Theorem 10.6
Let $P = (p_{ij})$ and $P' = (p'_{ij})$ be regular transition matrices over finite state space S, with the same stationary distribution π. If $p_{ij} \geq p'_{ij}$ for all $i \neq j$, then

$$v(f, P, \pi) \leq v(f, P', \pi)$$

holds for all functions $f : S \to \mathbb{R}$.

Proof: E.g. Brémaud page 300. □

Corollary 10.7
For a given candidate-generation matrix Q, the Metropolis-Hastings algorithm has optimal asymptotic variance in the class of Hastings algorithms.

Proof: Since the α_{ij} are probabilities, the upper bound on s_{ij} given in condition (10) cannot be exceeded. The Metropolis-Hastings algorithm matches the upper bound. □

11 Genetic Algorithms

Genetic algorithms (GA) are a general-purpose "black-box" optimisation method proposed by J. Holland (1975) and K. DeJong (1975).

The method has attracted lots of interest, but its theory is still incomplete and the empirical results somewhat inconclusive. Advantages of the technique are that it is general-purpose, parallelisable, and adapts incrementally to changing cost functions ("on-line optimisation"). Disadvantages, on the other hand include that GA's are typically very slow – thus the technique should be used with moderation for simple serial optimisation of a stable, easily evaluated cost function.

Some claim that GA's typically require fewer function evaluations to reach comparable results as e.g. simulated annealing. Thus the method may be good when function evaluations are expensive (e.g. require some acutal physical measurement).

11.1 The Basic Algorithm

We consider the so called "simple genetic algorithm"; also many other variations exist.

Assume we wish to maximise a cost function c defined on n-bit binary strings:

$$c : \{0,1\}^n \to \mathbb{R}.$$

Other types of domains must be encoded into binary strings, which is a nontrivial problem. View each of the candidate solutions $s \in \{0,1\}^n$ as an *individual* or *chromosome*. At each stage (*generation*) t the algorithm maintains a *population* of individuals $p_t = (s_1, \ldots, s_m)$.

There are three operations defined on populations:

- *selection* $\sigma(p)$ (“survival of the fittest”)
- *recombination* $\rho(p)$ (“mating”, “crossover”)
- *mutation* $\mu(p)$

The *Simple Genetic Algorithm* is then as follows:

```
function SGA($\sigma$, $\rho$, $\mu$):
    $p \leftarrow$ random initial population;
    while $p$ “not converged” do
        $p' \leftarrow \sigma(p)$;
        $p'' \leftarrow \rho(p')$;
        $p \leftarrow \mu(p'')$
    end while;
    return $p$ (or “fittest individual” in $p$).
end.
```

Selection

Denote $\Omega = \{0,1\}^n$. The selection operator $\sigma : \Omega^m \to \Omega^m$ maps populations probabilistically: given an individual $s \in p$, the expected number of copies of s in $\sigma(p)$ is proportional to the *fitness* of *s in p*. This is a function of the cost of s compared to the costs of other $s' \in p$.

Some possible fitness functions are:

- Relative *cost* ($\Rightarrow$ “canonical GA”):

$$f(s) = \frac{c(s)}{\frac{1}{m}\sum_{s' \in p} c(s')} \triangleq \frac{c(s)}{\bar{c}}.$$

- Relative *rank*:

  $$f(s) = \frac{r(s)}{\frac{1}{m}\sum_{s' \in p} r(s')} = \frac{2}{m+1} \cdot r(s),$$

  where $r(s)$ is the rank of individual s in a worst-to-best ordering of all $s' \in p$.

Once the fitness of individuals has been evaluated, selection can be performed in different ways:

- *Roulette-wheel selection* ("stochastic sampling with replacement"):
  - Assign to each individual $s \in p$ a probability to be selected in proportion to its fitness value $f(s)$. Select m individuals according to this distribution.
  - Pictorially: Divide a roulette wheel into m sectors of width proportional to $f(s_1), \ldots, f(s_m)$. Spin the wheel m times.
- *Remainder stochastic sampling*:
  - For each $s \in p$, select deterministically as many copies of s as indicated by the integer part of $f(s)$. After this, perform stochastic sampling on the fractional parts of the $f(s)$.
  - Pictorially: Divide a fixed disk into m sectors of width proportional to $f(s_1), \ldots, f(s_m)$. Place an outer wheel around the disk, with m equally-spaced pointers. Spin the outer wheel once.

Recombination

Given a population p, choose two random individuals $s, s' \in p$. With probability p_ρ, apply a *crossover operator* $\rho(s, s')$ to produce two new offspring individuals t, t' that replace s, s' in the population. Repeat the operation $m/2$ times, so that on average each individual participates once. Denote the total effect on the population as $p' = \rho(p)$. (A practical implementation: choose $\frac{p_\rho}{2} \cdot m$ random pairs from p and apply crossover deterministically.) Typically $p_\rho \approx 0.7 \ldots 0.9$.

Some possible crossover operators are illustrated in Figure 1.

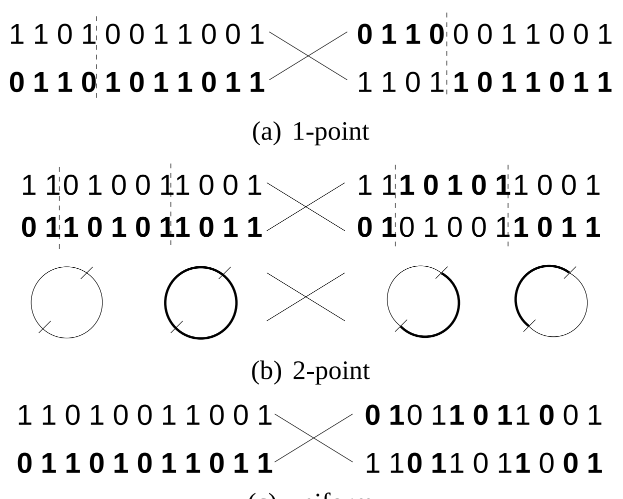

Figure 1: Typical crossover operators.

Mutation

Given population p, consider each bit of each individual and flip it with some small probability p_μ. Denote the total effect on the population as $p' = \mu(p)$. Typically, $p_\mu \approx 0.001 \ldots 0.01$. It appears that for n-bit strings a good choice is $p_\mu = 1/n$.

Theoretically mutation is disruptive. Recombination and selection should take care of optimisation; mutation is needed only to (re)introduce “lost alleles”, alternative values for bits that have the bits that have the same value in all current individuals.

In practice mutation plus selection equals local search. Mutation, even with quite high values of p_μ, can be efficient and is often more important than recombination.

Analysis of GA’s: Hyperplane sampling

The notion of hyperplane sampling presents a heuristic view of how a genetic algorithm works.

A *hyperplane* (actually subcube) is a subset of $\Omega = \{0,1\}^n$, where the values of some bits are fixed and other are free to vary. A hyperplane may be represented by a *schema* $H \in \{0,1,*\}^n$. E.g. the schema ’$0*1**$’ represents the 3-dimensional hyperplane (subcube) of $\{0,1\}^5$ where bit 1 is fixed to 0, bit 3 is fixed to 1, and bits 2, 4, and 5 vary.

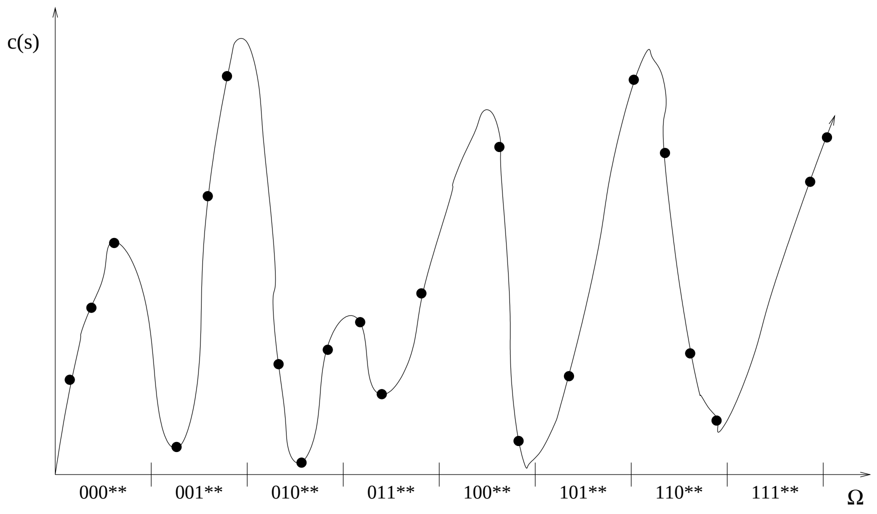

Figure 2: A population sampling hyperplanes.

An individual $s \in \{0,1\}^n$ *samples* hyperplane H, or *matches* the corresponding schema if the fixed bits of H match the corresponding bits in s. BY some abuse of notation, this situation is denoted as "$s \in H$". Note that a given individual generally samples many hyperplanes simultaneously, e.g. individual '101' samples '$10*$', '$1*1$', etc.

Define the *order* of a hyperplane H as:

$$\begin{aligned} o(H) &= \text{number of fixed bits in } H \\ &= n - \dim H. \end{aligned}$$

The *average cost* of hyperplane H is then:

$$c(H) = \frac{1}{2^{n-o(H)}} \sum_{s \in H} c(s).$$

Denoting by $m(H,p)$ the number of individuals in population p that sample hyperplane H, the *average fitness* of hyperplane H in population p is defined as:

$$f(H,p) = \frac{1}{m(H,p)} \sum_{s \in H \cap p} f(s,p)$$

A heuristic claim is then that selection drives the search towards hyperplanes of higher average cost (quality).

Consider e.g. the cost function and partition of Ω into hyperplanes (in this case, intervals) of order 3 presented in Figure 2. Here the current population of 21

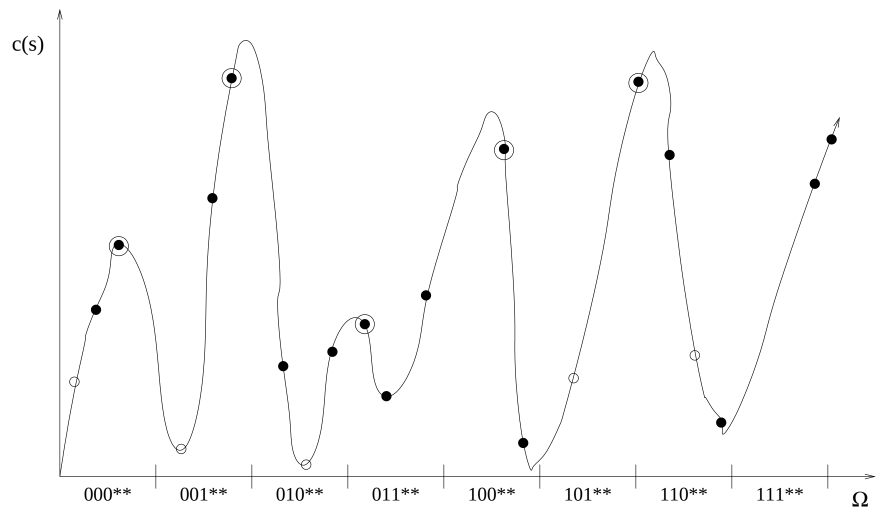

Figure 3: A sampling population after selection.

individuals samples the hyperplanes so that e.g. '$000**$' and '$010**$' are sampled by three individuals each, and '$100**$' and '$101**$' by two individuals each. Hyperplane '$010**$' has a rather low average fitness in this population, whereas '$111**$' has a rather high average fitness.

The result of e.g. roulette wheel selection on this population might lead to elimination of some individuals and duplication of others, as presented in Figure 3.

Then, in terms of expected values, one can show that

$$E[m(H,\sigma(p))] = m(H,p)\cdot f(H,p).$$

The effect of crossover on schemata

Consider a schema such as

$$H = **\underbrace{11**01*1}_{\Delta(H)=7}**$$

and assume that it is represented in the current population by some $s \in H$.

If s participates in a crossover operation and the crossover point is located between bit positions 3 and 10, then with large probability the offspring are no longer in H. In this case schema H is said to be *disrupted*. On the other hand, if the crossover point is elsewhere, then one of the offspring stays in H, and H is *retained*.

Generally, the probability that in 1-point crossover a schema $H = \{0,1,*\}^n$ is retained, is (ignoring the possibility of "lucky combinations")

$$\Pr(\text{retain } H) \approx 1 - \frac{\Delta(H)}{n-1},$$

where $\Delta(H)$ is the *defining length* of H, i.e. the distance between the first and last fixed bit in H.

More precisely, if H has $m(H,p)$ representatives in population p of total size m:

$$\Pr(\text{retain } H) \geq 1 - \frac{\Delta(H)}{n-1}\left(1 - \frac{m(H,p)}{m}\right).$$

The Schema "Theorem"

The Schema Theorem, proposed by J. Holland (1975), provides a Heuristic estimate of the changes in representation of a given schema H from one generation to the next.

Denote:

$$m(H,t) = \text{number of individuals in population at generation } t \text{ that sample } H.$$

Then:

(i) Effect of selection:

$$m(H,t') \approx m(H,t) \cdot f(H)$$

(ii) Effect of recombination:

$$\begin{aligned} m(H,t'') &\approx (1-p_\rho)m(H,t') + p_\rho\left(m(H,t')\Pr(\text{retain } H) + \underbrace{m \cdot \Pr(\text{luck})}_{\geq 0}\right) \\ &\geq (1-p_\rho)m(H,t') + p_\rho m(H,t')\left(1 - \frac{\Delta(H)}{n-1}\left(1 - \frac{m(H,t')}{m}\right)\right) \\ &= m(H,t')\left(1 - p_\rho\frac{\Delta(H)}{n-1}\left(1 - \frac{m(H,t')}{m}\right)\right) \end{aligned}$$

(iii) Effect of mutation:

$$m(H,t+1) \approx m(H,t'') \cdot (1-p_\mu)^{o(H)}$$

In summary, then:

$$m(H,t+1) \gtrsim m(H,t)\cdot f(H)\cdot\left(1-p_\rho\frac{\Delta(H)}{n-1}\left(1-\frac{m(H,t')}{m}\right)\right)\cdot(1-p_\mu)^{o(H)}.$$

The formula leads to so called *"Building Block Hypothesis"*: In a genetic search, short, above-average fitness schemata of low order ("building blocks") receive an exponentially increasing representation in the population.

The following criticisms have been expressed as regards the "Schema Theorem" and the Building Block Hypothesis, however:

- Many of the approximations used in deriving the "Schema Theorem" implicitly assume that the population is very large. In particular, it is assumed that all the relevant schemata are well sampled. This is clearly not possible in practice, because there are 3^n possible schemata of length n.
- The result cannot be used to predict the development of the population for much more than one generation, because:
  - firstly, the long-term development depends on the coevolution of the schemata, and the "theorem" considers only one schema in isolation;
  - secondly, an "exponential growth" cannot in any case continue for long in a finite population.

11.2 Genetic Algorithms as Stochastic Processes

A proper mathematical treatment of GA's would view them as stochastic processes. It is unfortunately very difficult to obtain any nontrivial analytical results in this direction. Here we outline a simple Markov chain model presented by Vose & Liepins (1991) and Rudolph (1994).

Consider the "canonical GA", i.e. the Simple Genetic Algorithm using the relative cost fitness function and standard proportional ("roulette-wheel") selection, in the form:

```
$p \leftarrow$ random initial population;
$p \leftarrow \sigma(p)$;            (selection)
**while** $p$ "not converged" **do**
    $p' \leftarrow \rho(p)$;         (recombination)
    $p'' \leftarrow \mu(p')$         (mutation)
    $p \leftarrow \sigma(p'')$;      (selection)
**end while**.
```

Encode a population of m individuals, each an n-bit string, as an integer (in binary representation)

$$p \in \{0,1\}^{mn} \equiv \underbrace{\{0,1,\dots,2^{mn}-1\}}_{\mathbb{Z}_{2^{mn}}}.$$

Then the CGA can be modeled as a Markov chain on state space $\mathbb{Z}_{2^{mn}}$, with the transitions probability matrix $P = CMS$, where

C	is the recombination (“crossover”) transition probability matrix
M	is the mutation transition probability matrix
S	is the selection transition probability matrix

A stochastic matrix $P = (p_{ij})$ is:

(i) *positive*, if $p_{ij} > 0$ for all i, j;

(ii) *primitive*, if P^k is positive for some $k \geq 0$;

(iii) *reducible*, if it can be converted to the form

$$\tilde{P} = \begin{bmatrix} C & 0 \\ R & T \end{bmatrix},$$

where C and T are square matrices, by applying the same permutation to the rows and the columns;

(iv) *irreducible*, if it is not reducible.

The interpretation of these definitions is that primitive matrices correspond to the irreducible and aperiodic Markov chains defined before. In a reducible matrix, the upper rows correspond to a “closed” or “absorbing” class of states, the lower rows to “transient” states. Note that a positive matrix is trivially primitive.

Theorem 11.1
Let P be a primitive stochastic matrix. Then the sequence P^k converges as $k \to \infty$ to a stochastic matrix P^∞ which has the form

$$P^\infty = \begin{bmatrix} p^\infty \\ \vdots \\ p^\infty \end{bmatrix},$$

where p^∞ is a stochastic vector with all components positive. (The vector p^∞ represents the stationary distribution of the chain.)

Theorem 11.2
Let P be a reducible stochastic matrix of the form

$$P = \begin{bmatrix} C & 0 \\ R & T \end{bmatrix},$$

where C is primitive, and T does not contain an irreducible submatrix. Then the sequence P^k converges as $k \to \infty$ to a stochastic matrix P^∞ of the form

$$P^\infty = \begin{bmatrix} p^\infty & 0 \\ \vdots & \vdots \\ p^\infty & 0 \end{bmatrix},$$

where p^∞ is a stochastic vector with all components positive.

Lemma 11.3
The transition probability matrix $P = CMS$ of the "canonical genetic algorithm", with mutation probability $0 < p_\mu < 1$ is positive and hence primitive.

Proof: Denote $C = (c_{ik}), M = (m_{kl}), S = (s_{lj})$. Then $P = (p_{ij})$, where

$$p_{ij} = \sum_{kl} c_{ik} m_{kl} s_{lj}.$$

Observe:

(i) $\forall\, i \exists\, k_i : c_{ik_i} > 0$ (Because C is stochastic $\Rightarrow \forall\, i : \sum_k c_{ik} = 1$)

(ii) M is positive: denote $N = mn$, $d(k,l)$ = Hamming distance between populations k,l. Then:

$$m_{kl} = p_\mu^{d(k,l)} \cdot (1 - p_\mu)^{N - d(k,l)} > 0.$$

(iii) $\forall\, j : s_{jj} > 0$ (Because with nonzero probability, selection does not change the population.)

Thus:

$$p_{ij} = \sum_{kl} c_{ik} m_{kl} s_{lj} \geq c_{ik_i} m_{k_i j} s_{jj} > 0. \;\square$$

Theorem 11.4
The CGA with mutation probability $0 < p_\mu < 1$ converges to a stationary distribution of populations where the probability of every population is nonzero.

Proof: Follows from Theorem 11.1 and Lemma 11.3. □

Assume the CGA is defined so as to maximize the function $c : \{0,1\} \to \mathbb{R}$. Denote

$$c^* = \max\{c(i) \mid i \in \{0,1\}^n\},$$

and for a population $\hat{i} = (i_1, \ldots, i_m)$:

$$c^*(\hat{i}) = \max\{c(i_k) \mid k = 1, \ldots, m\}.$$

Denote by $\hat{i}^{(t)}$ the population of the CGA at time t. The algorithm *converges to global optimum* if

$$\lim_{t\to\infty} \Pr(c^*(\hat{i}^{(t)}) = c^*) = 1.$$

Note that the simulated annealing algorithm converges to global optimum in exactly this sense.

Corollary 11.5
If nonoptimal solutions with respect to the cost function c exist (i.e. if $c(j) < c^*$ for some $j \in \{0,1\}^n$), then the CGA does not converge to the global optimum.

Proof: Let $\hat{j} = (j, j, \ldots, j)$ be a population such that $c^*(\hat{j}) < c^*$ By Theorem 11.2,

$$\lim_{t\to\infty} \Pr(\hat{i}^{(t)} = \hat{j}) = \varepsilon > 0,$$

and thus

$$\lim_{t\to\infty} \Pr(c^*(\hat{i}^{(t)}) = c^*) \leq 1 - \varepsilon < 1. \ \square$$

Theorem 11.6
On the other hand, if the best solution found is always kept in the population ("elitist" selection) and not mutated, then the CGA does converge to the global optimum.

Proof: Simple corollary to Theorem 11.2: the transition probability matrix P reduces in this case to the form

$$P = \begin{bmatrix} C & 0 \\ R & T \end{bmatrix},$$

where the upper rows correspond to the unique closed class of populations containing a globally optimal solution. □

Note that for practical purposes, such (non)convergence results are of course largely irrelevant. The important (but difficult) questions are:

- How fast does the CGA with elitist selection converge towards an optimal solution?
- Does the CGA without elitist selection converge to a population with mostly optimal solutions, and how fast?

12 Combinatorial Phase Transitions

12.1 Phenomena and Models

“Where the Really Hard Problems Are” (Cheeseman et al. 1991)

Many NP-complete problems can be solved in polynomial time “on average” or “with high probability” for reasonable-looking distributions of problem instances. E.g. Satisfiability in time $O(n^2)$ (Goldberg et al. 1982), Graph Colouring in time $O(n^2)$ (Grimmett & McDiarmid 1975, Turner 1984).

Where, then, are the (presumably) exponentially hard instances of these problems located? Could one tell ahead of time whether a given instance is likely to be hard?

Early studies of this issue done by: Yu & Anderson (1985), Hubermann & Hogg (1987), Cheeseman, Kanefsky & Taylor (1991), Mitchell, Selman & Levesque (1992), Kirkpatrick & Selman (1994), etc.

Hard Instances for 3-SAT

Mitchell, Selman & Levesque (AAAI 1992).

Experiments on the behaviour of the Davis-Putnam[-Logemann-Loveland] (DP[LL]) procedure on randomly generated 3-cnf Boolean formulas.

E.g. satisfiable 3-cnf formula

$$(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_4)$$

The expressions in parenthesis are *clauses* and the x’s are *literals*.

Distribution of test formulas:

- number of variables
- $m = \alpha n$ randomly generated clauses of 3 literals, $2 \leq \alpha \leq 8$