Computational Evolution & Digital Organisms

A look at a subset of Artificial Life

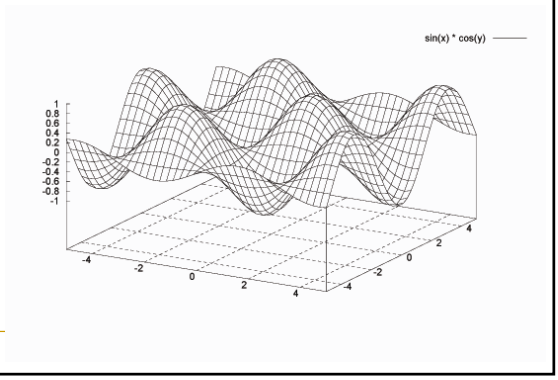
By Daniel Weise

Computational Evolution

- Attempts to elucidate principles of evolution
  - Builds models of self-replicating organisms
    - Computational cost limits physical fidelity of the model.
    - Digital or chemical models
  - Mutation creates variation in populations
  - Reproduction can be sexual or asexual
  - Ability to (out) reproduce its genome is the usual fitness measure
    - For some research, other fitness measures are used.

Not to be Confused With Evolutionary Computing

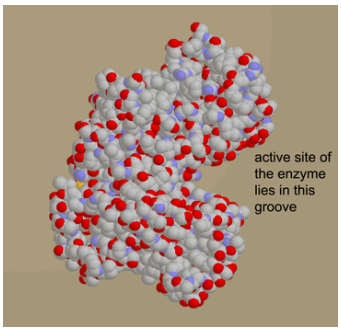
- A Search Technique inspired by biology
  - Points in search space represented as “genomes”
  - Crossover produces new points in search space
  - Mutation ensures variety
    - Ensures more of search space is sampled
  - Fitness function determines which subset of population become progenitors
  - Larger populations increase coverage of space.
  - Search usually walks through “invalid” points



Overview of Talk

- Motivation: The complexity of cellular life
- Tierra and the evolution of digital organisms
- Avida and other Tierra inspired work
- Lessons/Future Research

Hexokinase



Complexity of Regulatory Mechanisms

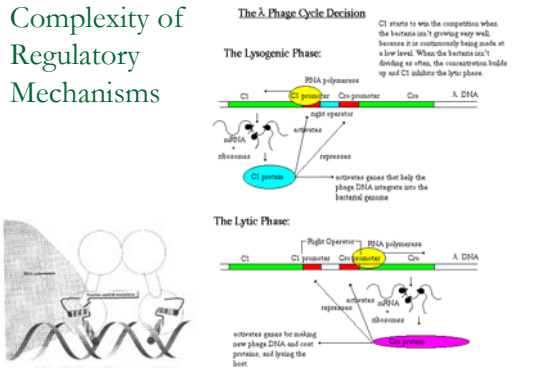


Image from http://web.mit.edu/esgbio/www/pge/pgeother.html

Nature made this from

- Molecules with differential binding affinities for DNA.
- Overlapping control regions.
- Positive and negative feedback.
- Cooperative binding.
- How did it make the recipe?

Tierra, a Platform for Digital Evolution

- Design Requirements/Inventions:
  - Organisms must be self-reproductive
  - Ability to out-reproduce the competition only fitness criteria
    - Avoids "artificial" fitness functions.
  - Control (jumps/calls) is effected through *templates* and *targets*, which are complementary "bit strings"
    - Jump nop1 nop0 nop1 goes to nop0 nop1 nop0
  - Organisms sense the environment
    - Dynamic "fitness" function

Tierra's Digital Organisms

- Each organism (cpu) has
  - 4 registers (A, B, C, D)
  - Instruction pointer
  - 10 word stack
- Time slicing "implements" parallel organisms
- When space for new organisms is needed, the oldest organisms are reaped (as a rule).

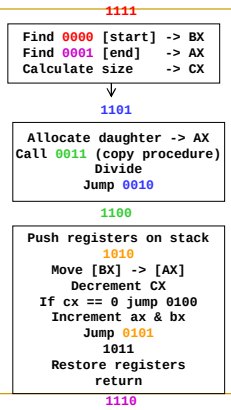
Tierra's Instruction Set

- Data Movement
  - PushA, PopA, PushB, PopB, etc for C and D
  - MOVDC (D <- C), MOVBA, COPY ([A] to [B])
- Control
  - JumpO, JumpB, Call, Ret, IfZ, nop0, nop1
- Calculation
  - subcab, subaac, inca, incb, decc, incd, zero, not, shl
- Biological and Sensing
  - adr, adrb, adrF, mal (allocate memory), divide

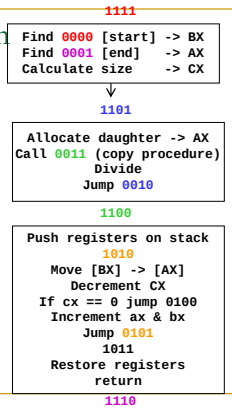
Mutational Sources

- A copy error every X copy instructions
- Cosmic rays
  - A bit in the soup gets flipped every Y instructions
  - Works because no cells are autosomes
  - Biased, not random
- Probabilistic results of instructions
  - Every so often an instruction misfires
  - E.g., incA adds 2
- No Insertion/deletions

The Tierran Ancestor



The Tierran Ancestor



Lots of redundancy

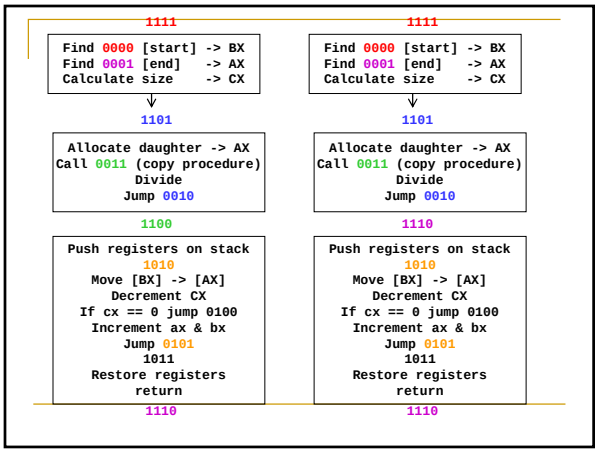
- Labels can be shortened
- Different control constructs
- Cells only replicate once or twice
- Templates can be labels
- Various return addresses can be used
- Control can use any matching code

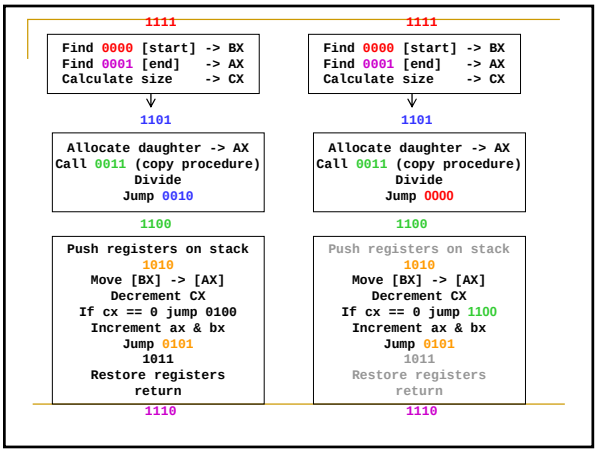
Image (and similar ones) either from or based on Tom Ray's Tierra paper.

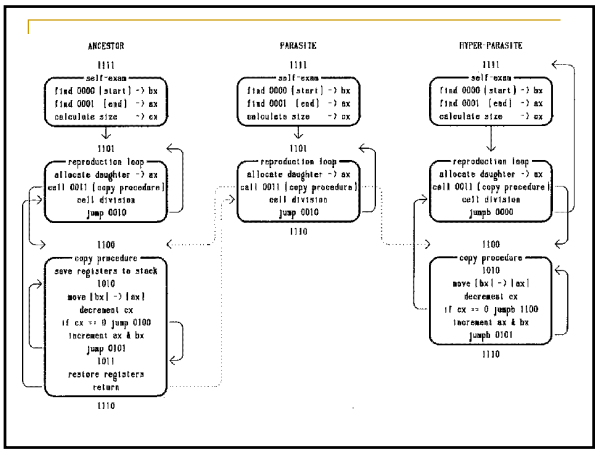
Ancestor Code

Quick Demo

Let's watch the ancestor evolve







An interesting chicken-and-egg mutation

- <C = size, B=@self>
- nop1 nop1 nop0 nop1
- mal
- call nop0 nop0 nop1 nop1
- divide
- jump nopo nop0 nop1 nop0
- ifz
- nop1 nop1 nop0 nop0
- <copy loop>

An interesting chicken-and-egg mutation

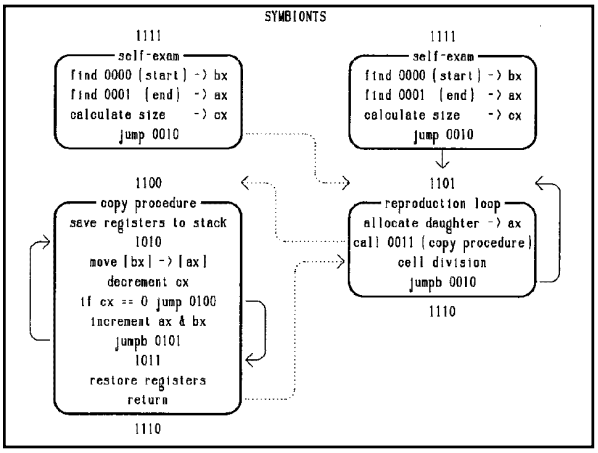
- <C = size, B=@self>
- nop1 nop1 nop0 nop1
- mal
- call nop0 nop0 nop1 nop1
- divide
- pushb (was jump) nopo nop0 nop1 nop0
- ifz
- ret (was nop1) nop1 nop0 nop0
- <copy loop>

A Copy-Once Parasite

- Stays just ahead of the reaper
  - nop1 nop1 zero not0 shl shl movdc
  - adrb nop0 nop0 pushc nop0
  - subaac
  - movba pushd nop0
  - adr nop0 nop1
  - inca
  - subcab pusha nop1 pushd nop1
  - mal
  - call nop0 nop0 nop1 nop0
  - divide

Two chances to find a copy loop

- <C = size, B = @self>
- mal pusha call movii pusha
- call nop0 nop0 nop1 nop1
- divide movii
- pusha
- mal
- call nop0 nop0 nop1 nop1
- divide mal subaac nop1
- ret zero nop1 zero (jumps to start of daughter)
- nop1 nop1 nop1 nop0



Feature or Bug?
CPU is independent of genome

- A very small self-replicating parasite (15 long)
  - Nop1
  - Adrb nop0
  - MovBA
  - Adrf nop0 nop0
  - subAAC
  - Jump nop0 nop0 nop1 nop0
  - Nop1 nop1
- Even smaller viable program:
  - Nop1

Feature or Bug?
Non-local effects

- A template can match *any* nearby target
- A request for memory can kill *any* organism, even one "fitter"
- A daughter cell can be placed *anywhere*
- Allocating a large amount of memory for a daughter can kill tens of organisms, creating a dieoff

Feature or Bug?
Spaghetti Code is a Frequent Occurrence

- Symbionts arise quite frequently
- When a target is mutated, the target in another cell is used.

Bug or Feature?
Parasites require necrophilia

- Instructions are left in memory when an organism is reaped.
- "Parasites" keep using these instructions.

Bug or Feature?
Sloppy replicators instead of Indels

- Tierra lacks insertion/deletion mutations
  - Biology uses indels
  - Harder to remove instructions without deletions
  - Harder to make room for new instructions
- Tierra makes up for it with sloppy replicators that move instructions around willy nilly
  - Buy maybe this is needed anyway?

Is Sloppiness needed to Bootstrap Complexity?

- Sloppiness (ad-hoc) mixing gave us
  - Mitochondia (ingestion without digestion)
  - Cloroplasts in bacteria (same story)
  - Gene mixing (via viruses)
  - Diploidy from Haploidy

Avida

- Inspired by Tierra, but
  - Controlled instruction pointers (less slopiness)
  - Insertion/Deletion mutations
  - 2 dimensional grid of organisms, not instructions
  - Only local next-neighbor effects
  - Fitness functions to augment reproduction
- Experiments to test biological theories
  - Evolution of Complexity
  - Evolution of Complex Functions
  - Relationship among evolution rate and landscape

Lots of questions raised by Avida paper we read.

- What happens when treated as search problem without using populations?
  - How does the system walk through deleterious steps in the search space?
- What insights are gained by treating the reduced trace of a program as its phenotype?
  - Does this remove epistatic measurement effects?
- What about sexual reproduction?
- What is the density of paths thru mutation space?
- Would a more Tierra-like system be better?
- What sized rewards would work?

Digital Biosphere

- Inspired by Tierra/Avida but
  - Want to design open-ended evolutionary frameworks
  - Focus is on evolutionary trajectories.
    - Are there principles regarding these trajectories?
  - Will exploit the constraints of physics
    - Conservation Laws!
    - Energy requirements and metabolism
  - Will eventually move to chemical modeling to get closer to biology.

Lessons

- Evolution finds corners of the search space
  - If you build it, they will exploit it
  - Complexity comes from exploiting environment
- Co-evolution makes the problem interesting and different
  - Changing fitness functions
- Designing a system for open-ended evolution is still very much an open-ended problem.

What's it all mean?
We have a source of new insights

- Watching evolving dynamical systems give insight and ideas.
- Biologists aren't trained to do this.
- Many insights will be gained that will eventually transfer over to biological thinking

More information

- Me: http://cs.washington.edu/homes/weise
- Reading course: http://cs.washington.edu/homes/weise/590ce.html
- Course will have a project based 3 credit option.

Open Questions/Future Research

- Investigate the “Worm-hole” hypothesis: no interesting genomes arise solely from single-step changes to existing genomes.
- Define phenotype as an organisms birthing trace. Now re-explain all subsequent papers in this light.
- How do we get true diversity s.t. environment changes kill half of everything?
- How do we automatically detect novelty?

On “designing” open-ended evolutionary systems.

In the days when Sussman was a novice, Minsky once came to him as he sat hacking at the PDP-6.

"What are you doing?", asked Minsky.

"I am training a randomly wired neural net to play Tic-tac-toe" Sussman replied.

"Why is the net wired randomly?", asked Minsky.

"I do not want it to have any preconceptions of how to play", Sussman said.

Minsky then shut his eyes.

"Why do you close your eyes?", Sussman asked his teacher.

"So that the room will be empty."

At that moment, Sussman was enlightened.

The Value of Diploidy?

- Most of the genes perform a walk from viable organism to viable organism.
- Some of the genes walk through non-viable points in the search space.