

PHY 411-506 Computational Physics II
Chapter 12: Interdisciplinary Topics
Lecture 8

Wednesday April 23, 2008

Lecture Outline

Ion Channels in Cell Membranes	3
Optimization Problems	4
The Thomson Problem	4
The Traveling Salesman Problem	4
Genetic Algorithms	6
More on Genetic Algorithms	10
Creating a gene pool	10
Encoding a configuration	11
Selection operation	12
Crossover operation	13
Mutation operation	13

Ion Channels in Cell Membranes

- Abundance of chemical elements
- Ion Channels
 - ◇ Calcium Channels

Optimization Problems

The Thomson Problem

- Determine the stable equilibrium positions of classical electrons constrained to move on the surface of a sphere and repelling each other by an inverse square law.
- MathWorld
- Thomson Applet at Syracuse University.

The Traveling Salesman Problem

- A famous example of an optimization problem is the Traveling Salesman Problem.
- Other techniques that have been applied to the traveling salesman problem:
 - ◇ Genetic Algorithms. Quite successful for solving TSP. Numerous applications to other types of problems. Software based approach.
 - Genetic Algorithm for the Traveling Salesman Problem
 - Genetic Algorithm Viewer

- ◇ Neural Networks. Not so successful for solving TSP. Many applications to other types of problems. Both software and hardware based approaches.

Genetic Algorithms

- Genetic algorithms try to model evolution by natural selection. In nature the *genetic code* is stored in DNA molecules as sequences of bases: adenine (A) which pairs with thymine (T), and cytosine (C) which pairs with guanine (G).
- The analog of DNA in a digital genetic algorithm is a sequence of binary digits (0) and (1).
- In nature, the genetic code describes a *genotype*, which is translated into an organism, a *phenotype*, by the process of cell division.
- Digital genetic algorithms can be used to solve a problem, such as finding the global minimum of a complicated energy landscape.
- The phenotype in a genetic algorithm is some state of the model: strings of binary digits are mapped to the states of the model to be solved.
- Evolution by natural selection is driven in part by changes to the genetic code:
 - ◇ **Mutations** Random changes can occur, for example caused by

radioactivity or cosmic rays damaging a DNA molecule. Mutations of the digital genotype can be modeled by choosing a random bit in the string and changing it $1 \rightarrow 0$ or $0 \rightarrow 1$.

- ◇ **Recombination or Crossover** During sexual reproduction the offspring inherit DNA from each of the parents. This can be simulated by taking two strings and exchanging two substrings.
- ◇ **Survival of the Fittest** There is some criterion of *fitness* such that when mutations or recombinations take place, the mutants or offspring either survive and reproduce or die out.
- These simple ingredients can be used to construct a very wide variety of genetic algorithms.
- A simple algorithm which can be applied to an energy landscape problem is illustrated by the random Ising model:

$$E = - \sum_{\langle ij \rangle} T_{ij} s_i s_j ,$$

where $s_i = \pm 1$ are Ising spins, and the coupling constants T_{ij} between nearest neighbors are chosen randomly to be ± 1 . This is a model of a

spin glass which has a very complicated energy landscape with numerous local minima.

- What is a genotype for this model? Suppose we have a 2-D lattice of spins with $i, j = 0, 1, \dots, (L - 1)$, then we can order the spins linearly using the formula $n = iL + j = 0, 1, \dots, (L^2 - 1)$ for example. A configuration of spins is mapped to a genotype of L^2 bits by setting the bit with index n to 0 or 1 if $s_{ij} = \pm 1$.
- Since we are seeking the global energy minimum, the *fitness* of a particular genotype can be taken to be $2L^2 - E$, since the minimum and maximum possible values for the energy are $\mp 2L^2$ for a 2-D square lattice and periodic boundary conditions. (Recall that the number of bonds is then twice the number of spins.)
- The following is one possible evolution protocol:
 - ◇ Start with a population of a fixed number N_0 of strings initialized in some way, for example by setting the string bits randomly.
 - ◇ Repeat the following “generations”:

- Allow some number of mutations. For example, choose 20% of the strings at random, and mutate a random bit (flip a random spin) in each string.
- Choose some number of pairs of strings at random and have them “reproduce” as follows: each pair produces two offspring which differ from the parents by exchange of a randomly chosen substring.
- The size of the population has now increased from N_0 to N due to reproduction, and the parents and children are competing for the same limited natural resources. Select N_0 fittest survivors as follows:

★ Construct a cumulative histogram

$$H_k = \sum_{i=1}^k (2L^2 - E_i) , \quad k = 1, 2, \dots, N ,$$

where k labels the strings in the population.

★ Repeat N_0 times:

★ Choose a random H between 0 and the maximum H_N .

★ Select the smallest k such that $H_k > H$.

- After many generations the population should converge to the global energy minimum configuration!
- There are numerous resources on genetic algorithms on the web, see for example Wikipedia Genetic Algorithm.

More on Genetic Algorithms

Computational Physics by Pang textbook has a more detailed discussion on genetic algorithms in Chapter 11, including how to encode real variables.

- Basic problem is to find the global minimum of a multi-variable function $g(r_1, r_2, \dots, r_n)$
- In a *binary algorithm* each configuration of (r_1, r_2, \dots, r_n) is represented by an array of 0s and 1s (false, true)
- Multi-objective optimization can involve several objective functions $g_k, k = 1, 2, \dots, \ell$

Creating a gene pool

- Need to choose the population size
 - ◇ Too small takes more time to explore the configuration space

- ◇ Too large slows the updating process
- Choose the initial pool randomly
- Convergence can be improved by generating twice as many random chromosomes and selecting the fittest half
- Need a function to sort the genes in a pool according to fitness

Encoding a configuration

- By scaling and shifting, all variables can be mapped into the unit interval $0 \leq r_i \leq 1$
- Can represent

$$r_i = \frac{y_{i1}}{2} + \frac{y_{i2}}{4} + \frac{y_{i3}}{8} + \dots = \sum_{j=1}^{\infty} \frac{y_{ij}}{2^j}$$

where $y_{ij} = 0, 1$

- Truncate this exact representation at some m

$$r_i \simeq \sum_{j=1}^m \frac{y_{ij}}{2^j}$$

- The formula for *encoding* is

$$y_{ij} = \text{int} \left[2^{j-1} r_i - \sum_{k=1}^{j-1} 2^{j-k-1} y_{ik} \right]$$

- y_{ij} , $j = 1, \dots, m$ is the i -th gene of the chromosome
- The whole chromosome is represented by a binary array
 $w = [y_{i1} \dots y_{1m} y_{21} \dots y_{2m} \dots y_{nm}]$
- To *decode* a chromosome, just decode each individual gene

$$r_i = \sum_{j=1}^m \frac{y_{ij}}{2^j}$$

Selection operation

- Many criteria have been used to select the next generation, see Wikipedia Genetic Algorithm
- A simple choice is to use the fittest half of the population as the parent pool

- A popular method for selecting mating pairs is to hold a *tournament*
 - ◇ Shuffle the gene indices of the pool
 - ◇ Select half of the fitter genes in the shuffled pool
- Need a function to shuffle the genes in the pool

Crossover operation

- Choose a pair of parents from the parent pool
- In *single point crossover* cut each gene at the same location and switch segments between the two parents
- Order the pool according to fitness – this requires a ranking function

Mutation operation

- A random mutation of a bit changes $0 \leftrightarrow 1$
- Choose some fraction of bits from the entire pool and mutate them
- The fraction (for example 1%) must be chosen carefully
 - ◇ More mutations help explore the configuration space faster
 - ◇ But too many mutations degrade the fitness of the pool