Bio-inspired Multiagent Systems

Radhika Nagpal
Computer Science, Harvard University
Americas School on Agents and Multiagent Systems, 2006

Outline
• Amorphous Computing
  – pattern formation in silico
• Collective Construction by Robot Swarms
  – shape and pattern in robotics

Many Other Areas:
  Swarm intelligence, Immune-inspired Systems, Evolutionary approaches, etc

Amorphous Computing
• How do we obtain a robust behavior from the cooperation of vast numbers of unreliable parts?
• How do we engineer pre-specified global behavior from local interactions?

An Amorphous Computer
A Simple Pattern: Polka Dots

A Simple Pattern

Gradient

Polka Dot Program
Polka Dot Program

Trigger → secrete I (and turn blue)
Receive I > threshold

How much more complex are these?

Suppose we want to create an arbitrary pattern?

An Inverter Program
A Set of Construction Rules

Initial Conditions: “Determinants”

Implementing the Rules

Individual Element

(define (axiom2-rule il i2 g1 g2 gend)
  (if il (create-gradient g1))
  (if i2 (begin (wait-for-gradient g1) (create-gradient g2)))
  (if il (begin (wait-for-gradient g1) (wait local-delay) (create-gradient gend))))

(wait-for-gradients gend)

(if (<= (abs (- g1 g2)) threshold) #t #f)
An Inverter Program

(define v1 (crease-l2l e23 e41))
(define v2 (crease-l2l v1 e23))
(define v3 (crease-l2l v1 e41))
(define IN (create-region e41 v3))
(define MID (create-region v1 (or v2 v3)))
(define OUT (create-region e23 v1))

; Similarly create horizontal regions...

; Lay down Material (differentiates)
(within-region IN (color h1 "poly"))
(within-region MID (color h2 h3 "poly"))
(within-region OUT (color h1 "poly"))
(within-region CNTR (color v3 "poly"))
(within-region UP (color v1 "n-diff"))
(within-region DOWN (color v1 "p-diff"))

(define contacts (intersect v1 (or e12 h1 e34))
(color contacts "contacts")

Pattern-Formation on an Amorphous Computer

A "Generative" Global Program

(define v1 (crease-l2l e23 e41))
(define v2 (crease-l2l v1 e23))
(define v3 (crease-l2l v1 e41))
(define IN (create-region e41 v3))
(define MID (create-region v1 (or v2 v3)))
(define OUT (create-region e23 v1))

; Similarly create horizontal regions...

; Lay down Material (differentiates)
(within-region IN (color h1 "poly"))
(within-region MID (color h2 h3 "poly"))
(within-region OUT (color h1 "poly"))
(within-region CNTR (color v3 "poly"))
(within-region UP (color v1 "n-diff"))
(within-region DOWN (color v1 "p-diff"))
(define contacts (intersect v1 (or e12 h1 e34))
(color contacts "contacts")

Inverter Chain Pattern

:: Segment into 8 COMPARTMENTS
:: Execute inverter pattern program within a region

(within-region r1 (create-inverter left-border1 right-border1))
Many Shapes and Patterns

Local State: Boolean per distinct point, line, region
Morphogen Gradients: Many, but short-lived
Could use as few as 6

Scale-Independence

Related Structures

Program the Cell, to "grow" the appropriate shape

(Cell can grow, not grow, or die)
Constructing a 2D shape

Two rules:
1. Grow a circle of a given radius
2. Elect reference points

Shape = Set of rules (proportional to the number of circles)

Self-Assembly using directed growth

Attila Kondacs, Catherine Chung

Self-Repair and Regeneration

Regenerating structures
- Absence of neighbor causes circle to recreate its neighbor, which in turn recreates its neighbor - thus regenerating the broken structure

Self-repairing patterns
- If a line is broken, one part dies off and the other regrows

Languages for Pattern Formation

Languages for "constructing" pattern

Behavior of an individual element

COMPILER
Lessons From Amorphous Computing

• Programmable Morphology
  – Global-to-local compilation
  – Small set of primitives
  – Scale-independence and regeneration

• How do these apply to computer systems?

Engineering Shape and Pattern

Self-Reconfiguration
In Modular Robots

Formation Control
for Robot Swarms

Collective Construction
by robot swarms

Collective Construction
by Robot Swarms

Joint work with Justin Werfel,
Daniela Rus (MIT), Yaneer Bar-Yam (NECSI)
Properties of a Swarm Approach

Advantages
- Exploits Parallelism
- Robust
  - Varying numbers of agents, agent loss or addition, asynchronous
- Simple Agent Capabilities

Disadvantages
- Hard to create local rules for a given goal

Our Goal

- Collective construction
  - Assemble a *user-specified* shape
  - Robust (similar to swarms)
  - Simple robot capabilities
- Does augmenting the environment help?
  - What if blocks can store state? or even communicate? (e.g. RFIDs or motes)

Abstract World

- Robots
- Marker for structure start
- Caches of blocks

Constraints on Block Placement

- Assembled shape
Agent Algorithm Strategy

• Given a Shape Map
  – Localize relative to the structure
  – Avoid bad intermediate states

Correctness: Multiple robots, no deadlocks

1: Inert Blocks

Robot Program:
- Find Landmark
- Keep track of position by counting blocks
- Using avoid-gaps rule and shape map, find an attachment site

Avoiding Gaps Rule

• Constraint
  – Avoid placing two blocks separated in the same row

• Avoid Gaps Rule
  • Right corner => ok
  • Left corner, followed by End-of-row => ok
  (correct for multiple robots)

2. Writeable Blocks

Robot Program:
- Read ANY block to localize
- Use avoid-gaps rule to prevent bad intermediate states
3: Communicating Blocks

“Block” Program:
- Tell robots the location
- Tell robots if it is possible to attach a block to that face (better ways to avoid gaps)

Comparison
- Common Properties
  - Robots act independently, but coordinate through interaction with the structure.
- Performance
- Robustness
- Cost

Implementation
- Robot Capabilities: Robust perimeter following, recognize block edges and corners, follow beacon, manipulate blocks
- Block Capabilities: Self-aligning
Conclusion

- We can address the two questions posed by Amorphous Computing.

- Still Many Questions:
  - Are some shapes harder than others? How do we think about functional shapes?
  - What about other types of global goals?
  - Can we know what is the “best” (or “simplest”) possible algorithm given an agent’s capabilities?

Acknowledgements

Amorphous Computing Group, MIT CSAIL
Gerry Sussman, Hal Abelson, Tom Knight, Daniel Coore, Ron Weiss, Jake Beal, Attila Kondacs, Catherine Chang, Lauren Clement

Reconfigurable Robots/Robot Swarms
Kasper Stoy (Univ of S. Denmark), Winston and Jimming Cheng (Harvard)

Collective Construction
Justin Werfel (MIT/Harvard), Matt Valente (MIT), Crystal Schuil (MIT), Daniela Rus (MIT), Yaneer Bar-Yam (NECSI)