Monte Carlo Methods

Kevin Cahill

cahill@unm.edu

http://dna.phys.unm.edu/

Abstract

Monte Carlo methods are robust and effective. They can solve an extremely broad class of problems. They can do integrals, simulate behavior, and design systems. They are naturally suited to statistical physics and quantum field theory.

Monte Carlo Methods Are Widely Applicable

One may use Monte Carlo methods to solve or approximate the solutions to an amazingly wide class of problems.

In this talk, I will show how to use them to approximate definite integrals, to design experiments, to simulate behavior, to discover new designs for complex systems, and to do statistical physics.

Definite Integrals

One may use random numbers to approximate the value of a definite integral. For instance, to ''Monte Carlo'' the integral

$$I = \int_a^b dx\, f(x), \tag{1}$$

one simply generates a large number N of random numbers x_i uniformly distributed on the interval (a, b), and boldly sets

$$I = \frac{(b-a)}{N} \sum_{i=1}^{N} f(x_i). \tag{2}$$

A great advantage of this method is that the user does not need to know very much. So it's my favorite method.

Multi-Dimensional Integrals

The method works equally well in two or more dimensions. Thus to ''Monte Carlo'' the integral

$$I_2 = \int_a^b dx_1 \int_c^d dx_2 \; f(x_1, x_2), \tag{3}$$

one has one's computer generate a large number N of pairs of random numbers (x_{1i}, x_{2i}) uniformly distributed on the rectangle of base (a, b) and side (c, d) and boldly sets

$$I_2 = \frac{(b-a)(d-c)}{N} \sum_{i=1}^{N} f(x_{1i}, x_{2i}). \tag{4}$$

The method is so effective and so simple that it makes mathematics seem trivial. One can get carried away.

Simulations of Experiments

Experimenters use random numbers and sets of rules to generate samples of events. They perform three kinds of Monte Carlo simulations:

For instance, in proton-antiproton experiments at Fermilab, $p\bar{p} \to X$, physicists use the parton model and empirical rules to predict the rates of production of various sets X of particles at specific momenta.

They then use random numbers to generate sets of particles and momenta that are representative of their experiments. These sets of particles then serve as input to their detectors, and the responses of the detectors are computed.

Physicists repeat these two steps to test various hypotheses, such as a particular model of supersymmetry. These tests are sometimes called pseudo-experiments.

Experimenters use these three Monte Carlo methods to design and analyze experiments.

Simulations of Behavior

Here one creates a set of actors, a set of actions, and a set of rules that assign conditional probabilities to actions.

One can, for instance, simulate a flock of birds. Here the actors are birds; the actions are ways of flying; and the rules assign probabilities to actions that depend upon the actions of the other actors.

Such Monte Carlo techniques were used to generate the behavior of herds and flocks in the ''Jurassic Park'' movies.

They also are used to generate the behavior of people in the ''Sim City'' computer games.

Genetic Algorithms

A genetic algorithm is a Monte Carlo method inspired by biology.

Suppose one wants to design a network of pipes and valves to transport natural gas quickly, safely, and cheaply from a set of gas wells and gas depots to a set of consumers.

One may use the Monte Carlo method to design the pipeline system. To do so, one writes the design of an arbitrary pipeline system as a sequence of separate instructions. Different sequences will have different lengths.

Next, one writes a program that computes the speed, cost, and safety of a pipeline system specified by an arbitrary sequence of instructions. The program will weigh speed, cost, and safety and assign an overall score to any sequence of instructions for building a pipeline.

Next, one uses random numbers to generate a population of instruction sequences. Now, one could stop here; one then would evaluate each sequence of instructions and find the best one.

A genetic algorithm goes one step further.

Sex and Evolution in Genetic Algorithms

The amusing new idea is to think of each sequence of instructions as the DNA of an individual and to imitate sex and evolution.

One allows individuals to mate and to produce offspring some of which carry mutations and some of which result from chromosomal crossing-over. Chromosomal crossing-over is the exchange of DNA between paired homologous chromosomes in division I of meiosis. This crossing-over is also know as genetic recombination. It is why we have sex.

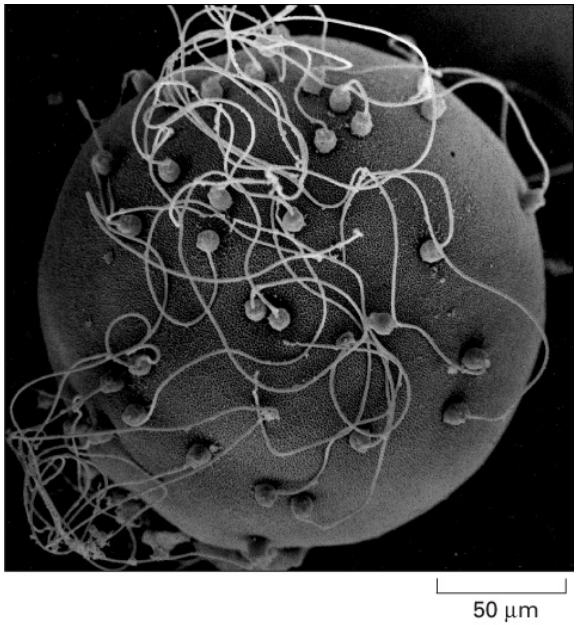

Figure 20–4. Molecular Biology of the Cell, 4th Edition.

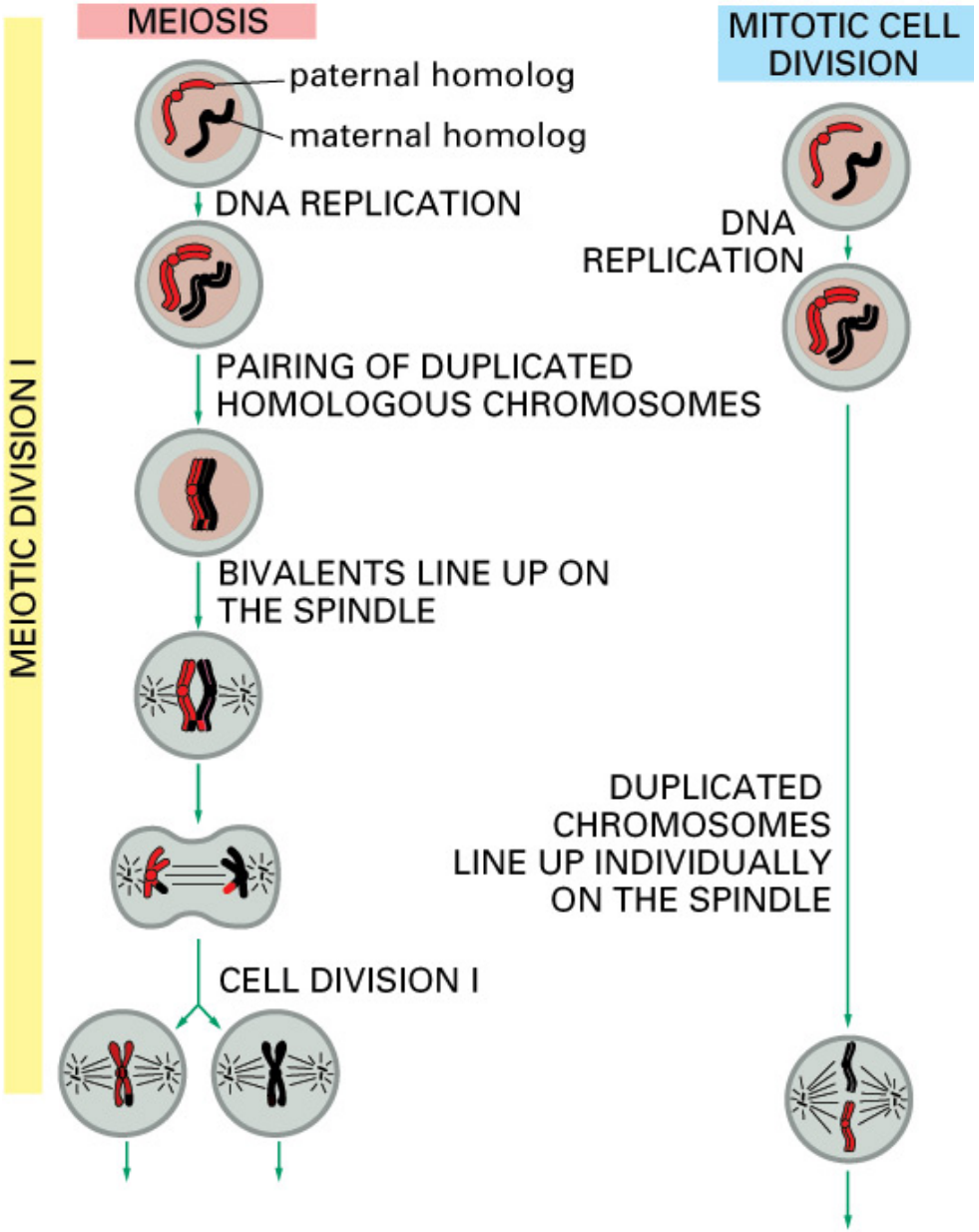

Figure 20–7 part 1 of 2. Molecular Biology of the Cell, 4th Edition.

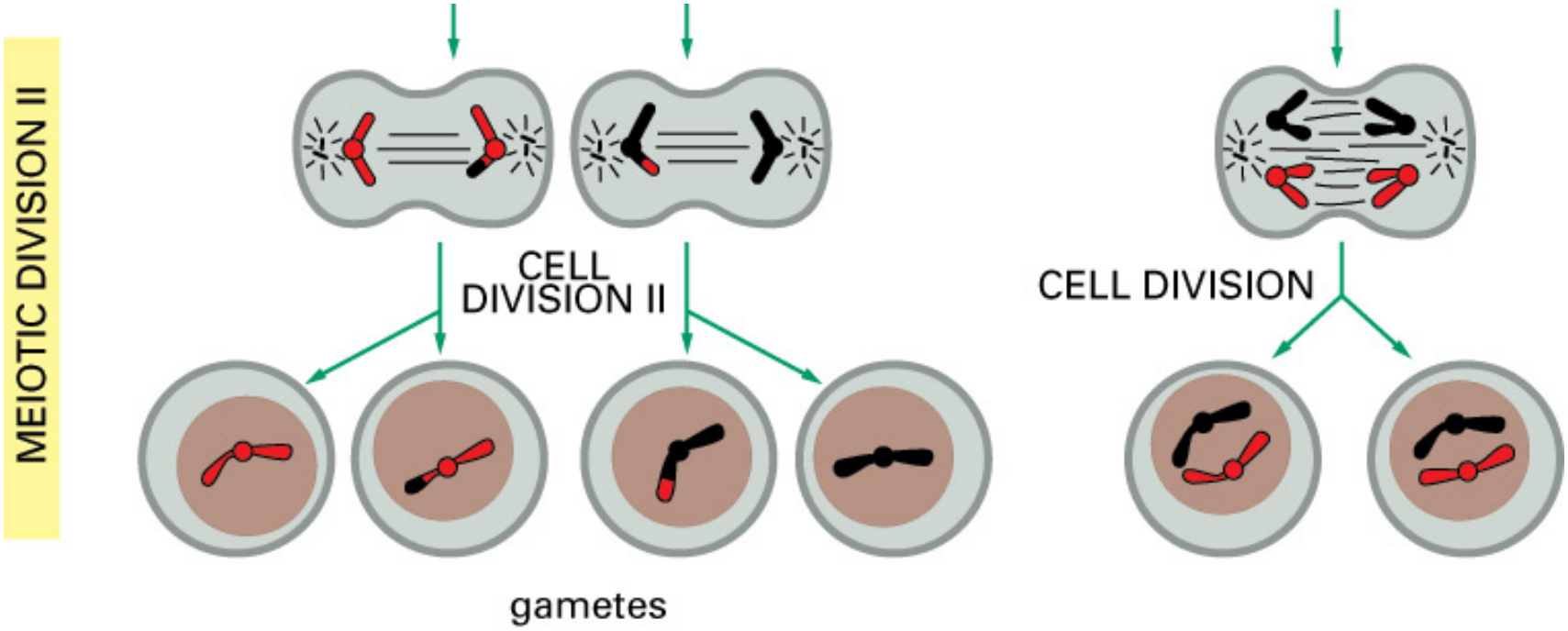

Figure 20–7 part 2 of 2. Molecular Biology of the Cell, 4th Edition.

A genetic algorithm uses genetic recombination (sex) to generate a large assortment of different designs for the pipeline system.

The genetic algorithm then tests each of the offspring of each generation and keeps the top 80% or so. Successive stages of mating produce new designs, which are tested.

In this way, the population of designs evolves to something Halliburton might use.

Enter Ludwig Boltzmann (1844-1906)

In statistical physics, a physical configuration c occurs with a probability determined by its energy $E(c)$ through the Boltzmann factor

$$P(c) \propto e^{-\beta E(c)} \tag{5}$$

where $\beta = 1/(kT)$, k is the Boltzmann constant, and T is the absolute temperature.

About 60 years ago, during the Manhattan Project, Fermi invented a way to generate ensembles of configurations c weighted by such factors. He was too busy to write it up, so he gave the idea to Metropolis and Teller, who published it.

The algorithm is simple: each configuration c randomly jumps to a trial configuration c'. If the energy $E(c')$ of the trial configuration c' is less than $E(c)$, i.e., if $E(c') \le E(c)$, then the trial configuration c' is accepted. If $E(c') > E(c)$, then the trial configuration c' is accepted with probability

$$P(c \to c') = \exp\left[-\beta \left(E(c') - E(c)\right)\right] \tag{6}$$

and returned to configuration c with probability $1 - P(c \to c')$.

Why the Metropolis Algorithm Works

Suppose we just have two configurations or states with energies E_1 and $E_2 > E_1$. Suppose they occur with probabilities P_1 and P_2. Then after one step, a system in state 2 can stay as 2 or jump to state 1. In both cases, the energy drops or stays the same, so each transition has probability $P = 1/2$. Now a system in state 1 can jump to state 2 with probability 1/2, but that transition is accepted with probability $P(1 \to 2) = \exp[-\beta\,(E_2 - E_1)]$ and sent back to state 1 with probability $1 - P(1 \to 2)$. The system also can just jump from state 1 to state 1 with probability 1/2. So the total probability that the system remains in state 1 is $P(1 \to 1) = 1/2 + (1 - P(1 \to 2))/2$. After one step then, the probabilities P_1 and P_2 change to P_1' and P_2' given by the matrix equation

$$\begin{pmatrix} P_1' \\ P_2' \end{pmatrix} = \begin{pmatrix} 1 - \frac{1}{2}P(1 \to 2) & \frac{1}{2} \\ \frac{1}{2}P(1 \to 2) & \frac{1}{2} \end{pmatrix} \begin{pmatrix} P_1 \\ P_2 \end{pmatrix}. \tag{7}$$

The steady-state solution is $P_1' = P_1$ and $P_2' = P_2$, so one easily finds the desired Boltzmann weighting

$$P_2 = P(1 \to 2)\,P_1 = \exp[-\beta\,(E_2 - E_1)]\,P_1. \tag{8}$$

Latent Heats of Vaporization of Argon and Krypton

I used a Monte Carlo method to compute the latent heats of vaporization of argon and krypton. The purpose of the computation was to test a hybrid form

$$V(r) = ae^{-b\,r}\,(1 - c\,r) - \frac{d}{r^6 + e\,r^{-6}}. \tag{9}$$

for the potential energy $V(r)$ of a pair of atoms separated by a distance r. I got the five parameters $a, \ldots, e$ by using Gnuplot to fit $V(r)$ to the best empirical data and to the best guesses of the quantum chemists.

Suppose $E\{R\}$ is the potential energy of a set R of N atoms,

$$E\{R\} = \sum_{i<j}^{N} V(|\vec{r}_i - \vec{r}_j|). \tag{10}$$

Then at temperature T, the potential energy U of the set is

$$U = \frac{\prod_{i=1}^{N} \int d^3 r_i \, E\{R\}\, e^{-\beta E\{R\}}}{\prod_{i=1}^{N} \int d^3 r_i \, e^{-\beta E\{R\}}}. \tag{11}$$

One Can Monte-Carlo Such MultiDimensional Integrals

For a cube of $N = 3000$ atoms, the integral (11) for U is a 9,000-dimensional integral, and the integrand involves $E\{R\}$ which is the complicated sum (10). Even an Intel computer with Itanium chips would have to run to the end of time to approximate this integral via the simple algorithm (4).

Instead one can use the Metropolis algorithm (6) to generate a sequence of $\mathcal{N}$ configurations R_n weighted by the Boltzmann factor. One then approximates the potential energy U by the average

$$U \approx \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} E\{R_n\}. \tag{12}$$

Each Metropolis step tried to change the position of a single atom; so successive configurations R_n differed at most by the location of a single atom. In my computation of the latent heats of Ar and Kr, the number $\mathcal{N}$ of configurations was well over a hundred million.

Results

The latent heat of vaporization is the difference between the potential energies U_{gas} and U_{liquid} plus the work done in expanding by ΔV against atmospheric pressure p

$$\Delta_{\mathrm{vap}} H = U_{\mathrm{gas}} - U_{\mathrm{liquid}} + p\,\Delta V. \tag{13}$$

The hybrid form V gave $\Delta_{\mathrm{vap}} H = 0.0694$ eV (per atom) for Ar and 0.0982 eV for Kr, which differ from the experimental values of 0.0666 and 0.0941 eV by 4.2% and 4.4%. In equivalent Monte Carlo simulations, the Lennard-Jones potential V_{LJ} fitted to r_0 and $V(r_0)$ gave and $\Delta_{\mathrm{vap}} H = 0.0787$ eV for Ar and 0.111 eV for Kr (errors of 18% and 18%). So the errors due to a lack of additivity are of the order of 4%, while those due to the Lennard-Jones potential are about 18%. Even in the liquid phase, limited additivity is less of a problem than the defects of the Lennard-Jones potential.

The code is publicly available at

http://bio.phys.unm.edu/~kevin/latentHeat/.

Acknowledgments

Thanks to Peter Cahill, Michael Gold, and Michael Malik for helpful conversations.