Computational Complexity and Evolutionary Computation

Ingo Wegener, Univ. Dortmund, Germany

More precisely:

How to apply methods from

classical algorithm analysis

to evolutionary computation.

But why?

These methods lead to

- theorems without any assumptions,
- theorems on the algorithm and not on a model of the algorithm,
- theorems for arbitrary problem dimension.

1. Introduction (survey later)

We discuss search heuristics
(= randomized algorithms)

including EA, ES, GA, GP, Sim. Ann., tabu search

for some kind of optimization on discrete search spaces.

Different types of problems:

one-shot scenario:	one function $\longrightarrow$ no theory
problem-specific scenario:	TSP, scheduling, ...
structural scenario:	pseudo-boolean polynomials degree $\leq d$, $\leq N$ terms, positive weights, ...

The scenario

Problem: Class of functions
– all linear functions $f\colon \{0,1\}^n \to \mathbb{R}$
– all TSP-functions
$f_D(\pi) =$ cost of tour π
w.r.t. distance matrix D.

Instance: one specific of these functions.

Important

↗ instance is known (cost matrix for TSP) and can be used by the algorithm

↘ instance is not known (black-box optimization)

Needle in the haystack

↗ trivial problem

↘ difficult problem

Given a problem and a randomized algorithm – what do we want to know?

The probability distribution of the "state"

of the algorithm depending on t and the instance

$\longrightarrow$ impossible in non-trivial situations.

→ Expected time until good event
(optimum found) happens
variance, moments, ...
success probabilities

→ only good estimates are possible.

DON'T TRY TO BE TOO EXACT!
YOU WILL FAIL.

Typical EA-theory approaches:

→ reasonable model, calculation in the model, experiments to "verify" the model

→ no result for large problem dimension n,

→ infinite populations

→ how to control the error?

→ studying the dynamics of the stochastic process

→ what is the meaning of the results?

→ studying the one-step behavior
(schema theory, quality gain, progress rate, ...)

→ what happens in many steps?

→ building block hypothesis

→ just a nice hypothesis (royal roads),

→ convergence results

→ I do not have enough time!

DON'T TRY TO BE TOO GENERAL!
RESULTS ARE NECESSARILY BAD.

Analysis of expected optimization time and success probability:

- no assumptions,
- results about the algorithm,
- only (good) estimates,
- error can be controlled,
  (upper and lower bounds).

$\longrightarrow$ Theorems (!), mathematically proven, for all problem dimensions n and instances

$\longrightarrow$ useful in 10 or 100 years,

$\longrightarrow$ no verification by experiments,

$\longrightarrow$ experiments are useful: what happens between the lower and the upper bound?

2. Survey

General tools

3. Markoff and Chebycheff Inequality
4. Chernoff Inequality
5. Multistart Techniques
6. Coupon Collector’s Theorem
7. Gambler’s Ruin Problem
8. Random Walks and Resistive Electrical Networks
9. Family Trees

Tools for the analysis of randomized search heuristics

10. Artificial Fitness Layers
11. Typical Runs
12. Potential Functions
13. Drift Analysis

Limitations to all heuristics

14. Yao’s Minimax Principle

15. Conclusions

3. Markoff and Chebycheff Inequality

Markoff Inequality

$X \geq 0$ and $t > 0 : \text{Prob}(X \geq t) \leq E(X)/t$.

Proof:

$$Y(\omega) := \begin{cases} t & \text{if } X(\omega) \geq t \\ 0 & \text{otherwise} \end{cases}$$

$\Rightarrow$ $X \geq Y$

$\Rightarrow$ $E(X) \geq E(Y) = 0 \cdot \text{Prob}(X < t) + t \cdot \text{Prob}(X \geq t)$

$\Rightarrow$ claim.

Often a crude estimate, but depends only on expectation.

Small expected optimization time $\Rightarrow$

good success probability in short time.

T : optimization time

$\text{Prob}(T \geq c \cdot E(T)) \leq \frac{E(T)}{c \cdot E(T)} = \frac{1}{c}$

and $\text{Prob}(T < c \cdot E(T)) \geq 1 - \frac{1}{c}$,

e. g. $c = 2,\ c = 10,\ c = n^{1/2},\ c = n, \ldots$

Example: n independent trials with success probability $1/2$.

$$X_i = \begin{cases} 1 & \text{success in trial } i \\ 0 & \text{otherwise} \end{cases}$$

$X := X_1 + \cdots + X_n =$ number of successes.

How small is $\text{Prob}(X \geq (1 + \varepsilon) \cdot E(X))$?

$E(X_i) = 1/2, E(X) = n/2.$

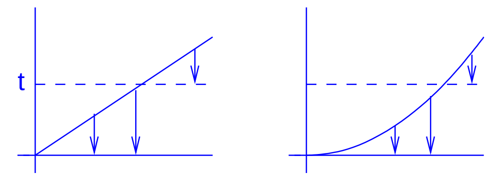
Markoff Inequality: $\text{Prob}(X \geq (1 + \varepsilon) \cdot E(X)) \leq \frac{1}{1+\varepsilon}$.

Chebysheff Inequality

$t > 0 : \text{Prob}(|X - E(X)| \geq t) \leq \frac{V(X)}{t^2}$.

Proof: $Y := |X - E(X)|$.
$Y^2 = (X - E(X))^2 \Rightarrow Y^2 \geq 0$ and $E(Y^2) = V(X)$.
Apply Markoff Inequality to Y^2 and t^2.

Markoff   Chebysheff



Sometimes better – sometimes worse.
It is necessary to know the variance.

Again example of n independent trials.
$V(X_i) = 1/4$ (simple calculation)
$V(X) = n/4$

$$\text{Prob}(X \geq (1+\varepsilon)\cdot E(X)) = \text{Prob}(X - E(X) \geq \varepsilon \cdot E(X))$$

$$\leq \text{Prob}(|X - E(X)| \geq \varepsilon \cdot E(X)) \overset{\text{Cheb.}}{\leq} \frac{V(X)}{\varepsilon^2 \cdot E(X)^2} = \frac{n/4}{\varepsilon^2 \cdot (n/2)^2} = \frac{1}{\varepsilon^2 \cdot n}$$

$\rightarrow$ better than Markoff if $\varepsilon \gg \frac{1}{n^{1/2}}$.

4. Chernoff Bounds

$X_1, ..., X_n$ independent 0 - 1 random variables,
$X = X_1 + \cdots + X_n$ (number of successes),
$\text{Prob}(X_i = 1) = p_i$ for some $0 < p_i < 1$

$\Rightarrow$

$E(X) = p_1 + \cdots + p_n$
$0 < \varepsilon < 1 : \text{Prob}(X \leq (1-\varepsilon)\cdot E(X)) \leq e^{-E(X)\cdot\varepsilon^2/2}$

Proof idea:

Markoff inequality for e^{-tX} and $t := -\ln(1-\varepsilon)$.

(The e-function is “more convex” than quadratic functions.)

In our example all $p_i = 1/2 \rightarrow$ the situation is symmetric.

$$\text{Prob}(X \geq (1+\varepsilon)\cdot E(X)) = \text{Prob}(X \leq (1-\varepsilon)\cdot E(X)) \leq e^{-n\cdot\varepsilon^2/2}.$$

Much better than Markoff and Chebysheff in this situation.

More precisely:

$\text{Prob}(X \geq 0.6 \cdot n) = \text{Prob}(X \geq 1.2 \cdot E(X)) \leq e^{-0.02\cdot n}$.

$\text{Prob}(X \geq n/2 + n^{3/4}) = \text{Prob}(X \geq (1 + 2n^{-1/4})\cdot E(X)) \leq e^{-2\cdot n^{1/2}}$.

$\text{Prob}(X \geq n/2 + n^{1/2}) = \text{Prob}(X \geq (1 + 2n^{-1/2})\cdot E(X)) \leq e^{-2}$.

Moreover:

These bounds are quite close to optimal.

Applications

Probability of fitness increasing step $\frac{1}{n}$

$\rightarrow$ almost surely $\Theta(n^2)$ steps to increase fitness n times

$\longrightarrow$

DO NOT INVESTIGATE SINGLE STEPS –
INVESTIGATE PHASES OF MODERATE LENGTH.

We can estimate the prob. of bad events.

Mutation prob. $1/n$, phase length n^2.

$\mathrm{Prob}(x_i$ has flipped less than $0.9n$ times
or more than $1.1n$ times) = expo. small

$$\mathrm{Prob}(\exists x_i : x_i \ldots) \overset{\text{union bound}}{\leq} n \cdot \text{ expo. small} = \text{expo. small}$$

Conclusion:

In single steps, events may happen or not.

In a phase of moderate length where all

steps have “similar properties”, it is

very likely that the number of steps where

some event happens is close to its expected value.

5. Multistart Techniques

T – random variable counting the number of steps until optimal search point is found.

$E(T)$ – expected optimization time
(should be a polynomial of small degree with a small constant factor).

$s(n) := \mathrm{Prob}(T \leq t(n))$ – success probability.

Which success probabilities are good enough?

Case 1: $s(n)$ is a lower bound on the success probability for each initial population.

Time $\quad t(n) \cdot \alpha(n)/s(n) \triangleq \alpha(n)/s(n)$ phases of length $t(n)$

$$\mathrm{Prob}(T(n) \geq t(n) \cdot \alpha(n)/s(n)) \leq (1 - s(n))^{\alpha(n)/s(n)}$$
$$= \underbrace{\left[(1 - \tfrac{1}{1/s(n)})^{1/s(n)}\right]}_{\approx e^{-1}}^{\alpha(n)} \approx e^{-\alpha(n)}.$$

E. g. $\alpha(n) = \ln n$, $s(n) = 1/n$, $t(n) = n^2 \rightarrow$
success probability in $n^3 \cdot \ln n$ steps at least $1 - 1/n$.

Case 2: $s(n)$ is only a lower bound on the success probability for the random initial population.

$\rightarrow$ Multistarts (in parallel)
but
how many and how long – if $s(n)$ and $t(n)$ are not known?

For $i = 0, 1, \ldots$:
perform 2^i runs of the EA for 2^i steps.

Length of phase i : 4^i.

Probability that phases $0, \ldots, i$ are without success:

$\leq 1 \quad$ if $i \leq \log t(n)$

$\leq (1 - s(n))^{2^i} \quad$ if $i > \log t(n)$

If $t(n)$ and $1/s(n)$ polynomially bounded:

- polynomial time for first $\log t(n)$ phases
- length of phases increases exponentially
- probability of no success decreases double exponentially

$\rightarrow$ polynomial expected optimization time.

6. Coupon Collector's Theorem

Randomized local search:

- Choose a bit position $i \in \{1, ..., n\}$ uniformly at random.
- Flip the ith bit.
- Some selection procedure.

How many steps do we need until each bit has flipped at least once?

If j bits have been flipped already,
probability of $(n-j)/n$ to flip a bit at a “new” position
$\rightarrow$ expected waiting time $\frac{n}{n-j}$.

$$E(T) = n \cdot \left(\tfrac{1}{n} + \tfrac{1}{n-1} + \cdots + \tfrac{1}{2} + 1\right) \approx n \cdot \ln n.$$

Even better:

$$\lim_{n\to\infty} \mathrm{Prob}\,(T \le n \cdot \ln n - c \cdot n) = e^{-e^{c}}$$

$$\lim_{n\to\infty} \mathrm{Prob}\,(T \ge n \cdot \ln n + c \cdot n) = 1 - e^{e^{-c}}.$$

Sharp threshold result: T almost sureley close to $E(T)$.

Application

(1 + 1) EA for a function with a unique optimum.

Chernoff: Almost surely at least $0.4n$ wrong bits after initialization.

RLS: $E(T) \ge n \cdot \left(\frac{1}{0.4n} + \cdots + 1\right) \approx n \cdot (\ln n + \ln 0.4)$.

(1 + 1) EA: Chernoff: Number of flipping bits almost surely less than $(1+\varepsilon)\cdot$ number of steps
$\rightarrow$ analysis of RLS.

If at least 2 bits flip in one step, they are different
($\rightarrow$ not independent) $\rightarrow$ minor effect that can be controlled.

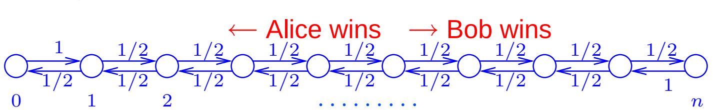
Conclusion: There is a general $\Omega(n \cdot \log n)$ bound
for mutation-based EAs for functions with a unique optimum
like ONEMAX.

Further applications:

- one-point crossover at many positions,
- collecting pictures of football players if you get two of them for buying a bar of chocolat.

7. Gambler’s Ruin Problem

Part 1: Fair games



Starting at n: How long does it take to reach state 0?

Chernoff: Number of Alice’s wins $\approx$ number of Bob’s wins.

But: N^2 steps, deviation of $\Theta(N)$ may be possible.
Expected time $\Theta(n^2)$?!

Difference equations:

$E(i) :=$ expected time if starting at i.

$E(0) = 0$.

$E(n) = 1 + E(n-1)$.

$E(i) = 1 + \frac{1}{2} \cdot E(i-1) + \frac{1}{2} \cdot E(i+1)$ if $0 < i < n$.

Solve for $E(1), E(2), \dots, E(10)$, guess solution,
verify by induction $\rightarrow \Theta(n^2)$.

Applications

(1+1) EA on a path of length l
where the final point is optimal
and all other have the same fitness $\rightarrow \Theta(nl^2)$

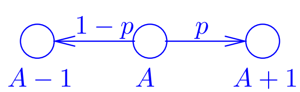
Ising model on the ring $\rightarrow O(n^3)$

Maximum matching on a path $\rightarrow O(n^4)$

Part 2: Unfair games

Alice owns A dollars, Bob owns B dollars, $n = A + B$.

Alice's probability of winning one round: $p \neq 1/2$.



How large is the probability that Alice ruins Bob before she is ruined?

Say: $A = 90, B = 10, p = 1/3$.

$P(A) =$ Alice's winning probability if she has A dollars
(and Bob has $n - A$ dollars)
$P(0) = 0$.
$P(n) = 1$.
$P(A) = p \cdot P(A+1) + (1-p) \cdot P(A-1)$ if $0 < A < n$.

Solution: $t := \frac{1-p}{p}$

$$P(A) = \frac{1-t^A}{1-t^{A+B}} = \frac{1-t^A}{1-t^n} = \frac{t^A-1}{t^n-1}$$

Example: $p = 1/3 \Rightarrow t = 2$

$$P(A) = \frac{2^{90}-1}{2^{100}-1} \approx 2^{-10}.$$

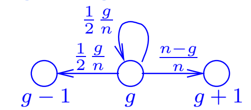
Application

Separable problem, n parts,
one part is chosen randomly.
If a good part is chosen, it
turns bad with probability $1/2$.
If a bad part is chosen, it
turns good with probability 1.

We start with $n/2$ good and $n/2$ bad parts.

Expected time until all parts are good?

$g :=$ number of good pairs



There is a point of time where $g = \frac{7}{8}n$.
Then we have to reach $g = n$ before $g = \frac{6}{8}n$.

During that period $\xleftarrow{\geq \frac{3}{8}} \bigcirc \xrightarrow{\leq \frac{2}{8}}$.

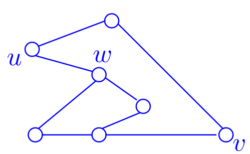
Don't count loops $\xleftarrow{\geq \frac{3}{5}} \bigcirc \xrightarrow{\leq \frac{2}{5}}$.

Calculate with the best values $\xleftarrow{0.6} \bigcirc \xrightarrow{0.4}$.

Prob of reaching $g = n$ before $g = \frac{6}{8}n$ exponentially small
$\rightarrow$ waiting time exponential with overwhelming probability.

8. Random Walks and Resistive Electrical Networks

Random walks on graphs



Sitting at w choose one of the (three) adjacent edges uniformly at random.

$H_{u,v} :=$ expected time to reach v starting at u (hitting time),

$C_{u,v} := H_{u,v} + H_{v,u}$ (commute time).

Resistive Electrical Networks

Edges $\rightarrow$ branch resistance 1 Ohm.

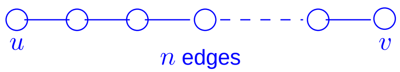
Ohm's Law: $U = R \cdot I$ (voltage=resistance $\cdot$ current).

Kirchhoff's Law : $\forall$ node v: entering current = leaving current.

Effective resistance between u and v:
$R_{u,v}$ absolute value of voltage difference between u and v
if one ampere is injected into u and removed from v.

Theorem: $C_{u,v} = 2 \cdot |E| \cdot R_{u,v}$.

Example



$R_{u,v} = n \Rightarrow C_{u,v} = 2n^2 \Rightarrow$ (by symmetry) $H_{u,v} = n^2$.

It may happen that $H_{u,v} \neq H_{v,u}$:

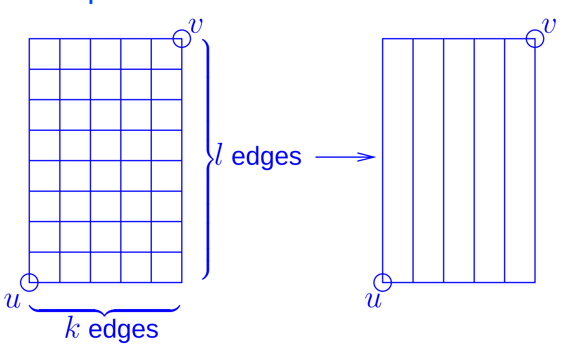
Theorem: $H_{u,v} = |E| \cdot R_{u,v} + \frac{1}{2} \sum_{w \in V} \overset{\text{degree}}{\overset{\downarrow}{d(w)}} (R_{u,w} - R_{v,w})$.

It may be difficult to calculate all effective resistances $\rightarrow$ simplifications of the network lead to upper and lower bounds.

Rayleigh's Short-cut Principle
Effective resistance is never raised by lowering the resistance on an edge or by merging two nodes.

Effective resistance is never lowered by raising the resistance on an edge, by eliminating an edge, or by cutting a node into two nodes.

Example



$k+1$ parallel paths from u to v each of length $l+k$.
Effective resistance per path $l+k$.
$R_{u,v} \leq \frac{l+k}{k+1} \leq \frac{l}{k} + 1$.

Number of edges $= (l+1) \cdot k + l \cdot (k+1) = 2kl + k + l$.

$C_{u,v} \leq 2 \cdot (2kl + k + l) \cdot (\frac{l}{k} + 1) = 4l^2 + O(kl)$.

Remark:

$u, v \in \{0,1\}^n$ – hypercube

$\rightarrow R_{u,v} \in [1/n, n]$.

These are results for RLS.

What about EA-mutation?

→ complete graph (all transition probabilities positive)
different transition probabilities for different Hamming distances.

→ similar theory with weighted graphs
resistance per edge $= 1/$transition probability.

9. Family Trees

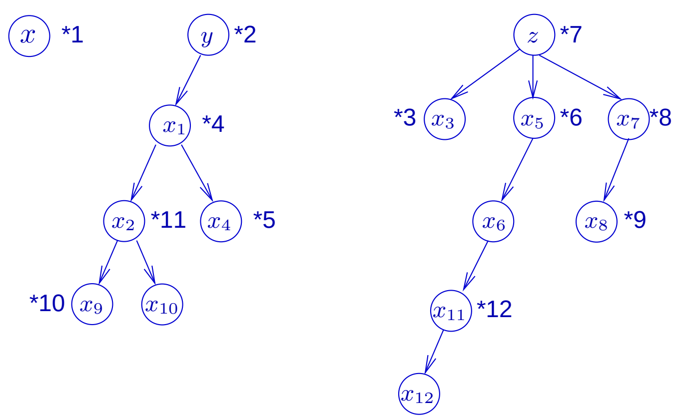
Lower bound techniques for population-based EAs?
Consider a quite general $(\mu + 1)$ EA for maximization.

- Random initialization.
- Selection for reproduction:
  uniform.
- Mutation of selected individual.
- Selection for replacement:
  choose a best individual and keep it
  probability for a worst individual at least $1/\mu$.

How to illustrate the current population
and its history:

let $\mu = 3$, x, y, z individuals of initial population,
x_t new individual in generation t,

directed edge $a \rightarrow b$: b was created as child of a,
$*t'$: eliminated in step t',
without $*$: current generation.



Current generation: x_6, x_{10}, x_{12}.

Mutation is a quite local operator.

Typically members of initial population far from optimum (w.r.t. Hamming distance).

$\rightarrow$ The path from the root to the first optimal individual cannot be too short.

How fast are these trees growing w.r.t. depth?

Stochastic model of randomized recursive trees

T_0 contains only its root.

$T_{t-1} \rightarrow T_t$: choose a random node v of T_{t-1} and create a new child of v.

In our situation: Some nodes are “dead” $\rightarrow$

$1/\mu$-randomized trees:

$\forall$ node v: probability of choosing v is at most $1/\mu$.

$X(t,d)$: random number of nodes on level d.
$E(t,d)$: expected number of nodes on level d.

$E(t,0) = 1$.
$E(0,d) = 0$ for $d \geq 1$,

$$E(t,d) \leq \sum_{0 \leq i \leq t} \frac{E(i,d-1)}{\mu} \overset{\text{induction}}{\leq} \frac{1}{d!} \left(\frac{t}{\mu}\right)^d .$$

$D(t)$: random variable describing the depth.

$$\text{Prob}(D(t) \geq d) = \text{Prob}(X(t,d) \geq 1) \overset{\text{Markoff}}{\leq} E(t,d) \leq \frac{1}{d!} \left(\frac{t}{\mu}\right)^d .$$

$\rightarrow$ Prob$(D(t) \geq 3t/\mu) = 2^{-\Omega(t/\mu)}$

$\rightarrow$ rather small depth for large μ.

$\rightarrow$ If optimum is unique,

expected run time $\Omega(\mu n + n \log n)$.

(ONEMAX, LEADING ONES, ...)

even $\Omega(\mu n + n^2)$

$\rightarrow$ To be fast, you have to prefer the best individuals $\rightarrow$ choose (1+1) EA.

After these general tools:

tools for the analysis of

randomized search heuristics.

10. Artifi cial Fitness Layers

Let $f : \{0,1\}^n \to \mathbb{R}$, $\quad A, B \subseteq \{0,1\}^n$.
$A <_f B :\Leftrightarrow\ \forall a \in A, b \in B : f(a) < f(b)$.

$A_1, \ldots, A_m \subseteq \{0,1\}^m$ $<_f$-partition $:\Leftrightarrow$
$\bigcup A_i = \{0,1\}^m,\ A_i \cap A_j \neq \emptyset,$
$A_1 <_f A_2 <_f \cdots <_f A_m,$
A_m contains only global optima.

$A_1, \ldots, A_m$ $<_f$-partition,
$p(A_i) := \mathrm{Prob}($initial search point in$A_i)$,
$s(a) := \mathrm{Prob}($mut$(a) \in A_{i+1} \cup \cdots \cup A_m)$ if $a \in A_i$,
$s_i := \min\{s(a) | a \in A_i\} \quad \longrightarrow$

expected opt. time of (1+1) EA
$\leq \sum_{1 \leq i \leq m-1} p(A_i) \left(s_i^{-1} + \cdots + s_{m-1}^{-1}\right) \leq s_1^{-1} + \cdots + s_{m-1}^{-1}$.

Applications

ONEMAX

$A_i = \{a | a_1 + \cdots + a_n = i\},\ 0 \leq i \leq n,$
$s(a) \geq \frac{n-i}{en}$ if $a \in A_i \quad \longrightarrow \quad en\left(1 + \cdots + \frac{1}{n}\right) \leq en\,(\ln n + 1)$.

LEADING ONES

$A_i = \{a | a$ has i leading ones$\},\ 0 \leq i \leq n,$
$s(a) \geq \frac{1}{en} \quad \longrightarrow \quad en^2.$

Unimodal functions

Each non-optimal a has a better Hamming neighbor.

$f : \{0,1\}^n \to \{0,1,\dots,b\}$

$A_i = \{a | f(a) = i\},\ 0 \le i \le b,$

$s(a) \ge \frac{1}{en} \quad \longrightarrow \quad enb.$

Binary Value

$$BV(a) = \sum_{1 \le i \le n} a_i \cdot 2^{n-i} \qquad en(2^n + 1).$$

Linear functions

$f^*(a) = v_1 a_1 + \cdots + v_n a_n$

$\downarrow$

equivalent to

$f(a) = w_1 a_1 + \cdots + w_n a_n$ where $w_1 \ge w_2 \ge \cdots \ge w_n > 0$

$A_i = \{a | w_1 + \cdots + w_i \le f(a) < w_1 + \cdots + w_{i+1}\},\quad 0 \le i \le n.$

$\longrightarrow \exists a, a' \in A_i : f(a) \ne f(a')$

$s(a) \ge \frac{1}{en} \quad \longrightarrow \quad en^2$ (also for BV).

Lower bounds with artificial fitness layers

$M_i := \max\{s(a) | a \in A_i\}$

$\to$ expected opt. time of (1+1) EA

$$\ge \sum_{1 \le i \le m-1} p(A_i) \cdot \mu_i^{-1}.$$

$\to$ bounds often weak since we typically have to leave many fitness layers.

Assume that we have to cross $\ge \ell$ layers, assume that expected number of layers that we cross per step is bounded above by $\alpha \in \mathbb{R}$.

$\to$ expected optimization time $\ge \ell/\alpha$?

Not so easy, variance, ...

$\to$ drift analysis in Section 13

11. Typical Runs

The local behavior (within a short phase) of a randomized heuristic may have a large variance

but

the global behavior (within a phase of sufficient length) can be a quite stable (see Chernoff bounds, coupon collector).

$\rightarrow$
Describe the typical behavior within certain phases (chosen by us) and estimate the failure probability (behavior not typical).

$\rightarrow$ typical behavior with overwhelming probability

$\rightarrow$ leads often also to results on expected optimization time.

Application 1 - a lower bound

$$\text{TRAP}(a) = \begin{cases} \text{ONEMAX}(a) & \text{if } a \neq 0^n \\ 2n & \text{if } a = 0^n \end{cases}$$

$\rightarrow$ exp. opt. time of (1+1) EA $\geq (1 - o(1)) \cdot n^n$.

Phase 1: initialization
failure a has less than $n/3$ ones
$\rightarrow$ small failure probability by Chernoff bounds.

Phase 2: length $cn^2 \log n$
failure $a \neq 1^n$
failure 1 $a = 0^n$.
$\rightarrow$ mutation flips $\geq n/3$ steps
$\rightarrow$ expo. small probability (also in $cn^2 \log n$ steps)
or
failure 2 (1+1) EA does not optimize ONEMAX
$\rightarrow$ prob $1/2$ for $cn \log n$ steps (Markoff inequality)
$\rightarrow$ prob $(1/2)^n$ for n of these subphases.

Phase 3: If $a = 1^n$, we have to flip all n bits in one step
$\rightarrow$ exp. time $= n^n$.

Application 2 - an upper bound

Steady state GA with population size n, uniform crossover, and crossover probability small enough $(1/n \log n)$, $m = \log n$.

$$\text{JUMP}(a) = \begin{cases} \text{ONEMAX}(a) & \text{if } a = 1^n \text{ or } a_1 + \cdots + a_n \leq n - m \\ 0 & \text{otherwise.} \end{cases}$$

Idea: Uniform crossover can find the optimum if applied to a and b both with m zeros but at different places.

Expected optimization time $O(n^3 \log n)$.

Phase1: $cn^3 \log n$ steps
failure: optimum not found and population contains a string with less than $n - m$ ones.
$\rightarrow$ small failure probability applying results on ONEMAX.

Phase 2: $cn^3 \log n$ steps
failure: optimum not found and there is a bit position i such that more than $n/(4m)$ individuals share a 0 at position i.
$\rightarrow$ difficult estimation of failure probability

All following phases
$\rightarrow$ good chance that crossover does the job
and, otherwise,
small failure probability of the failure event of Phase 2.

12. Potential Functions

The selection steps of the EA are based on the fitness – may be difficult to analyse – in particular, if we analyse classes of functions, e.g., all linear functions

$$w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Idea from classical algorithm analysis:

– find artificial “fitness” (called potential) to measure the progress of the search according to the potential function (the EA uses still the real fitness)

Difficult: the right intuition to define a suitable potential function

Application 1

$BV(a) = a_n + 2a_{n-1} + 4a_{n-2} + \cdots + 2^{n-1}a_1$

(1+1) EA with mutation probability $\frac{1}{2n}$

potential: number of ones

1	1	1	1	0		1		0	

accepted step: leftmost flipping bit is a 0-bit ($\leftarrow$ don’t consider more flipping 0-bits) expected number of flipping ones $\leq 1/2$

$\rightarrow$ expected gain of potential: $\geq 1/2$.

$\rightarrow$ expected number of accepted steps to increase potential from l to at least $l+1$: $\frac{1}{\text{expected gain}} \leq 2.$

$\rightarrow$ expected number of accepted steps to reach optimum $\leq 2n.$

$\rightarrow$ Calculate the number of non-accepted steps $\rightarrow O(n \log n)$

Mutation probability $1/n$?

Phase 1: consider only left half of string
Phase 2: consider only steps not flipping bit in first half

$\rightarrow O(n \log n)$

Application 2

Arbitrary linear function $w_1a_1 + \cdots + w_na_n$
(w.l.o.g. $w_i \geq 0$ and $w_1 \geq \cdots \geq w_n$)

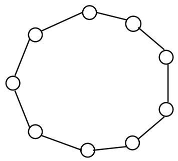
Potential: $2(a_1 + \cdots + a_{n/2}) + a_{n/2+1} + \cdots + a_n$

$\rightarrow$ expected gain of an accepted step $\geq c > 0$

$\longrightarrow$ analysis like BV.

Further applications

– Ising model on the ring



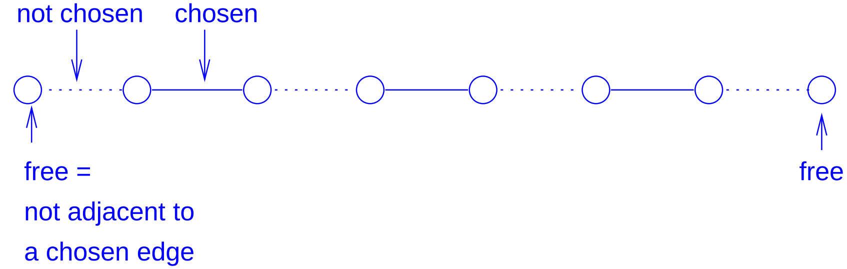
color the nodes with two colors

maximize number of monochromatic edges

potential: length of shortest monochromatic subblock

– Maximum matchings in undirected graphs
find a maximal-size set of edges which do not share a node

Augmenting path



Potential: length of shortest augmenting path.

13. Drift Analysis

We have seen:

not too small positive expected gain and never large gain.
$\rightarrow$ small expected time for a large gain.

$$X_i = \begin{cases} 2^{2n} & \text{probability } 2^{-n} \\ -1 & \text{otherwise} \end{cases}$$

$E(X_i) \geq 2^n - 1$ but expected time for a positive gain: 2^n.

Can we prove

– negative expected gain

$\rightarrow$ small probability for a positive gain?

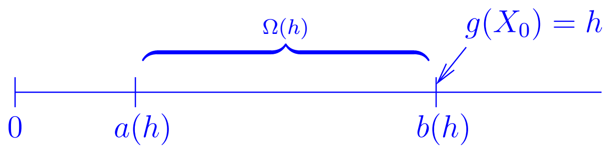
No,

$$X_i = \begin{cases} -2^{2n} & \text{probability } 2^{-n} \\ +1 & \text{otherwise.} \end{cases}$$

However, something in this direction should be true.

Drift theorem (Hajek (1982))

$X_0, X_1, X_2, \ldots$ Markov process,
e.g., random values of a potential,
$g : \mathbb{R} \to \mathbb{R}_0^+$, $0 \le a(h) \le b(h)$, $\lambda > 0$, r a polynomial.



$$E\left(e^{-\lambda(g(X_{t+1})-g(X_t))} \,\middle|\, X_t, a(h) < g(X_t)\right) \le 1 - 1/r(h)$$

$(e^0 = 1 \Rightarrow -\lambda(g(X_{t+1}) - g(X_t))$ has a negative tendency
$\Rightarrow g(X_{t+1})$ tends to be larger than $g(X_t))$.

Let $T := \min\{t | g(X_t) \le a(h)\}$.

$\Rightarrow$ Prob($T \le B$) $\le B \cdot \underbrace{e^{\lambda(a(h)-b(h))}}_{\text{expo. small}} \cdot r(h)$.

Applications are not so easy.

– (1+1) EA is bad on some graphs

and the maximum matching problem.

Lower bounds for all black-box heuristics

14. Yao's Minimax Principle

due to Andy Yao (Turing Award Winner, 2001)

Algorithm design for a problem (with many possible instances) is a game between the algorithm designer and his adversary choosing the instance.

Consider first only deterministic algorithms and remember that a randomized algorithm is a probability distribution on the set of deterministic algorithms.

Assume fixed problem dimension n

$\rightarrow$ finitely many problem instances, e. g., graphs

$\rightarrow$ finitely many deterministic heuristics (if we omit to evaluate search points twice).

	deterministic search algorithms	A
problem instances		
f		$T(f, A)$ $=$ number of fitness evaluations until A finds optimum of f $=$ money algorithm designer has to pay to adversary

$P = \{p|$ prob. distribution on problem instances$\}$
$Q = \{q|$ prob. distribution on det. search algos$\}$

Minimax Theorem for Two-Person Zero-Sum Games

$$\min_{q\in Q} \max_{f} T(f, A_q) \quad = \quad \max_{p\in P} \min_{A} T(f_p, A)$$
$$\left(\quad \geq \quad \min_{A} T(f_p, A) \text{ for any } p \in P \right).$$

expected run time of best rand. algo. w. r. t. worst-case instance

average-case run time of best deterministic (!) algo. w. r. t. random instance

Application 1

Needle scenario

consider all $N_b(a) = \begin{cases} 1 & a = b \\ 0 & \text{otherwise.} \end{cases}$

$\rightarrow$ each rand. algo. needs expected time $\geq 2^{n-1} + \frac{1}{2}$ for at least one needle function.

Application 2

Unimodal functions

$$f : \{0,1\}^n \to \{0,\dots,b(n)\}, 2n \le b(n) = 2^{o(n)}$$

$\rightarrow$ each rand. algo. needs expected time $\Omega(b(n)/\log^2 b(n))$ for some unimodal functions.

15. Conclusions

– EAs are algorithms and should be analyzed as other algorithms.

– Algorithm analysis has a long history, is a fundamental discipline of computer science, deep results and clever methods are known.

The EA community should adopt methods from theoretical computer science like it has adopted methods from physics, engineering, and experimental disciplines.

$\rightarrow$ This is the way to become a well-respected core discipline in computer science.