Runtime Analysis of Evolutionary Algorithms: Basic Introduction[1]

Introductory Tutorial at GECCO 2019

Per Kristian Lehre
University of Birmingham
Birmingham B15 2TT, UK
P.K.Lehre@cs.bham.ac.uk

Pietro S. Oliveto
University of Sheffield
Sheffield S1 4DP, UK
P.Oliveto@sheffield.ac.uk

UNIVERSITY OF BIRMINGHAM

The University Of Sheffield.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author(s).

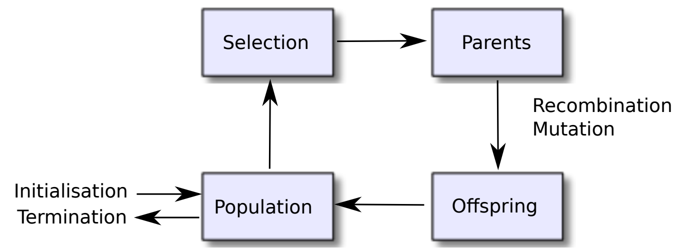
GECCO '19 Companion, Prague, Czech Republic

© 2019 Copyright is held by the owner/author(s).

ACM ISBN 978-1-4503-6748-6/19/07...$15.00.
doi:10.1145/3319619.3323394

[1] For the latest version of these slides, please go to http://www.cs.bham.ac.uk/~lehrepk/gecco2019

Evolutionary Algorithms


Aims and Goals of this Tutorial

- The scope of this tutorial is restricted to population-based
  - generational, non-elitist EAs (first part), and
  - steady-state, elitist EAs (second part).
- This tutorial will provide an overview of
  - the goals of runtime analysis of EAs
  - selected, generally applicable techniques
- You should attend if you wish to
  - theoretically understand the behaviour and performance of the EAs you design
  - familiarise yourself with some of the techniques used
  - pursue research in the area
- enable you or enhance your ability to
  1. understand theoretically population-dynamics of EAs on different problems
  2. perform time complexity analysis of population-based EAs on common toy problems
  3. have the basic skills to start independent research in the area

Outline

Black Box Optimisation Algorithms and Runtime

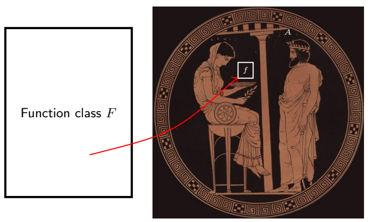


Photo: E. Gerhard (1846).

- An unknown optimisation problem f is chosen, possiby adversarily, from a problem class F known to the algorithm.
- For every $t \in \mathbb{N}$, using the obtained information $(x_1, f(x_1)), \ldots, (x_t, f(x_t))$, the algorithm queries a new search point x_{t+1} to obtain $f(x_{t+1})$ from the oracle.

Definition
The **runtime** of algorithm A on fitness function[2] $f : \{0,1\}^n \to \mathbb{R}$ is

$$T_{A,f} := \min_{t \in \mathbb{N}} \{t \mid \forall y \in \{0,1\}^n,\ f(x_t) \geq f(y)\}.$$

The **worst case** expected runtime of algorithm A on problem class F is

$$T_{A,F} := \max_{f \in F} \mathbf{E}\left[T_{A,f}\right]$$

Droste, Jansen, and Wegener [2006]

[2] This definition assumes that the objective is to maximise the fitness function.

Black Box Optimisation Algorithms and Runtime

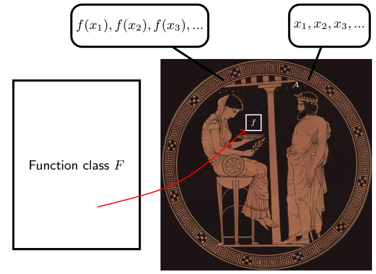


Photo: E. Gerhard (1846).

- An unknown optimisation problem f is chosen, possiby adversarily, from a problem class F known to the algorithm.
- For every $t \in \mathbb{N}$, using the obtained information $(x_1, f(x_1)), \ldots, (x_t, f(x_t))$, the algorithm queries a new search point x_{t+1} to obtain $f(x_{t+1})$ from the oracle.

Definition
The **runtime** of algorithm A on fitness function[2] $f : \{0,1\}^n \to \mathbb{R}$ is

$$T_{A,f} := \min_{t \in \mathbb{N}} \{t \mid \forall y \in \{0,1\}^n,\ f(x_t) \geq f(y)\}.$$

The **worst case** expected runtime of algorithm A on problem class F is

$$T_{A,F} := \max_{f \in F} \mathbf{E}\left[T_{A,f}\right]$$

Droste et al. [2006]

[2] This definition assumes that the objective is to maximise the fitness function.

Black Box Optimisation Algorithms and Runtime

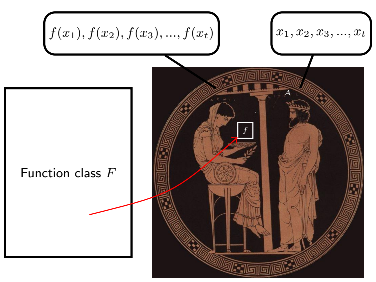


Photo: E. Gerhard (1846).

- An unknown optimisation problem f is chosen, possiby adversarily, from a problem class F known to the algorithm.
- For every $t \in \mathbb{N}$, using the obtained information $(x_1, f(x_1)), \ldots, (x_t, f(x_t))$, the algorithm queries a new search point x_{t+1} to obtain $f(x_{t+1})$ from the oracle.

Definition
The **runtime** of algorithm A on fitness function[2] $f : \{0,1\}^n \to \mathbb{R}$ is

$$T_{A,f} := \min_{t \in \mathbb{N}} \{t \mid \forall y \in \{0,1\}^n,\ f(x_t) \geq f(y)\}.$$

The **worst case** expected runtime of algorithm A on problem class F is

$$T_{A,F} := \max_{f \in F} \mathbf{E}\left[T_{A,f}\right]$$

Droste et al. [2006]

[2] This definition assumes that the objective is to maximise the fitness function.

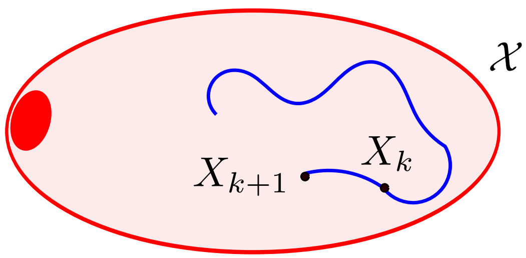
Runtime Analysis of Evolutionary Algorithms

Analysis of Evolutionary Algorithms

- Infinite population size
- Markov chain analysis He and Yao [2003]
- No parent population, or monomorphic populations
  - (1+1) EA
  - (1+λ) EA Jansen, Jong, and Wegener [2005]
  - (1,λ) EA Rowe and Sudholt [2012]
- Fitness-level techniques
  - (μ+1) EA Witt [2006]
  - (N+N) EAs Chen, He, Sun, Chen, and Yao [2009]
  - non-elitist EAs with unary variation operators Lehre [2011b], Dang and Lehre [2014]
  - (μ+1) GA Corus and Oliveto [2017]
- Stochastic dominance Doerr [2018]
- Drift analysis and martingale theory
  - Fitness proportionate selection Neumann, Oliveto, and Witt [2009a], Oliveto and Witt [2014, 2015]
- Family trees
  - (μ+1) EA Witt [2006]
  - (μ+1) IA Zarges [2009]
- Multi-type branching processes Lehre and Yao [2012]
  - Negative drift theorem for populations Lehre [2011a]
- Level-based analysis Corus, Dang, Eremeev, and Lehre [2014]

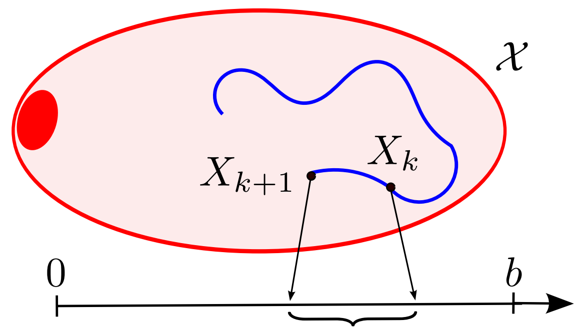
Drift Analysis

What is Drift[3] Analysis?



[3]NB! (Stochastic) drift is a different concept than *genetic drift* in population genetics.

What is Drift[3] Analysis?



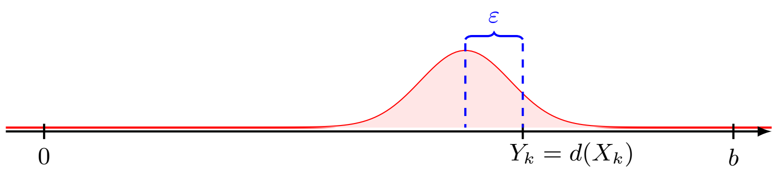
$$\Delta_k := d(X_{k+1}) - d(X_k)$$

- Prediction of the long term behaviour of a process X
  - hitting time, stability, occupancy time etc.

  from properties of Δ.

[3]NB! (Stochastic) drift is a different concept than *genetic drift* in population genetics.

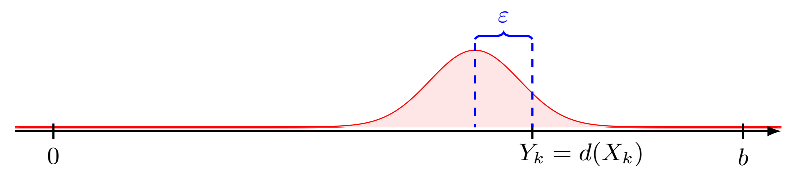
Additive Drift Theorem



(C1+) $\forall k \quad \mathbf{E}\left[Y_{k+1} - Y_k \mid Y_k > 0\right] \leq -\varepsilon$

(C1−) $\forall k \quad \mathbf{E}\left[Y_{k+1} - Y_k \mid Y_k > 0\right] \geq -\varepsilon$

Additive Drift Theorem



(C1+) $\forall k \quad \mathbf{E}\left[Y_{k+1} - Y_k \mid Y_k > 0\right] \leq -\varepsilon$

(C1−) $\forall k \quad \mathbf{E}\left[Y_{k+1} - Y_k \mid Y_k > 0\right] \geq -\varepsilon$

Theorem (He and Yao [2001], Jägersküpper [2007], Jägersküpper [2008])

Given a stochastic process $Y_1, Y_2, \ldots$ over an interval $[0, b] \subset \mathbb{R}$.
Define $T := \min\{k \geq 0 \mid Y_k = 0\}$, and assume $\mathbf{E}\left[T\right] < \infty$.

- *If (C1+) holds for an $\varepsilon > 0$, then $\mathbf{E}\left[T \mid Y_0\right] \leq b/\varepsilon$.*
- *If (C1−) holds for an $\varepsilon > 0$, then $\mathbf{E}\left[T \mid Y_0\right] \geq Y_0/\varepsilon$.*

LeadingOnes

$$\textsc{LeadingOnes}(x) := \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$$

$$x = \overbrace{1111111111111111}^{\text{Leading 1-bits}}\overbrace{0****************}^{\text{Remaining bits}}.$$

Left-most 0-bit

Runtime Analysis of (1,λ) EA on LeadingOnes

Theorem

The (1,λ) EA with $\lambda = n$ optimises LeadingOnes *in $O(n^2)$ expected time.*

Runtime Analysis of (1,λ) EA on LeadingOnes

Theorem
The (1,λ) EA with $\lambda = n$ optimises LeadingOnes *in $O(n^2)$ expected time.*

Proof

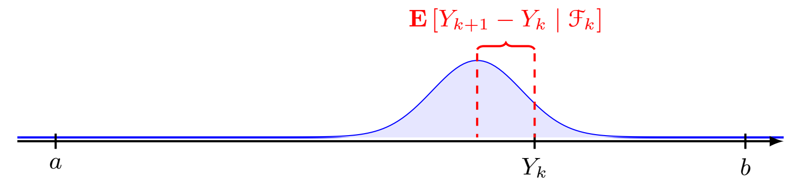
- Distance: let $d(x) = n - i$ where i is the number of leading ones;
- Drift:

$$\mathbf{E}\left[d(X_t) - d(X_{t+1}) | d(X_t) = n - i\right]$$
$$\geq 1 \cdot \left(1 - \left(1 - \frac{1}{n}\left(1 - \frac{1}{n}\right)^{n-1}\right)^{\lambda}\right) - n \cdot \left(1 - \left(1 - \frac{1}{n}\right)^{n}\right)^{\lambda}$$
$$= c_1 - n \cdot c_2^n = \Omega(1)$$

Hence,

$$\mathbf{E}[T] \leq \lambda \cdot \frac{\text{max distance}}{\text{drift}} = \lambda \cdot \frac{n}{\Omega(1)} = O(n^2)$$

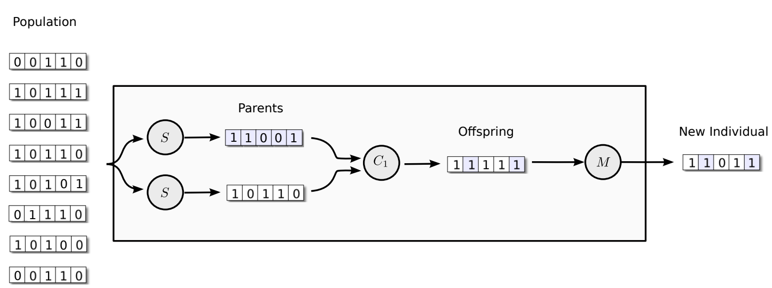
Some Drift Theorems



Drift Condition[4]	Statement	Note
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \leq Y_k - \varepsilon_0$	$\mathbf{E}[\tau_a] \leq$	Additive drift [He and Yao, 2001, Jägersküpper, 2008]
	$\Pr(\tau_a > B) \leq$	[Hajek, 1982]
	$\Pr(\tau_b < B) \leq$	Negative drift [Hajek, 1982, Oliveto and Witt, 2010]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \geq Y_k - \varepsilon_0$	$\leq \mathbf{E}[\tau_a]$	Additive drift (lower b.) [He and Yao, 2001, Jägersküpper, 2007]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \leq Y_k$	$\mathbf{E}[\tau_a] \leq$	Supermartingale [Neumann, Sudholt, and Witt, 2009b]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \leq \delta Y_k$	$\mathbf{E}[\tau_a] \leq$	Multiplicative drift [Doerr, Johannsen, and Winzen, 2010, Fischer, Olbrich, and Vöcking, 2008]
	$\Pr(\tau_a > B) \leq$	[Doerr and Goldberg, 2010]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \geq \delta Y_k$	$\leq \mathbf{E}[\tau_a]$	Multiplicative drift (lower b.) [Lehre and Witt, 2012]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \leq h(Y_k)$	$\mathbf{E}[\tau_a] \leq$	Variable drift [Johannsen, 2010]
$\mathbf{E}\left[Y_{k+1} \mid \mathcal{F}_k\right] \leq h(Y_k)$	$\Pr(\tau_a > B) \leq$	[Lehre and Witt, 2014]
	$\Pr(\tau_b < B) \leq$	[Lehre and Witt, 2014]
$\mathbf{E}\left[e^{\lambda Y_{k+1}} \mid \mathcal{F}_k\right] \leq \frac{e^{\lambda Y_k}}{\alpha_0}$	$\Pr(\tau_b < B) \leq$	Population drift [Lehre, 2011a]

[4] Some drift theorems need additional conditions.

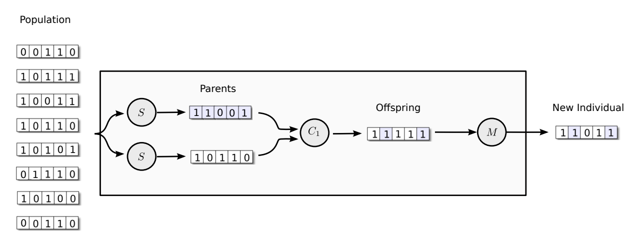
Outline of a Non-elitist, Generational Evolutionary Algorithm[5]



initialise a population P_0 of λ individuals uniformly at random.
for $t = 0, 1, 2, \ldots$ until termination condition **do**
    evaluate the individuals in population P_t.
    for $i = 1$ to λ **do**
        select two parents from population P_t.
        recombine the two parents.
        mutate the offspring and add it to population P_{t+1}.

[5] Pseudo-code adapted from Eiben and Smith [2003].

A Model of Non-elitist, Generational EAs

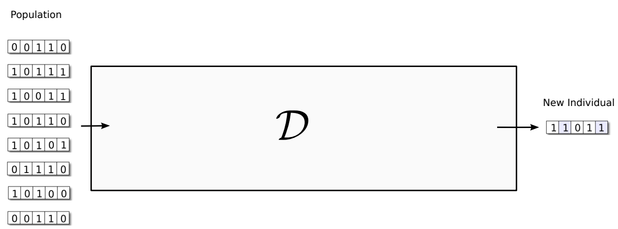


Wide range of evolutionary algorithms...

- selection mechanisms (ranking selection, (μ, λ)-selection, tournament selection, ...)
- fitness models (deterministic, stochastic, dynamic, partial, ...)
- variation operators
- search spaces (e.g. bitstrings, permutations, ...)

We will describe many of these with a general mathematical model.

A Model of Non-elitist, Generational EAs



Require: Search space $\mathcal{X}$ and random operator $\mathcal{D} : \mathcal{X}^\lambda \rightarrow \mathcal{X}$

```
P_0 ~ Unif(X^λ)
for t = 0, 1, 2, ... until termination condition do
    for i = 1 to λ do
        P_{t+1}(i) ~ D(P_t)
```


Level-based Theorem[6]

[6] Corus, Dang, Eremeev, and Lehre [2017]

Level-based Analysis

Problem

- Given any target set $B \subset \mathcal{X}$ (e.g. global optima), let

$$T_B := \min\{\lambda t \mid P_t \cap B \neq \emptyset\}$$

  where $P_0, P_1, \ldots$ are the populations generated by the algorithm.
- How does $\mathbf{E}[T_B]$ depend on $\mathcal{D}$ and λ? Informally, how much time does the algorithm require to discover the target set.

Level-based Theorem (LBT)

- If $\mathcal{D}$ and λ satisfy certain conditions, then LBT provides an upper bound for $\mathbf{E}[T_B]$.

Level Partitioning of Search Space $\mathcal{X}$

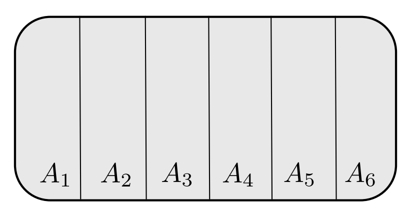
Definition

$(A_1, \ldots, A_m)$ is a level-partitioning of search space $\mathcal{X}$ if

- $\bigcup_{j=1}^m A_j = \mathcal{X}$ (together, the levels cover the search space)
- $A_i \cap A_j = \emptyset$ whenever $i \neq j$ (the levels are nonoverlapping)
- the last level A_m covers the optima for the problem

We write $A_{\geq j}$ to denote everything in level j and higher, i.e.,

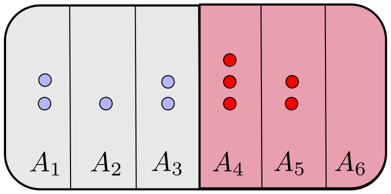
$$A_{\geq j} := \bigcup_{i=j}^m A_i.$$



Notation

- For any population $P = (y_1, \ldots, y_\lambda) \in \mathcal{X}^\lambda$ and $j \in [m]$, let
$$|P \cap A_{\geq j}| \ := \ |\{i \mid x_i \in A_{\geq j}\}|,$$
i.e, the number of individuals in P that is in subset $A_{\geq j}$.

Example



$|P \cap A_{\geq 4}| = 5$ where $A_{\geq 4}$ corresponds to the red region.

Current level of a population P wrt $\gamma_0 \in (0, 1)$

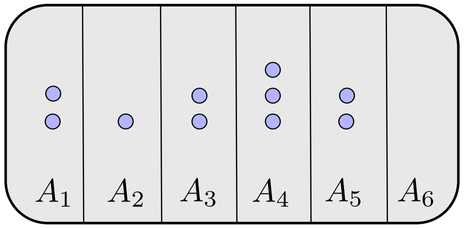
Definition

The unique integer $j \in [m - 1]$ such that

$$|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$$

Example

Current level wrt $\gamma_0 = \frac{1}{2}$ is .....



Current level of a population P wrt $\gamma_0 \in (0, 1)$

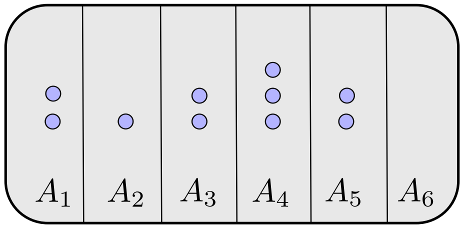
Definition

The unique integer $j \in [m - 1]$ such that

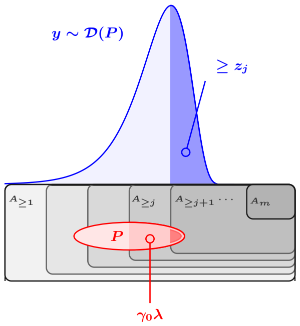
$$|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$$

Example

Current level wrt $\gamma_0 = \frac{1}{2}$ is 4.



Level-based Theorem [Corus, Dang, Eremeev, and Lehre, 2017]



If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

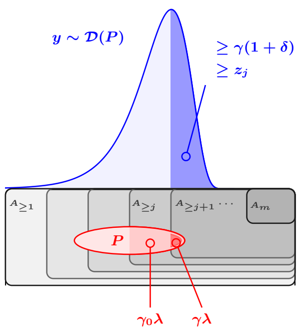
$$\mathbf{Pr}\left(y \in A_{\geq j+1}\right) \geq z_j, \qquad \text{(G1)}$$

(G2)

where $j \in [m - 1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$$

Level-based Theorem [Corus, Dang, Eremeev, and Lehre, 2017]



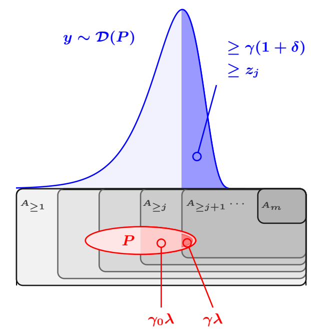
If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j, \quad \text{(G1)}$$
$$\Pr\left(y \in A_{\geq j+1}\right) \geq \gamma(1+\delta), \quad \text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0\lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

Level-based Theorem [Corus, Dang, Eremeev, and Lehre, 2017]



If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j, \quad \text{(G1)}$$
$$\Pr\left(y \in A_{\geq j+1}\right) \geq \gamma(1+\delta), \quad \text{(G2)}$$

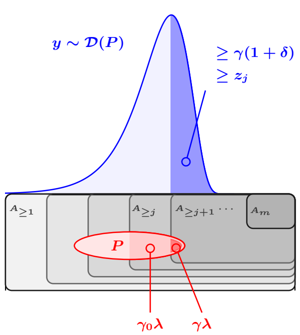
where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0\lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

and the population size λ is bounded from below by

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right) \ln\left(\frac{128m}{z_{\min}\delta^2}\right), \quad \text{(G3)}$$

Level-based Theorem [Corus, Dang, Eremeev, and Lehre, 2017]



If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j, \quad \text{(G1)}$$
$$\Pr\left(y \in A_{\geq j+1}\right) \geq \gamma(1+\delta), \quad \text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0\lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

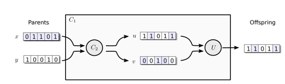
and the population size λ is bounded from below by

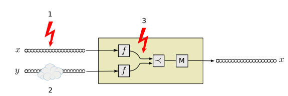
$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right) \ln\left(\frac{128m}{z_{\min}\delta^2}\right), \quad \text{(G3)}$$

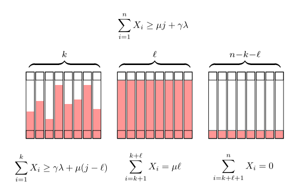
then the algorithm reaches the last level A_m in expected time

$$\mathrm{E}\left[T_{A_m}\right] \leq \left(\frac{8}{\delta^2}\right) \sum_{j=1}^{m-1} \left(\lambda \ln\left(\frac{6\delta\lambda}{4 + z_j\delta\lambda}\right) + \frac{1}{z_j}\right).$$

Applications of Level-based Theorem







- **EAs with crossover operator**
  - (Corus, Dang, Eremeev, Lehre, PPSN'14)
- **Theory-led design of EAs**
  - Shortest paths with genetic algorithms (Corus & Lehre, MIC'15)
- **Optimisation under uncertainty**
  - Stochastic fitness functions (Dang & Lehre, FOGA'15)
  - Partially observable fitness functions (Dang & Lehre, GECCO'14)
  - Dynamic fitness functions (Dang, Jansen, Lehre, GECCO'15)
- **Estimation of Distribution Algorithms (EDAs)**
  - UMDA (Dang & Lehre, GECCO'15) (Nguyen & Lehre, GECCO'17)
  - PBIL (Lehre & Nguyen, PPSN'18)
  - GOMEA (Hoog, 2016)
- **Self-adaptive EAs**
  - Self-adaptation of mutation rates (Dang & Lehre, PPSN'16)

Suggested recipe for application of level-based theorem

1. Find a partition $(A_1, \ldots, A_m)$ of $\mathcal{X}$ that reflects the state of the algorithm, and where A_m consists of all goal states.
2. Find parameters γ_0 and δ and a configuration of the algorithm (e.g., mutation rate, selective pressure) such that whenever $|P \cap A_{\geq j+1}| = \gamma\lambda > 0$, condition (G2) holds, i.e.,

$$\Pr\left(y \in A_{\geq j+1}\right) \geq \gamma(1+\delta)$$

3. For each level $j \in [m-1]$, estimate a lower bound $z_j \in (0,1)$ such that whenever $|P \cap A_{\geq j+1}| = 0$, condition (G1) holds, i.e.,

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j$$

4. Calculate the sufficient population size λ from condition (G3).
5. Read off the bound on expected runtime.

The Level-Based Theorem (LBT) is "tight"[7]

Theorem
For any valid set of parameters $\Theta = (A, \gamma_0, \delta, z)$ for LBT, there exists a mapping $\mathcal{D}_{slow}$ satisfying (G1) and (G2) of LBT st.

$$\mathbf{E}\left[T_{A_m}\right] \geq \left(\frac{2}{3\delta}\right) \sum_{j=1}^{m-2} \left(\lambda \ln\left(\frac{\gamma_0\delta\lambda}{1/\delta + 2z_j\delta\lambda}\right) + \frac{1}{z_j}\right)$$

[7] Corus, Dang, Eremeev and Lehre (IEEE TEVC 2018) `https://arxiv.org/abs/1407.7663`

The Level-Based Theorem (LBT) is "tight"[7]

Theorem
For any valid set of parameters $\Theta = (A, \gamma_0, \delta, z)$ for LBT, there exists a mapping $\mathcal{D}_{slow}$ satisfying (G1) and (G2) of LBT st.

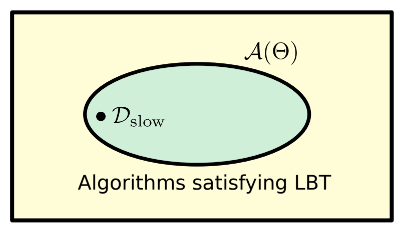
$$\mathbf{E}\left[T_{A_m}\right] \geq \left(\frac{2}{3\delta}\right) \sum_{j=1}^{m-2} \left(\lambda \ln\left(\frac{\gamma_0\delta\lambda}{1/\delta + 2z_j\delta\lambda}\right) + \frac{1}{z_j}\right)$$

$$\mathbf{E}\left[T_{A_m}\right] \leq \left(\frac{8}{\delta^2}\right) \sum_{j=1}^{m-1} \left(\lambda \ln\left(\frac{6\delta\lambda}{4 + z_j\delta\lambda}\right) + \frac{1}{z_j}\right).$$

[7] Corus, Dang, Eremeev and Lehre (IEEE TEVC 2018) `https://arxiv.org/abs/1407.7663`

More info about $\mathcal{D}$ required for more precise bounds

All algorithms



Logical structure of LBT
For all parameter settings Θ, and all mappings $\mathcal{D} \in \mathcal{A}(\Theta)$

$$\mathbf{E}\left[T_{\mathcal{D}}\right] \leq f(\Theta) + \varepsilon.$$

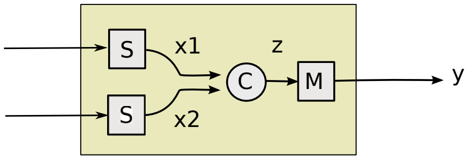
Also, there exists $\mathcal{D} \in \mathcal{A}(\Theta)$ st

$$\mathbf{E}\left[T_{\mathcal{D}}\right] \geq f(\Theta) - \varepsilon$$

Assume you have applied the LBT to your algorithm, how precise is the bound?

- The only LBT knows about your algorithm $\mathcal{D}$ is that it satisfies the conditions for the parameters Θ. (Many other processes satisfy the conditions for the same Θ.)
- The lower bound implies that the LBT gives the best possible ($\pm\varepsilon$) runtime bound for your algorithm given the information that is available.
- Some algorithms in $\mathcal{A}(\Theta)$, including yours, could be faster than $f(\Theta)$. However, more information about the algorithm required to prove so, i.e.,
  - a more precise set of parameters Θ', or
  - a different way of characterising algorithms than $\mathcal{A}(\Theta)$

Example application – (μ,λ) GA on Onemax



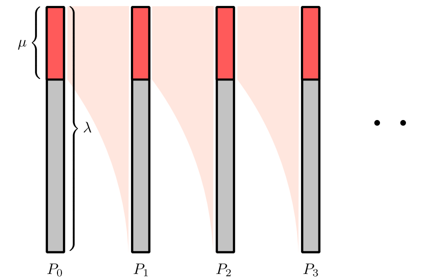
$$\textsc{OneMax}(x) := \sum_{i=1}^{n} x_i.$$

(μ,λ) Genetic Algorithm (GA)

for $t = 0, 1, 2, \ldots$ until termination condition **do**
  for $i = 1$ to λ **do**
    Select a parent x from population P_t acc. to (μ, λ)-selection
    Select a parent y from population P_t acc. to (μ, λ)-selection
    Apply uniform crossover to x and y, i.e. $z :=$ crossover(x, y)
    Create $P_{t+1}(i)$ by flipping each bit in z with probability χ/n.

Theorem
If $\lambda > c\ln(n)$ for a sufficiently large constant $c > 0$, and $\frac{\lambda}{\mu} > 2e^{\chi}(1+\delta)$ for any constant $\delta > 0$, then the expected runtime of (μ,λ) GA on $\textsc{OneMax}$ *is $O(n\lambda)$.*

(μ, λ)-selection mechanism



1. Sort the current population $P = (x_1, \ldots, x_\lambda)$ such that

$$f(x_1) \geq f(x_2) \geq \ldots \geq f(x_\lambda)$$

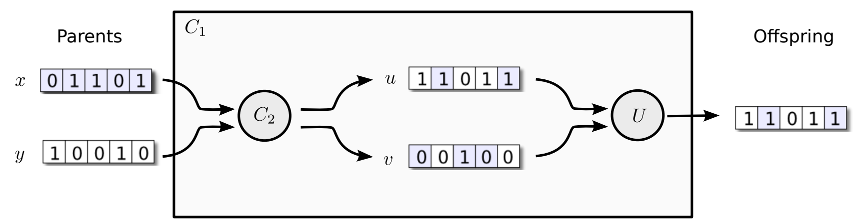
2. **return** $\mathrm{Unif}(x_1, \ldots, x_\mu)$

Partition of Search Space into Levels

Partition into $m := n + 1$ levels $A_0, \ldots, A_n$

$$A_j := \{x \in \{0,1\}^n \mid \textsc{OneMax}(x) = j\}$$

Analysis of Crossover Operator

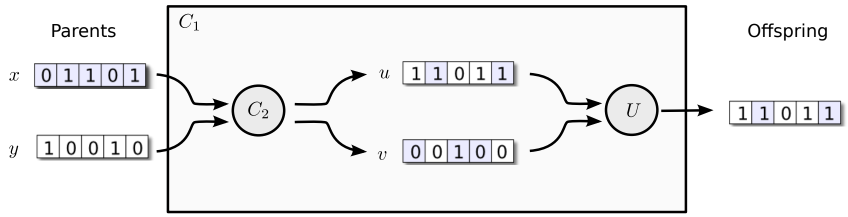


Proof.
Assume that $x \in A_{\geq j+1}$ and $y \in A_{\geq j}$,

$$2j + 1 \leq |x| + |y|$$

□

Analysis of Crossover Operator

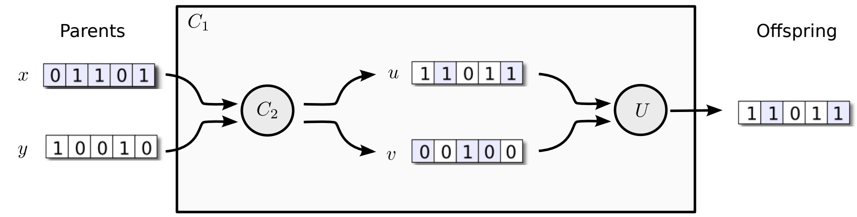


Proof.
Assume that $\boldsymbol{x} \in A_{\geq j+1}$ and $\boldsymbol{y} \in A_{\geq j}$,

$$\begin{aligned} 2j+1 &\leq |\boldsymbol{x}| + |\boldsymbol{y}| \\ &= |\boldsymbol{u}| + |\boldsymbol{v}| \end{aligned}$$

□

Analysis of Crossover Operator

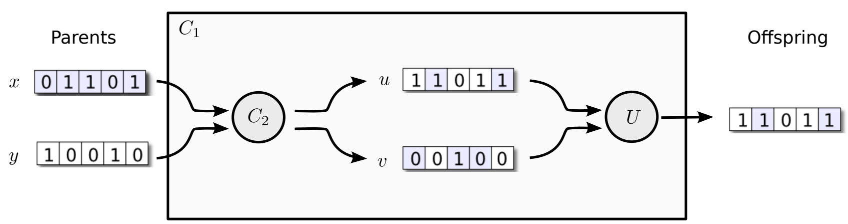


Proof.
Assume that $\boldsymbol{x} \in A_{\geq j+1}$ and $\boldsymbol{y} \in A_{\geq j}$, and w.l.o.g. that $|\boldsymbol{u}| \geq |\boldsymbol{v}|$

$$\begin{aligned} 2j+1 &\leq |\boldsymbol{x}| + |\boldsymbol{y}| \\ &= |\boldsymbol{u}| + |\boldsymbol{v}| \\ &\leq 2|\boldsymbol{u}|. \end{aligned}$$

□

Analysis of Crossover Operator



Proof.
Assume that $\boldsymbol{x} \in A_{\geq j+1}$ and $\boldsymbol{y} \in A_{\geq j}$, and w.l.o.g. that $|\boldsymbol{u}| \geq |\boldsymbol{v}|$

$$\begin{aligned} 2j+1 &\leq |\boldsymbol{x}| + |\boldsymbol{y}| \\ &= |\boldsymbol{u}| + |\boldsymbol{v}| \\ &\leq 2|\boldsymbol{u}|. \end{aligned}$$

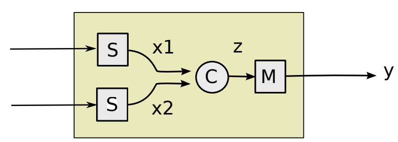
Therefore $\Pr\left(\boldsymbol{u} \in A_{\geq j+1}\right) = 1$ and

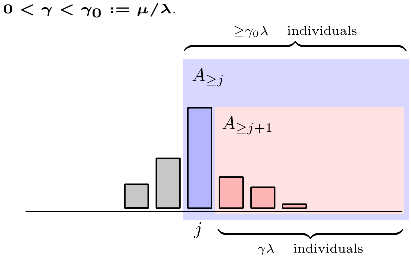
$$\Pr\left(\text{crossover}(\boldsymbol{x}, \boldsymbol{y}) \in A_{\geq j+1} \mid \boldsymbol{x} \in A_{\geq j+1} \text{ and } \boldsymbol{y} \in A_{\geq j}\right) \geq \frac{1}{2} =: \varepsilon.$$

□

Verification of Condition (G2)

Assume that $|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$ where $0 < \gamma < \gamma_0 := \mu/\lambda$.





we have

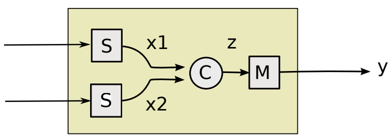
$$\begin{aligned} \Pr\left(z \in A_{\geq j+1}\right) &= \Pr\left(x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j+1}\right) \Pr\left(z \in A_{\geq j+1} \mid x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j+1}\right) \\ &\geq 1 \cdot \frac{\gamma\lambda}{\mu} \cdot (1/2) \end{aligned}$$

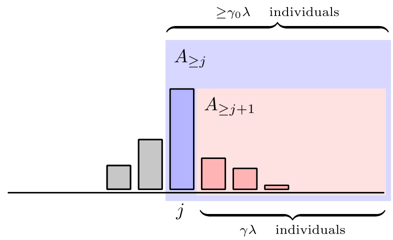
hence, we get

$$\begin{aligned} \Pr\left(y \in A_{\geq j+1}\right) &= \Pr\left(z \in A_{\geq j+1}\right) \Pr\left(y \in A_{\geq j+1} \mid z \in A_{\geq j+1}\right) \\ &\geq \Pr\left(z \in A_{\geq j+1}\right) \left(1 - \frac{\chi}{n}\right)^n \\ &\geq \end{aligned}$$

Verification of Condition (G2)

Assume that $|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$ where $0 < \gamma < \gamma_0 := \mu/\lambda$.





If $\lambda \geq 2\mu \left(1 - \frac{\chi}{n}\right)^{-n} (1+\delta) \approx 2\mu e^{\chi}$, we have

$$\begin{aligned}\Pr\left(z \in A_{\geq j+1}\right) &= \Pr\left(x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j+1}\right) \Pr\left(z \in A_{\geq j+1} \mid x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j+1}\right) \\ &\geq 1 \cdot \frac{\gamma\lambda}{\mu} \cdot (1/2) \geq \gamma(1+\delta)\left(1 - \frac{\chi}{n}\right)^{-n}\end{aligned}$$

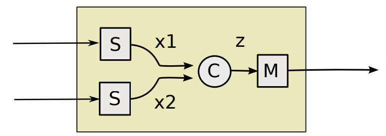
hence, we get

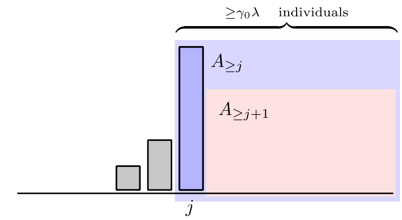
$$\begin{aligned}\Pr\left(y \in A_{\geq j+1}\right) &= \Pr\left(z \in A_{\geq j+1}\right) \Pr\left(y \in A_{\geq j+1} \mid z \in A_{\geq j+1}\right) \\ &\geq \Pr\left(z \in A_{\geq j+1}\right)\left(1 - \frac{\chi}{n}\right)^{n} \\ &\geq \gamma(1+\delta).\end{aligned}$$

30 / 71

Verification of Condition (G1)

Assume that $|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$ where $0 \leq \gamma < \gamma_0 := \mu/\lambda$.





We have

$$\begin{aligned}\Pr\left(z \in A_{\geq j}\right) &= \Pr\left(x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j}\right) \Pr\left(z \in A_{\geq j} \mid x_1 \in A_{\geq j} \wedge x_2 \in A_{\geq j}\right) \\ &\geq 1 \cdot 1/2\end{aligned}$$

hence, we get

$$\begin{aligned}\Pr\left(y \in A_{\geq j+1}\right) &= \Pr\left(z \in A_{\geq j}\right) \Pr\left(y \in A_{\geq j+1} \mid z \in A_{\geq j+1}\right) \\ &\geq \Pr\left(z \in A_{\geq j}\right) \cdot (n-j) \cdot \left(\frac{\chi}{n}\right)\left(1 - \frac{\chi}{n}\right)^{n-1} \\ &\geq (\chi/2) \cdot (1 - j/n)\left(1 - \frac{\chi}{n}\right)^{n-1} \\ &\geq \frac{(1-\delta)\chi}{2e^{\chi}} \cdot (1 - j/n) =: z_j\end{aligned}$$

31 / 71

Application of the Level-based Theorem

If for all populations $P \in \mathcal{X}^{\lambda}$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j, \qquad \text{(G1)}$$
$$\text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}|$$

32 / 71

Application of the Level-based Theorem

If for all populations $P \in \mathcal{X}^{\lambda}$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr\left(y \in A_{\geq j+1}\right) \geq z_j, \qquad \text{(G1)}$$
$$\Pr\left(y \in A_{\geq j+1}\right) \geq \gamma(1+\delta), \qquad \text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0 \lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

32 / 71

Application of the Level-based Theorem

If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr(y \in A_{\geq j+1}) \geq z_j, \quad \text{(G1)}$$
$$\Pr(y \in A_{\geq j+1}) \geq \gamma(1+\delta), \quad \text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0\lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

and the population size λ is bounded from below by

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right) \ln\left(\frac{128m}{z_{\min}\delta^2}\right), \quad \text{(G3)}$$

Application of the Level-based Theorem

If for all populations $P \in \mathcal{X}^\lambda$, an individual $y \sim \mathcal{D}(P)$ has

$$\Pr(y \in A_{\geq j+1}) \geq z_j, \quad \text{(G1)}$$
$$\Pr(y \in A_{\geq j+1}) \geq \gamma(1+\delta), \quad \text{(G2)}$$

where $j \in [m-1]$ is the current level of population P, i.e.,

$$|P \cap A_{\geq j}| \geq \gamma_0\lambda > |P \cap A_{\geq j+1}| = \gamma\lambda,$$

and the population size λ is bounded from below by

$$\lambda \geq \left(\frac{4}{\gamma_0\delta^2}\right) \ln\left(\frac{128m}{z_{\min}\delta^2}\right), \quad \text{(G3)}$$

then the algorithm reaches the last level A_m in expected time

$$\mathbf{E}[T_{A_m}] \leq \left(\frac{8}{\delta^2}\right) \sum_{j=1}^{m-1} \left(\lambda \ln\left(\frac{6\delta\lambda}{4+z_j\delta\lambda}\right) + \frac{1}{z_j}\right).$$

Bounding the first term (first attempt, imprecise)

$$\sum_{j=0}^{n-1} \ln\left(\frac{6\delta\lambda}{4+z_j\delta\lambda}\right) < \sum_{j=0}^{n-1} \ln\left(\frac{6\delta\lambda}{4}\right) = \mathcal{O}(n\ln(\lambda)).$$

- This upper bound is imprecise because it does not exploit that the upgrade probabilities z_j are large for small j.

Bounding the first term (second attempt, more precise)

$$\sum_{j=0}^{n-1} \ln\left(\frac{6\delta\lambda}{4+z_j\delta\lambda}\right) < \sum_{j=0}^{n-1} \ln\left(\frac{6}{z_j}\right)$$

using $\ln(a)+\ln(b)=\ln(ab)$ and defining $c := \frac{12e^\chi}{(1-\delta)\chi}$

$$= \ln\left(\prod_{j=0}^{n-1} \frac{cn}{n-j}\right) = \ln\left(\frac{(cn)^n}{n!}\right)$$

and using the lower bound $n! > (n/e)^n$

$$< \ln\left(\frac{(cn)^n e^n}{n^n}\right) = n\ln(ec) = \mathcal{O}(n).$$

Bounding the second term

Recall the definition of the n-th Harmonic number

$$H_n := \sum_{i=1}^{n} \frac{1}{i} = \mathcal{O}(\ln(n)).$$

The second term can therefore be bounded as

$$\sum_{j=0}^{n-1} \frac{1}{z_j} = \mathcal{O}\left(\sum_{j=0}^{n-1} \frac{n}{n-j}\right) = \mathcal{O}(n \ln(n))$$

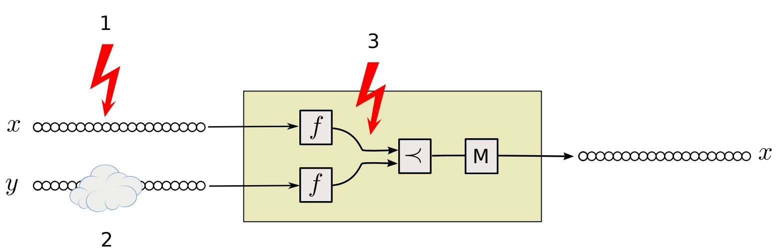
Completing the proof

Theorem

If $\lambda > c\ln(n)$ for a sufficiently large constant $c > 0$, and $\frac{\lambda}{\mu} > 2e^{\chi}(1+\delta)$ for any constant $\delta > 0$, then the expected runtime of (μ,λ) GA on OneMax *is*

$$\left(\frac{8}{\delta^2}\right)\left(\lambda \sum_{j=0}^{n-1} \ln\left(\frac{6\delta\lambda}{4 + z_j\delta\lambda}\right) + \sum_{j=0}^{n-1} \frac{1}{z_j}\right)$$
$$= \mathcal{O}(n\lambda) + \mathcal{O}(n \ln n) = \mathcal{O}(n\lambda).$$

Uncertainty in Comparison-based PSVAs



Sources of uncertainty

1. Droste noise model (Droste, 2004)
2. Partial evaluation
3. Noisy fitness (Prügel-Bennet, Rowe, Shapiro, 2015)

Sufficient with mutation rate $\delta/(3n)$ and

$$\mathbf{Pr}\left(x \text{ choosen} \mid f(x) > f(y)\right) \geq \frac{1}{2} + \delta \quad \text{with } 1/\delta \in \mathrm{poly}(n)$$

Dang and Lehre [2015] and Dang and Lehre [2016a]

Lower Bounds

Lower Bounds

Problem
Consider a sequence of populations $P_1, \ldots$ over a search space $\mathcal{X}$, and a target region $A \subset \mathcal{X}$ (e.g., the set of optimal solutions), let

$$T_A := \min\{ \lambda t \mid P_t \cap A \neq \emptyset \}$$

We would like to prove statements on the form

$$\Pr(T_A \leq t(n)) \leq e^{-\Omega(n)}. \quad (1)$$

- i.e., with overwhelmingly high probability, the target region A has not been found in $t(n)$ evaluations
- lower bounds often harder to prove than upper bounds
- will present an easy to use method that is applicable in many situations

Algorithms considered for lower bounds

Definition (Non-elitist EA with selection and mutation)

for $t = 0, 1, 2, \ldots$ until termination condition **do**
  for $i = 1$ to λ **do**
    Select parent x from population P_t according to p_{sel}
    Flip each position in x independently with probability χ/n.
    Let the i-th offspring be $P_{t+1}(i) := x$.
    (i.e., create offspring by mutating the parent)

Assumptions
- population size $\lambda \in \text{poly}(n)$, i.e. not exponentially large
- bitwise mutation with probability χ/n, but no crossover.
- results hold for any non-elitist selection scheme p_{sel} that satisfy some mild conditions to be described later.

Reproductive rate[8]

Definition
For any population $P = (x_1, \ldots, x_\lambda)$ let $p_{\text{sel}}(x_i)$ be the probability that individual x_i is selected from the population P

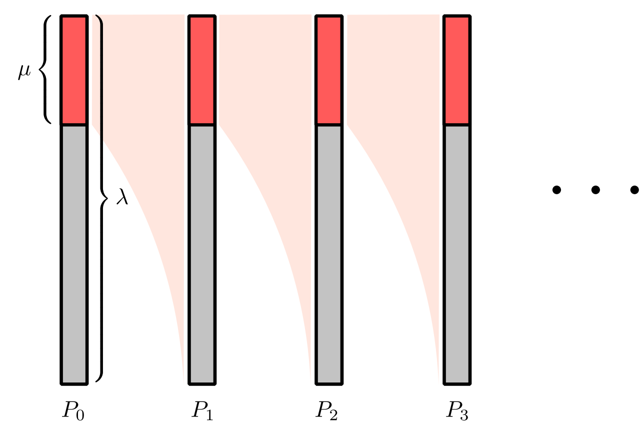
- The reproductive rate of individual x_i is $\lambda \cdot p_{\text{sel}}(x_i)$.
- The reproductive rate of a selection mechanism is bounded from above by α_0 if

$$\forall P \in \mathcal{X}^\lambda, \ \forall x \in P \quad \lambda \cdot p_{\text{sel}}(x) \leq \alpha_0$$

(i.e., no individual gets more than α_0 offspring in expectation)

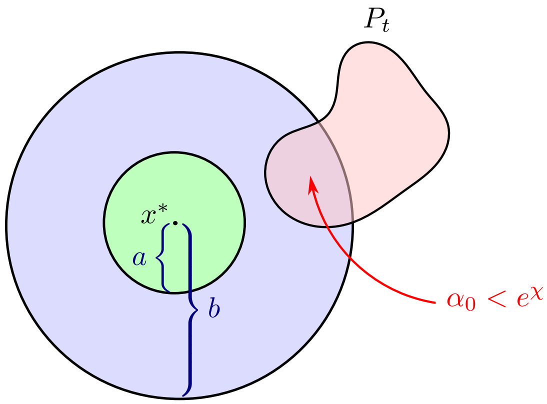
[8] The reproductive rate of an individual as defined here corresponds to the notion of "fitness" as used in the field of population genetics, i.e., the expected number of offspring.

(μ, λ)-selection mechanism



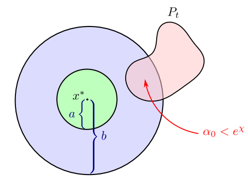
Probability of selecting i-th individual is $p_i \in \{0, \frac{1}{\mu}\}$.
- reproductive rate bounded by $\alpha_0 = \lambda/\mu$

Negative Drift Theorem for Populations (informal)



If individuals closer than b of target has reproductive rate $\alpha_0 < e^{\chi}$, then it takes exponential time $e^{c(b-a)}$ to reach within a of target.

Negative Drift Thm. for Populations [Lehre, 2011a]



Consider the non-elitist EA with

- population size $\lambda = \mathrm{poly}(n)$
- bitwise mutation rate χ/n for $0 < \chi < n$

let $T := \min\{t \mid H(P_t, x^*) \leq a\}$ for any $x^* \in \{0,1\}^n$.

If there are constants $\alpha_0 \geq 1$, $\delta > 0$ and integers $a(n)$ and $b(n) < \frac{n}{\chi}$ where $b(n) - a(n) = \omega(\ln n)$, st.

(C1) If $a(n) < H(x, x^*) < b(n)$ then $\lambda \cdot p_{\mathrm{sel}}(x) \leq \alpha_0$.

(C2) $\psi := \ln(\alpha_0)/\chi + \delta < 1$

(C3) $b(n) < \min\left\{\frac{n}{5}, \frac{n}{2}\left(1 - \sqrt{\psi(2-\psi)}\right)\right\}$

then there exist constants $c, c' > 0$ such that

$$\Pr\left(T \leq e^{c(b(n)-a(n))}\right) \leq e^{-c'(b(n)-a(n))}.$$

The worst individuals have low reproductive rate

Lemma

Consider any selection mechanism which for $x, y \in P$ satisfies

(a) *If $f(x) > f(y)$, then $p_{sel}(x) \geq p_{sel}(y)$.*
(selection probabilities are monotone wrt fitness)

(b) *If $f(x) = f(y)$, then $p_{sel}(x) = p_{sel}(y)$.*
(ties are drawn randomly)

If $f(x) = \min_{y \in P} f(y)$, then $p_{sel}(x) \leq 1/\lambda$.
(individuals with lowest fitness have reproductive rate ≤ 1)

Proof.

- By (a) and (b), $p_{\mathrm{sel}}(x) = \min_{y \in P} p_{\mathrm{sel}}(y)$.
- $1 = \sum_{x \in P} p_{\mathrm{sel}}(x) \geq \lambda \cdot \min_{y \in P} p_{\mathrm{sel}}(y) = \lambda \cdot p_{\mathrm{sel}}(x)$.

□

Example 1: Needle in the haystack

Definition

$$\textsc{Needle}(x) = \begin{cases} 1 & \text{if } x = 1^n \\ 0 & \text{otherwise.} \end{cases}$$

Theorem

The optimisation time of the non-elitist EA with any selection mechanism satisfying the properties above[9] *on* $\textsc{Needle}$ *is at least e^{cn} with probability $1 - e^{-\Omega(n)}$ for some constant $c > 0$.*

[9] From black-box complexity theory, it is known that $\textsc{Needle}$ is hard for all search heuristics (Droste et al 2006).

Example 1: Needle in the haystack (proof[10])

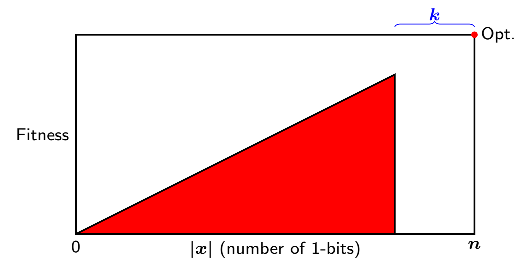
- Apply negative drift theorem with $a(n) := 1$.
- By previous lemma, can choose $\alpha_0 = 1$ for any $b(n)$, hence $\psi = \ln(\alpha)/\chi + \delta = \delta < 1$ for all χ and $\delta < 1$.
- Choosing the parameters $\delta := 1/10$ and $b(n) := n/6$ give

$$\min\left\{\frac{n}{5}, \frac{n}{2}\left(1 - \sqrt{\psi(2-\psi)}\right)\right\} = \frac{n}{5} > b(n).$$

- It follows that $\Pr\left(T \leq e^{c(b(n)-a(n))}\right) \leq e^{-\Omega(n)}$.

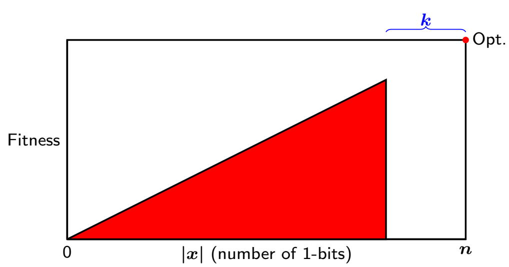
[10] For simplicity, we assume that $\chi \leq 6$, thus $b(n) = n/6 \leq n/\chi$ holds.

Exercise: Optimisation time on Jump$_k$



$$\text{JUMP}_k(x) := \begin{cases} 0 & \text{if } n-k \leq |x| < n, \\ |x| & \text{otherwise.} \end{cases}$$

Exercise: Optimisation time on Jump$_k$



Recipe

- $a(n) = 1$
- $b(n) = k$
- $\alpha_0 = 1$ as before
- small δ

$$\text{JUMP}_k(x) := \begin{cases} 0 & \text{if } n-k \leq |x| < n, \\ |x| & \text{otherwise.} \end{cases}$$

When the best individuals have low reproductive rate

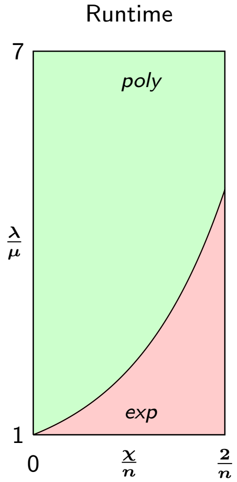
Remark

- The negative drift conditions hold trivially if $\alpha_0 < e^{\chi}$ holds for all individuals.

Example (Insufficient selective pressure)

Selection mechanism	Parameter settings
Linear ranking selection	$\eta < e^{\chi}$
k-tournament selection	$k < e^{\chi}$
(μ,λ)-selection	$\lambda < \mu e^{\chi}$
Any in cellular EAs	$\Delta(G) < e^{\chi}$

Mutation-selection balance



Example

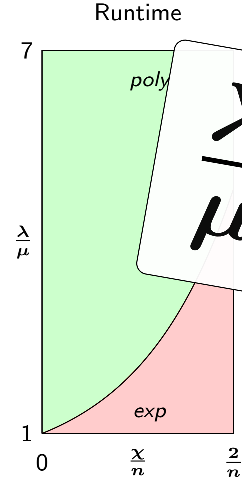
The runtime T of a non-elitist EA with

- (μ, λ)-selection
- bit-wise mutation rate χ/n
- population size $\lambda > c\log(n)$

on LeadingOnes has expected value

$$\mathbf{E}[T] = \begin{cases} e^{\Omega(n)} & \text{if } \lambda < \mu e^{\chi} \\ O(n\lambda\ln(\lambda) + n^2) & \text{if } \lambda > \mu e^{\chi} \end{cases}$$

Mutation-selection balance



$$\frac{\lambda}{\mu} > e^{\chi}$$

Example

The runtime T of a non-elitist EA with

- (μ, λ)-selection
- bit-wise mutation rate χ/n
- population size $\lambda > c\log(n)$

on LeadingOnes has expected value

$$\mathbf{E}[T] = \begin{cases} e^{\Omega(n)} & \text{if } \lambda < \mu e^{\chi} \\ O(n\lambda\ln(\lambda) + n^2) & \text{if } \lambda > \mu e^{\chi} \end{cases}$$

Other Example Applications

Expected runtime of EA with bit-wise mutation rate χ/n

Selection Mechanism	High Selective Pressure
Fitness Proportionate	$\nu > f_{\max}\ln(2e^{\chi})$
Linear Ranking	$\eta > e^{\chi}$
k-Tournament	$k > e^{\chi}$
(μ, λ)	$\lambda > \mu e^{\chi}$
Cellular EAs	
OneMax	$O(n\lambda)$
LeadingOnes	$O(n\lambda\ln(\lambda) + n^2)$
Linear Functions	$O(n\lambda\ln(\lambda) + n^2)$
r-Unimodal	$O(r\lambda\ln(\lambda) + nr)$
Jump$_r$	$O(n\lambda + (n/\chi)^r)$

Other Example Applications

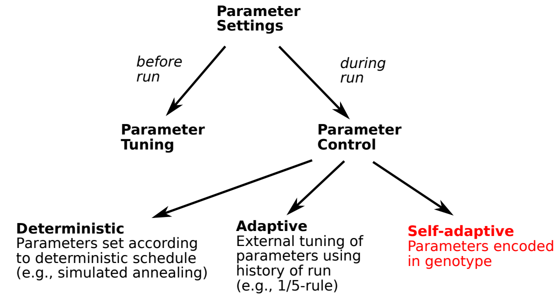
Expected runtime of EA with bit-wise mutation rate χ/n

Selection Mechanism	High Selective Pressure	Low Selective Pressure
Fitness Proportionate	$\nu > f_{\max}\ln(2e^{\chi})$	$\nu < \chi/\ln 2$ and $\lambda \geq n^3$
Linear Ranking	$\eta > e^{\chi}$	$\eta < e^{\chi}$
k-Tournament	$k > e^{\chi}$	$k < e^{\chi}$
(μ, λ)	$\lambda > \mu e^{\chi}$	$\lambda < \mu e^{\chi}$
Cellular EAs		$\Delta(G) < e^{\chi}$
OneMax	$O(n\lambda)$	$e^{\Omega(n)}$
LeadingOnes	$O(n\lambda\ln(\lambda) + n^2)$	$e^{\Omega(n)}$
Linear Functions	$O(n\lambda\ln(\lambda) + n^2)$	$e^{\Omega(n)}$
r-Unimodal	$O(r\lambda\ln(\lambda) + nr)$	$e^{\Omega(n)}$
Jump$_r$	$O(n\lambda + (n/\chi)^r)$	$e^{\Omega(n)}$

Self-adaptive Evolutionary Algorithms[11]

[11] Dang and Lehre [2016b]

Parameter settings in optimisation algorithms

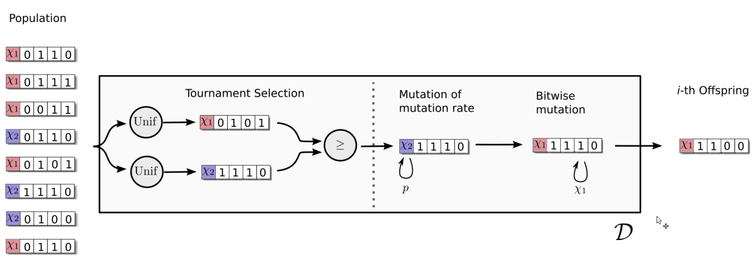


Eiben, Hinterding, and Michalewicz [1999]

Can self-adaptive EAs control parameters effectively, i.e., on a problem

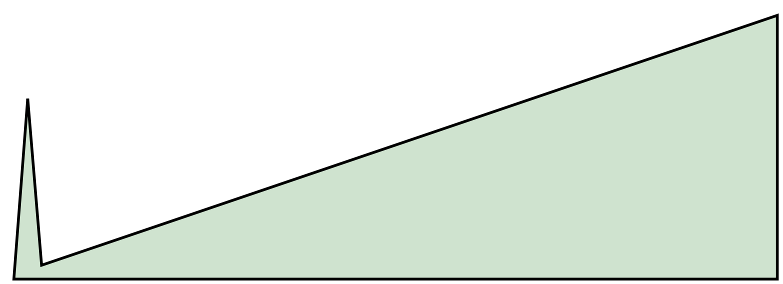
- any fixed parameter setting $\Longrightarrow$ exponential runtime
- uniform mixing of parameters $\Longrightarrow$ exponential runtime
- self-adaptation over parameters $\Longrightarrow$ polynomial runtime

Self-adaptive, non-elitist EA without crossover



- search space is $\{\chi_1, \chi_2\} \times \{0,1\}^n$, where $\chi_1, \chi_2 \in \Theta(n)$ are two mutation rates
- parent selection via binary tournament selection
- mutation rate switched with probability p, and obtain offspring by mutation with new mutation rate

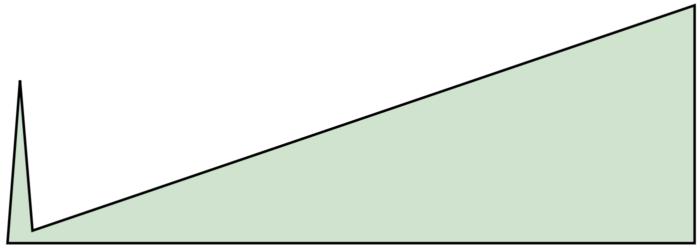
Peaked Fitness Landscape



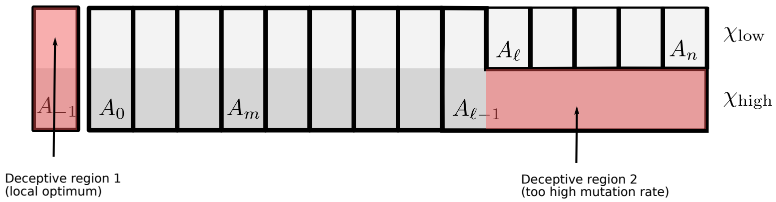
Intuition

- Low mutation rate (or elitist selection mechanism) $\Longrightarrow$ exponential time to escape local optimum
- Mutation rate above error threshold $\Longrightarrow$ exponential runtime via negative population drift (L., PPSN 2010)

Fitness function and level structure



$$\textsc{PeakedLO}(x) := \begin{cases} m & \text{if } x = 0^n \\ \sum_{i=1}^{n} \prod_{j=1}^{i} x_j & \text{otherwise.} \end{cases}$$



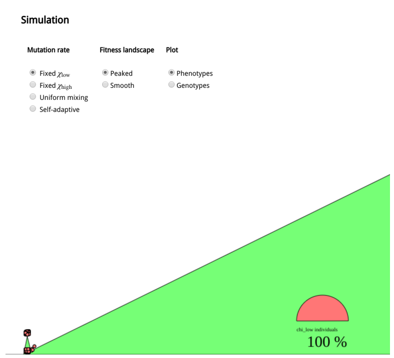
Results imply benefit of non-elitism and self-adaptation[12]

Mutation control	Runtime	Proof idea
Fixed rate χ_{low}	$e^{\Omega(n)}$	Most individuals remain on the peak. Too low selective pressure among sub-optimal individuals. (Negative drift in populations).
Fixed rate χ_{high}	$e^{\Omega(n)}$	Most individuals fall off the peak, but mutation rate is too high wrt selective pressure to reach opt. (Negative drift in populations).
Uniform mixing	$e^{\Omega(n)}$	Most individuals fall off the peak, but the effective mutation rate is too high wrt selective pressure. (Negative drift in populations).
Self-adaptation	$\mathcal{O}(n^2)$	Most individuals fall off the peak. Peak individuals do not dominate. A sub-population surviving off the peak switches to low mutation rate. (Level-based analysis).
$(\mu+\lambda)$ EA	$e^{\Omega(n)}$	Elitism prevents escape from peak.

Dang and Lehre [2016b]

[12] The results assume appropriate choices of the mutation rates χ_1 and χ_2, the strategy parameter p, and the problem parameter m.

Interactive Simulation of Results



http://www.cs.bham.ac.uk/~lehrepk/selfadapt/

Fitness proportional selection + crossover Oliveto and Witt [2014, 2015]

Definition (Simple Genetic Algorithm (SGA) (Goldberg 1989))

for $t = 0, 1, 2, \ldots$ until termination condition **do**
  for $i = 1$ to λ **do**
    Select two parents x and y from P_t proportionally to fitness
    Obtain z by applying uniform crossover to x and y with $p = 1/2$
    Flip each position in z independently with $p = 1/n$.
    Let the i-th offspring be $P_{t+1}(i) := x$.
    (i.e., create offspring by crossover followed by mutation)

Application to OneMax

Expected Behaviour

- Backward drift due to mutation close to the optimum
- no positive drift due to crossover
- selection too weak to keep positive fluctuations

Difficulties When Introducing Crossover:

- Variance of offspring distribution
- # flipping bits due to mutation Poisson-distributed → variance $O(1)$
- # of one-bits created by crossover binomially distributed according to Hamming distance of parents and $1/2$ → deviation $\Omega(\sqrt{n})$ possible

Negative Drift Theorem With Scaling

Let X_t, $t \geq 0$, random variable describing a stochastic process over finite state space $S \subseteq \mathbb{R}$;

If there ∃ interval $[a, b]$ and, possibly depending on $\ell := b - a$, bound $\epsilon(\ell) > 0$ and scaling factor $r(\ell)$ st.

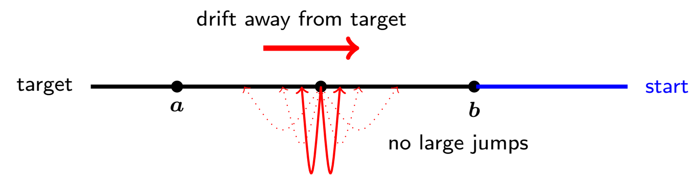
(C1) $E(X_{t+1} - X_t \mid X_0, \ldots, X_t \;\wedge\; a < X_t < b) \;\geq\; \epsilon$,

(C2) $\mathrm{Prob}(|X_{t+1} - X_t| \geq jr \mid X_0, \ldots, X_t \;\wedge\; a < X_t) \;\leq\; e^{-j}$ for $j \in \mathbb{N}_0$,

(C3) $1 \leq r \leq \min\{\epsilon^2\ell, \sqrt{\epsilon\ell/(132\log(\epsilon\ell))}\}$.

then

$$\Pr\left(T \leq e^{\epsilon\ell/(132r^2)}\right) = O(e^{-\epsilon\ell/(132r^2)}).$$



Negative Drift Theorem With Scaling

Let X_t, $t \geq 0$, random variable describing a stochastic process over finite state space $S \subseteq \mathbb{R}$;

If there ∃ interval $[a, b]$ and, possibly depending on $\ell := b - a$, bound $\epsilon(\ell) > 0$ and scaling factor $r(\ell)$ st.

(C1) $E(X_{t+1} - X_t \mid X_0, \ldots, X_t \;\wedge\; a < X_t < b) \;\geq\; \epsilon$,

(C2) $\mathrm{Prob}(|X_{t+1} - X_t| \geq jr \mid X_0, \ldots, X_t \;\wedge\; a < X_t) \;\leq\; e^{-j}$ for $j \in \mathbb{N}_0$,

(C3) $1 \leq r \leq \min\{\epsilon^2\ell, \sqrt{\epsilon\ell/(132\log(\epsilon\ell))}\}$.

then

$$\Pr\left(T \leq e^{\epsilon\ell/(132r^2)}\right) = O(e^{-\epsilon\ell/(132r^2)}).$$

Potential Function

For drift theorem, capture whole population in one value: For $X = \{x_1, \ldots, x_\mu\}$ let $g(X) := \sum_{i=1}^{\mu} e^{\kappa \textsc{OneMax}(x_i)}$.

Negative Drift Theorem With Scaling

Let X_t, $t \geq 0$, random variable describing a stochastic process over finite state space $S \subseteq \mathbb{R}$;

If there ∃ interval $[a, b]$ and, possibly depending on $\ell := b - a$, bound $\epsilon(\ell) > 0$ and scaling factor $r(\ell)$ st.

(C1) $E(X_{t+1} - X_t \mid X_0, \ldots, X_t \;\wedge\; a < X_t < b) \;\geq\; \epsilon$,

(C2) $\mathrm{Prob}(|X_{t+1} - X_t| \geq jr \mid X_0, \ldots, X_t \;\wedge\; a < X_t) \;\leq\; e^{-j}$ for $j \in \mathbb{N}_0$,

(C3) $1 \leq r \leq \min\{\epsilon^2\ell, \sqrt{\epsilon\ell/(132\log(\epsilon\ell))}\}$.

then

$$\Pr\left(T \leq e^{\epsilon\ell/(132r^2)}\right) = O(e^{-\epsilon\ell/(132r^2)}).$$

Problem: maybe $r(\ell) = \Omega(\sqrt{\ell})$

Solution

Find bits that are "converged" within population, i.e., either ones or zeros only. Crossover is irrelevant for these.

Diversity

X_t: # individuals with 1 in some fixed position at time t

Assume uniform selection (and no mutation). Then:

- The probability crossover produces an individual with 1 in the fixed position is ($X_t = k$):
- $\frac{k}{\mu} \cdot \frac{k}{\mu} + 2 \cdot \frac{1}{2} \cdot \frac{k(\mu-k)}{\mu^2} = \frac{k}{\mu}$
- $\{X_t\} \approx B(\mu, k/\mu) \rightsquigarrow E(X_t \mid X_{t-1} = k) = k$ (martingale)
- But random fluctuations $\rightsquigarrow$ absorbing state 0 or μ due to the variance ($E(T_{0 \vee \mu}) = O(\mu \log \mu)$ [drift analysis]).
- Progress by crossover is at most $n^{1/2+\epsilon}$ w.o.p. (Chernoff Bounds when ones are $n/2$).
- If $\mu \leq n^{1/2-\epsilon}$ a bit has converged to 0 before optimum is found w.o.p.

Diversity

X_t: # individuals with 1 in some fixed position at time t

Assume uniform selection (and no mutation). Then:

- The probability crossover produces an individual with 1 in the fixed position is ($X_t = k$):
- $\frac{k}{\mu} \cdot \frac{k}{\mu} + 2 \cdot \frac{1}{2} \cdot \frac{k(\mu-k)}{\mu^2} = \frac{k}{\mu}$
- $\{X_t\} \approx B(\mu, k/\mu) \rightsquigarrow E(X_t \mid X_{t-1} = k) = k$ (martingale)
- But random fluctuations $\rightsquigarrow$ absorbing state 0 or μ due to the variance

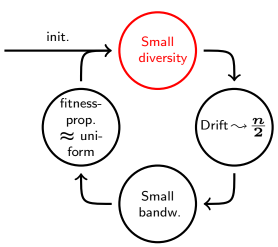
Compare fitness-prop. and uniform selection:

- Basically no difference for small population *bandwidth* (difference of best and worst OneMax-value in pop.)
- $E(X_t \mid X_{t-1} = k) = k \pm 1/(7\mu)$

Result

Let $\mu \leq n^{1/8-\epsilon}$ for an arbitrarily small constant $\epsilon > 0$. Then with probability $1 - 2^{-\Omega(n^{\epsilon/9})}$, the SGA on OneMax does not create individuals with more than $(1+c)\frac{n}{2}$ or less than $(1-c)\frac{n}{2}$ one-bits, for arbitrarily small constant $c > 0$, within the first $2^{n^{\epsilon/10}}$ generations. In particular, it does not reach the optimum then.

Overall Proof Structure



Not a loop, but in each step only exponentially small failure prob.

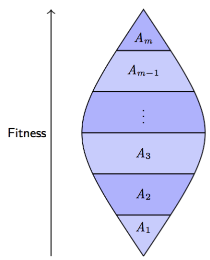
Steady-State GA

Algorithm 1: (μ+1) GA

$P \leftarrow \mu$ individuals, uniformly at random from $\{0,1\}^n$;
repeat
  Select $x, y \in P$ with replacement using an operator abiding (1);
  $z \leftarrow$ Uniform crossover with probability $1/2$ (x, y);
  Flip each bit in z with probability c/n;
  $P \leftarrow P \cup \{z\}$;
  Choose one element from P with lowest fitness and remove it from P, breaking ties at random;
until *termination condition satisfied*;

$$\forall x, y: \; f(x) \geq f(y) \implies Prob(\text{select } x) \geq Prob(\text{select } y). \quad (1)$$

Artificial Fitness Levels [DJW 2002]



Definition

A tuple $(A_1, A_2, \ldots, A_m)$ is an f-**based partition** of $f : \mathcal{X} \to \mathbb{R}$ if

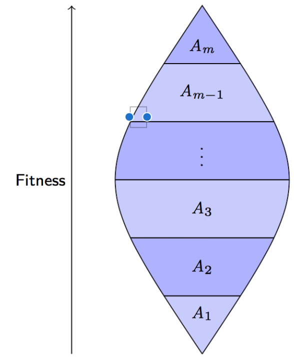
1. $A_1 \cup A_2 \cup \cdots \cup A_m = \mathcal{X}$
2. $A_i \cap A_j = \emptyset$ for $i \neq j$
3. $f(A_1) < f(A_2) < \cdots < f(A_m)$
4. $f(A_m) = \max_x f(x)$

Example

Partition of $\textsc{OneMax}$ into $n+1$ levels

$$A_j := \{x \in \{0,1\}^n \mid \textsc{OneMax}(x) = j\}$$

Artificial Fitness Levels



s_i : prob. of starting in A_i
u_i : prob. of jumping from A_i to any A_j, $i < j$.
T_i : Time to jump from A_i to any A_j, $i < j$.

Expected runtime

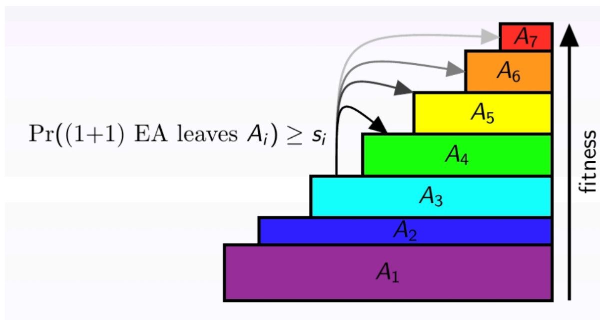
$$\begin{aligned}\mathbb{E}[T] &\leq \sum_{i=1}^{m-1} s_i \mathbb{E}\left[\sum_{j=i}^{m-1} T_j\right] \\ &= \sum_{i=1}^{m-1} s_i \sum_{j=i}^{m-1} \mathbb{E}[T_j] \\ &= \sum_{i=1}^{m-1} s_i \sum_{j=i}^{m-1} 1/u_j \leq \sum_{j=1}^{m-1} 1/u_j.\end{aligned}$$

(1+1) EA

Algorithm ((1+1)-EA)

- *Initialise P_0 with $x \in \{1,0\}^n$ by flipping each bit with $p = 1/2$;*
  Repeat
- *Create x' by flipping each bit in x with $p = 1/n$;*
- **If** $f(x') \geq f(x)$ **Then** $x' \in P_{t+1}$ **Else** $x \in P_{t+1}$;
- *Let $t = t+1$;* **Until** *stopping condition.*

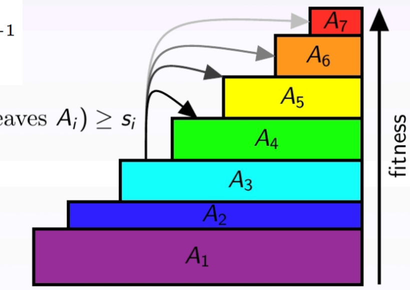
(1+1)-EA for OneMax via Artificial Fitness Levels [DJW 2002]



(1+1)-EA for OneMax via Artificial Fitness Levels [DJW 2002]

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

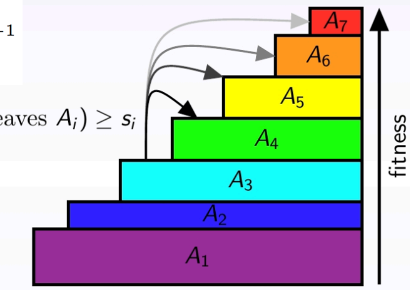
$$\Pr((1+1)\text{ EA leaves } A_i) \geq s_i$$



(1+1)-EA for OneMax via Artificial Fitness Levels [DJW 2002]

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

$$\Pr((1+1)\text{ EA leaves } A_i) \geq s_i$$



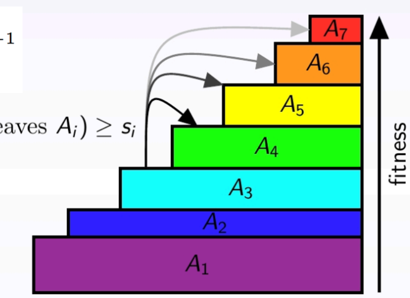
Ai : number of 0-bits;

$$s_i \geq i \cdot \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i}{en}$$

(1+1)-EA for OneMax via Artificial Fitness Levels [DJW 2002]

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

$$\Pr((1+1)\text{ EA leaves } A_i) \geq s_i$$



Ai : number of 0-bits;

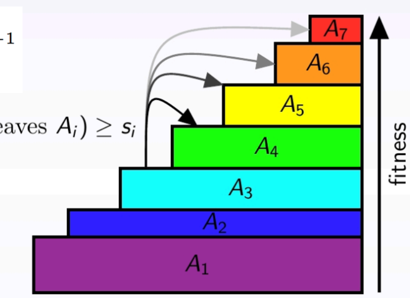
$$s_i \geq i \cdot \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i}{en}$$

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1} \leq \sum_{i=1}^{n} \frac{en}{i} \leq e \cdot n \sum_{i=1}^{m-1} \frac{1}{i} \leq e \cdot n \cdot (\ln n + 1)$$

(μ+1)-EA for OneMax via Artificial Fitness Levels

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

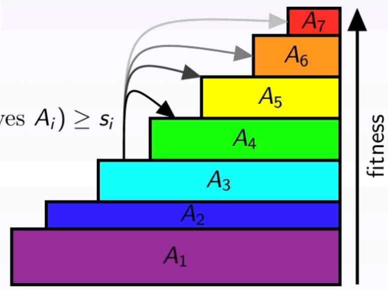
$$\Pr((1+1)\text{ EA leaves } A_i) \geq s_i$$



(μ+1)-EA for OneMax via Artificial Fitness Levels

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

$$\Pr((1{+}1)\text{ EA leaves } A_i) \geq s_i$$



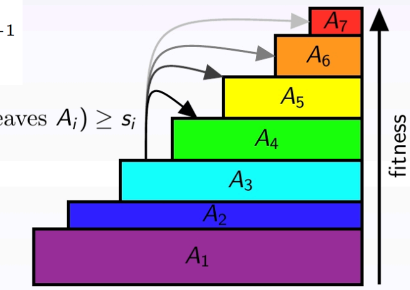
Ai : number of 0-bits;

$$s_i \geq \frac{1}{\mu} \cdot i \cdot \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i}{e\mu n}$$

(μ+1)-EA for OneMax via Artificial Fitness Levels

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1}$$

$$\Pr((1{+}1)\text{ EA leaves } A_i) \geq s_i$$



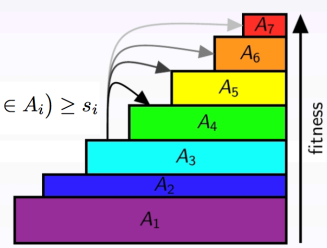
Ai : number of 0-bits;

$$s_i \geq \frac{1}{\mu} \cdot i \cdot \frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{i}{e\mu n}$$

$$E(T) \leq \sum_{i=1}^{m-1} s_i^{-1} \leq \sum_{i=1}^{n} \frac{e\mu n}{i} \leq e\mu n \sum_{i=1}^{n} \leq e\mu n(\ln + 1) = O(\mu n \ln n)$$

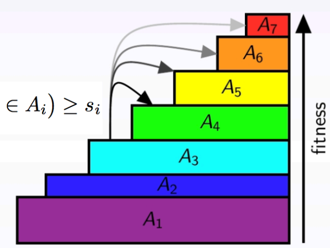
Artificial Fitness Levels for Steady state populations [Witt 2006]

$$Pr((\mu+1)\ EA \text{ leaves } A_i \mid \chi(i) \in A_i) \geq s_i$$



Artificial Fitness Levels for Steady state populations [Witt 2006]

$$Pr((\mu+1)\ EA \text{ leaves } A_i \mid \chi(i) \in A_i) \geq s_i$$



Let:

- T_o be the expected time for a fraction $\chi(i)$ of the population to be in level A_i
- s_i be the probability to leave level A_i for A_j with $j > i$ given $\chi(i)$ in level A_i
- **Then:**

$$E(T) \leq \sum_{i=1}^{m-1} \left(\frac{1}{s_i} + T_o\right)$$

Theorem

The expected runtime of the $(\mu{+}1)$-EA for OneMax *is* $O(\mu n + n\log n)$.

Theorem

The expected runtime of the $(\mu{+}1)$-EA for OneMax *is* $O(\mu n + n\log n)$.

Given that there are i copies,
the probability of creating a new copy is:

$$Pr(\text{copy} \mid i \in A_j) \geq \frac{i}{\mu}\left(1-\frac{1}{n}\right)^n \geq \frac{i}{2e\mu}$$

Theorem

The expected runtime of the $(\mu{+}1)$-EA for OneMax *is* $O(\mu n + n\log n)$.

Given that there are i copies,
the probability of creating a new copy is:

$$Pr(\text{copy} \mid i \in A_j) \geq \frac{i}{\mu}\left(1-\frac{1}{n}\right)^n \geq \frac{i}{2e\mu}$$

The expected time to create
at most $\min\{n/j, \mu\}$ copies is:

$$T_0 \leq \sum_{i=1}^{n/j} \frac{2e\mu}{i} \leq 2e\mu(\ln(n/j))$$

Theorem

The expected runtime of the $(\mu{+}1)$-EA for OneMax *is* $O(\mu n + n\log n)$.

Given that there are i copies,
the probability of creating a new copy is:

$$Pr(\text{copy} \mid i \in A_j) \geq \frac{i}{\mu}\left(1-\frac{1}{n}\right)^n \geq \frac{i}{2e\mu}$$

The expected time to create
at most $\min\{n/j, \mu\}$ copies is:

$$T_0 \leq \sum_{i=1}^{n/j} \frac{2e\mu}{i} \leq 2e\mu(\ln(n/j))$$

And,

$$E\left(\sum_{j=1}^{n} T_0\right) \leq \sum_{j=1}^{n-1} 2e\mu \ln\left(\frac{n}{j}\right) \leq 2e\mu \sum_{j=1}^{n-1} \ln\left(\frac{n}{j}\right) \leq 4e\mu n$$

Theorem

The expected runtime of the (μ+1)-EA for ONEMAX *is* $O(\mu n + n \log n)$.

Given that there are i copies, the probability of creating a new copy is:

$$Pr(\text{copy} \mid i \in A_j) \geq \frac{i}{\mu}\left(1 - \frac{1}{n}\right)^n \geq \frac{i}{2e\mu}$$

The expected time to create at most $\min\{n/j, \mu\}$ copies is:

$$T_0 \leq \sum_{i=1}^{n/j} \frac{2e\mu}{i} \leq 2e\mu(\ln(n/j))$$

And,

$$E\left(\sum_{j=1}^{n} T_0\right) \leq \sum_{j=1}^{n-1} 2e\mu \ln\left(\frac{n}{j}\right) \leq 2e\mu \sum_{j=1}^{n-1} \ln\left(\frac{n}{j}\right) \leq 4e\mu n$$

Overall,

$$E(T) \leq \sum_{j=1}^{n}\left(\frac{1}{s_j} + T_0\right) \leq \sum_{j=1}^{n} \frac{1}{s_j} + \sum_{j=1}^{n} T_0$$

$$\leq \sum_{j=1}^{n} \frac{\mu}{\min\{n/j, \mu\}} \cdot \frac{en}{j} + 4e\mu n \leq en \ln n + e\mu n + 4e\mu n$$

$$= O(n \ln n + \mu n)$$

(μ+1) GA for OneMax [Corus,Oliveto,TEVC 2017]

Theorem

The expected runtime of the (μ+1) GA with $\mu \geq 3$ and mutation rate c/n for any constant c on ONEMAX *is:*

$$E[T] \leq \frac{3e^c n \log n}{c(3+c)} + \mathcal{O}(n\mu \log \mu).$$

For $\mu = o(\log n / \log\log n)$, the bound reduces to:

$$E[T] \leq \frac{3}{c(3+c)} e^c n \log n\,(1 + o(1)).$$

(μ+1) GA for OneMax [Corus,Oliveto,TEVC 2017]

Theorem

The expected runtime of the (μ+1) GA with $\mu \geq 3$ and mutation rate c/n for any constant c on ONEMAX *is:*

$$E[T] \leq \frac{3e^c n \log n}{c(3+c)} + \mathcal{O}(n\mu \log \mu).$$

For $\mu = o(\log n / \log\log n)$, the bound reduces to:

$$E[T] \leq \frac{3}{c(3+c)} e^c n \log n\,(1 + o(1)).$$

- Mutation rate 1/n: 3/4 e n log n (1 + o(1)) (GA) versus e n log n (1+ o(1)) (no crossover); [Sudholt,TEVC 2013]

(μ+1) GA for OneMax [Corus,Oliveto,TEVC 2017]

Theorem

The expected runtime of the (μ+1) GA with $\mu \geq 3$ and mutation rate c/n for any constant c on ONEMAX *is:*

$$E[T] \leq \frac{3e^c n \log n}{c(3+c)} + \mathcal{O}(n\mu \log \mu).$$

For $\mu = o(\log n / \log\log n)$, the bound reduces to:

$$E[T] \leq \frac{3}{c(3+c)} e^c n \log n\,(1 + o(1)).$$

- Mutation rate 1/n: 3/4 e n log n (1 + o(1)) (GA) versus e n log n (1+ o(1)) (no crossover); [Sudholt,TEVC 2013]

- Higher mutation rates may be beneficial: 0.72 e n log n (1 + o(1)) ~1.3/n (GA);

(μ+1) GA for OneMax [Corus,Oliveto,TEVC 2017]

Theorem

The expected runtime of the (μ+1) GA with $\mu \geq 3$ and mutation rate c/n for any constant c on ONEMAX *is:*

$$E[T] \leq \frac{3e^c n \log n}{c(3+c)} + \mathcal{O}(n\mu \log \mu).$$

For $\mu = o(\log n / \log\log n)$, the bound reduces to:

$$E[T] \leq \frac{3}{c(3+c)} e^c n \log n (1 + o(1)).$$

- Mutation rate 1/n: 3/4 e n log n (1 + o(1)) (GA) versus e n log n (1+ o(1)) (no crossover); [Sudholt, TEVC 2013]
- Higher mutation rates may be beneficial: 0.72 e n log n (1 + o(1)) ~1.3/n (GA);
- Populations may be beneficial: for μ=2 the bound is 4/5 e n log n (1 + o(1)).

Proof Idea [Corus,Oliveto,TEVC 2017]

1) We divide the search space in canonical fitness levels $L_i = \{x \in \{0,1\}^n | \text{ONEMAX}(x) = i\}$;

2) Each level *i* is represented by a Markov Chain (and all individuals are at least in *Li*) ;

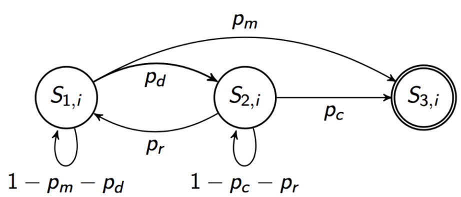
3) The runtime is upper bounded by the time it takes to discover the next level E[*Li*] + the time it takes for the entire population to take over the level (E[$T_{takeover}$] = O(μ log μ)) .

Proof Idea [Corus,Oliveto,TEVC 2017]

1) We divide the search space in canonical fitness levels $L_i = \{x \in \{0,1\}^n | \text{ONEMAX}(x) = i\}$;

2) Each level *i* is represented by a Markov Chain (and all individuals are at least in *Li*) ;

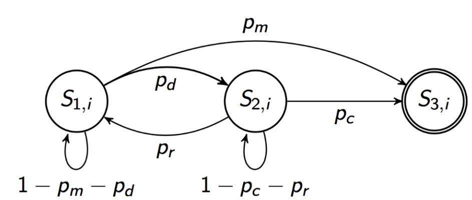
3) The runtime is upper bounded by the time it takes to discover the next level E[*Li*] + the time it takes for the entire population to take over the level (E[$T_{takeover}$] = O(μ log μ)) .



S_1: no diversity;
S_2: at least one diverse individual;
S_3: at least one individual in *Li+1* or higher.

$$E[T] \leq \sum_{i=0}^{n-1} \left(E[L_i] + E[T_{takeover}] \right)$$

Proof Idea (2) [Corus,Oliveto,TEVC 2017]



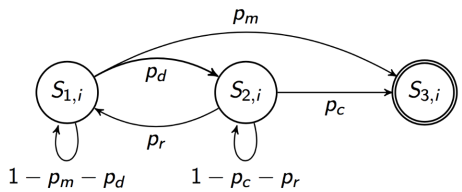
Absorbing time from State S_1:

$$E[T_1] = \frac{p_c + p_r + p_d}{p_c p_d + p_c p_m + p_m p_r} \leq \frac{p_c + p_r}{p_c p_d + p_c p_m + p_m p_r} + \frac{1}{p_c}$$

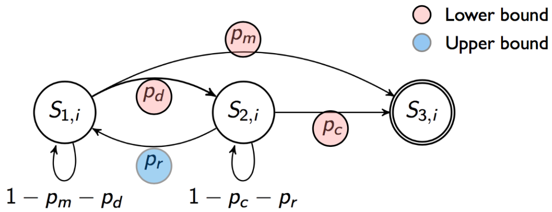
But the exact transition probabilities are tedious to calculate!

Proof Idea (3) [Corus,Oliveto,TEVC 2017]

M1:

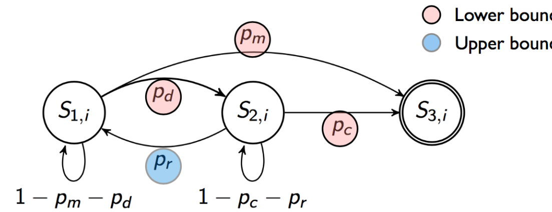


M2:



As long as $p_m < p_c$ then $E[T_{M_1}] \leq E[T_{M_2}]$

Proof Idea (4) [Corus,Oliveto,TEVC 2017]



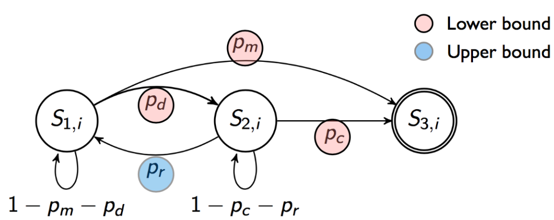
Lemma

The transition probabilities p_d, p_c, p_r and p_m are bounded as follows:

$$p_d \geq \frac{\mu}{(\mu+1)} \frac{i(n-i)c^2}{n^2(e^c + \mathcal{O}(1/n))} \qquad p_c \geq \frac{\mu - 1}{2\mu^2(e^c + \mathcal{O}(1/n))}$$

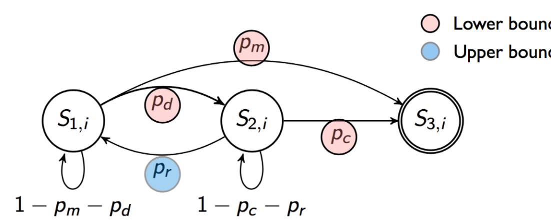
$$p_r \leq \frac{(\mu-1)(2\mu - 1 + \mathcal{O}(1/n))}{2e^c\mu^2(\mu+1)} \qquad p_m \geq \frac{c(n-i)}{n(e^c + \mathcal{O}(1/n))}.$$

Proof Idea (5) [Corus,Oliveto,TEVC 2017]



$$E[T_i] \leq \frac{p_c + p_r}{p_c p_d + p_c p_m + p_m p_r} + \frac{1}{p_c} \leq \frac{e^c n}{c(n-i)} \frac{3}{3+c} + \mathcal{O}(\mu)$$

Proof Idea (5) [Corus,Oliveto,TEVC 2017]

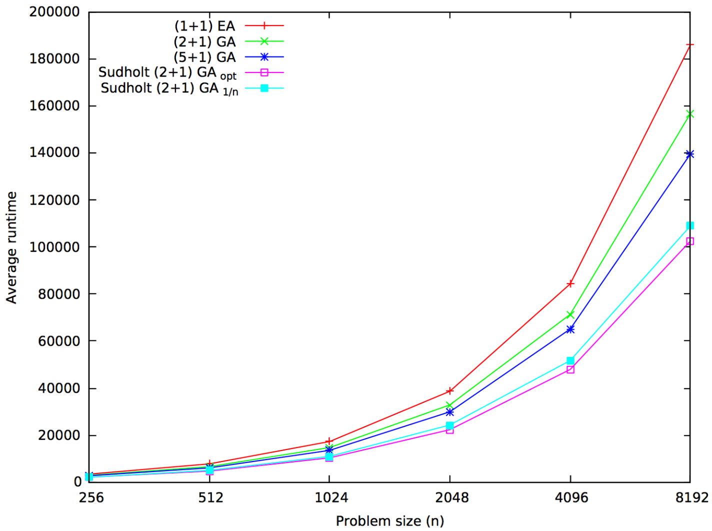


$$E[T_i] \leq \frac{p_c + p_r}{p_c p_d + p_c p_m + p_m p_r} + \frac{1}{p_c} \leq \frac{e^c n}{c(n-i)} \frac{3}{3+c} + \mathcal{O}(\mu)$$

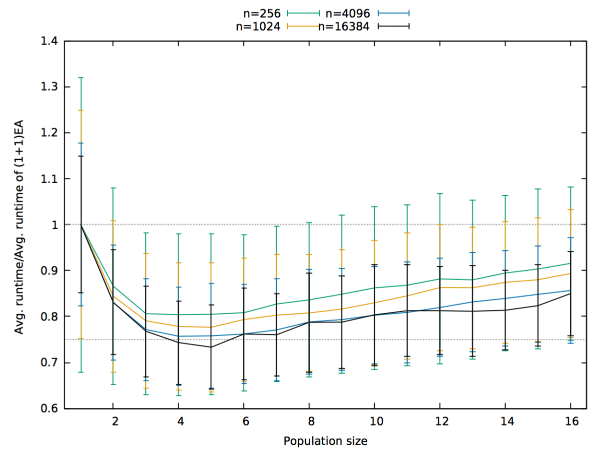
$$E[T] \leq \sum_{i=n-n/(4c(1+e^c))}^{n} \left(\frac{e^c n}{c(n-i)} \frac{3}{3+c} + \mathcal{O}(\mu) + \mathcal{O}(\mu \log \mu) \right)$$

$$\leq \frac{3e^c n \log n}{c(3+c)} + \mathcal{O}(n\mu \log \mu)$$

Experimental Comparisons [Corus,Oliveto,TEVC 2017]

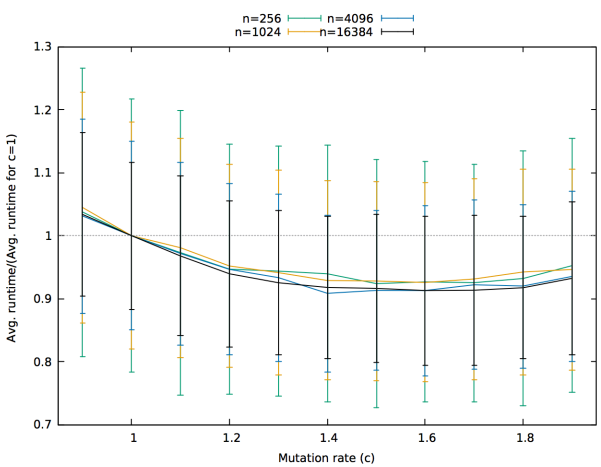


Population Size [Corus,Oliveto,TEVC 2017]



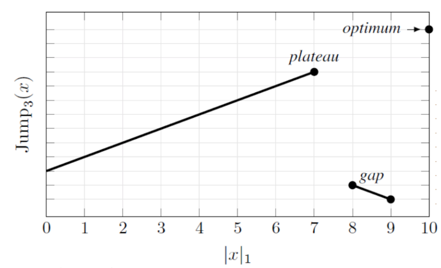
Is the best population size constant (eg., 5) or does it grow with the problem size?

Mutation Rate [Corus,Oliveto,TEVC 2017]



We got the best upper bound on the runtime with 1.3 - the best rate seems to be larger!

GAs Can Escape More Quickly [DFKKLoSS, IEEE TEVC 2018]



Expected Runtimes ($k > 2$)

- (μ+1) EA with $p_m = 1/n$: $\Theta(n^k)$ (i.e., no crossover);
- (μ+1) GA with $p_m = 1/n$: $O(n^{k-1} \log n)$ [$\mu = \Theta(n)$];
- (μ+1) GA with $p_m = (1+\delta)/n$ is $O(n^{k-1})$ [$\mu = \Theta(\log n)$].

The interplay between mutation and crossover can create diversity on top of the plateau; then crossover + mutation can jump more quickly!

Conclusions

Steady state GAs are faster than any standard bit mutation-only evolutionary algorithm!

Evidence that evolving populations via sexual recombination is beneficial!
What is the best population size?

Evidence that higher mutation rates than standard are beneficial;
What is the best mutation rate?

Steady state GAs escape local optima of the Jump benchmark function a linear time faster than mutation-only EAs
Larger problem classes?

What about $(\mu+\lambda)$ GAs and generational GAs?

Summary

- Runtime analysis of evolutionary algorithms
  - mathematically rigorous statements about EA performance
  - most previous results on simple EAs, such as (1+1) EA
  - special techniques developed for population-based EAs
- Drift Analysis
- Level-based method [Corus et al., 2014]
  - EAs analysed from the perspective of EDAs
  - Upper bounds on expected optimisation time
  - Example applications include crossover, noise, and self-adaptation
- Negative drift theorem [Lehre, 2011a]
  - reproductive rate vs selective pressure
  - exponential lower bounds
  - mutation-selection balance
- Diversity + Bandwidth analysis for fitness proportional selection [Oliveto and Witt, 2014, 2015]
  - analysis of crossover
  - low selection pressure
  - exponential lower bounds
- Speed-up via crossover for steady state GAs to hillclimb OneMax and escape local optima [Dang, Friedrich, Kötzing, Krejca, Lehre, Oliveto, Sudholt, and Sutton, 2017, Corus and Oliveto, 2017]

Other Theory-related Tutorials and Workshops at GECCO 2018

- Theory for Non-Theoreticians (B. Doerr, Today 14:50, Conference Room 2 (3F))
- Theory of Estimation-of- Distribution Algorithms (C. Witt, Tomorrow 9:00, Conference Room 2 (3F))
- BB-DOB - Black Box Discrete Optimization Benchmarking Workshop (COST Action IMAPPNIO, Tomorrow, 9:00 - 12:40, Training Room 3 (2F))
- Adaptive Parameter Choices in Evolutionary Computation (C. Doerr, Tomorrow 14:00, Conference Room Medium (2F))

Acknowledgements

- Dogan Corus, University of Sheffield, UK
- Duc-Cuong Dang, independent researcher
- Anton Eremeev, Omsk Branch of Sobolev Institute of Mathematics, Russia
- Carsten Witt, DTU, Lyngby, Denmark

The research leading to these results has received funding from the EPSRC under grant no EP/M004252/1.

References I

Tianshi Chen, Jun He, Guangzhong Sun, Guoliang Chen, and Xin Yao. A new approach for analyzing average time complexity of population-based evolutionary algorithms on unimodal problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(5):1092–1106, Oct. 2009. ISSN 1083-4419. doi: 10.1109/TSMCB.2008.2012167.

Dogan Corus and Pietro S. Oliveto. Standard steady state genetic algorithms can hillclimb faster than mutation-only evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2017. doi: 10.1109/TEVC.2017.2745715.

Dogan Corus, Duc-Cuong Dang, Anton V. Eremeev, and Per Kristian Lehre. Level-based analysis of genetic algorithms and other search processes. In *Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Ljubljana, Slovenia, September 13-17, 2014. Proceedings*, pages 912–921, 2014. doi: 10.1007/978-3-319-10762-2_90. URL http://dx.doi.org/10.1007/978-3-319-10762-2_90.

Dogan Corus, Duc-Cuong Dang, Anton V. Eremeev, and Per Kristian Lehre. Level-based analysis of genetic algorithms and other search processes. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2017. ISSN 1089-778X. doi: 10.1109/TEVC.2017.2753538.

Duc-Cuong Dang and Per Kristian Lehre. Refined Upper Bounds on the Expected Runtime of Non-elitist Populations from Fitness-Levels. In *Proceedings of the 16th Annual Conference on Genetic and Evolutionary Computation Conference (GECCO 2014)*, pages 1367–1374, 2014. ISBN 9781450326629. doi: 10.1145/2576768.2598374.

Duc-Cuong Dang and Per Kristian Lehre. Efficient optimisation of noisy fitness functions with population-based evolutionary algorithms. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*, FOGA '15, pages 62–68, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3434-1. doi: 10.1145/2725494.2725508. URL http://doi.acm.org/10.1145/2725494.2725508.

Duc-Cuong Dang and Per Kristian Lehre. Runtime analysis of non-elitist populations: From classical optimisation to partial information. *Algorithmica*, 75(3):428–461, 2016a. doi: 10.1007/s00453-015-0103-x. URL https://doi.org/10.1007/s00453-015-0103-x.

Duc-Cuong Dang and Per Kristian Lehre. Self-adaptation of Mutation Rates in Non-elitist Populations. In *Parallel Problem Solving from Nature - PPSN XIV*, Lecture Notes in Computer Science, pages 803–813. Springer, Cham, September 2016b. ISBN 978-3-319-45822-9 978-3-319-45823-6. doi: 10.1007/978-3-319-45823-6_75. URL https://link.springer.com/chapter/10.1007/978-3-319-45823-6_75.

References II

Duc-Cuong Dang, Tobias Friedrich, Timo Kötzing, Martin S. Krejca, Per Kristian Lehre, Pietro Simone Oliveto, Dirk Sudholt, and Andrew M. Sutton. Escaping local optima using crossover with emergent diversity. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2017. ISSN 1089-778X. doi: 10.1109/TEVC.2017.2724201.

Benjamin Doerr. Better Runtime Guarantees Via Stochastic Domination. *arXiv:1801.04487 [cs]*, January 2018. URL http://arxiv.org/abs/1801.04487. arXiv: 1801.04487.

Benjamin Doerr and Leslie Ann Goldberg. Drift analysis with tail bounds. In *Proceedings of the 11th international conference on Parallel problem solving from nature: Part I*, PPSN'10, pages 174–183, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-15843-9, 978-3-642-15843-8.

Benjamin Doerr, Daniel Johannsen, and Carola Winzen. Multiplicative drift analysis. In *GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 1449–1456, New York, NY, USA, 2010. ACM.

Stefan Droste, Thomas Jansen, and Ingo Wegener. On the analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science*, 276:51–81, 2002.

Stefan Droste, Thomas Jansen, and Ingo Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of Computing Systems*, 39(4):525–544, 2006.

A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, July 1999. ISSN 1089-778X. doi: 10.1109/4235.771166.

Agoston E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. SpringerVerlag, 2003. ISBN 3540401849.

Simon Fischer, Lars Olbrich, and Berthold Vöcking. Approximating wardrop equilibria with finitely many agents. *Distributed Computing*, 21(2):129–139, 2008.

Bruce Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 14(3):502–525, 1982.

Jun He and Xin Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 127(1):57–85, March 2001.

Jun He and Xin Yao. Towards an analytic framework for analysing the computation time of evolutionary algorithms. *Artificial Intelligence*, 145(1-2):59–97, 2003.

References III

Jens Jägersküpper. Algorithmic analysis of a basic evolutionary algorithm for continuous optimization. *Theoretical Computer Science*, 379(3):329–347, 2007. ISSN 0304-3975. doi: http://dx.doi.org/10.1016/j.tcs.2007.02.042.

Jens Jägersküpper. A Blend of Markov-Chain and Drift Analysis. In *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN 2008)*, 2008.

Thomas Jansen, Kenneth A. De Jong, and Ingo Wegener. On the choice of the offspring population size in evolutionary algorithms. *Evolutionary Computation*, 13(4):413–440, 2005. doi: 10.1162/106365605774666921.

Daniel Johannsen. *Random combinatorial structures and randomized search heuristics*. PhD thesis, Universität des Saarlandes, 2010.

Per Kristian Lehre. Negative drift in populations. In *Proceedings of Parallel Problem Solving from Nature - (PPSN XI)*, volume 6238 of *LNCS*, pages 244–253. Springer Berlin / Heidelberg, 2011a.

Per Kristian Lehre. Fitness-levels for non-elitist populations. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, (GECCO 2011), pages 2075–2082, New York, NY, USA, 2011b. ACM. ISBN 978-1-4503-0557-0.

Per Kristian Lehre and Carsten Witt. Black-box search by unbiased variation. *Algorithmica*, pages 1–20, 2012. ISSN 0178-4617.

Per Kristian Lehre and Carsten Witt. Concentrated Hitting Times of Randomized Search Heuristics with Variable Drift. In *Algorithms and Computation*, number 8889 in Lecture Notes in Computer Science, pages 686–697. Springer International Publishing, December 2014. ISBN 978-3-319-13074-3 978-3-319-13075-0. URL http://arxiv.org/abs/1307.2559. Best paper award.

Per Kristian Lehre and Xin Yao. On the impact of mutation-selection balance on the runtime of evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 16(2):225 –241, April 2012. ISSN 1089-778X. doi: 10.1109/TEVC.2011.2112665.

Frank Neumann, Pietro Simone Oliveto, and Carsten Witt. Theoretical analysis of fitness-proportional selection: landscapes and efficiency. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation (GECCO 2009)*, pages 835–842, New York, NY, USA, 2009a. ACM. ISBN 978-1-60558-325-9. doi: http://doi.acm.org/10.1145/1569901.1570016.

References IV

Frank Neumann, Dirk Sudholt, and Carsten Witt. Analysis of different mmas aco algorithms on unimodal functions and plateaus. *Swarm Intelligence*, 3(1):35–68, 2009b.

Pietro Oliveto and Carsten Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. *Algorithmica*, pages 1–18, 2010. URL http://dx.doi.org/10.1007/s00453-010-9387-z. 10.1007/s00453-010-9387-z.

Pietro S. Oliveto and Carsten Witt. On the runtime analysis of the simple genetic algorithm. *Theoretical Computer Science*, 545:2–19, 2014.

Pietro S. Oliveto and Carsten Witt. Improved time complexity analysis of the simple genetic algorithm. *Theoretical Computer Science*, 605:21–41, 2015.

Jonathan E. Rowe and Dirk Sudholt. The choice of the offspring population size in the $(1,\lambda)$ ea. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, GECCO '12, pages 1349–1356, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1177-9.

Dirk sudholt. A new method for lower bounds on the running time of evolu- tionary algorithms. *IEEE Transactions on Evolutionary Computation*, 17(3):418–435, 2013. doi: 10.1109/TEVC.2017.2745715.

Carsten Witt. Runtime Analysis of the $(\mu + 1)$ EA on Simple Pseudo-Boolean Functions. *Evolutionary Computation*, 14(1):65–86, 2006.

Christine Zarges. On the utility of the population size for inversely fitness proportional mutation rates. In *FOGA 09: Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms*, pages 39–46, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-414-0. doi: http://doi.acm.org/10.1145/1527125.1527132.