Indirect vs. Direct Control

Ron Goldman
Sun Labs

Traditional software control structures

- Conditionals
- Loops
- Goto
- Subroutines (including RPC)

All designed for sequential execution on a single computer

New structures

- Threads
- Aspects
- Patterns
- Client-server
- Multi-agents
- SOA

Attempts to deal with multiple threads of control across many machines

Direct control

- All imperative commands
- Do (exactly) this
- Aimed at directly achieving a global goal
- What you do is what you get

Direct control is great when it can be used

- Understandable model

How scalable is it?

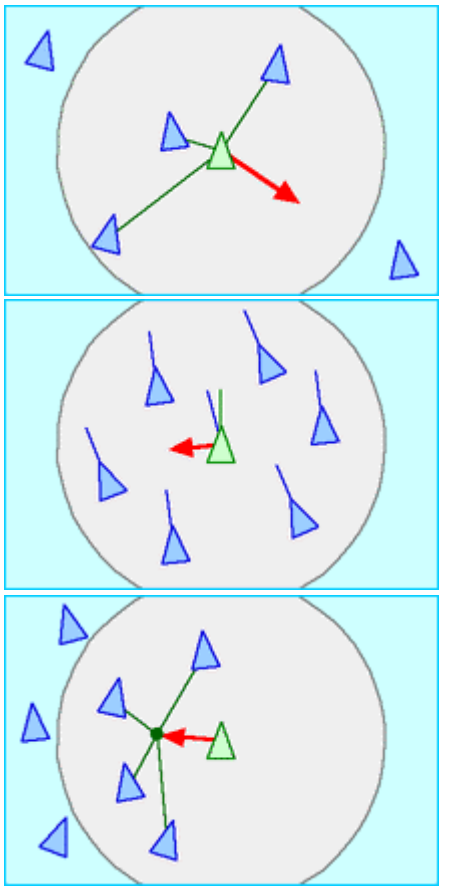
- Problem: compute velocities for birds in a flock

- Centralized control
  - Bandwidth needed to communicate global state
  - Computation needed to solve $O(N^2)$

Boids

Craig Reynolds 1986 – 4 rules

1. Separation: steer to avoid crowding local flockmates
2. Alignment: steer towards the average heading/speed of local flockmates
3. Cohesion: steer to move toward the average position of local flockmates
4. Avoidance: steer to avoid any physical obstacles



Indirect control

- Still all imperative commands
- Do something local
- Aimed at indirectly achieving a global goal
- What you get is somehow the result of what you do

Indirect control is harder than direct control

- Model more complicated
- Global effect of local actions harder to understand

Emergence

- Local control => indirect global results
- Local rules may be hard to determine & hard to tune
- Emergence is not mystical, but reflects the difficulty of seeing how low-level behavior achieves higher-level goals

Asymmetry of computation

- Predicting the properties of water from those of hydrogen and oxygen is hard
- Determining that the properties of water are consistent with the known properties of hydrogen and oxygen is much easier

Factoring a large number is orders of magnitude harder than verifying product of factors

- Why public key encryption works

Need new tools

- Traditional software development tools are not all that great
  - IDE's
  - Debuggers
  - Profilers
- Programming emergent systems requires new tools
  - New language constructs
  - Simulators

Examples from computing

- Ethernet unfairness
  - Capture effect seen when hardware got fast enough to fully exploit the timing allowed by the specification
- Internet congestion
  - October 1986 a series of "congestion collapses" due to TCP parameters
  - BGP (Border Gateway Protocol) message storms

Applying swarm intelligence

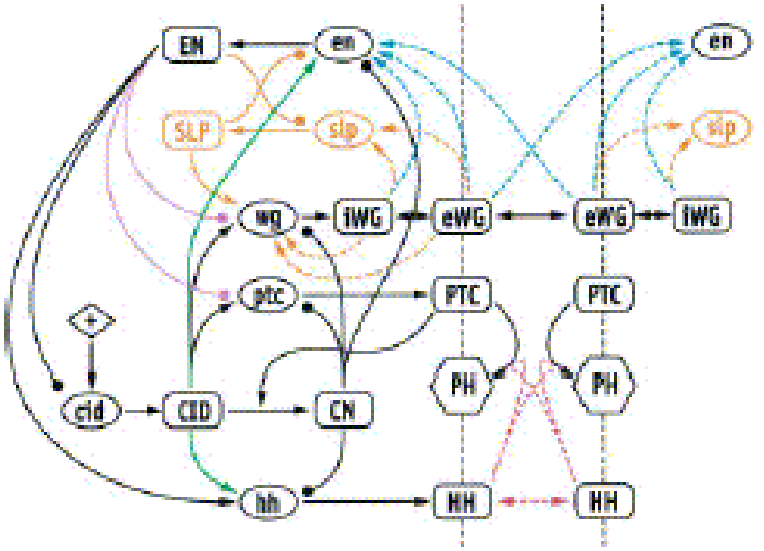

Truck painting at GM

- Based on how honeybees shift work roles
- Changing paint color is time consuming & costly
- Auction system: booths bid based on current color & queue
- Saved over $1 million/year just in paint
- Handled breakdowns and surges in orders flexibly and robustly

Problems accepting emergent solutions

- Next system for truck painting at GM went back to a conventional solution
- Some people need proof of correctness
- Not comfortable with gap between local rules and global outcome
- Worried might have unexpected behavior
  - Of course same concerns apply to conventional code
- How to test?

Structural robustness



The segment polarity network is a robust developmental module

- George von Dassow et.al. 2000

- Change in network (adding two more reaction pathways) changed system from having no solutions to having many
- “The most striking systems-level property we report is the robustness to parameter variation”
- “The simplest model that works at all emerged complete with unexpected robustness to variation in parameters and initial conditions.”

Robust, yet fragile

Highly Optimized Tolerance (HOT)
- Carlson & Doyle 1999

- Robust to common disturbances the system is designed to handle, yet fragile to rare events and flaws in the design.
- Shielding certain functions of a system may require additional control loops; this shielding leads to higher complexity and new potential sources of fragility
  - e.g. cutting a wire in a computer or 777 airplane

Software development

- Initially entire software system was developed by one group
- With software services many groups are now involved
- Future software development will be even more decentralized and independent
- Open source => mob software