3-28-2011
Do you have any questions about the quiz?

Minute essay comment:
- Getting up to speed with SwiftVis.

ScalaLab:
- http://code.google.com/p/scala-lab/

I'd like to thank Wikipedia for help with this lecture.
This is a very broad title for quite a few different techniques that are based on the central themes of evolution.

The field of evolutionary computation was born in the 1950 and there were three separate groups working independently through the 1960s on different approaches.

All methods are metaheuristic optimization algorithms.
Evolutionary Algorithm

- Subset of Evolutionary Computation.
- Utilize mechanisms from biological evolution:
  - Reproduction, mutation, recombination, and selection.
- Optimize some form of fitness function.
- It includes five main subcategories:
  - Genetic Algorithm (GA), Genetic Programming (GP), Evolutionary Programming, Evolutionary Strategy, and Neuroevolution
Short Descriptions

- Genetic Algorithm
  - Most popular.
  - Seeks solution of a problem in the form of strings of numbers by applying operators such as recombination and mutation.
  - Strings should have a form that reflects the nature of the problem.

- Genetic programming
  - Solutions are in the form of computer programs.
  - Fitness is determined by their ability to solve a computational problem.
More Short Descriptions

- **Evolutionary programming**
  - Similar to genetic programming, but the structure of the program is fixed and its numerical parameters are allowed to evolve.

- **Evolution strategy**
  - Works with vectors of real numbers as representations of solutions, and typically uses self-adaptive mutation rates.

- **Neuroevolution**
  - Similar to genetic programming but the genomes represent artificial neural networks by describing structure and connection weights.
Genetic Algorithms

- **Required Elements:**
  - Encoding for “individuals”. Typically this is done as an arrays of values to keep things simple.
  - Fitness function that evaluates quality of a solution.

- **Processes**
  - Initialization - Population is initialized randomly, perhaps with seeds for areas expected to be good.
  - Selection - Select from the population based on fitness.
  - Reproduction - Reproduce by doing crossover from selected population. Optionally apply mutation as well.
  - Termination – Stop when it has gone too long or a sufficient quality is reached.
GA Algorithm

- Initialize
- Evaluate fitness
- Repeat the following until termination
  - Select best-fit individuals
  - Breed selected individuals
  - Evaluate fitness of offspring
  - Replace least-fit with new individuals
- Consider 0,1-knapsack problem.
- It is customary to use binary strings and use Gray coding instead of normal binary representation.
Jiva-NG

- http://code.google.com/p/jiva-ng/
- This is a GA library written for Scala.
Genetic Programming

- This works very much like GA, but with programs that are typically represented as trees.
- Crossover is done by swapping branches of trees.
- Mutation can change branches of leaf values.
- You must either take care to only generate valid programs or to be fail-safe.
What would you like us to spend our time doing next class? Try Jiva-NG? Write GP code in Scala?