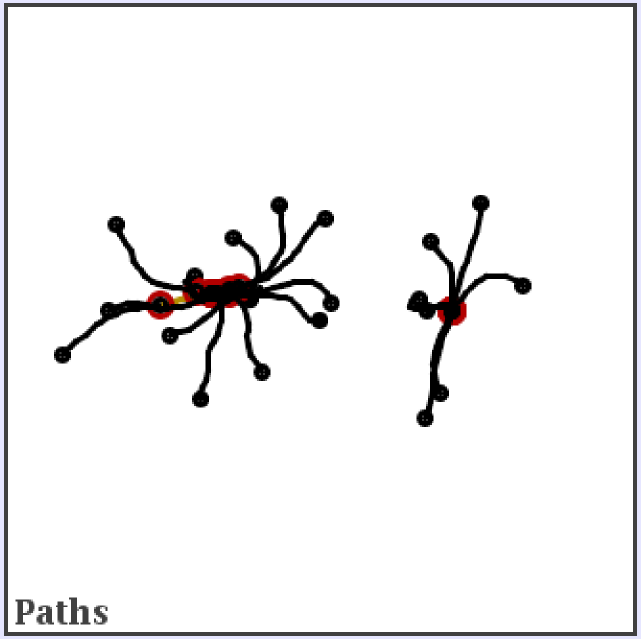
(ii) Our contributions

(iii) Conclusions / bibliography

Connectivity and collective behavior



R-disk communication graph

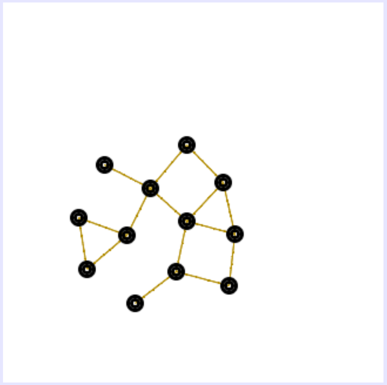


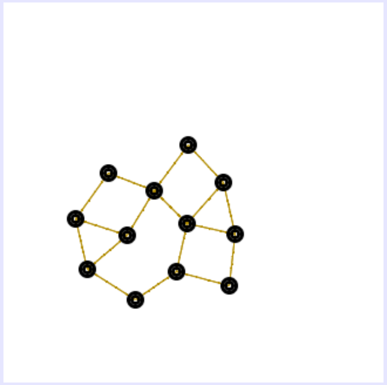
Failure to rendezvous due to lapse in connectivity

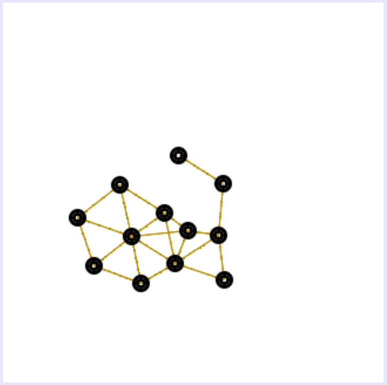
If communication network becomes disconnected, it is, at best, as if we have two smaller swarms.

In a practical sense, connectivity might translate into the ability to find all of one's robots after a task is complete.

Staying connected / How connected? (1 of 2)

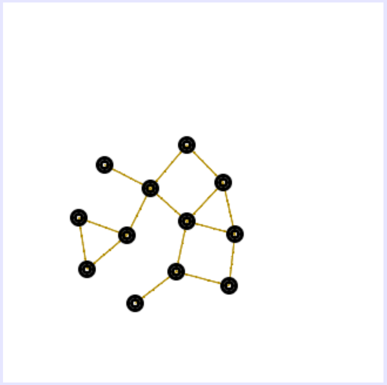

This graph is poorly connected

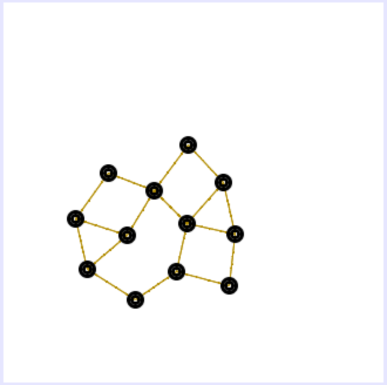

This graph is more connected

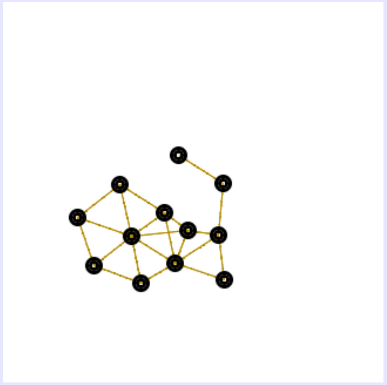

What about this one?

Degree to which a network is connected determines rate of convergence of many robotic control algorithms

Staying connected / How connected? (1 of 2)


This graph is poorly connected


This graph is more connected

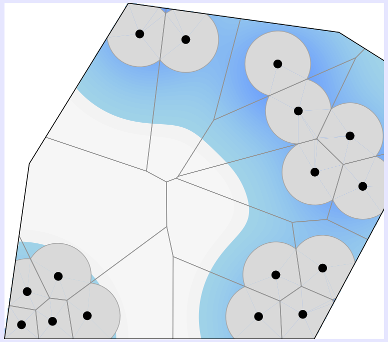

What about this one?

We will discuss a particular measure of connectivity, the *algebraic connectivity* of a communication graph.

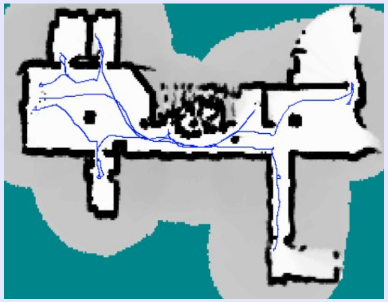
Practical examples of connectivity

Connectivity is important while doing some other task So we need tools to combine connectivity maintenance with another task.

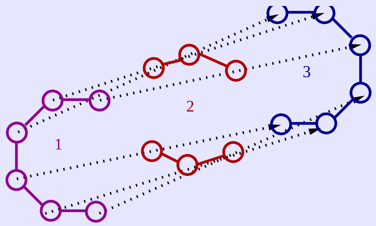
Examples:



Deployment while maintaining connectivity, image from (MBCF07b).



Mapping/exploration while maintaining connectivity, image from (SAB+00).



Formation morphing while maintaining connectivity

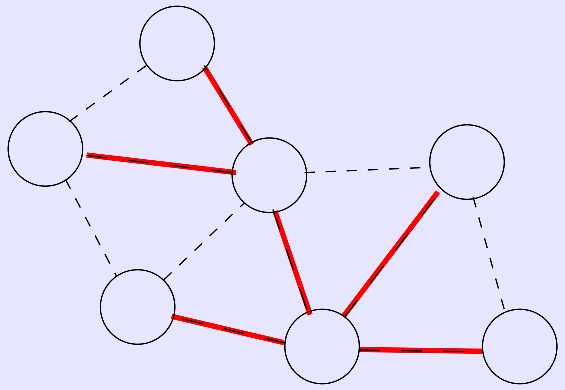
Outline

Connectivity literature

- In (NSBJ06), the authors solve connectivity, but require that one fixed spanning tree remain in the set of connected edges at all times.
- (ZP07b) presents a flocking algorithm which preserves connectivity.
- (dGJ06) presents a distributed connectivity maximization algorithm. Maximizes a better measure of connectivity, but requires a substantial amount of communication per move.
- (Boy06), (ZP05) and (KM06) solve the problem in a centralized manner.
- (YFG$^+$08) builds estimator and (ZP07a) uses market-based approach

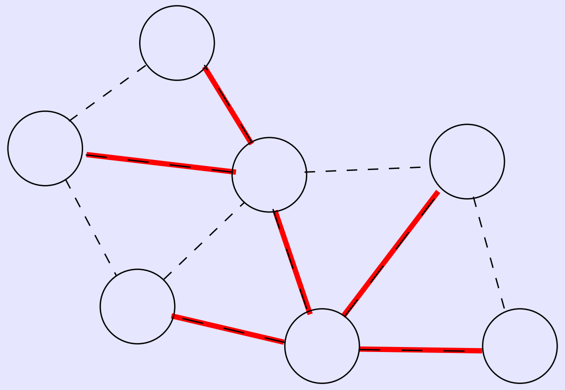
Outline

Spanning tree connectivity(1 of 3)


Proximity graph with spanning tree highlighted in red.

There exist many algorithms which, given a spanning tree, guarantee that a distributed control algorithm doesn't break any spanning tree edges (See Notarstefano et al (NSBJ06)).

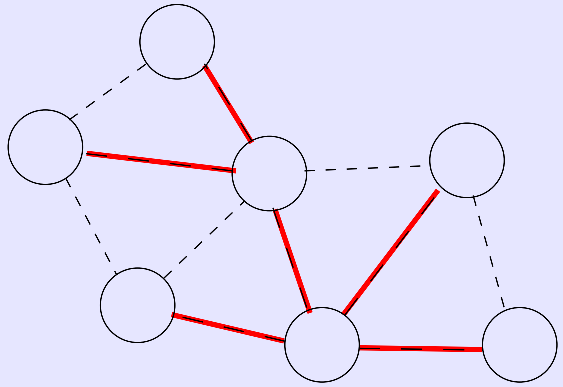
Spanning tree connectivity(2 of 3)



Proximity graph with spanning tree highlighted in red.

We created an algorithm to adjust tree edges according to preferences of another motion coordination algorithm.

Spanning tree connectivity(3 of 3)

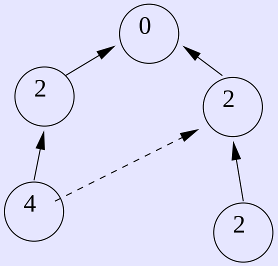

Proximity graph with spanning tree highlighted in red.

Our intent is that by ***coupling our algorithm with another motion coordination algorithm***, we can modify the other algorithm to maintain connectivity.

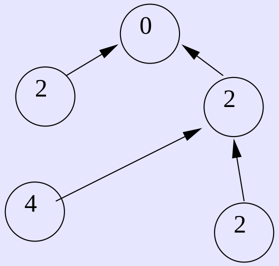
Spanning tree connectivity - Core ideas

- Depth estimate for robot i at round t ($d(i,t)$)
- Depth update: ($d(i,t+1) \leftarrow d(\text{parent}(i),t)+1$)
- Rule for allowed re-arrangements (propose-parent$(i) \leftarrow j$ is allowed if $d(j,t) \leq d(i,t)$.
- Tie-breaking for re-arrangements between robots at same depth If $d(\text{propose-parent}(i),t) = d(i,t)$ and $\exists j$ s.t. $d(j,t) = d(i,j)$ and propose-parent$(j) = i$ and $j < i$ and propose-parent$(i) < i$, then do not connect to parent.

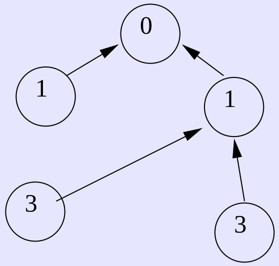
Sample execution



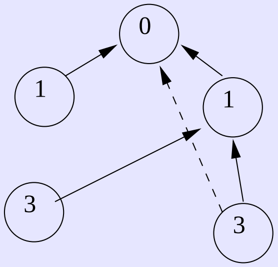
Propose re-arrangements



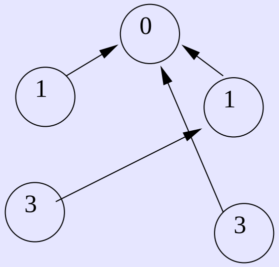
Perform re-arrangements



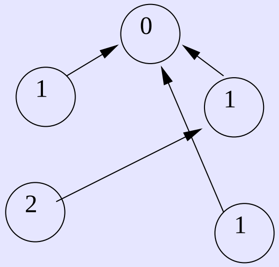
Depth update



Propose re-arrangements



Perform re-arrangements



Depth update

Spanning tree connectivity - preferences

Recall that our algorithm takes re-arrangement preferences from another motion coordination algorithm.

Re-arrangement preferences can take many forms.

- Distance
- Desired topology (sufficient information in $\log(n)$ bits)
- Distance after t time units under unconstrained control action.

Analysis of algorithm

Correctness result in next section.

Proposition: *If any two nodes, i and j, would prefer to be connected to each-other over each of their neighbors, they will do so within one cycle of re-arrangement rounds.*

Outline

(i) Intro

(ii) Our contributions

  (a) **Connectivity Maintenance**

    i. Basic algorithm

    ii. ***Repair***

    iii. Reachability

    iv. Simulation Results

  (b) Algebraic Connectivity Algorithm

(iii) Conclusions / Bibliography

Motivation

Issues with Connectivity Maintenance

- Link failures
- How to initialize tree?
- Can we do “good enough” if graph is severed?
- Correctness result (postponed from last section)

Link failures

Edge in underlying graph ceases to exist (for whatever reason)

If edge, $(i, f_{\mathsf{p}}^{[i]})$, from i to parent fails, remove $(i, f_{\mathsf{p}}^{[i]})$ from constraint tree.

When this happens, i no longer has parent (represent with $f_{\mathsf{p}}^{[i]} \leftarrow i$)

Solution (1 of 2)

Each agent, i, has $n^{[i]}_{\text{root}}$.

If i has parent ($f^{[i]}_{\text{p}}$), set $n^{[i]}_{\text{root}} \leftarrow n^{[f^{[i]}_{\text{p}}]}_{\text{root}}$

If i has no parent, set $n^{[i]}_{\text{root}} \leftarrow i$

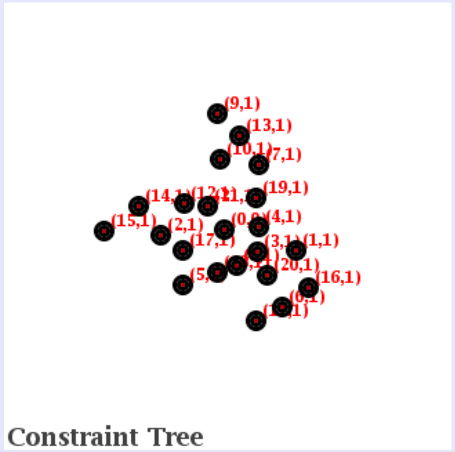
Solution (2 of 2)

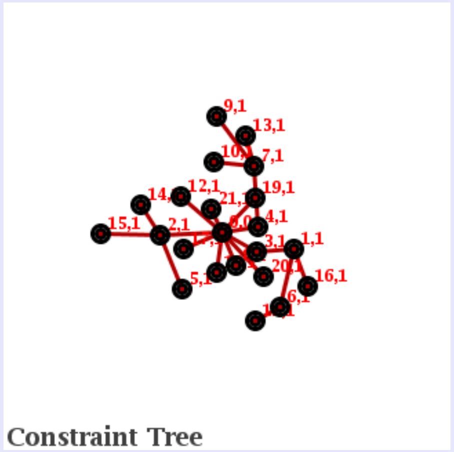
Prefer to attach to agents with smaller $n_{\text{root}}^{[\cdot]}$ values.

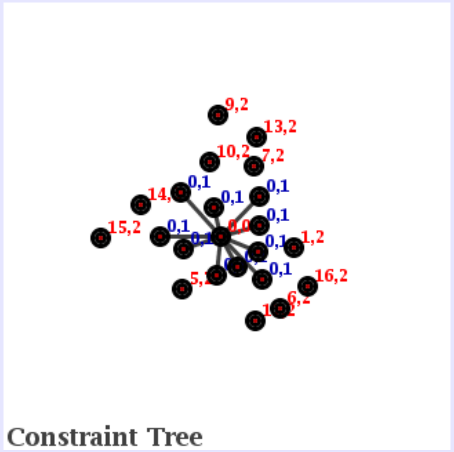
Preserve links between agents with different $n_{\text{root}}^{[\cdot]}$

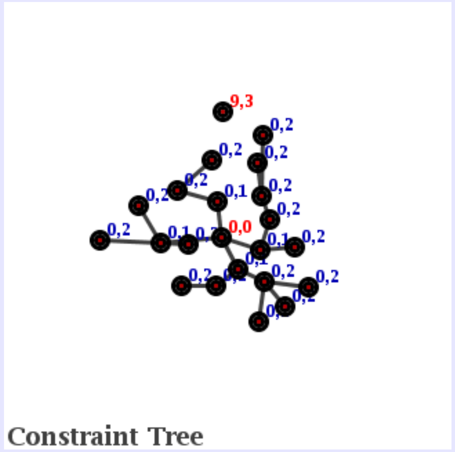
If i has no parent, and $i \neq 0$ ($n_{\text{root}}^{[i]} \neq 0$) increase depth every round.

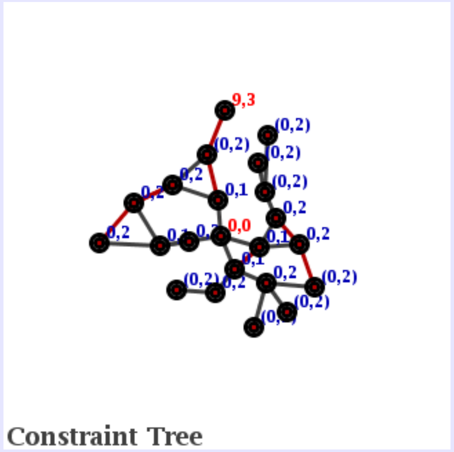
Results (simulation)

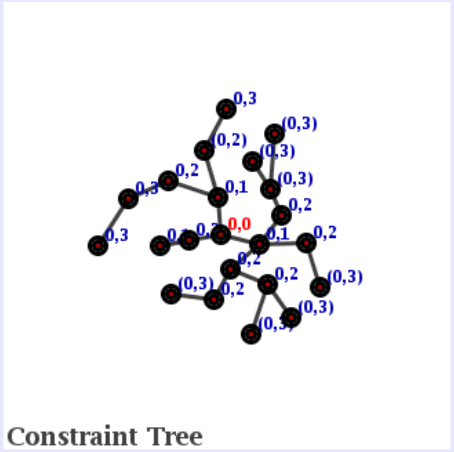












Results (correctness)

Proposition: *Assume the graph induced by $(i, f_p^{[i]}), i \in \mathbb{Z}_n$ starts with k disjoint connected components. Then, at all times during the execution of* CM ALGORITHM*, the graph induced by the parent relation among the agents contains no cycles other than those it started with (and, having n edges, retains at most k disjoint connected components), so long as no edge of the form $(i, f_p^{[i]})$ or $(i, g_p^{[i]})$ disappears from the underlying proximity graph.*

Proposition: *Since no node i with $n_{root}^{[i]} = 0$ prefers to attach to a node j with $n_{root}^{[j]} \neq 0$, if no edges of the form $(i, f_p^{[i]})$ or $(i, g_p^{[i]})$ are removed, the graph of $(i, f_p^{[i]})$ for i having $n_{root}^{[i]} = 0$ remains connected and never decreases in size.*

Results (repair)

Proposition: *There is no distributed repair algorithm which can allow links to break and, at the same time, recover from all possible underlying hardware failures which leave the communication graph connected.*

The next result shows that this algorithm can repair breaks in the spanning tree whenever the underlying graph remains connected.

Proposition: *Let the communication graph of some component, K, of the network remain connected and not connect to any other components. Assume the (not necessarily connected) constraint tree is such that the only cycles are self-loops (i.e., $i = f_{\mathsf{p}}^{[i]}$). Let id$_K$ be the smallest UID of any node in the connected component, and denote the number of agents in the component by n_K. Within id$_K(n+1)+n+n_K$ iterations of* CM ALGORITHM*, every node, $i \in K$, will have $n_{\mathsf{root}}^{[i]} =$ id$_K$.*

Results (dynamic repair)

Proposition: *If the graph starts in any initial state (even if the underlying graph is disconnected), and the evolution of the network follows the constraints (no constraint tree edge or edge between nodes of different $n_{root}^{[\cdot]}$ broken) for $2n$ rounds, then, if the underlying graph becomes connected again, it will stay connected for all time, thus building a connected constraint tree.*

Outline

(i) Intro

(ii) Our contributions

   (a) **CONNECTIVITY MAINTENANCE**

      i. Basic algorithm

      ii. Repair

      iii. ***Reachability***

      iv. Simulation Results

   (b) Algebraic Connectivity Algorithm

(iii) Conclusions / Bibliography

Reachability

How flexible is this algorithm?

Given any target constraint tree, can we reach it?

Reachability Result

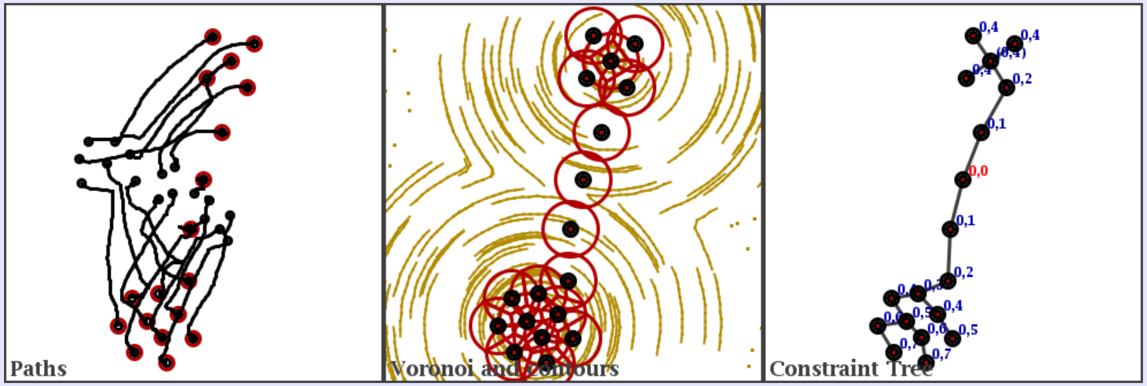
Yes.

Provided the target constraint tree is a subgraph of the communication graph.

Proven in (SC09)

Outline

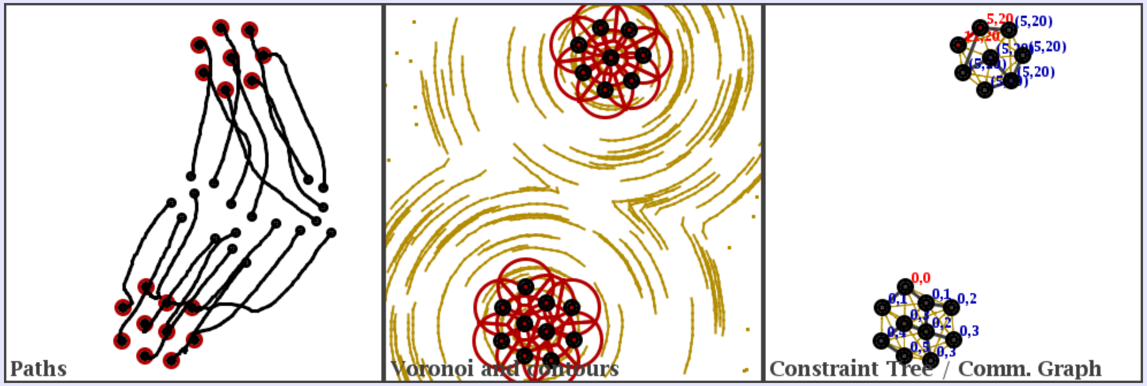
Example 1 : Deployment



Agents

simulation

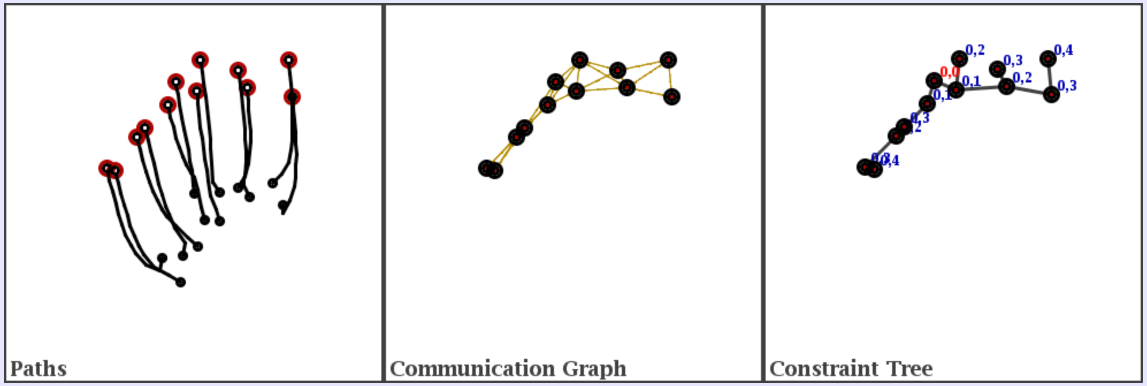
Example 2 : Deployment (no constraints)



Agents

simulation

Example 3 : Flocking



Agents

simulation

Outline

Algebraic Graph Theory

Given a graph, $G = (V, E)$, define the Laplacian matrix, $L(G)$ be the matrix

$$L_{i,j} = \begin{cases} -1 & (i,j) \in E \\ \deg i & i = j \\ 0 & \text{otherwise} \end{cases}$$

If the graph is weighted, i.e. for each $(i,j) \in E$ there is a $w_{i,j} \in \mathbb{R}$, we can define a weighted Laplacian matrix $L(G)$ by

$$L_{i,j} = \begin{cases} -w_{i,j} & (i,j) \in E \\ \sum_{k \neq i} w_{i,k} & i = j \\ 0 & \text{otherwise} \end{cases}$$

The Graph Laplacian

The Laplacian has several nice properties

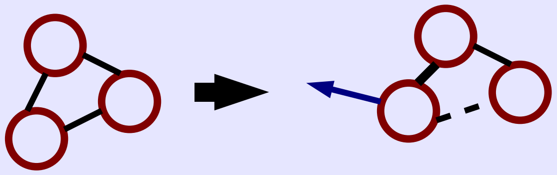
- $L\mathbf{1} = \mathbf{0}$
- The multiplicity of the zero eigenvalue is the number of components of the graph.
- The speed of convergence of common control algorithms for flocking, rendezvous and consensus depend on the second smallest eigenvalue, λ_2 of the Laplacian matrix of the communication graph.

Problem Setup (1 of 2)

Suppose a communication graph for a swarm of robots is weighted, and the weights depend on the relative positions of the two robots sharing a communication link.

Then $\lambda_2(L(G))$ depends on the positions of the robots in the swarm.

An instantaneous motion of a robot creates an instantaneous change in $\lambda_2(L(G))$.



Evolution of graph

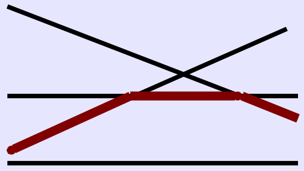
Problem Setup (2 of 2)

Whenver $L(G)$ has a distinct second smallest eigenvalue, gradient of $\lambda_2(L(G))$ with respect to $L(G)$ is $v_2 v_2^T$ where $L(G)v_2 = \lambda_2(L(G))v_2$.

Nonsmooth. Let $f_{\lambda_i}(L)$ map $L \in \text{Sym}(n)$ to $\lambda_i(L)$.

Nonsmooth gradient is:

$$\begin{aligned} f^\circ_{\lambda_i}(M;X) &= \max_{\{v \in \mathbb{S}^n \,:\, Mv = \lambda_i v\}} vv^T \bullet X, \\ \partial f_{\lambda_i}(M) &= \text{co}_{\{v \in \mathbb{S}^n \,:\, Mv = \lambda_i v\}} \{vv^T\}. \end{aligned}$$


Example nonsmooth function

Notation

Quickly

LAP$(n) \subseteq$ Sym(n) is the space of valid Laplacian matrices, i.e. $L \in$ LAP(n) implies $L\mathbf{1} = \mathbf{0}$ and $L_{i,j} \leq 0$ for $i \neq j$.

LAP$_\pm(n)$ is an extension of this space. Lacks the $L_{i,j} \leq 0$ requirement. Rates of change of a Laplacian matrices live in LAP$_\pm(n)$

$A \leq_{\text{LAP}} B$ if and only if $A_{i,j} \geq B_{i,j}$ for all $i \neq j$.
Interval $[A, B]_{\text{LAP}}$ for $A, B \in$ LAP$_\pm(n)$ defined in the natural way.

Prior work

Prior work revolves around finding gradient of f_{λ_2} in space of robot positions and moving in direction of that gradient.

Difficult to do in a distributed fashion. Centralized solutions include (Boy06) and (KM06).

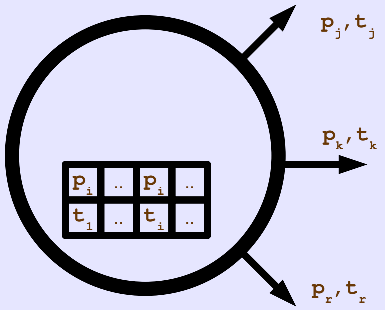
Decentralized solution (dGJ06) follows gradient approach and has problems.

- Communication complexity required to compute eigenvalue
- Nonsmoothness of eigenvalue gradient.

Yang and Freeman – continuous time estimator for gradient Zavlanos and Pappas – discrete algorithm combined with edge constraints

Our solution

- Information dissemination algorithm.



Information Dissemination (all to all broadcast)

- Each robot has bounds on value of Laplacian matrix
- Game against world-picking opponent.

Outline

Our solution (1 of 3)

Game in space of matrices.

Given $A, B \in \text{LAP}(n)$, $A \leq_{\text{LAP}} B$, and $\lambda_+ \in \mathbb{R}$ find a direction $X \in \text{LAP}_\pm(n)$ having $X \bullet V \geq 0$ for every V in the generalized gradient of some $L \in [A, B]_{\text{LAP}}$ having $f_{\lambda_2}(L) \leq \lambda_+$.

Suffices to find $X \in \text{LAP}_\pm(n)$ having $X \bullet (vv^T) \geq 0$ for every v having $Lv = f_{\lambda_2}(L)v$ for some $L \in [A, B]_{\text{LAP}}$ having $f_{\lambda_2}(L) \leq \lambda_+$.

Our solution (2 of 3)

Find set enclosing the set of every v having $Lv = f_{\lambda_2}(L)v$ for some $L \in [A, B]_{\mathsf{LAP}}$ having $f_{\lambda_2}(L) \leq \lambda_+$.

Such a v must have 2 components.

- Component in m lowest eigenvectors for some m having $f_{\lambda_{m+1}}(A) \geq \lambda_+$
- Component in other eigenvectors of a small enough magnitude that multiplying by $f_{\lambda_{m+1}}(A)$ and adding to contribution of other component to Lv keeps Lv under $\lambda_+ v$.

Our solution (3 of 3)

Pick basis for first component, $M_u(m)$. Pick ball radius enclosing second component, $\epsilon_A(m) = \sqrt{\frac{\lambda_+ - \lambda_2(A)}{\lambda_{m+1}(A) - \lambda_2(A)}}$

For a proposed direction in the space of Laplacian matrices, $X \in \text{LAP}_\pm(n)$

- Compute $\min(\text{eigs}(M_u^T(m) X M_u(m)))$ and $\min(\min(\text{eigs}(X)), 0)$.
- If $(1 - \epsilon_A(m)^2) \min(\text{eigs}(M_u^T(m) X M_u(m))) + \epsilon_A(m)^2 \min(\min(\text{eigs}(X)), 0) \geq 0$ direction is "safe"

Actually determines if all Y having $X \leq_{\text{LAP}} Y$ win game.

Outline

(i) Intro

(ii) Our contributions

  (a) CONNECTIVITY MAINTENANCE

  (b) ***Algebraic Connectivity Algorithm***

    i. Algebraic Connectivity

    ii. Key idea / game

    iii. ***Final algorithm***

(iii) Conclusions / Bibliography

Motion Test Algorithm

Given proposed motion by an individual robot, compute lower bound on instantaneous rate of change of Laplacian matrix. If the actual (unknown) rate of change is Y, we want X (known) having $X \leq_{\text{LAP}} Y$.

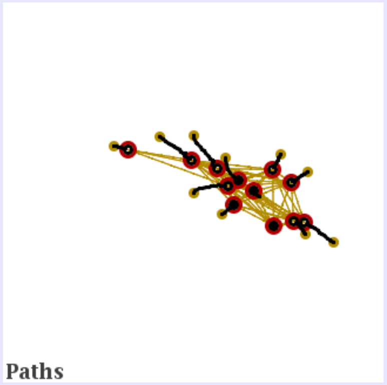
If each robot moves in a direction such that the associated Laplacian rate of change satisifes test from previous slide, $f_{\lambda_2}(L(G))$ does not drop below λ_+.

Motion Projection Algorithm

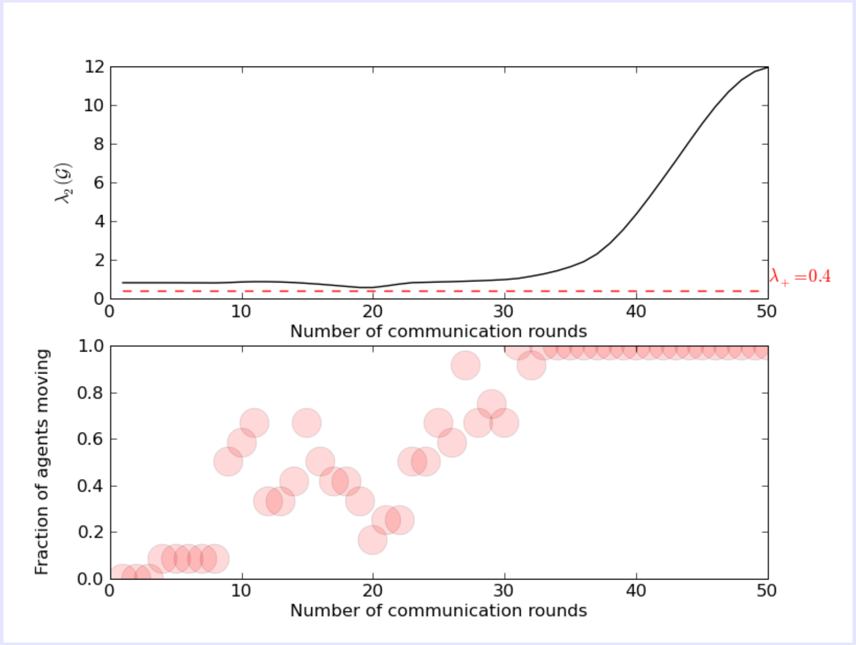
Combine this with a root finder on the space of physical directions of robot motion.

Gives an algorithm which finds valid directions.

Example 1 : Rendezvous



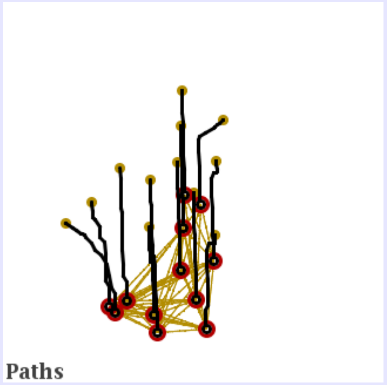
Agents



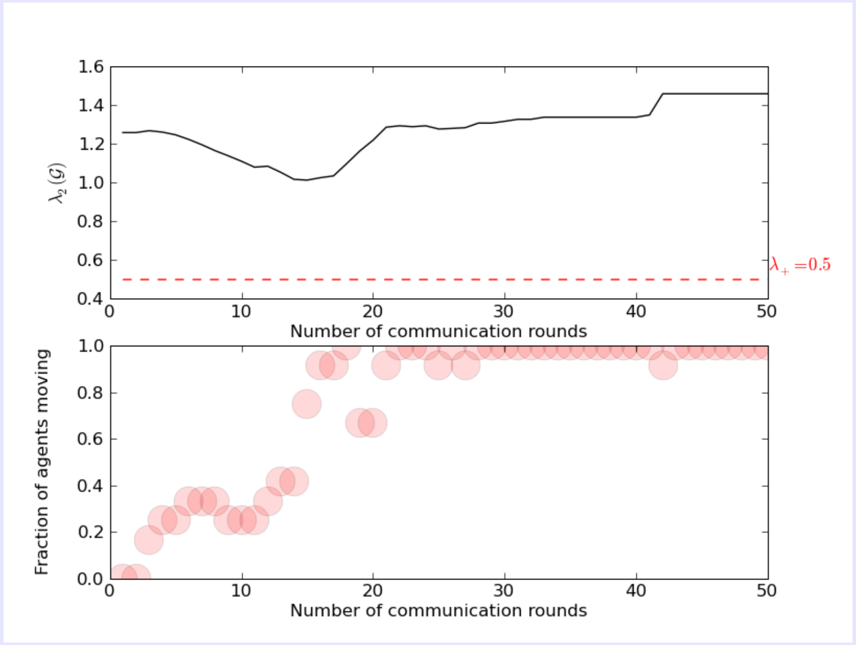
λ_2 and fraction of agents moving

simulation

Example 2 : Flocking



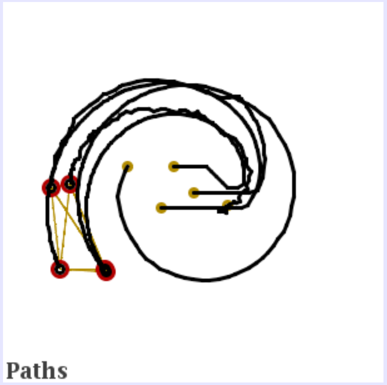
Agents



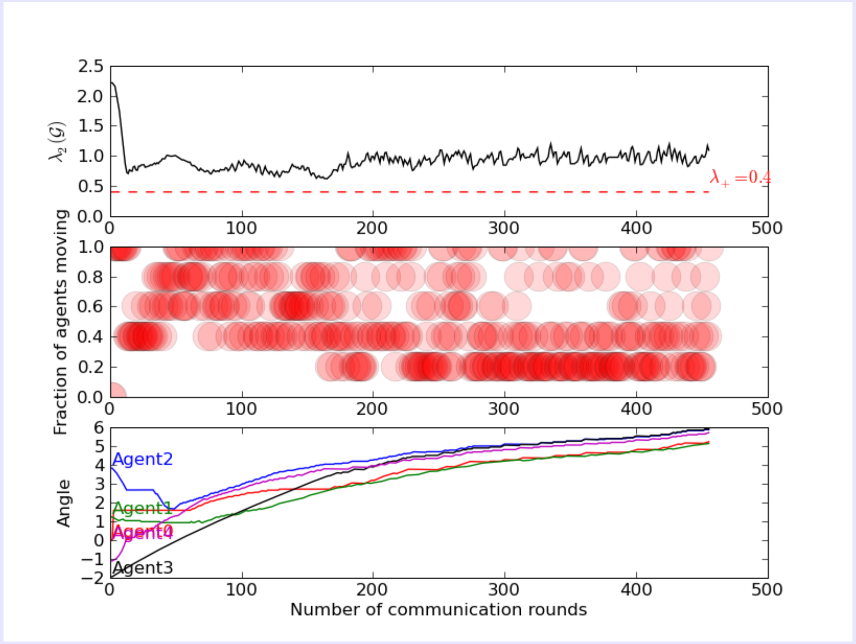
λ_2 and fraction of agents moving

simulation

Example 3 : Multiple control directives



Agents



λ_2 and fraction of agents moving

simulation

Outline

(i) Intro

(ii) Our contributions

(iii) ***Conclusions / Bibliography***

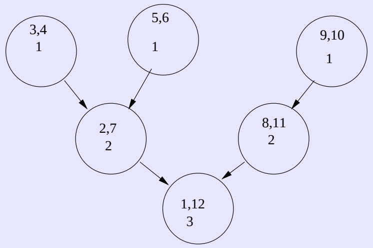
Conclusions

Connectivity constraints are realizable

- In a more flexible setting than previously thought
- Without explicit global transfer of information
- In a manner which can be coupled with a wide set of algorithms
- Multiple approaches.

Work presented in (SC07, SC09, SC08a, SC08b) and (SC06b) see also (SC06a) and (SC06c). Also see (Sch08).

Embedding desired topology in preferences



Run *depth first search* on target spanning tree, for each node, i, mark down

$n_{\textit{first-vst}}(i) =$ ***number of nodes visited before*** i ***was first visited***,

$n_{\textit{last-vst}}(i) =$ ***number of nodes visited before*** i ***was last visited***

and

$d(i) =$ ***depth of*** i.

If $n_{\text{first-vst}}(j) > n_{\text{first-vst}}(i)$ and $n_{\text{last-vst}}(j) < n_{\text{last-vst}}(i)$ then i is the $(j-i)$th ancestor of j.

References

(Boy06) S. Boyd. Convex optimization of graph Laplacian eigenvalues. In *Proceedings of the International Congress of Mathematicians*, pages 1311–1319, Madrid, Spain, August 2006.

(CAS00) G Caprari, K O Arras, and R Siegwart. The autonomous miniature robot alice: from prototypes to applications. In *Intelligent Robots and Systems, 2000*, Takamatsu, Japan, 2000.

(CS) Gilles Caprari and Roland Siegwart. Autonomous micro robotics : Alice.

(dGJ06) M. C. de Gennaro and A. Jadbabaie. Decentralized control of connectivity for multi-agent systems. In *IEEE Conf. on Decision and Control*, pages 3628–3633, San Diego, CA, December 2006.

(KM06) Y. Kim and M. Mesbahi. On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian. *IEEE Transactions on Automatic Control*, 51(1):116–120, 2006.

(MBCF07a) S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks - Part I: models, tasks, and complexity. *IEEE Transactions on Automatic Control*, 52(12):2199–2213, 2007.

(MBCF07b) S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks - Part II: time complexity of rendezvous and deployment algorithms. *IEEE Transactions on Automatic Control*, 52(12):2214–2226, 2007.

(NSBJ06) G. Notarstefano, K. Savla, F. Bullo, and A. Jadbabaie. Maintaining limited-range connectivity among second-order agents. In *American Control Conference*, pages 2124–2129, Minneapolis, MN, June 2006.

(SAB$^+$00) R. G. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *AAAI/IAAI*, pages 852–858, 2000.

(SC06a) M. Schuresko and J. Cortés. Safe graph rearrangements for distributed connectivity of robotic networks. Technical Report ams2006-17, Department of Applied Mathematics and Statistics. University of California, Santa Cruz, September 2006. Available electronically at `http://www.ams.ucsc.edu`.

(SC06b) M. D. Schuresko and J. Cortés. Correctness analysis and optimality bounds of multi-spacecraft formation initialization algorithms. In *IEEE Conf. on Decision and Control*, pages 5974–5979, San Diego, CA, December 2006.

(SC06c) M. D. Schuresko and J. Cortés. Correctness analysis and optimality bounds of multi-spacecraft formation initialization algorithms. Technical Report ams2006-17, Department of Applied Mathematics and Statistics. University of California, Santa Cruz, September 2006. Available electronically at `http://www.ams.ucsc.edu`.

(SC07) M. D. Schuresko and J. Cortés. Safe graph rearrangements for distributed connectivity of robotic networks. In *IEEE Conf. on Decision and Control*, pages 4602–4607, New Orleans, LA, 2007.

(SC08a) M. D. Schuresko and J. Cortés. Distributed motion constraints for algebraic connectivity of robotic networks. In *IEEE Conf. on Decision and Control*, pages 5482–5487, Cancun, Mexico, December 2008.

(SC08b) M. D. Schuresko and J. Cortés. Distributed motion constraints for algebraic connectivity of robotic networks. In *Journal of Intelligent and Robotic Systems*, 2008. Special issue on "Special Issue on Unmanned Autonomous Vehicles." Submitted.

(SC09) M. D. Schuresko and J. Cortés. Distributed tree rearrangements for reachability and robust connectivity. *SIAM Journal on Control and Optimization*, 2009. Submitted.

(Sch08) M. D. Schuresko. CCLsim. a simulation environment for robotic networks, 2008. Electronically available at http://www.soe.ucsc.edu/˜mds/cclsim.

(YFG$^+$08) P. Yang, R. A. Freeman, G. Gordon, K. M. Lynch, S. Srinivasa, and R. Sukthankar. Decentralized estimation and control of graph connectivity for mobile sensor networks. In *American Control Conference*, Seattle, WA, 2008.

(ZP05) M. M. Zavlanos and G. J. Pappas. Controlling connectivity of dynamic graphs. In *IEEE Conf. on Decision and Control*, pages 6388–6393, Seville, Spain, 2005.

(ZP07a) M. M. Zavlanos and G. J. Pappas. Distributed connectivity control of mobile networks. In *IEEE Conf. on Decision and Control*, New Orleans, LA, 2007.

(ZP07b) M. M. Zavlanos and G. J. Pappas. Flocking while preserving network connectivity. In *IEEE Conf. on Decision and Control*, New Orleans, LA, 2007.