Evolutionary Learning, Part 1

Some Examples of Biologically Inspired AI

- Neural networks

- Evolutionary computation (e.g., genetic algorithms)

- Immune-system-inspired computer/network security

- Insect-colony optimization (ants, bees, etc.)

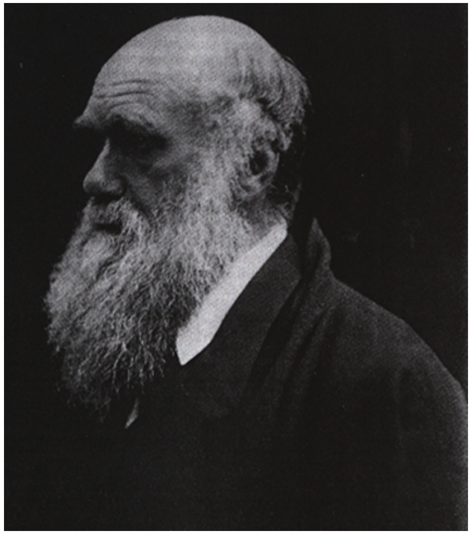
- Slime-mould path-finding

- Swarm intelligence (e.g., decentralized robots)

Evolution by Natural Selection

Evolution by Natural Selection

in computers



Charles Darwin, 1809–1882

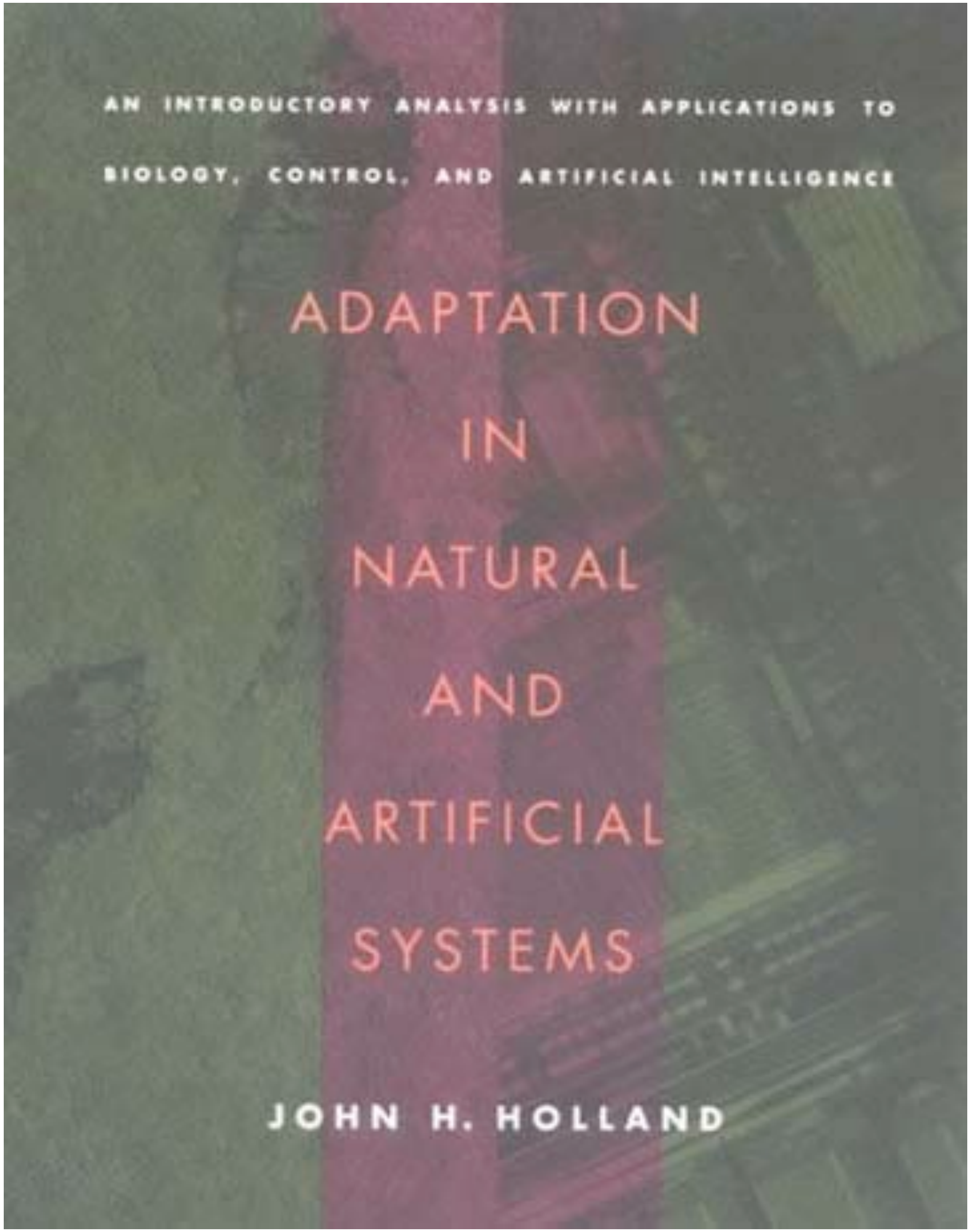
- Organisms inherit traits from parents

Computer (e.g., programs)

- Traits are inherited with some variation, via mutation and sexual recombination
- Due to competition for limited resources, the organisms best adapted to the environment tend to produce the most offspring.
- This way traits producing adapted individuals spread in the population

John Holland

Genetic Algorithms



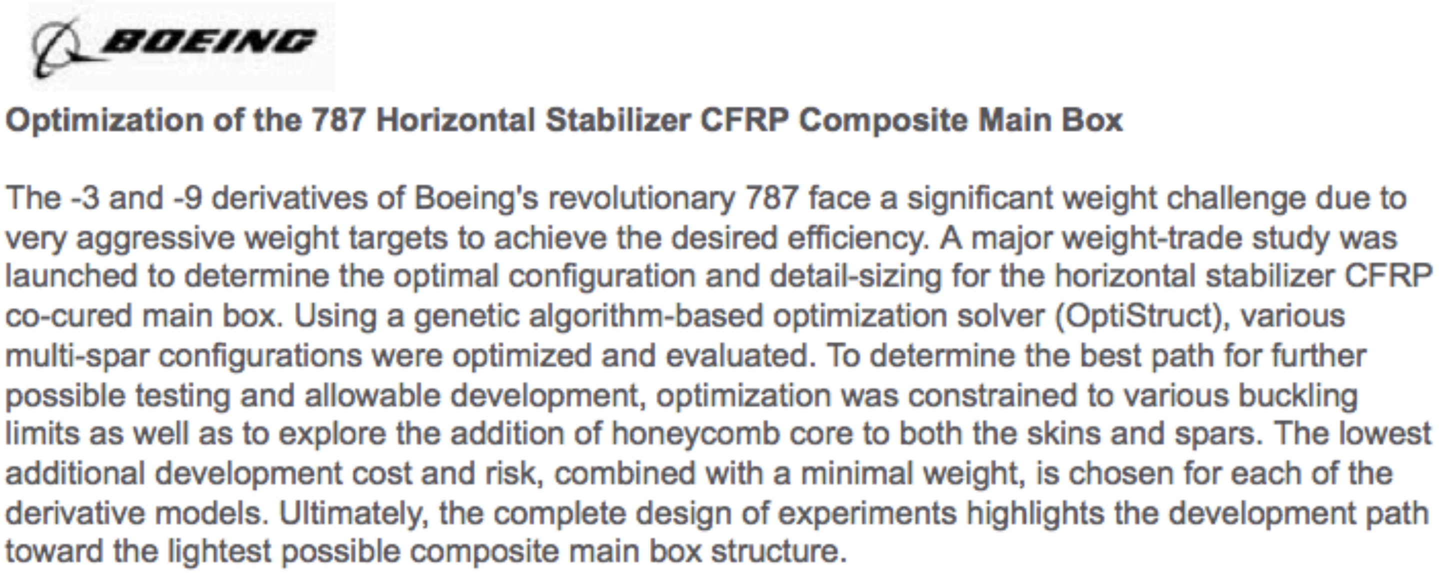
AN INTRODUCTORY ANALYSIS WITH APPLICATIONS TO BIOLOGY, CONTROL, AND ARTIFICIAL INTELLIGENCE

ADAPTATION IN NATURAL AND ARTIFICIAL SYSTEMS

JOHN H. HOLLAND

Some Real-World Examples of Genetic Algorithms

Designing parts of aircraft (GE and Boeing)



BOEING

Optimization of the 787 Horizontal Stabilizer CFRP Composite Main Box

The -3 and -9 derivatives of Boeing's revolutionary 787 face a significant weight challenge due to very aggressive weight targets to achieve the desired efficiency. A major weight-trade study was launched to determine the optimal configuration and detail-sizing for the horizontal stabilizer CFRP co-cured main box. Using a genetic algorithm-based optimization solver (OptiStruct), various multi-spar configurations were optimized and evaluated. To determine the best path for further possible testing and allowable development, optimization was constrained to various buckling limits as well as to explore the addition of honeycomb core to both the skins and spars. The lowest additional development cost and risk, combined with a minimal weight, is chosen for each of the derivative models. Ultimately, the complete design of experiments highlights the development path toward the lightest possible composite main box structure.

http://resources.altair.com/altairadmin/images/resource_library/graphics-en-US/htc07_boeing.jpg

Spacecraft antenna design (NASA)

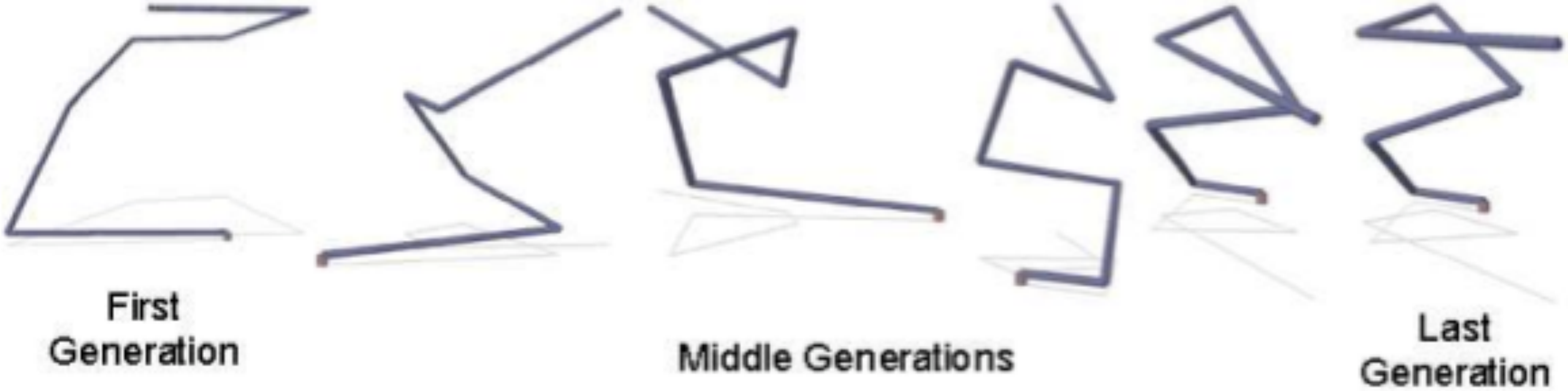
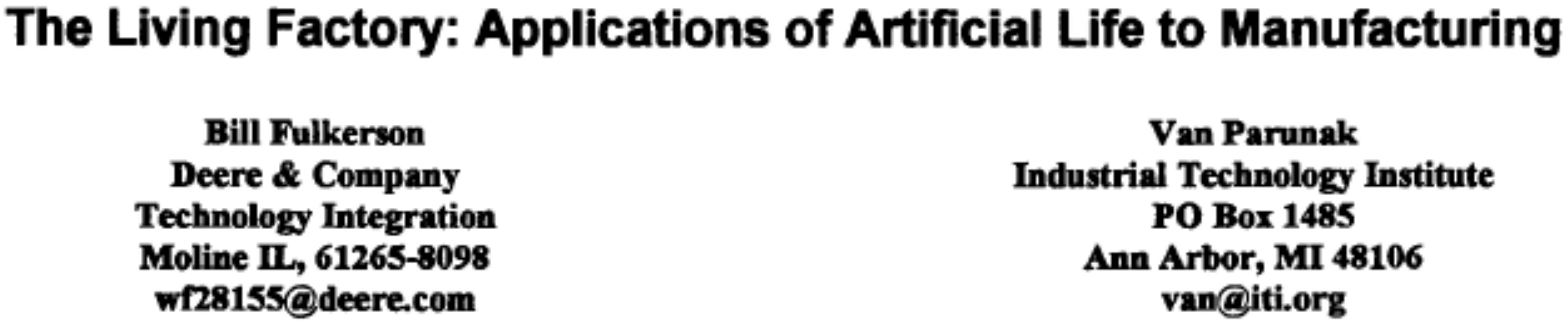


Fig. 10. Sequence of evolved antennas leading up to antenna ST5-33.142.7.

http://idesign.ucsc.edu/papers/lohn_gptp05.pdf

Assembly line scheduling (John Deere Co.)



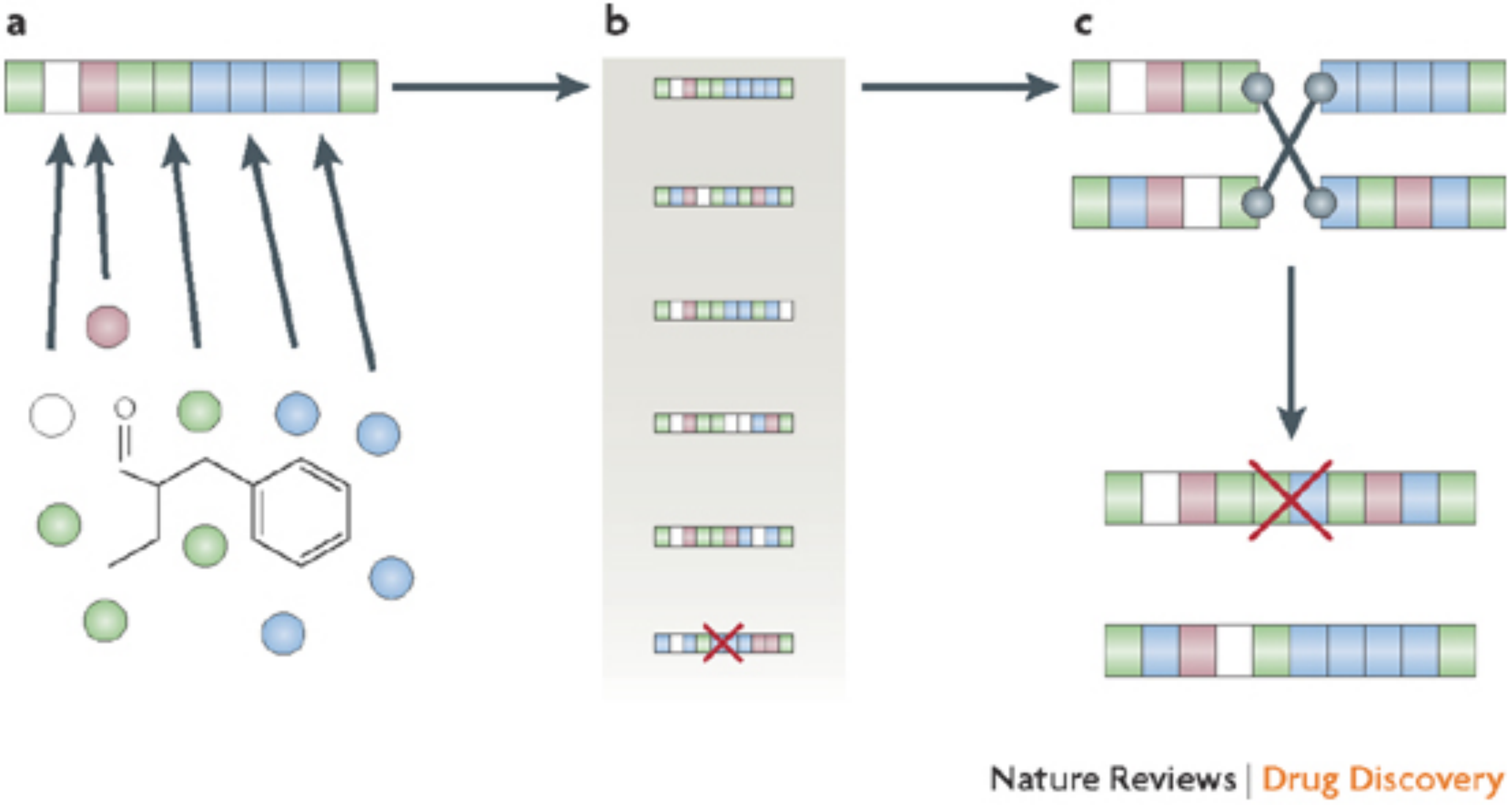
The Living Factory: Applications of Artificial Life to Manufacturing

Bill Fulkerson
Deere & Company
Technology Integration
Moline IL, 61265-8098
wf28155@deere.com

Van Parunak
Industrial Technology Institute
PO Box 1485
Ann Arbor, MI 48106
van@iti.org

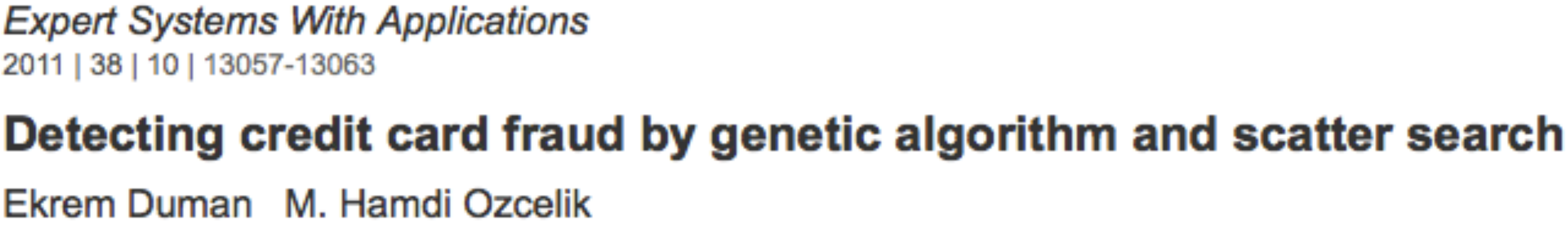
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=398998

Automated drug discovery (several companies)



http://www.nature.com/nrd/journal/v7/n8/fig_tab/nrd2615_F4.html

Fraud detection (credit cards, financial trading)



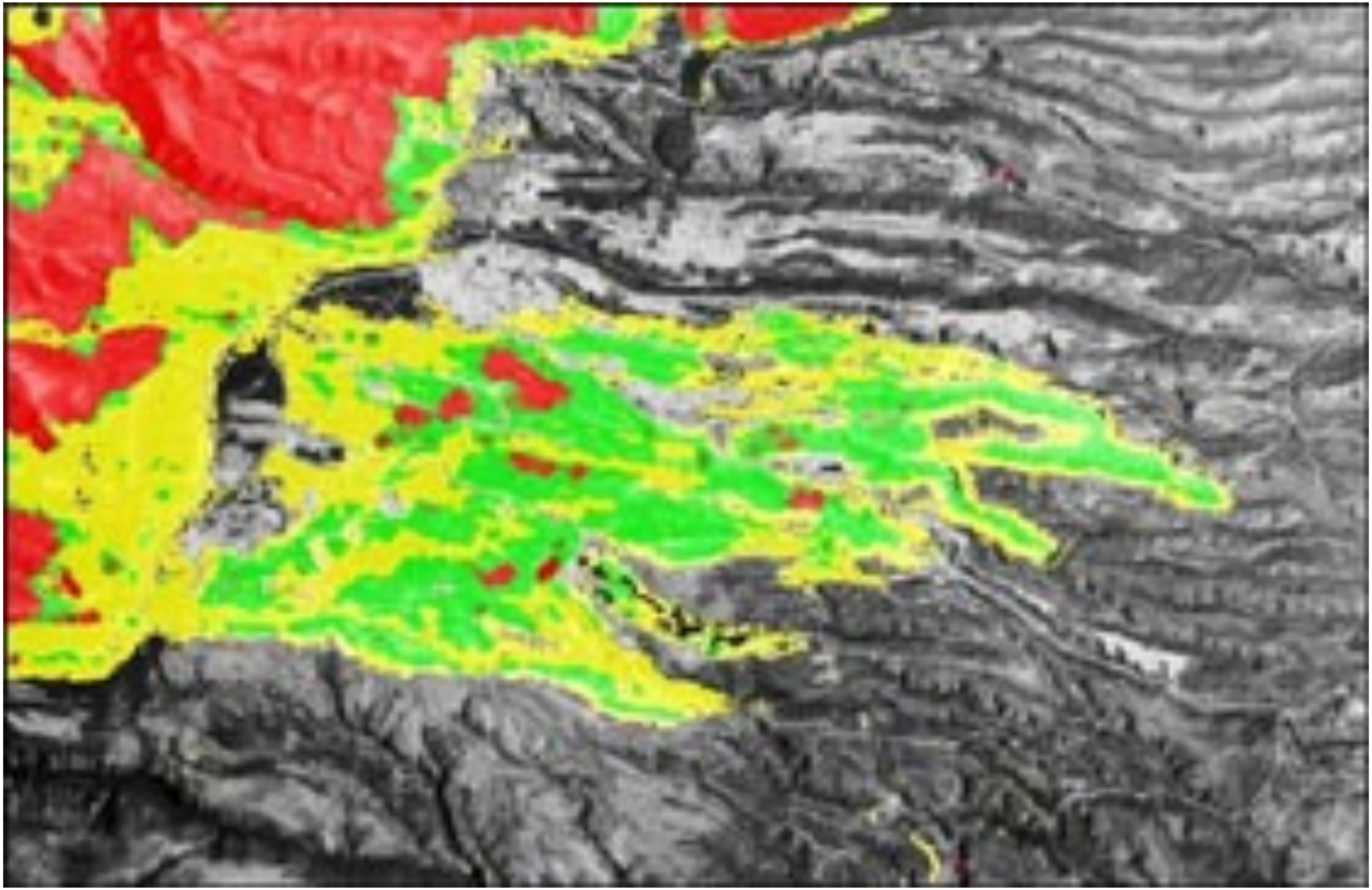
Expert Systems With Applications

2011 | 38 | 10 | 13057-13063

Detecting credit card fraud by genetic algorithm and scatter search

Ekrem Duman   M. Hamdi Ozcelik

Automated analysis of satellite images (Los Alamos National Lab)



http://www.eurekalert.org/features/doe/images/danl-lag080702.1.jpg

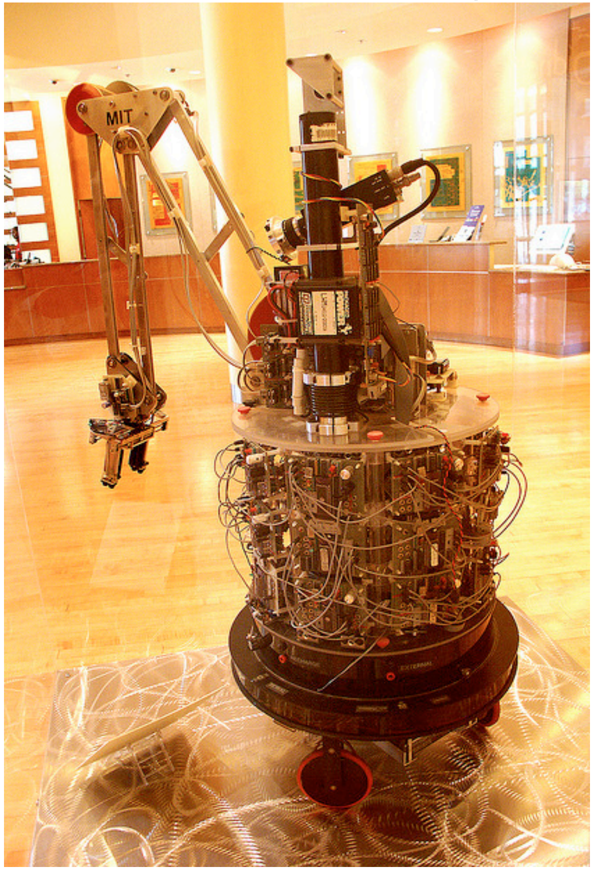
Generation of realistic computer animation (*Lord of the Rings: The Return of the King* and *Troy)*

http://www.wired.com/wired/archive/12.01/stuntbots.html

Genetic Algorithm Example:

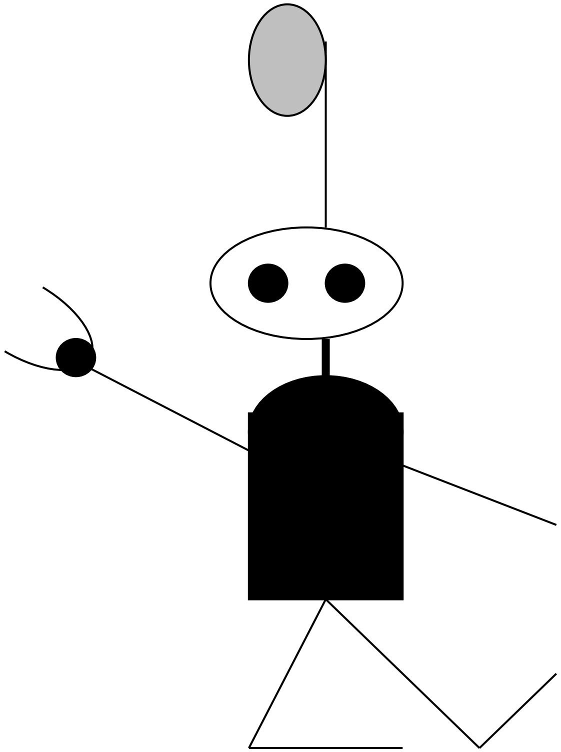
Evolving a Control Program for a Virtual “Robot”

Herbert:
The Soda Can Collecting Robot
(Connell, Brooks, Ning, 1988)

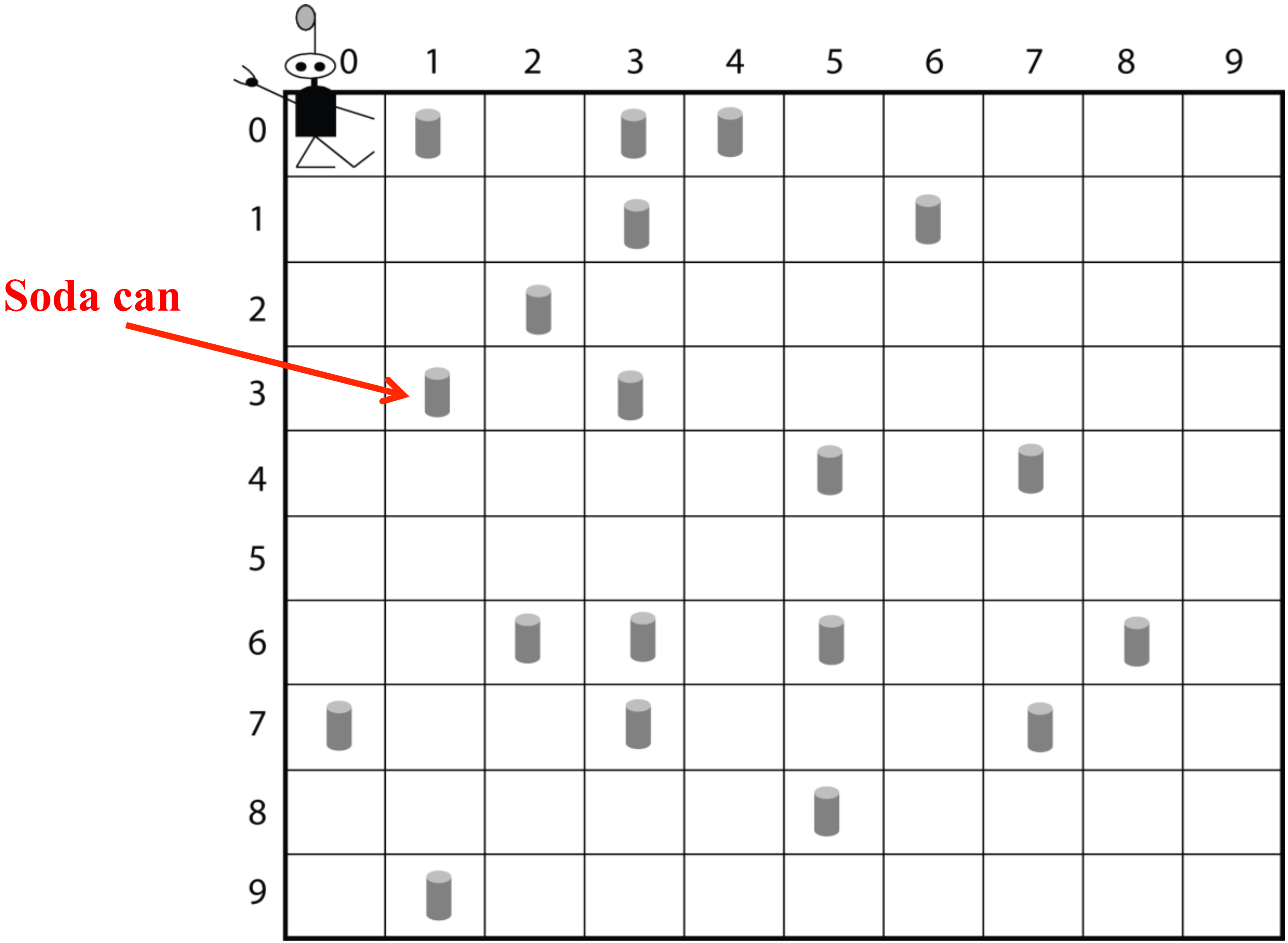


http://cyberneticzoo.com/?p=5516

Robby:
The Virtual Soda Can Collecting Robot
(Mitchell, 2009)



Robby’s World


Soda can

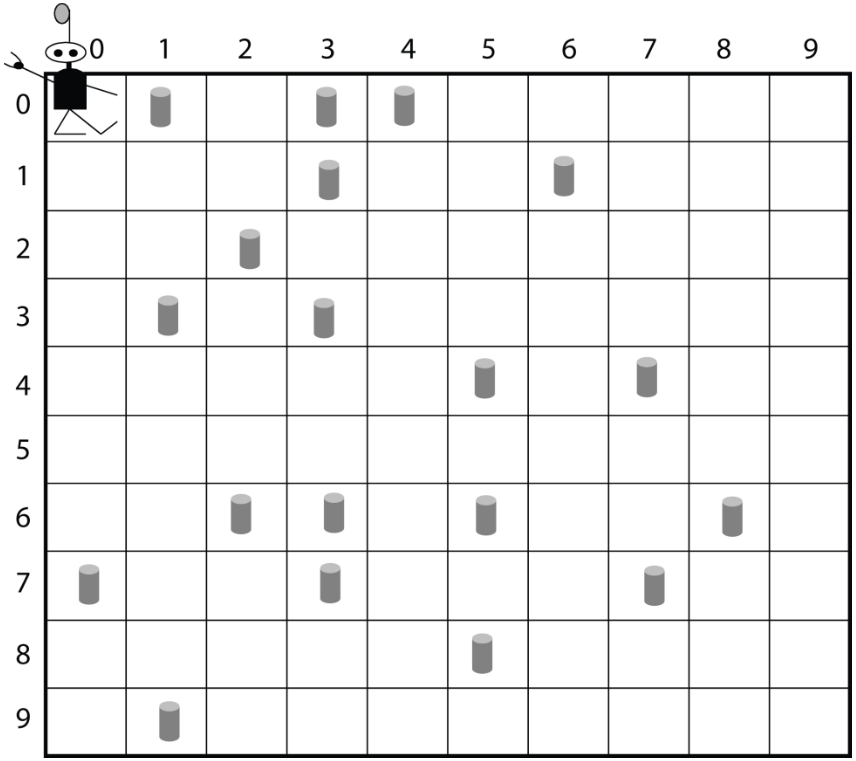
What Robby Can See and Do

Input:

Contents of **N**orth, **S**outh, **E**ast, **W**est, **C**urrent

Possible actions:

```
Move N

Move S

Move E

Move W

Move random

Stay put

Try to pick up can
```



Rewards/Penalties (points):

Picks up can: **10**

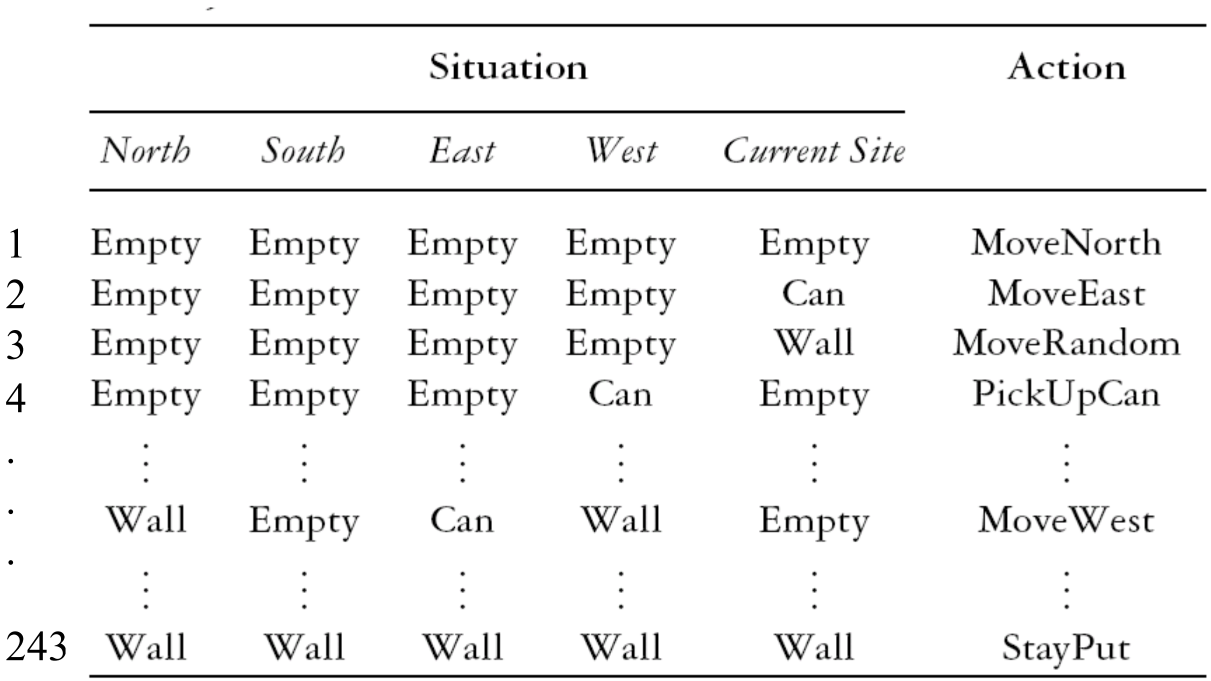
Tries to pick up can on empty site: **-1**

Crashes into wall: **-5**

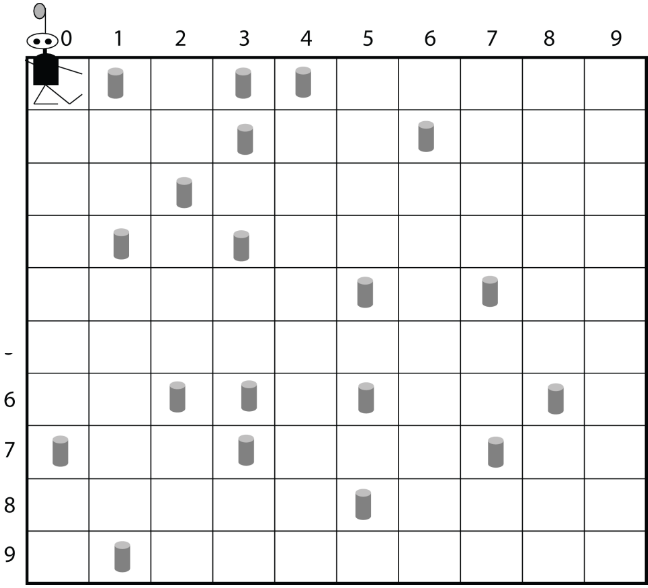
Robby's Score: Sum of rewards/penalties

Goal: Use a genetic algorithm to evolve a control program (i.e., *strategy*) for Robby.

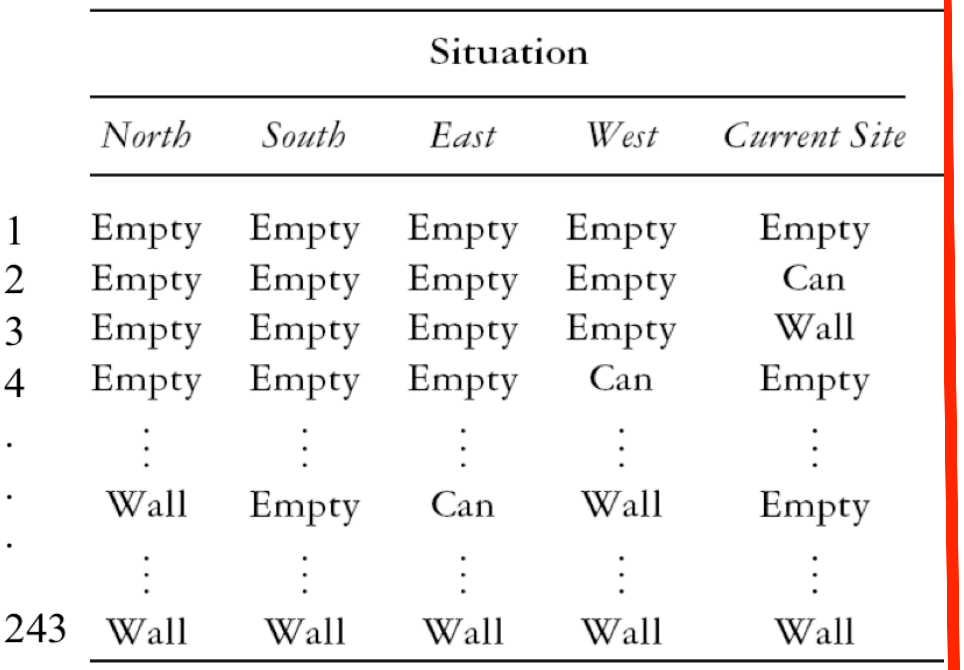
One Example Strategy



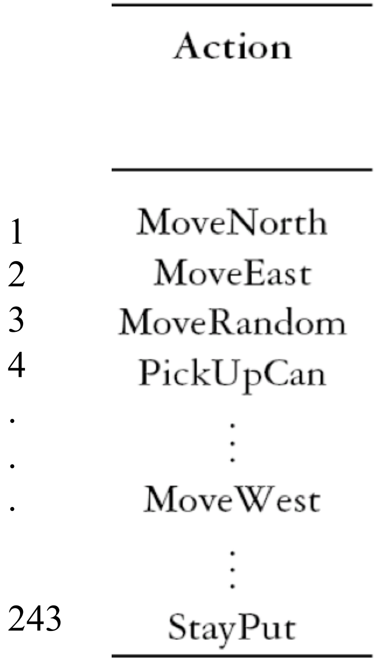
	Situation					Action
	North	*South*	*East*	*West*	*Current Site*	
1	Empty	Empty	Empty	Empty	Empty	MoveNorth
2	Empty	Empty	Empty	Empty	Can	MoveEast
3	Empty	Empty	Empty	Empty	Wall	MoveRandom
4	Empty	Empty	Empty	Can	Empty	PickUpCan
.	⋮	⋮	⋮	⋮	⋮	⋮
.	Wall	Empty	Can	Wall	Empty	MoveWest
.	⋮	⋮	⋮	⋮	⋮	⋮
243	Wall	Wall	Wall	Wall	Wall	StayPut



Encoding a Strategy

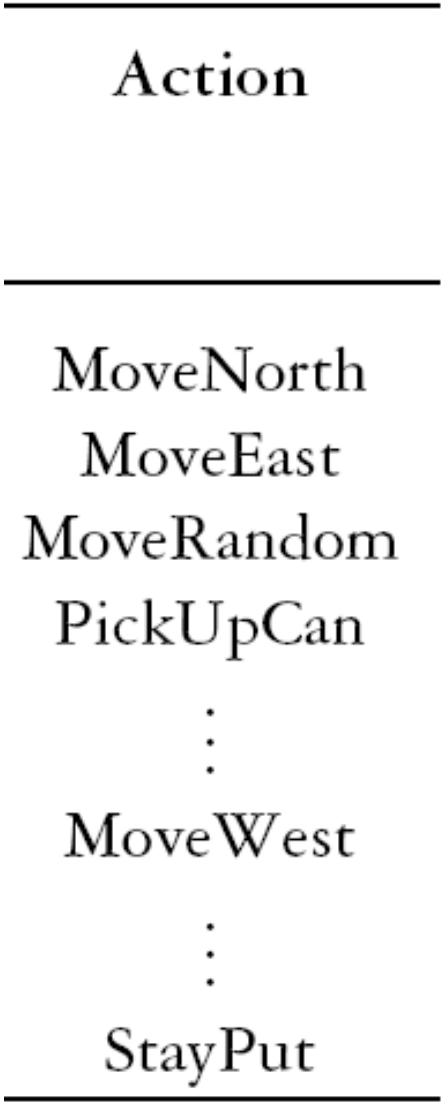


	Situation				
	North	*South*	*East*	*West*	*Current Site*
1	Empty	Empty	Empty	Empty	Empty
2	Empty	Empty	Empty	Empty	Can
3	Empty	Empty	Empty	Empty	Wall
4	Empty	Empty	Empty	Can	Empty
.	⋮	⋮	⋮	⋮	⋮
.	Wall	Empty	Can	Wall	Empty
.	⋮	⋮	⋮	⋮	⋮
243	Wall	Wall	Wall	Wall	Wall



	Action
1	MoveNorth
2	MoveEast
3	MoveRandom
4	PickUpCan
.	⋮
.	MoveWest
.	⋮
243	StayPut

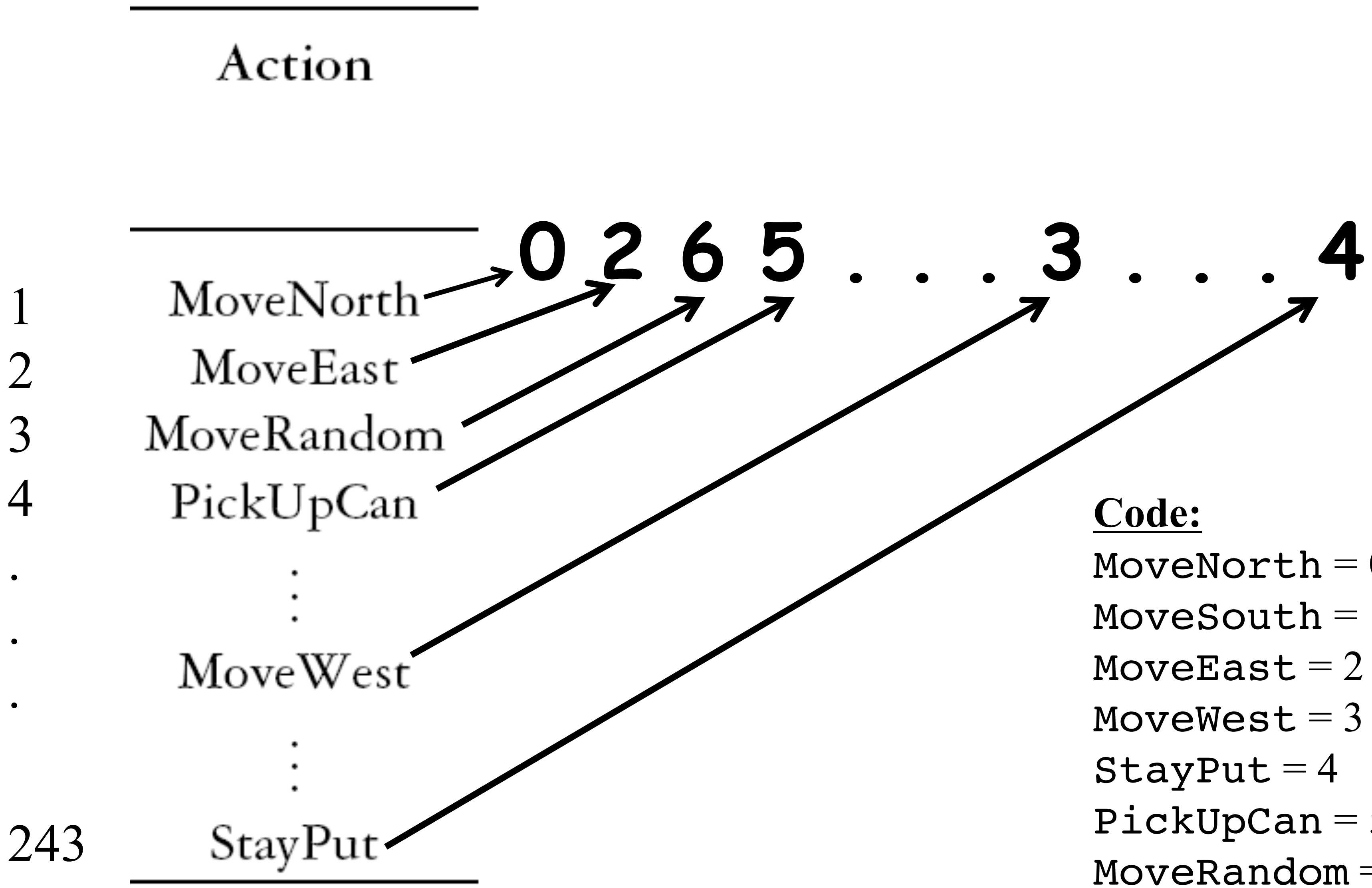
	Action
1	MoveNorth
2	MoveEast
3	MoveRandom
4	PickUpCan
.	⋮
.	MoveWest
.	⋮
243	StayPut



	Action
1	MoveNorth
2	MoveEast
3	MoveRandom
4	PickUpCan
.	⋮
.	MoveWest
.	⋮
243	StayPut

Code:

MoveNorth = 0
MoveSouth = 1
MoveEast = 2
MoveWest = 3
StayPut = 4
PickUpCan = 5
MoveRandom = 6



	Action
1	MoveNorth
2	MoveEast
3	MoveRandom
4	PickUpCan
.	⋮
.	MoveWest
.	⋮
243	StayPut

0 2 6 5 . . . 3 . . . 4

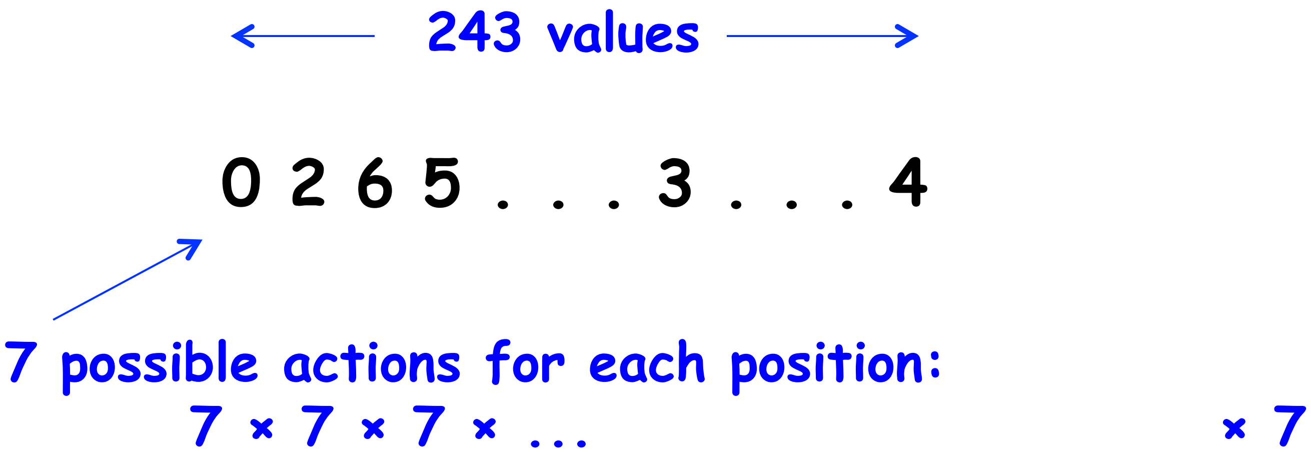
Code:

`MoveNorth` = 0
`MoveSouth` = 1
`MoveEast` = 2
`MoveWest` = 3
`StayPut` = 4
`PickUpCan` = 5
`MoveRandom` = 6

Question: How many possible strategies are there in our representation?

0 2 6 5 . . . 3 . . . 4

Question: How many possible strategies are there in our representation?



243 values

0 2 6 5 . . . 3 . . . 4

7 possible actions for each position:

7 × 7 × 7 × ... × 7

Goal: Have GA search intelligently in this vast space for a good strategy

Genetic algorithm for evolving strategies

1. Generate 200 random strategies (i.e., programs for controlling Robby)

2. For each strategy, calculate fitness (average reward minus penalties earned on random environments)

3. The strategies pair up and create offspring via crossover with random mutations — the fitter the parents, the more offspring they create.

4. Keep going back to step 2 until a good-enough strategy is found (or for a set number of generations)

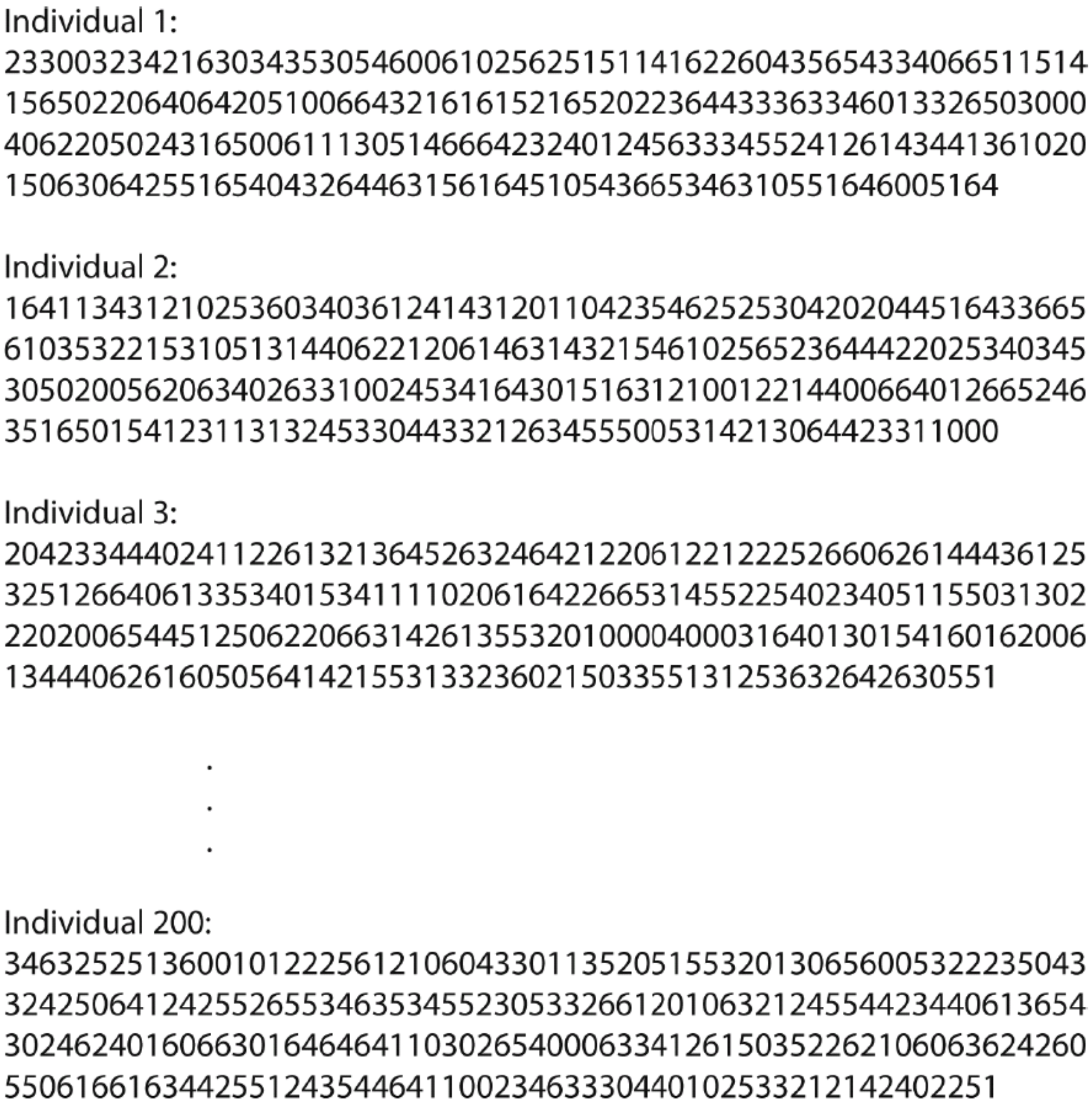
Robby’s fitness function

```
Calculate_Fitness (Robby) {
  Total_Reward = 0 ;
  Average_Reward = 0 `
  For i = 1 to NUM_ENVIRONMENTS {
    generate_random_environment( ); /* .5 probability
                                     * to place can at
                                     * each site */
    For j = 1 to NUM_MOVES_PER_ENVIRONMENT {
      Total_Reward = Total_Reward + perform_action(Robby);
    }
  }

  Fitness = Total_Reward / NUM_ENVIRONMENTS;
  return(Fitness);
}
```

Genetic algorithm for evolving strategies for Robby

1. Generate 200 random strategies (i.e., programs for controlling Robby)

Random Initial Population

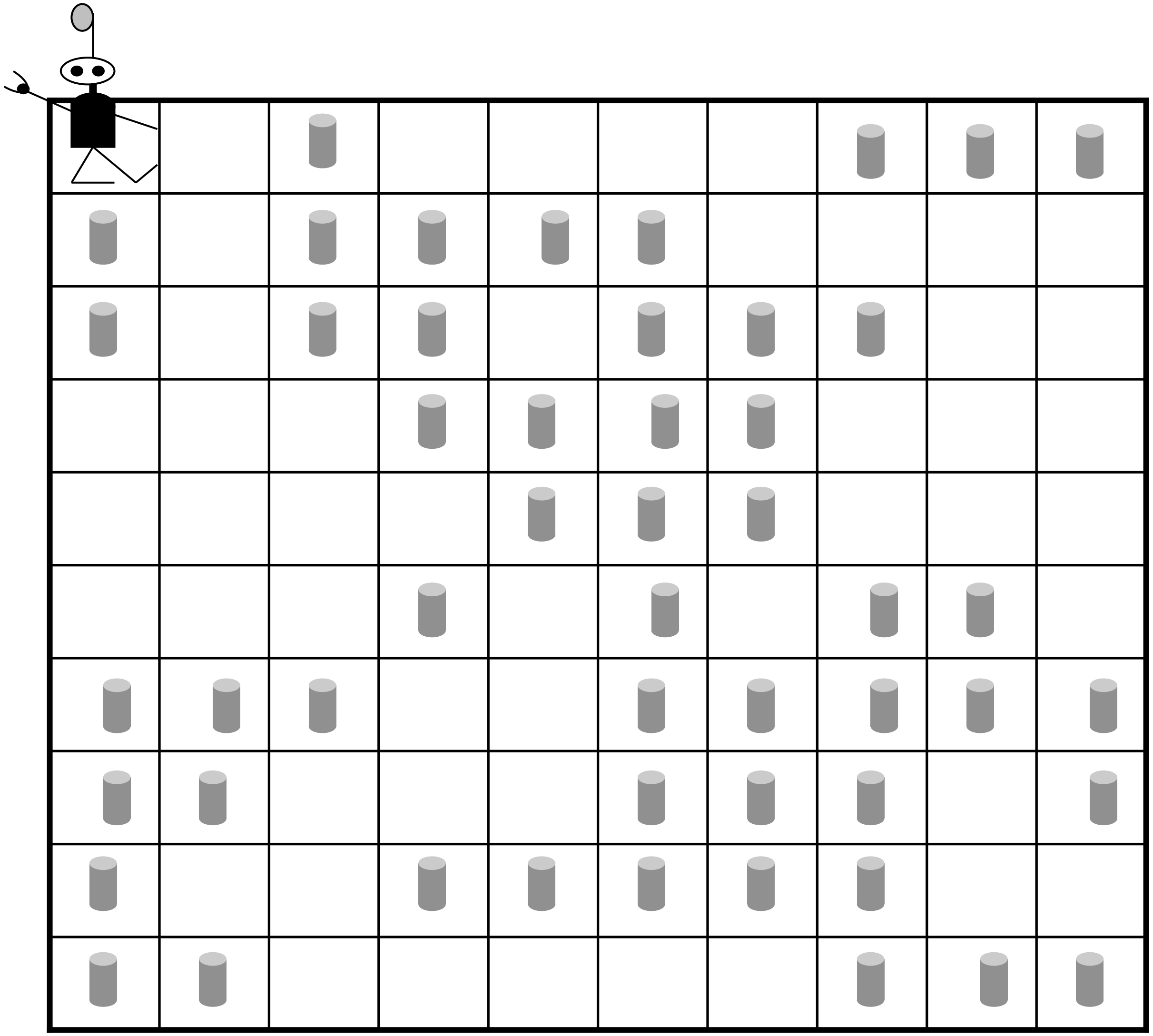


```
Individual 1:
23300323421630343530546006102562515114162260435654334066511514
15650220640642051006643216161521652022364433363346013326503000
40622050243165006111305146664232401245633345524126143441361020
150630642551654043264463156164510543665346310551646005164

Individual 2:
16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002453416430151631210012214400664012665246
351650154123113132453304433212634555005314213064423311000

Individual 3:
20423344402411226132136452632464212206122122252660626144436125
32512664061335340153411110206164226653145522540234051155031302
22020065445125062206631426135532010000400031640130154160162006
134440626160505641421553133236021503355131253632642630551

                    .
                    .
                    .

Individual 200:
34632525136001012225612106043301135205155320130656005322235043
32425064124255265534635345523053326612010632124554423440613654
30246240160663016464641103026540006334126150352262106063624260
550616616344255124354464110023463330440102533212142402251
```

Genetic algorithm for evolving strategies for Robby

1. Generate 200 random strategies (i.e., programs for controlling Robby)

2. For each strategy in the population, calculate fitness (average reward minus penalties earned on random environments)



Fitness =
Average final score from
N moves on each of M
random environments

Genetic algorithm for evolving strategies for Robby

1. Generate 200 random strategies (i.e., programs for controlling Robby)

2. For each strategy in the population, calculate fitness (average reward minus penalties earned on random environments)

3. Strategies are selected according to fitness to become parents. (See code for choice of selection methods.)

Genetic algorithm for evolving strategies for Robby

1. Generate 200 random strategies (i.e., programs for controlling Robby)

2. For each strategy in the population, calculate fitness (average reward minus penalties earned on random environments)

3. Strategies are selected according to fitness to become parents. (See code for choice of selection methods.)

4. The parents pair up and create offspring via crossover with random mutations.

Parent 1:

```
16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002453416430151631210012214400664012665246
351650154123113132453304433212634555005314213064423311000
```

Parent 2:

```
20423344402411226132136452632464212206122122252660626144436125
32512664061335340153411110206164226653145522540234051155031302
22020065445125062206631426135532010000400031640130154160162006
134440626160505641421553133236021503355131253632642630551
```

Parent 1:

164113431210253603403612414312011042354625253042020445164336656
10353221531051314406221206146314321546102565236444220253403453
05020056206340263310024534164301516312100122144006640126652463
51650154123113132453304433212634555005314213064423311000

Parent 1:

```
16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002453416430151631210012214400664012665246
351650154123113132453304433212634555005314213064423311000
```

Parent 2:

```
20423344402411226132136452632464212206122122252660626144436125
32512664061335340153411110206164226653145522540234051155031302
22020065445125062206631426135532010000400031640130154160162006
134440626160505641421553133236021503355131253632642630551
```

Child:

```
16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002456135532010000400031640130154160162006
134440626160505641421553133236021503355131253632642630551
```

Parent 1:

16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002453416430151631210012214400664012665246
351650154123113132453304433212634555005314213064423311000

Parent 2:

20423344402411226132136452632464212206122122252660626144436125
32512664061335340153411110206164226653145522540234051155031302
22020065445125062206631426135532010000400031640130154160162006
1344406261605056414215531332360215033551312536326426305551

Mutate to “0”

Child:

16411343121025360340361241431201104235462525304202044516433665
61035322153105131440622120614631432154610256523644422025340345
30502005620634026331002456135532010000400031640130154160162006
1344406261605056414215531332360215033551312536326426305551

Mutate to “4”

Genetic algorithm for evolving strategies for Robby

1. Generate 200 random strategies (i.e., programs for controlling Robby)

2. For each strategy in the population, calculate fitness (average reward minus penalties earned on random environments)

3. Strategies are selected according to fitness to become parents. (See code for choice of selection methods.)

4. The parents pair up and create offspring via crossover with random mutations.

5. The offspring are placed in the new population and the old population dies.

6. Keep going back to step 2 until a good-enough strategy is found!

My hand-designed strategy:

“If there is a can in the current site, pick it up.”

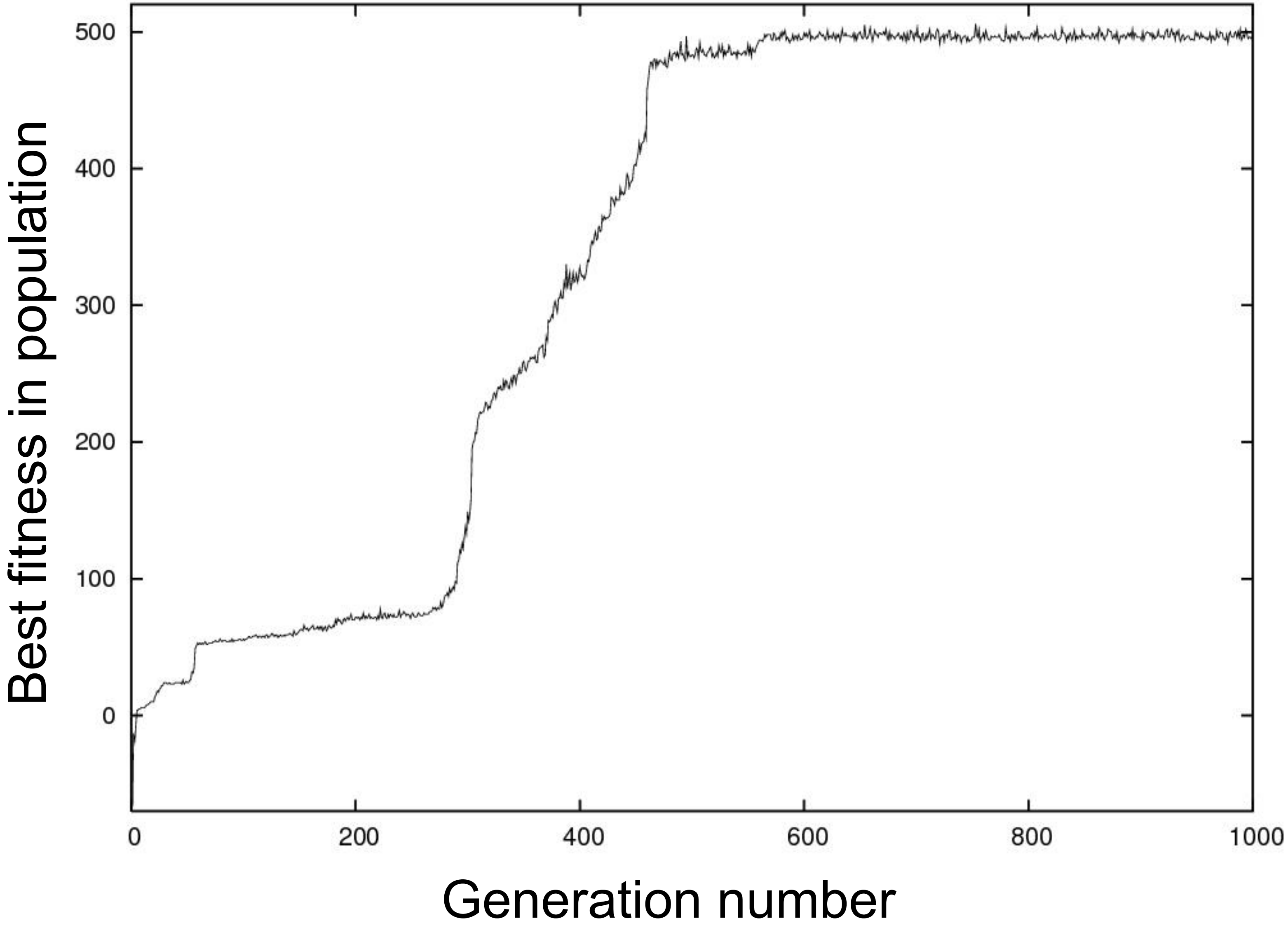
“Otherwise, if there is a can in one of the adjacent sites, move to that site.”

“Otherwise, choose a random direction to move in, avoiding walls.”

Average fitness of this strategy:
346
(out of max possible ≈ 500)

Average fitness of GA evolved strategy: **486**
(out of max possible ≈ 500)

One Run of the Genetic Algorithm



GA Demo

http://www.cs.pdx.edu/~mm/RobbyTheRobot