ENGI 128

INTRODUCTION TO ENGINEERING SYSTEMS

Lecture 18: Communications Networks and Distributed Algorithms

“Understand Your Technical World”

Using Communications

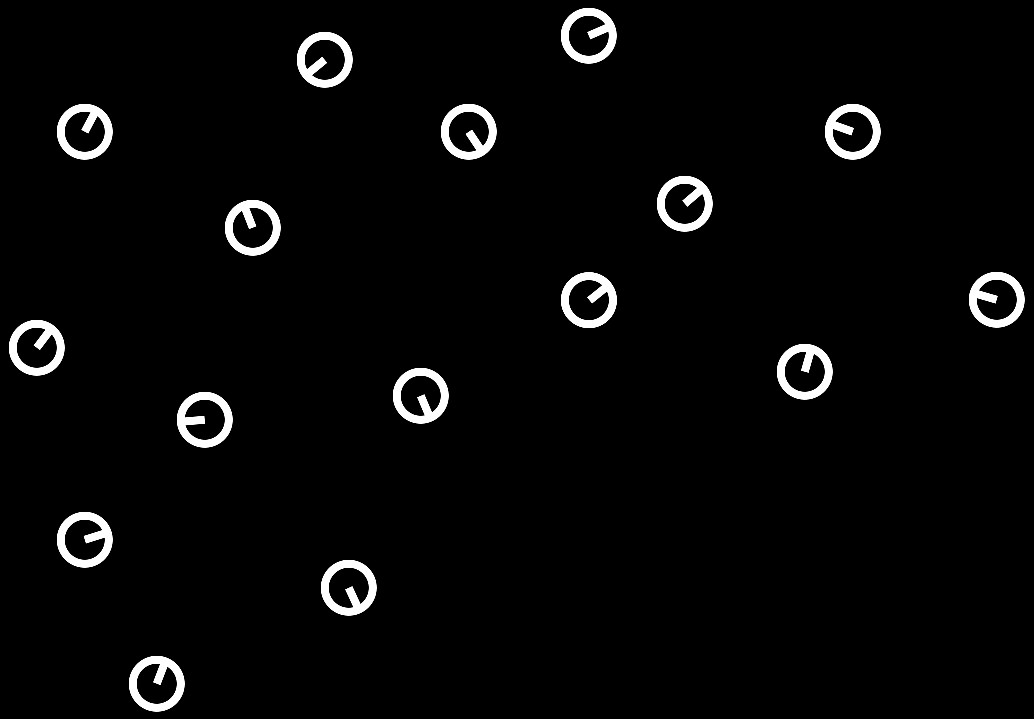
The robot

A robot is too complicated to reason (do math) about.

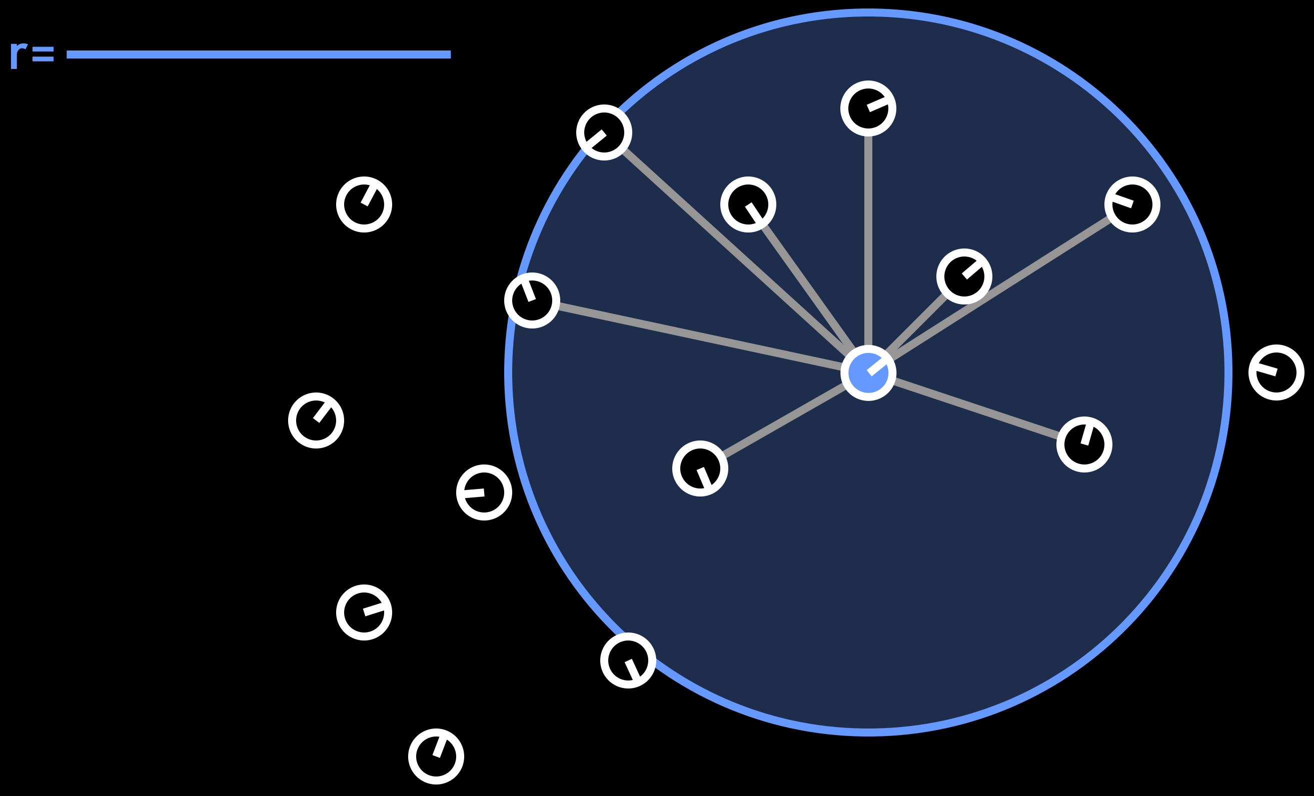
We need to abstract this to a simple model

Computation Model

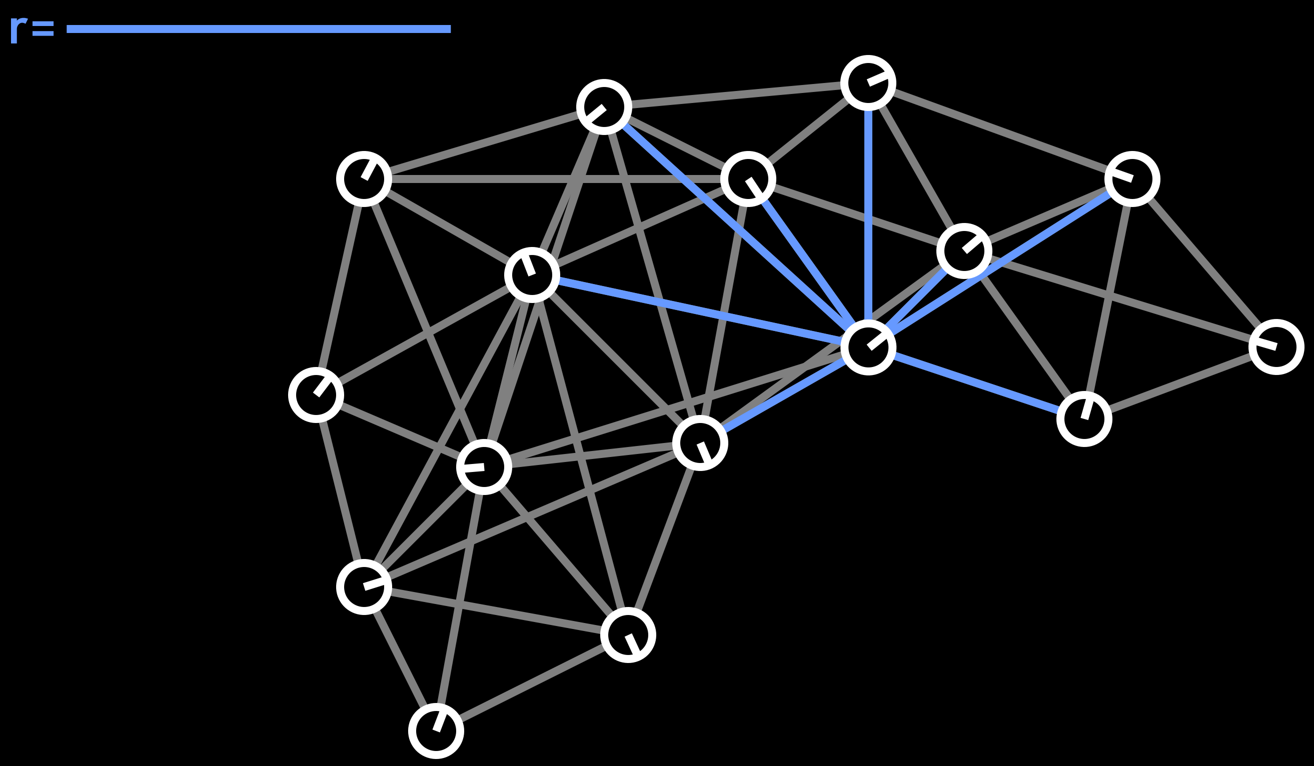
Computation Model



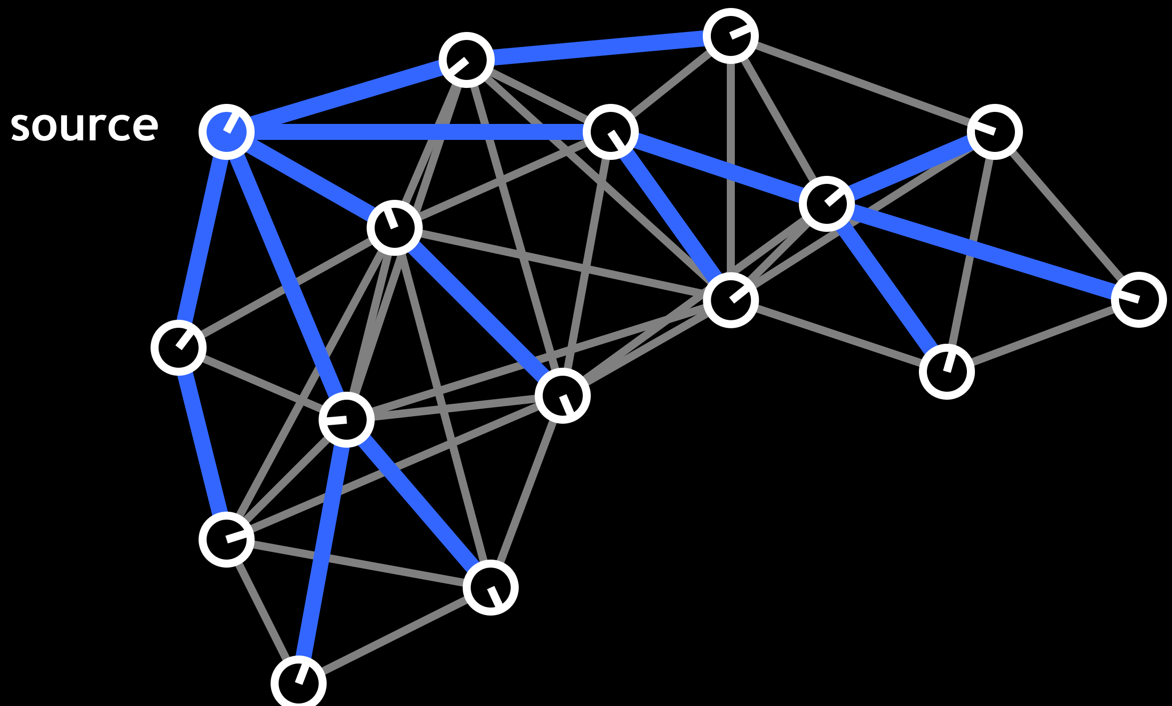
Local Communications



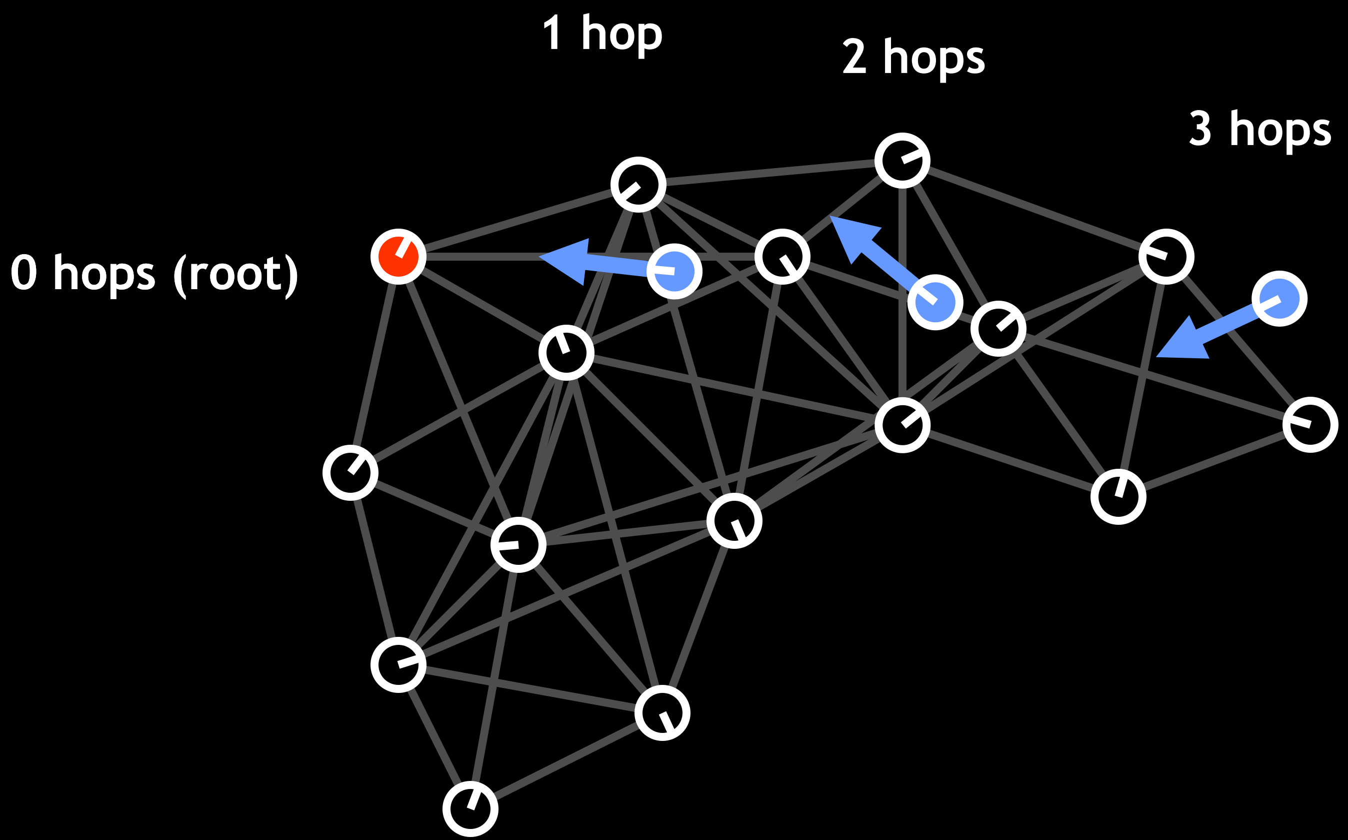
Global Network



Multi-Hop Message Broadcast: Building a “Tree”

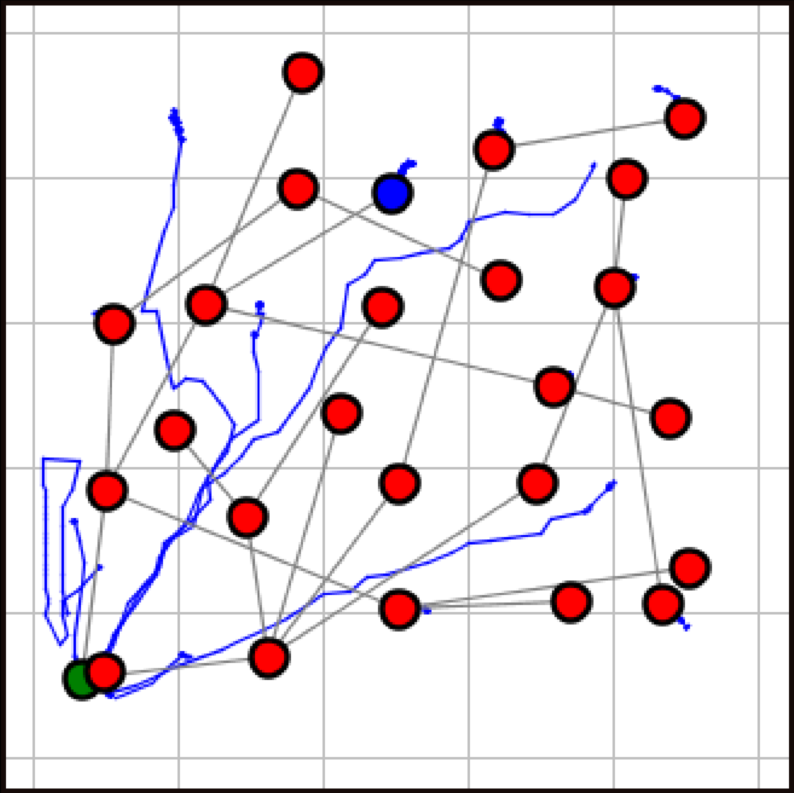


Broadcast Tree Navigation



Broadcast Tree Navigation

n = 32, RSR = 0



Li and Rus, *Navigation Protocols in Sensor Networks*, 2005
Batalin, Sukhatme, and Hattig, *Mobile Robot Navigation using a Sensor Network*, ICRA2004

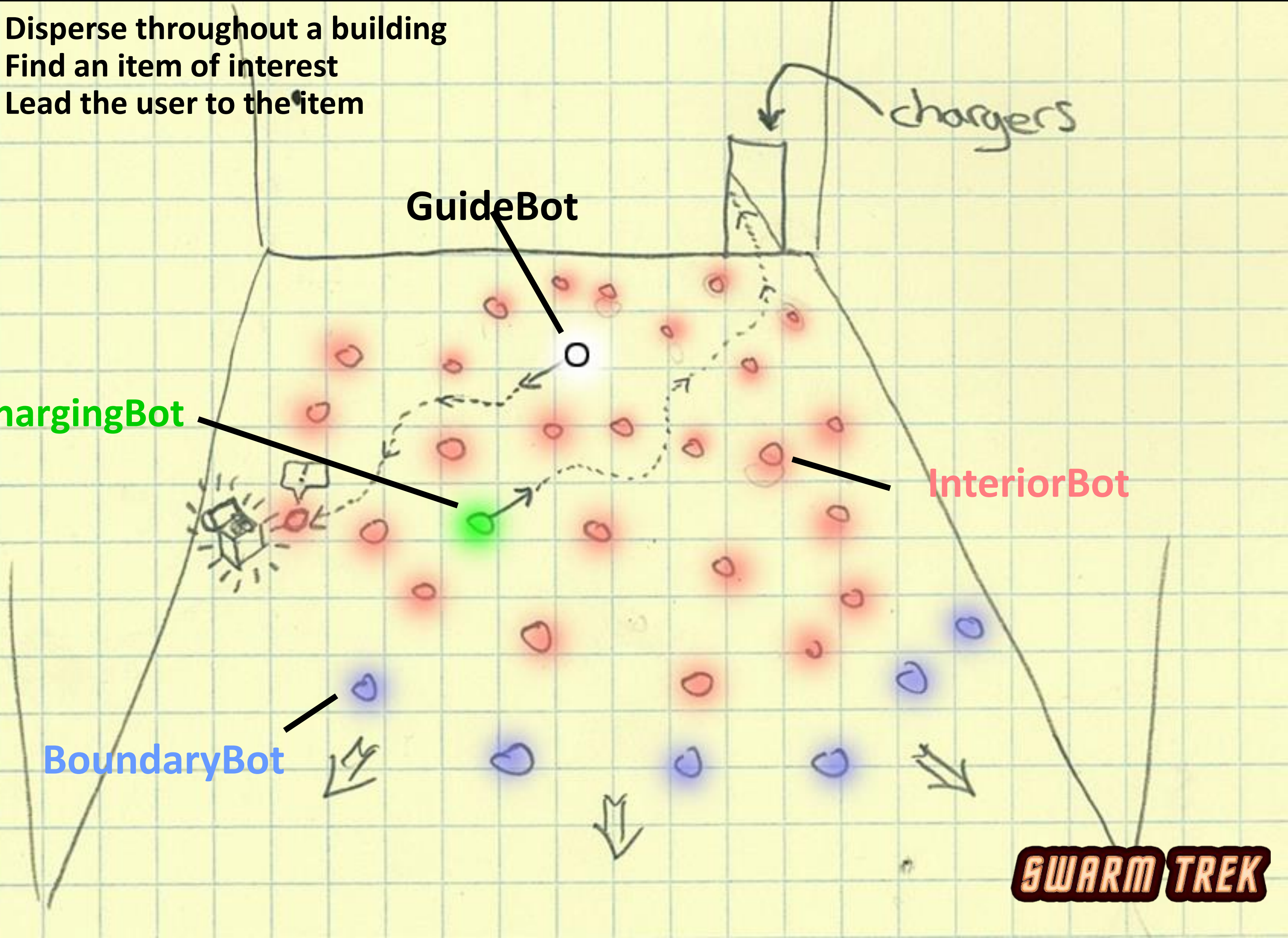
Message Speed

$s_{message}$ (computed)= 3.66 m/s
$s_{message}$ (measured)= 3.64 m/s

n = 44, t = 0.250s, speed = 0, RSR = 0

An Example Application: Building Search

1. Disperse throughout a building
2. Find an item of interest
3. Lead the user to the item




Thu Jan 29 18:49:06 2004


Thu Jan 29 13:15:44 2004

Distributed Algorithms: Consensus and Agreement

Whoa. That's a lot of big words

"Distributed Algorithms: Agreement and Consensus"

Algorithm?

Distributed?

Agreement?

Consensus?

Algorithm?

Algorithm:

A procedure for getting something done

- Input Data
- Procedure
- Output Data

The output and execution has provable properties:

- How long will it take?
- How accurate will the output be?
- How much computer power will I need?

Distributed: To run on many computers

- Like the internet
- Our your nervous system
- Our lots of little robots

Distributed?

Distributed:

To run on many computers

- Like the internet,
- or your nervous system,
- or lots of little robots

Communications is key

But you can't share all the data

(This would take too much communications)

You have to pick very carefully

Consensus?

Consensus:

Consensus is an algorithm to get computers to agree on something

Formal definition:

- All computers agree on a quantity
- All computers know that they agree
- All computers eventually finish within a fixed (bounded) time

Let's do some consensus:

Swarm School: Consensus

SWARM

Consensus #1: Leader Election

We want to elect a leader in this classroom with the following properties

- All students agree on the leader
- All students know that they agree
- All students eventually finish within a fixed (bounded) time

Constraints:

- You can only talk to people you can touch
- You have to whisper

Ideas?

Leader election on the r-one robots

They do this all the time

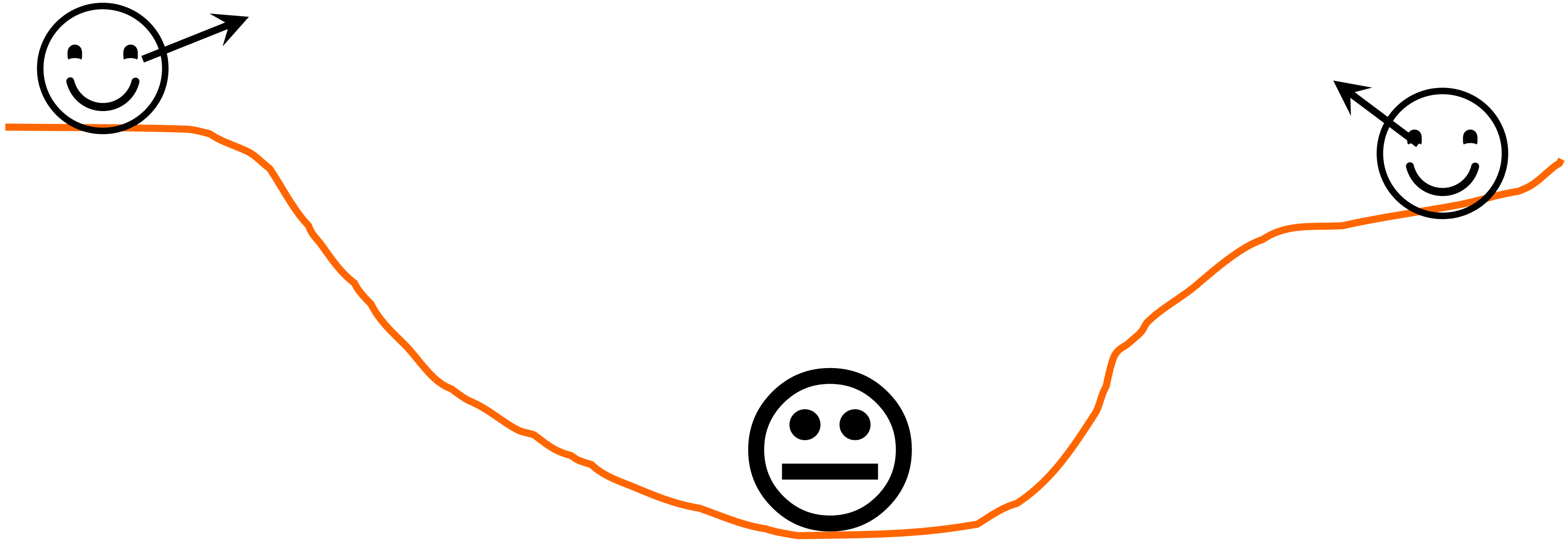
This is how they select a leader for follow-the-leader

Consensus #2: The Byzantine Generals Problem

Two Generals need to attack at the same time, or be defeated

They can send messengers back and forth, but the messengers might not make it.

Can they agree on a time to attack?

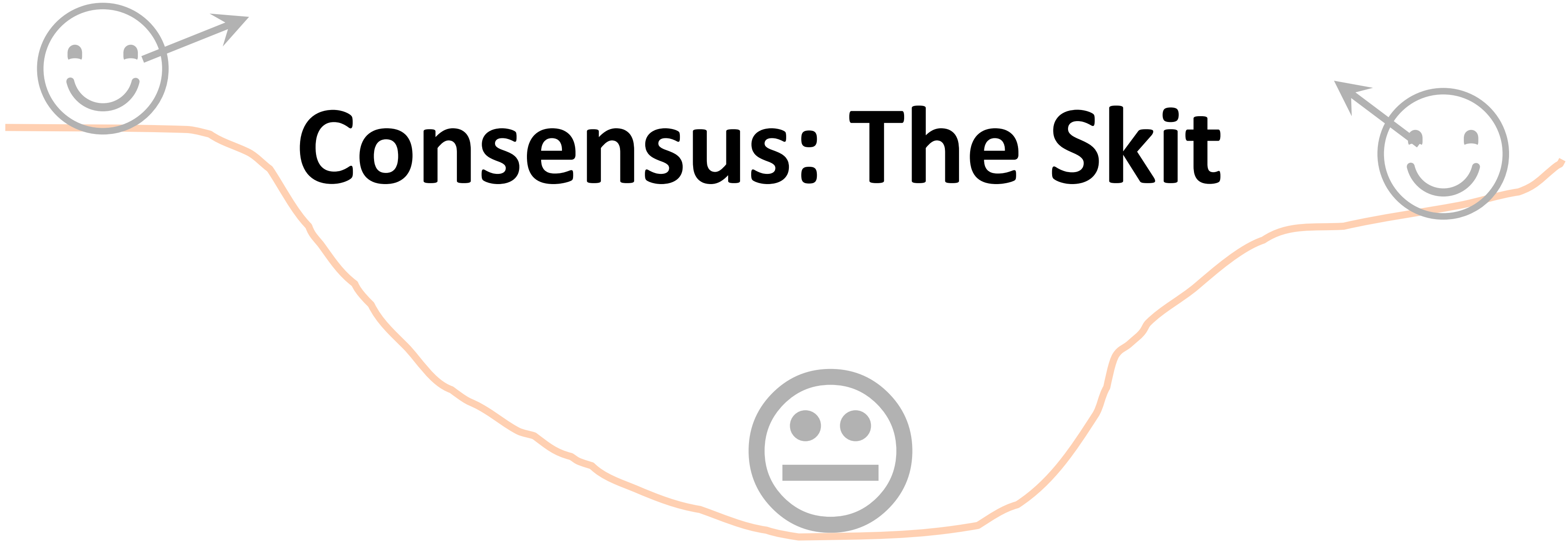


Consensus #2: The Byzantine Generals Problem

Two Generals need to attack at the same time, or be defeated

They can send messengers back and forth, but the messengers might not make it.

Can they agree on a time to attack?



Consensus: The Skit

Are you serious, it really doesn’t work?

Nope. Consensus is not possible with faulty communications

- One of the most famous results in distributed algorithms
- This is called an “Impossibility Proof”

But what about my bank records?

- Databases deal with this by using “transactions” and the concept of “rollback”

Ok, Consensus stinks. Agreement is better, right?

Agreement?

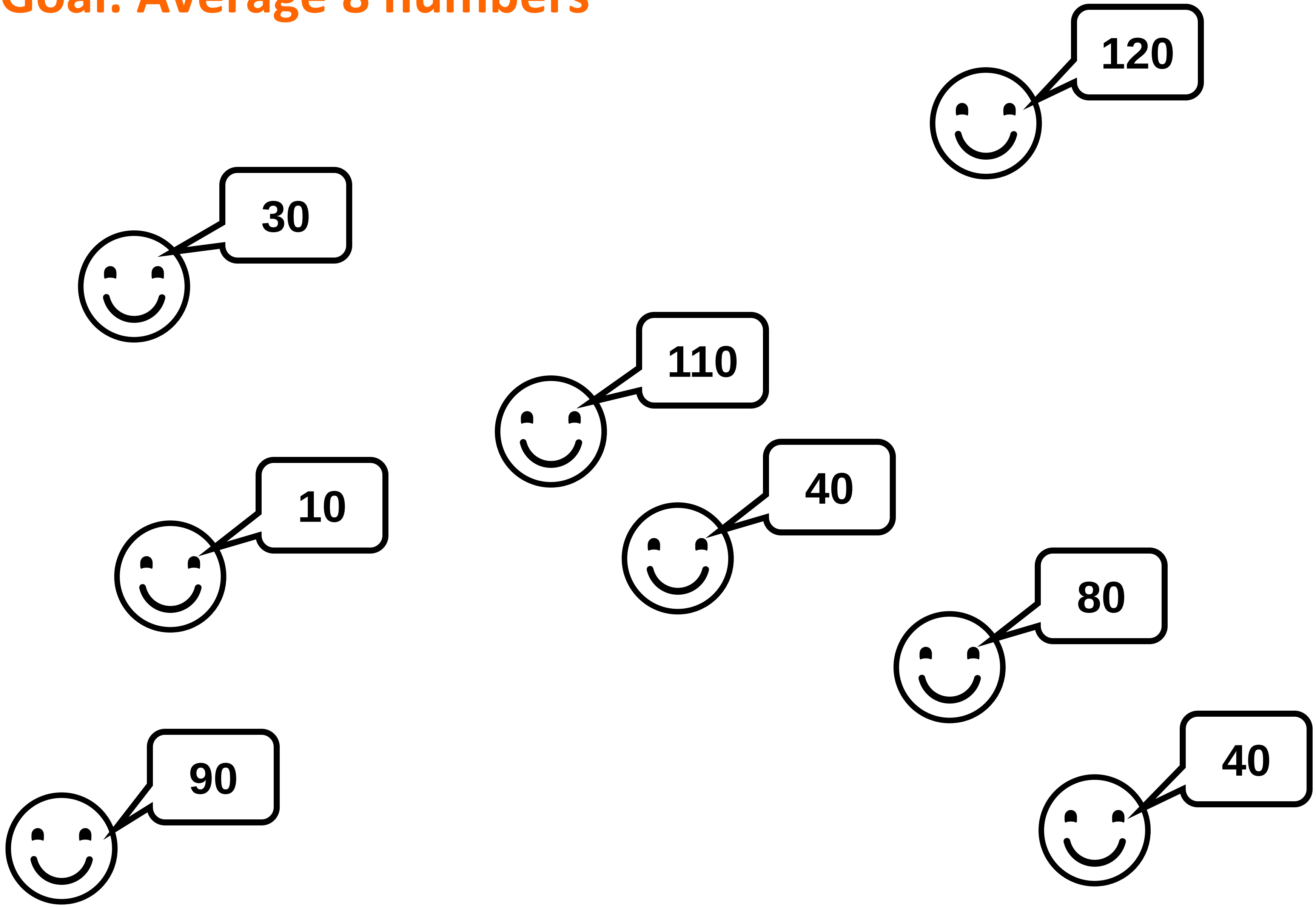
Agreement:

It’s like consensus for real-valued quantities

- Processors share real-valued quantities
- All processors converge to the same quantity.

Swarm School: Agreement - Distributed Averaging

Goal: Average 8 numbers



Goal: Average 8 numbers

Agreement: The Skit

Average Consensus

The simplest agreement algorithm

Start with *n* robots, whose state is stored in *n* variables:

$$x_1; x_2; \ldots ; x_n$$

Find a partner. Run the update rule:

$$x_1^0 = \frac{x_1 + x_2}{2}$$

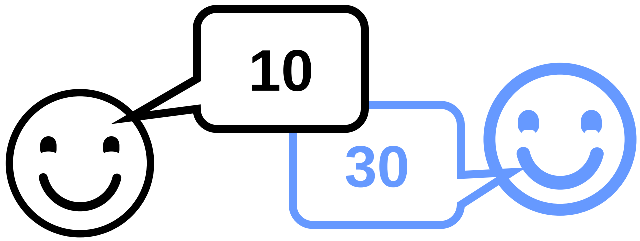
$$x_2^0 = \frac{x_2 + x_1}{2}$$

Then switch partners.

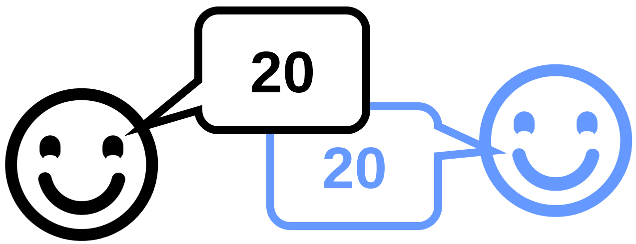
Repeat.

Instructions:

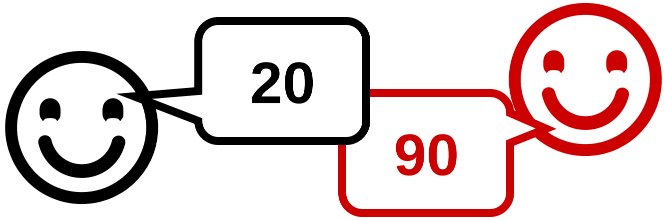
1. Enter your starting number into your calculator.



2. Pick another person and average your two numbers. (Add theirs to yours and divide by two) Don't round off, keep all the digits. Both people should end up with the same number.



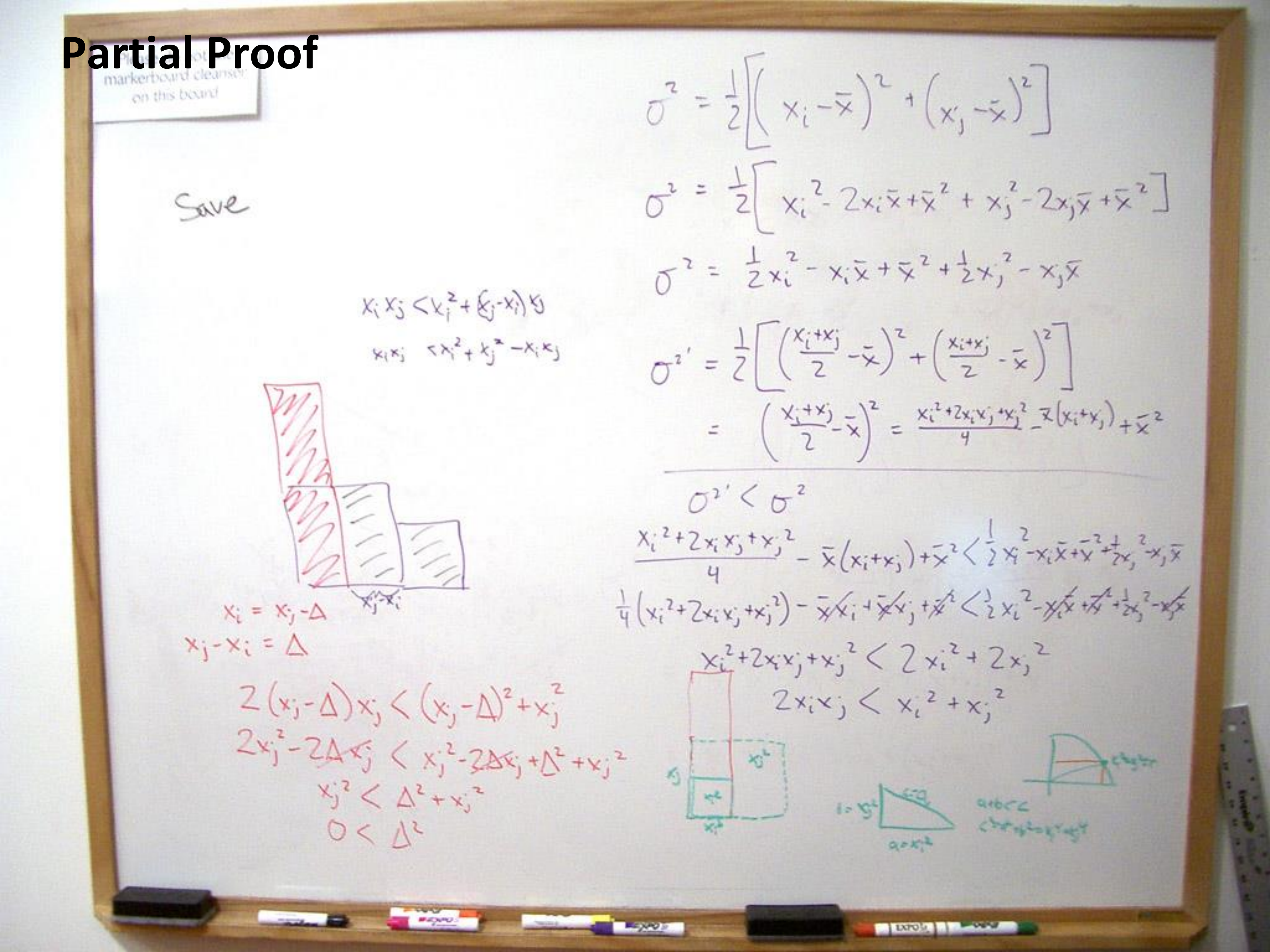
3. Repeat 12 times. Try to visit different people.



The answer is

65

Partial Proof



INVITED PAPER

Consensus and Cooperation in Networked Multi-Agent Systems

Algorithms that provide rapid agreement and teamwork between all participants allow effective task performance by self-organizing networked systems.

By Reza Olfati-Saber, *Member IEEE*, J. Alex Fax, and Richard M. Murray, *Fellow IEEE*

ABSTRACT | This paper provides a theoretical framework for analysis of consensus algorithms for multi-agent networked systems with an emphasis on the role of directed information flow, robustness to changes in network topology due to link/node failures, time-delays, and performance guarantees. An overview of basic concepts of information consensus in networks and methods of convergence and performance analysis for the algorithms are provided. Our analysis framework is based on tools from matrix theory, algebraic graph theory, and control theory. We discuss the connections between consensus problems in networked dynamic systems and diverse applications including synchronization of coupled oscillators, flocking, formation control, fast consensus in small-world networks, Markov processes and gossip-based algorithms, load balancing in networks, rendezvous in space, distributed sensor fusion in sensor networks, and belief propagation. We establish direct connections between spectral and structural properties of complex networks and the speed of information diffusion of consensus algorithms. A brief introduction is provided on networked systems with nonlocal information flow that are considerably faster than distributed systems with lattice-type nearest neighbor interactions. Simulation results are presented that demonstrate the role of small-world effects on the speed of consensus algorithms and cooperative control of multivehicle formations.

KEYWORDS | Consensus algorithms; cooperative control; flocking; graph Laplacians; information fusion; multi-agent systems; networked control systems; synchronization of coupled oscillators

Manuscript received August 8, 2005; revised September 7, 2006. This work was supported in part by the Army Research Office (ARO) under Grant W911NF-04-1-0316.
R. Olfati-Saber is with Dartmouth College, Thayer School of Engineering, Hanover, NH 03755 USA (e-mail: olfati@dartmouth.edu).
J. A. Fax is with Northrop Grumman Corp., Woodland Hills, CA 91367 USA (e-mail: alex.fax@ngc.com).
R. M. Murray is with the California Institute of Technology, Control and Dynamical Systems, Pasadena, CA 91125 USA (e-mail: murray@caltech.edu).
Digital Object Identifier: 10.1109/JPROC.2006.887293

I. INTRODUCTION

Consensus problems have a long history in computer science and form the foundation of the field of *distributed computing* [1]. Formal study of consensus problems in groups of experts originated in *management science and statistics* in 1960s (see DeGroot [2] and references therein). The ideas of *statistical consensus theory* by DeGroot reappeared two decades later in aggregation of information with uncertainty obtained from multiple sensors[1] [3] and medical experts [4].

Distributed computation over networks has a tradition in systems and control theory starting with the pioneering work of Borkar and Varaiya [5] and Tsitsiklis [6] and Tsitsiklis, Bertsekas, and Athans [7] on *asynchronous asymptotic agreement problem* for distributed decision-making systems and parallel computing [8].

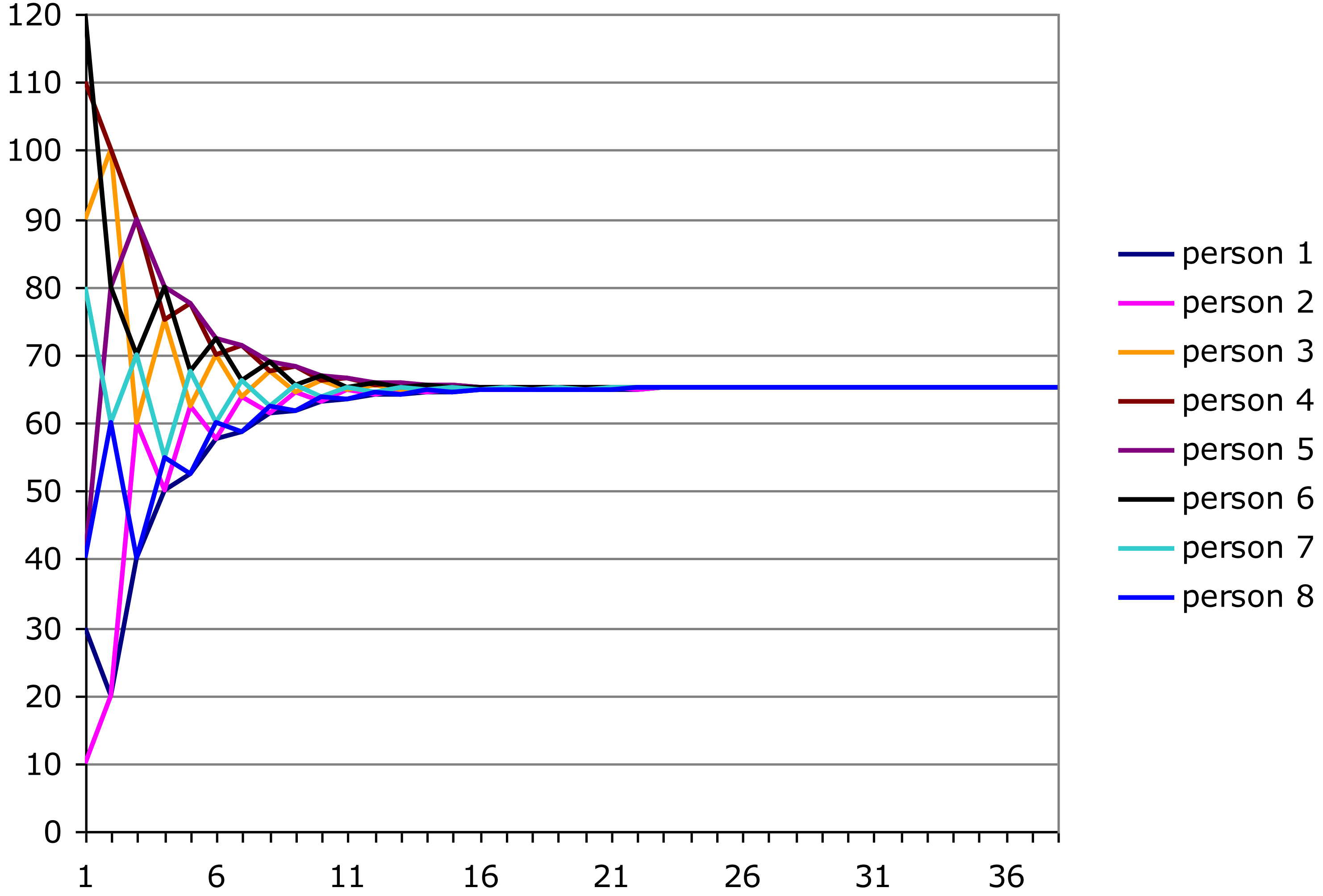
In networks of *agents* (or dynamic systems), "consensus" means to reach an agreement regarding a certain quantity of interest that depends on the state of all agents. A "consensus algorithm" (or protocol) is an interaction rule that specifies the information exchange between an agent and all of its neighbors on the network.[2]

The theoretical framework for posing and solving *consensus problems* for networked dynamic systems was introduced by Olfati-Saber and Murray in [9] and [10] building on the earlier work of Fax and Murray [11], [12]. The study of the *alignment problem* involving reaching an agreement—without computing any objective functions—appeared in the work of Jadbabaie *et al.* [13]. Further theoretical extensions of this work were presented in [14] and [15] with a look toward treatment of directed information flow in networks as shown in Fig. 1(a).

[1]This is known as *sensor fusion* and is an important application of modern consensus algorithms that will be discussed later.

[2]The term "nearest neighbors" is more commonly used in physics than "neighbors" when applied to particle/spin interactions over a lattice (e.g., Ising model).

Simulation



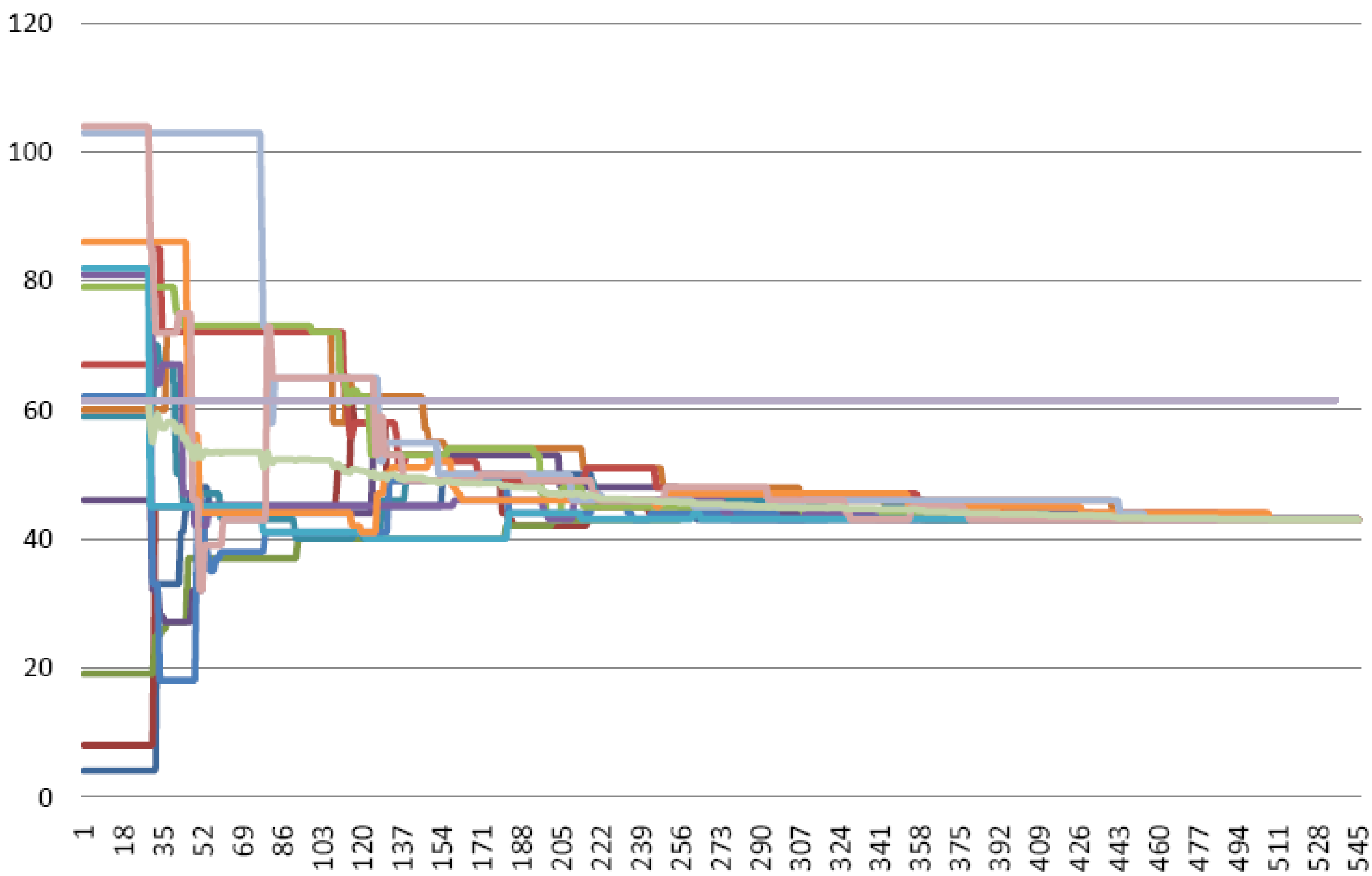
120
110
100
90
80
70
60
50
40
30
20
10
0
1
6
11
16
21
26
31
36
person 1
person 2
person 3
person 4
person 5
person 6
person 7
person 8

Who Would Compute an Average Using this Crazy Technique?

Honeybees! Workers share food all the time, computing a global average. This lets an individual worker know when the ***hive*** is hungry by measuring when ***she*** is hungry.

Agreement Experiment

Perfect agreement on systems that lose messages is impossible



Why Doesn't This Work on the Robots?

?

Agreement and consensus together

4evr…

Flocking and Consensus

Can we use consensus for more physical algorithms?

Like flocking

[white board]

Google “boids”…

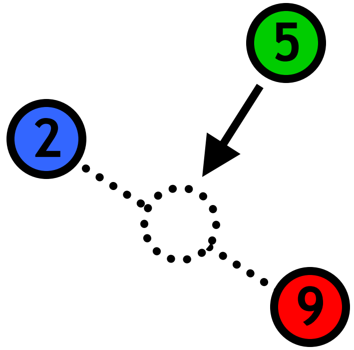
Flocking and Consensus

Can we use consensus for even more physical algorithms?

Like sorting?

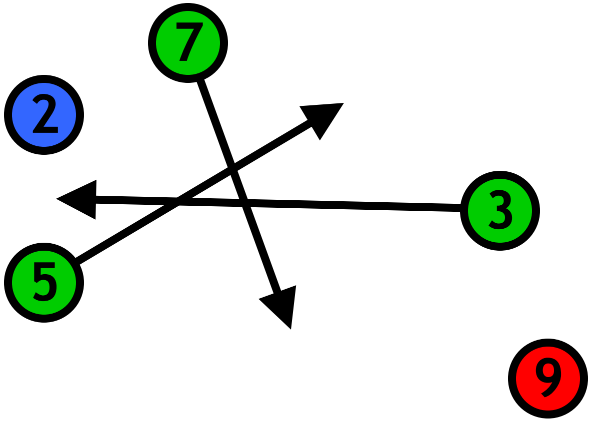
Physical Bubble Sort

Goal: Sort the robots by their robot ID



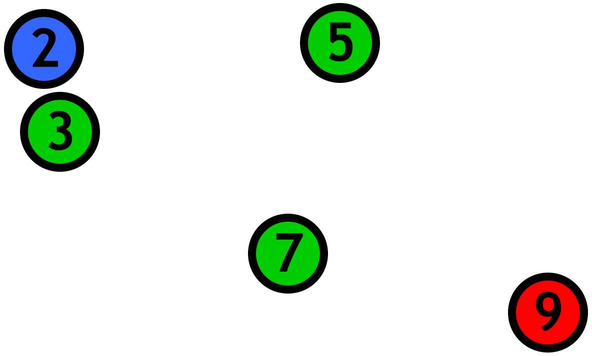
Physical Bubble Sort

Goal: Sort the robots by their robot ID



Physical Bubble Sort

Goal: Sort the robots by their robot ID



Physical Bubble Sort

Goal: Sort the robots by their robot ID