Centrum Wiskunde & Informatica

Genetic Algorithms (II)

WISM454 Laboratory Class Scientific Computing, Jan-Willem Buurlage

May 1, 2019

Genetic Algorithms

Summary of GAs

- A population of candidate solutions evolves over time.
- The current set of candidates at some iteration is called a generation.
- This generation evolves to the next generation using three operations
  - Selection
  - Combination
  - Mutation
- Solutions in a generation are also called members.
- Members are often represented by bitstrings, in this case also called chromosomes.

Idea of Genetic Algorithms

- Each chromosome can be seen as an artificial creature, struggling to survive.
- Evolution in nature leads to stronger species, and we try to mimic this process in our algorithm.
- Fit creatures survive, mate, and undergo mutation.
- This setting also allows for adaptivity, if the objective function changes, evolution will adapt the current generation. In this sense, GAs are robust.

Fitness

- The fitness of a member x_i, is a measure for its relative performance in the current generation.
- The fitness is always a function of the objective value $f(x_i)$.

1. Subtract maximum (if known):

$$F(x) = f(x) - \max_{t \in \mathcal{D}} f(t)$$

1. Scale objective values of current members in $[a, b]$:

$$F(x) = a + (b - a)\left(\frac{f(x) - f_-}{f_+ - f_-}\right).$$

What impact does the choice of a and b have?

1. ranking. Sort the candidate solutions by ascending objective values, and assign as fitness their position in the sorted list.

Stage 1: Selection

- Determine which members from the current generation survive to the next.
- Based on the fitness of individuals.

  1. Fitness proportional selection is based on the discrete distribution: given n outcomes, with weights w_i, define:

  $$P(i) = w_i / \left(\sum_j w_j \right).$$

  1. Tournament selection randomly select a pair of members and have them 'face off'. The fittest wins, and survives to the next generation.

Stage 2: Combination

- Assume chromosomes, i.e. bitstring representations. Here, combination is the creation of a pair of new members from a pair of survivors.
- For example: crossover. Given two bitstring, swap one or more substrings between bitstrings. For example, for a single crossover point:

$$\begin{array}{lccccc|ccc} x_1: & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ & & & & & & \updownarrow & \updownarrow & \updownarrow \\ x_2: & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{array}$$

Leading to the offspring:

$$\begin{array}{lccccc|ccc} y_1: & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ y_2: & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \end{array}$$

- Can have multiple crossover points, determined randomly (how?).

Stage 3: Mutation

- Slight changes in chromosomes. Allows a population to get out of a local optimum.
- For example: random flips. Each bit of a chromosome is flipped according to some small probability $p \ll 1$.

$$\begin{array}{lcccccccc} x: & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ \tilde{x}: & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Flip probability usually chosen very low, $p \approx 0.001$ (most bits, or even members, untouched).

Bitstring representations

- For an integer k, there is the standard representation as a bitstring:

$$k = \sum_{i=0}^{m-1} b_i 2^i.$$

Here, m is the number of bits and $b_i \in \{0, 1\}$.

- However, e.g. mutation (and crossover) can have a huge impact on the underlying solution!
- This is undesirable, small changes to the bitstring should correspond to small changes in the original value and vice versa. For example:

31:	0	1	1	1	1	1
32:	1	0	0	0	0	0

→ maximum distance in bitstring, minimum distance in original value.

Gray code

- A Gray code is a bitstring representation of integers, that has the property that if the integer is incremented or decremented, only a single bit changes in the representation.
- Let B_m be the set of all bitstrings of size m. We represent a Gray code as a bijective function $\chi : B_m \to B_m$, and define what it does to individual bits in the standard representation. An example is:

$$\chi(\{b_j\})_i = \begin{cases} b_i & \text{if } i = m-1 \\ 0 & \text{if } i < m-1 \text{ and } b_{i+1} = b_i \\ 1 & \text{otherwise} \end{cases} .$$

The encoded example of the previous slide:

31:	0	1	1	1	1	1
$\chi(31)$:	0	1	0	0	0	0
32:	1	0	0	0	0	0
$\chi(32)$:	1	1	0	0	0	0

Elitism

- In the basic strategies outlined above, the fittest members sometimes die, and if they survive they can get mutated.
- This is not always desirable, since it can decrease overall fitness.
- If elitism is used, the k best members of a generation always survive and are immune to mutation.

Schemata

- Bitstrings are words over the alphabet $\{0,1\}$. This set is written $\{0,1\}^*$.
- Schemata are words in $\{0,1,\bullet\}^*$, where $\bullet$ represents a wildcard.
- A schema H defines a subset $H \subset \{0,1\}^*$, where a chromosome $w \in H$ if it matches all the non-wildcard characters of the schema H (and has the same length).
- We identify schemata and the induced subsets

$$H = \bullet 10 \bullet \quad \leftrightarrow \quad \{0100, 0101, 1100, 1101\} \subset \{0,1\}^*.$$

Properties of schemata

- The order of a schema is the number of non-wildcard letters:

$$o(H) = |\{i \mid H_i \neq \bullet\}|.$$

- The defining length of a schema is the distance between the first and last non-wildcard letter

$$\delta(H) = \max\{j - i \mid j > i,\ H_i \neq \bullet,\ H_j \neq \bullet\}.$$

- For example:

$$H = \bullet 10 \bullet \quad \rightarrow \quad o(H) = 2, \quad \delta(H) = 1.$$

How schemata perform in GAs

- Let us consider the effect that basic selection has on the presence of a schema H.
- Write $\mathcal{G}_t$ for the generation at timestep t. Let $m(H, t)$ be the number of examples of H in generation $\mathcal{G}_t$:

$$m(H, t) = |\mathcal{G}_t \cap H|.$$

- Assume each member survives according to its relative fitness to the total fitness, and we select as many survivors as members (with replacement). Then after selection:

$$E(m(H, t+1)) = m(H, t) \cdot n \cdot \frac{f(H)}{\sum_i f(x_i)} = m(H, t) \frac{f(H)}{\bar{f}},$$

where $n = |\mathcal{G}_t|$, and $f(H)$ is the average fitness of examples of H in generation t, and $\bar{f}$ is the average fitness in $\mathcal{G}_t$.

Exponential growth of schemata

- We have found (with selection only):

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

- Assuming that $f(H) = (1+c)\bar{f}$ with $c > 0$, then observe:

$$\begin{aligned} E(m(H,t+1)) &= m(H,t)\frac{(1+c)\bar{f}}{\bar{f}} \\ &= m(H,t)(1+c) \\ &= m(H,0)(1+c)^t \end{aligned}$$

In other words, schemata with above-average fitness experience exponential growth!

Adding diversity

- Of course, for many fit schemata, we will have $m(H, 0) = 0$ or close to zero, never giving these schemata a chance to grow.
- For this, crossover and mutation are included. We want to analyse the effect this has on 'good' schemas that are part of the curent population.
- For the sake of analysis, we perform single site (chosen uniformly at random) crossover of random pairs, and we crossover in-place, meaning we don't form offspring but modify the surviving chromosomes directly.
- Mutation is defined as before (independent random bitflips with probability p).

The effect of crossover

- Let x_i be a chromosome in $\mathcal{G}_t$ which matches a schema H. Assume it undergoes crossover with another chromosome.
- The resulting string still matches H if the crossover site is outside of the 'defined region' of H.
- Say we have bitstrings of length m, then there are $m-1$ candidate crossover sites. The probability of the crossover site missing H is clearly related to its defining length. The probability p_s of a schema example surviving crossover is at least:

$$p_s \geq 1 - \delta(H)/(m-1).$$

The effect of mutation

- For a schema to survive mutation, it is enough if all specific bits (i.e. non-wildcards) remain untouched.
- The probability of this happening is clearly:

$$p_m = (1 - p)^{o(H)}.$$

- If $p \ll 1$, this is approximately

$$p_m = 1 - p \cdot o(H).$$

Schema Theorem

- Combining what we have found, we see:

$$
\begin{aligned}
E(m(H,t+1)) &= m(H,t)\frac{f(H)}{\bar{f}} p_s p_m, \\
&\geq m(H,t)\frac{f(H)}{\bar{f}} \left(1 - \frac{\delta(H)}{m-1}\right)(1 - p \cdot o(H)), \\
&\approx m(H,t)\frac{f(H)}{\bar{f}} \left(1 - \frac{\delta(H)}{m-1} - p \cdot o(H)\right),
\end{aligned}
$$

- We see that short and small schemata (with low defining length and order) will experience exponential growth. Other schemata get penalized by crossover and mutation operations.
- These operations have to be included to diversify (i.e. allow new schemata to occur) the current population.

Exercises

(Also in `exercises.pdf`)

- Implement elitism to your GA skeleton.
- Implement Gray codes. Either subclass `lcsc::bitstring`, or add a boolean flag to it (`is_gray_encoded`).
- Prove correctness of Algorithm 4.2 (i.e. that χ it is a bijection), and that this indeed gives a Gray encoding. Give also an implementation using bitwise operations.
- Hand-in on ranges (due 15-05).