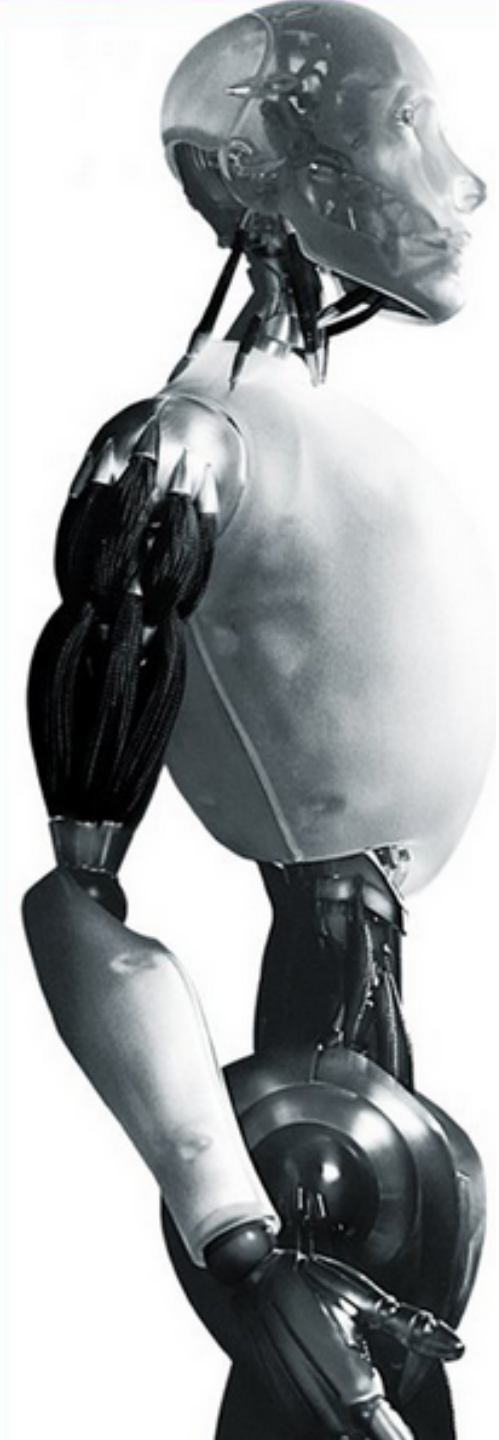
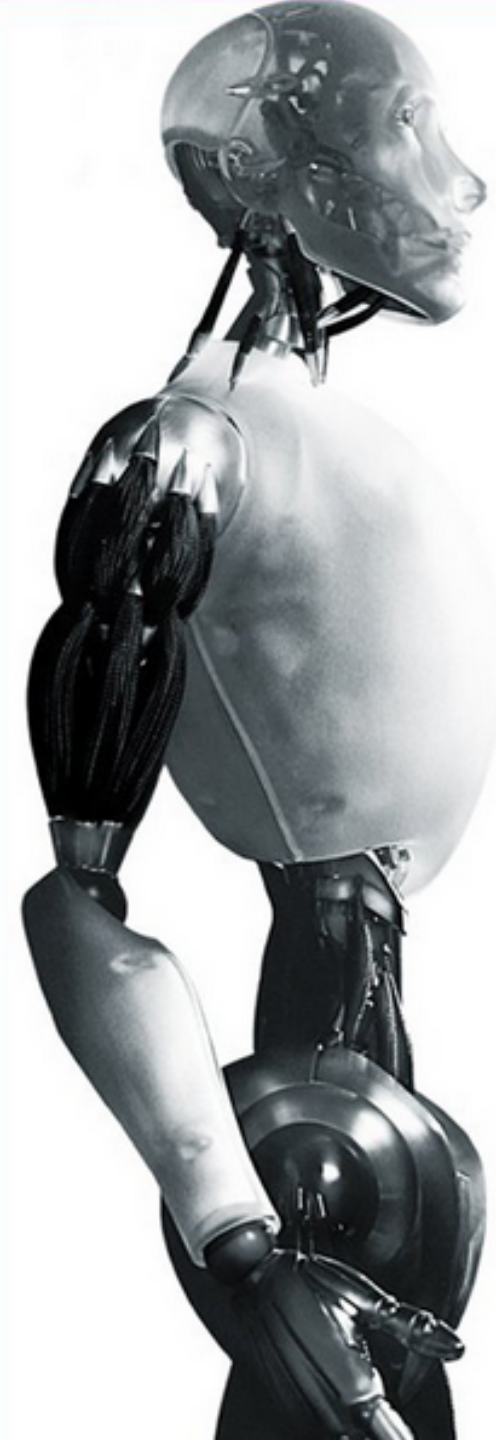


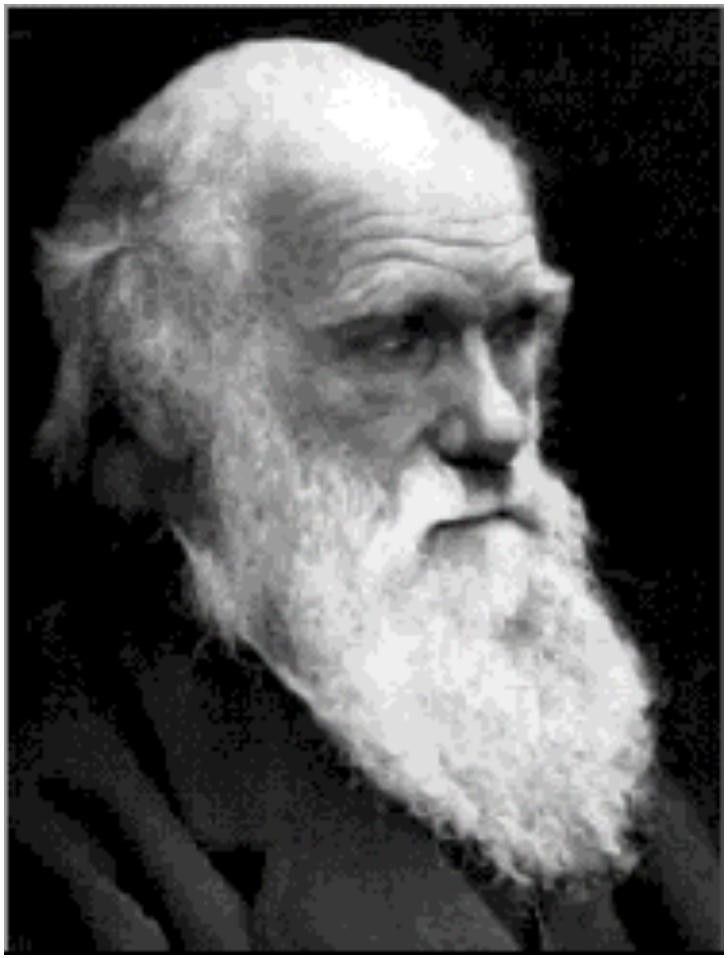
Artificial Intelligence

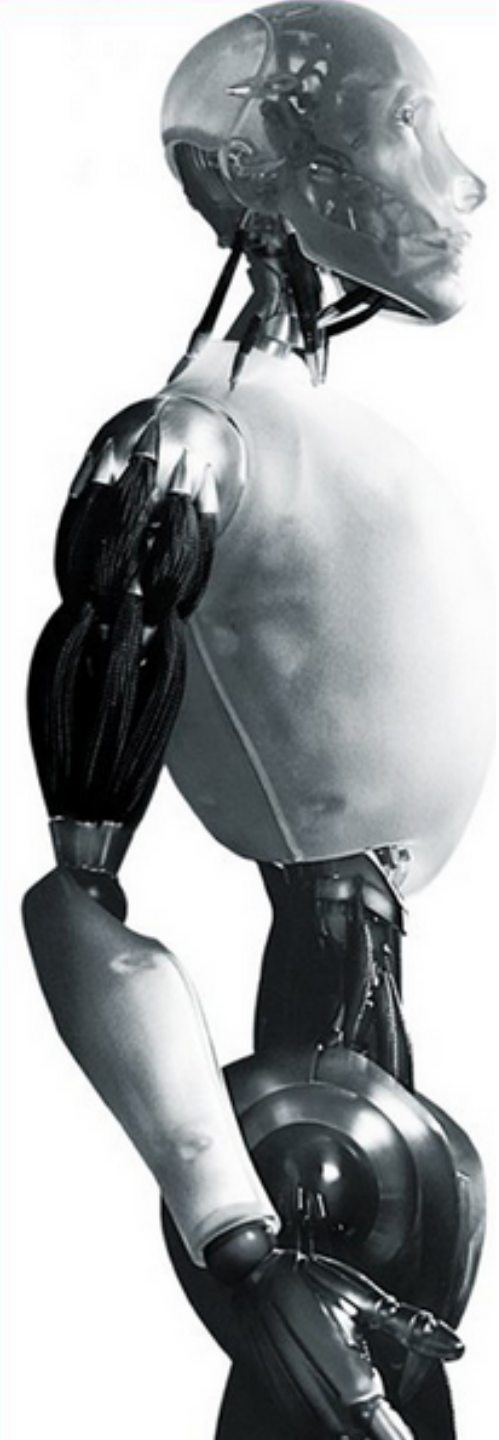
Genetic Algorithms

Prof Alexiei Dingli



Charles Darwin

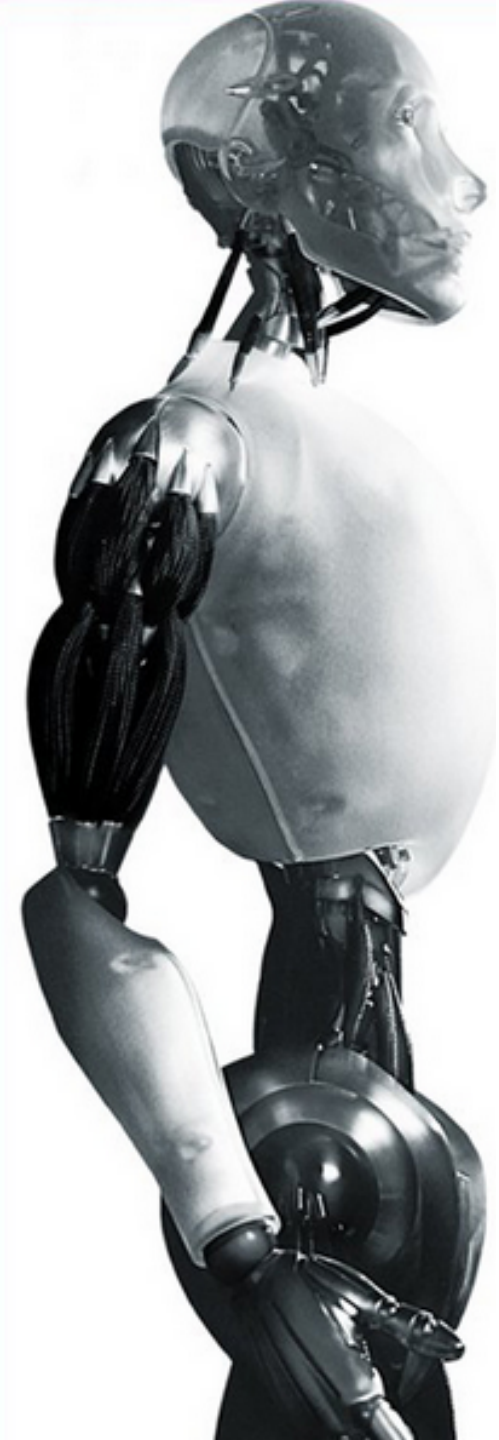




"Genetic Algorithms are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you might not otherwise find in a lifetime."

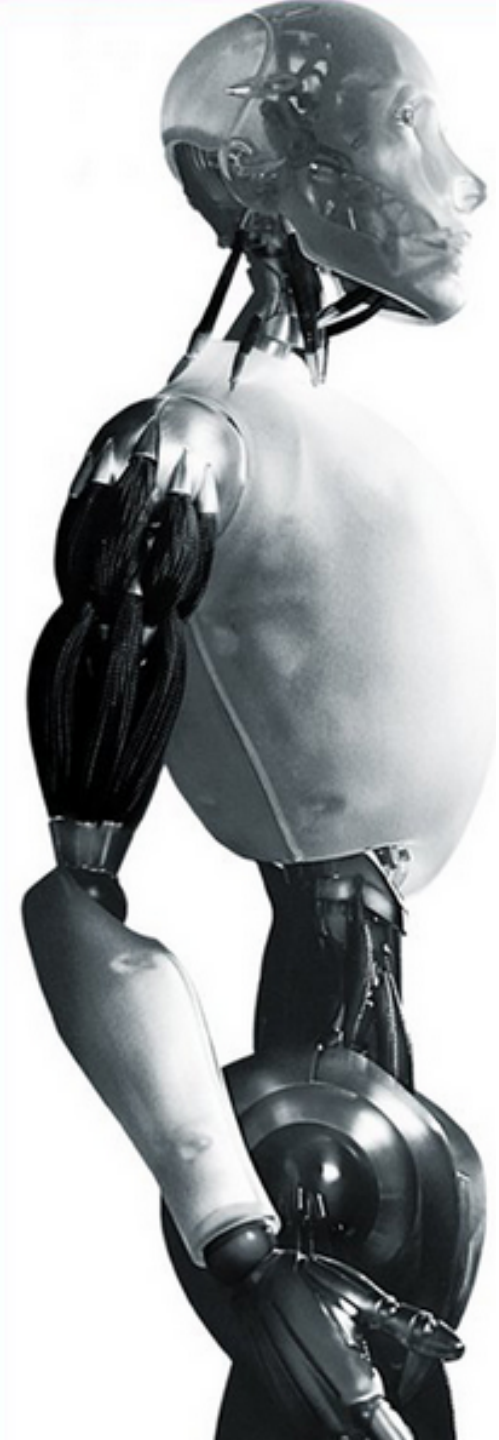
- Salvatore Mangano

Computer Design, May 1995



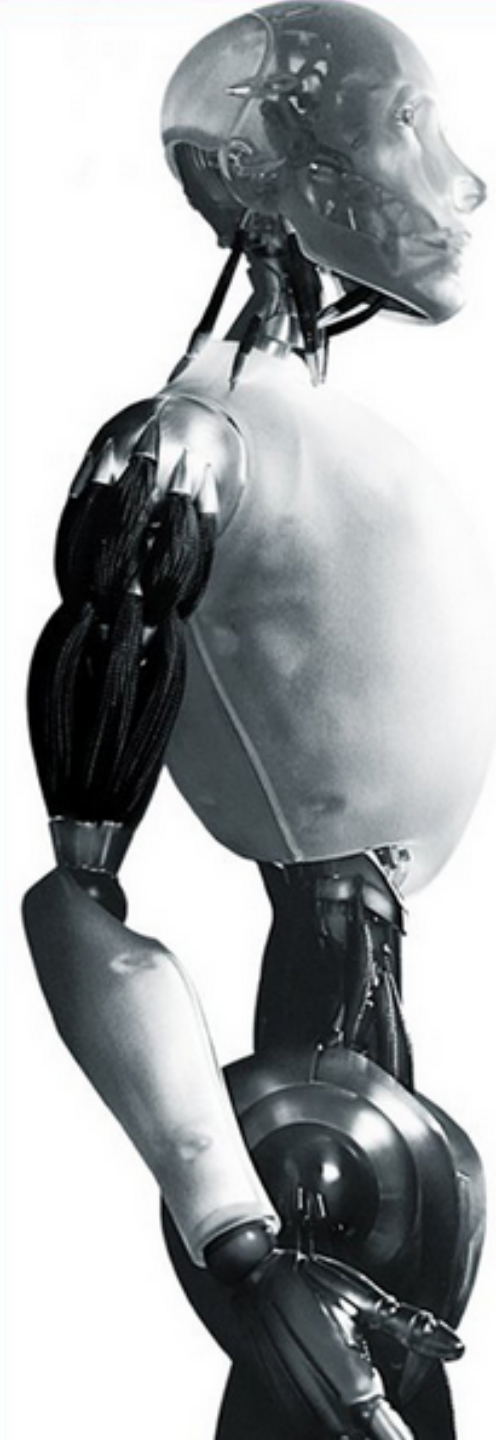
The Genetic Algorithm

- Directed search algorithms based on the mechanics of biological evolution
- Developed by John Holland, University of Michigan (1970's)
  - To understand the adaptive processes of natural systems
  - To design artificial systems software that retains the robustness of natural systems



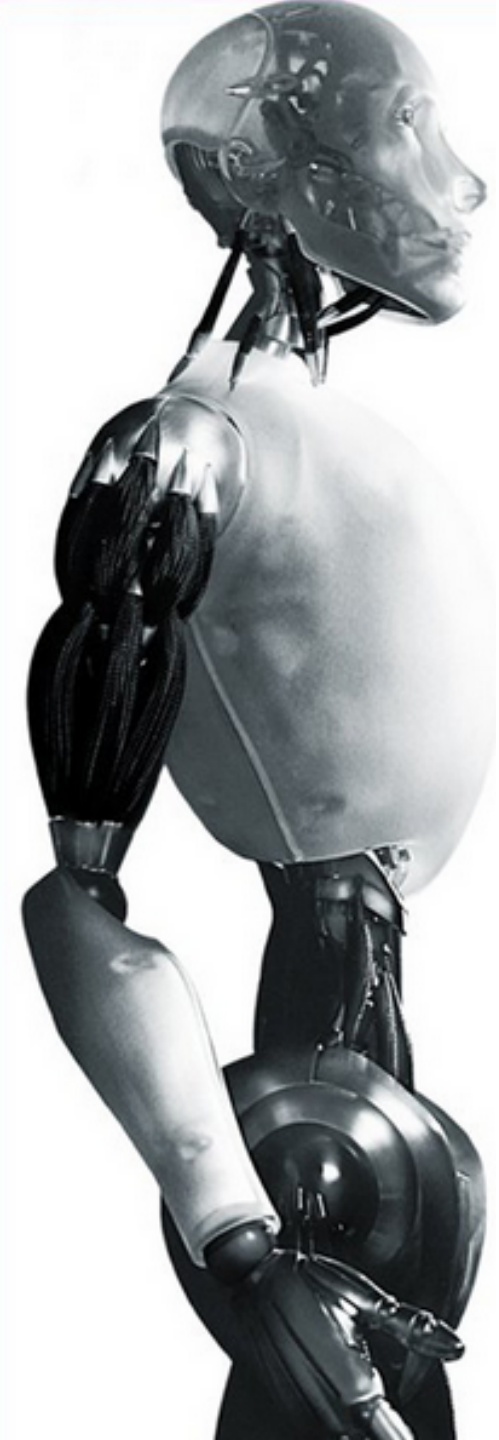
The Genetic Algorithm

- Provide efficient, effective techniques for optimization and machine learning applications
- Widely-used today in business, scientific and engineering circles



Evolutionary Computing

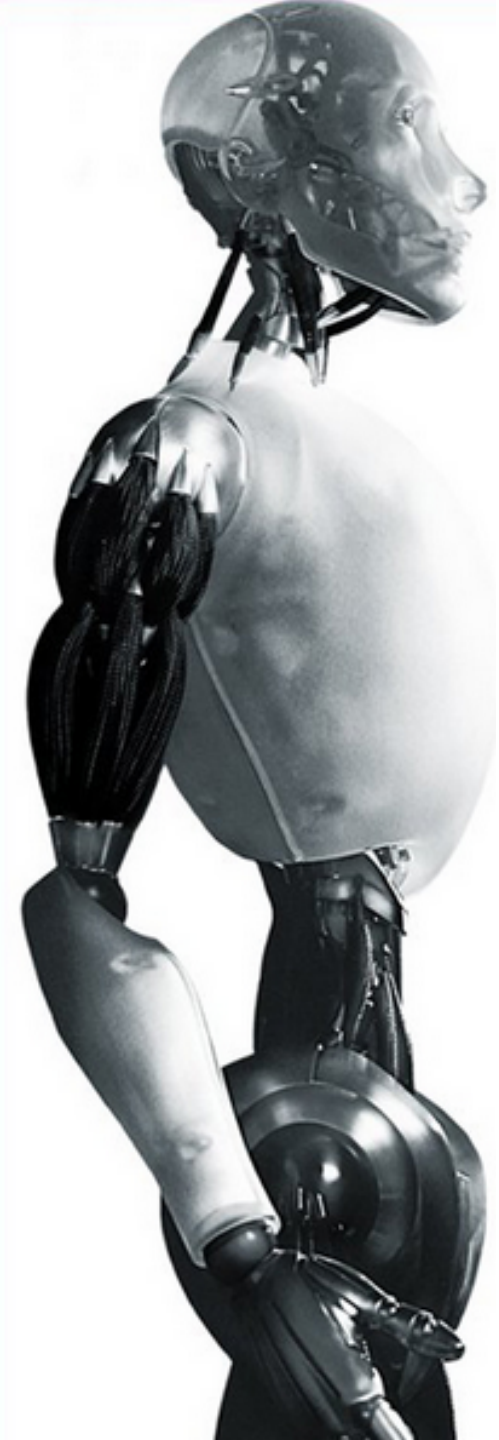
- *A collection of computational methods inspired by biological evolution:*
- A population of candidate solutions evolves over time, with the fittest at each generation contributing the most offspring to the next generation
- Offspring are produced via crossover between parents, along with random mutations and other "genetic" operations.



Components of a GA

A problem to solve, and ...

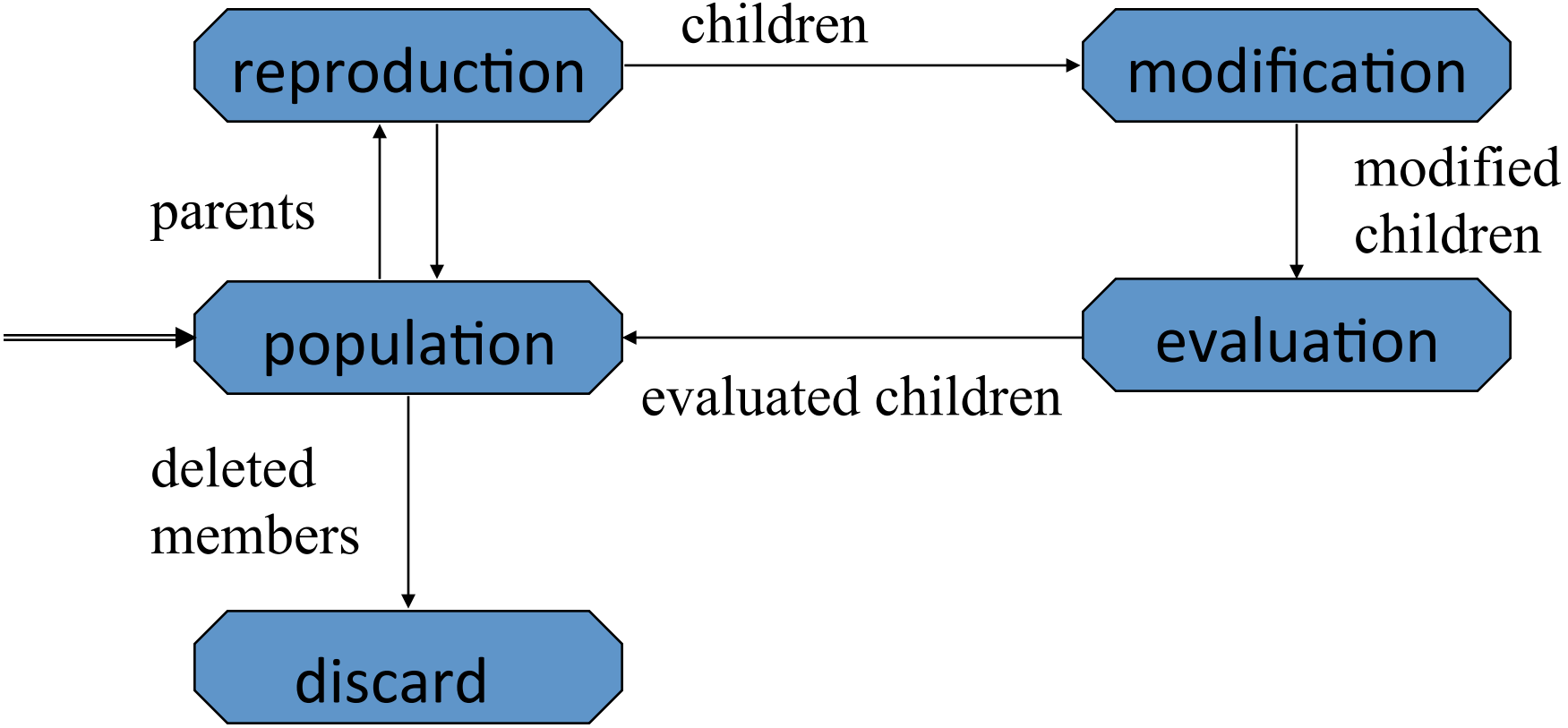
- Encoding technique (*gene, chromosome*)
- Initialization procedure *(creation)*
- Evaluation/fitness function *(environment)*
- Selection of parents *(reproduction)*
- Genetic operators *(mutation, recombination)*
- Parameter settings *(practice and art)*

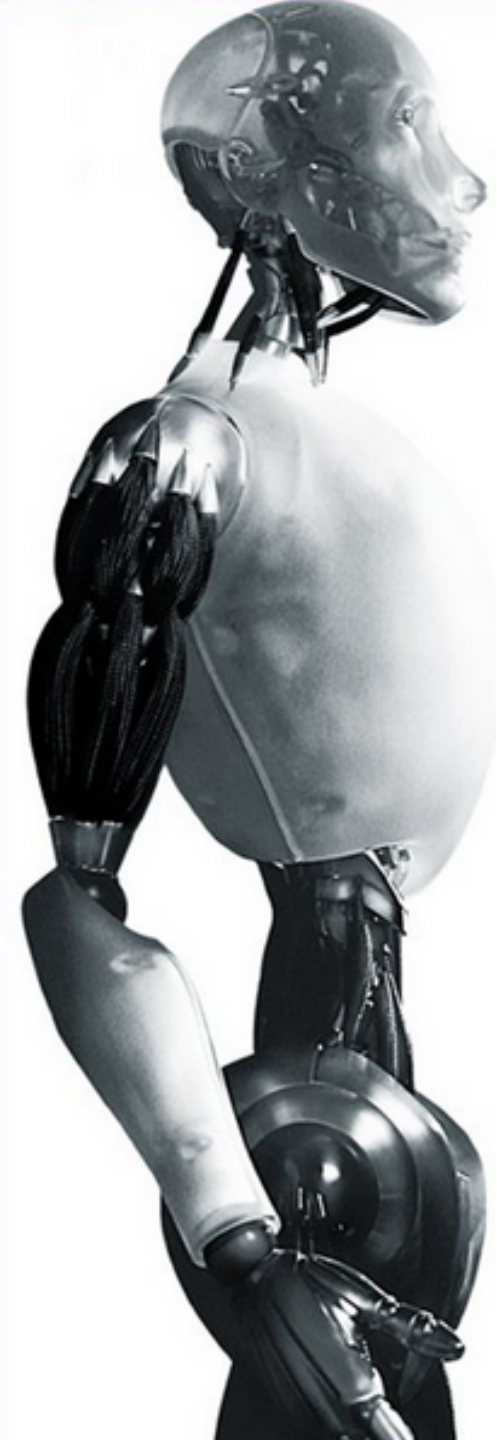


Simple Genetic Algorithm

```
{
    initialize population;
    evaluate population;
    while TerminationCriteriaNotSatisfied
    {
        select parents for reproduction;
        perform recombination and
          mutation;
        evaluate population;
    }
}
```

The GA Cycle of Reproduction

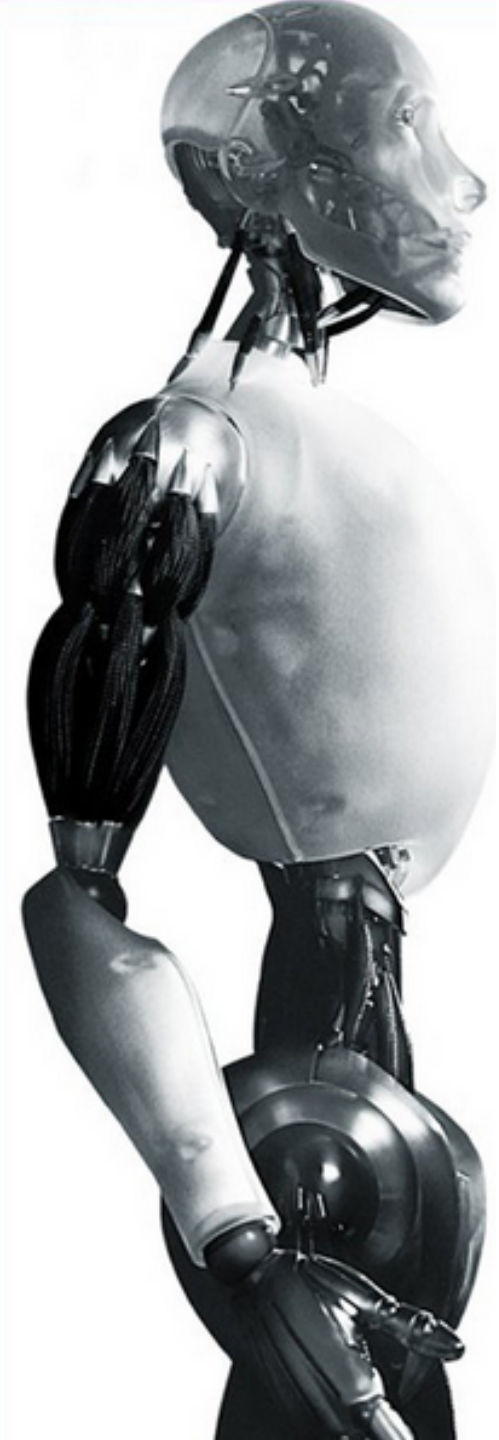




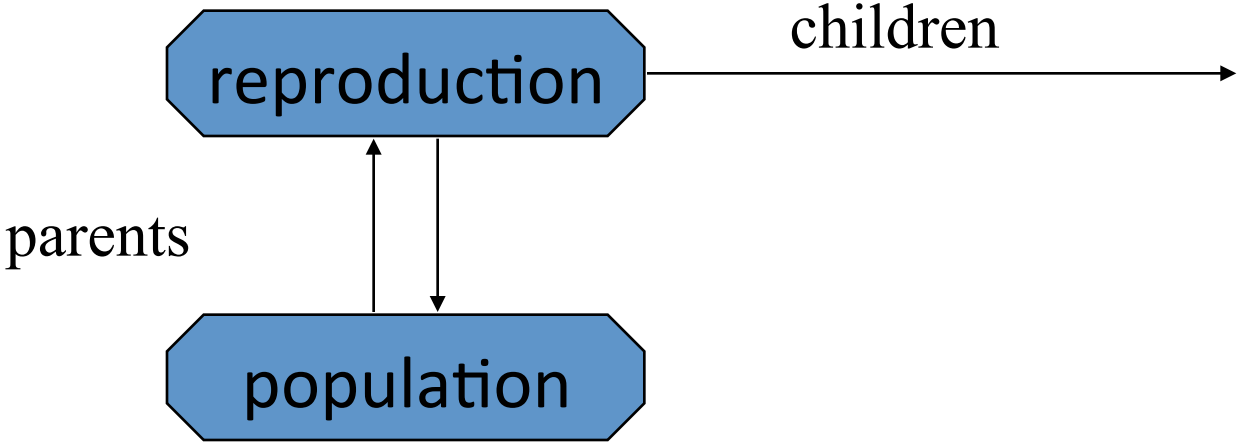
Population

Chromosomes could be:

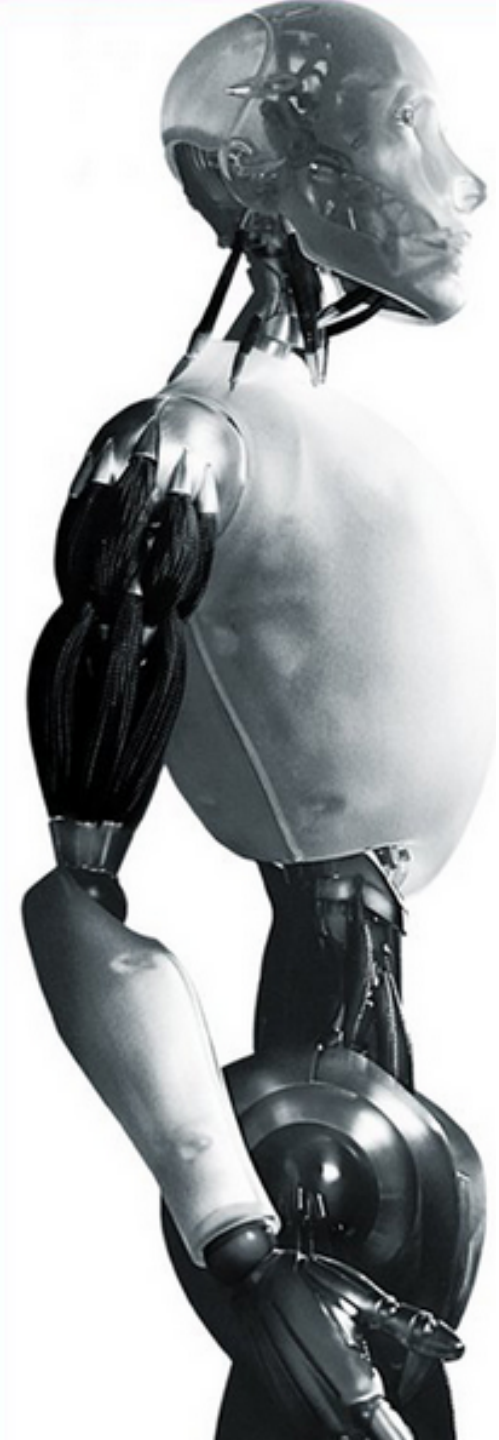
- Bit strings (0101 ... 1100)
- Real numbers (43.2 -33.1 ... 0.0 89.2)
- Permutations of element (E11 E3 E7 ... E1 E15)
- Lists of rules (R1 R2 R3 ... R22 R23)
- Program elements (genetic programming)
- ... any data structure ...



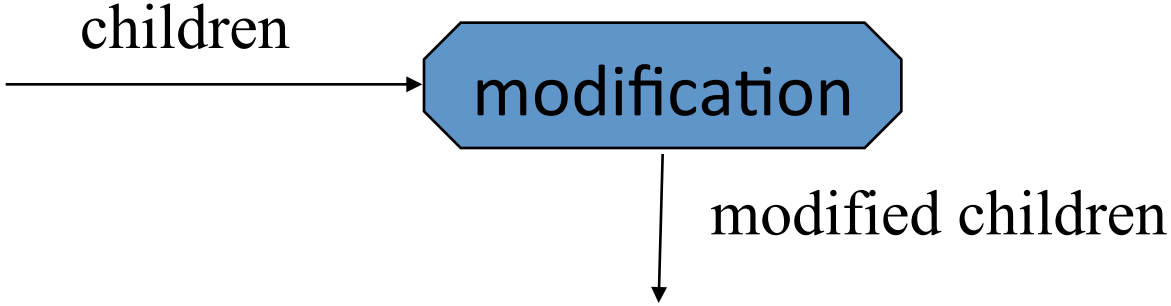
Reproduction



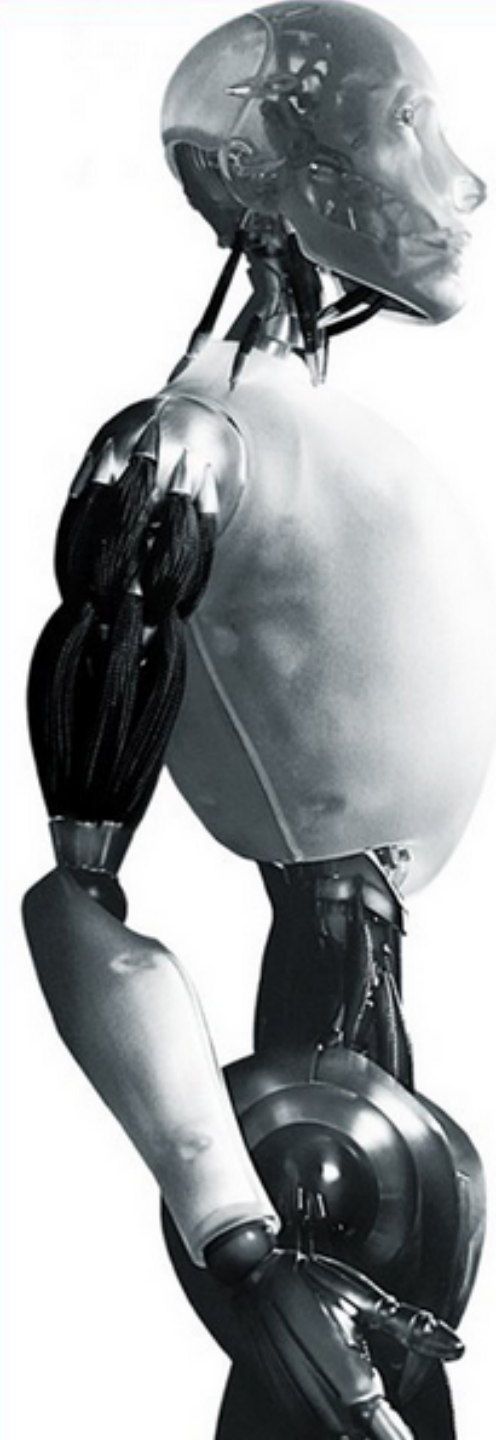
Parents are selected at random with selection chances biased in relation to chromosome evaluations.



Chromosome Modification



- Modifications are randomly triggered
- Operator types are:
  - Mutation
  - Crossover (recombination)



Mutation: Local Modification

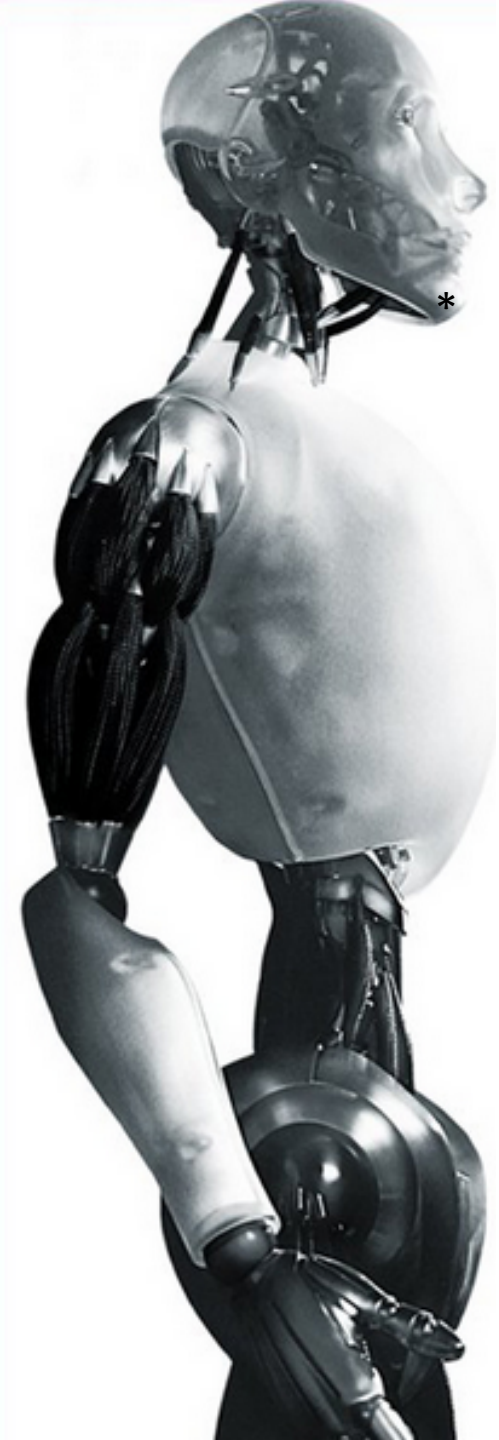
Before: (1 0 1 **1** 0 1 1 0)

After: (1 0 1 **0** 0 1 1 0)

Before: (1.38 **-69.4** 326.44 0.1)

After: (1.38 **-67.5** 326.44 0.1)

- Causes movement in the search space (local or global)
- Restores lost information to the population



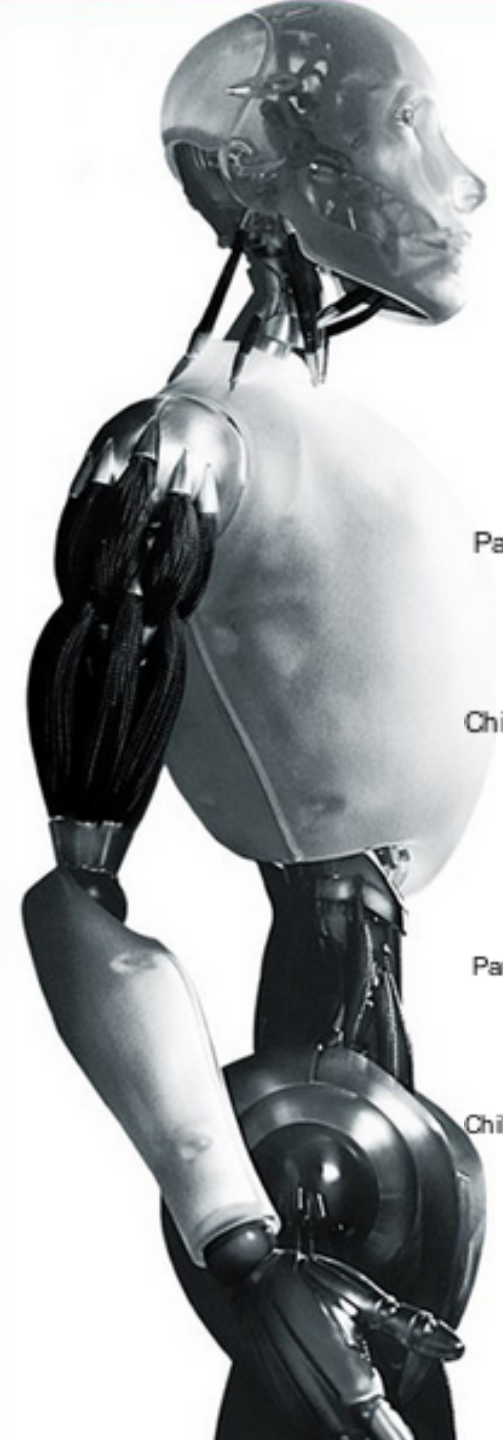
Crossover: Recombination

P1 (0 1 1 0 1 0 0 0) (0 1 1 0 1 0 1 0) C1

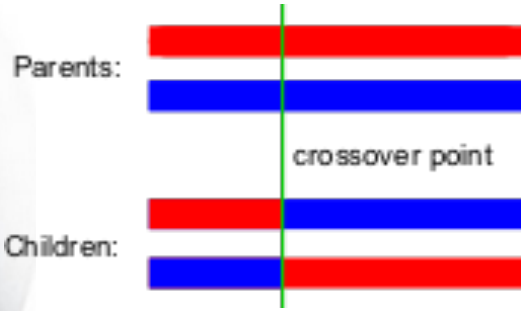
P2 (1 1 1 1 1 0 1 0) (1 1 1 1 1 0 0 0) C2

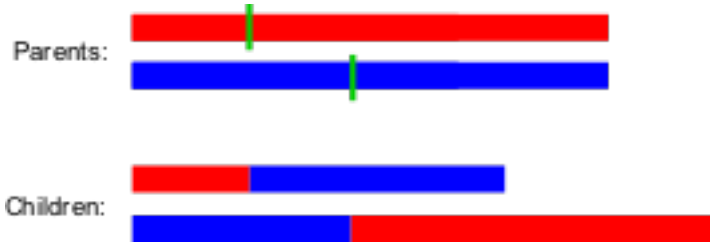
Crossover is a critical feature of genetic algorithms:

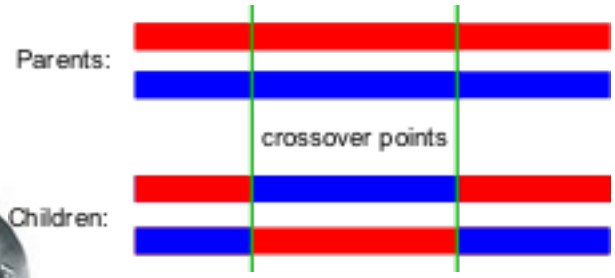
- It greatly accelerates search early in evolution of a population
- It leads to effective combination of schemata (subsolutions on different chromosomes)

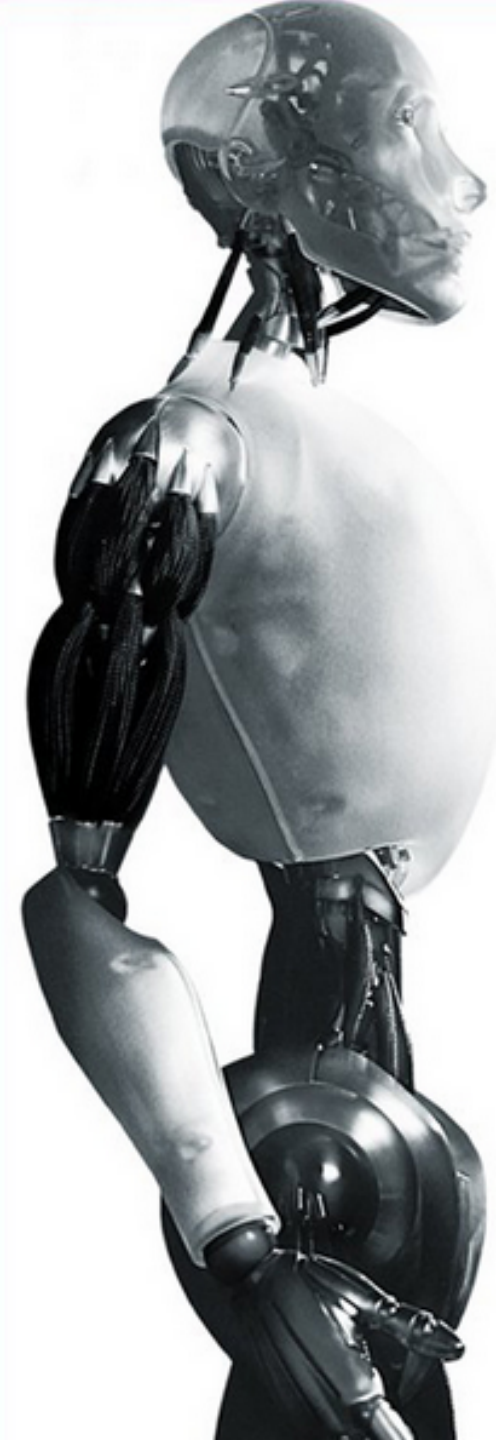


More Crossover

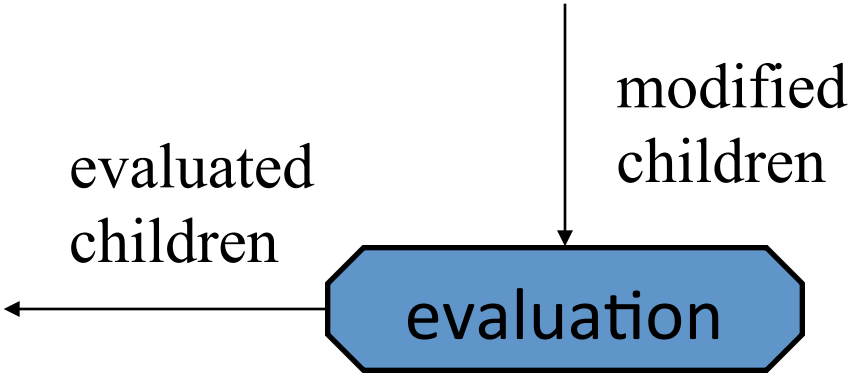




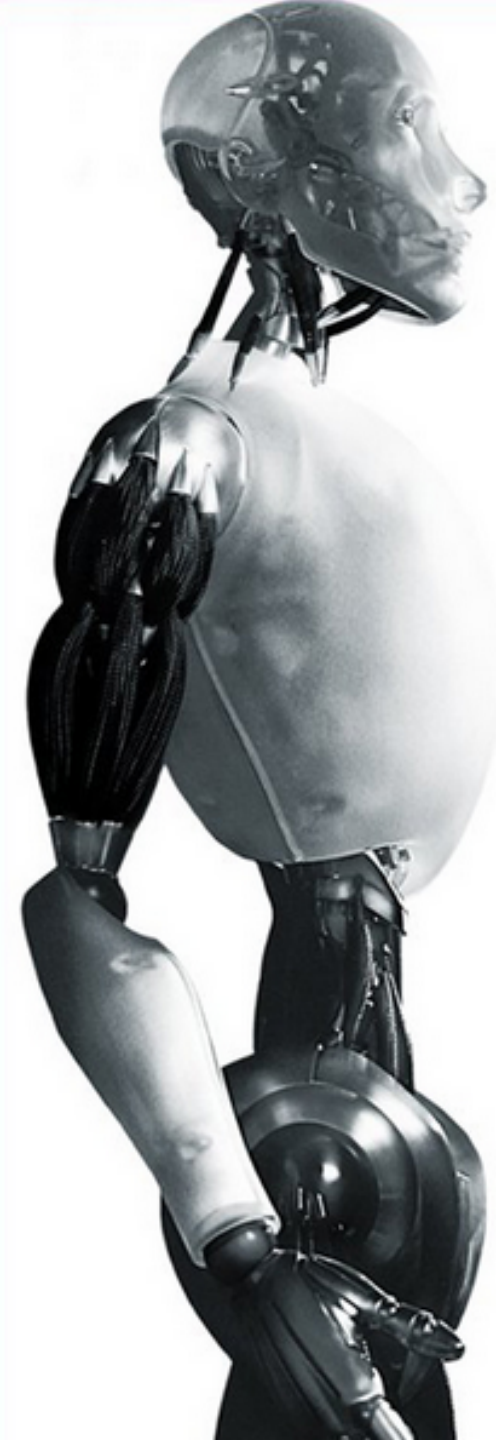




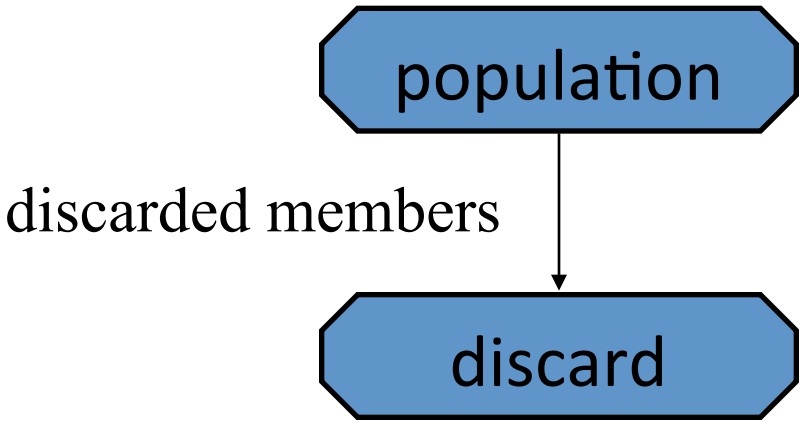
Evaluation



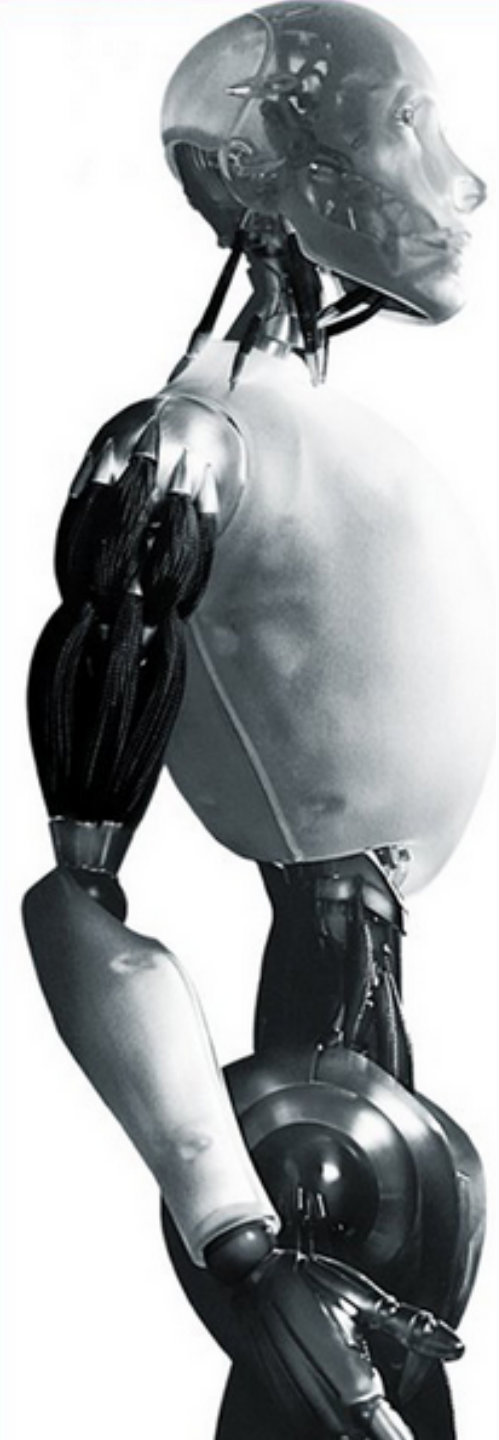
- The evaluator decodes a chromosome and assigns it a fitness measure
- The evaluator is the only link between a classical GA and the problem it is solving



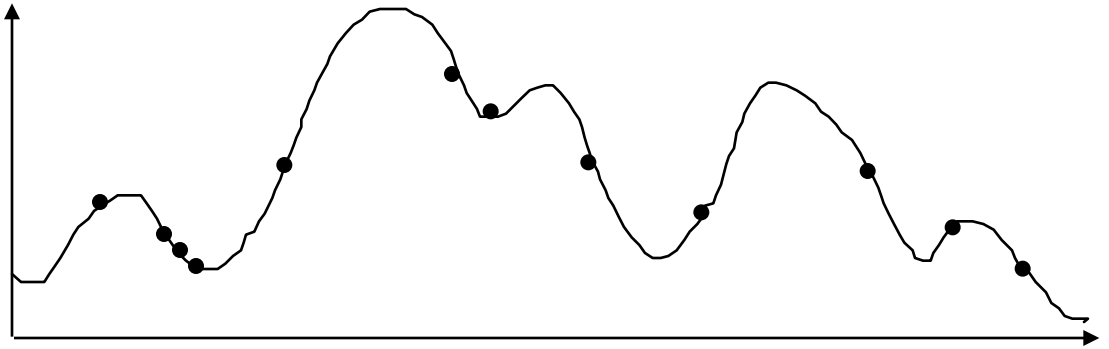
Deletion



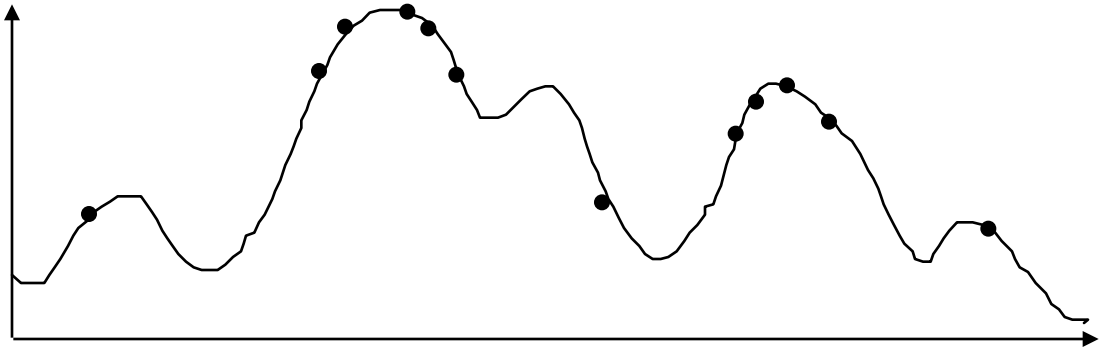
- *Generational* GA:
  entire populations replaced with each iteration
- *Steady-state* GA:
  a few members replaced each generation



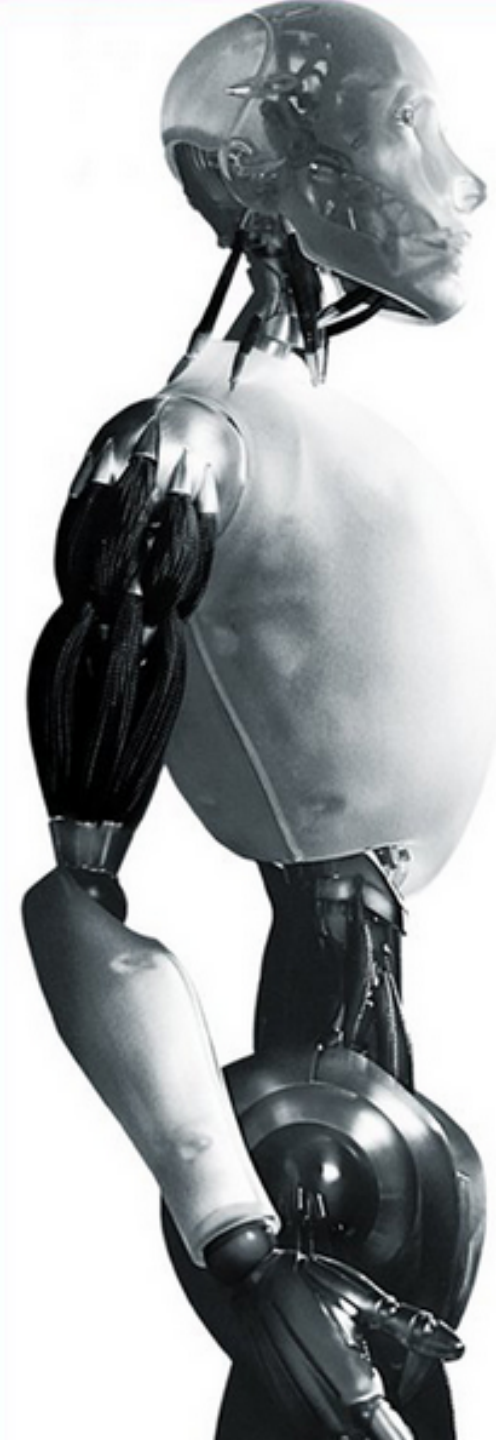
An Abstract Example



Distribution of Individuals in Generation 0



Distribution of Individuals in Generation N

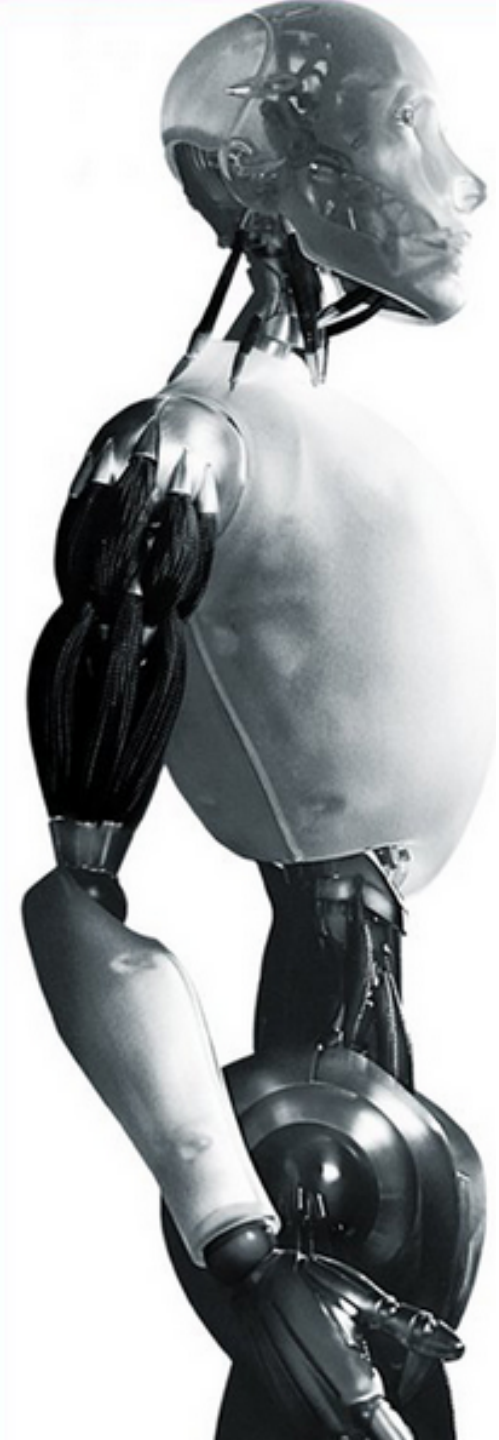


A Simple Example

The Traveling Salesman Problem:

Find a tour of a given set of cities so that

- each city is visited only once
- the total distance traveled is minimized



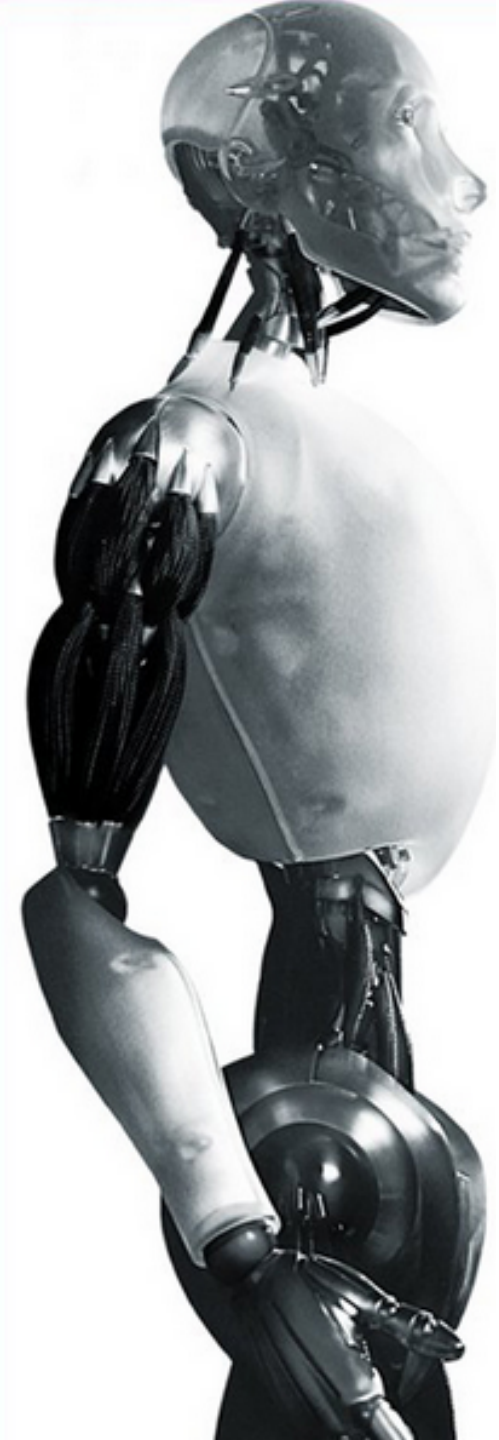
Representation

Representation is an ordered list of city numbers known as an *order-based* GA.

1) London 3) Valletta 5) Beijing 7) Tokyo
2) Venice 4) Singapore 6) Rome 8) Prague

CityList1 (3 5 7 2 1 6 4 8)

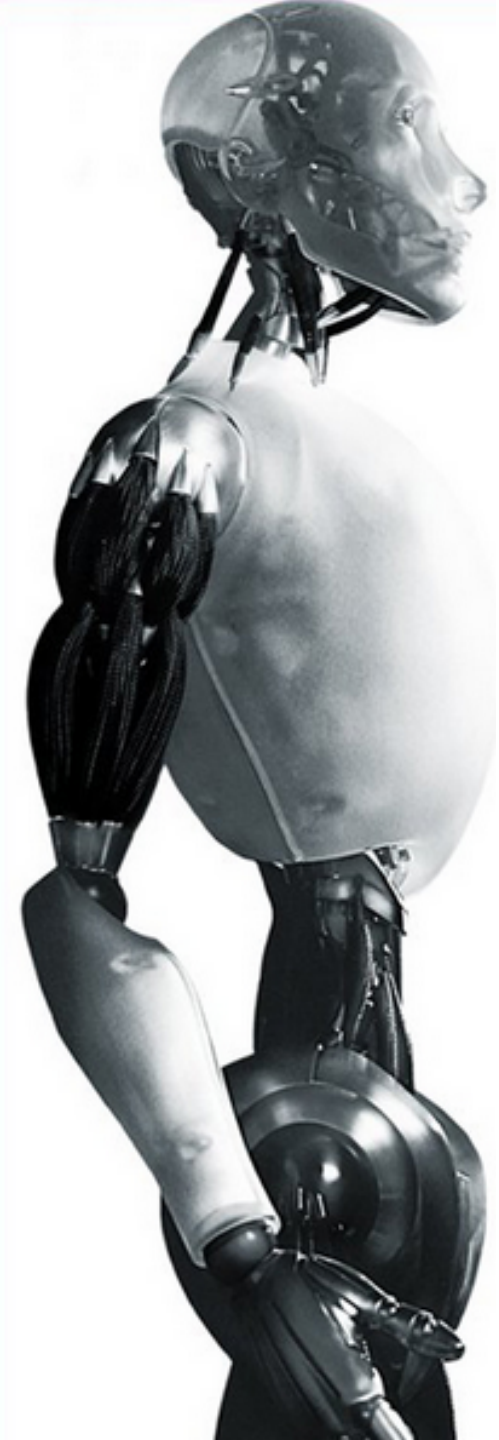
CityList2 (2 5 7 6 8 1 3 4)



Crossover

Crossover combines inversion and recombination:

			*			*		
Parent1	(3	5	7	2	1	6	4	8)
Parent2	(2	5	7	6	8	1	3	4)
Child	(8	5	7	2	1	6	3	4)
Child	(3	5	7	6	8	1	4	**2**)

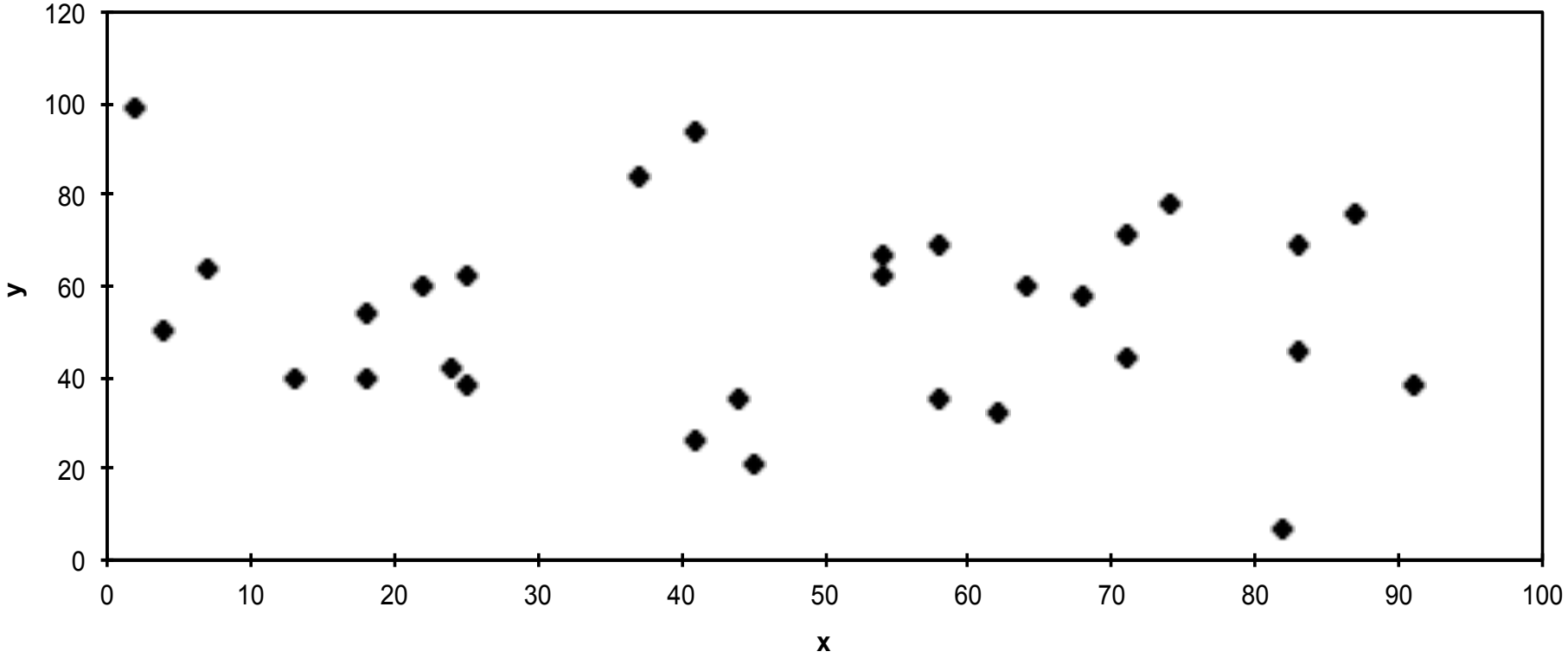


Mutation

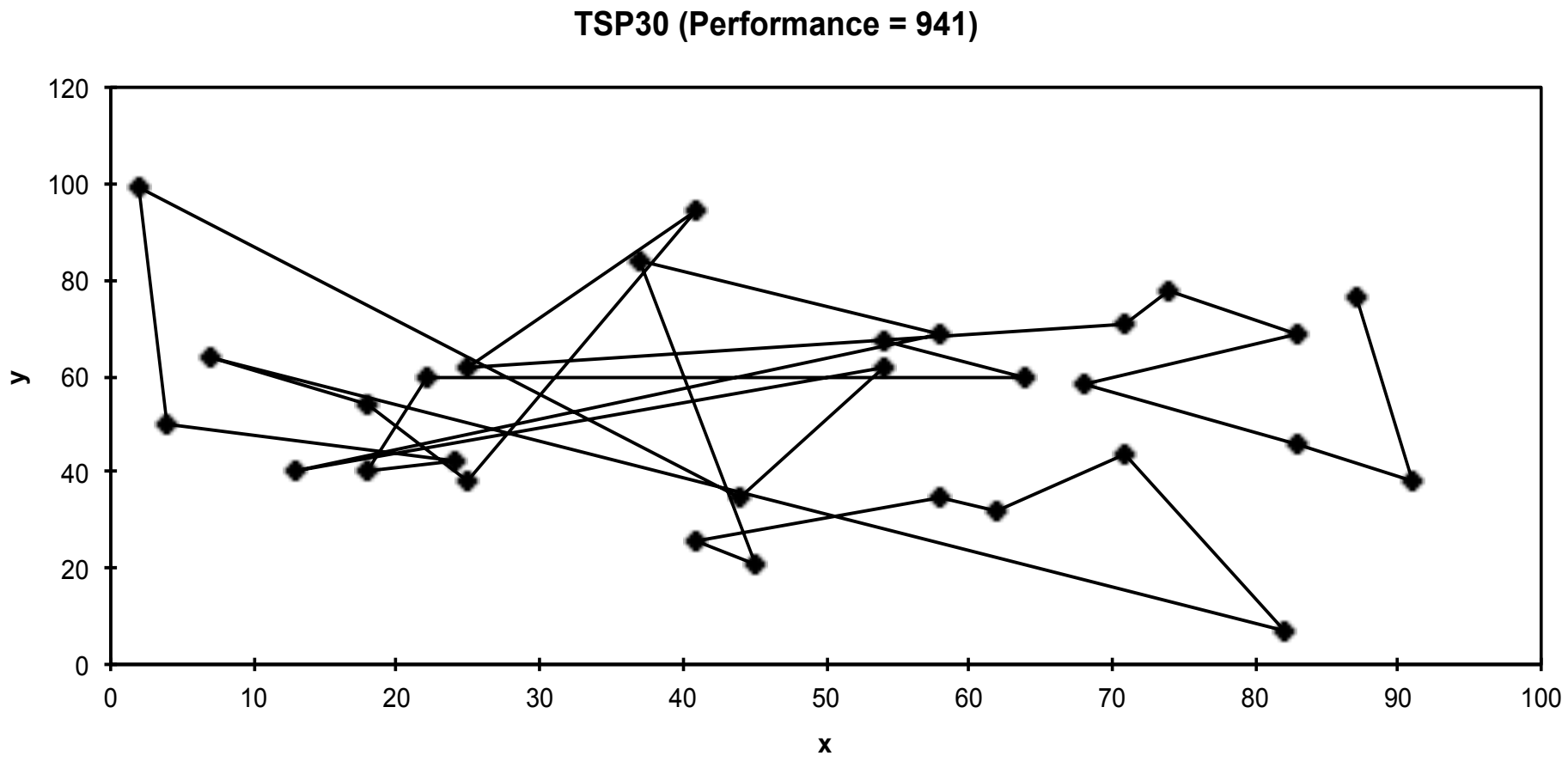
Mutation involves reordering of the list:

			*			*		
Before:	(5	8	7	2	1	6	3	4)
			↔					
After:	(5	8	6	2	1	7	3	4)

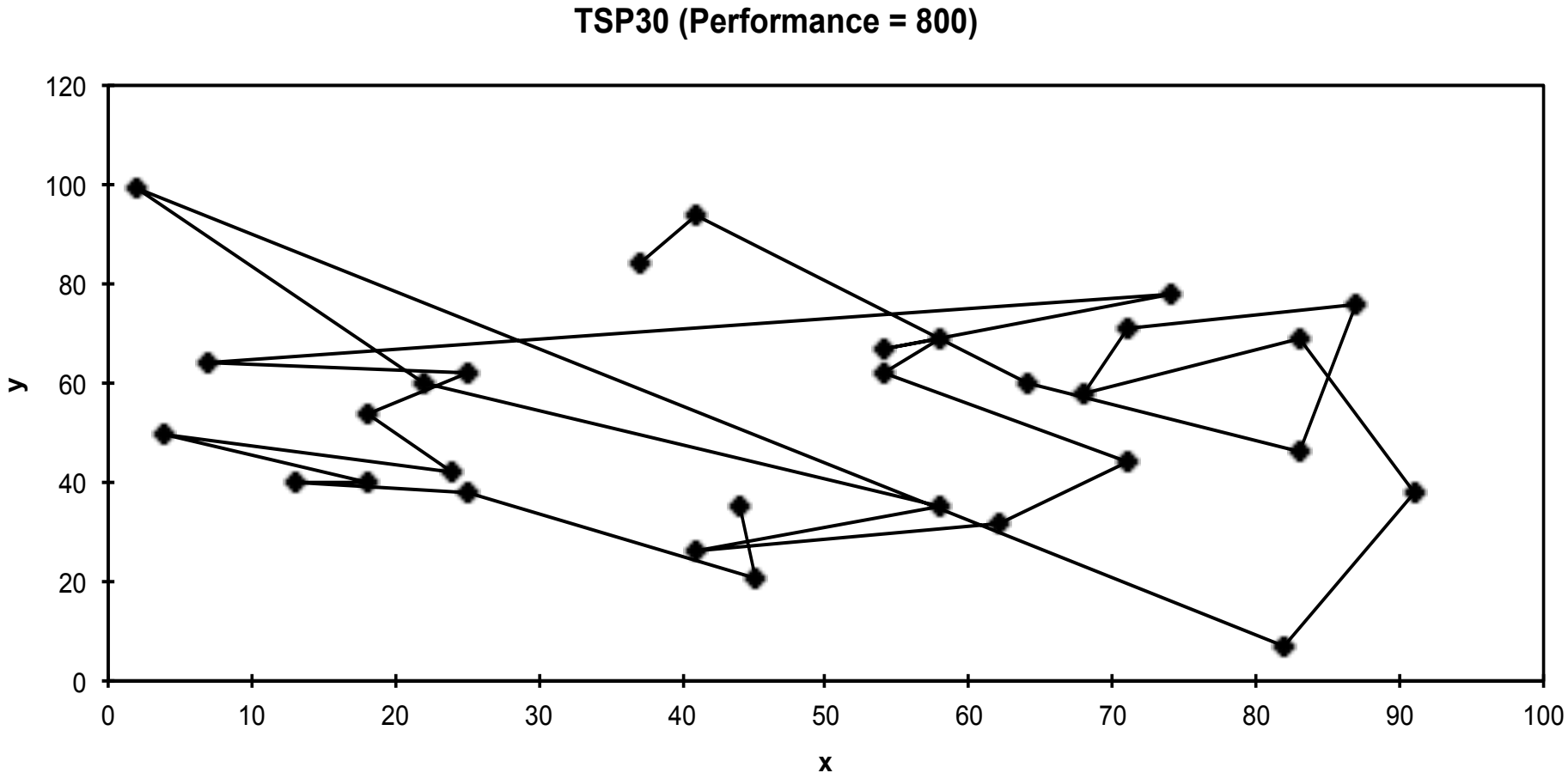
TSP Example: 30 Cities



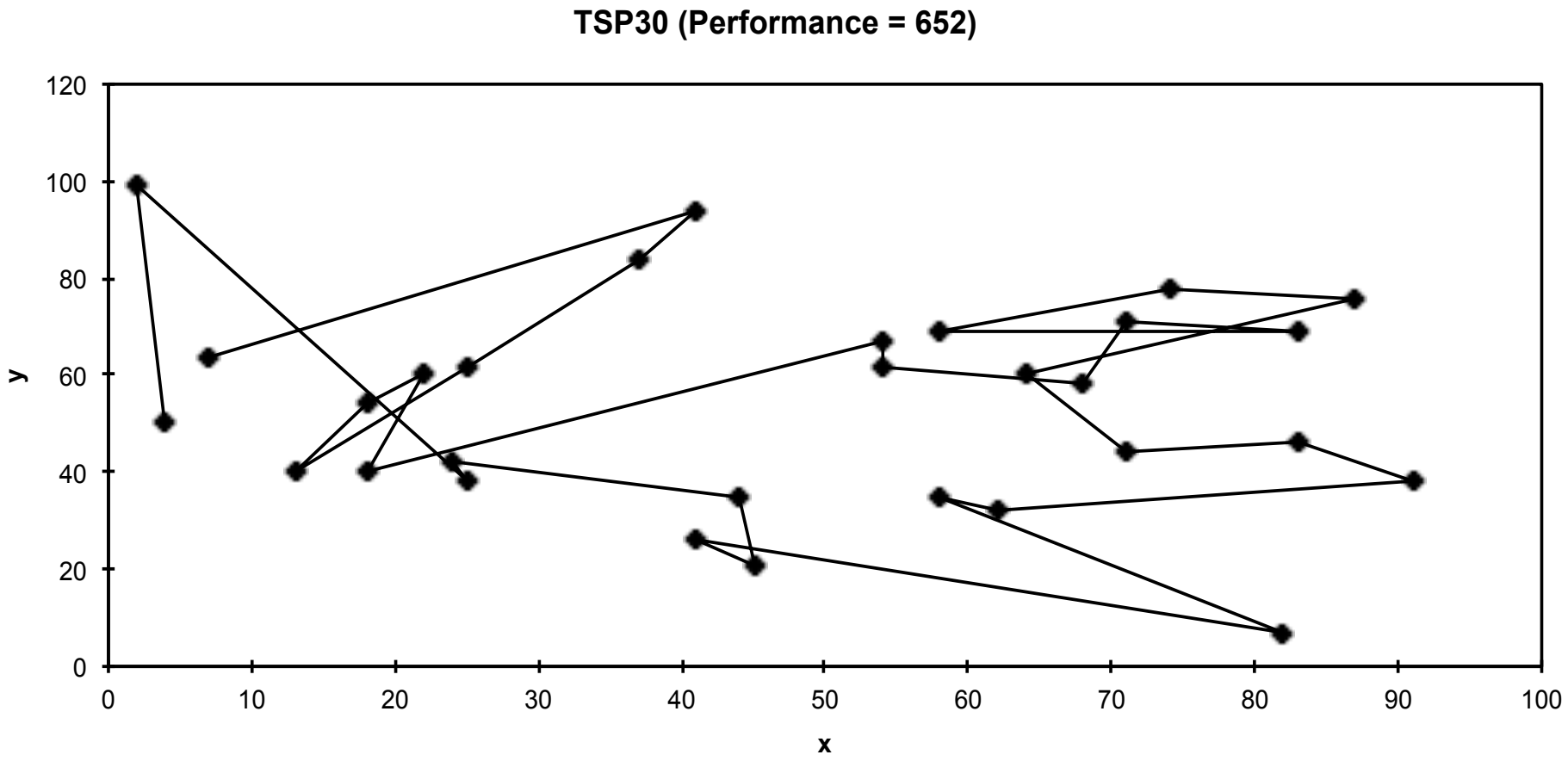
Solution ¡ (Distance = 941)



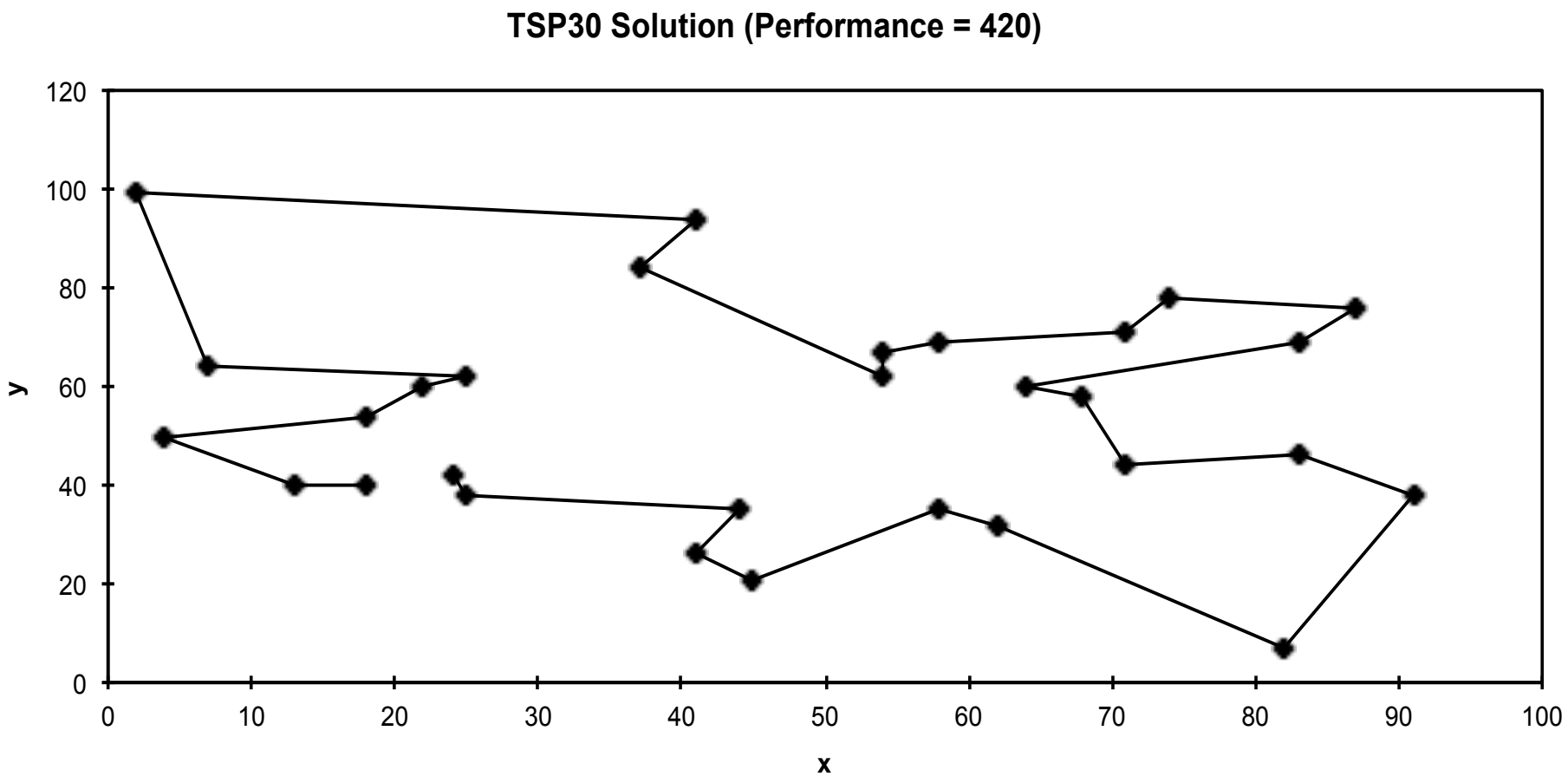
Solution $_j$(Distance = 800)



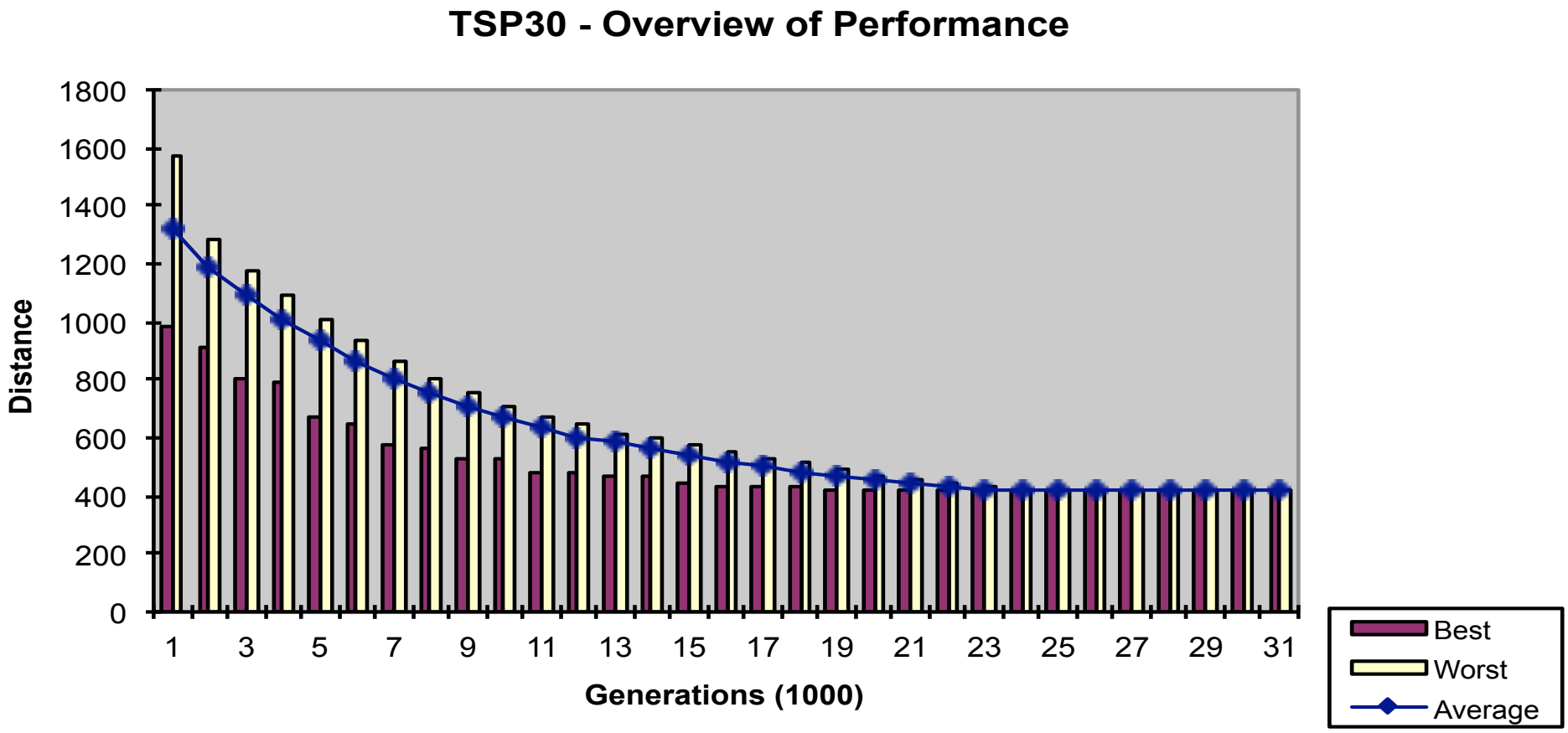
Solution $_k$(Distance = 652)

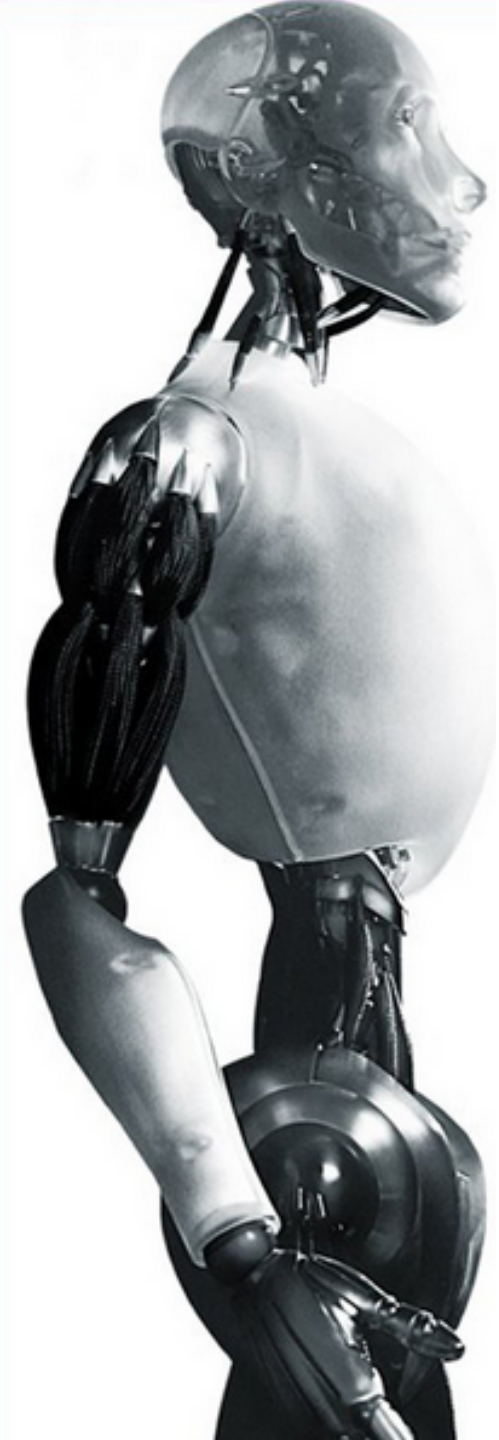


Best Solution (Distance = 420)



Overview of Performance

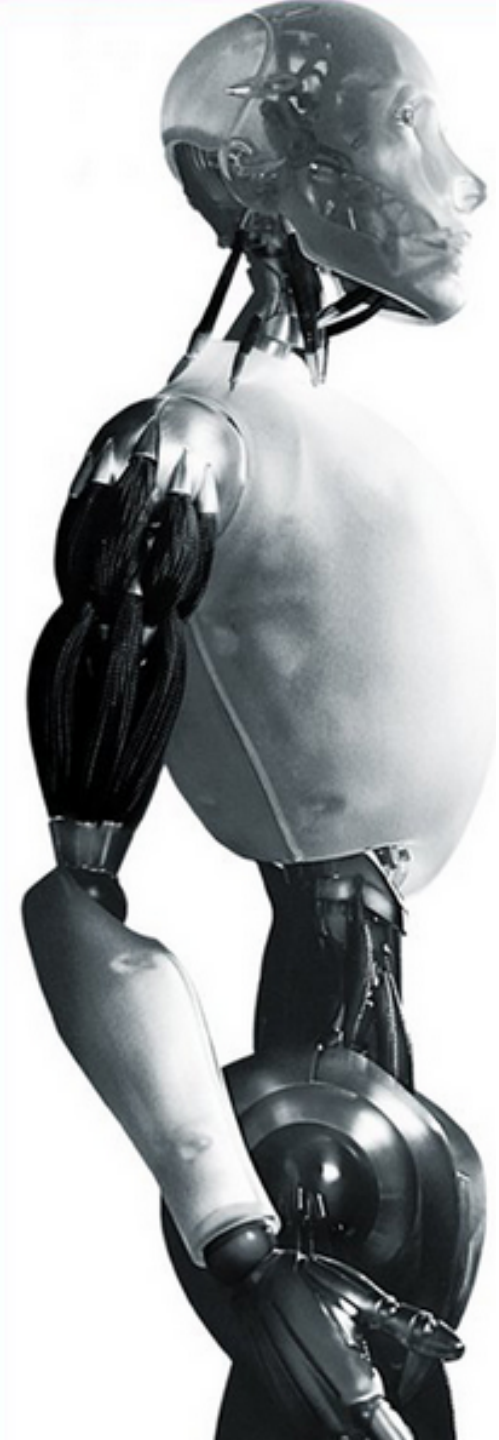




Considering the GA Technology

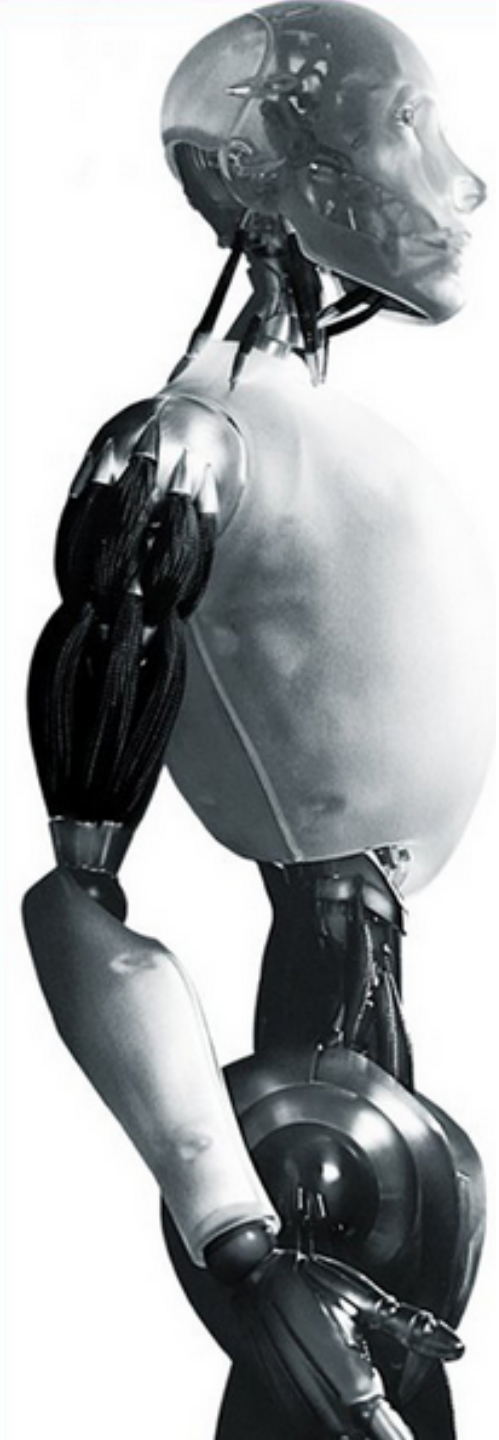
"Almost eight years ago ... people at Microsoft wrote a program [that] uses some genetic things for finding short code sequences. Almost all Microsoft applications products have shipped with pieces of code created by that system."

- Nathan Myhrvold,
Microsoft Advanced Technology Group, Wired Magazine



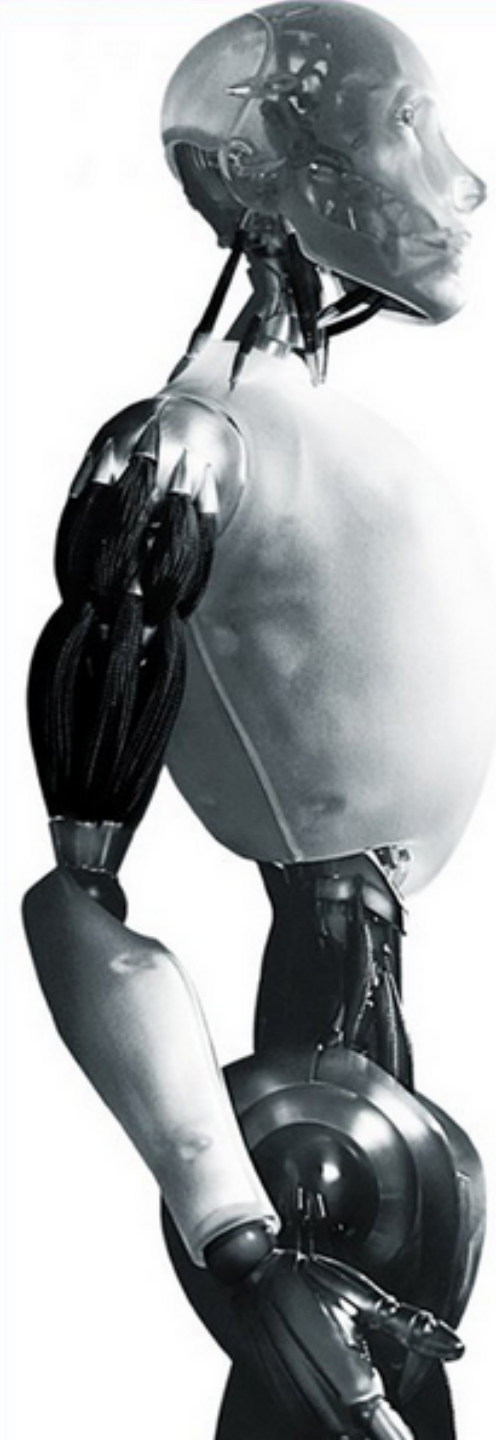
Issues for GA Practitioners

- Choosing basic implementation issues:
  - representation
  - population size, mutation rate, ...
  - selection, deletion policies
  - crossover, mutation operators
- Termination Criteria
- Performance, scalability
- Solution is only as good as the evaluation function (often hardest part)



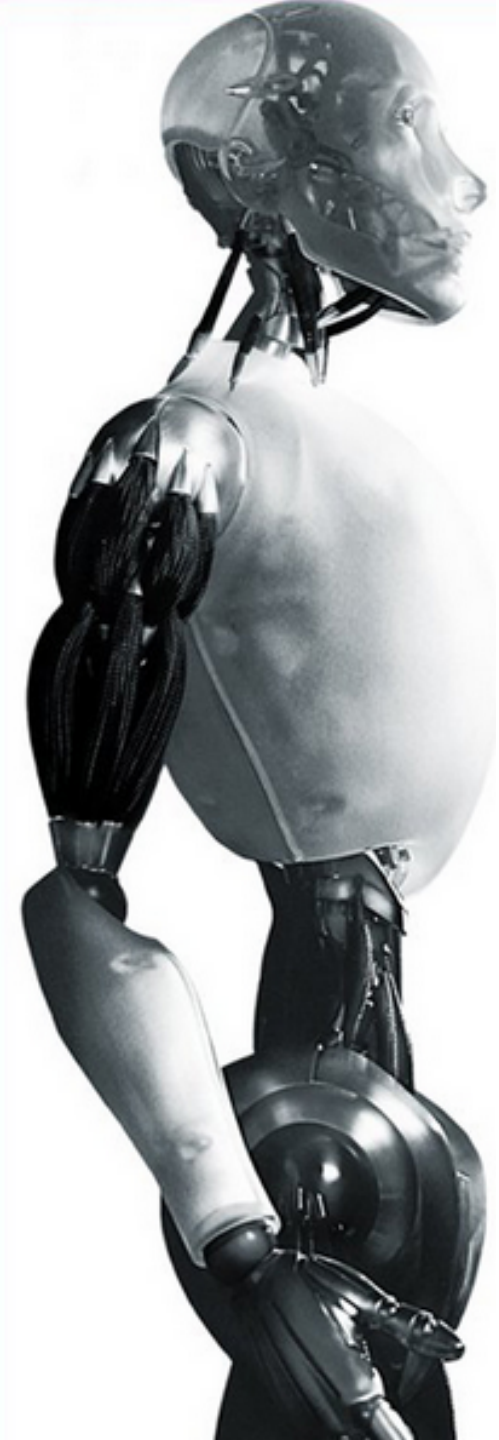
Benefits of Genetic Algorithms

- Concept is easy to understand
- Modular, separate from application
- Supports multi-objective optimization
- Good for "noisy" environments
- Always an answer; answer gets better with time
- Inherently parallel; easily distributed



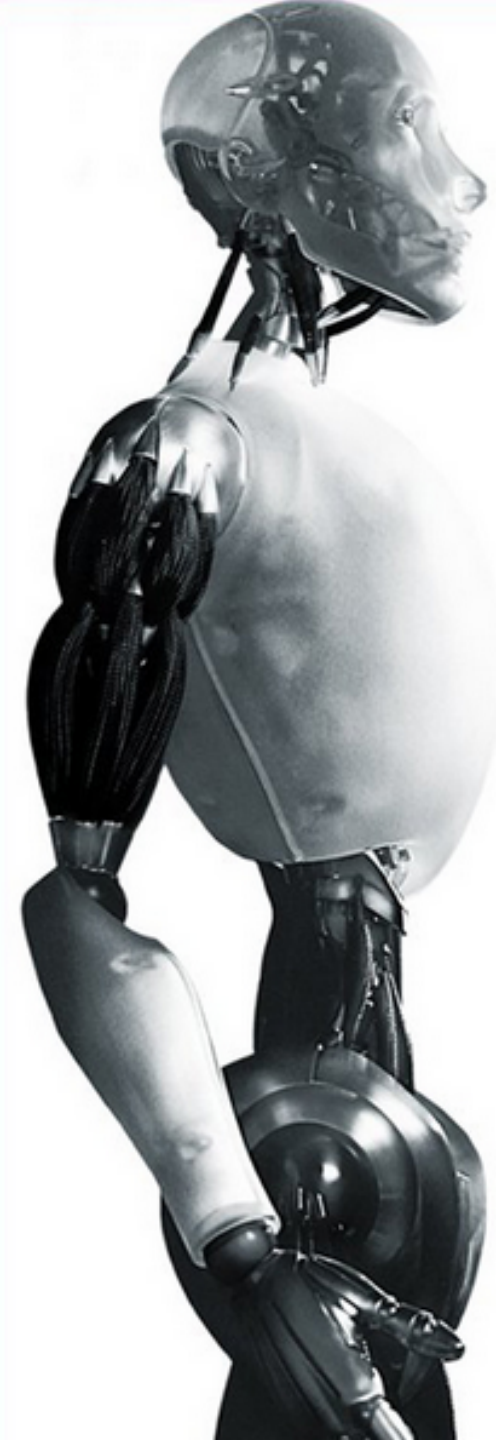
Benefits of Genetic Algorithms (cont.)

- Many ways to speed up and improve a GA-based application as knowledge about problem domain is gained
- Easy to exploit previous or alternate solutions
- Flexible building blocks for hybrid applications
- Substantial history and range of use



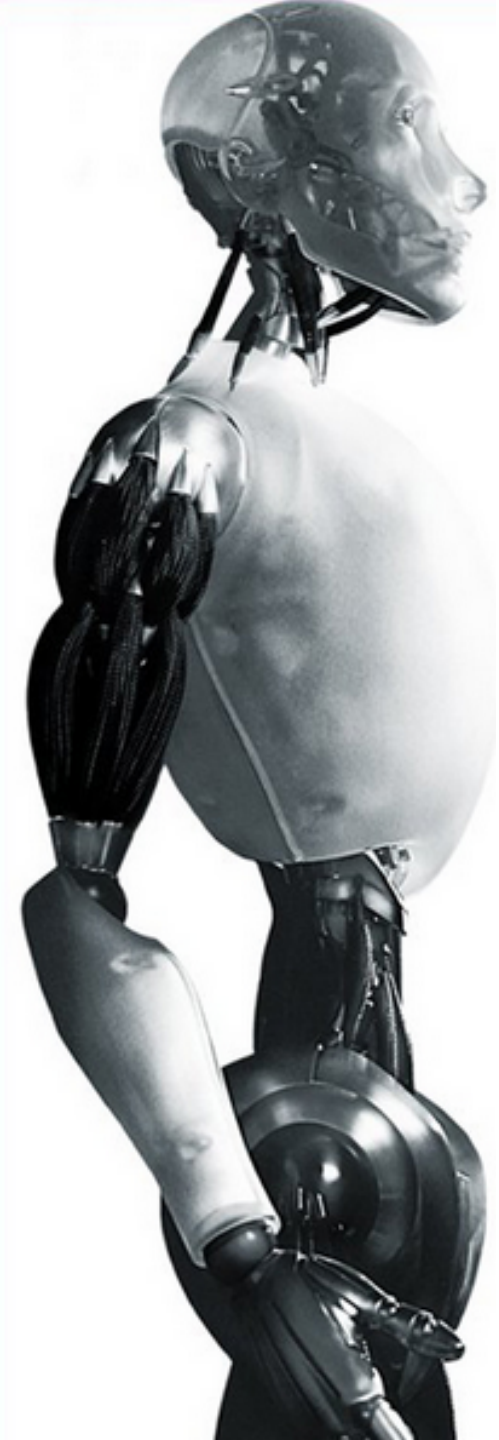
When to Use a GA

- Alternate solutions are too slow or overly complicated
- Need an exploratory tool to examine new approaches
- Problem is similar to one that has already been successfully solved by using a GA
- Want to hybridize with an existing solution
- Benefits of the GA technology meet key problem requirements



Some GA Application Types

Domain	Application Types
Control	gas pipeline, pole balancing, missile evasion, pursuit
Design	semiconductor layout, aircraft design, keyboard configuration, communication networks
Scheduling	manufacturing, facility scheduling, resource allocation
Robotics	trajectory planning
Machine Learning	designing neural networks, improving classification algorithms, classifier systems
Signal Processing	filter design
Game Playing	poker, checkers, prisoner's dilemma
Combinatorial Optimization	set covering, travelling salesman, routing, bin packing, graph colouring and partitioning



Conclusions

Question:

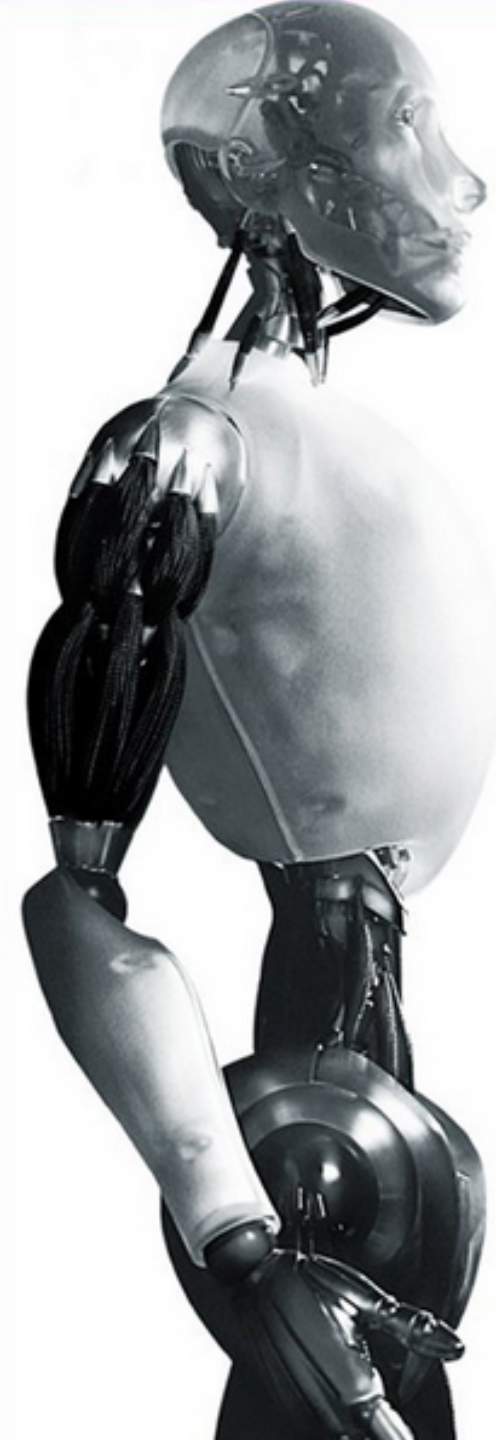
'If GAs are so smart, why ain't they rich?'

Answer:

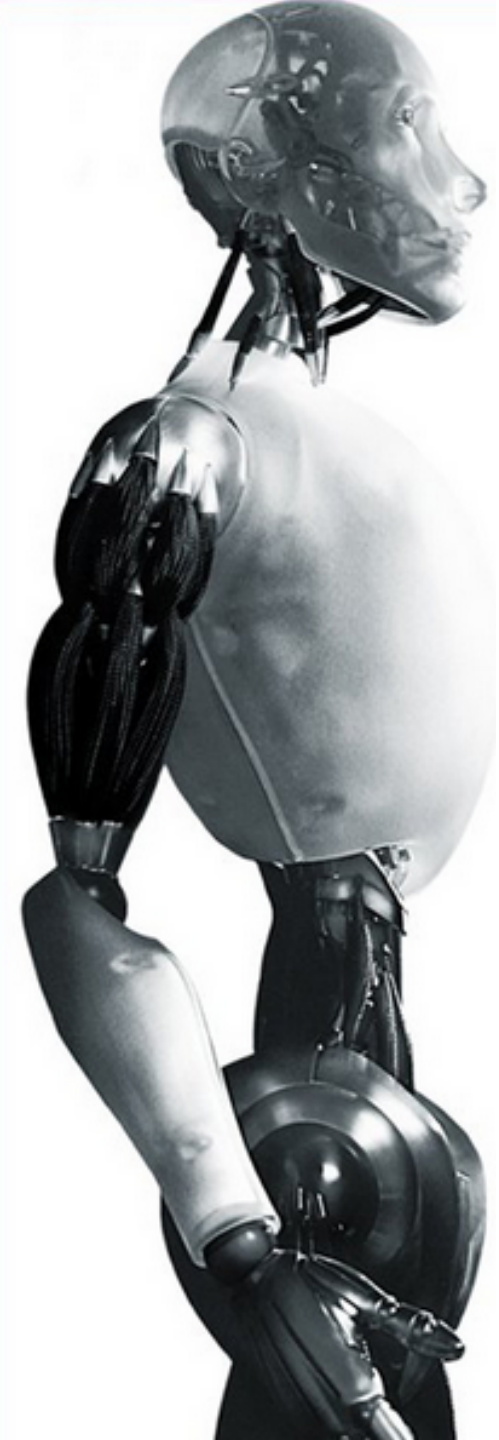
*'Genetic algorithms **are** rich - rich in application across a large and growing number of disciplines.'*

- David E. Goldberg,

Genetic Algorithms in Search, Optimization and Machine Learning

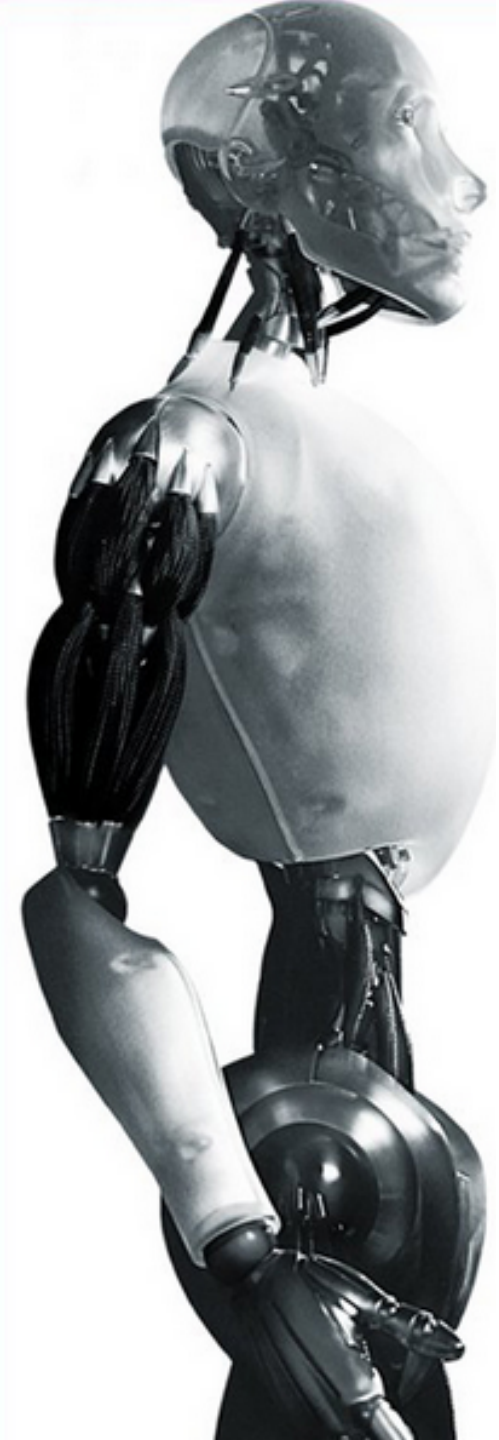


Questions?



Exercise

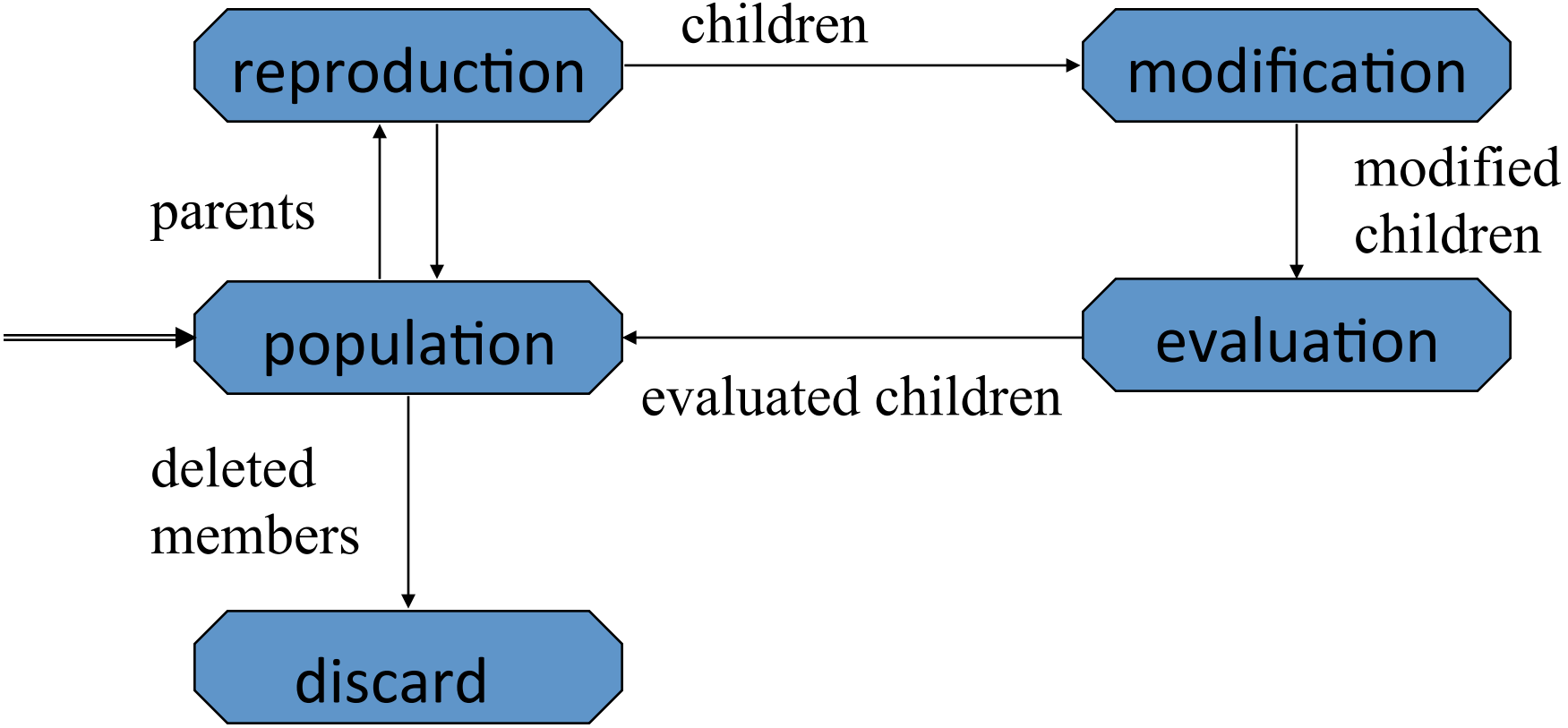
- Implement the Travelling Salesman problem using GAs



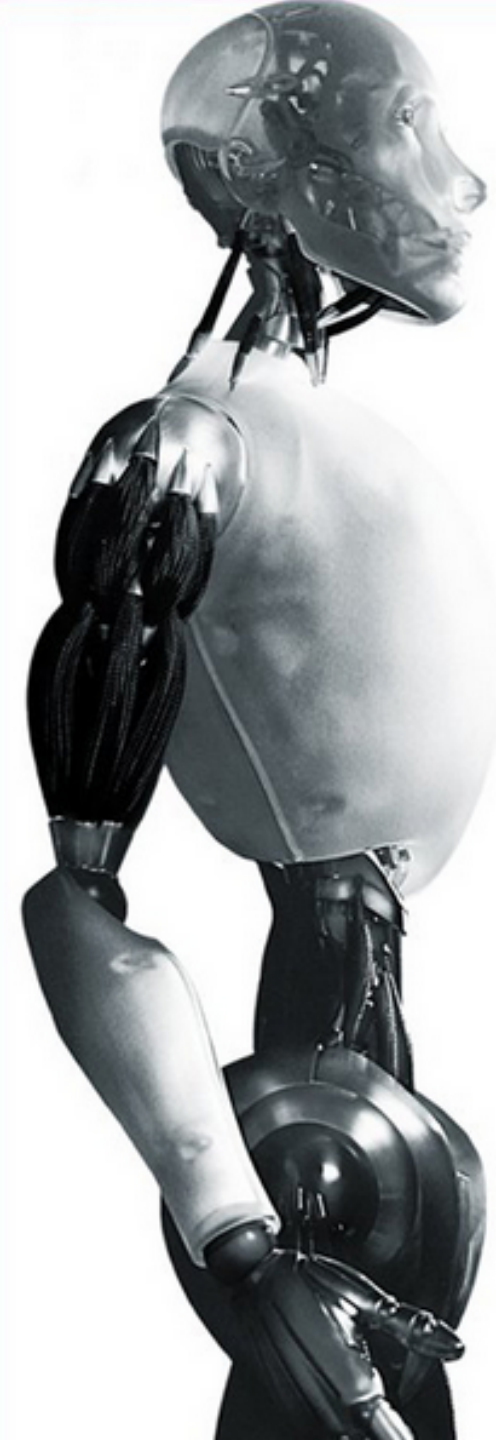
Simple Genetic Algorithm

```
{
    initialize population;
    evaluate population;
    while TerminationCriteriaNotSatisfied
    {
        select parents for reproduction;
        perform recombination and
          mutation;
        evaluate population;
    }
}
```

The GA Cycle of Reproduction



	Barcelona	Belgrade	Berlin	Brussels	Bucharest	Budapest	Copenhagen	Dublin	Hamburg	Istanbul
Barcelona	0	1528	1498	1063	1968	1499	1758	1469	1472	2230
Belgrade	1528	0	999	1373	447	316	1327	2145	1230	809
Berlin	1498	999	0	652	1293	689	354	1315	255	1735
Brussels	1063	1373	652	0	1770	1132	767	773	490	2179
Bucharest	1968	447	1293	1770	0	640	1572	2535	1544	446
Budapest	1499	316	689	1132	640	0	1011	1895	928	1065
Copenhagen	1758	1327	354	767	1572	1011	0	1238	288	2017
Dublin	1469	2145	1315	773	2535	1895	1238	0	1073	2950
Hamburg	1472	1230	255	490	1544	928	288	1073	0	1984
Istanbul	2230	809	1735	2179	446	1065	2017	2950	1984	0



Questions …

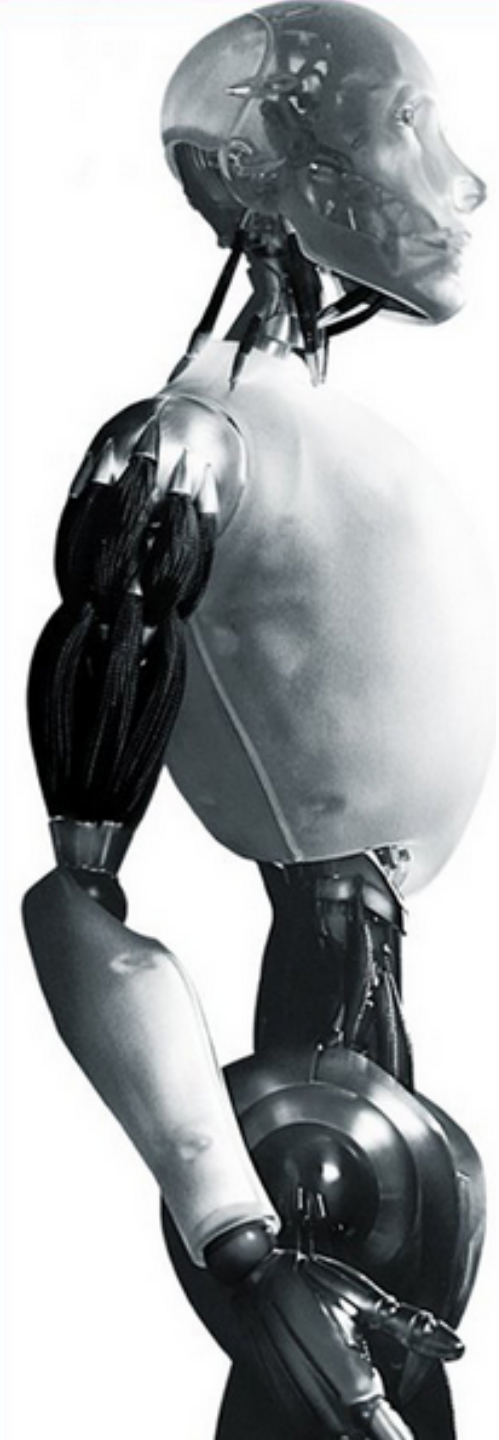
- Combinations possible

  3,628,800

- What is the best route?

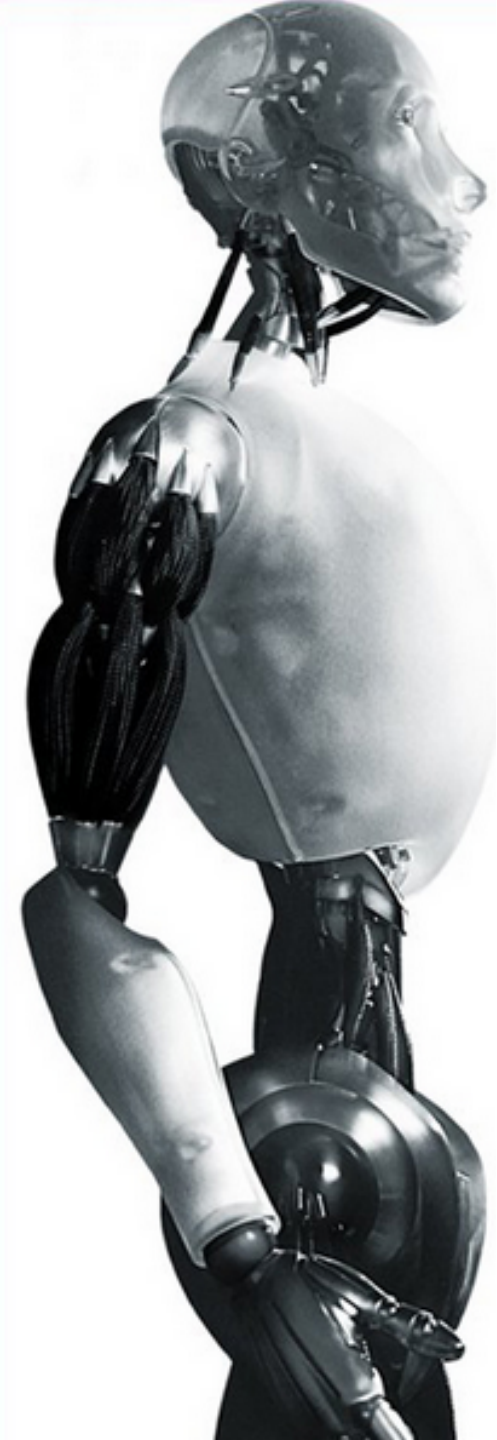
- How many generations did it take to find it?

- How can we optimize it further?



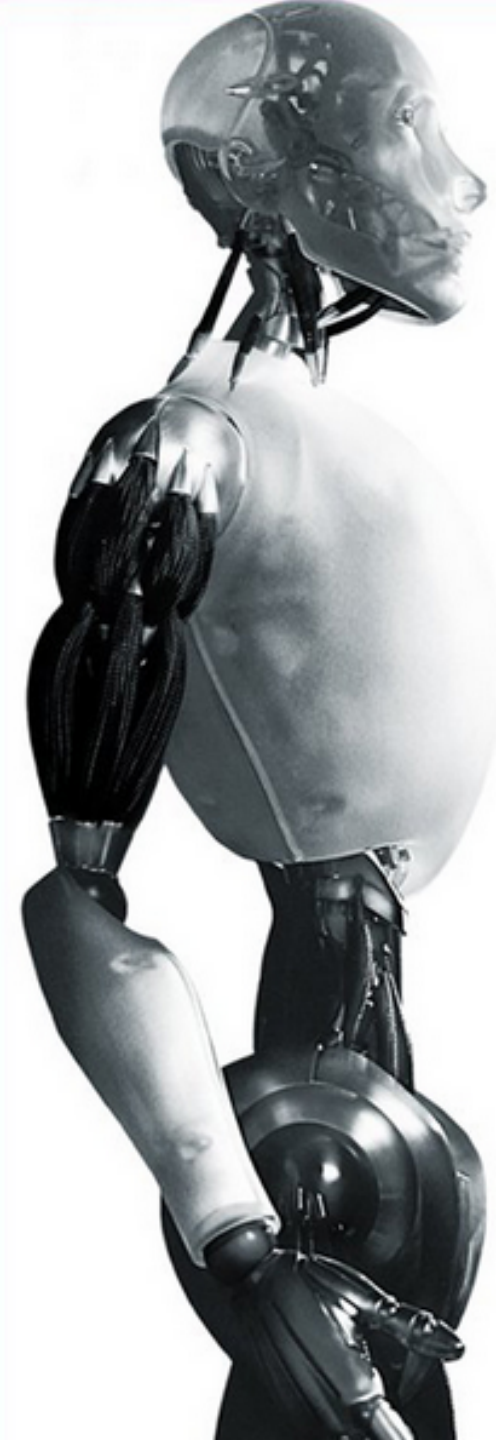
If we add more cities what happens?

	Madrid	Milan	Moscow
Barcelona	505	725	3007
Belgrade	2027	885	1711
Berlin	1868	841	1608
Brussels	1314	697	2253
Bucharest	2470	1331	1498
Budapest	1975	789	1565
Copenhagen	2072	1158	1559
Dublin	1450	1413	2792
Hamburg	1785	900	1780
Istanbul	2735	1669	1754
Kiev	2859	1673	757
London	1263	958	2498
Madrid	0	1188	3438
Milan		0	2283
Moscow			0



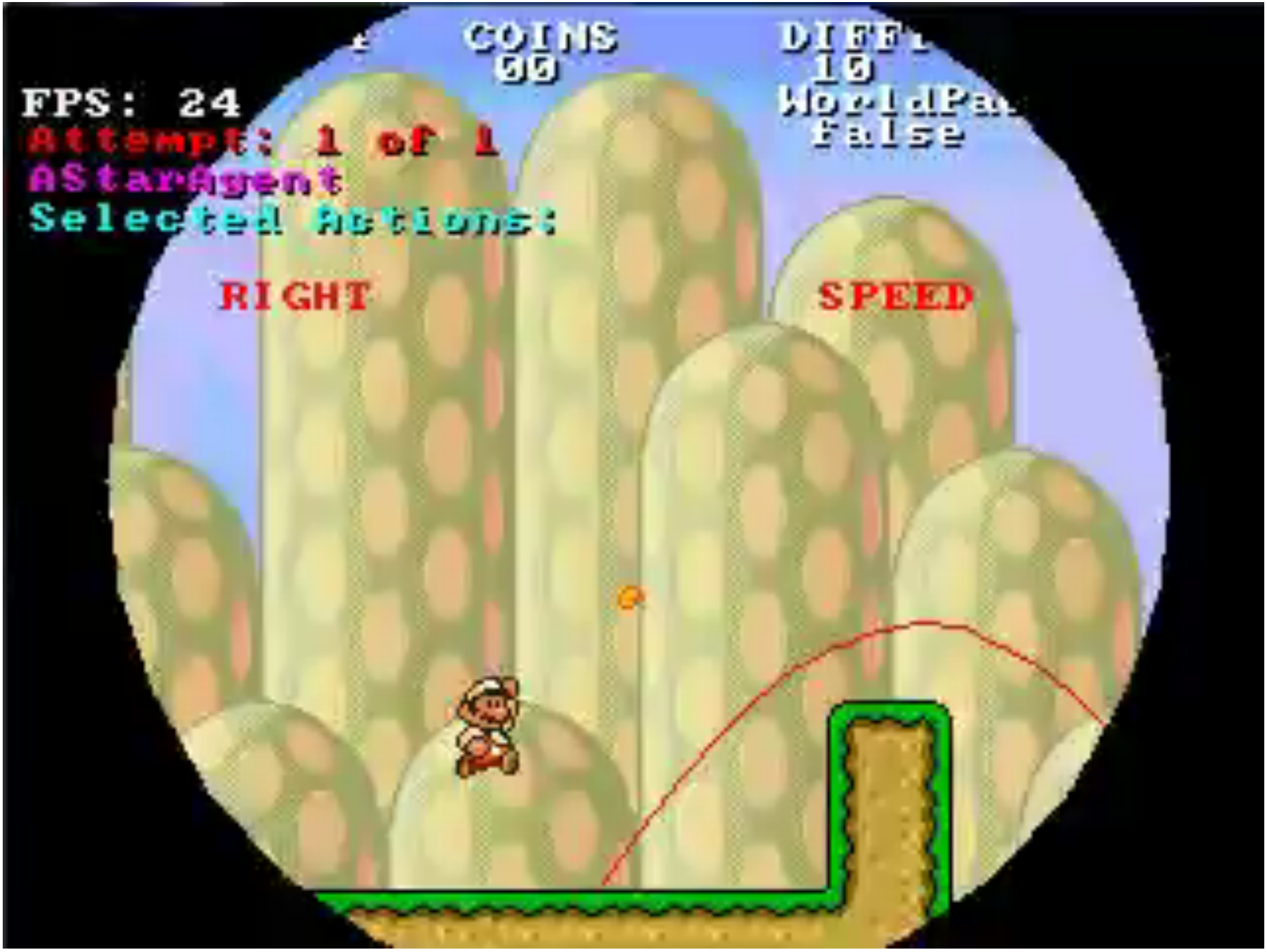
Questions …

- Combinations possible with 11
  39,916,800
- Combinations possible with 12
  479,001,600
- Combinations possible with 13
  6,227,020,800



Exercise

- How would you implement GA's in Super Mario?
  - Fitness Function
  - Mutation
  - Crossover
  - Survival rate


COINS
00
DIFF
10
WorldPa
false
FPS: 24
Attempt: 1 of 1
AStarAgent
Selected Actions:
RIGHT
SPEED