Communication Complexity of Multi-Vehicle Systems

ERIC KLAVINS

Computer Science
California Institute of Technology

with: Jason Hickey and Richard Murray

A Theory of Decentralized Control Systems

(Gleaned from the LPE Reading Group)

= Control + Computer Science

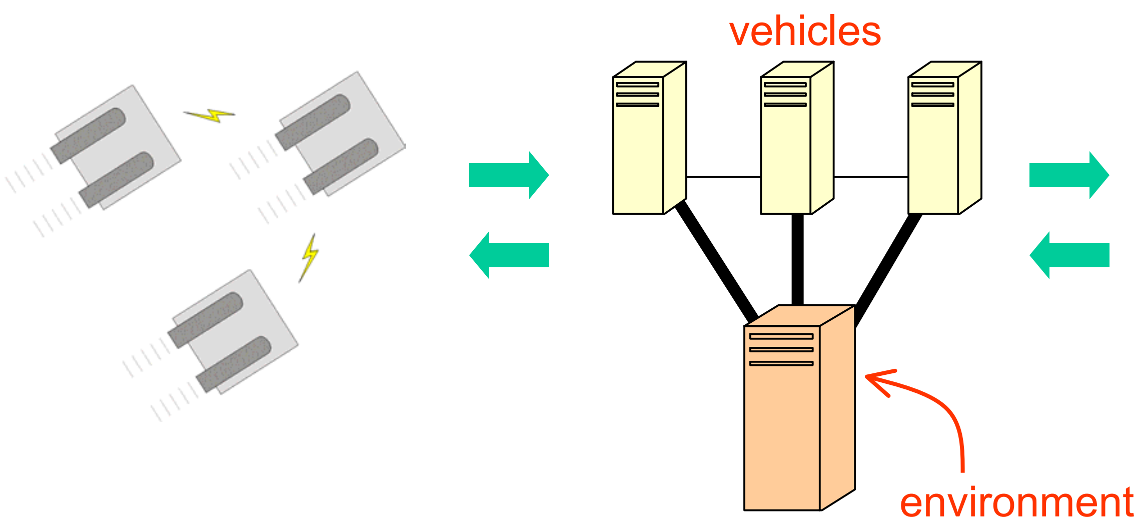
- Complexity and Scalability
- Combinatorics/Graph Theory
- Logic, Specification and Formal Methods
- Networking and Communications

Outline

- Specification of decentralized control systems
- Communication complexity
- Schemes with various complexities
- Related work: (Semi) Automatic Verification

Decentralized Control as Parallel Processing

Multi-vehicle system *Parallel Processing System* *DRL Specification*



$$\Pi = \Pi_{veh}(1) \circ \Pi_{veh}(2) \circ \Pi_{veh}(3) \circ \Pi_{dyn}$$

DRL [Klavins&Hickey, Submitted to CDC 2002] is language for specifying and reasoning about parallel control systems. Based on *UNITY* [Chany&Misra, 1990] .

A Comparison of Formalisms

Hybrid Automata Based Modeling	DRL
•Continuous/Discrete	•Discrete
•Simple Dynamics	•Arbitrary Dynamics
•State Based/Model Checking	•Symbolic Theorem Proving
•Small Systems	•Potentially large (homogenous) systems
	•Domain Knowledge and Reuse

Modeling Dynamical Systems in DRL

Initial (I): $t = 0$ ← Defines the initial conditions of the system

Clauses (C):

$$x_1 > 0 \; : \; u'_1 < 0$$
$$x_1 < 0 \; : \; u'_1 > 0$$
$$true \; : \; u'_2 = -k(x_2 - x_1)$$

Dynamics (Δ):

$$true : t' = t + \delta \; \wedge \; \forall i \; ||x'_i - (x_i + \delta u_i)|| < \varepsilon$$

Modeling Dynamical Systems in DRL

Initial (I): $t = 0$

Clauses (C):

A set of *clauses* defines the program or controller. The rules may be nondeterministic.

$$x_1 > 0 \ : \ u_1' < 0$$
$$x_1 < 0 \ : \ u_1' > 0$$
$$true \ : \ u_2' = -k(x_2 - x_1)$$

Dynamics (Δ):

$$true : t' = t + \delta \ \wedge \ \forall i \ ||x_i' - (x_i + \delta u_i)|| < \varepsilon$$

Modeling Dynamical Systems in DRL

Initial (I): $t = 0$

Clauses (C):

$$x_1 > 0 \; : \; u_1' < 0$$
$$x_1 < 0 \; : \; u_1' > 0$$
$$true \; : \; u_2' = -k(x_2 - x_1)$$

These clauses are owned by agent 1

This clause are owned by agent 2

Dynamics (Δ):

$$true : t' = t + \delta \; \wedge \; \forall i \; ||x_i' - (x_i + \delta u_i)|| < \varepsilon$$

Modeling Dynamical Systems in DRL

Initial (I): $t = 0$

Clauses (C):

$$x_1 > 0 \ : \ u_1' < 0$$
$$x_1 < 0 \ : \ u_1' > 0$$
$$true \ : \ u_2' = -k(x_2 - x_1)$$

← This clause refers to a variable owned by agent 1, therefore incurs a cost of 1 every time it is applied.

Dynamics (Δ):

$$true : t' = t + \delta \ \wedge \ \forall i \ ||x_i' - (x_i + \delta u_i)|| < \varepsilon$$

Modeling Dynamical Systems in DRL

Initial (I): $t = 0$

Clauses (C):

$$x_1 > 0 \; : \; u'_1 < 0$$
$$x_1 < 0 \; : \; u'_1 > 0$$
$$true \; : \; u'_2 = -k(x_2 - x_1)$$

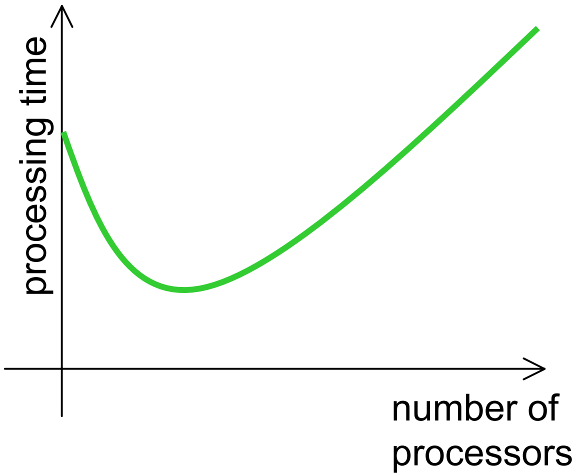
Dynamics (Δ): ← This clause models the environment. Note the nondeterminism.

$$true : t' = t + \delta \; \wedge \; \forall i \; ||x'_i - (x_i + \delta u_i)|| < \varepsilon$$

Scalability

- Scalability depends on computation and coordination
- "Coordination complexity" measures how much each agent relies on other agents
- Communication complexity is a surrogate for this
- Bad: $O(n^2)$. Good: $O(n)$

The drag about parallel computation:



But this isn't the case for all tasks.

What is the analogous way to think about decentralized control systems?

Communication Complexity

Fix $\Pi = (I, C, \Delta)$. Suppose each clause $c \in C$ has cost $\gamma(c) \in \mathbb{N}$.

The **communication complexity** of a state s is

$$cc(s) = \sum_{c \in C \wedge c.g(s)} \gamma(c).$$

The **communication complexity** of an execution $\{s_k\}$ of Π is

$$cc(\{s_k\}) = \lim_{T \to \infty} \frac{1}{T} \sum_{k=1}^{T} cc(s_k).$$

The **communication complexity** of Π is given by

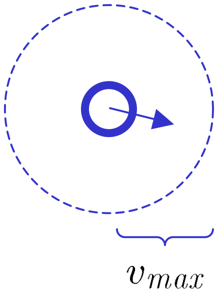
$$cc(\Pi) = \max_{\{s_k\} \in \mathcal{E}(\Pi)} cc(\{s_k\}).$$

Task 1: Full Communication

Define Π_{dyn} to have the clause

$$true \; : \; t' = t + \delta \wedge \forall i \; (\; ||x_i' - x_i|| < \delta v_{max} \;)$$

Suppose the task is to maintain the propertiy



$$||e_{i,j} - x_j|| < \varepsilon$$

Initial (I):

$t = 0 \wedge \forall i, j \; (\; e_{i,j} = x_j \wedge l_{i,j} = 0 \;)$

Clauses (C): For all $i \neq j$, include the clause $c_{i,j}$

$(t + \delta - l_{i,j}) v_{max} \geq \varepsilon \; : \; e'_{i,j} = x_j \; \wedge \; l'_{i,j} = t$ $\quad \gamma(c_{i,j}) = 1$

Thm 1: Each clause is applied every $\tau \triangleq \frac{1}{\delta} \lfloor \frac{\epsilon}{v_{max}} - \delta \rfloor$ steps. Thus, the cc is $\frac{1}{\tau} n(n-1) = O(n^2)$.

Task 2: Communication with Neighbors

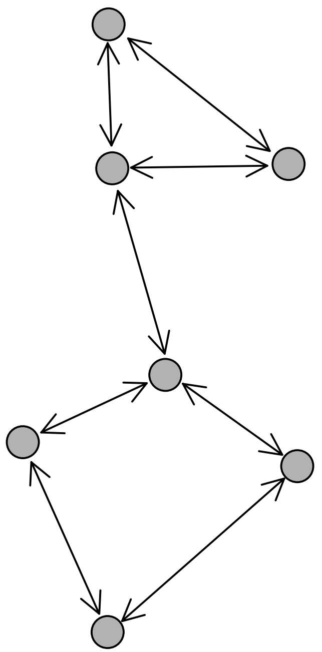
New task. Maintain

$$(i,j) \in E \;\Rightarrow\; ||e_{i,j} - x_j|| < \varepsilon$$

For all i, j such that $(i,j) \in E$, let $c_{i,j}$ be

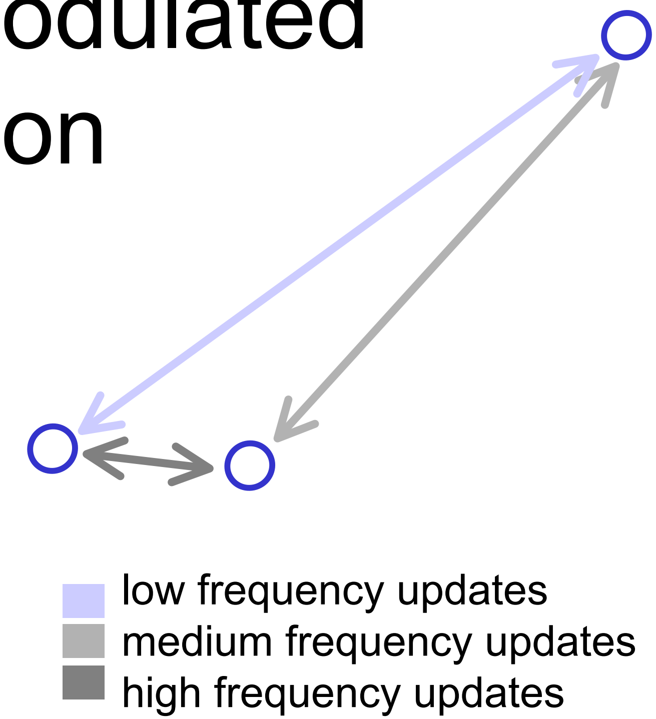
$$(t + \delta - l_{i,j})v_{max} \geq \varepsilon \;:\; e'_{i,j} = x_j \;\wedge\; l'_{i,j} = t$$

Thm 2: If the maximal degree of (V,E) is constant for any n, then $cc(\Pi) = O(n)$.



Adjacency relation is constant.

Task 3: Distance Modulated Communication



New task. Maintain the property

$$||e_{i,j} - x_j|| < k||x_i - x_j||$$

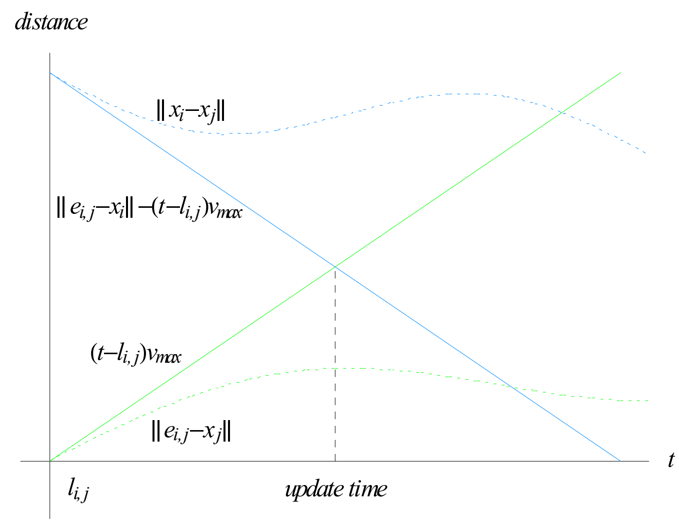
Initial (I):

$$\forall i, j \; (\; e_{i,j} = x_j \wedge l_{i,j} = 0 \wedge t = 0 \;)$$

Clauses (C): For each $i \neq j$ add the clauses $c_{i,j}$

$$(t - l_{i,j})v_{max} \geq k \left(||e_{i,j} - x_i|| - (t - l_{i,j})v_{max} \right) :$$

$$e'_{i,j} = x_j \wedge l'_{i,j} = t$$

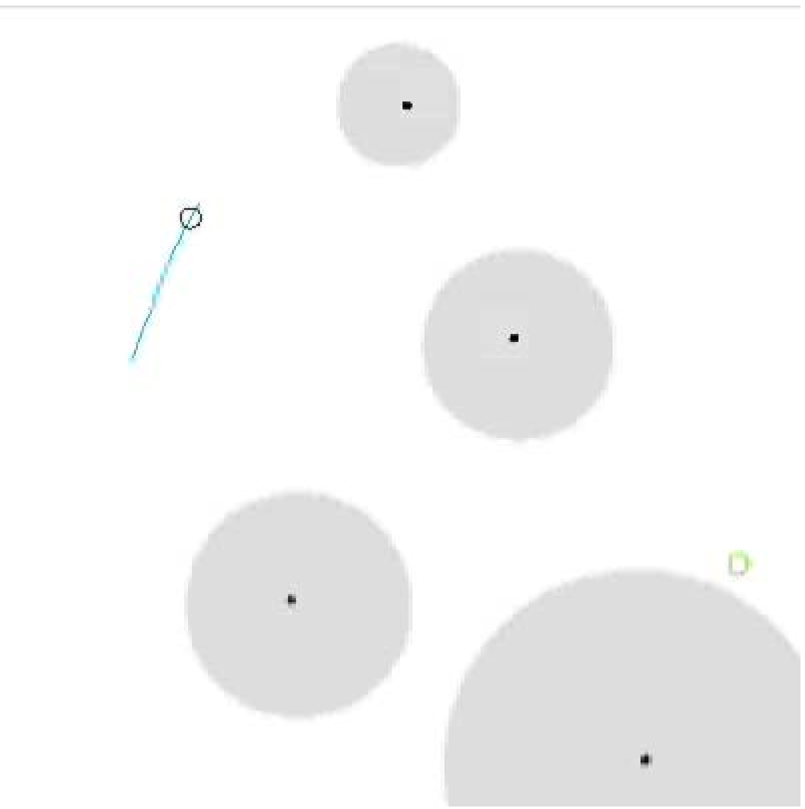


How to Use DMC

Using $e_{i,j}$ and $l_{i,j}$ and knowing v_{max}

1) Define $O_{i,j}$ to be the set reachable by agent j before the next communication event.

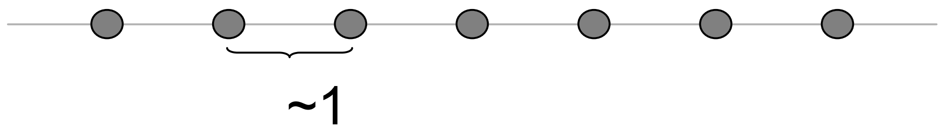
2) Plan around these sets to your goal.



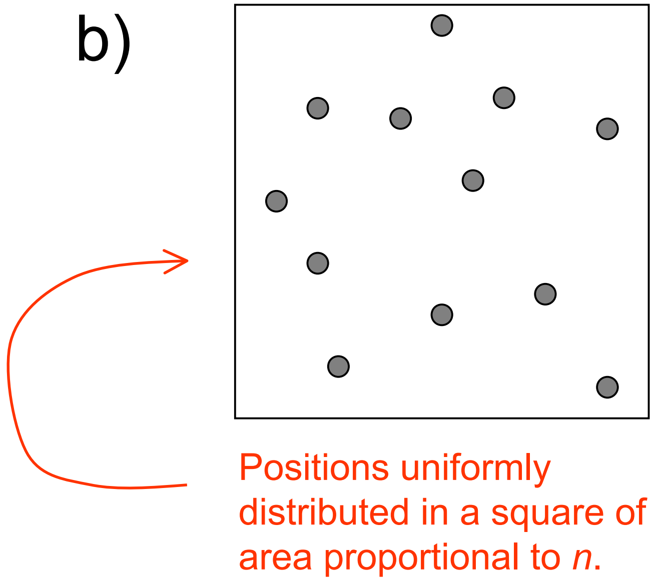
Complexity of DMC

The communication complexity of this algorithm depends on how the vehicles are arranged.

a)



b)



Thm. 3: If the vehicles are regulated to be (a) equispaced on a straight line, then $cc(\Pi_{DMC}) = O(n \log n)$. If they are regulated to be (b) uniformly distributed in a square of area n/ρ then $cc(\Pi_{DMC}) = O(n^{1.5})$.

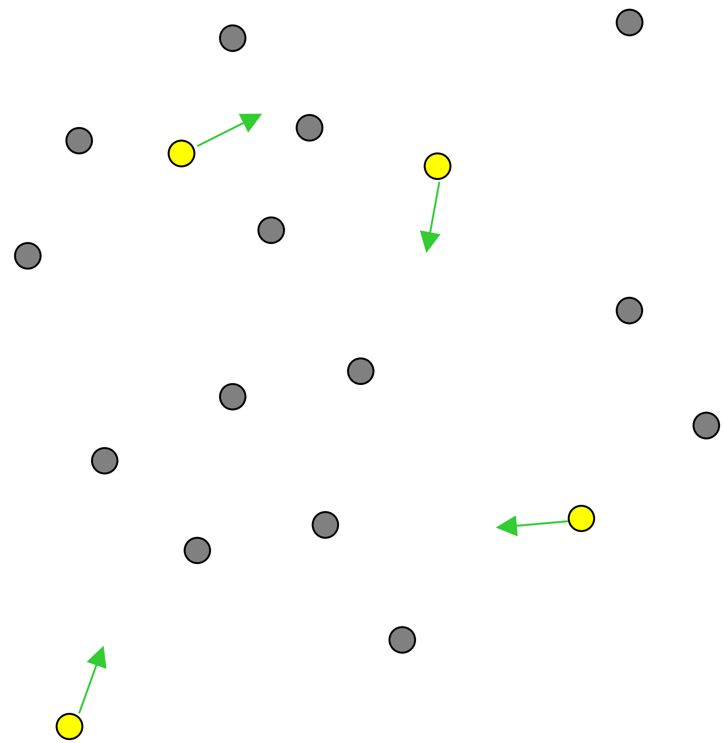
Task 4: Wanderer Communication Scheme

The task is still to maintain $||e_{i,j}-x_j|| < \varepsilon$, but only for a constant number of “wandering” vehicles.

•“Wanderers” must have accurate estimates of other’s positions.

•Fixed vehicles can be ignorant.

•Wanderers may hand their right to move to a fixed vehicle.



WCS Algorithm

Initial (I):

$$\forall i[(i \leq \kappa \rightarrow q_i = 1) \wedge (i > \kappa \rightarrow q_i = 3)$$

$$\wedge r_i = \perp \wedge \forall j(e_{i,j} = x_j)]$$

q_i=1: wandering
q_i=2: uploading
q_i=3: fixed

Clauses (C): For each $i \in \{1, ..., n\}$ define three clauses

Clause	cost
$q_i = 1 : \forall j(q_j \neq 3 \rightarrow e'_{i,j} = x_j)$	κ-1
$q_i = 1 \wedge coin(p,t) : q'_i = 2$	0
$q_i = 2 : q'_i = 3 \wedge r'_i = \perp \wedge \exists j[q_j = 3 \wedge q'_j = 1 \wedge r'_j = i \wedge \forall k(e'_{j,k} = e_{i,k})]$	n

WCS Algorithm

Dynamics (Π_{dyn}):

$$true : t' = t + \delta \wedge \forall i[(q_i = 1 \rightarrow ||x'_i - x_i|| \leq \delta v_{max}$$

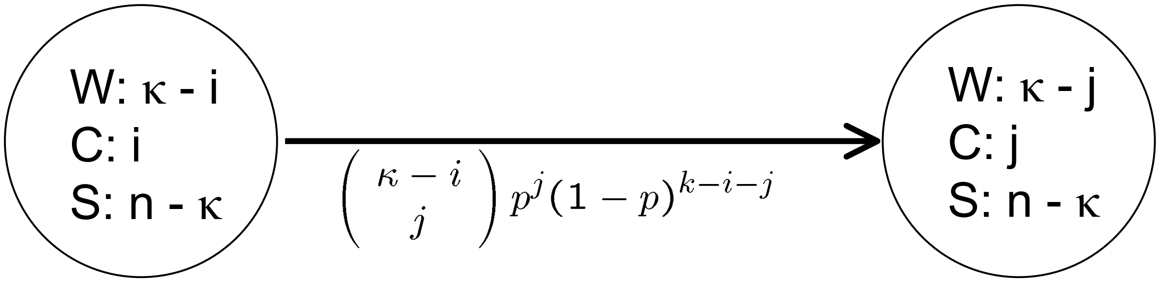
$$\wedge(q_i \neq 1 \rightarrow x'_i = x_i)]$$

Thm. 4: If the value of p in $\Pi_{WCS}(n)$ is constant, then

$$E[cc(\Pi_{WCS}(n))] = O(n)$$

and if $p \triangleq 1/n$ then

$$E[cc(\Pi_{WCS}(n))] = O(1)$$



Other Questions

- Define a “power aware” cost $p(c) = ||x_i - x_j||^2 \gamma(c)$.

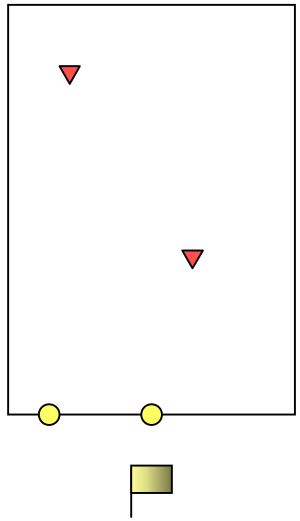
  - What are the efficient communication schemes?
  - Hops are better than direct (unlike with normal CC).

- Sensing costs and tradeoffs

- Incorporating control: $\Pi_{dyn} \circ \Pi_{com} \circ \Pi_{control}$

Current Work: (Semi) Automatic Verification

The RoboFlag Drill



$\Pi_{opp} = (I, C)$ where

$$I \equiv \forall i \in \mathbb{N}\ (b(i).y >= 0 \\ \wedge\ b(i+1).y > b(i).y + \delta v \\ \wedge\ b(i).x \in [0, max])$$

States that each opponent is above the line and in the playing field [0,max]. Also states that opponents are separated vertically (a convenience).

$$C = \{\ true\ :\ b` = \lambda i.(q(i).x, q(i).y - \delta v)\ \}$$

and

$$q = \textbf{if}\ b(0).y - \delta v < 0\ \textbf{then}\ rest(b)\ \textbf{else}\ b$$

The opponents are modeled as a sequences of points. The new value of b is obtained from the old value by decreasing each y coordinate and throwing out the first element if it has crossed the line.

Adding Defenders

$\Pi_{def}(k) = (I, C)$ where

$$I \equiv x_k \in [0, max] \wedge y_k = 0$$

$$C = \{\ true\ :\ x'_k = x_k + \delta u_k\ \}$$

$\Pi_1 \circ \Pi_2 = (I_1 \wedge I_2, C_1 \cup C_2)$

A system with n defenders is given by

$$\Pi(n) = \Pi_{opp} \circ \Pi_{def}(1) \circ \ldots \circ \Pi_{def}(n)$$

The goal is to define control specifications such that

$$\Pi(n) \circ \Pi_{control}(1) \circ \ldots \circ \Pi_{control}(n)$$

has

$$b(i).y \in B_{\varepsilon_1}(0) \Rightarrow \exists k(||x_k - b(i).x|| < \varepsilon_2)$$

as an invariant.

i.e. we want that anytime an opponent crosses the line, there is a defender near it.

Toward a DRL Verification Assistant using Isabelle [Paulson et al., 1994]