Machine Learning B
KU

09 January 2007

DI Michael Pfeiffer
pfeiffer@igi.tugraz.at
Inst. für Grundlagen der Informationsverarbeitung
TU Graz

Genetic Algorithms

- Plan for today:
  - Short Example
  - Genetic Algorithms Toolbox GAOT
  - Truck Backer-Upper Task

Genetic Algorithms: Components

- Population of chromosomes / genomes
- Fitness function
- Selection Function
- Reproduction Functions
  - Mutation
  - Crossover
- Initial Population
- Stopping Criterion
Scheme of GAs

- \( i = 1 \)
- repeat
  - \( P_i' = \text{select}(P_{i-1}) \)
  - \( P_i = \text{reproduce}(P_i') \)
  - \( i = i+1 \)
- until stopping criterion applies

- **Solution**: individual with maximal fitness in any population \( P_i \)

Illustrative Example

- **MAXONE**: minimize number of 1's in a finite bit-string
  - \( x \in \{0,1\}^\mu \)
  - Fitness:
    - \( f(x) = \sum_{i} x_i \)

<table>
<thead>
<tr>
<th>Population ( P_0 )</th>
<th>( N=6, m=6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000 ( f_1=1 )</td>
<td></td>
</tr>
<tr>
<td>010101 ( f_2=3 )</td>
<td></td>
</tr>
<tr>
<td>110101 ( f_3=4 )</td>
<td></td>
</tr>
<tr>
<td>001100 ( f_4=2 )</td>
<td></td>
</tr>
<tr>
<td>101101 ( f_5=4 )</td>
<td></td>
</tr>
<tr>
<td>000000 ( f_6=0 )</td>
<td></td>
</tr>
</tbody>
</table>

Roulette Wheel Selection

- "Fitness Roulette Wheel"
**Binary Mutation**

P₀:
- 111010 101101 101101
- 111010 101101 101101 001111 ...

P₁:
- 101101
- 001111

2 bit flips

**Binary Crossover**

P₀:
- 111010 101101 101101
- 111010 101101 101101 001111 111101 101010

P₁:
- 111010 101101 101101 001111 111101 101010

**Improvement**

P₀:
- max. fitness: 4
- avg. fitness: 2.33
- f₁=1, f₂=3, f₃=4, f₄=2, f₅=4, f₆=0

P₁:
- max. fitness: 5
- avg. fitness: 4
- f₁=4, f₂=4, f₃=4, f₄=4, f₅=5, f₆=3
Genetic Algorithm Optimization Toolbox

**GAOT**

- Toolbox for Genetic Algorithms in Matlab
- Standard Set of Genetic Algorithms:
  - Binary and Real-valued codes
  - Selection Functions
  - Mutation and Crossover Operators
  - Termination criteria
- Available from [http://www.ise.ncsu.edu/mirage/GAToolBox/gaot/](http://www.ise.ncsu.edu/mirage/GAToolBox/gaot/)
  - Old and simple, but useful as introduction

---

GAOT Syntax

```matlab
[x, endPop, bPop, traceInfo] = ga(bounds, evalFN, evalParams, params, startPop, termFN, termParams, selectFN, selectParams, xOverFNs, xOverParams, mutFNs, mutParams);
```

- `x` ... best solution
- `endPop` ... final population
- `bounds` ... upper and lower bounds of genes
- `evalFN` ... Name of evaluation Function
- `startPop` ... initial population
- `termFN` ... Stopping criterion
- `selectFN` ... Name of selection function
- `xOverFNs` ... String of names of XOver operators
- `mutFNs` ... String of names of mutation operators
- `<...>Params` ... Parameters for the called function

---

Evaluation Function

- General Form:
  - function `[x, val] = evalFN(x, parameters)`
  - `parameters = [current_generation, evalParams]`
- Calculates fitness values and may modify genetic strings
- Writing this function is one of the main tasks!
- E.g. evaluating your ANN in one or several problems is done here
Mutation Operators 1/2

- Binary Mutation: random bit-flip
  - 'binaryMutation'
  - 10101 → 101101
- Uniform Mutation:
  - randomly select one variable of the string and set it to an uniform random number in [a, b]
  - 'unifMutation'
  \[
  x_i = \begin{cases} 
  \frac{b - a}{2} + \frac{b - a}{2} \cdot r_i & \text{if } i = j \\ 
  x_i & \text{otherwise}
  \end{cases}
  \]

Mutation Operators 2/2

- Boundary Mutation:
  - 'boundaryMutation'
  \[
  x_i = \begin{cases} 
  a & \text{if } i \neq j < 0.5 \\
  b & \text{if } i \neq j \geq 0.5 \\
  x_i & \text{otherwise}
  \end{cases}
  \]
- Non-uniform Mutation:
  - mutation probability sinks with time
  - 'nonUnifMutation', 'multiNonUnifMutation'
  \[
  f(G) = \left( 1 - \frac{G}{G_{\text{max}}} \right)^b
  \]

Crossover Operators

- Simple Crossover: 'simpleXover'
- Arithmetic Crossover: 'arithXover'
  \[
  X^* = rX + (1-r)Y \\
  Y^* = (1-r)X + rY \\
  r \sim U(0,1)
  \]
- Heuristic Crossover: 'heuristicXover'
  - use fitness information: f(X) > f(Y)
  \[
  X^* = X + r(X - Y) \\
  Y^* = X \\
  r \sim U(0,1)
  \]
  if X is feasible (i.e. within bounds)
Selection Functions 1/2

- **Roulette Wheel**: 'roulette'

- **Normalized Geometric Selection**:
  - 'normGeomSelect'
  - rank individuals according to fitness
  - negative fitness is allowed
  
  \[ P[\text{selecting the } i^{th} \text{ individual}] = \frac{q^i}{1 - (1 - q)} \]

  \begin{align*}
  q & \quad \text{prob. of selecting the best individual} \nonumber \\
  r & \quad \text{rank of the individual} \nonumber \\
  P & \quad \text{population size} \nonumber 
  \end{align*}

Selection Functions 2/2

- **Tournament Selection**
  - 'tournSelect'
  - select randomly (with replacement) \( j \) individuals from the population
  - insert best of the \( j \) into new population
  - repeat until \( N \) are selected

Termination Functions

- **'maxGenTerm'**
  - Terminate after fixed number of generations

- **'optMaxGenTerm'**
  - Terminate as soon as some optimale fitness value is reached or after fixed number of generations
Example 1: Function Optimization

- Chromosome:
  - \([x, y]\)

- Fitness = \(f(x, y)\)

- Standard real-valued Mutation and Crossover

---

Travelling Salesman Example

- Genetic Code: Permutation of cities
  - e.g. \((5 \, 3 \, 1 \, 4 \, 2)\)

- Mutation:
  - Flip two cities
  - e.g. \((5 \, 2 \, 1 \, 4 \, 3)\)

- Crossover:
  - Preserve order in two tours!
  - e.g. \((53142), \,(14253)\)
  - \(\rightarrow (53214), \,(1253)\)

---

Truck Backer-Upper Problem

- Find a good ANN controller to steer the rear midpoint of a trailer to the loading dock (with 90° angle)
  - Only cab can be controlled (steering angle)
  - Constant backwards speed

- 4D Input:
  - \((x, y)\) of read midpoint
  - Angles between trailer respectively cab and x-axis

---

<table>
<thead>
<tr>
<th>Variables</th>
<th>Default Value</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>10 km/h</td>
<td>speed of truck</td>
</tr>
<tr>
<td>(\beta)</td>
<td>0 deg</td>
<td>angle of trailer</td>
</tr>
<tr>
<td>(L_b)</td>
<td>4 m</td>
<td>length of cab</td>
</tr>
<tr>
<td>(L_c)</td>
<td>8 m</td>
<td>length of trailer</td>
</tr>
</tbody>
</table>
Evolution of ANN Controllers

- Weights of ANN are evolved
  - Not trained with backprop
- Inputs:
  - \((x, y, \theta_c, \theta_t) + \text{bias}!!\)
- Weights are given as vectors of real numbers
  - Transform genome into appropriate network
- Output:
  - Control variable \(u\)

Truck Backer-Upper Files

- `sim_TruckBackerUpper`
  - physical environment
  - Simulates one step with your ANN controller
  - Can be modified for more efficient simulation
  - Do not change the physical model itself!
- `plot_TruckBackerUpper`
  - plots trajectory of controller

Your Task

- Configure GA and parameters to evolve good controller for trailer backup task
  - Use physical model to define a fitness function
  - Everything about the performance needs to be taken into account for the fitness value
    - crashed? final position, final angle, time, ...
  - Define selection, mutation, crossover
  - If you want, you can implement your own operators
- Let the GA run and watch mean and maximum fitness