Comparative Analysis of Genetic Algorithm Implementations

Robert Soricone
Dr. Melvin Neville

Department of Computer Science
Northern Arizona University
Flagstaff, Arizona

Outline

- Introduction to Genetic Algorithms (GA)
- Problems solved with GA in Ada95
- GA toolkit for Matlab
- Compare implementation and performance
- Refactoring the Ada95 GA
- Application using the Ada95 GA

Genetic Algorithms

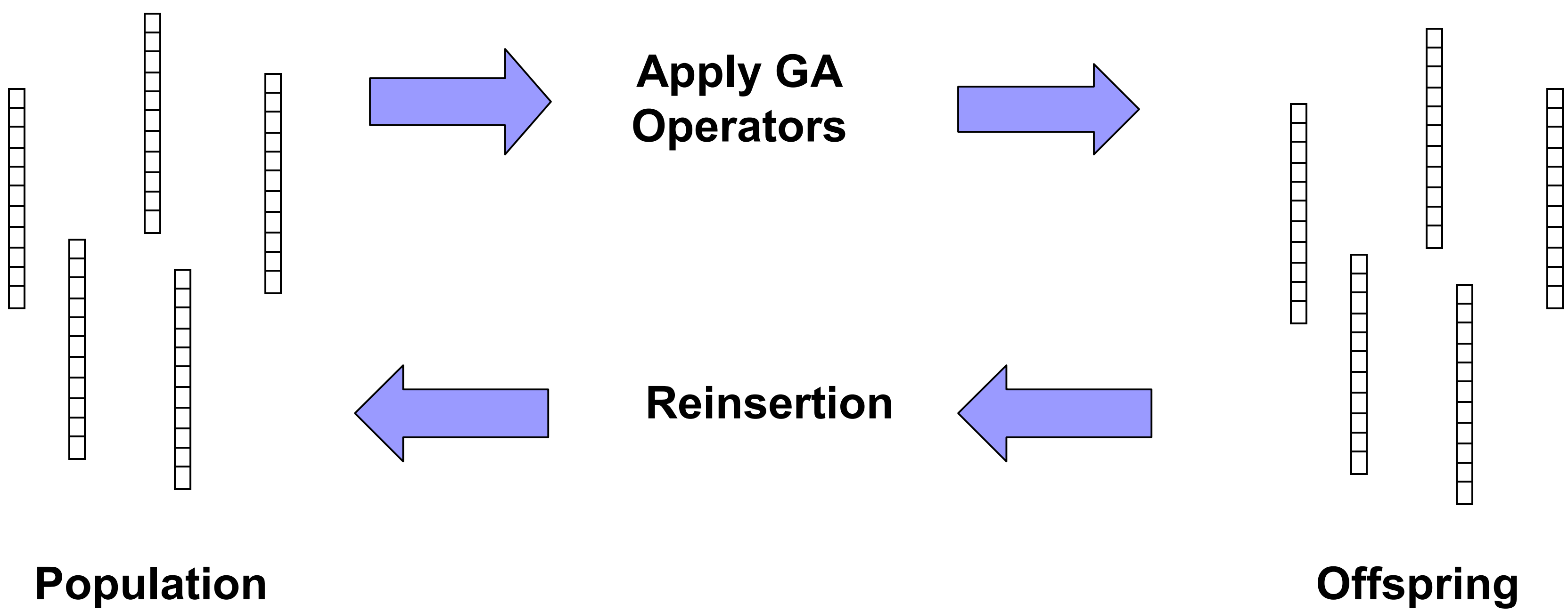
- Search method based on principles of evolution and heredity
- Apply principle of Survival of the Fittest to a set of possible solutions
- Goal: produce (hopefully) produce better approximations to solution
- GA process leads to evolution of solutions

Chromosomes

- Encodes a solution to the problem
- Classical GA uses bit strings
  - Alternatives: real values, permutation, decision trees

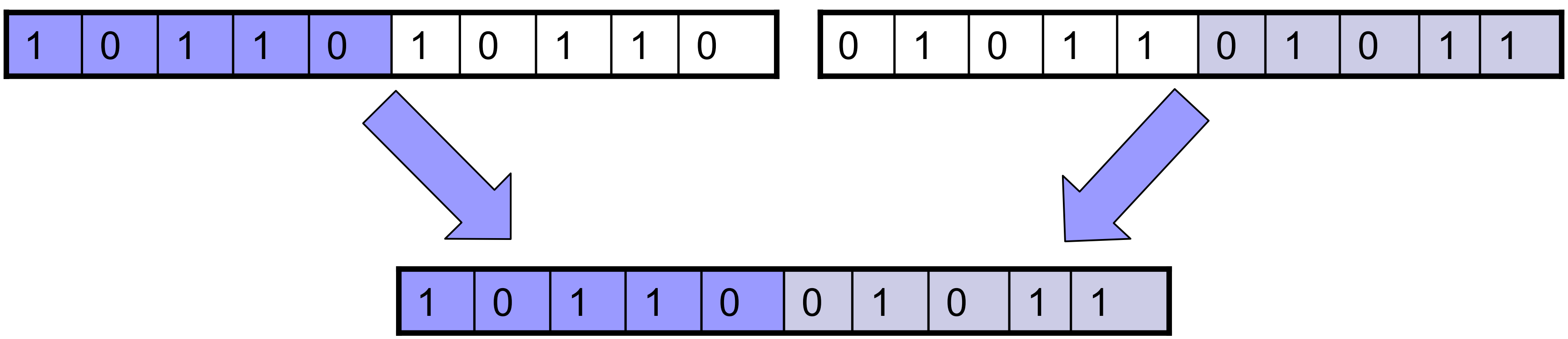
1	0	1	1	0	0	1	0	1	0

Genetic Algorithms



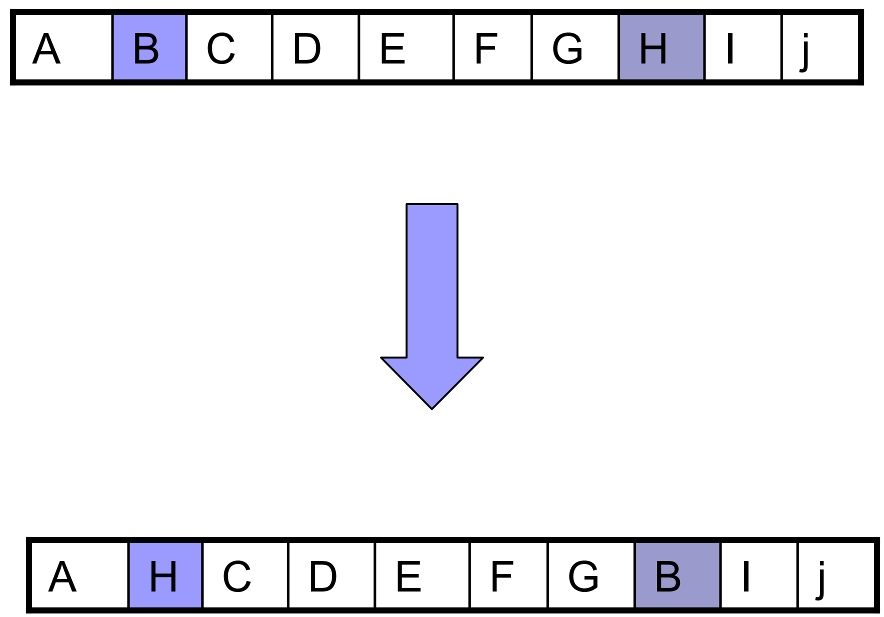
Genetic Operators

- Crossover
  - Exchange information between selected parents
  - Recombined genetic material used to produce offspring
  - Number of segments and length of segments depends on crossover implementation
  - Example: single point crossover



Genetic Operators

- Mutation
  - Compliments the crossover operation
  - Example: Arbitrary swapping of 2 or more values within the chromosome
  - Help introduce variability into the population



Structure of a GA

```
Procedure Genetic_Alg is
Begin
   generation = 0
   initialize Population
   evaluate Population
   loop
        generation ++
        select individuals
        apply genetic operators
        evaluate new individuals
        replace original population
        exit when criteria met
   end loop
End Genetic_Alg
```

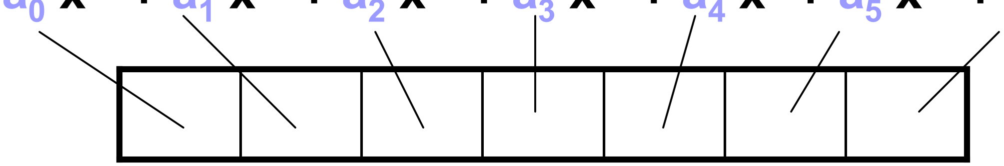
Evolution of Coefficients

- Coefficients of the power series that represents the sine function are evolved.
- Beginning with the fundamental series:
  - **$\sin(x) = a_0 x^0 + a_1 x^1 + a_2 x^2 + a_3 x^3 + a_4 x^4 + \ldots$**
- Actual answer is known to be:
  - **$\sin(x) = x^1 /1! - x^3 /3! + x^5 /5! - x^7/7! + \ldots$**

Evolution of Coefficients

- Chromosomes encoded with long float
- Crossover accomplished by swapping of sequences between chromosomes
- Mutation implemented by increasing/decreasing random array index values
- Led to functions that could perform within 0.25% of the actual sine values in the range 0 – 90 degrees

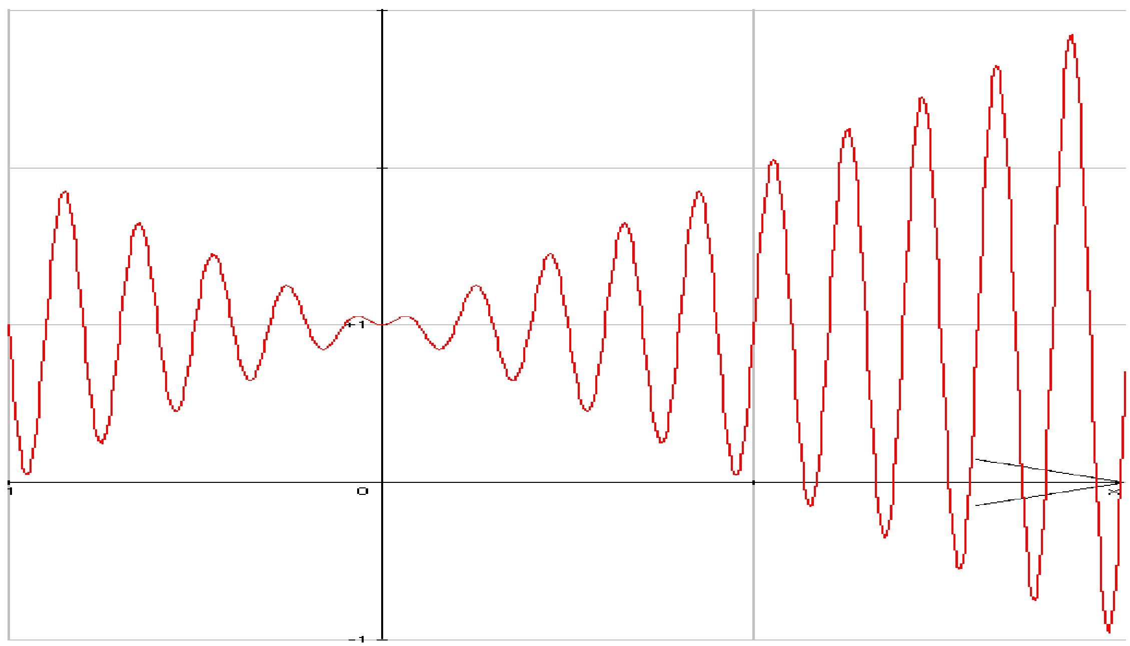
$$\sin(x) = a_0 x^0 + a_1 x^1 + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6$$



Chromosome

Maximization of a Function

- Maximization of a function of one variable
- Function:
  - $f(x) = x * \sin(10 * \pi * x) + 1.0$



Maximization of a Function

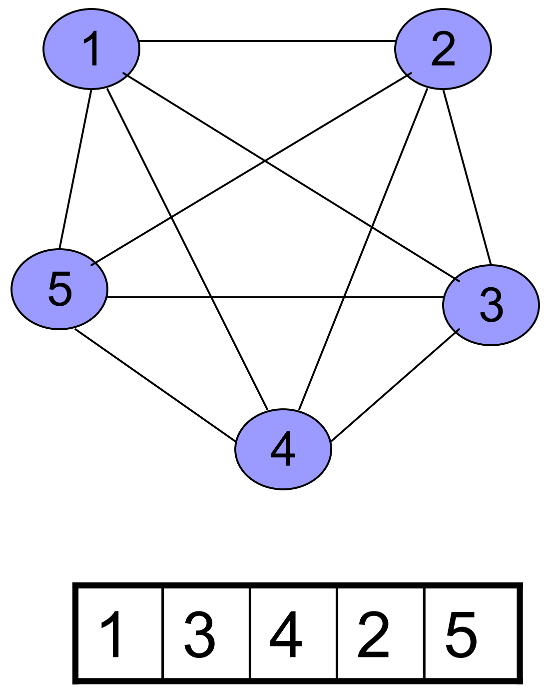
- Chromosomes encoded as bit strings
- Crossover is a swap of two equal-length substrings between two chromosomes
- Mutation is the flipping of the gene's value from one binary value to the other
- problem was immediately and accurately solved

Traveling Salesman Problem

- TSP starts with a graph of fully interconnected nodes or towns, where the distances between the towns are measured (a “weighted graph”)
- The problem lies in finding the best route in which each town is visited once and only once in a tour which begins and ends with the same node
- “Best” here is in the sense of lowest total distance measured along the final route

TSP Chromosome Encoding

- Natural value correspond to node ID
- Represents a particular tour of the given graph
- "Order" crossover operator
- Mutation: swapping random elements in chromosome



Chromosome

TSP: Chromosome Value

Chromosome

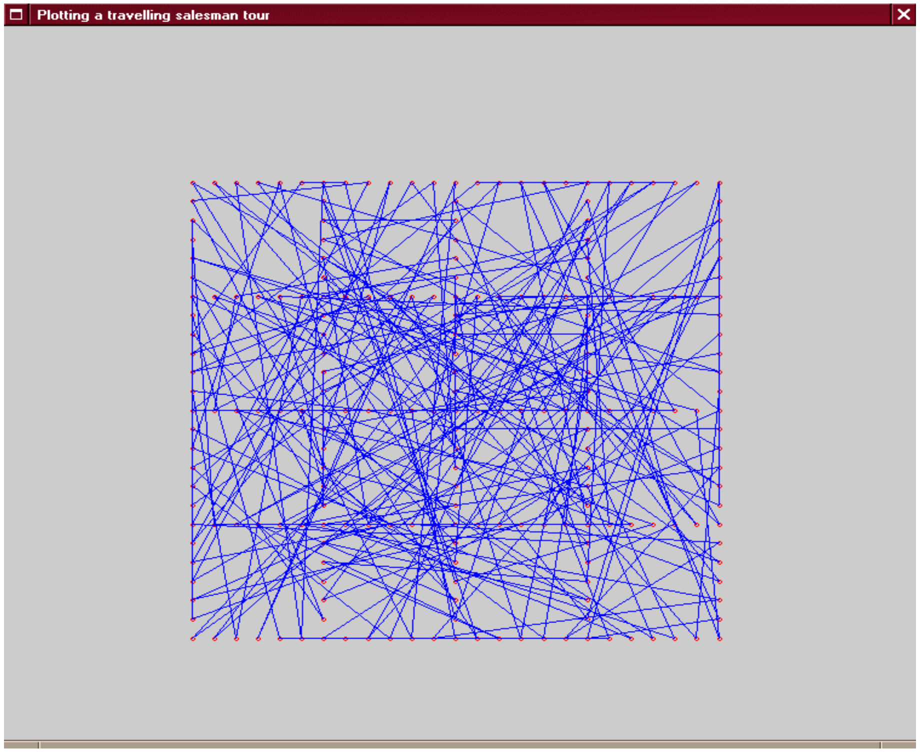
1	3	4	2	5

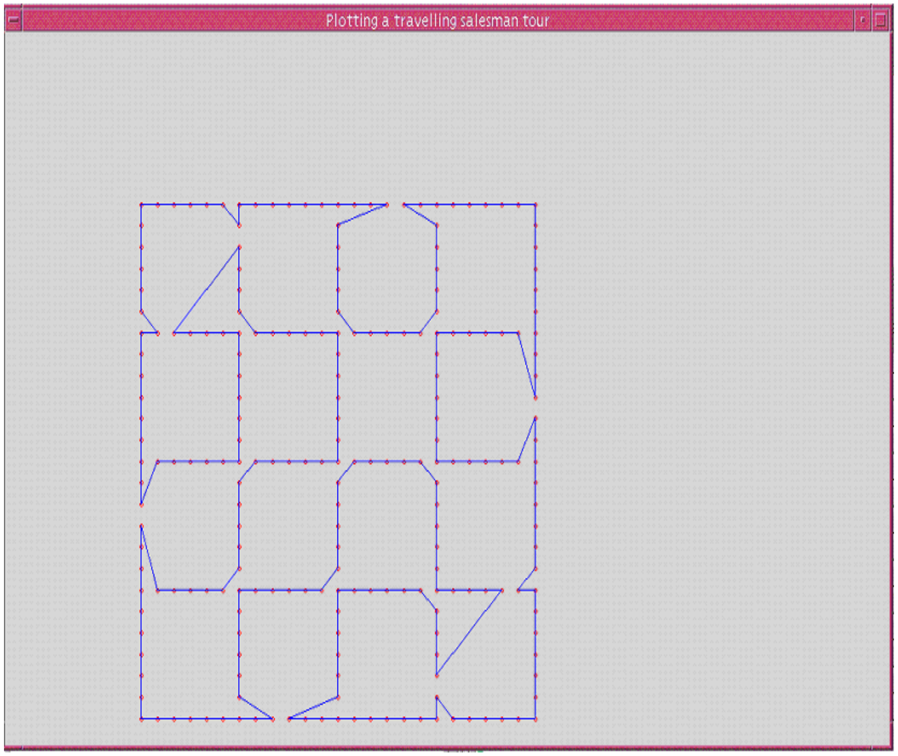
Distance Matrix

	1	2	3	4	5
1	1.023	.984	.343	.651	.876
2	1.923	.586	.344	.658	.874
3	3.023	.922	.347	.656	.876
4	3.023	1.986	1.343	.652	.873
5	65.23	.1986	6.343	.633	.849

Value of Chromosome: Sum of distances along path

TSP: 225 Node Tour

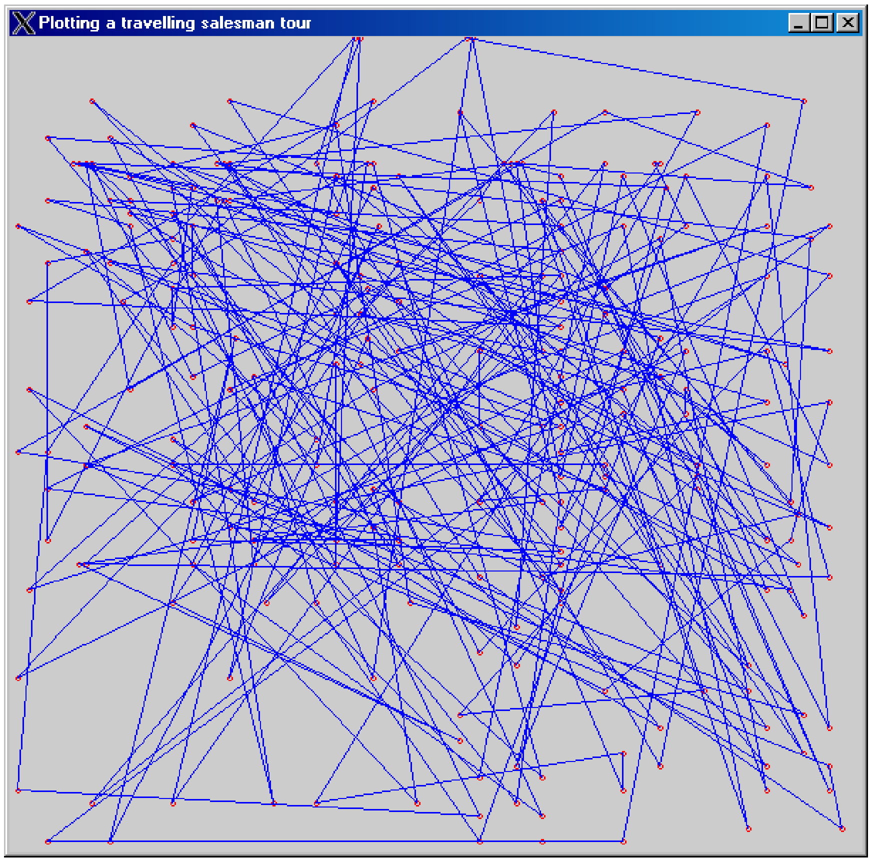




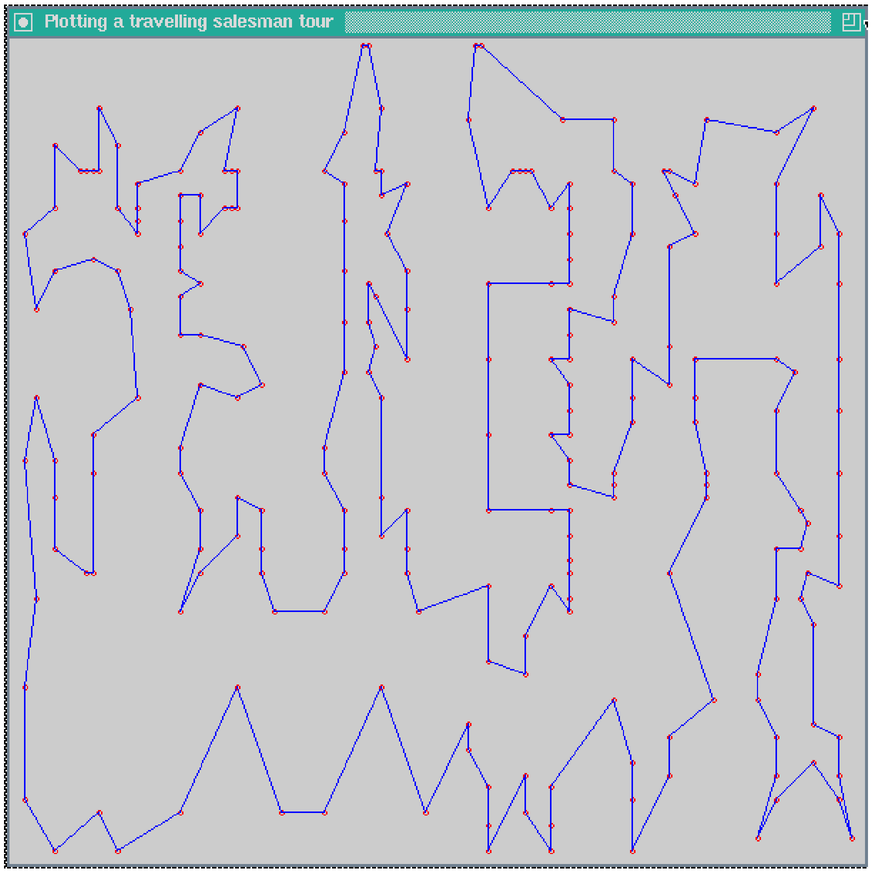
Starting tour length: 1,575,904.000000

Final tour length: 126445.9336

TSP: 237 Node Tour

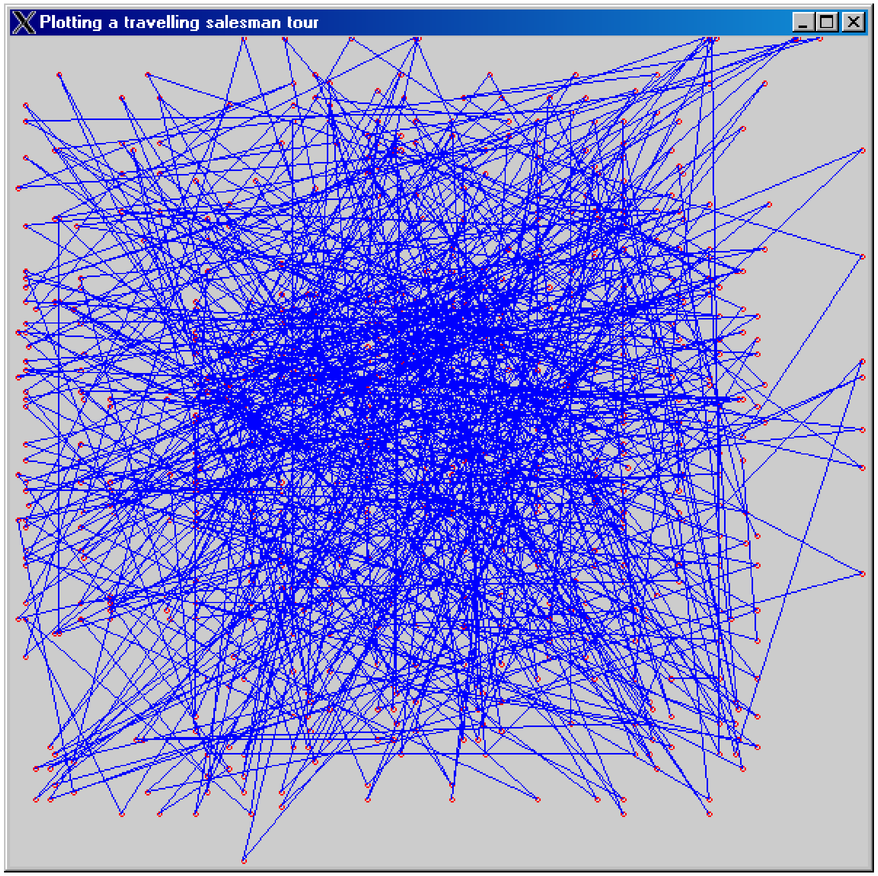


Starting tour length: 11969.209961

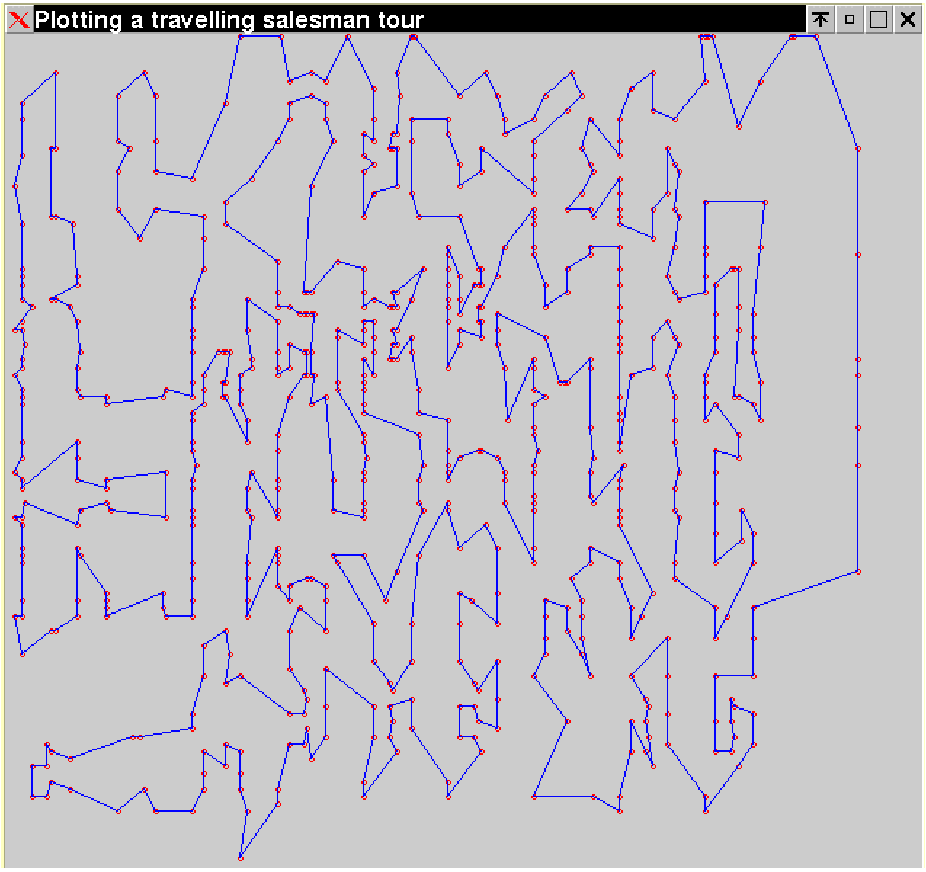


Final tour length: 1069.562134

TSP: 662 Node Tour



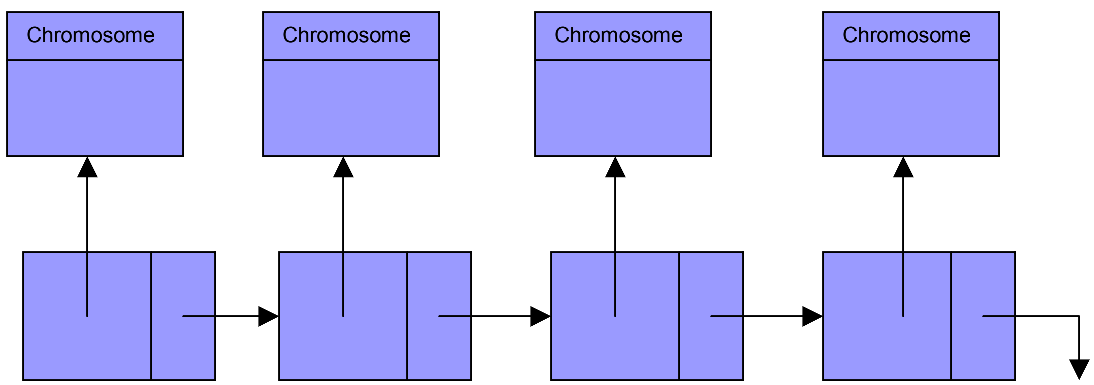
Starting tour length: 53434.769531



Final tour length: 2729.069580

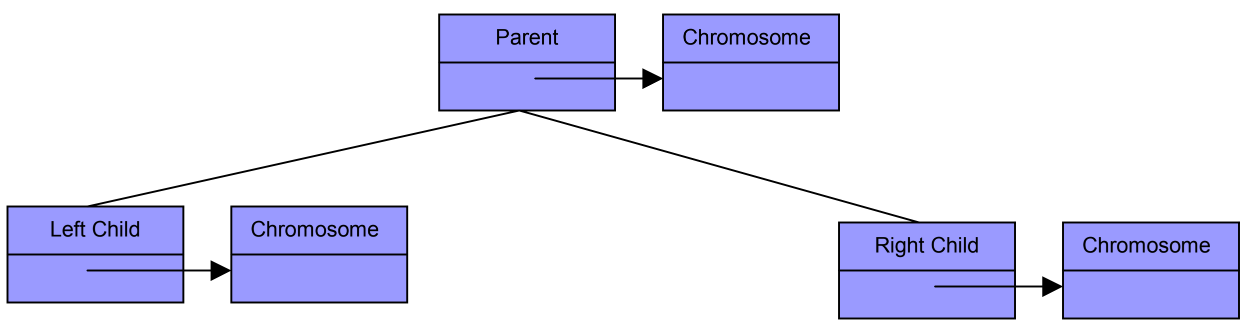
GA in Ada95: Main Data Structures

- Population: Linked list of list nodes
- Each node holds an access to chromosome type
- Members selected from list processed by genetic operators



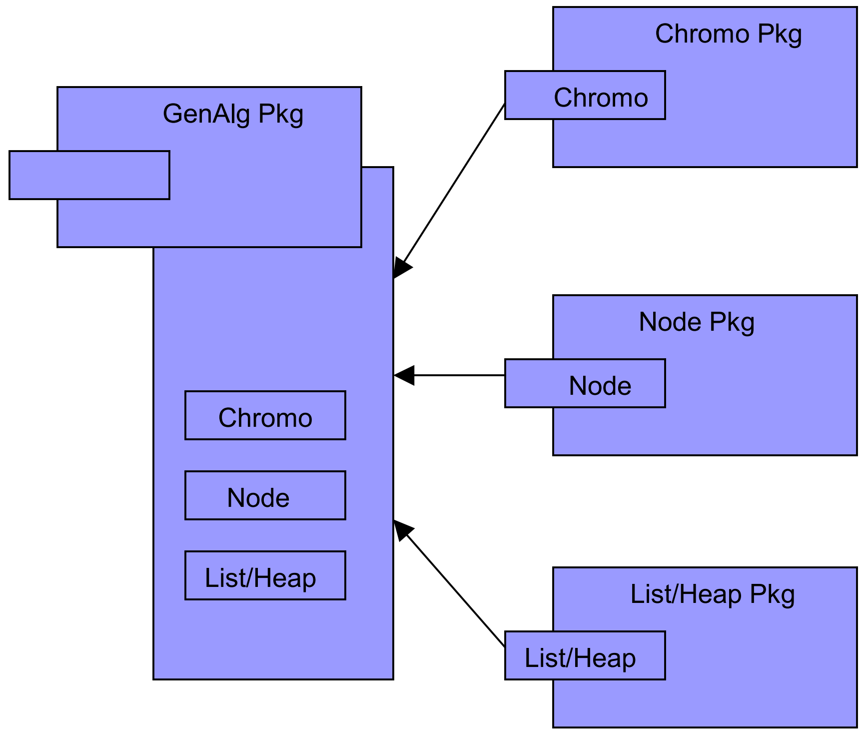
GA in Ada95: Main Data Structures

- Priority queue (binary heap) holds new offspring
- Sort key for the priority queue is the value of the chromosome
- Provide convenient access to best performers of the generation



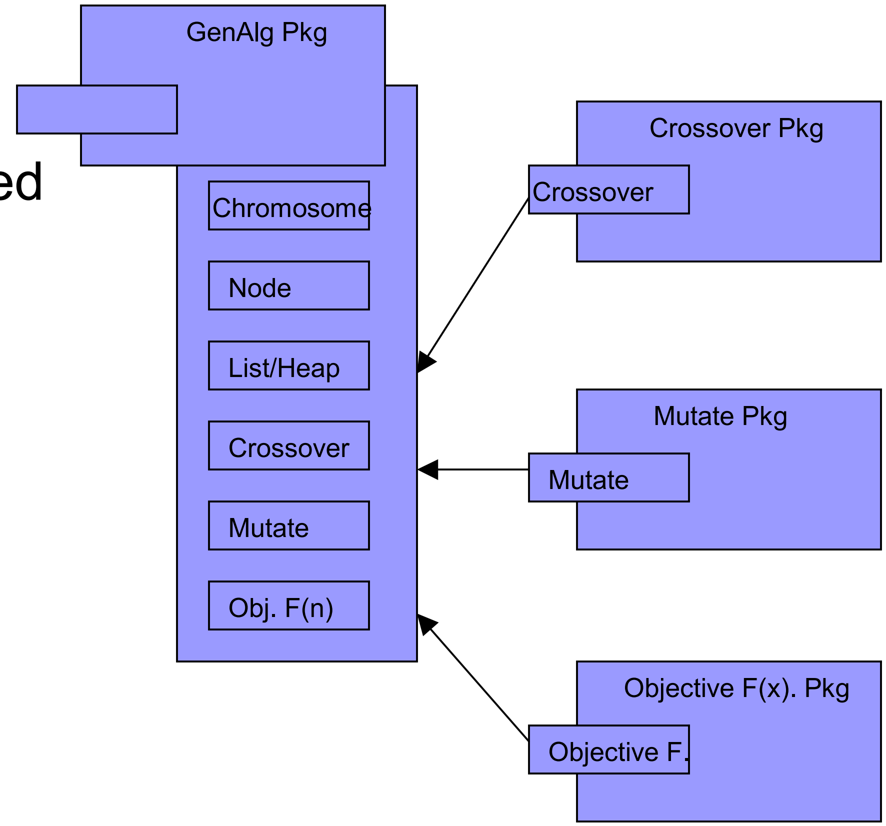
Core Packages

- Formal package allows for the composition of multiple generic packages
- One package is used as a parameter of another package to maintain a consistent hierarchy
  - Chromosome is generic with respect to array type (chromosome encoding)
  - Nodes for the list and heap are instantiated with chromosome package
  - Lists/heap instantiated with the new nodes
  - GA instantiates all of the above packages



Problem-Specific Packages

- Crossover and mutation packages are also instantiated
- Crossover and mutation operators are tightly coupled with interpretation of chromosome
- Current design requires rewriting operators for new problems
- Search for GA "toolkit" that may simplify changing problems



Matlab

- Numerical scripting language
- Part of integrated computing environment: graphics, visualization tools and the Matlab scripting language
- Data structures & operations pre-made for matrices, linear algebra
- Toolboxes added later for specific problem domains

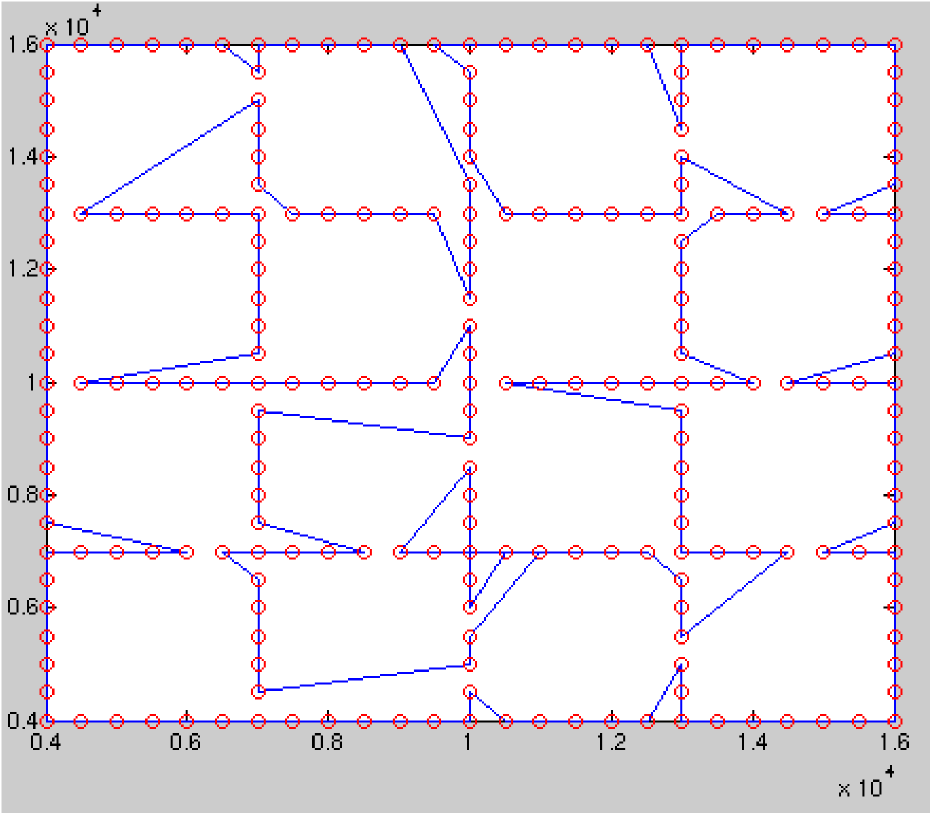
Natural Selection TSP Demo

- www.natural-selection.com
- Example of using GA to solve TSP
- Population, node distances held in matrix
- Chromosomes encoded with integer values for node ID

225 Node Graph Comparison

Distance from optimal	Ada95 GA	Matlab GA
Less than 15%	40	22
Less than 10%	40	8
Less than 5%	21	1
Less than 1%	3	0
Optimal	1	0

TSP Demo: 225 Node Tour



Final tour length: 130480.0

Matlab GA Toolbox

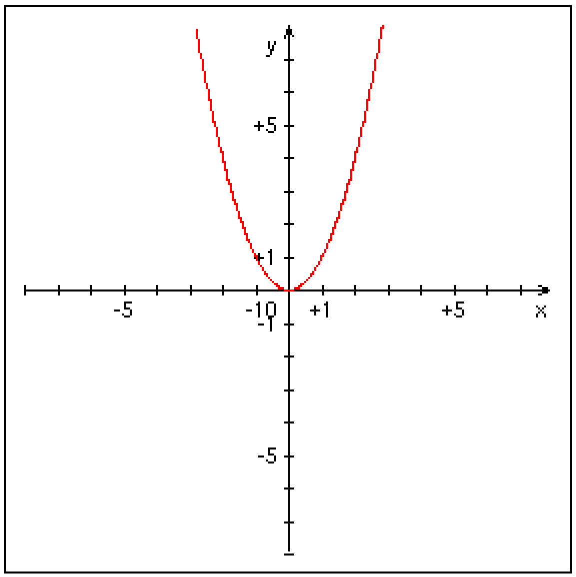
- University of Sheffield
  - http://www.shef.ac.uk/~gaipp/ga-**toolbox**/
- Collection of m-files (Matlab script) which implement functions in a GA
- Toolbox developed for control engineering applications
- Provides framework for experimenting with GA

Toolbox Data Structures

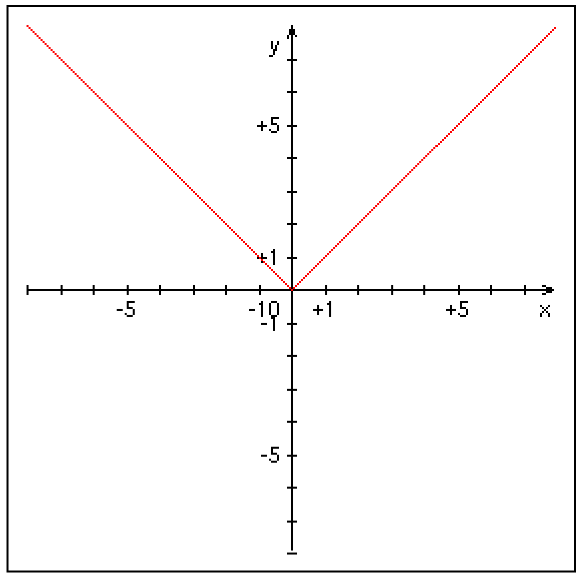
- Matlab essentially supports only one data type: rectangular matrix of real or imaginary numbers
- Chromosome population contained in matrix, each row corresponding to unique individual

$$\begin{bmatrix} \text{Chromosome 1} \\ \text{Chromosome 2} \\ . \\ . \\ . \\ \text{Chromosome n} \end{bmatrix}$$

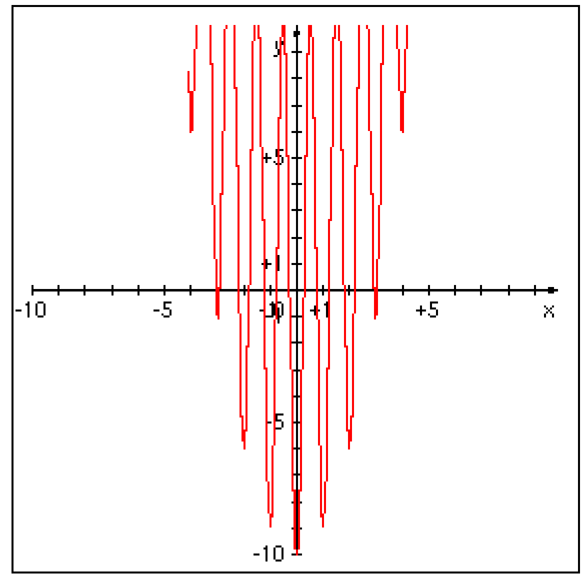
Test Functions



$f(x) = x^2$



$f(x) = |x|$



$f(x) = x^2 - 10*Cos(2\pi x)$

Test Function Results

GA	Avg. Bouts/SD	Avg. Time/SD
#1: Ada #1:Matlab	3.8/.95 28.75/9.68	.13/0.0 14.32/.22
#2: Ada #2: Matlab	11.9/2.27 28.05/8.9	.11/.02 14.37/.15
#3: Ada #3: Matlab	8.85/2.68 13.6/5.12	.12/.04 11.93/.14

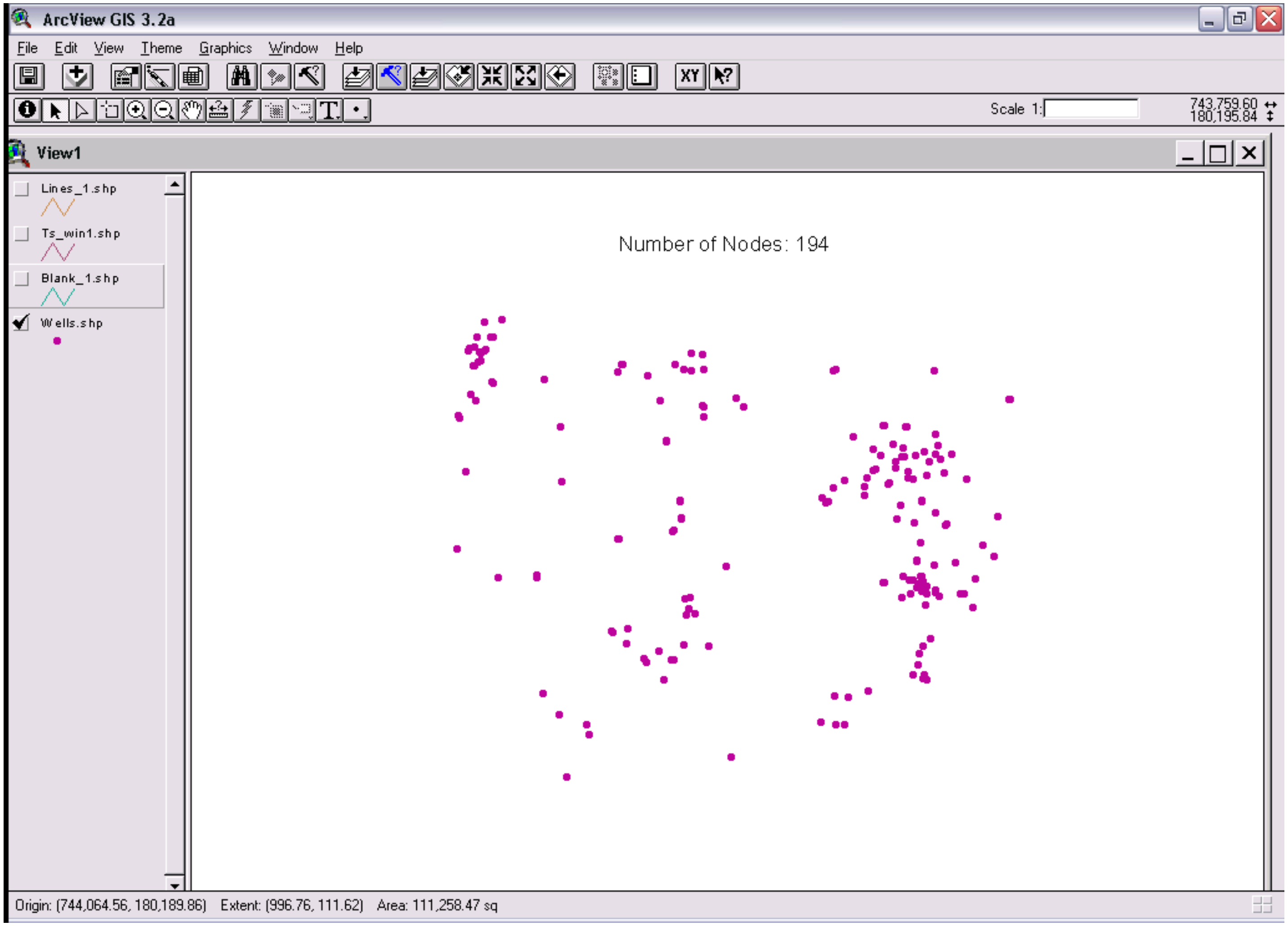
Matlab Conclusions

- Implementation
  - Other chromosome encoding schemes possible
  - User provides algorithm, crossover, mutation
  - Less work involved in writing new procedures
- Performance
  - Matlab script is interpreted, Ada95 compiled
  - Ada95 GA better equipped to escape locally optimal solutions

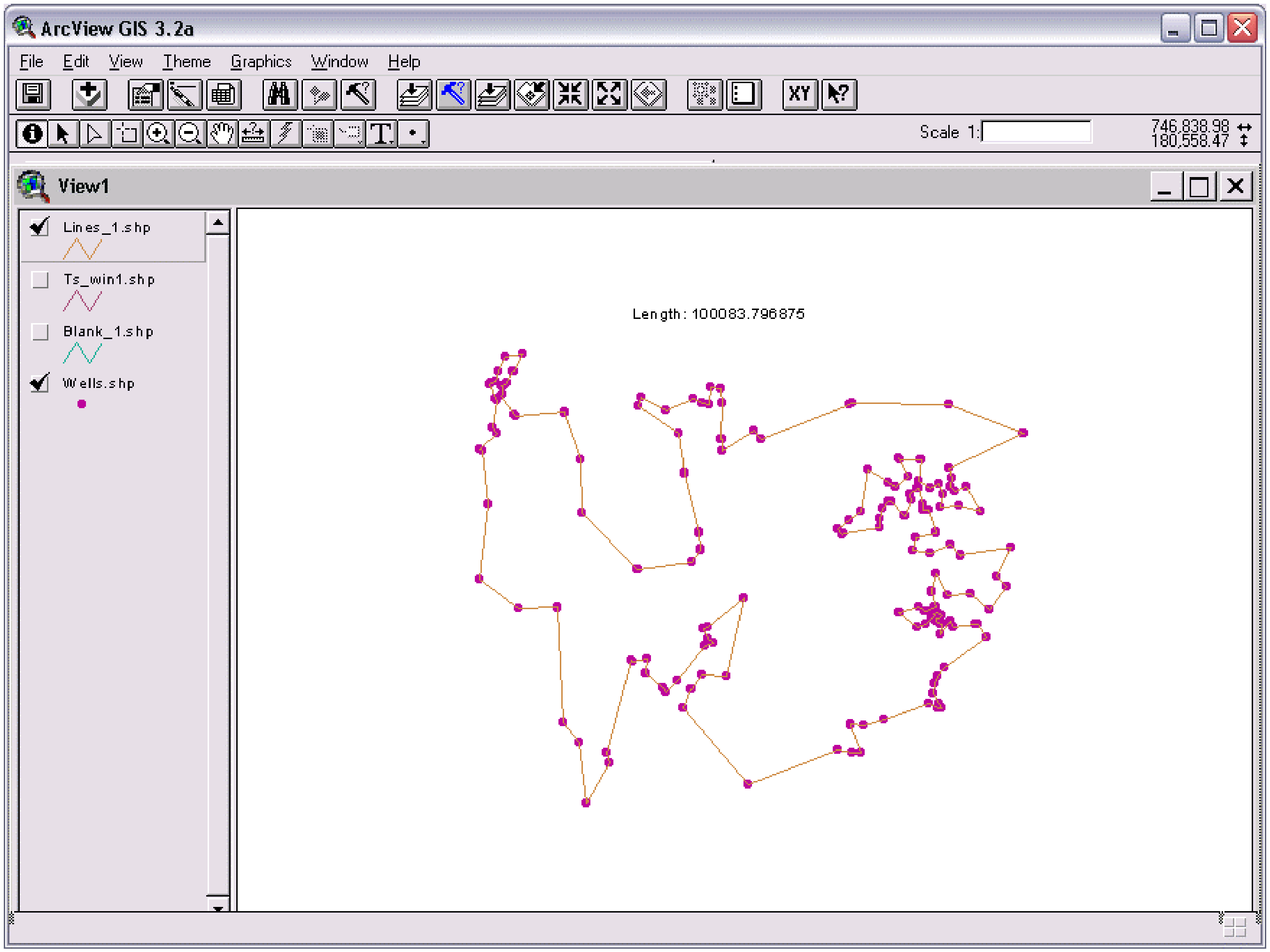
Flexible GA

- Ada95 GA provides same abstractions as Matlab GA
  - Data structures, utility packages provided
  - User implements chromosome, crossover, mutation, objective function
- Generics allow for change of chromosome encoding through instantiation
- Three problems already implemented, user can extend existing types for new problems
- http://astrogeology.usgs.gov/Projects/flexible_ga

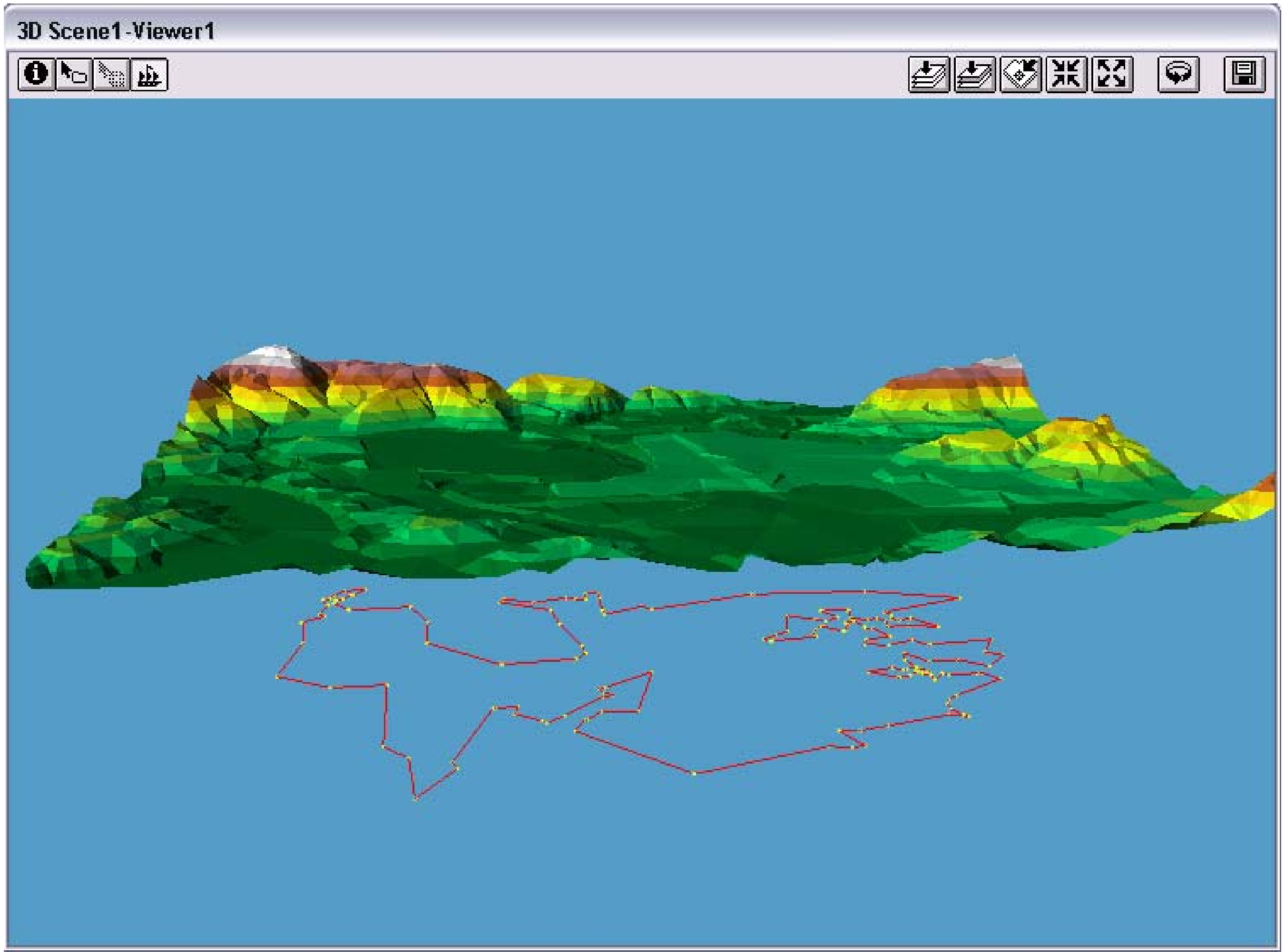
GIS TSP Module



GIS TSP Module



GIS TSP Module



GIS TSP Module