technische universität dortmund

Computational Intelligence

Winter Term 2010/11

Prof. Dr. Günter Rudolph

Lehrstuhl für Algorithm Engineering (LS 11)

Fakultät für Informatik

TU Dortmund

Important Parameters of EAs (1)

- dimension n of search space
  - no parameter of EA, but given by the problem
  - measures the size of the search space: $\{0,1\}^n$, $\mathbb{R}^n$, S_n
  - plays the same role as input length in classical runtime analysis
  - other parameters are often chosen dependent on n (e. g. mutation probability $p_m = 1/n$)
- population size μ
  - obviously $\mu = n^{O(1)}$
  - often $\mu = \Theta(n)$ or $\mu = \Theta(\sqrt{n})$
  - $\mu = O(1)$ or even $\mu = 1$ are not unusual
- number of offspring λ
  - obviously $\lambda = n^{O(1)}$
  - often $\lambda = 1$
  - $\lambda = \mu$ or $\lambda \gg \mu$ not unusual
  - selection method influences reasonable choice of λ

Important Parameters of EAs (2)

- crossover probability p_c
  - in general $p_c \in [0;1]$ arbitrary
  - often $p_c \in [1/2; 4/5]$ constant
- probability of applying mutation
  - don't confuse with mutation probability!
  - we will always use 1
  - Remark
    $p_m = 1/n \Rightarrow \text{Prob}\,(\text{no mutation}) = (1-1/n)^n \approx 1/e$

Methods for parameter control

- static parameter control
  parameter values constant during the whole run
  - often used
  - \+ simple
  - − maybe it's better to vary the parameter value during the run?!
- dynamic parameter control
  parameter values change during the run according to some time-dependent scheme
  - \+ more flexible than static approach
  - − cannot deal with non-time-dependent changes
  - unusual for EAs
- adaptive parameter control
  parameter values can change dependently on every individual and any random experiment
  - \+ very flexible
  - − hard to analyze
  - − computationally expensive
  - often used for EAs

Self-adaptation

Idea good parameter values evolve together with good individuals

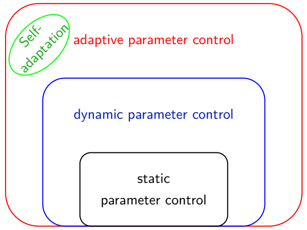
implementation code parameter values together with individual

formally $S \times Q$ instead of S
unchanged $f : S \to R$

e. g. for mutation probability

- every individual has its own mutation probability
- first vary the mutation probability
- then mutate with varied mutation probability
- afterwards normal selection
- important don't swap steps

Hierarchy of parameter control methods



Evolutionary Algorithms: Historical Notes — Lecture 11

Idea emerged independently several times: about late 1950s / early 1960s.

Three branches / "schools" still active today.

- **Evolutionary Programming (EP)**:
  Pioneers: Lawrence Fogel, Alvin Owen, Michael Walsh (New York, USA).

  Original goal: Generate intelligent behavior through simulated evolution.
  Approach: Evolution of finite state machines predicting symbols.
  Later (~1990s) specialized to optimization in $\mathbb{R}^n$ by David B. Fogel.

- **Genetic Algorithms (GA)**:
  Pioneer: John Holland (Ann Arbor, MI, USA).

  Original goal: Analysis of adaptive behavior.
  Approach: Viewing evolution as adaptation. Simulated evolution of bit strings.
  Applied to optimization tasks by PhD students (Kenneth de Jong, 1975; et al.).

- **Evolution Strategies (ES)**:
  Pioneers: Ingo Rechenberg, Hans-Paul Schwefel, Peter Bienert (Berlin, Germany).

  Original goal: Optimization of complex systems.
  Approach: Viewing variation/selection as improvement strategy. First in $\mathbb{Z}^n$, then $\mathbb{R}^n$.

Evolutionary Algorithms: Historical Notes — Lecture 11

"Offspring" from GA branch:

- **Genetic Programming (GP)**:
  Pioneers: Nichael Lynn Cramer 1985, then: John Koza (Stanford, USA).

  Original goal: Evolve programs (parse trees) that must accomplish certain task.
  Approach: GA mechanism transfered to parse trees.
  Later: Programs as successive statements → Linear GP (e.g. Wolfgang Banzhaf)

Already beginning early 1990s:

Borders between EP, GA, ES, GP begin to blurr ...

⇒ common term **Evolutionary Algorithm** embracing all kind of approaches

⇒ broadly accepted name for the field: **Evolutionary Computation**

scientific journals: *Evolutionary Computation* (MIT Press) since 1993,
IEEE Transactions on Evolutionary Computation since 1997,
several more specialized journals started since then.

Design of EAs

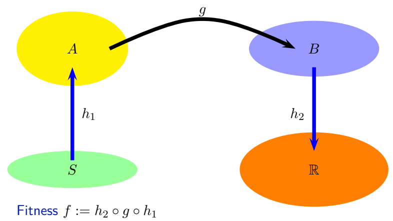
Idea Methodology to apply standard EAs

Goal standard EAs do not have to be changed

Requirement problem is given as $g\colon A \to B$
g has to be maximized (or minimized)
A arbitrary set, B partially ordered

EA operates on search space S
'maximizes' fitness $f : S \to \mathbb{R}$

Definition of mappings



Fitness $f := h_2 \circ g \circ h_1$

h_1 is genotype-phenotype-mapping.

Genotype-Phenotype-Mapping $\mathbb{B}^n \to [L, R] \subset \mathbb{R}$

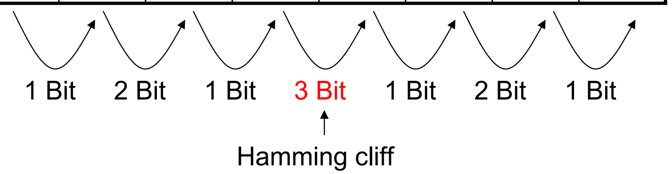
- Standard encoding for $b \in \mathbb{B}^n$

$$x = L + \frac{R-L}{2^n - 1} \sum_{i=0}^{n-1} b_{n-i}\, 2^i$$

→ Problem: *hamming cliffs*

000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7

L = 0, R = 7
n = 3



Genotype-Phenotype-Mapping $\mathbb{B}^n \to [L, R] \subset \mathbb{R}$

- Gray encoding for $b \in \mathbb{B}^n$

Let $a \in \mathbb{B}^n$ standard encoded. Then $b_i = \begin{cases} a_i, & \text{if } i = 1 \\ a_{i-1} \oplus a_i, & \text{if } i > 1 \end{cases}$

$\oplus$ = XOR

000	001	011	010	110	111	101	100	← genotype
0	1	2	3	4	5	6	7	← phenotype

OK, no hamming cliffs any longer …

⇒ small changes in phenotype „lead to“ small changes in genotype

since we consider evolution in terms of Darwin (not Lamarck):

⇒ small changes in genotype lead to small changes in phenotype!

but: 1-Bit-change: **000** → **100** ⇒ ☹

Genotype-Phenotype-Mapping $\mathbb{B}^n \rightarrow \mathbb{P}^n$ (example only)

- e.g. standard encoding for $b \in \mathbb{B}^n$

individual:

010	101	111	000	110	001	101	100	← genotype
0	1	2	3	4	5	6	7	← index

consider index and associated genotype entry as unit / record / struct;

sort units with respect to genotype value, old indices yield permutation:

000	001	010	100	101	101	110	111	← genotype
3	5	0	7	1	6	4	2	← old index

= permutation

Requirements on h_1 and h_2

obvious requirements

- h_1 and h_2 can be computed efficiently
- h_2 suits g, i. e. good points in B are mapped to good points in $\mathbb{R}$
- h_1 maps on many (all) important points of A
- Optima of f correspond to optima of g

Caution requirements can be hard to achieve in practice

for non-obvious requirements a metric is important

Definition

Mapping $d\colon M \times M \rightarrow \mathbb{R}_0^+$ is a metric on the set M :⇔

1. $\forall x, y \in M: x \neq y \Leftrightarrow d(x,y) > 0$ (positivity)
2. $\forall x, y \in M: d(x,y) = d(y,x)$ (symmetry)
3. $\forall x, y, z \in M: d(x,y) + d(y,z) \geq d(x,z)$ (triangle inequality)

Metric-based EAs

Assumption Metric d_A on A known
(d_A reflects application knowledge)

Requirement metric d_S is known

if h_1 injective, $d_S(x, x') := d_A(h_1(x), h_1(x'))$ is metric

Requirement monotonicity

$$\forall x, x', x'' \in S: \quad d_S(x,x') \leq d_S(x,x'') \Rightarrow d_A(h_1(x), h_1(x')) \leq d_A(h_1(x), h_1(x''))$$

Variation as randomized mapping

now Design-rules for variation operators

hence Formalize variation operators as randomized mappings

$r\colon X \rightarrow Y$ randomized mapping
$\Leftrightarrow r(x) \in Y$ depends on $x \in X$ and random experiment

formally probability space (Ω, p)

$$r\colon X \times \Omega \rightarrow Y$$

$$\text{Prob}\,(r(x) = y) = \sum_{\omega \in \Omega\colon r(x,\omega)=y} p(\omega)$$

Example 1-bit mutation
$\Omega := \{1, 2, \ldots, n\}$, $\forall i \in \Omega\colon p(i) = 1/n$

1-bit mutation is randomized mapping $m\colon \{0,1\}^n \rightarrow \{0,1\}^n$
where $m(x,i) := x \oplus 0^{i-1}10^{n-i}$

Design-rules for mutation

favor small changes

$$\forall x, x', x'' \in S: \quad d_S(x,x') < d_S(x,x'') \Rightarrow \text{Prob}\,(m(x) = x') > \text{Prob}\,(m(x) = x'')$$

no bias

$$\forall x, x', x'' \in S: \quad d_S(x,x') = d_S(x,x'') \Rightarrow \text{Prob}\,(m(x) = x') = \text{Prob}\,(m(x) = x'')$$

Design-rules for crossover

offspring similar to parents

$$\forall x, x', x'' \in S: \quad \text{Prob}\,(c(x,x') = x'') > 0 \Rightarrow \max\{d_S(x,x''), d_S(x',x'')\} \leq d_S(x,x')$$

no bias

$$\forall x, x' \in S: \forall \alpha \in \mathbb{R}_0^+: \text{Prob}\,(d_S(x, c(x,x')) = \alpha) = \text{Prob}\,(d_S(x', c(x,x')) = \alpha)$$

Any EA that fulfills these four design-rules is called a metric-based EA (MBEA).