Massachusetts Institute of Technology
6.895: Advanced Distributed Algorithms
Professor Nancy Lynch

Lecture 26
December 10, 2002

Distributed coordination of mobile robots

Papers

- Xavier Defago, Akihiko Konagaya, Circle Formation for Oblivious Anonymous mobile Robots with No Common Sense of Orientation, POMC '02
- Flocchini, Prencipe, Hard Tasks for Weak Robots: The Role of Common Knowledge in Pattern Formation by Autonomous Mobile Robots.
- Flocchini, Grevasi, On the Intelligent Behavior of Stupid Robots
- Suzuki, Yamashita, Distributed Anonymous Mobile Robots: Formation of Geometric Patters.
- J. Walter, J. Welch, and N. Amato. Distributed reconfiguration of metamorphic robot chains. PODC 2000, Portland, Oregon.
- J. Walter, J. Welch, and N. Amato. Concurrent metamorphosis of hexagonal robot chains into simple connected configurations. To appear in IEEE Transactions on Robotics and Automation, 2002.

1 Overview

The problem being addressed is distributed coordination of a set of mobile robots to achieve some common *goal*. The *goal* in most cases is to achieve a given spatial distribution. What makes this interesting is that the robots are very *very* weak in a way. Therefore for achieving these goals, often it is required to solve more general problems like agreement on the coordinate systems, symmetry breaking, or electing a leader.

There are two classes of models that researchers have used to address the coordination problem. We discuss the results based on the point-robot models in detail. This model was first developed by Suzuki and Yamashita, there are other variants of this model which came later. The second class of models is called metamorphic robots. Here the robots have finite size and shape, and this is the basis of the work of Walter, Welch.

2 Point-Robot Models

In this model the robots are points, and roughly, they can move to any arbitrary point in the plane in a given time step. We wont consider failures or joins explicitly, however, since most of the algorithms are oblivious (no memory), failures are handled automatically.

2.1 Problems Addressed

Given a predicate π defining a pattern, for instance a circle, a point, or any arbitrary pattern:

Convergence problem is solved if the robots' distribution matches π irrespective of the number of robots and the timing with which they become active. The pattern formed by the robots could be a translated, rotated, scaled or flipped version of π.

Formation problem is solved if the convergence problem is solved in a finite number of steps. e.g. F-point is the problem of converging to a point.

Following, flocking, capture A leader acts independently of the others (controlled by human), followers are to follow the leader (following), in formation (flocking), in formation around the leader(capture).

2.2 General characterisitcs of Point-Robot model

1. Symmetric, identical, anonymous (no id)
2. Unique initial positions.
   Two or more can occupy the same location without colliding during the course of the algo.
3. No message passing. Only means for communication is the position sensor, knowledge of every other robot, possibly restricted by a certain visibility range.
4. In general there is no common coordinate system.
5. Algorithms operate in LOOK-THINK-LEAP steps.

   - An *oblivious* algorithm uses only the *current* positions of to compute the target.
   - A *non-oblivious* algorithm can use history of positions, i.e., it has unbounded memory.

   There are two variants of the point-robot model:

2.3 Strong model: SYm (Suzuki, Yamashita)

This is synchronous. The execution of the algorithms in each robot proceed in rounds. In a particular round a robot may or may not be *active*. If it is active then it performs the usual LOOK-THINK-LEAP sequence.

- Each round may span over any finite length of time.
- Robots activated infinitely often, such that at least one robot is active at every time instant.
- Instantaneous relocation to target. Distance covered is *less than* some $\epsilon_i > 0$.
  In other words robots cannot be seen while they are moving.
- The robots can detect multiplicity.

This is strong because the behavior of the peers can be predicted based on the current observations. To see this, suppose the time for completion of the rounds are $t_0, t_1, \ldots$, then if r_1 observes r_2 at time t_i in position p_2, then it can confidently assume that r_2 will not move before observing the positions at $t > t_i$.

2.4 Weak model: CORDA (Flocchini, Prencipe)

No rounds or common notion of time. Each robot does the LOOK-THINK-LEAP cycle in asynchrony.

- Each cycle finishes in arbitrary finite time. Which means that, within a cycle, LEAP may start at an arbitrary finite time after LOOK.

  Since there is no common notion of rounds a robot may be observed any number of times by the other robots within one of its own steps.
- Distance moved is finite and *greater than* some constant $\delta_i > 0$.
- Oblivious
- Can *not* detect multiplicity.

3 Results

We will discuss the point formation or gather problem first, then the circle formation problem and finally the problem for attaining any arbitrary pattern. Since SYm is a stronger model than CORDA, any impossibility result carries over from SYm to CORDA and any algorithm should carry over from CORDA to SYm.

3.1 F-Point / Gather

SYm model.

Theorem 3.1 There is no oblivious algorithm for solving F-Point for the case of n=2.

Proof sketch: Suppose ψ is an oblivious algorithm for F-point(2).

First, observe the following:

In order for ψ to finish, there must be distinct positions p and q for r_i and r_j such that either (1) ψ move r_i to q while r_j remains stationary, or (2) ψ moves r_j to p while r_i remains stationary, when both get activated simultaneously.

Suppose this was not the case, then for ψ to finish, at time t, r_i and r_j are in distinct positions p and q at time $t-1$, and they are in the same position r at time t. Then:

> If both r_i and r_j were active at $t-1$ then by assumption $r \neq p$ and $r \neq q$. Now if the adversary activates only one robot instead of both then they wont meet at t.
>
> If only one of them were active at $t-1$, say r_i then $r = q$. And then the adversary can make both of them active at $t-1$ and by assumption we have at time t r_j moving to $s \neq q$.

Like this we can construct an infinite execution, in which each robot is activated infinitely often but they will never meet.

From the above fact, ψ finishes by moving one of r_i and r_j which are in distinct positions p and q. But (oblivious) as all moves of the robots depend on the local coordinate system and the position of the other robot, we can modify $Z-i$ and Z_j such that each robots observes the same world.

Algorithm Non-oblivious algorithm for F-point(2).
First translate and rotate the coordinate system such that r_i is at the origin of Z_i and r_j is along the positive y-axis of Z_i. Then r_i moves in the positive x direction of Z_i every time it is active, until it observes that r_j has moved twice. Since r_j is moving along the positive x-axis of Z_j, r_i can now determine the original location of r_j. r_i moves to the mid point of the original location of r_i and r_j.

Algorithm *Oblivious algorithm for F-point(n)* $n \geq 3$.
Starting form distinct locations, the robots are moved so that, eventually there will be exactly one location, say p, which is occupied by more than two robots. Then, all the robots which are not located at p can gather at p (assuming that the robots can detect multiplicity).

- For n = 3. If a,b,c are collinear with b in the middle then a and c move towards b.
- If a,b,c are isosceles, with b at the apex of the equal sides, then b moves towards the foot to of the perpendicular dropped from its current position to $\overline{ac}$, avoiding becoming equilateral. This reduces to previous case.
- If all sides are unequal, then the point opposite to the longest side moves towards the foot of the perpendicular while the other two remain stationary.
- If a,b,c are equilateral, then move toward the center.
- For $n \geq 4$, let the smallest enclosing circle at time t be C_t. If there is only one robot inside C_t then it moves towards any robot on C_t. All others remain stationary.
- If there are two or more robots inside C_t then they all move towards the center of C_t. All others remain stationary.
- If there are no robots in the interior then they all move towards the center of C_t.

Observations: Use multiplicity detection. Use the agreement on coordinate independent geometric point (also used in circle formation). *Note* Obvious choice might be to converge at the Weber point, i.e., the point in a plane which minimizes the sum of distances to all robot positions, but unfortunately Weber point is not computable.

Summary: F-Point

1. **[SYm]** With unlimited visibility, F-point is unsolvable for $n = 2$.
2. **[SYm]** F-point solvable for any $n \geq 3$.
   [CORDA] F-point solvable for $n = 3, 4$ if the robots can detect multiplicity. For $n > 4$ solvable if not in biangular configuration initially.

3.2 Circle Formation

We look at another formation problem in the SYm model. Once again we assume that there is no common knowledge about the direction or orientation of the axes.

Theorem 3.2 There exists oblivious algorithms for circle formation, and convergence to a circle with uniform distribution.

Algorithm *Oblivious algorithm for circle formation*

Circle Formation: Compute the smallest enclosing circle C_t (this is coordinate independent). We have to ensure that, two robots do not occupy the same location

- If on the boundary of C_t then stay still
- Else move towards the point in the voronoi cell which is closest to C_t. If there are more than one such points then break ties using the x,y coordinates in lexical order.

Proof sketch: C_t is an invariant.
Safety: No robot moves further away from C_t than its current location.
Liveliness: There exists at least one robot in the interior of C_t which can make a non null move towards C.

To see the last fact, suppose there exists a configuration in which none of the robots inside C can make any progress. Let r be the robot in P_{in} which is furthest from the center. Since r is in the convex hull of P_{in}, it must be blocked by some robot in $P \setminus P_{in}$. It must be the case that there are two blocking robots r_a and r_b, otherwise r would have simply moved around the blocking robot.

It can be shown that $\forall D_r(t), x_r(t) \geq \frac{\Delta_{ab}}{2} > 0$. Where x_r is the maximum distance r can move towards the mid-point of r_a and r_b, D_r is the distance between r and $\overline{ab}$, Δ_{ab} is the distance between r_a and r_b. Therefore r can always make progress. All robots in P_{in} reach the boundary of C in a finite number of steps.

Uniform Transformation
Find the mid-point p between left and right neighbors. Move half-way between current location and p. This converges towards a configuration in which all the robots are arranged at regular intervals at the boundary of the circle.

Proof is not in the paper, show that the uniform distribution is stable, i.e., the robots are stationary when they reach this configuration. Also show that the max of the angle between any two direct neighbors is monotonically decreasing, while the min is monotonically increasing.

3.3 Formation of Arbitrary Patterns

We shall now switch to the CORDA model to discuss the ability of the robots to form any arbitrary pattern given by a set of points $P = \{p_1, \ldots, p_n\}$ with different levels of common knowledge about the coordinate system. Similar problems are addressed in the SY paper, the CORDA results seem to convey a rather complete story.

3.3.1 Direction and Orientation of both XY axes are known

Goal pattern P described as a set of points $p_1, \ldots, p_n$, in lexicographic order. Let $\alpha = \angle p_1, p_2$. Since directions and orientations of x and y axes are common knowledge. Therefore each robot can order the position of all the others in lexicographic order $a_1, \ldots, a_n$. Let $\beta = \angle a_1, a_2$.

Theorem 3.3 Any arbitrary pattern can be formed.

Algorithm

a_2 remains stationary.
a_1 moves to make $\beta = \alpha$ while remaining lexicographically first.
All others do nothing until $\alpha = \beta$, once this happens, $\overline{a_1 a_2}$ can be used as a common unit of distance.
Calculate final positions $F = f_1, \ldots, f_n$ by scaling P with above unit.
If already in F, then do nothing, else move to closest *free* final position. If more than one robot is closest to a particular f_i then the one with lower a_j is chosen to occupy f_i.

3.3.2 Direction and Orientation of Y axis is known and n is odd

This implies that everyone knows that ... knows the direction of X axis but not its orientation.

Theorem 3.4 For odd n, any arbitrary pattern can be formed.

Algorithm
Since the direction of X is known, can find the median pattern point, let this be p_m. We can also find the median robot r_m, which divides the entire set of robots into 2 subsets of equal size $(n-1)/2$; One to the left of K and below r_m and the other to the right of K and above r_m. We need to find unique outer-most robot position w.r.t r_m, to define an orientation for X and also a scale.

The outermost robot from p_m is found as follows:

Let K = vertical line through r_m. Consider the furthest robots from K, in lexicographic order of Y. There can be at most two such robots. If there is one then everyone decides on positive X direction, outer robot, and outer line.
Else, r_m moves to its *right* by ϵ.
Outer robots move away from K by ϵ.
Unit = distance between p_m and outermost pattern point.

3.3.3 Direction of Y axis is known and n is odd

Theorem 3.5 For odd n, any arbitrary pattern can be formed.

The agreement on the orientation of X axis can be achieved without using the orientation of Y axis by a slight modification of the previous case.

Find the line K through median robot (does not require orientation of X or Y).
Find the outermost robots w.r.t K. Since we do not know the orientation of Y, there can be at most 4 such robots.
We can break the symmetry, as in the previous case, by making the median robot move ϵ to its right. This will leave at most two of the outermost robots to the left. And we have agreement on the orientation of X axis.
Similarly, the median robot could move ϵ up or down to reach an agreement on the orientation of the axis.

3.3.4 Direction and Orientation of Y axis is known and n is even

Theorem 3.6 Arbitrary pattern formation in impossible when n is even.

Theorem 3.7 Leader election in impossible when n is even.

Proof: Let $\mathcal{A}$ be a leader election algorithm. Assume the following initial distribution: for every r there is a corresponding r', such that the directions of X axis for r and r' are opposite and their view of the world is identical. So $\mathcal{A}$ would cause r and r' to move in the same way. By making all the robots move at the same time, the symmetry is preserved. Thus the robots cannot deterministically agree on a single robot.

Theorem 3.8 Any asymmetric pattern cannot be formed when n is even.

Proof: We show by contradiction that this would enable the robots to elect a leader. Consider the final asymmetric formation F. Since the robots agree on the direction and orientation of the Y axis they can do the following to elect a leader as follows:

> The robots must agree on the outermost lines through F, let S_o and S_o' be the sets of robots on the two extreme outermost lines. If $|S_o| = |S_o'|$ and the Y-coordinate of every robot in S_o equals the Y coordinate of some robot in S_o' then select the next two outermost lines S_i and S_i'. Since F is asymmetric this must stop somewhere before hitting the central axis of F.
> If $|S_i| \neq |S_i'|$, select the robot with the largest Y coordinate as the leader from the larger of the two sets.
> If $|S_i| = |S_i'|$, select the robot with largest Y coordinate amongst those which are not matched.

Theorem 3.9 A symmetric pattern in which all the axes of symmetry pass through a vertex, cannot be formed when n is even.

Proof: Again we show that a leader can be elected. If F is not symmetric w.r.t. any axis Y' parallel to Y, then we can use the same technique as in the previous proof. If F is symmetric w.r.t. some Y' parallel to Y then Y' must pass through at least one robot. Since all robot positions are unique, therefore we can select the vertex on Y' with largest Y coordinate as the leader.

Finally,

Theorem 3.10 The only patterns which can be formed with even number of robots are symmetric ones with at least one axis of symmetry not passing through any vertex.

Definition 3.11 Agreement configuration: all robots agree on position of vertical axis K in the plane. K has no robots on it, and splits the plane into two sides with n/2 robots on each side.

Algorithm

1. Choose the smallest axis of symmetry (every robot can select the same axis), rotate the input pattern to align this axis along the common Y axis.
2. The outermost topmost robots reach a same height, until which all other robots do not move. This is possible in a finite number of moves because, there is common notion of height, and the higher robot can be forced to move towards the lower.
3. At this point all the robot can agree on the median axis K.
4. The highest robot on K moves towards the side with fewer robots.
5. Once there are no more robots on K, then robots from the more populated side moves to the other side of K. This finishes in a finite number of steps when there are equal number of robots on both sides of K.
6. Calculate final positions.

Summary: Arbitrary patterns

1. [**CORDA**] With a common coordinate system the robots can form any arbitrary pattern.
2. [**CORDA**] With agreement on only one axis direction and orientation, the formation problem is unsolvable when n is even and it is solvable when n is odd.
3. [**CORDA**] With agreement on only one axis direction and orientation, an even number of robots can form only symmetric patterns that have at least one axis of symmetry not passing through any vertex of the pattern.
4. [**CORDA**] With no agreement at all the robots cannot form any arbitrary pattern.(what can they form, it is known that they can form a circle).

3.4 Open Problems

F-point
Boundaries of solvability oblivious, n = 2, ...what else ? Hybrid model ?

Circle formation
Uniform transformation in finite time — paper conjectures that such an algorithm does not exist in the SYm model. Are there any non-oblivious algorithms ?

Circle formation in CORDA: weaker model, no synchrony, and maybe limited visibility.

General pattern formation
No agreement on direction or orientation of any axis — what is the class of patterns which could be formed ? Circles .. and ?

New problems
Spreading and exploration for constructing a map.

4 Metamorphic robots

Here the robots are finite sized hexagons and their movements are much more constrained. They can only move to a neighboring cell under certain conditions. This is a completely synchronous model, there is a common coordinate system, but once again there is no communication.

4.1 Problem

Given a initial configuration of the robots I and a final, determine a sequence of robot motions required to reach a goal configuration G.

4.2 Model

1. Identical hexagons. Finite sized and occupy distinct *cells* in space,
2. No communication between modules.
   - knowledge of current location, that is, the coordinates of the current cell. There is a common coordinate system.
   - knowledge of the occupancy of the neighboring cells.
3. Movement is in lock step rounds.
4. Only one robot tries to move to a particular cell in each round.
5. M can move to C iff
   - C is currently empty.
   - M has a neighbor S (called substrate) that does not move in the current round and S is also adjacent to C.
   - The neighboring cell to M on the other side of C is also empty.
6. Collisions. May or may not be *movable* in a given configuration.
7. Robots achieve locomotion by moving over a substrate composed of other robots.
8. Hexagonal coordinate system. Lattice distances.

4.3 Algorithm for Collinear Case

G and I are collinear with only one overlap between them. Simple algorithm; classify the modules according to *contacts*: `Trapped`, `Free`, or `Other`. Iniitally each module decides its direction CW or CCW using the parity of its distance from the overlap module. In each round *contacts* is updated automatically. This is used to compute the *isFree* function. If a module is free then it moves according to its predetermined direction, if it is not already in one of the cells in G.

Invariants

WOLG assume I and G run fro north to south, number the cells from 1 through $2n-1$. And the module originally in cell i is numbered module i. For each i, $1 \leq i \leq n$, and each round $S \geq 1$, at the end of round S,

- If $S < 2i - 1$, then module i is in cell i.
  (module i moves for the first time in round $2i - 1$).
- If $S = 2i - 1 + j$, $0 \leq j \leq n - 1$,
  - If i is odd, then module i is SE of cell $i + j$,
  - If i is even, then module i is SW of cell $i + j$,
- If $S \geq 2i - 1 + n$, then module i is in cell $i + n$.
  (after n moves from the beginning of movement i reaches its target cell).

Proof is by induction and case analysis, not particularly enlightening.

4.4 Algorithm for non-collinear chain

The consider the 5 possible case when I is oriented NS and G is SW-NE. The definition of `Trapped`, `Free`, and `Other`, changes but the algorithm is the same as case 0.

4.5 Lower bounds

Bound on the number of moves and the total time elapsed for reconfiguration. Lower bounds are functions of D_0, the initial lattice distance sum. Given I and G, for a particular deterministic algorithm A, $S_A = \Sigma_i LD(I(i), G(i))$. $D_0 = min_A S_A$.

Theorem 4.1 For case 0 and 1, $D_0 \geq n^2 - n$.

Proof: Cells are numbered 1 to n in I, and n to 2n-1 in G. Let v_i be the lattice distance between module i's initial position,that is, cell i, and its final position, cell g_i. For both case 0 and 1, $v_i = g_i - i$ (thats how lattice distance is defined!). Also g_1 to g_n is a permutation of $n \ldots 2n-1$. So, $D_0 \geq \Sigma_i v_i = \Sigma_i (g_i - i) = n^2 - n$.

Theorem 4.2 For case 2, $D_0 \geq (n^2 -)/2$, for cases 3 to 5, $D_0 = \Omega(n^2)$.

4.5.1 Module movements

Theorem 4.3 (Chirikjian, Pamecha) Any algorithm for reconfiguration must cause at least D_0 module movements.

The above lower bounds on D_0 give us the lower bounds on module movements.
The bound for case 0 (collinear) is tight for goal stopping algorithms. This is so because the n^{th} module does not move, and every other module has to take an extra step to move around it. Therefore adding $n-1$ to D_0 gives the tight lower bound $n^1 - 1$, which matches with the algorithm given earlier. The bound for case 0 is not tight for non-goal stopping algorithms, there exists a non-goal stopping algorithm which takes roughly $n^2 - n/2$ steps.

4.5.2 Number of rounds

Theorem 4.4 Any algorithm for case 0 must take $\frac{D_0}{\lfloor n/2 \rfloor}$ rounds.

In each round the lattice distance to final position of any module can decrease by at most one.

Claim 4.5 For case 0, maximum number of modules that can reduce their lattice distance to their final position in one round is $\lfloor \frac{n}{2} \rfloor$.

Proof: At most $3n/4$ modules can move in a round (why ?).
However for case 0, at most 2 out of 4 modules can reduce their distance to final position, they claim that this is shown by an exhaustive listing of all possible moves of three modules over one substrate.

Plugging in D_0, the lower bound on the number of rounds $\frac{D_0}{n/2}$ for case 0, and $\frac{D_0}{3n/4}$ for case 1 through 5.

4.6 More general configurations

WLOG, assume I to be N-S and in the leftmost column of the plane.
Assume G and I intersect only at the southernmost cell of I.
Define *admissibility* of goal configuration G with a centralized algorithm.
Give distributed algorithms for achieving G from I.

4.6.1 Admissibility

A *substrate path* is a contiguous sequence of cells, if the first cell (from N-S) is in I, and all subsequent cells are in G and the last cell is in the easternmost column of G. A *south(north) segment* is a contiguous subsequent in which every cell is in the south (north) of the previous.

Define a admissible path whose surface would allow movement of the modules without collision or deadlock, that is there are no pockets and corners in the surface where the modules might get stuck. A contiguous subsequence p is an *admissible path* if

- p is a east-monotone, each cell is adjacent to the previous cell in p, but not to the west.
- every north segment of p ending with c_i, the next 3 consecutive SE cells from c_i are not goal cells and the next 4 cells to the north of c_i do not form any south segments.
- every south segment of p ending with c_i, the next 3 consecutive NE cells from c_i are not goal cells,and the next 4 cells to the north of c_i do not form any north segments.

G is an admissible goal configuration if there is an admissible substrate path in it.

4.6.2 Algorithm

Check if there is an orientation of G in which each column G_i is contiguous.

- Label the cells of columns of G_i as $G_{i,1}, G_{i,2}, \ldots$ from N-S. (Cell $G_{11,1}$ is in I).
- Construct a graph H with each cell of G_i as a node. Extra node $G_{0,1}$