Genetic Algorithms

CSE 590 DATA MINING

Prof. Anita Wasilewska
SUNY Stony Brook

A presentation by:
Rucha Lale, Gaurav Naigaonkar, Kumaran Shanmugam

References

- D. E. Goldberg, 'Genetic Algorithm In Search, Optimization And Machine Learning', New York: Addison – Wesley (1989)
- John H. Holland 'Genetic Algorithms', Scientific American Journal, July 1992.
- Kalyanmoy Deb, 'An Introduction To Genetic Algorithms', Sadhana, Vol. 24 Parts 4 And 5.
- T. Starkweather, et al, 'A Comparison Of Genetic Sequencing Operators', International Conference On Gas (1991)
- http://obitko.com/tutorials/genetic-algorithms/introduction.php
- http://en.wikipedia.org/wiki/Genetic_algorithm
- http://en.wikipedia.org/wiki/Fitness_function
- http://en.wikipedia.org/wiki/Crossover_(genetic_algorithm)
- http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php/
- Tutorial: Wendy Williams Metaheuristic Algorithms, Genetic Algorithms a Tutorial"
- http://brainz.org/15-real-world-applications-genetic-algorithms/
- Data Mining: Concepts & Techniques by Jiawei Han and Micheline Kamber

Quick Background

- Idea of evolutionary computing was introduced in the 1960s by I. Rechenberg in his work "*Evolution strategies*" (*Evolutionsstrategie* in original).

- Genetic Algorithms (GAs) were invented by John Holland and developed by him and his students and colleagues. This lead to Holland's book "*Adaption in Natural and Artificial Systems*" published in 1975.

- In 1992 John Koza used genetic algorithm to evolve programs to perform certain tasks. He called his method "Genetic Programming".

Citation : "http://obitko.com/tutorials/genetic-algorithms/introduction.php"

What is GA?

- "A genetic algorithm (**GA**) is a search technique used in computing to find exact or approximate solutions to optimization and search problems."
- Inspired by Theory of Evolution by Darwin the solutions generated by GA are evolved over the time by using evolutionary biology techniques such as mutation, selection etc.

Citation "http://en.wikipedia.org/wiki/Genetic_algorithm"

Role of GA

- To solve optimization and search related problems.

- Applying genetic algorithms for attributes selection as discussed in "*Genetic Algorithms as a Tool for Feature Selection in Machine Learning*" by Vafaie and De Jong, and "*Feature Subset Selection Using A Genetic Algorithm*" by Yang and Honavar.

- Evolutionary search for attribute selection for clustering as introduced in "*Feature Selection in Unsupervised Learning via Evolutionary Search*" by Kim, Street, and Menczer.

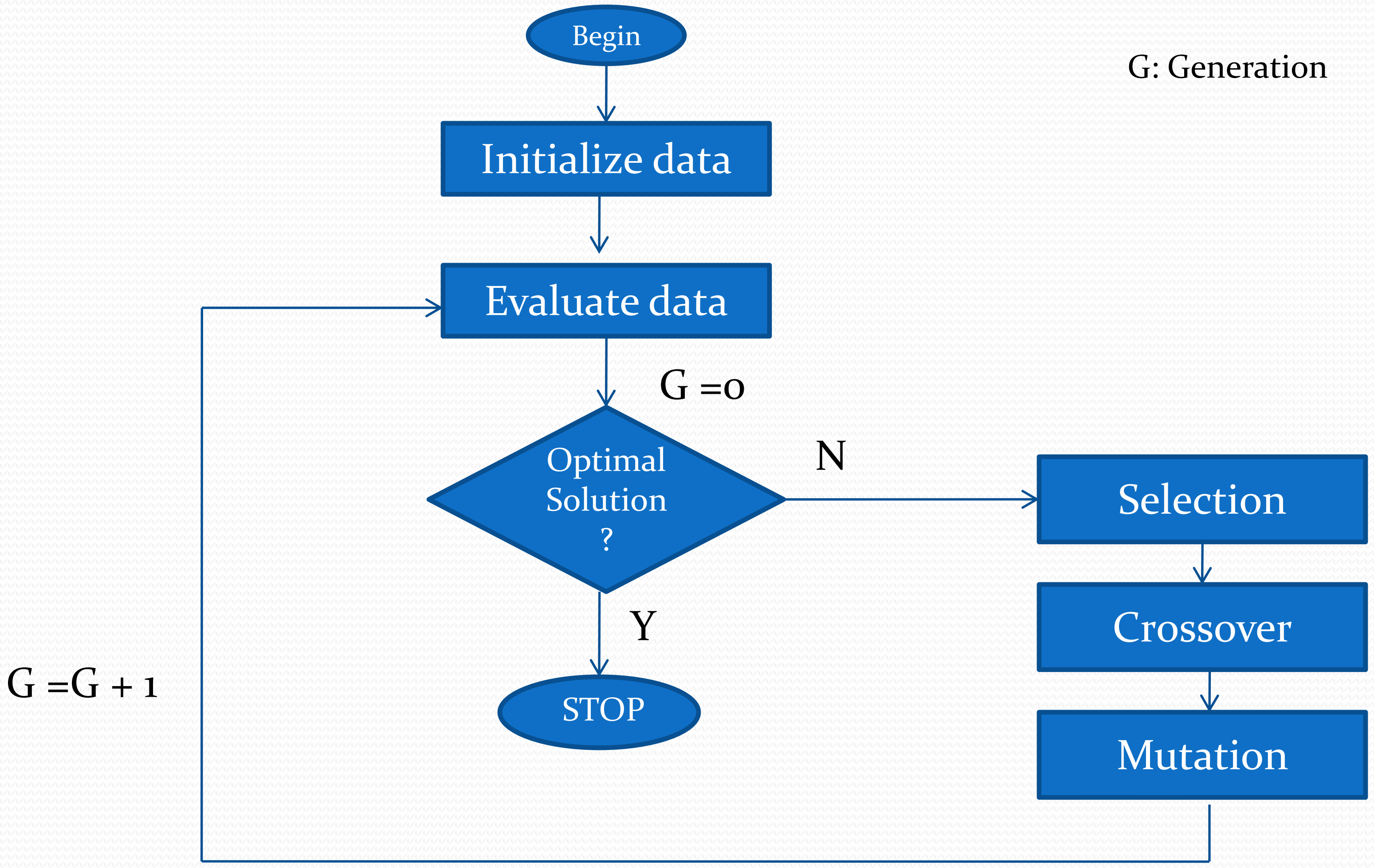
Citation "http://www.public.iastate.edu/~olafsson/mining_course_info.html "

Components of GA

- Encoding (e.g. binary encoding, permutation encoding)
- Initialization Function
- Evaluation Function (also called fitness function)
- Selection Function (Roulette selection, tournament selection, etc.)
- Reproduction (using crossover, mutation, elitism etc.)
- Termination

Citation “http://en.wikipedia.org/wiki/Genetic_algorithm”

GA in a gist



Citation "http://obitko.com/tutorials/genetic-algorithms/ga-basic-description.php"

Overall Algorithm

1. Generate random population say *n*.

2. Evaluate the fitness *f(x)* of each *x* in the population

3. Create a new population by repeating following
   a) Select two parents from a population according to their fitness
   b) Crossover the parents to form a new offspring (children).
   c) Mutate new offspring at each locus (position).

4. Use new generated population for a further run of algorithm

5. If the end condition is satisfied(i.e. desired result is optimal), **stop**, and return the best solution in current population

Citation "http://obitko.com/tutorials/genetic-algorithms/ga-basic-description.php"

Encoding types

- Binary Encoding
  - The input is expressed as strings of 0s and 1s. Each bit presents a particular characteristic.
  - E.g if the input is a set of chromosomes one such chromosome may be presented as 101100101100101100101

Chromosome A	101100101100101011100101
Chromosome B	111111100000110000011111

- Permutation Encoding
  - The input is represented as a string of numbers.
  - E.g. If the input is a sequence of cities to be visited then one such sequence can be 123456

Chromosome A	1 5 3 2 6 4 7 9 8
Chromosome B	8 5 6 7 2 3 1 4 9

Citation "http://obitko.com/tutorials/genetic-algorithms/operators.php"

Encoding types (contd.)

- Value Encoding
  - The input is a set of values where each value is for a specific characteristic. E.g. For finding weights in neural networks, one possible input can be 2 0.3 -0.3 0.1 1 0.2 -0.2

Chromosome A	1.235 5.323 0.454 2.321 2.454
Chromosome B	(left), (back), (left), (right), (forward)

- Others: Tree encoding, etc.

Evaluation (Fitness Function)

- *"Means to quantify the optimality of the solution so that it can be ranked against all other solution."*

- It shows "how close" is the given solution to the desired output.

- The value of a fitness function depends on the problem that we are trying to solve.

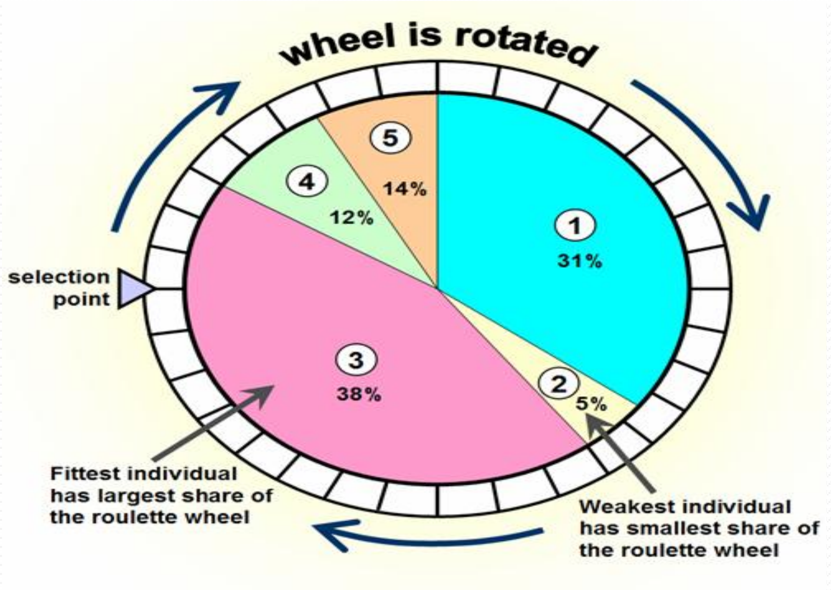
E.g in case of TSP the value of the fitness function is the sum of the distances between the cities.

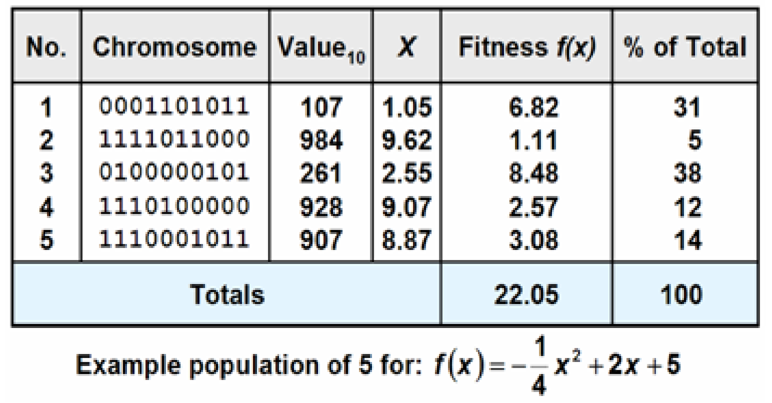
Citation "http://en.wikipedia.org/wiki/Fitness_function"

Selection

Roulette Wheel Selection:

- Fitness level is used to associate a probability of selection with each individual solution e.g. chromosome.
- We first calculate the fitness for each input and then represent it on the wheel in terms of percentages.
- In a search space of ‘N’ chromosomes, we spin the roulette wheel


wheel is rotated
selection point
5
14%
4
12%
1
31%
3
38%
2
5%
Fittest individual has largest share of the roulette wheel
Weakest individual has smallest share of the roulette wheel



No.	Chromosome	Value$_{10}$	X	Fitness $f(x)$	% of Total
1	0001101011	107	1.05	6.82	31
2	1111011000	984	9.62	1.11	5
3	0100000101	261	2.55	8.48	38
4	1110100000	928	9.07	2.57	12
5	1110001011	907	8.87	3.08	14
Totals				22.05	100

Example population of 5 for: $f(x) = -\frac{1}{4}x^2 + 2x + 5$

Value$_{10}$: Value of chromosome to the base 10
X = Value$_{10}$ normalized between 0 to 10 range

Citation “http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php/ ”

Selection (contd.)

- Tournament Selection:
  - Two solutions are picked out of the pool of possible solutions, their fitness is compared, and the better is permitted to reproduce.
  - Deterministic tournament selection selects the best individual in each tournament.
  - Can take advantage of parallel architecture

- Others:
  - Rank Selection
  - Steady State Selection etc.

Citation “http://obitko.com/tutorials/genetic-algorithms/operators.php”

Selection: Elitism

- Elitism first copies the best solutions to the next set of solutions. E.g. it copies the best input chromosome (or few chromosomes) to the new population.

- The rest of the solution is done by any of the above mentioned ways.

- Elitism can very rapidly increase performance of GA, because it prevents losing the best found solution.

Citation "http://obitko.com/tutorials/genetic-algorithms/selection.php"

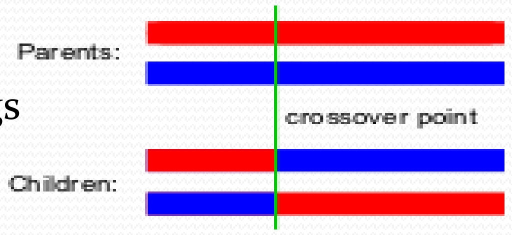
Crossover

- *Varies the programming of input solutions from one generation to the next*.
- Two strings are picked from the input pool at random to cross over.
- The method chosen depends on the Encoding Method.
- E.g. single point crossover, two point crossover, uniform crossover etc.

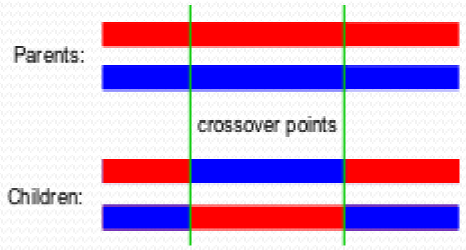
Citation "Wendy Williams Metaheuristic Algorithms, Genetic Algorithms a Tutorial"

Crossover Methods

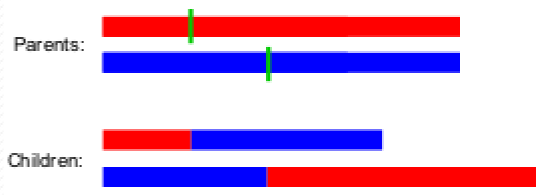
- Single Point Crossover
  - A single crossover point on both parents' organism strings is selected.
  - Data beyond that point in either organism string is swapped



- Two Point Crossover
  - Two points to be selected on the parent organism strings.
  - Everything between the two points is swapped between the parent organisms.



- Cut and Splice
  - Results in a change in length of the children strings.
  - Each parent string has a separate choice of crossover point



- Others like Uniform Crossover etc.

Citation "http://en.wikipedia.org/wiki/Crossover_(genetic_algorithm) "

Mutation

- Mutation is a genetic operator that alters one ore more gene values in a chromosome from its initial state.
- Maintains the genetic diversity from one generation of a population to the next. E.g. for binary encoded chromosomes
- **Flip Bit** -Simply inverts the value of the chosen gene (0 goes to 1 and 1 goes to 0). This mutation operator can only be used for binary genes.

Chromosome A	1011 1100 1110 1100
Mutated A	1010 1100 1100 1101

- **Boundary** -Replaces the value of the chosen gene with either the upper or lower bound for that gene (chosen randomly). This mutation operator can only be used for integer and float genes.

Chromosome A	1.235 5.323 0.454 2.321 2.454
Mutated A	1 5 0 2 2

- **Others: Uniform, Non-Uniform, Gaussian**

Citation "http://www.nd.com/products/genetic/mutation.htm"

Termination

- After the reproduction steps we may decide to terminate if:
  - The fitness function gives output solutions very close to the desired output. i.e. the difference between the two outputs is below some threshold .
  - A specified number of generations have elapsed.

- The resultant output is the optimal solution.

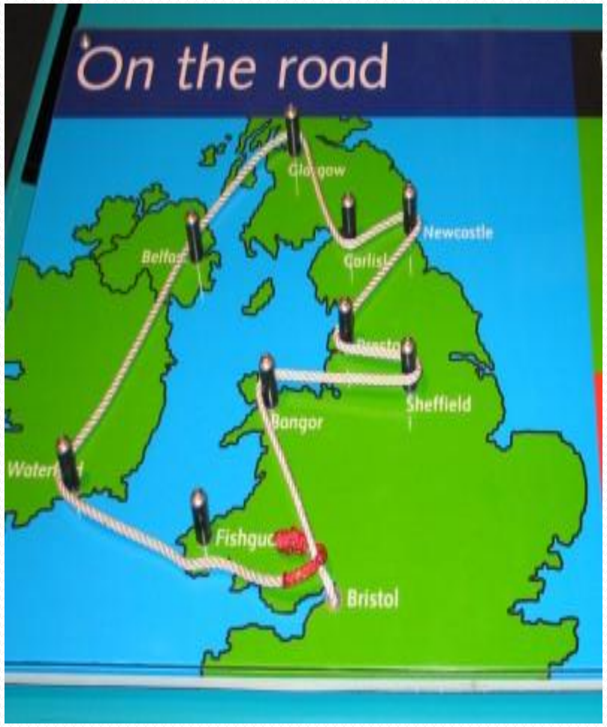
APPLICATIONS

The Traveling Salesman Problem (TSP)

The Traveling Salesman Problem

Given:

- a set of cities &
- a symmetric distance matrix that indicates the cost of travel from each city to every other city.



Goal:

To find the shortest circular tour, visiting every city exactly once, so as to minimize the total travel cost, which includes the cost of traveling from the last city back to the first city

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Encoding

- Every city represented as an integer
- We will take an example involving 6 cities, namely {A, B, C, D, E, F}
- Representation:

City	Encoding
A	1
B	2
C	3
D	4
E	5
F	6

Citation “D. Whitley, et al , ‘Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination’, Handbook Of Genetic Algorithms, New York ”

Encoding (contd.)

- A path between two cities is represented as a sequence of integers from 1 to 6
- For example, the path [1 2 3 4 5 6] represents

  A -> B -> C -> D -> E -> F -> A
- This is an example of "**Permutation Encoding**" as the position of the elements determines the fitness of the solution

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Distance Matrix For TSP

Cities	1	2	3	4	5	6
1	0	500	3000	2500	1000	2000
2	500	0	2500	1000	1500	2000
3	3000	2500	0	1500	2500	2000
4	2500	1000	1500	0	2750	3000
5	1000	1500	2500	2750	0	500
6	2000	2000	2000	3000	500	0

All distances in miles

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Fitness Function

- The "fitness function" is the total cost of the tour represented by each chromosome.

- For example, for the path [1 2 3 4 5 6] the total cost would be sum of distances involves in travelling from A to B to C to D to E to F and back to A

- Fitness = $d(1,2) + d(2,3) + d(3,4) + d(4,5) + d(5,6) + d(6,1)$

  $= 500 + 2500 + 1500 + 2750 + 500 + 2000$

  $= 9750$ *miles*

The Lesser The Sum, The Fitter The Solution Represented By That Chromosome

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Selection Operator

- Now, for selection "***Tournament Selection***" is used.

- Here, tournaments are played between two solutions and the better solution is chosen and placed in the "mating pool".

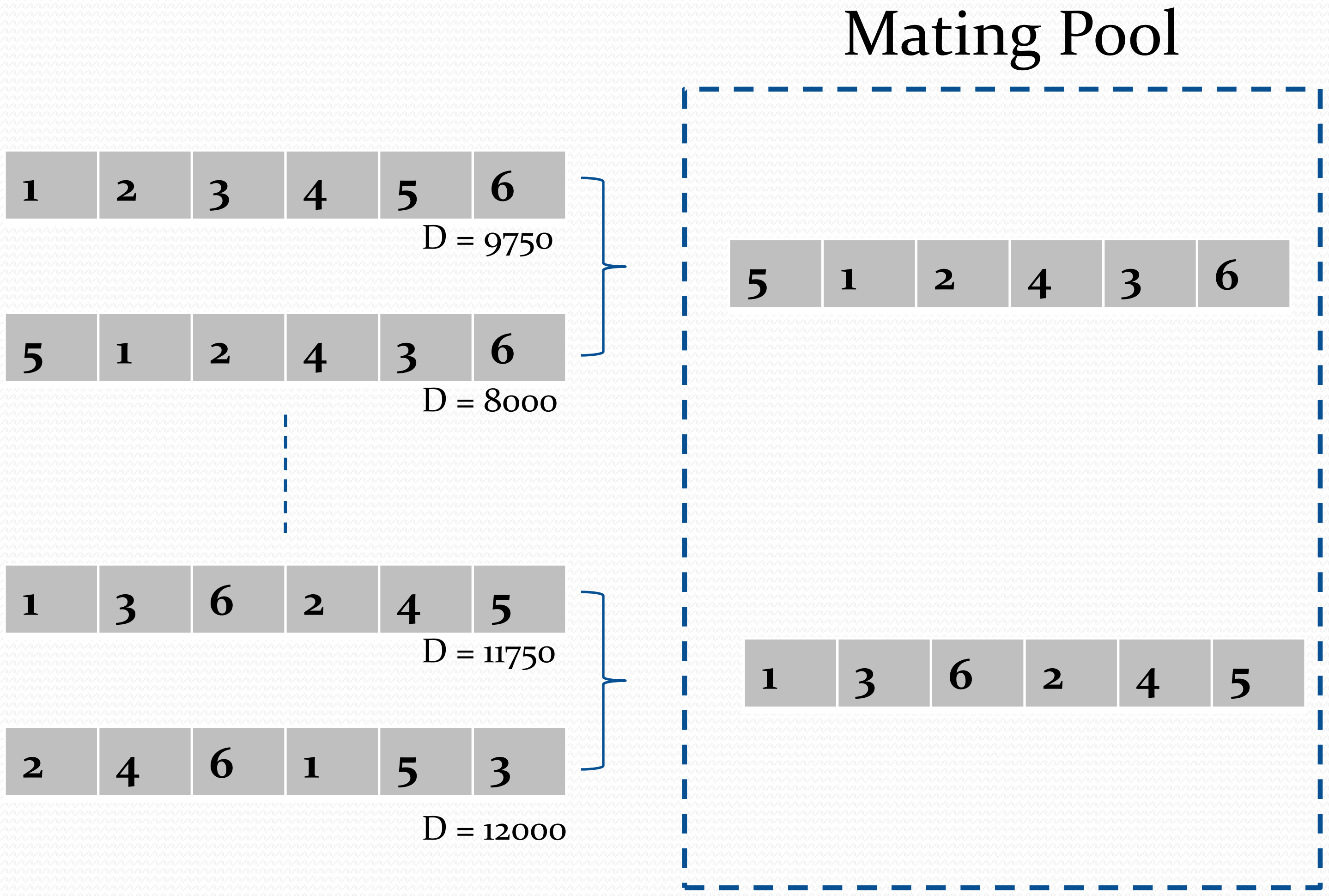
- Note that, tournaments are played between each pair of solutions possible.

  Eg: If there are say fours solutions S1, S2, S3 and S4, then tournaments would be played between S1-S2, S1-S3, S1-S4, S2-S3, S2-S4 and S3-S4

- A mating pool is a set of solutions from which further selection would be made in the next generation.

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Tournament Selection



Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Crossover Operator

- Now, for crossover use "Enhanced Edge Recombination" operator . This involves creating an "Edge Table"
- The Edge Table is an *adjacency table* that lists links into and out of a city found in the two parent sequences.
- If an item is already in the edge table and we are trying to insert it again, that element of a sequence must be a "common edge" and is represented by inverting it's sign.

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Crossover Operator

Parent 1	**1**	**2**	**3**	**4**	**5**	**6**
Parent 2	**1**	**3**	**4**	**2**	**5**	**6**

1	2	-6	3	
2	1	3	4	5
3	2	-4	1	
4	-3	5	2	
5	4	-6	2	
6	-5	-1		

Edge Table

Citation “D. Whitley, et al , ‘Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination’, Handbook Of Genetic Algorithms, New York ”

Enhanced Edge Recombination Algorithm

1. Choose the initial city from one of the two parent tours. This is the *current city*.

2. Remove all occurrences of the *current city* from the RHS of edge table.

3. If the *current city* has entries in it's edge-list, go to *step 4* otherwise go to *step 5*.

4. Determine which of the cities in the edge-list of the *current city* has the fewest entries in it's own edge-list. The city with fewest entries becomes the *current city*. In case a negative integer is present, it is given preference. Ties are broken randomly. Go to *step 2*.

5. If there are no remaining *unvisited* cities, then *stop*. Otherwise, randomly choose an *unvisited* city and go to *step 2*.

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Example

Step 1

1	2	-6	3	
2	1	3	4	5
3	2	-4	1	
4	-3	5	2	
5	4	-6	2	
6	-5	-1		

1					

Step 2

1	2	-6	3
2	3	4	5
3	2	-4	
4	-3	5	2
5	4	-6	2
6	-5		

1	6				

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Example (contd.)

Step 3

1	2	3	
2	3	4	5
3	2	-4	
4	-3	5	2
5	4	2	
6	-5		

1	6	5			

Step 4

1	2	3
2	3	4
3	2	-4
4	-3	2
5	4	2
6		

1	6	5	4		

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

Example (contd.)

Step 5

1	2	3
2	3	
3	2	
4	-3	2
5	2	
6		

1	6	5	4	3	

Step 6

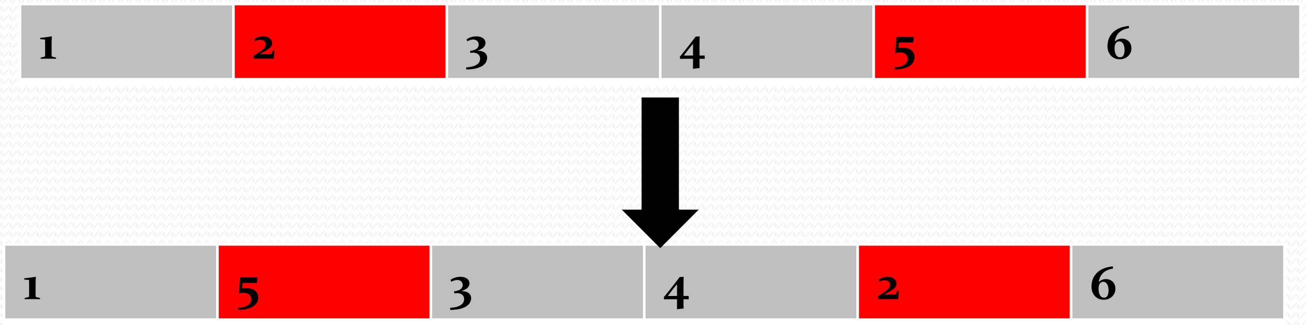
1	2
2	
3	2
4	2
5	2
6	

1	6	5	4	3	2

Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

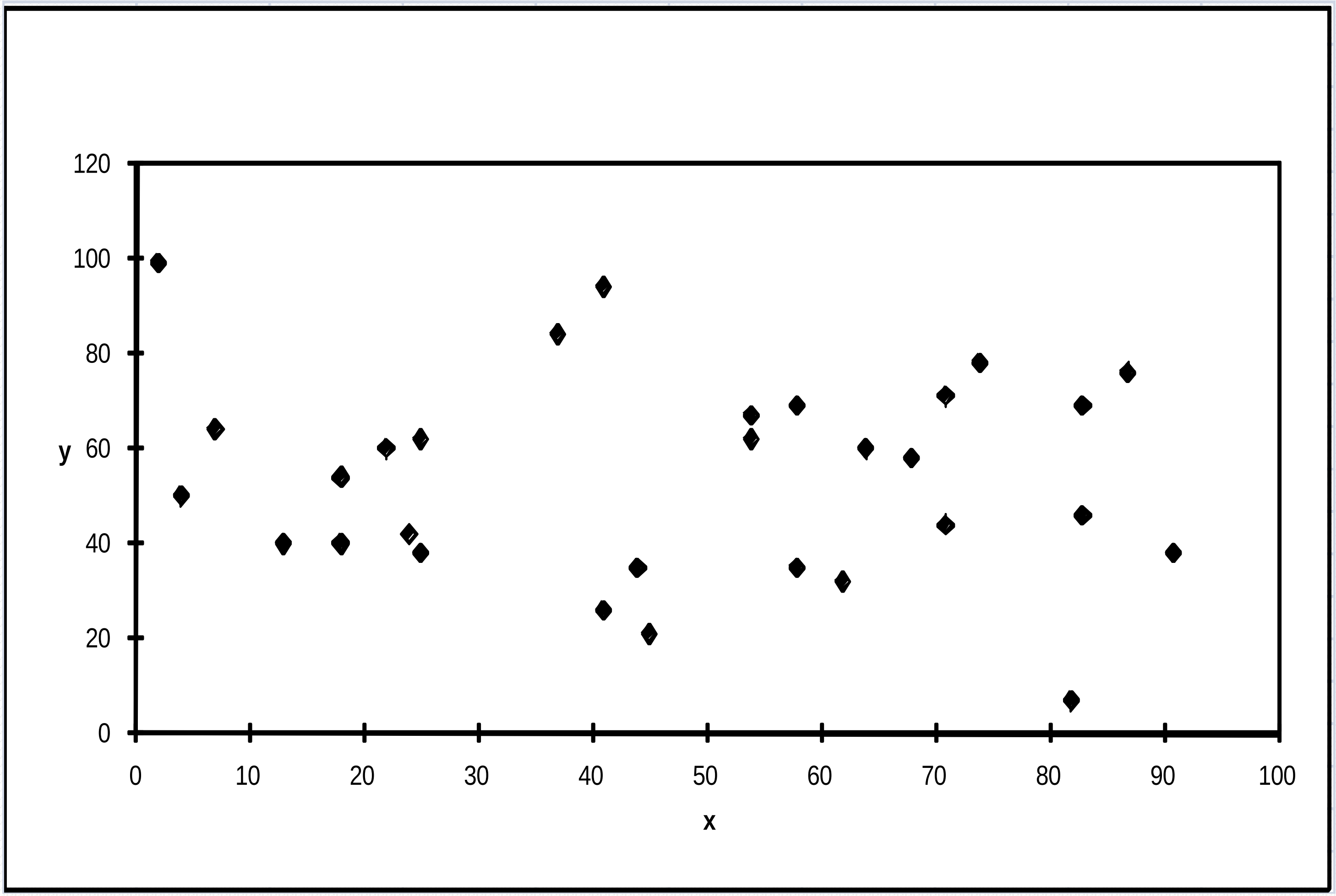
Mutation Operator

- The mutation operator induces a change in the solution, so as to maintain diversity in the population and prevent Premature Convergence.
- We mutate the string by randomly selecting any two cities and interchanging their positions in the solution, thus giving rise to a new tour.



Citation "D. Whitley, et al , 'Traveling Salesman And Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination', Handbook Of Genetic Algorithms, New York "

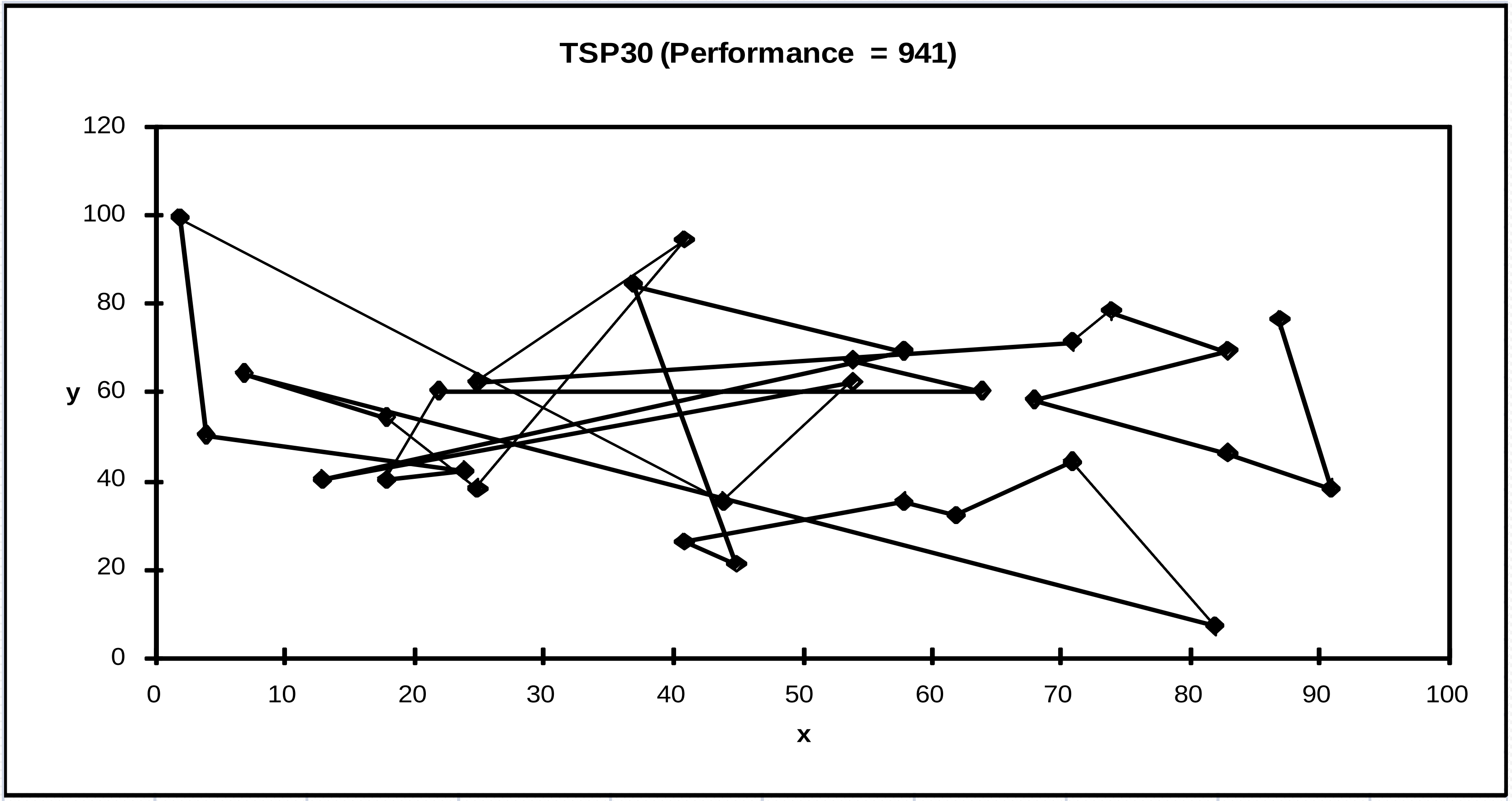
TSP Example: 30 Cities



Example from: “Genetic Algorithms: A Tutorial” by Wendy Williams

Solution$_i$ (Distance = 941)

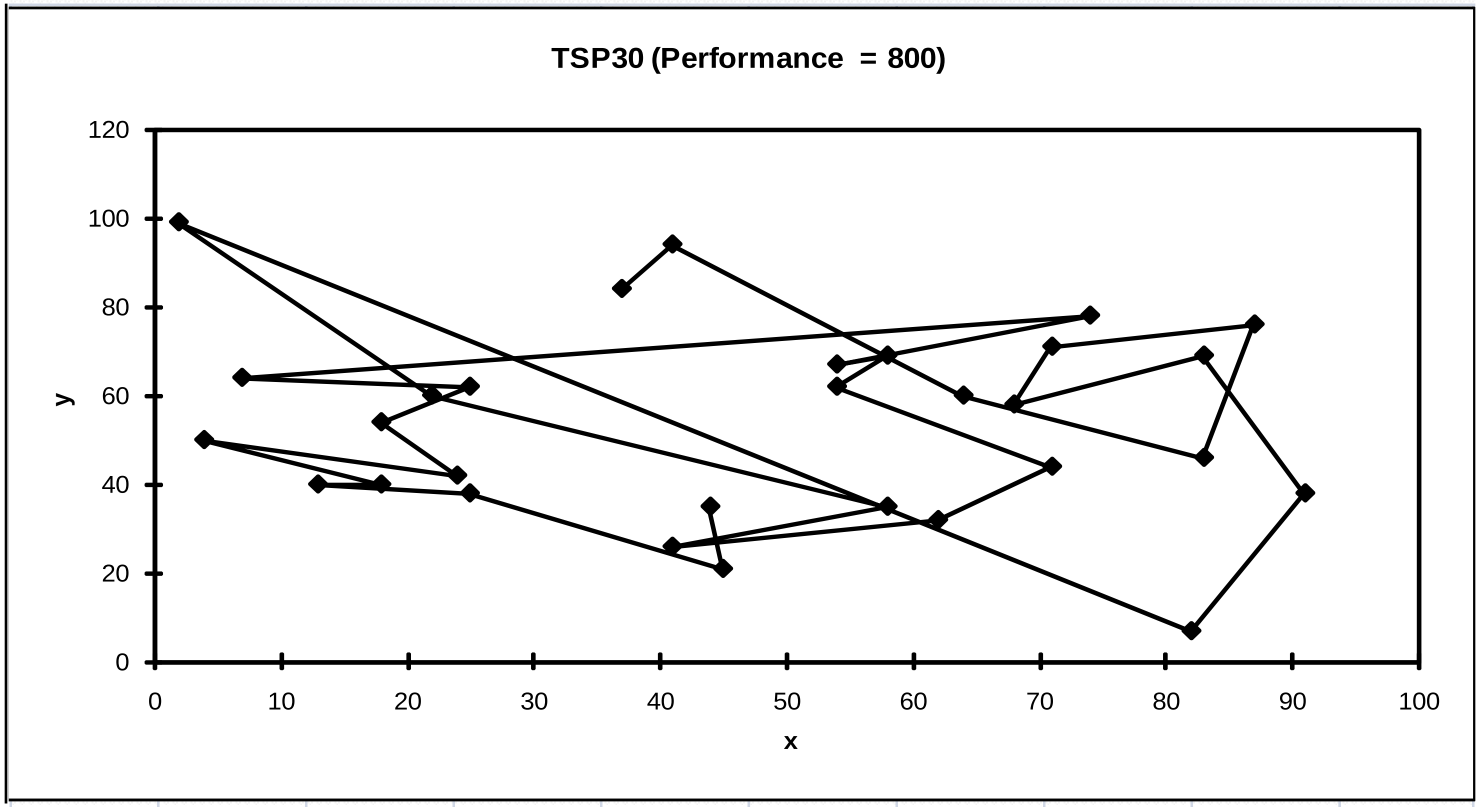
After around 3-4 generations



Example from: "Genetic Algorithms: A Tutorial" by Wendy Williams

Solution$_i$ (Distance = 800)

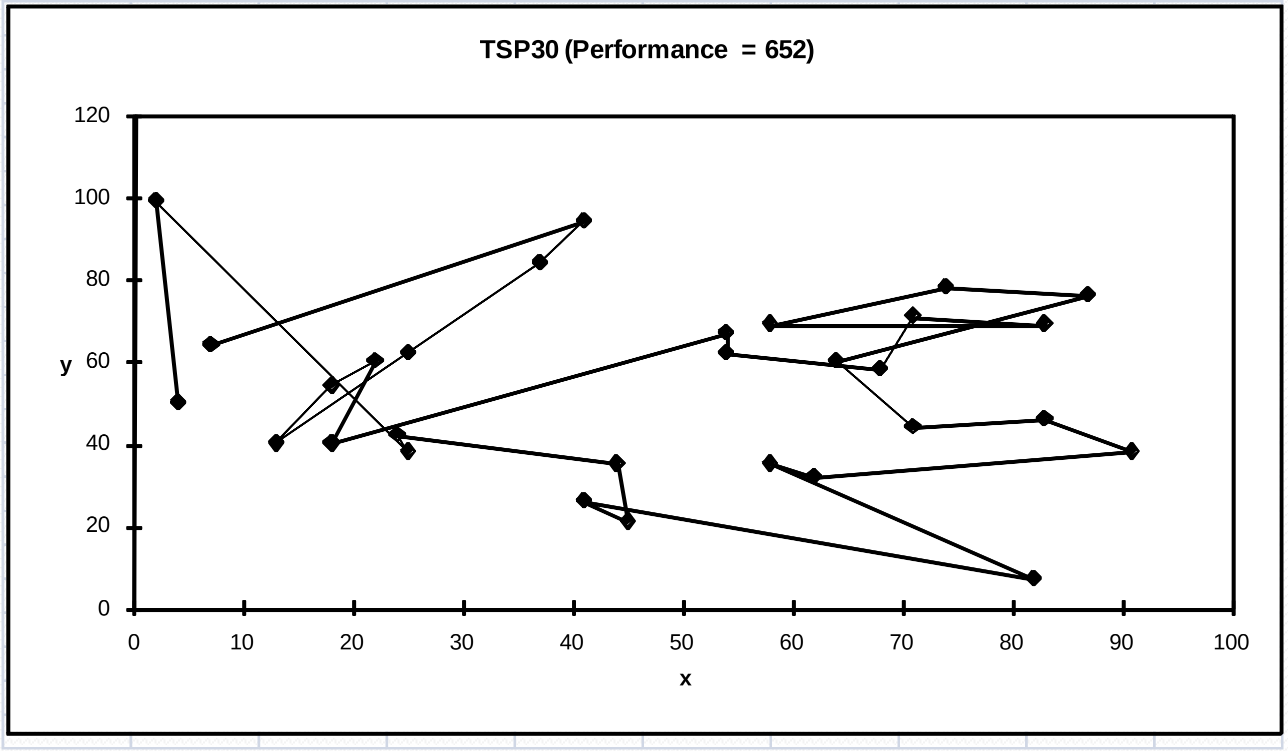
After around 7-8 generations



Example from: "Genetic Algorithms: A Tutorial" by Wendy Williams

Solution$_i$ (Distance = 652)

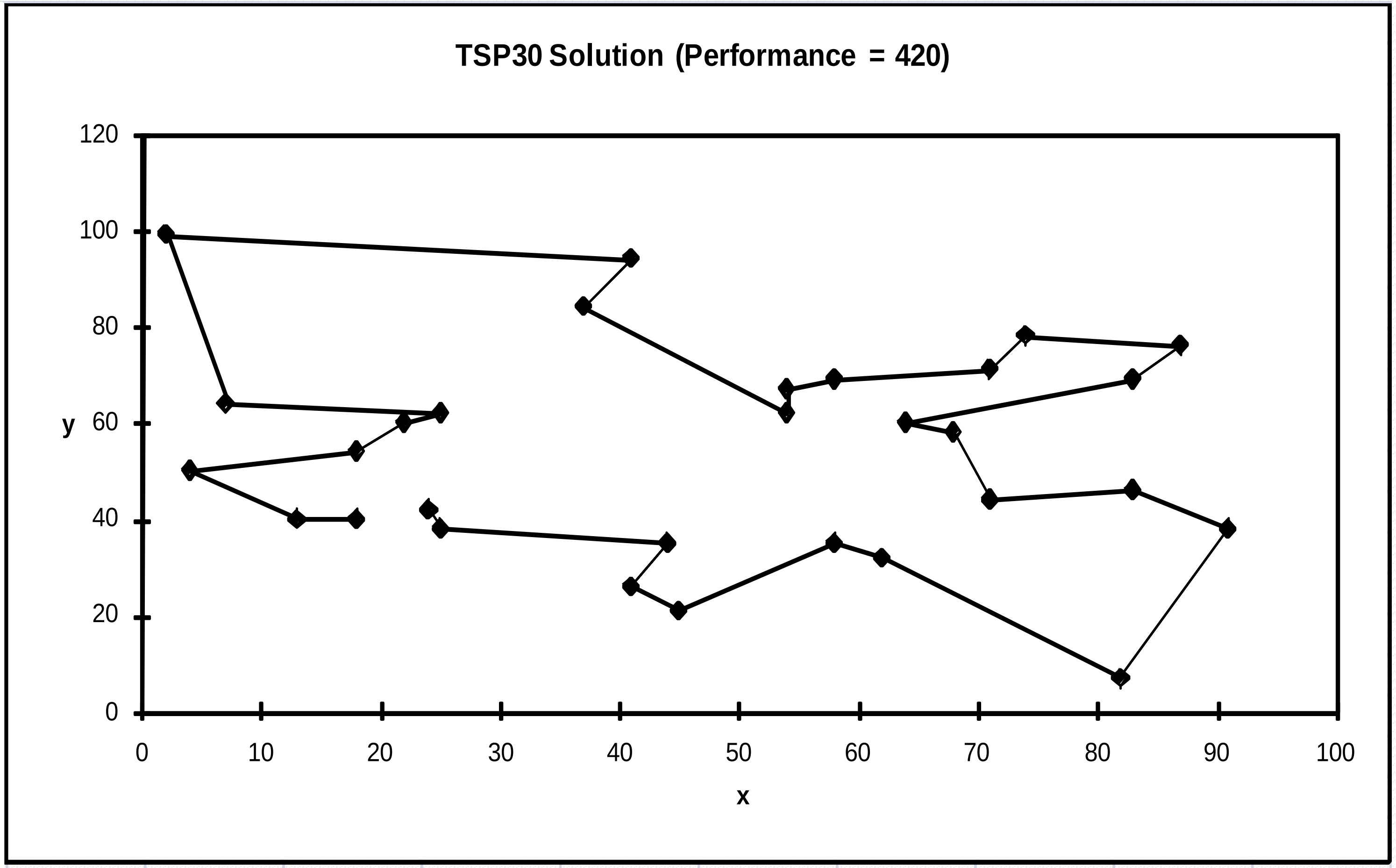
After around 10-11 generations



Example from: “Genetic Algorithms: A Tutorial” by Wendy Williams

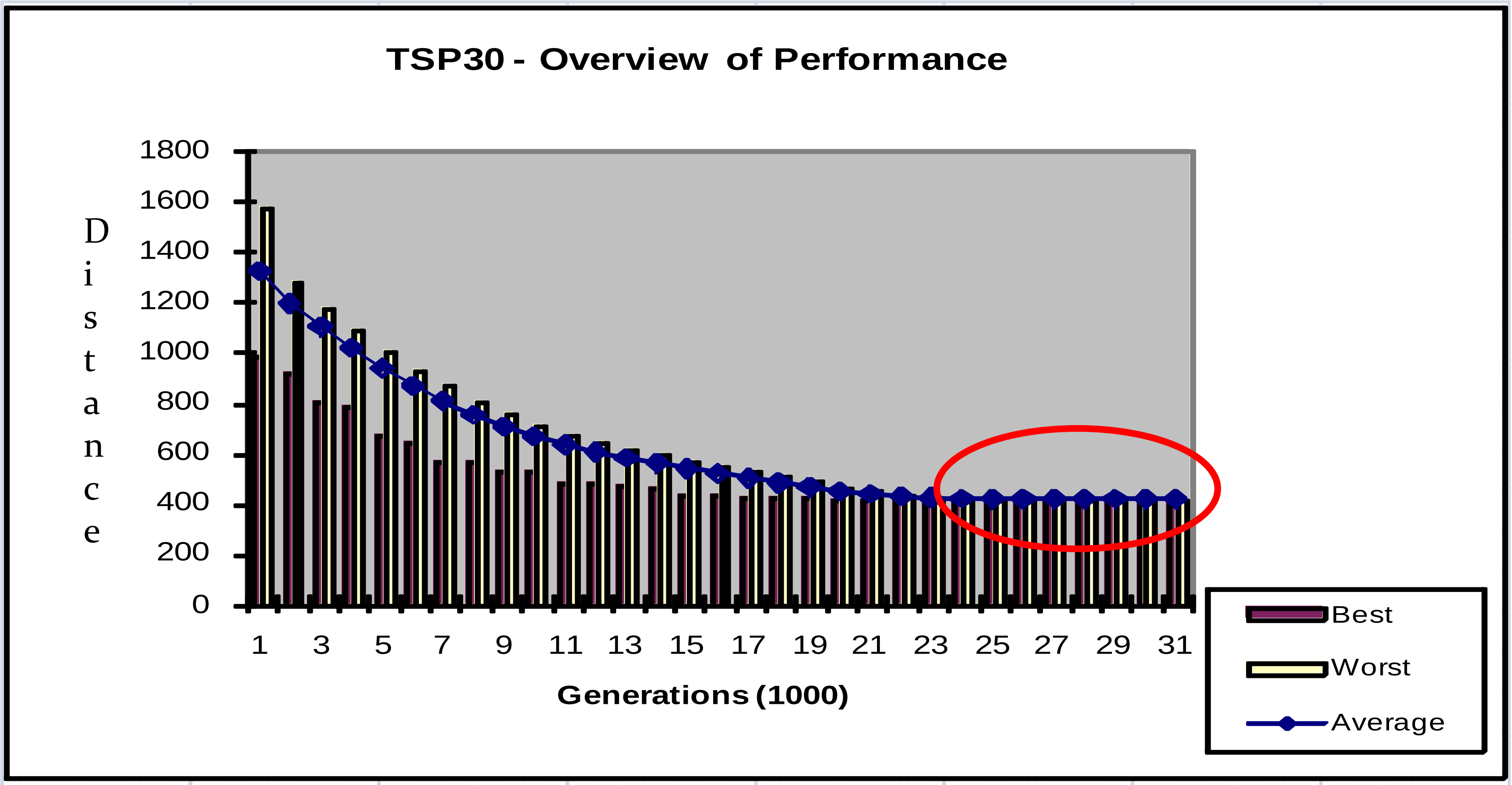
Solution$_i$ (Distance = 420)

After around 28-30 generations



Example from: “Genetic Algorithms: A Tutorial” by Wendy Williams

Overview of Performance

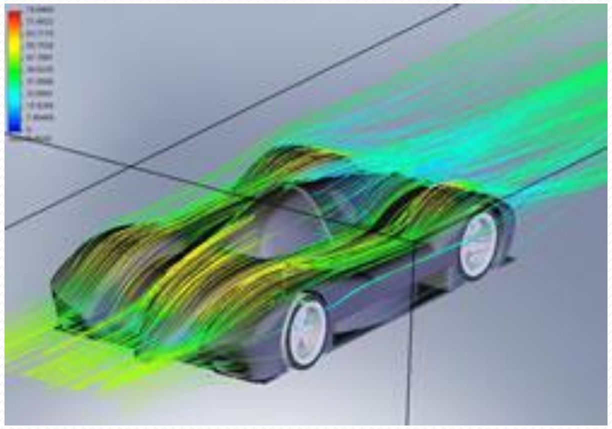


Example from: “Genetic Algorithms: A Tutorial” by Wendy Williams

OTHER APPLICATIONS

Automotive Design

- To identify combinations of best materials and best engineering to provide faster, lighter, more fuel efficient and safer vehicles for all the things we use vehicles for.



Bio-mimetic Invention

- GA programmers work on applications that not only analyze the natural designs themselves for a return on how they work, but can also combine natural designs to create something entirely new that can have exciting applications.

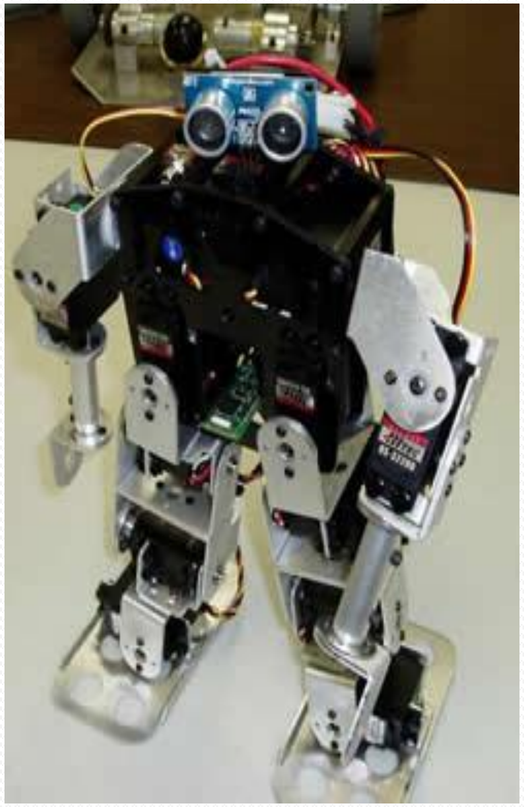
Reference:
http://brainz.org/15-real-world-applications-genetic-algorithms/

OTHER APPLICATIONS (contd.)

Robotics

- Generally a robot's design is dependent on the job it is intended to do, so there are many different designs possible.

- GAs can be programmed to search for a range of optimal designs and components for each specific use, and/or return results for entirely new types of robots that can perform multiple tasks and have more general application.

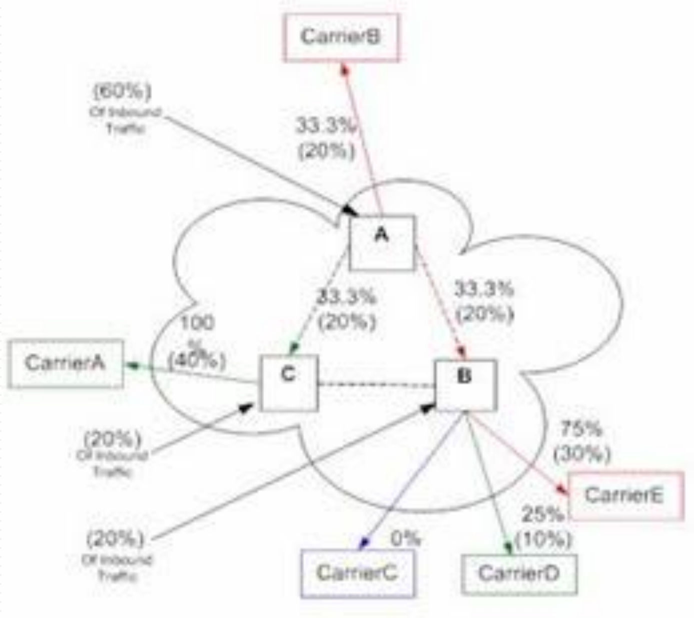


Reference:
http://brainz.org/15-real-world-applications-genetic-algorithms/

OTHER APPLICATIONS (contd.)

Telecommunications Routing

- GAs can be used to identify optimized routes within telecommunications networks.
- These could take notice of your system's instability and anticipate your re-routing needs.
- GAs are being developed to optimize placement and routing of cell towers for best coverage and ease of switching



Reference:
http://brainz.org/15-real-world-applications-genetic-algorithms/

OTHER APPLICATIONS (contd.)

Finance and Investment Strategies

- GAs can be used to explore different parts of the search space and produce solutions which potentially capture different patterns in the data
- GAs can also be used for noise filtering and achieve enhanced pattern detection for improving the overall learning accuracy

Altera Corp.
Rising Bottoms
Falling Tops
Resistance
Support
Resistance
Support
Volume
StockScore
Signal
Sentiment

Encryption and Code Breaking

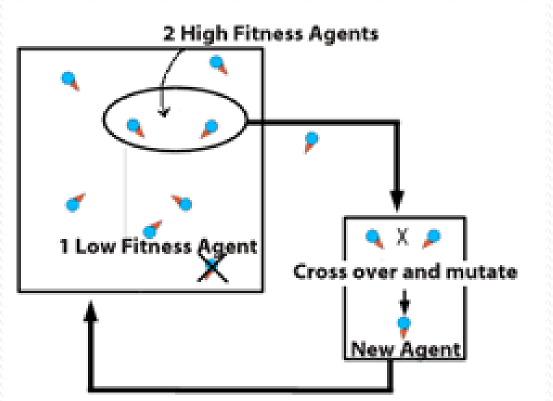
- GAs can be used both to create encryption for sensitive data as well as to break those codes.

Reference: http://brainz.org/15-real-world-applications-genetic-algorithms/

OTHER APPLICATIONS (contd.)

Computer Gaming

- GAs have been programmed to incorporate the most successful strategies from previous games - the programs 'learn' - and usually incorporate data derived from game theory in their design.



Marketing and Merchandising

- GAs are indeed being put to work to help merchandisers to produce products and marketing consultants design advertising and direct solicitation campaigns to sell stuff.

Reference:
http://brainz.org/15-real-world-applications-genetic-algorithms/

Advantages & Disadvantages

Advantages:

- Modular, separate from application
- Good for “noisy” environments
- Answer gets better with time
- Inherently parallel; easily distributed

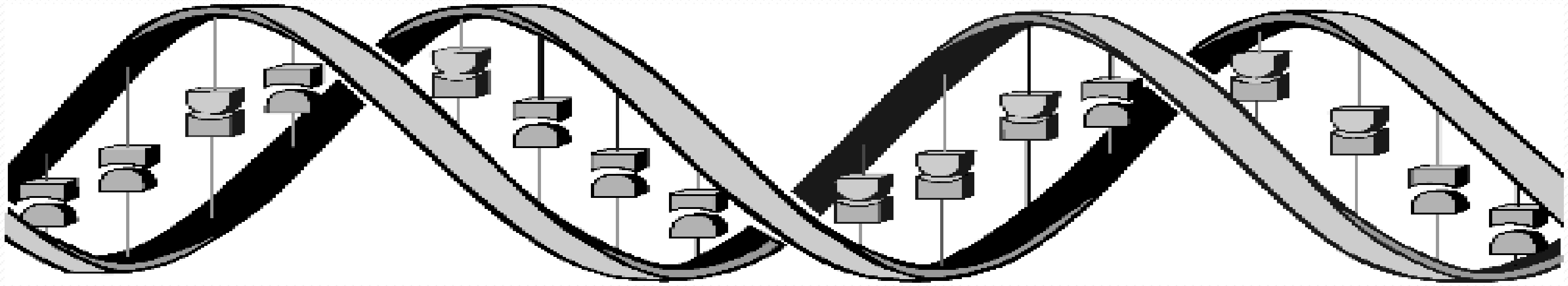
Disadvantages:

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

Citation “Wendy Williams Metaheuristic Algorithms, Genetic Algorithms a Tutorial”

Conclusion

GAs have been applied to a variety of function optimization problems and have been shown to be highly effective in searching a large, poorly defined search space even in the presence of difficulties such as high-dimensionality, multi-modality, discontinuity and noise.



--David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*

Thank You!!

Watch Demo :

1. Evolution of Mona Lisa: http://www.youtube.com/watch?v=S1ZPSbImvFE
2. Development of chromosomes : http://obitko.com/tutorials/genetic-algorithms/example-function-minimum.php

A Genetic Algorithm Based Approach to Data Mining

Ian W. Flockharta and Nicholas J Radclie
{iwf,njr}@quadstone.co.uk

Presented at "AAAI: Knowledge Discovery and DataMining 1996", Portland, Oregon

CSE 590 DATA MINING
Prof. Anita Wasilewska

<u>Presented by</u>
Gaurav Naigaonkar
Kumaran Shanmugam
Rucha Lale

Paper References

- S. Augier, G. Venturini, and Y. Kodrato, 1995. Learning first order logic rules with a genetic algorithm. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, Proceedings of the First International Conference on Knowledge Discovery and Data Mining. AAAI Press.
- Kenneth A. DeJong, William M Spears, and Diana F Gordon, 1993. Using genetic algorithms for concept learning. Machine Learning, 13:161 - 188.
- Ian W. Flockhart and Nicholas J. Radclie, 1995. GA-MINER: Parallel data mining with hierarchical genetic algorithms. Technical Report EPCC-AIKMS-GA-MINER-REPORT, Edinburgh Parallel Computing Centre.
- William J. Frawley, 1991. Using functions to encode domain and contextual knowledge in statistical induction. In Gregory Piatetsky-Shapiro and William J. Frawley, editors, Knowledge Discovery in Databases, pages 261 - 275. MIT Press.
- Attilio Giordana, Filippo Neri, and Lorenza Saiat, 1994. Search-intensive concept induction. Technical report, Univerita di Torino, Dipartimento di Informatica, Corso Svizzera 185, 10149 Torino, Italy.
- David Perry Green and Stephen F. Smith, 1993. Competition-based induction of decision models from examples. Machine Learning, 13:229-257.
- David Perry Greene and Stephen F. Smith, 1994. Using coverage as a model building constraint in learning classifier systems. Evolutionary Computation, 2(1).
- John H. Holland, 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press (Ann Arbor).
- John H. Holland, 1986. Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems. Machine Learning, an artificial intelligence approach, 2.

Types of Data Mining

- Undirected Data Mining (Pure Data Mining)
  - The kind of rule which is expected is not specified
  - Maximum freedom to identify patterns

  e.g. "Tell me something interesting about my data"
- Directed Data Mining
  - User asks for more specific information
  - Constraints imposed on the system

  eg: "Characterize my high spending customers"
- Hypothesis refinement
  - User specifies a hypothesis
  - System evaluates the hypothesis and refines it, if needed.

  eg: "I think that there is a positive correlation between sales of peaches and sales of cream: am I right"

Objective

- To design a system which can perform all three types of data mining tasks using Parallel Genetic Algorithms and produces explicit rules for maximum comprehensibility

- The System is named 'GA-MINER'

Reproductive Plan Language

- GA-MINER is implemented using Reproductive Plan Language [RPL2]
- For Stochastic Search Algorithms with special emphasis on evolutionary algorithms such as GA
- Features pertinent to GA-MINER
  - Automatic parallelism
  - Arbitrary representations (important for rules)
  - Large library of functions

Pattern Representation

- Patterns represented using subset description
- Subset descriptions
  - Clauses used to select subsets of databases
  - Units used by Genetic Algorithm
  - consist of disjunction of conjunction of attribute value or attribute range constraints

```
Subset Description   =: Clause {or Clause}
Clause               =: Term and {Term}
Term                 =: Attribute in Value Set
                        | Attribute in Range
```

Pattern Representation (contd.)

- Patterns supported by GA-MINER
  - Explicit Rule Patterns

    (e.g) if C then P

    where C, P are subset descriptions representing the condition and prediction respectively

  - Distribution Shift Patterns

    (e.g) *The distribution of A when C and P*

    The distribution of A when C

    A is the hypothesis variable, C and P are subset descriptions

  - Correlation Patterns

    (e.g) when C the variables A and B are correlated

    A and B are hypothesis variables and C is a subset description

Pattern Templates

- Used to constrain the system to particular forms of patterns i.e., to allow certain features of the pattern to be restricted
- Can be regarded as an initial genome upon which all other genomes in the population are based
- Component parts of Pattern Templates can be marked
  - Fixed
    - inherited by every pattern in the population
    - NEVER modified by crossover or mutation
  - Initialized
    - appear in all generated patterns
    - MAY BE modified

Pattern Templates (contd.)

- Undirected mining
  - Performed with a minimal template
- Directed mining
  - Performed by restricting the pattern
- Hypothesis refinement
  - Performed by seeding the initial population with patterns based on the template and
  - Random components
  - Search can modify these patterns

The Genetic Algorithm in GA-MINER

- Structured population model
- Reproductive partners are selected from the same neighborhood to improve diversity
- Also to identify several patterns in a single run
- Crossover is performed at the different levels
    - Subset description
    - Clause – both uniform and single point crossover
    - Term – uniform crossover
- Mutation also done at various levels
    - With separate probability for mutating each of the component parts
- The population is updated using a heuristic like replacing the lowest fit

Crossover

- **Subset Description Crossover**
  - Each clause in the first parent is crossed with the clause in the corresponding position in the second parent.
  - For each clause, we use uniform clause crossover with probability rPUCross and single-point crossover with probability (1–rPUCross).
- e.g. **A : *Clause A1 or Clause A2 or Clause A3 or Clause A4***

  B : *Clause B1 or Clause B2*

Crossover (contd.)

- **Uniform Clause Crossover**
  - performs an "alignment" of terms between the two parent clauses
  - terms concerning the same variable will be crossed with each other
- e.g. A : *Age = 20 .. 30*

  B : *Sex = M and Age = 0 .. 25*

  After alignment, we get

  A : *Age = 20 .. 30*

  B : *Sex = M and Age = 0 .. 25*

http://citeseer.ist.psu.edu/cache/papers/cs/3487/http:zSzzSzwww.quadstone.co.ukzSz~ianzSzaikmszSzreport.pdf/flockhart95gaminer.pdf

Crossover (contd.)

- **Single-Point Clause Crossover**
  - **Alignment of clauses**
  - **Selection of crossover point**

e.g.

A : *Age = 20 .. 30 and Height = 1.5 .. 2.0*

B : *Sex = M and Age = 0 .. 25*

After alignment, we get

A : *Age = 20 .. 30 and Height = 1.5 .. 2.0*

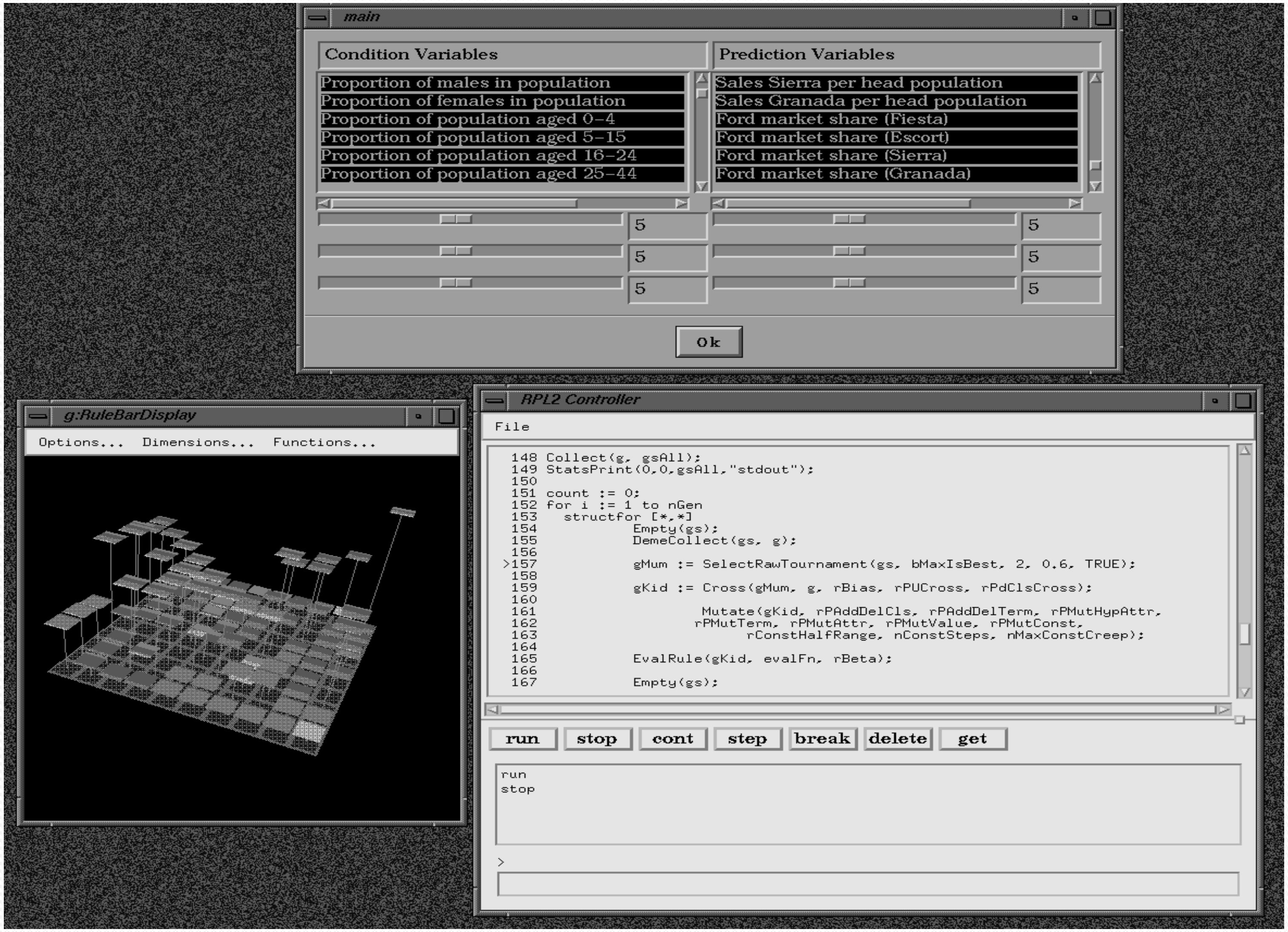
B : *Sex = M and Age = 0 .. 25*

Result: C : *Age = 0 .. 25*

Crossover (contd.)

- **Term Crossover**
  - to combine two terms concerning the same variable
  - Crossover of *value and range* terms is handled differently.

GA-Miner GUI


main
Ok
g:RuleBarDisplay
Options... Dimensions... Functions...
RPL2 Controller
File
run stop cont step break delete get

Condition Variables

- Proportion of males in population
- Proportion of females in population
- Proportion of population aged 0–4
- Proportion of population aged 5–15
- Proportion of population aged 16–24
- Proportion of population aged 25–44

Prediction Variables

- Sales Sierra per head population
- Sales Granada per head population
- Ford market share (Fiesta)
- Ford market share (Escort)
- Ford market share (Sierra)
- Ford market share (Granada)

5 5 5 5 5 5

```
Collect(g, gsAll);
StatsPrint(0,0,gsAll,"stdout");

count := 0;
for i := 1 to nGen
  structfor [*,*]
        Empty(gs);
        DemeCollect(gs, g);

        gMum := SelectRawTournament(gs, bMaxIsBest, 2, 0.6, TRUE);

        gKid := Cross(gMum, g, rBias, rPUCross, rPdClsCross);

                Mutate(gKid, rPAddDelCls, rPAddDelTerm, rPMutHypAttr,
               rPMutTerm, rPMutAttr, rPMutValue, rPMutConst,
                     rConstHalfRange, nConstSteps, nMaxConstCreep);

        EvalRule(gKid, evalFn, rBeta);

        Empty(gs);
```

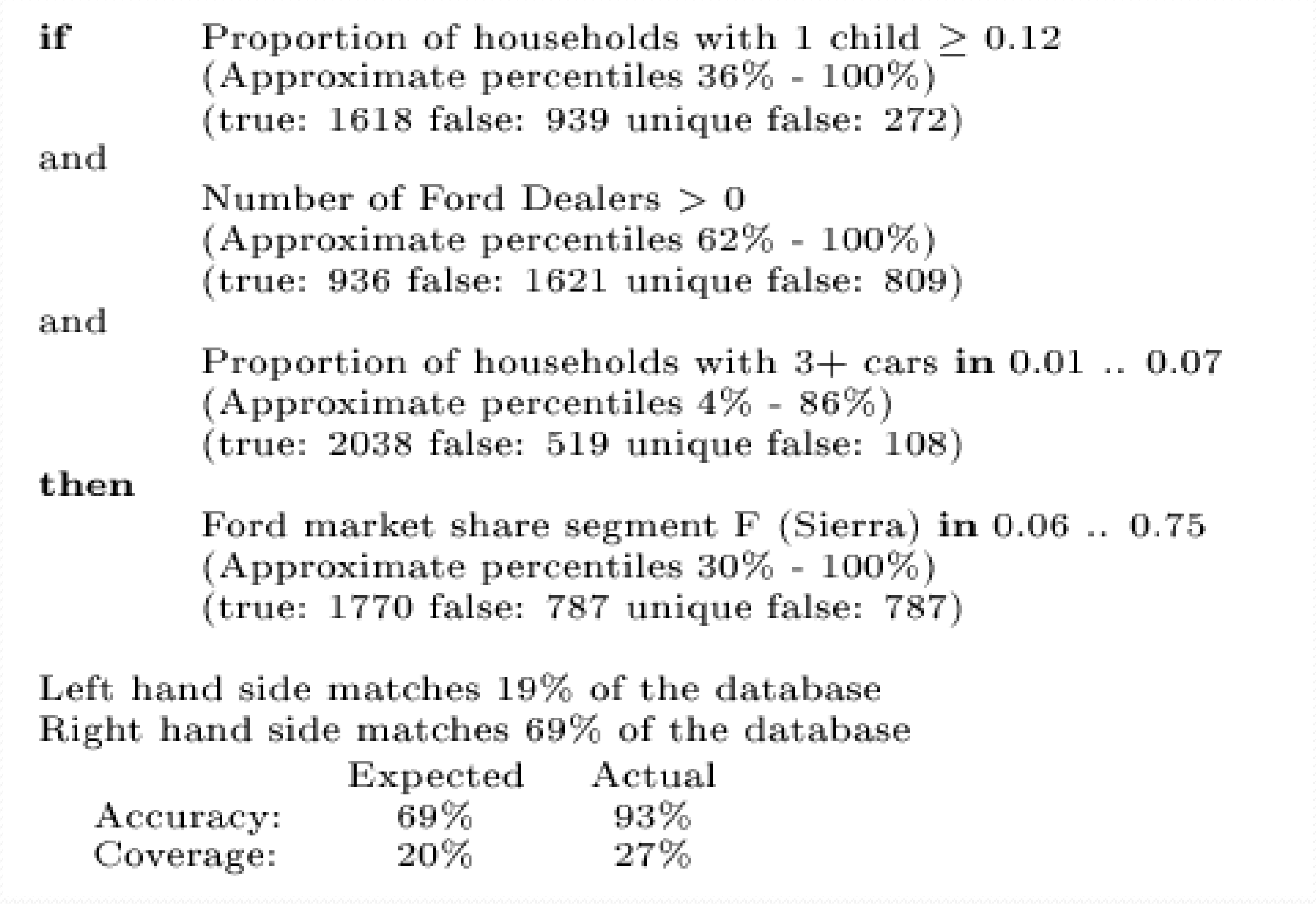
```
run
stop
```

```
>
```

http://citeseer.ist.psu.edu/cache/papers/cs/3487/http:zSzzSzwww.quadstone.co.ukzSz~ianzSzaikmszSzreport.pdf/flockhart95gaminer.pdf

Examples of discovered patterns

Explicit Rule Pattern



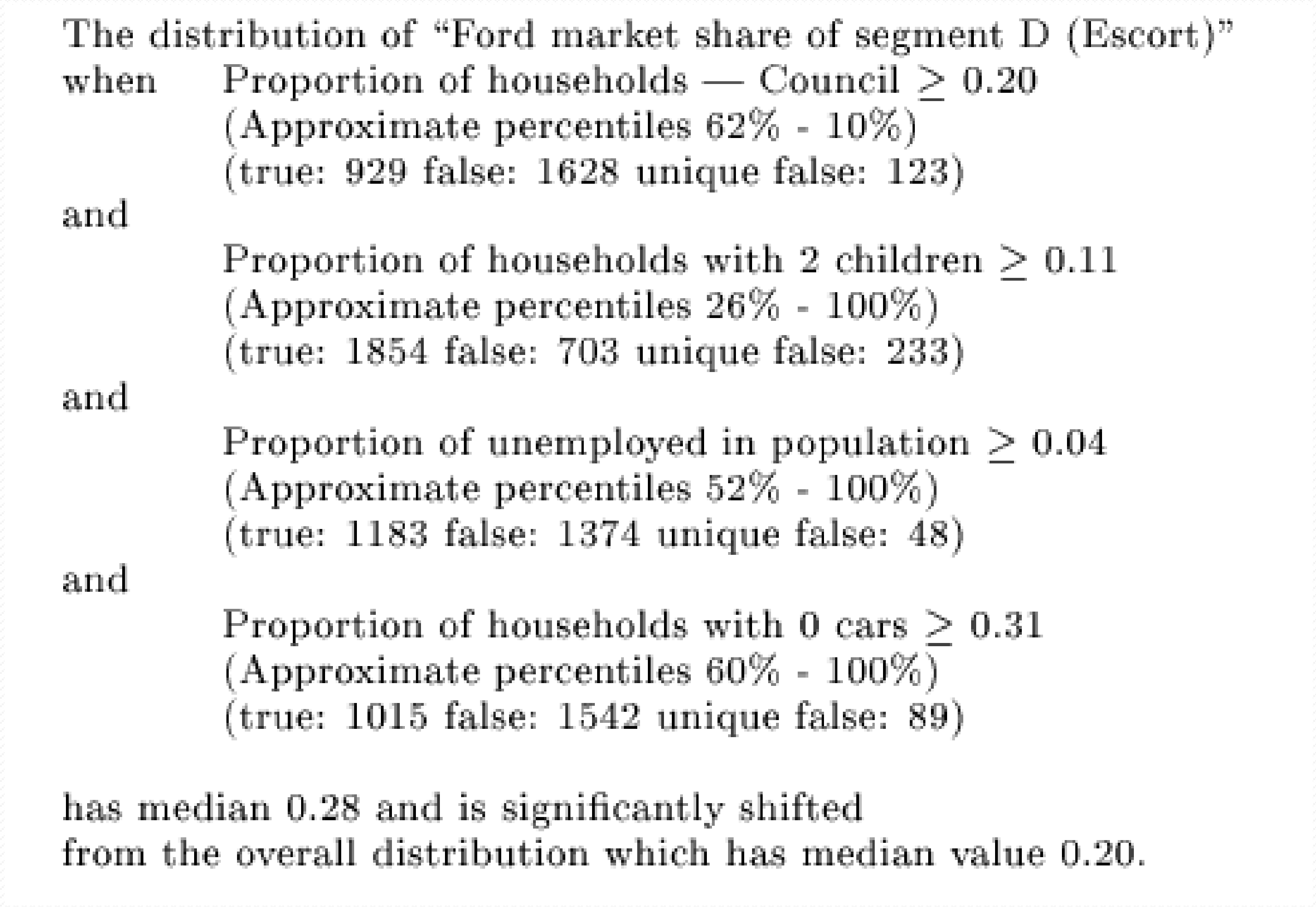
```
if      Proportion of households with 1 child ≥ 0.12
        (Approximate percentiles 36% - 100%)
        (true: 1618 false: 939 unique false: 272)
and
        Number of Ford Dealers > 0
        (Approximate percentiles 62% - 100%)
        (true: 936 false: 1621 unique false: 809)
and
        Proportion of households with 3+ cars in 0.01 .. 0.07
        (Approximate percentiles 4% - 86%)
        (true: 2038 false: 519 unique false: 108)
then
        Ford market share segment F (Sierra) in 0.06 .. 0.75
        (Approximate percentiles 30% - 100%)
        (true: 1770 false: 787 unique false: 787)
```

Left hand side matches 19% of the database
Right hand side matches 69% of the database

	Expected	Actual
Accuracy:	69%	93%
Coverage:	20%	27%

Examples of discovered patterns (contd.)

<u>Distribution shift Pattern</u>



The distribution of "Ford market share of segment D (Escort)"
when Proportion of households — Council ≥ 0.20
(Approximate percentiles 62% - 10%)
(true: 929 false: 1628 unique false: 123)
and
Proportion of households with 2 children ≥ 0.11
(Approximate percentiles 26% - 100%)
(true: 1854 false: 703 unique false: 233)
and
Proportion of unemployed in population ≥ 0.04
(Approximate percentiles 52% - 100%)
(true: 1183 false: 1374 unique false: 48)
and
Proportion of households with 0 cars ≥ 0.31
(Approximate percentiles 60% - 100%)
(true: 1015 false: 1542 unique false: 89)

has median 0.28 and is significantly shifted
from the overall distribution which has median value 0.20.

Conclusion

- GA's application towards pattern discovery in addition to classification and concept learning
- GA – well suited for undirected mining
- Rules produced by GA-MINER is more understandable than those produced by other unsupervised methods like Neural Networks
- Comprehensibility of the rules is IMPORTANT as it is widely used by non-experts

THANK YOU!