

Linear and Combinatorial Optimization

Fredrik Kahl

Matematikcentrum

Lecture 12: Genetic Algorithms

- Introduction
- Idea
- Survival statistics
- The two-armed bandit
- Implicit parallelism
- Implementation and examples

Genetic algorithms

Algorithm is based on ideas from nature:

- The strong survives
- Reproduction
- Mutation

We assume during the lecture that the problem is formulated as a maximization problem with a positive objective function.

One has to be able to represent feasible solutions as strings.

$$\max_{x \in D} f(x)$$

Example

D - set of 5-bit strings

Each string is binary coding of an integer.

Objective function is $f = x^2$.

	string	x	fitness	of total
1	01101	13	169	0.144
2	11000	24	576	0.492
3	01000	8	64	0.055
4	10011	19	361	0.309
<hr/>				
	Σ		1170	1

Consider the string as individuals in a small population.

Let the strings reproduce proportionally to their objective function.

Genetic algorithms

- Consider the objective function for each string in the population.
- Let strings reproduce proportionally to their objective function.
- At reproduction, two parent strings are mixed randomly together - crossover.
- In addition, there is a small risk of individual bits being mutated.
- Repeat with next generation.

Crossover

After reproduction, one proceeds with crossover - the strings of two parents are mixed.

An integer position k is randomly chosen. First, copy parents and then swap all bits between $k + 1$ and the length of the string.

Example: String nr 1 and 2 reproduce. Swapping occurs after bit nr 4.

Parent 1	Child *
01101	01100
****-	

Parent 2	Child -
11000	11001
-----*	

**What information is contained
in a population?**

string	fitness
01101	169
11000	576
01000	64
10011	361

Schemata

Singular: Schema

Introduce a new letter in the alphabet ‘*’ - don’t care symbol.

Schema 1*000 = the set {10000, 11000}

How many schemata are there?

3^5

Consider two strings

1***0 and **11*

which one has highest probability of surviving to the next generation?

More about schemata

Define the **length** $\delta(H)$ of schema H as the distance between the first and last specific string position.

Example:

$$\delta(*10 * *1 * **) = 4$$

$$\delta(*10 * * * * * *) = 1$$

Define the **order** $o(H)$ of a schema H as the number of specific string positions.

Example:

$$o(*10 * *1 * **) = 3$$

$$o(*10 * * * * * *) = 2$$

Survival statistics

Suppose there is a large population with $m(H, t)$ strings of the schema H at generation t .

How many of these survive?

Each string x_i is reproduced on the average $n \frac{f(x_i)}{\sum_j f(x_j)}$ times.

Let $\bar{f} = \frac{1}{n} \sum_{j=1}^n f(x_j)$ be the mean of the objective function and $f(H)$ be the mean of the objective function for strings with H .

Then, an estimate of the number of string with H in the next generation is

$$m(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}}$$

Survival statistics

Suppose that certain schemata are better than average

$$f(H) = (1 + c)\bar{f}$$

then

$$m(H, t + 1) = m(H, t) \frac{f(H)}{\bar{f}} = (1 + c)m(H, t)$$

We get an exponential growth

$$m(H, t) = (1 + c)^t m(H, 0)$$

NB: This works only a few generations.

What schemata transfer to the children?

If the cut is outside the schema's specific letters then the schema is transferred to its children.

Otherwise the schema can be destroyed.

Example

$x_i = 011|1000$

$H_1 = *1*|***0$

$H_2 = ***|10**$

Let l denote the string length.

Then the cut is in the schema's specific part with probability

$$\frac{\delta(H)}{l-1}$$

The survival probability of a schema can be estimated with

$$m(H, t+1) = m(H, t) \frac{f(H)}{\bar{f}} \left(1 - \frac{\delta(H)}{l-1} \right)$$

Mutation

In addition, each bit has a small probability p_m of being mutated in each step. Thus, the probability that no schema specific part is mutated is

$$(1 - p_m)^{o(H)} \approx (1 - o(H)p_m)$$

If higher order terms are neglected, we get

$$m(H, t+1) = m(H, t) \frac{f(H)}{\bar{f}} \left(1 - \frac{\delta(H)}{l-1} - o(H)p_m \right)$$

CONCLUSION:

Short schemata that are better than average and of lower order grow exponentially in future generations.

The two-armed bandit

Suppose we have a two-armed bandit where one can bet 1 krona on each arm.

One arm gives profit $N(m_1, \sigma)$ and the other arm $N(m_2, \sigma)$, ($m_1 > m_2$), but one doesn't know which one.

Suppose we are going to play N times

How should one maximize profit?

Idea: Use a small number n of tests on each arm. Then make a decision on which arm is best and then bet the rest of the money on that arm.

Let the outcome of the left arm be x_1, \dots, x_n and the right arm y_1, \dots, y_n . Form

$$Q = \bar{m}_L - \bar{m}_H = \frac{1}{n} \sum x_i - \frac{1}{n} \sum y_i$$

Then $Q = N(m_L - m_H, \sigma\sqrt{2}/\sqrt{n})$

Choose the arm with best average.

How should one choose the optimal trial period?

Analysis of this problem gives that one should bet exponentially many games $N \approx C_1 e^{C_2 n}$ on the best arm after n starting trials.

Analogy with genetic algorithms: At the same time, one examines a large number of schemata and one chooses exponentially many more schemata with high fitness.

Implicit parallelism

In each generation of genetic algorithms, n strings are examined. At the same time, a much larger number of schemata are examined automatically.

This is called **implicit parallelism**.

How many schemata are processed usefully?

* Long schemata are seldom reproduced!

Consider only schemata that are often $(1 - \epsilon)$ reproduced.

$$p_s \approx 1 - \frac{\delta(H)}{l - 1} > 1 - \epsilon$$

$$\epsilon > \frac{\delta(H)}{l - 1} \Rightarrow (l - 1)\epsilon > \delta(H)$$

Set the maximal length to l_s .

The number of schemata n_s of length no longer than l_s in each string is

$$2^{l_s - 1} (l - l_s + 1)$$

In the whole population, we get at most

$$n2^{l_s-1}(l - l_s + 1)$$

However, we get a lot of duplicates of low order.

If there are $n = 2^{l_s/2}$ strings in the population, then there are not so many duplicate strings of order higher than $l_s/2$. If we count only high order length between $l_s/2$ and l_s , we get

$$n2^{l_s-2}(l - l_s + 1)$$

that is, with $n = 2^{l_s/2}$

$$n_s = \frac{(l - l_s + 1)n^3}{4} = Cn^3$$

The building block hypothesis

Consider genetic algorithms as way of automatically processing blocks of substrings and recombine them to better solutions.

Minimal deceptive problem

What happens if 11 is optimal, but all other smaller schemata indicate zeroes?

$$0^* > 1^* \text{ or } *0 > *1$$

That is,

$$f(00) + f(01) > f(10) + f(11)$$

$$f(00) + f(10) > f(01) + f(11)$$

If $f(11) > f(00)$ then

$$f(01) > f(10)$$

$$f(10) > f(01)$$

If one of the two inequalities hold, then the problem is deceptive. There are two cases:

Particulars

- **Scaling:** It may be advantageous to rescale the objective function. For small populations, often one uses $f_{max} \approx 2f_{average}$.
- Dominant and recessive genes
- Several sexes.
- Different strategies for genetic optimization.
- Tribes with different strategies, migration.

Implementation

1. Initiate generation 1 randomly.
2. Create next generation through:
 - (a) Randomly select parents according to the objective function (fitness).
 - (b) Compute children through crossover and mutation.
 - (c) Compute the objective function for the children.
3. Go to step 2 if the stop criterion is not fulfilled.

Vigenere cipher

The key is short string, for example, 'HELLO'.

At encryption and decryption, the key is written periodically below the text. Each letter in the key gives the number of (cyclic) shifts for the above letter in the alphabet

a -> b -> c -> ... -> y -> z -> a -> b ...

A means no shift, B one step, C two steps etc.

Example

abcdefghijklmnopqrstuvwxyz

this text should be encrypted

HELLOHELLOHELLOHELLOHELLO

altdhlbedvvywopllynftepr

matlab-kod

```
%Initiera
N=nalfabet*period*4;
gamla=zeros(N,period);
fgamla=zeros(N,1);
for i=1:N,
    kod=randomindomain(problem);
    gamla(i,:)=kod;
    fgamla(i)=min(max(1-evaluate(problem,kod),0.1),1);
end;

[fmax,maxi]=max(fgamla);
xmax=gamla(maxi,:);
```

matlab-kod

```
for t=1:nr_of_generations;
    lotteri=cumsum(fgamla);
    lotteri=lotteri/sum(fgamla);
    nya=zeros(N,period);
    fnya=zeros(N,1);
    ii=0;

    for i=1:N/2,
        foralder1=gamla(sum(lotteri<rand)+1,:);
        foralder2=gamla(sum(lotteri<rand)+1,:);
        [barn1,barn2] = breed(problem,foralder1,foralder2);
        fbarn1=min(max(1-evaluate(problem,barn1),0.1),1);
        fbarn2=min(max(1-evaluate(problem,barn2),0.1),1);
        ii=ii+1;
        nya(ii,:)=barn1;
        fnya(ii,:)=fbarn1;
        ii=ii+1;
        nya(ii,:)=barn2;
        fnya(ii,:)=fbarn2;
    end;
```

```
fmin=min(fnya);  
fmean=mean(fnya);  
fstd=std(fnya);  
  
[t fmin fmean tmax fstd fmax]  
res=[res; t fmin fmean tmax fstd fmax];  
  
gamla=nya;  
fgamla=fnya;  
end;
```

sdxmmhär

zähh oqb

hxxövåca

mehtrvub

åbqulykg

yzsfqiii

cgvkegvc

faevshbs

faeh oqb

zähvshbs

faevshvc

cgvkegbs

fdxmmhär

saevshbs

mehtrvub

zähh oqb

Resultat

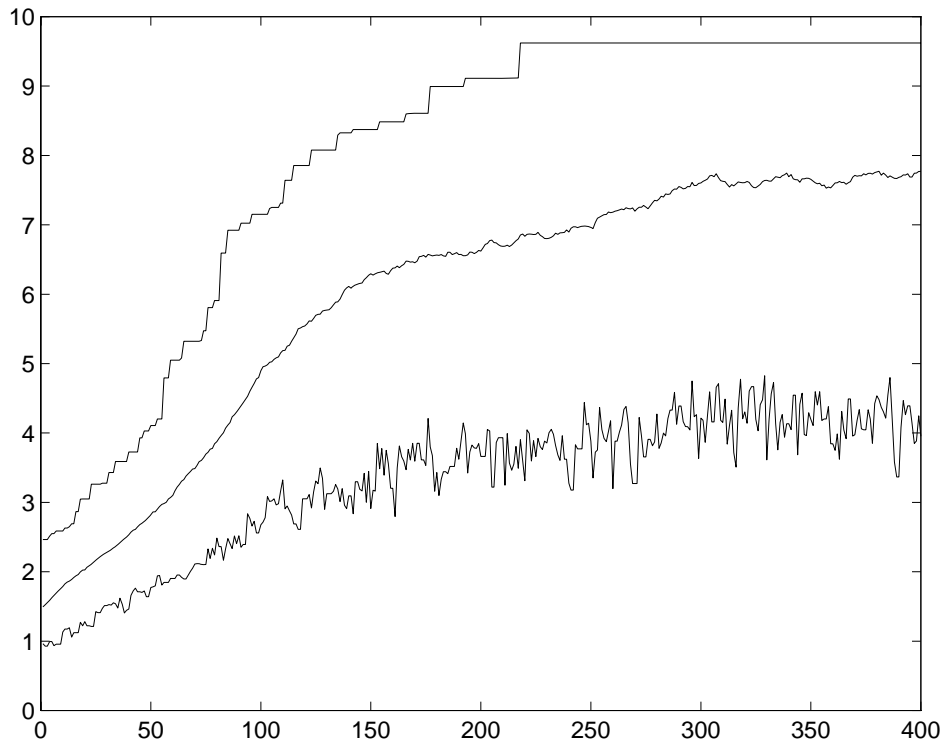
Exempel 1: Text med 200 bokstäver. Nyckel på 4 bokstäver. Alfabet med 30 bokstäver.

t	fmin	fmedel	fmax	fstd	totalmax
1	0.53	1.7018	4.7282	0.6563	4.7282
2	0.72	1.9231	5.5949	0.7805	5.5949
3	0.71	2.2287	5.5949	0.9491	5.5949
4	0.75	2.5825	7.1498	1.0967	7.1498
5	0.75	3.0904	7.1498	1.2335	7.1498
6	0.95	3.5107	7.1498	1.2409	7.1498
7	1.26	3.9777	7.1498	1.3030	7.1498
8	1.26	4.4479	7.1498	1.2314	7.1498
9	1.60	4.7831	7.1498	1.1794	7.1498
10	1.62	5.0292	9.8270	1.2037	9.8270
11	2.25	5.3183	9.8270	1.2417	9.8270
12	2.25	5.5270	9.8270	1.2638	9.8270

Kod: komb Poang: 9.827 Text: två grannar
jag har i min boning den ene är
sentimental jag hör honom högt
deklamera om sorg och livets kval

Resultat

Exempel 2: Text med 2000 bokstäver. Nyckel på 24 bokstäver. Alfabet med 30 bokstäver. Antal tillåtna lsg: $30^{24} = 2.824310^{35}$.



Del av befolkningen generation 1.

pleqrwxsyqnyqjomooäkiyrv
poinvrtönaöbjdwihzmyzeju
g up kå xivöäkqyhykyroüz
deeofazcdkpikermödrbrlix
slzajprta mäömgcåånpqåoh
fönpzhexejyqjfenyiecemd
guesijpfzök dwibmzäöhuj
jawvzfyäötonfteozwguäyhs
xrjjåfrmpxdäybxäyukqrix
qxasösjgqustbuljbpåszgtv
åiavöewonkj kjxhnuåbc po
ölxleyswjyyäeåqoflksunvq
pdimö huäblomc evqåcäqqå
qhdwgbtk ö jarxmrypvwägy
qajq mpitgfiht ölrt eözl
vu åpüzgmätwözgrqäomfynä
hrs antobbulssd laucu xå
xhsvqicljvznrslipnqsvö
srvdhnhtlsdbqcdehqlqöay
t cälyyocsntjbccwqjtcöuh

Del av befolkningen generation 400.

omginatokink optimering
kombinatorisk oktimpldfnd
kombinatorisknoptimering
kombinatorisk optimerivg
kombinatoriskhoptmmering
kombinatorisk optimering
kombinatorisk optimering
kombinatorisk oprimering
kombinatorisk oprimering
kombinatorisk optimwrsnq
kombinatorisk optimering
kombinatorisk ogtimerifg
kombinatorisk opåiperung
kombinatorisk optimezing
kombinatorisk optimaring
kombinatorisk optimering
kombinatoriskåoptimeriig
kombinatorisk optimering
kombinatorisk optimerind
kombinatorisk wttimering

Den bästa lösningen

Kod: kombinatorisk optimering

Poang: 9.72

Text: två grannar jag
har i min boning den
ene är sentimental jag
hör honom högt
deklamera om sorg
och livets kval

...

lejon lurar hon i sitt
förstäck och såsom en
modig rövarska

Conclusions

- Ideas from nature
- Hard to analyse
- Easy to implement
- Room for improvements - creative ideas welcome