Genetic Algorithms and Evolvable Hardware on a Cell Matrix

Nick Macias

Cell Matrix Corporation

Genetic Algorithm (GA) Approach to Problem Solving

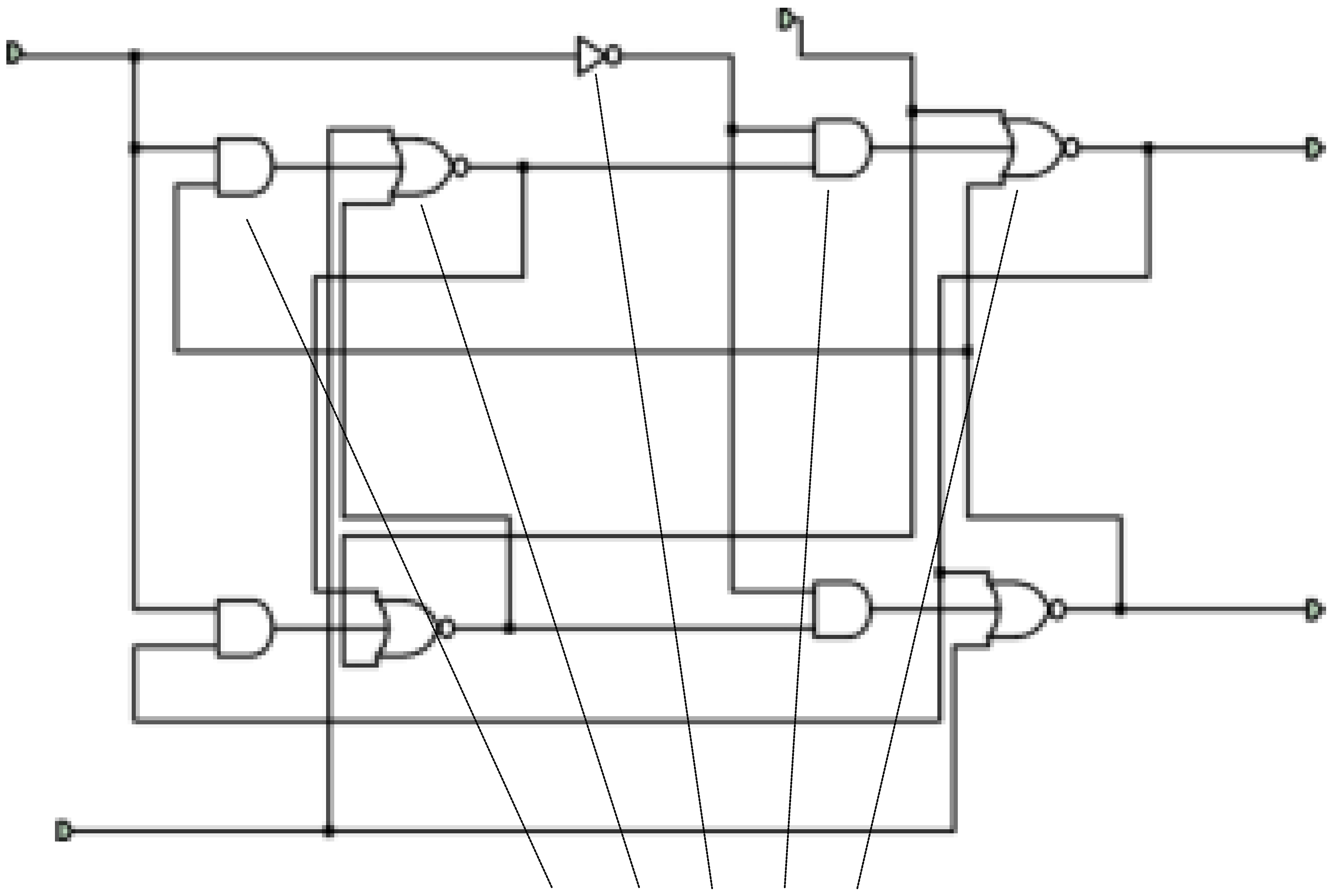
- Implement a potential solution with unspecified parameters (*individual*)
  - Start with semi-random parameter settings
- Determine fitness of this individual
- Repeat for a population of individuals
- Combine most-fit individuals
- Repeat for a number of generation
- Mutate and/or create random individuals

Application to Hardware Design

- GAs are well suited to finding solutions which are difficult to construct but easy to recognize
- Good example: Designing computer circuits
  - Fixed set of inputs, well-define output behavior
- IDEA: Use a GA to evolve a digital circuit

Evolvable Hardware

- Design mapping from circuits to strings (genome)
- Evaluate fitness of circuits
- Mate/mutate strings
- Try to evolve circuits with high fitness level



Genome: 01 10 11 01 10…01101001110101...

Fitness Evaluation

- Simulate evolved circuits in software
- Slow, since HW is parallel and simulations are sequential
- Inexact-simulators aren’t perfect (Thompson)

Reconfigurable Hardware

- Allows individuals within the GA’s population to be implemented in hardware-faster fitness evaluation
- Much work in the 1990s, especially Xilinx 6200 series FPGAs
- Newer devices don’t work as well for EHW
- *Extrinsic* evolution (external control)

EHW on a Cell Matrix

- Nice general-purpose reconfigurable platform for EHW work
- Can treat a set of truth tables as a single genome
- Evolve the truth tables, thereby evolving the hardware configuration

Genome Construction

TT_1	TT_2	TT_3
TT_4	TT_5	TT_6

Genome: $TT_1\,TT_2\,TT_3\,TT_4\,TT_5\,TT_6$

Cell Matrix Advantages for EHW

- All truth tables are valid-impossible to damage the device
- Natural configuration mechanism-all bits “equally important”
- Tools available (and free!)
- *Intrinsic* evolution potential (internal control): Parallel, fast

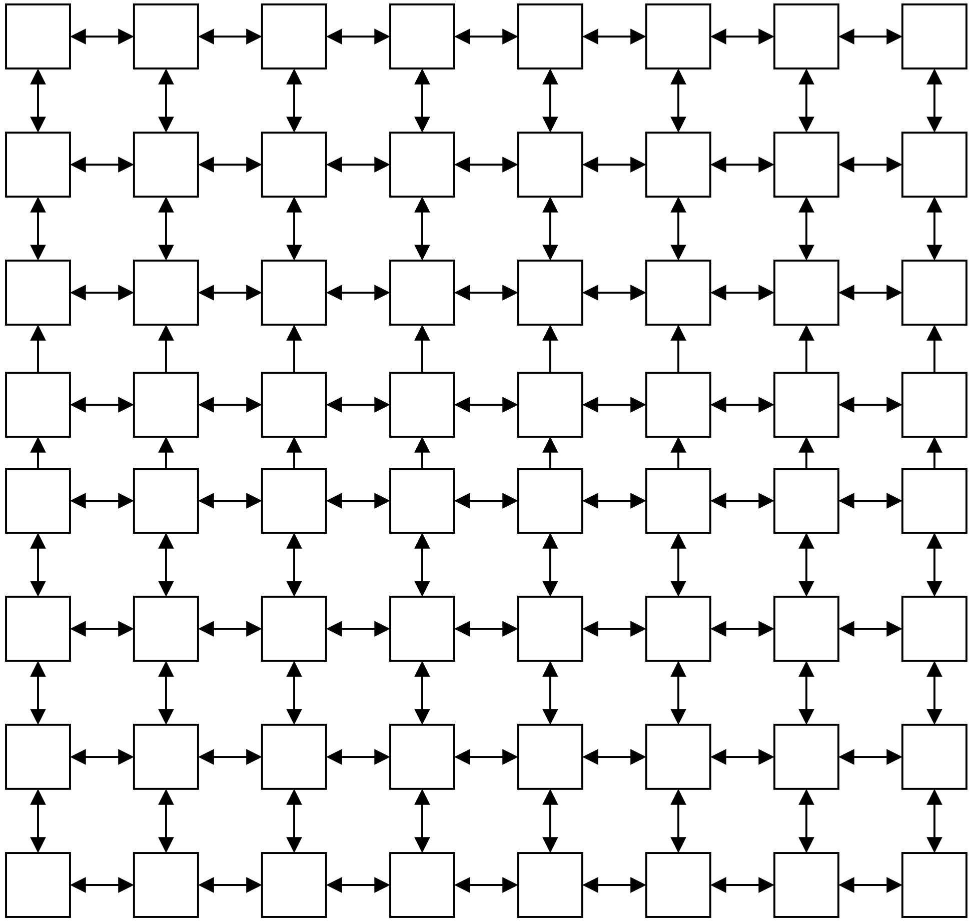
Cell Matrix Advantages (cont.)

- Flexible hardware supports a wide range of processing (e.g. parallel sensor/image input)
- Fault tolerant hardware-essential for very large systems
- Ease of manufacture
- Autonomous operation-remote environments, etc.

Work To Date

- Evaluated suitability of Cell Matrix to a parallel GA
- “Ringed GA”-Combines two parallel models
- Presented at 1999 Congress on Evolutionary Computation

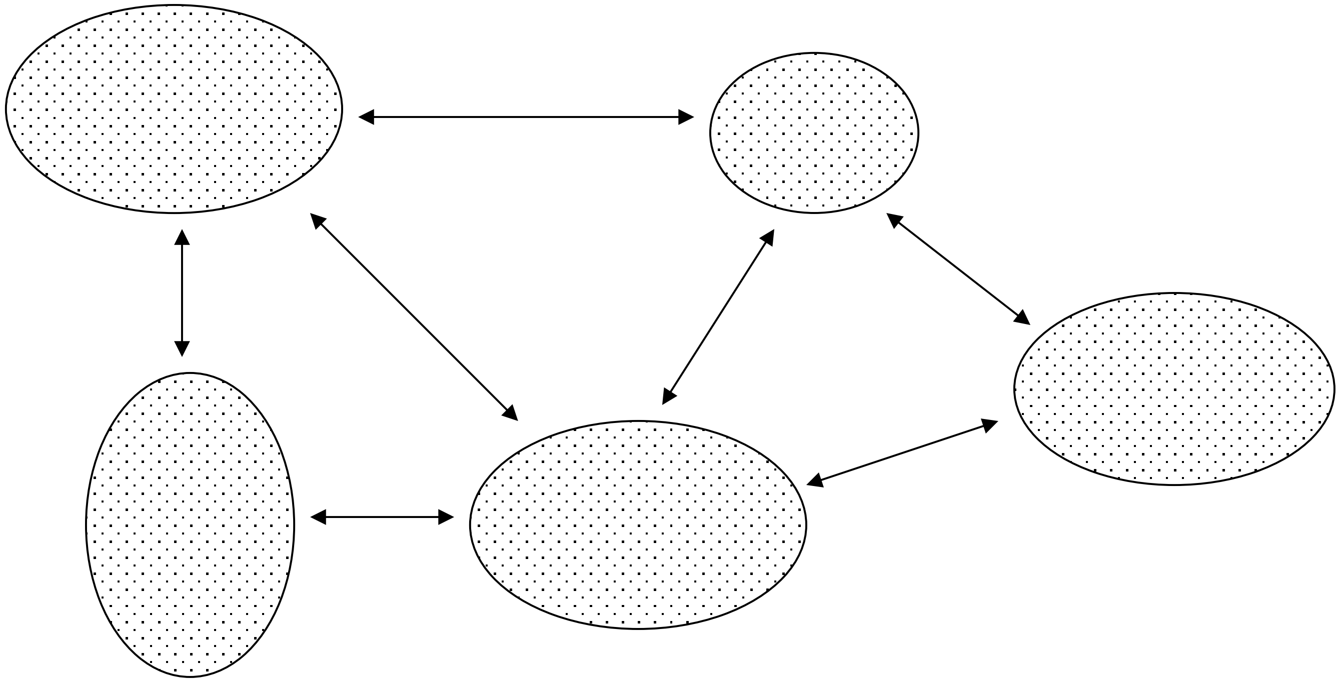
Grid Model



Grid Model

- Very fast since number of interactions is limited
- Relatively easy to implement
- Prone to local maxima

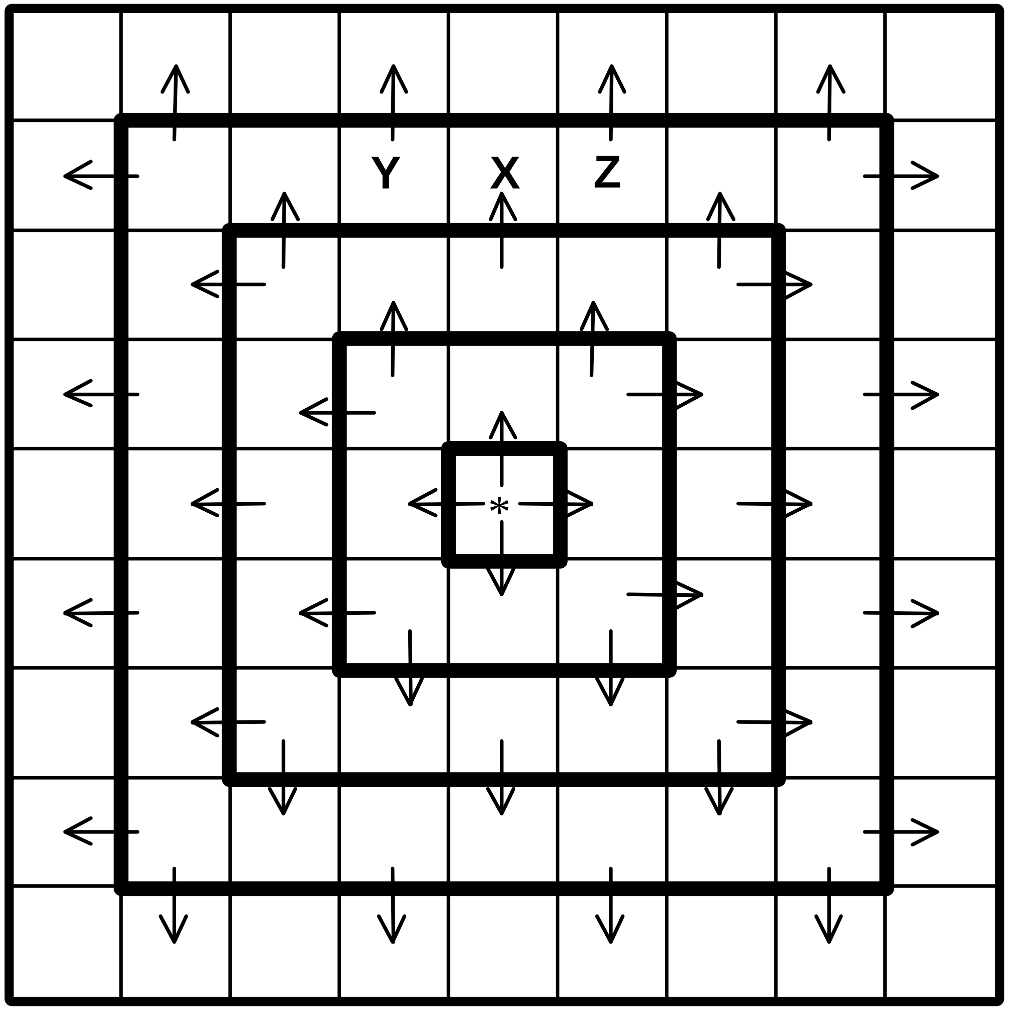
Island Model



Island Model

- Slower than Grid Model since each island’s evolution speed depends on island population
- Periodic migrations distribute local maxima
- Tends towards global maxima

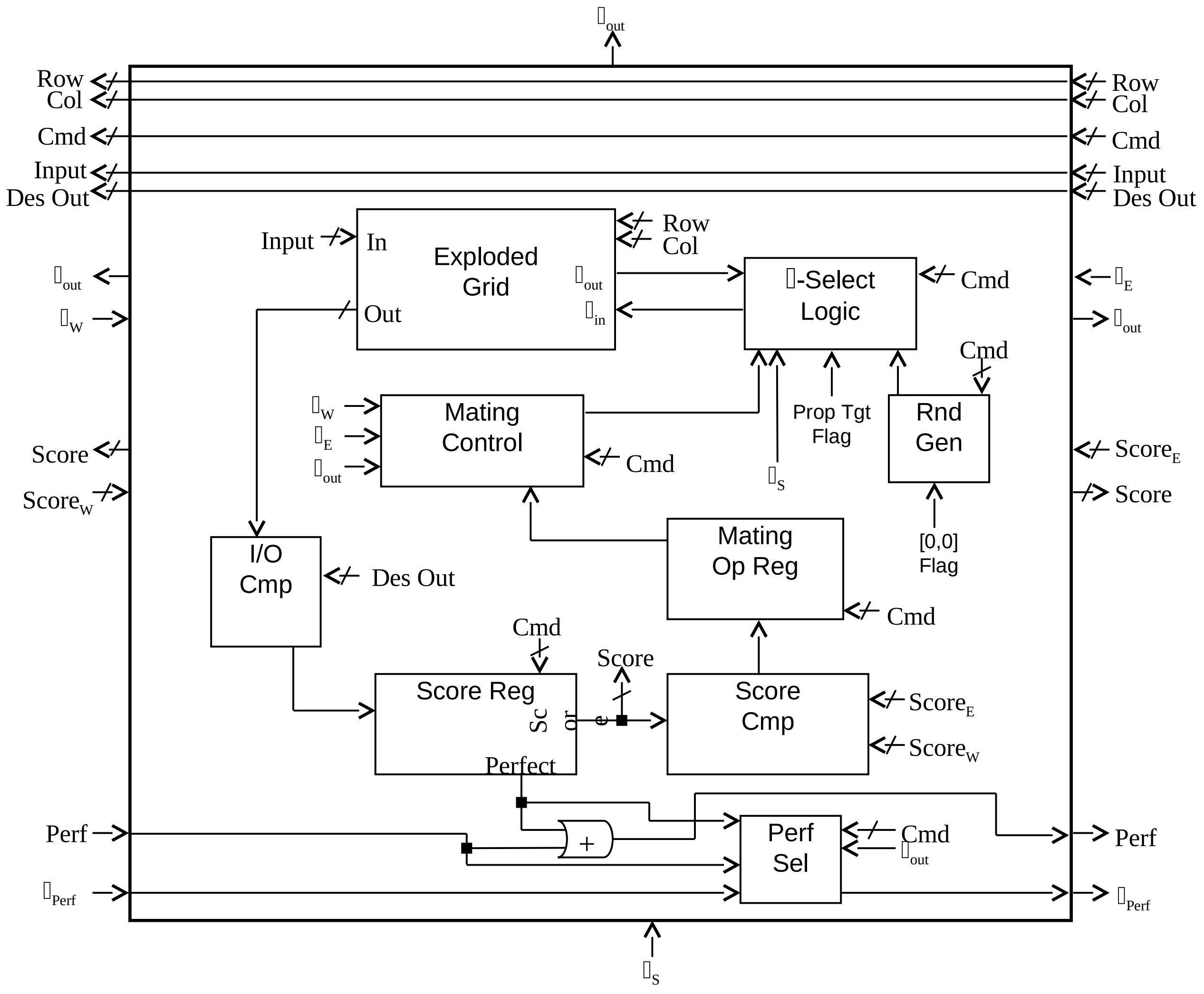
Ringed GA



Ringed GA

- Combines parallelism of grid model with migrations of island model
- Natural implementation on Cell Matrix
- Distribute the GA among all individuals
- Trades HW for speed

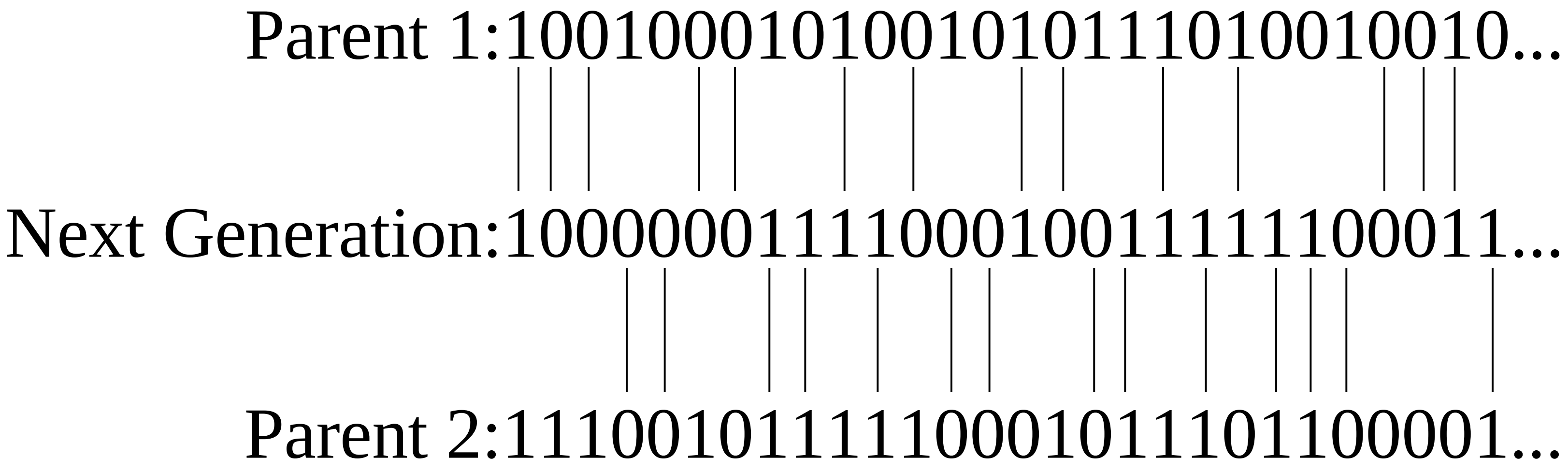
Circuit for One Individual



Ringed GA Experiments

- 8 rings, 225 individuals
- Migrate every 9 GA cycles
- Prob(Random bit=1)=0.15

Mating Options:Selection



Parent 1:1001000101001010111010010010...

Next Generation:1000000111100010011111100011...

Parent 2:1110010111110001011101100001...

Might want to skew selection probability
based on fitness difference

Mating Options:OR

Parent 1:10010001010010101110100100 10...

Next Generation:11110101111110101111111110010...

Parent 2: 11100101111100010111011000 01...

(Can also do AND)

Mating Options:Crossover

Select random crossover point

Parent 1:1001000101001010111010010010...

Next Generation:1001000111110001011101100001...

Parent 2: 1110010111110001011101100001...

Can have multiple crossover points

Can also be done per cell

Ringed GA Experiments

- LOTS of mating possibilities
- For present experiments, used simple selection
  - PROB(better parent’s bit)=0.60
- Allow random mutations in next generation
  - PROB(mutation)=0.0125 per bit

First Expeiment: 4-Bit Odd Parity Generator

- Input 4 bits, output 1 bit
- Want total # of bits to be odd

f(0,0,0,0)=1   f(0,1,0,0)=0

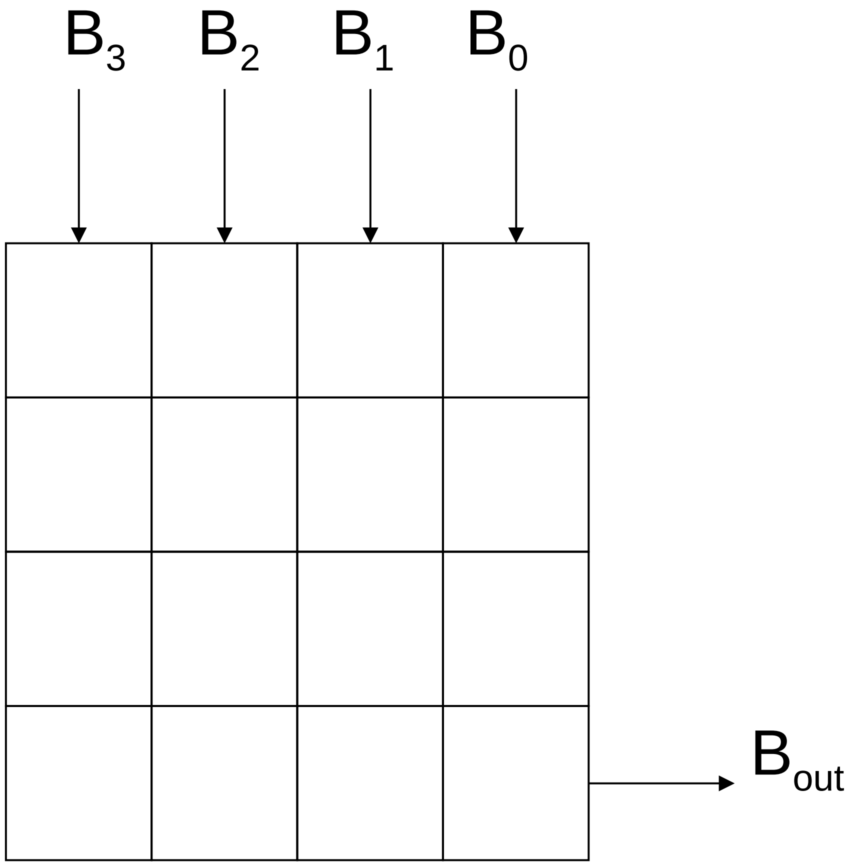
f(0,0,0,1)=0   f(0,1,0,1)=1

f(0,0,1,0)=0   f(0,1,1,0)=1

f(0,0,1,1)=1   f(0,1,1,1)=0

etc.

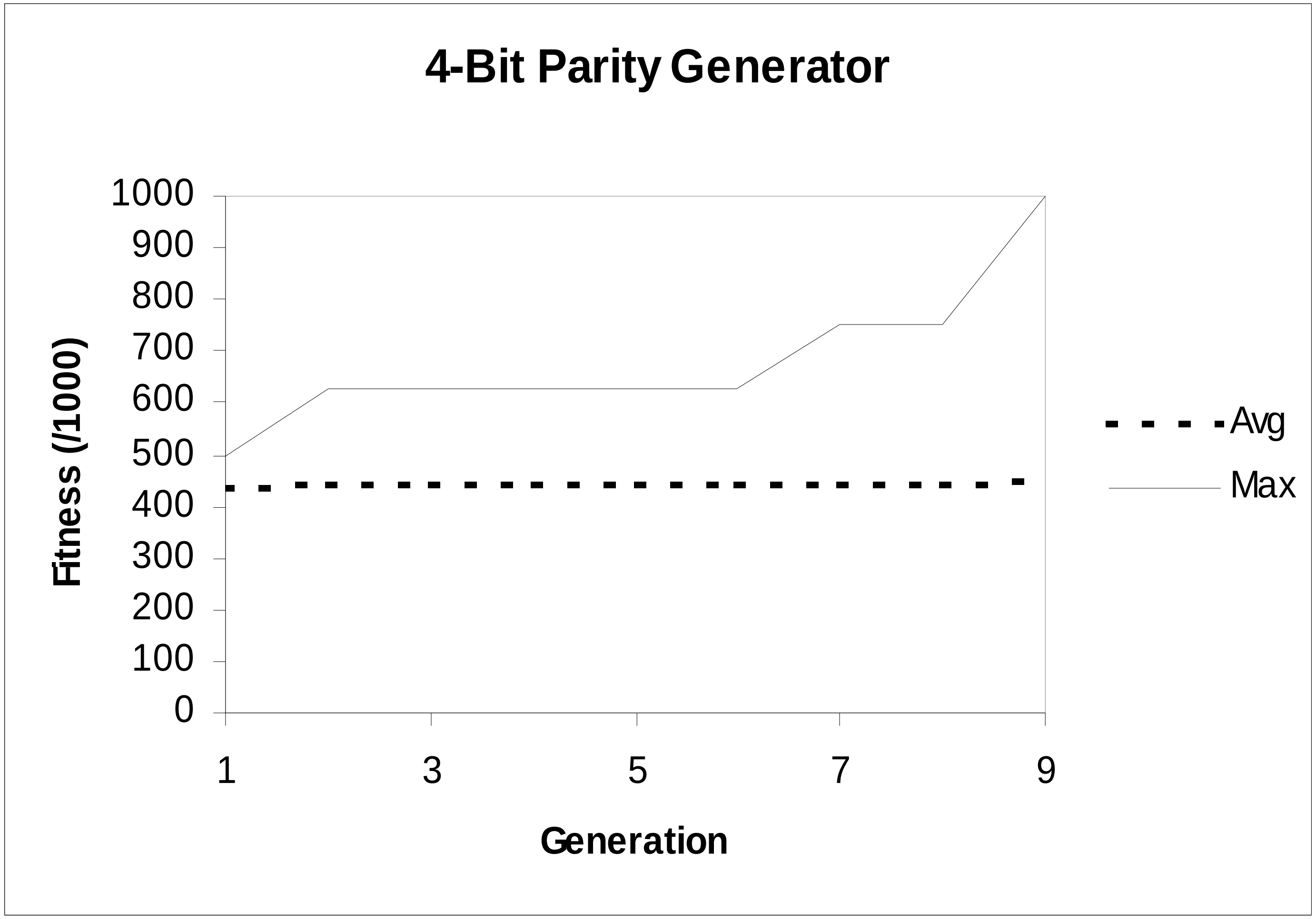
Target Circuit Layout



Fitness Function

- Simple count of # of correct outputs across all input combinations
- Scale to 1000=perfect score
- (Slightly questionable measure!)

Parity Generator Results

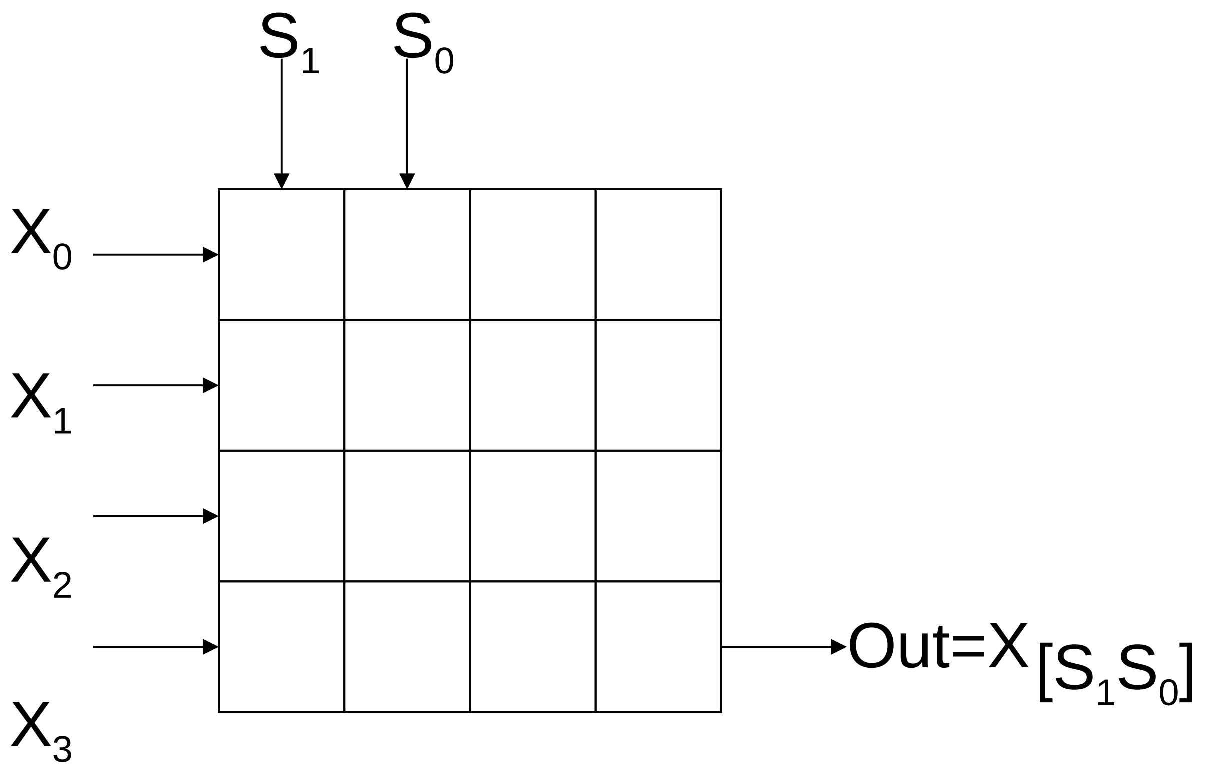


Second Experiment: 4-1 Multiplexer

- Input four bits X_0-X_3
- Also input two selection bits S_0, S_1
- Select one input based on S_0 and S_1

S_1S_0	Output
0 0	X_0
0 1	X_1
1 0	X_2
1 1	X_3

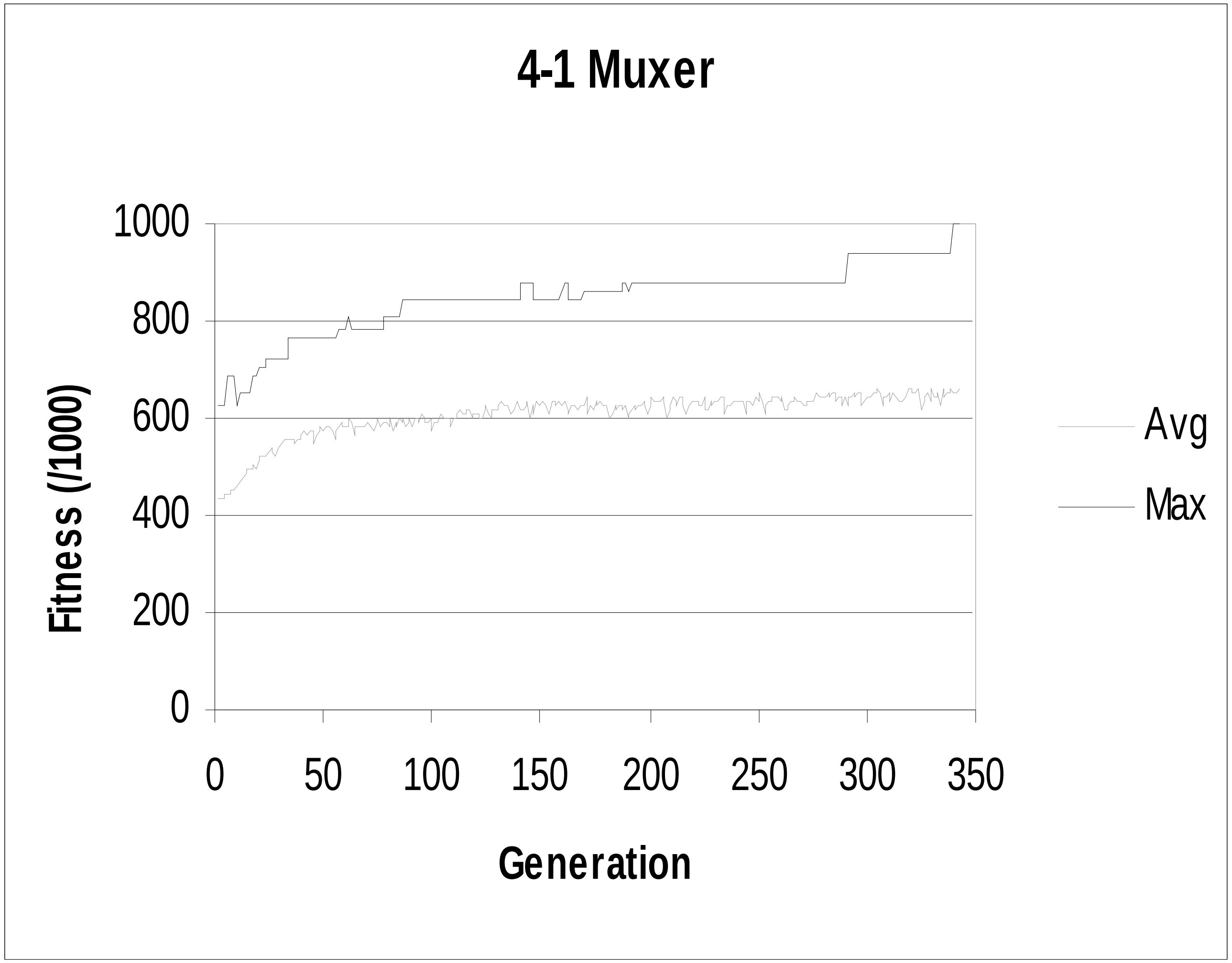
Target Circuit Layout



Fitness Function

- Same as parity generator
- Simple count of # of correct outputs

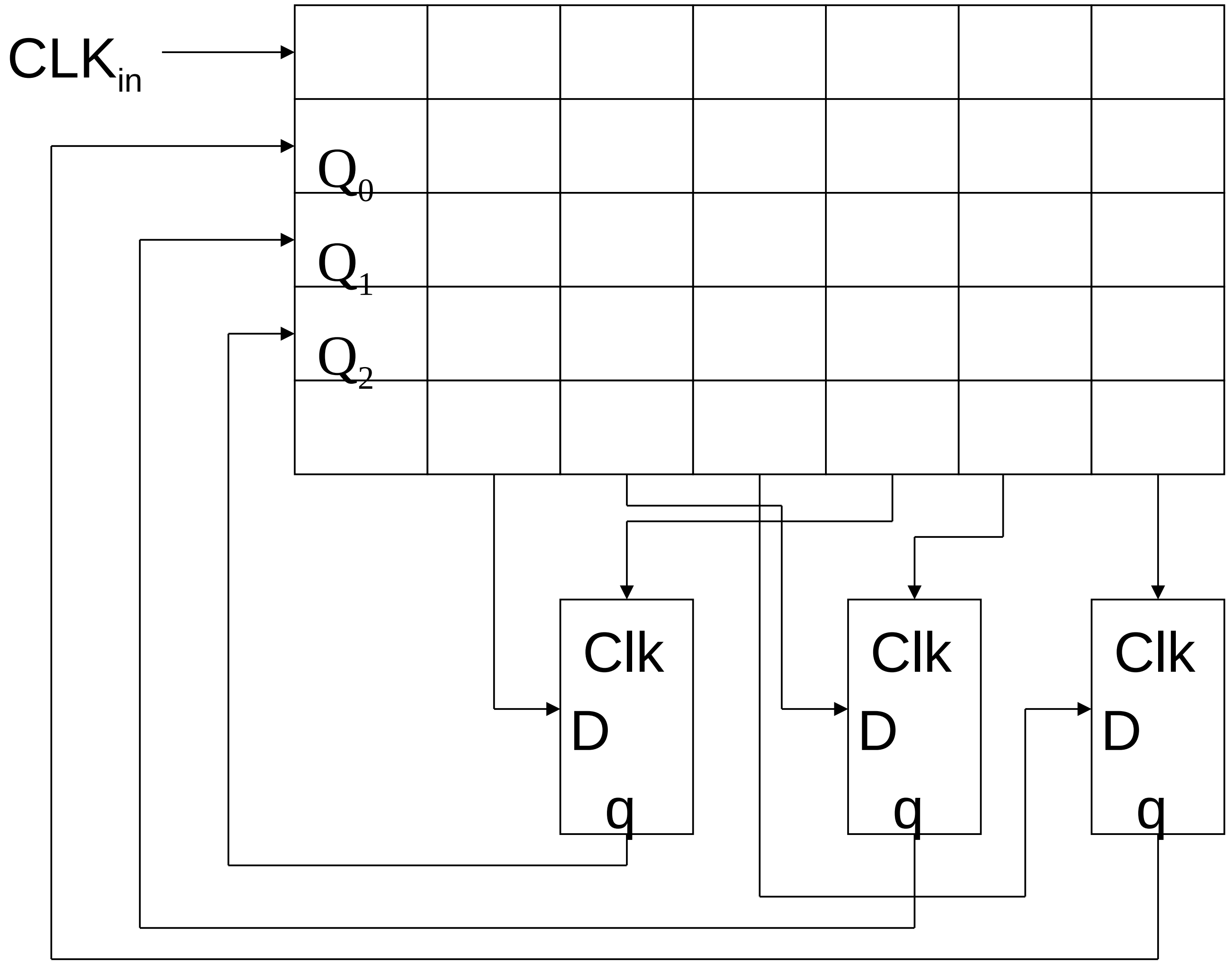
4-1 Multiplexer Results



Sequential Experiment

- 3-bit Counter
- Designed general state machine
  - 8 States
- Try to evolve state transition logic

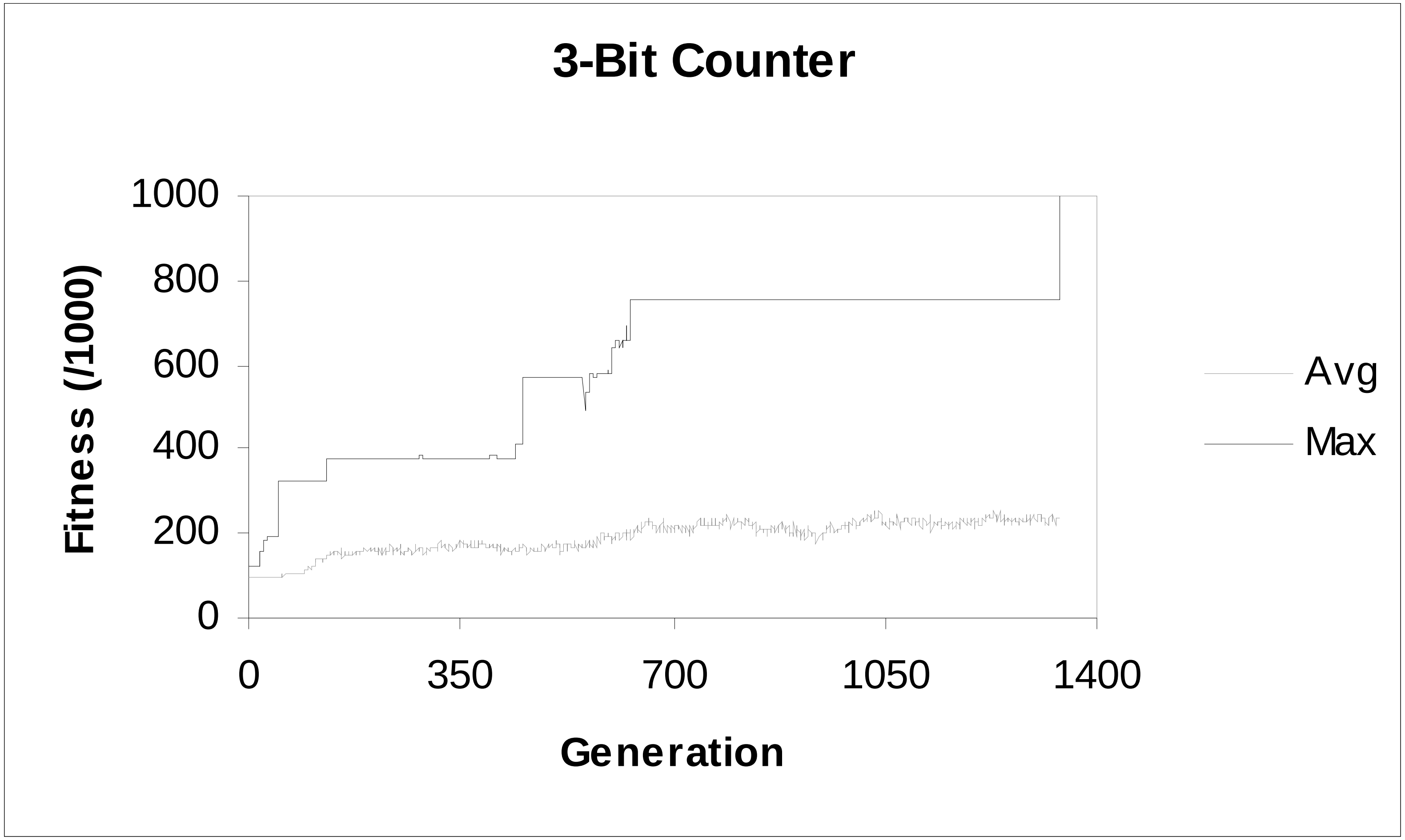
Target Circuit Layout
(all composed of Cell Matrix Cells)



Fitness Function

- More complex-needs to account for sequential behavior
- Start with fitness=0
- Run 16 clock steps
  - If output matches desired value, +30
  - else if output is one more than previous, +5
  - else if output different from previous, +1
  - else +0

3-Bit Counter Results



Summary

- Above work is proof-of-concept
- Focused on particular parallel implementation (Ringed GA)
- Many other areas for exploration
- Lots of interesting research/thesis projects!

Algorithm Development

- Develop other intrinsic algorithms beside the Ringed GA
- New ways to exploit parallelism of Cell Matrix
- New ways to exploit self-configurability of Cell Matrix
- Implementation details

Using the GA

- Try evolving different circuits:
  - Adder/Subtractor
  - Sort-optimize # of steps?
  - Multiplier (fuzzy?)
  - Image recognition
  - Fault-tolerant circuit
  - Game playing
- Evolve Sequential Circuits

More Projects...

- Evolve self-modifying (C-mode) circuits- **completely new area**
- Evolve the GA itself
- Evolve low-level modules, then evolve higher-order circuits built from those (Hugo’s multi-module evolution question)
- Anything else you can think of!

What Next?

- We can tailor an API for a simulator based on your project
- Wide choice of platforms, languages, etc.
- Visit the Web site (www.cellmatrix.com)
- Play with the simulators
- Talk to us about ideas