Lecture 23.

Genetic Algorithms.

Note Title 5/30/2006

Inspired by evolution. Represent potential solutions to a specific problem on a data structure like a chromosome.

Implementation

→ Initial population of random chromosones

→ Allocate reproduction opportunities so that chromosomes which represent a better solution to the target problem have more chance to reproduce.

Broader Sense. Any population-based model that uses selection and recombination operators to generate new samples in a search space.

Optimize: $F(x_1, \ldots, x_L)$

variables x_i are discrete (usually a power of two).

require evaluating $F(x_1, x_2, \ldots x_L)$ to be fast (because we need to do this for each member of the population.)

Size of the search space is large ~ 2^L

Canonical Genetic Algorithm:

Initialize strings at random

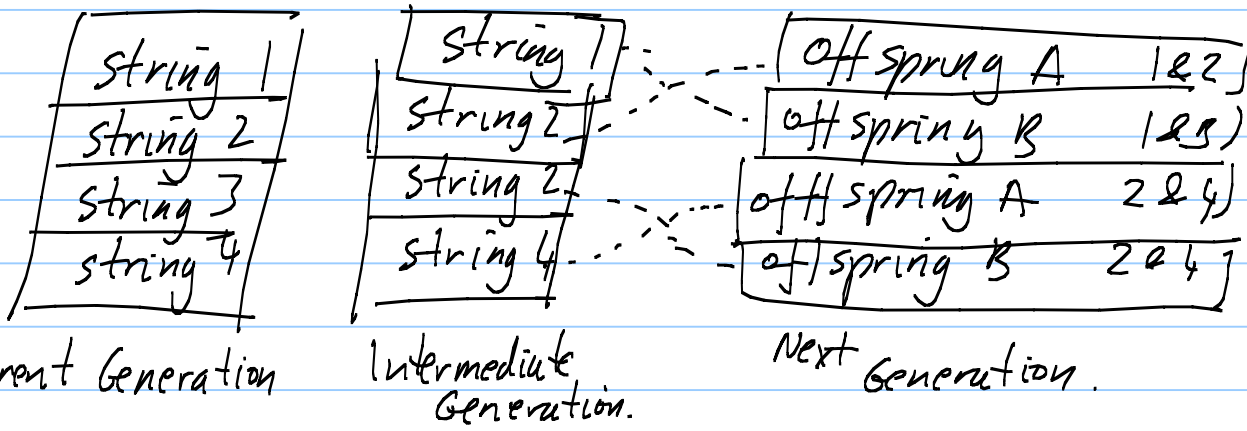
Evaluate each string and assign a fitness function.

Canonical fitness $\frac{F(x)}{\bar{F}}$ / Here $\bar{F}$ is the average for the population.

Execution in two stages:

(1) Current Population selection is applied to create an intermediate population.

(2) Recombination & Mutation are applied to the intermediate population to create the next population



Probability that strings are selected (duplicated) is proportional to their current fitness $\frac{F(\underline{x})}{\bar{F}}$.

Several Ways to do selection.

(1) Spin the Roulette Wheel
"stochastic sampling with replacement".

(2) "remainder stochastic sampling".
For each $\underline{x}$ s.t. $\frac{F(\underline{x})}{\bar{F}} > 1$

take the integer portion $\left[\frac{F(\underline{x})}{\bar{F}}\right]$ and place this number of copies in the intermediate population. All strings place additional copies with prob according to the fractional portion.
(e.g. $\frac{F(\underline{x})}{\bar{F}} = 1.36$ places one copy of $\underline{x}$ and another copy with probability 0.36)

Next recombination

Crossover is applied to randomly paired strings with probability P_c.

Recombine these strings to form two new strings.

E.G. $\quad 11010 \mid 01100101101$

$\qquad\quad yxyyx \mid yxxyyyxyxxy$

($x=1, y=0$ or $x=0, y=1$)

gives two offsprings

$11010\,yxxyyyxyxxy$

$yxyyx\,01100101101$

Mutation.

For each bit of the string, mutate with probability p_m $(p_m \approx 0.01)$

Output next population.

Why does this work?

It encourages strings with good fitness to reproduce — and discourages those with poor fitness.

Mutation is similar to standard MCMC — with mutation there is no interaction between strings.

Crossover & Recombination are new.

Schema Hypothesis

suppose a subpart of the problem can be encoded easily.

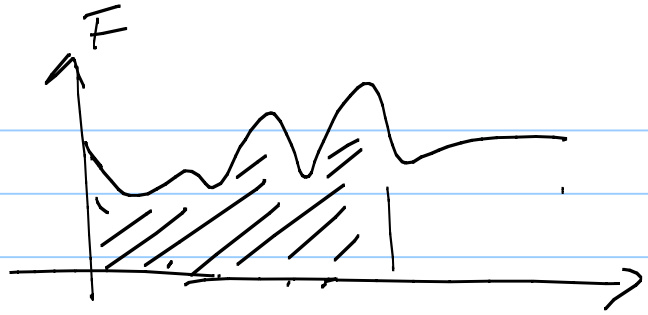
E.g. o 10101 ******* ← wildcards

a string with wildcards is called a schemata.

Low order hyperplanes are sampled by numerous strings in the population.

Implicit Parallelism — many hyperplanes computations are simultaneously solved in parallel.

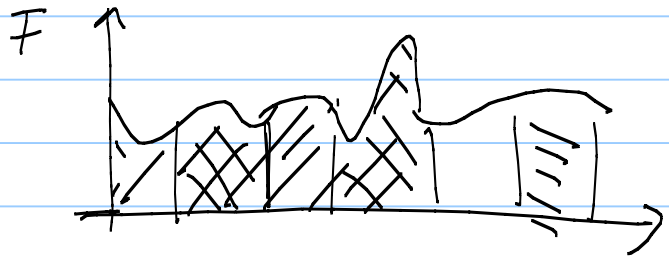
Alternative View.



0 * * ... * spans first half of space (shaded \\\)
1 * * ... * spans second half of space.

strings with 0 * * * ... * have on average better fitness that those 1 * * * ... *

So 0 * * * ... * will be encouraged.

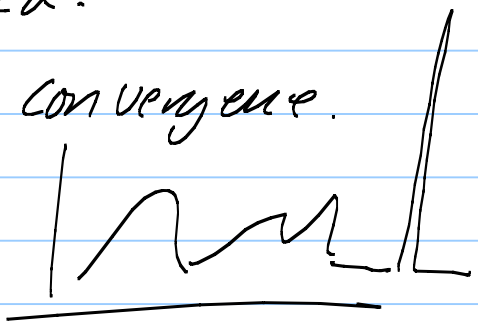


The strings with
* * 1 * * * * * *
are shaded \\\

Shows that 0 * 1 * * * * ... * strings are good.

Local optima do not affect this.
Crossover & Recombination allows good partial solutions to be selected.

But doesn't guarantee convergence.
E.g hard to find a narrow spike.



What does recombination do?

Order-1 hyperplanes (eg. $*y***.*$) are unaffected.

Order-2 hyperplanes

for 1-point crossover

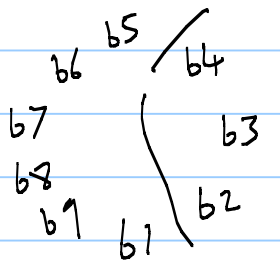
$1{*}{*}\ldots\ldots{*}$ → prob of bits seperated is $\frac{1}{L-1}$

$1{*}{*}\ldots\ldots{*}1$ → prob of bits seperated is $\frac{L-1}{L-1} = 1$.

Hence order is very important.

2-point crossover

- two randomly chosen crossover points.
- exchange the segment that falls between these two points.



(Note: 1-point crossover is a special case, where one crossover point always occurs between the first and last bits).

Bits which are close together are less likely to be affected by crossover.

Favours compact representations.

Define the length of a schema to be the distance between the first and last bit.

***1**0**10** length 7

The Schema theorem provides a lower bound on the change in sampling rate for a single hyperplane.

$M(H,t)$ be no. of strings sampling hyperplane H at time t.

$$M(H,t+1) = M(H,t)\,\frac{F(H,t)}{\bar{F}}$$

// Here $F(H,t)$ is the average fitness of strings in H.

$$M(H,t+1) = (1-p_c)\,M(H,t)\,\frac{F(H,t)}{\bar{F}} + p_c\left\{ M(H,t)\,\frac{F(H,t)}{\bar{F}}\,(1-\text{losses}) + \text{gains} \right\}$$

Conservation Assumption

crossover within the defining length of the schema is always disruptive.

E.G. schema $11**\ldots*$

If string 111010 were combined with 100000 or 010000 then no disruptions will occur.

Ignore gains

Also recombining 1000.. with 010...0 would give a gain.

Conservative Assumption:

ignore gains, treat all disruptions as losses.

then $M(H,t+1) \geq (1-p_c)\, M(H,t)\, \frac{F(H,t)}{\bar{F}} + p_c \left\{ M(H,t)\, \frac{F(H,t)}{\bar{F}} (1 - \text{disruptions}) \right\}$

Description is $\frac{\Delta(H)}{L-1} (1 - p(H,t))$

$\Delta(H)$ is the defining length associated with 1-point crossover.

($p(H,t)$ is the proportional representation of H obtained by dividing $M(H,t)$ by the population size)

Schema Theorem

$$p(H,t+1) \geq p(H,t)\, \frac{F(H,t)}{\bar{F}} \left\{ 1 - p_c \frac{\Delta(H)}{L-1} \left(1 - p(H,t) \frac{F(H,t)}{\bar{F}}\right) \right\}$$

Schema Theorem suggests that mutation is less important than crossover.

But mutation is necessary to prevent the system from getting stuck — e.g. 0.00.

Note: 1-point crossover is easy to model analytically.

But 2-point crossover is better for minimizing crossover disruption.

How effective are genetic algorithms?

Claim:

The standard genetic algorithm works poorly on most problems.

Non-standard genetic algorithms, which incorporate knowledge of specific problems work better.

Hybrids are good – combine existing algorithms with population methods.

Evolutionary Monte Carlo.

$\pi(\underline{x}) \propto e^{-E(\underline{x})}$, $\underline{x} \in \mathcal{X}$ sample space.

$\underline{X} = \{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_m\}$ population

m temperatures, $T = (t_1, t_2, \ldots, t_m)$

ordered $t_1 > t_2 > \ldots > t_{m-1} > t_m$.

Each chromosome $\underline{x}_i$ have a temperature t_i.

Gibbs for i^{th} chromosome is $\pi_i(\underline{x}_i) = \frac{1}{Z_i(t_i)} e^{-E(\underline{x}_i)/t_i}$.

$$\pi(\underline{x}) = \frac{1}{Z[T]} e^{-\sum_{i=1}^{m} E(\underline{x}_i)/t_i}. \qquad Z[T] = \prod_i Z_i(t_i).$$

$\underline{x}_i = (b_{i,1}, \ldots, b_{i,d})$ $b_{i,j} = 0$ or 1.

Mutation select $\underline{x}_k$ at random, mutate $\underline{x}_k$ to $\underline{y}_k$.

New population $\underline{Y} = \{\underline{x}_1, \ldots, \underline{y}_k, \ldots \underline{x}_N\}$

Accept with prob $\min(1, r_m)$ $r_m = e^{-(E(\underline{y}_k) - E(\underline{x}_k))/t_k}$

Crossover : $\underline{x}_i$ & $\underline{x}_j$ are selected to generate two offsprings $\underline{y}_i$ & $\underline{y}_j$

New population $\underline{Y} = \{\underline{x}_1, \ldots, \underline{y}_i, \ldots, \underline{y}_j, \ldots \underline{x}_m\}$,

accepted with probability $\min\{1, r_c\}$,

$$r_c = \exp\left\{ - \frac{(E(\underline{y}_i) - E(\underline{x}_i))}{t_i} - \frac{(E(\underline{y}_j) - E(\underline{x}_j))}{t_j} \right\} \frac{T(\underline{Y}|\underline{X})}{T(\underline{X}|\underline{Y})}$$

Transition

$$T(\underline{Y}|\underline{X}) = P[(\underline{x}_i, \underline{x}_j)|\underline{X}] \; P[(\underline{y}_i, \underline{y}_j)|(\underline{x}_i, \underline{x}_j)]$$

$P[(\underline{x}_i, \underline{x}_j)|\underline{X}]$: selection probability of $(\underline{x}_i, \underline{x}_j)$ from $\underline{X}$

$P[(\underline{y}_i, \underline{y}_j)|(\underline{x}_i, \underline{x}_j)]$: prob. of generating $\underline{y}_i \& \underline{y}_j$

e.g. could have

$$p((x_i, x_j)|\underline{X}) \propto \{ e^{-E(x_i)/t_i} + e^{-E(x_j)/t_j} \}, \quad x_i \neq x_j$$

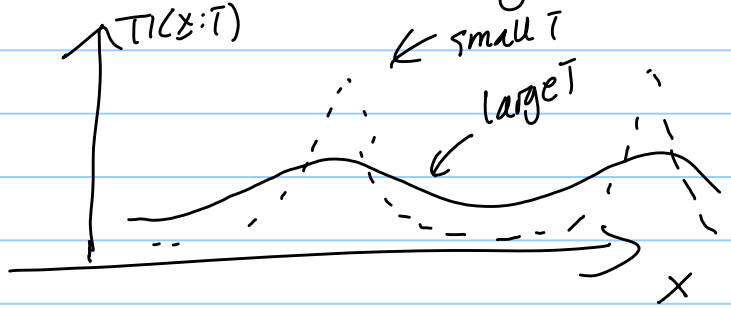
weighted samples.

Simulated Annealing

Want to sample from. $\pi(\underline{x}) = \frac{1}{Z} e^{-E(\underline{x})}$ ..

Introduce a temperature T, create a family of distributions $\pi(\underline{x}:T) = \frac{1}{Z(T)} e^{-E(\underline{x})/T}$

These all have the same peaks — at places where $E(\underline{x})$ is small — but distributions with larger T are smoother.



So it is usually quicker to sample $\pi(\underline{x}:T)$ for large T. (easier to sample a smooth distribution)

Key Idea of SA

Sample from $\pi(\underline{x}:T)$ for large T

e.g. by Metropolis-Hastings

after M iterations, $x^1, \ldots x^M$

reduce $T \to T\lambda$ $(\lambda < 1)$

and sample $x^{M+1}, \ldots x^{M+1}$ from $\pi(\underline{x}:T\lambda)$

repeat. until $T = 1$.

i.e. $T\lambda^n = 1$, n number of times you decrease T

Question - what value for M? what value of λ?

Answer - Problem dependant - find by experiments

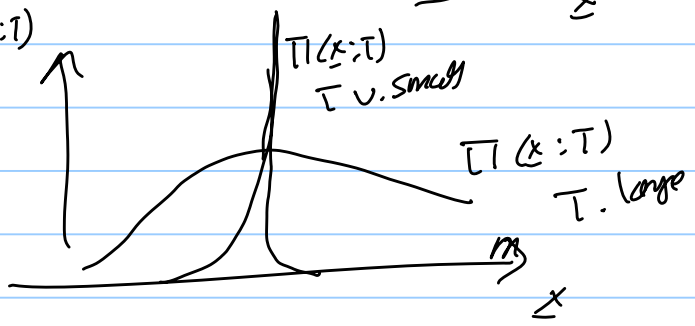
Page 14

You can also apply S.A. to find the lowest energy states → i.e. $\hat{x} = \arg\min_{x} E(x)$

$$\Pi(x:T) = \frac{1}{Z[T]} e^{-E[x]/T}$$

As $T \to 0$

$\Pi(x:T) = 0$, $x \neq \arg\min_{x} E(x)$



Sample from $\Pi(x:T)$

start with T-large

gradually reduce T

$T \to \lambda T$

until $T \approx 0$.

For M.H. → Proposal $T(x^{t+1}|x^t)$

Accept $\min\left\{1, \frac{\Pi(x^{t+1};T)}{\Pi(x^t;T)} \cdot \frac{T(x^t|x^{t+1})}{T(x^{t+1}|x^t)}\right\}$

$$\frac{\Pi(x^{t+1};T)}{\Pi(x^t;T)} = e^{-\{E(x^{t+1}) - E(x^t)\}/T}$$

So as T decreases you alter the acceptance probability.

Note: S.A. is useful for some problems. It is not as successful as originally hoped. Theoretical proofs exist which guarantee convergence – but these require extremely slow smoothing rates (λ) – so not useful in practice.