Evolutionary Algorithms Workflow

• To design an effective evolutionary algorithm, one needs to consider the problem at hand.

EA Components are Connected

Outline

• Evolutionary Algorithms Design
• Representations and Fitness Function
  – Problem Modeling
• Fixed-length Representation: mostly used for optimization type of problems
  – Binary String
  – Integer String
  – Real-Number Vector
  – Permutation
Design an Effective EA

- The representation (genotype, phenotype, developmental process) allows genetic operators to maintain a fitness link between parents and offspring (inherit genes and fitness).
- The selection method not only allows fit individuals to survive but also maintains sufficient genetic diversity in the population.
- Goal: evolve better and better solutions!!!

Modeling the Problem

- Problem solving is to find the solution to a model of the problem.
- All models are a simplification of the real world.
- Problem solving therefore consists of 2 steps:
  - Creating a model of the problem
  - Using that model to generate a solution.
- When applying EC for problem solving, the problem is modeled using appropriate representation and fitness function.

Representation

- Representation describes the solution of the given problem.
  - If wrongly designed, EA is solving a different problem.
- Representation consists of genotype, phenotype and developmental process.
- Genotype defines the search space while phenotype characterizes the solution space.
- Many problems can be solved using genotype alone without phenotype or developmental process.

Binary String

- A genotype consists of a fixed-length binary numbers of 1 and 0.
  - Example: 0 1 0 1 1 0 0 1 1
- Search space: $2^n$, n is the length of the string
- Binary string can be used to encode integer optimization problems.
  - Example: 01001001000110
  - Decoding: $x_1 \ x_2 \ x_3$
Problem Example

• Feature Selection:
  – Select the genes from the patients' microarray data that contributes to a disease.
  – **Variables**: $x_1 \ldots x_n$ are the selection of $n$ genes.
  – **Representation**: binary string $x_1 \ldots x_n$. Bit 1 indicates the gene is selected, bit 0 indicates it is not.
  – 10 genes example: 1 0 1 1 0 0 1 0 1 1

Integer String

• A genotype consists of a fixed-length integer numbers within a given range.
  – Example: 10 9 7 23 6
• Search space: $\prod_{i}^{R_i}$, where $n$ is the length of string and $R_i$ is the value range of the $i$th integer.
• Compared to the binary string representation, do you like this representation better for integer optimization problems? Why or why not?

Problem Example

• Knapsack problems:
  – Fill a knapsack that can hold a total weight of $W$ with some combination of items from a list of $n$ possible items each with weight $w_i$ and value $v_i$. The goal is to determine the number of each item to include in the knapsack so that the total value is maximized.

Knapsack Problem

• **Variables**: $x_1 \ldots x_n$ are the quantities of the $n$ items.
• **Representation**: $x_1 \ldots x_n$ of $n$ integers.
• **Constraint**: $\sum_{i=1}^{n} x_i w_i \leq W$
• **Fitness function**: maximize $\sum_{i=1}^{n} x_i v_i$
Real Number Vectors

- Genotype is a real-number vector containing the parameters to be optimized.
  - Example:

\[
\begin{array}{cccc}
1.45 & 7.98 & 5.45 & 2.31 \\
x_1 & x_2 & x_3 & x_4 \\
3.34 & & & \\
x_5 \\
\end{array}
\]

- Real-number vector is the natural representation for numerical optimization.
- The search space is continuous.
- What is the size of the search space?

Problem Example

- Objective: Maximize \( f(x) \):

\[
f(x) = \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2 \prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} x_i^2}}
\]

subject to

\[
C_{\gamma_i} x_i = 0.75; \sum_{i=1}^{n} x_i = 7.5 \text{ and } 0 \leq x_i \leq 10; 1 \leq i \leq n
\]

- Variables: \( x_1..x_n \) of \( n \) parameters for optimization
- Representation: real value vector for \( x_i \)

Permutation

- A permutation of a finite set is an arrangement of its elements into a row (Knuth, 1973).
  - \{a, b, c, d, e\}, \{b, c, d, e, a\}, \{c, d, e, a, b\} ....
- Search space: \( n! \) for \( n \) elements
- Permutation is a suitable coding for combinational optimization problem.
  - Scheduling, network routine etc.

Traveling Salesman Problem

- The salesman must visit every city in his territory exactly once and then return home covering the shortest distance.
- Variables: \( x_1..x_n \) are the \( n \) city names
- Representation: permutation of \( n \) cities.
- Fitness function: minimize \( \sum_{i=1}^{n} \sum_{j=1}^{n} dis(x_i, x_j) \)
- 10 cities example:
  - 1 3 4 10 9 8 2 5 6 7

- Diagram of cities and connections.
TSP Search Space

• Given \( n \) unique objects, there are \( n! \) permutations. Searching the shortest path is an NP-hard problem.
• There are multiple equivalent solutions for some problems.
  – TSP: If starting point is not important, and the distance from city i to j is the same as that from city j to i, each path a-b-c-d-e can be represented in \( 2n \) ways and give the same distance.
• Search space: \( n!/(2n)=(n-1)!/2 \).

Representation Design Exercises

• There are multiple ways to represent a problem solution.
• Some exercises may be solved by hand, do not need evolutionary algorithms. But it doesn’t matter, as we are practicing problem modeling.
• Gives variables, representation, constraints and fitness function.

What are the prices in 7-11

• A customer purchased 4 items at a local 7-11 store. The product of the 4 items prices is \$7.11\). The sum of the 4 items prices is also \$7.11\). What are the prices of the 4 items? – from [Michalewicz and Fogel]

The Vehicle Routing Problem

There are 3 vehicles at the depot. Each of the 8 customers has to be visited precisely once. Each vehicle needs a route that takes it from the depot and back to the depot. What is the least cost (distance/fuel/vehicles) solution? Several vehicles normally needed, since visiting 3 or 4 customers may take the whole day, but all customers need to be serviced today. Also, vehicles have capacity constraints. Let’s ignore the capacity at this moment as we don’t have the information about the load to be delivered.
Order of Ages

- Andrew, Carol, Jessica, Luke, and Tommy are sitting around a circular table. Carol is 12 years older than her neighbor to the left. Tommy is five years older than his neighbor to the right. Jessica is 14 years older than her neighbor to the left. Luke is five years younger than his neighbor to the left. By age, from youngest to oldest, they are ordered as follows: Luke, Tommy, Andrew, Jessica, and Carol. Luke is 16 and Carol is 40. The total of the ages is 135. In which order are the people sitting (starting with Tommy and then proceeding in clockwise order) and what are their ages? - from Shasha

Capital budgeting

- There are four possible projects, which each run for 3 years and have the following characteristics.

<table>
<thead>
<tr>
<th>project</th>
<th>Return (million)</th>
<th>Capital required</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2</td>
<td>1st: 0.5; 2nd: 0.3; 3rd: 0.2</td>
</tr>
<tr>
<td>2</td>
<td>0.3</td>
<td>1st: 1.0; 2nd: 0.8; 3rd: 0.2</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>1st: 1.5; 2nd: 1.5; 3rd: 0.3</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>1st: 0.1; 2nd: 0.4; 3rd: 0.1</td>
</tr>
</tbody>
</table>

- Available budget: year 1: 3.1, year 2: 2.5, year 3: 0.4
- We have a decision problem here: Which projects would you choose in order to maximize the total return?

Scheduling Tradition

- There are 12 school teams, unimaginatively named A, B, C, D, E, F, G, H, I, J, K, and L. They must play one another on 11 consecutive days on six fields. Every team must play every other team exactly once. Each team plays one game per day. Can you provide a schedule for this tournament? - from Shasha

Sudoku

- The target state is to fill a 9 by 9 grid with digits between 1 and 9. Each digit should appear exactly once in each row, once in each column, and once in each non-overlapping three by three box starting from the upper left corner. - from Shasha
### Sudoku Game Example

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>4</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>9</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>