How to Treat Evolutionary Algorithms as Ordinary Randomized Algorithms

Carsten Witt

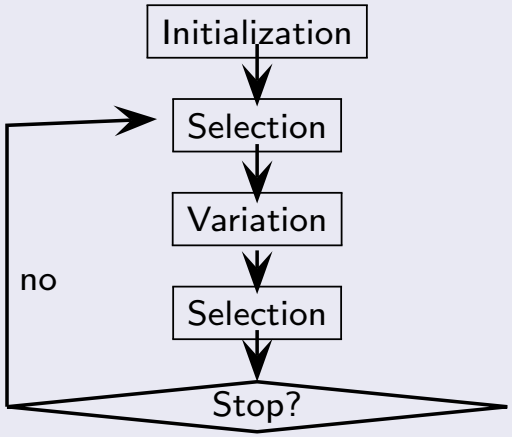
Technical University of Denmark

(Workshop "Computational Approaches to Evolution", March 19, 2014)

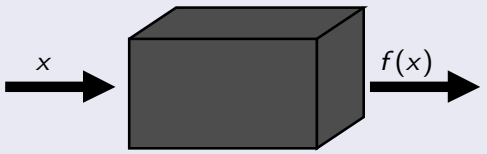
Context

Running time/complexity analysis of randomized search heuristics (RSHs)

- e. g., evolutionary algorithms (EA), local search, simulated annealing (SA), ant colony optimization, particle swarm optimization



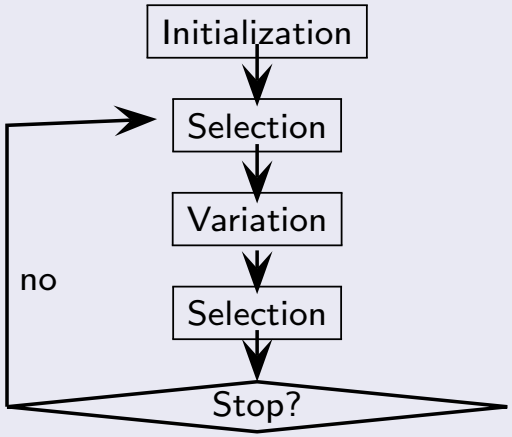
Purpose: find optimum of objective/fitness function $f : D \to \mathbb{R}$



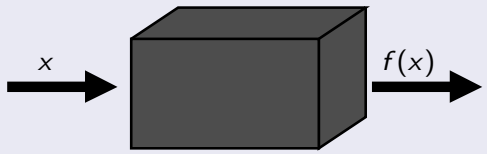
Context

Running time/complexity analysis of randomized search heuristics (RSHs)

- e. g., evolutionary algorithms (EA), local search, simulated annealing (SA), ant colony optimization, particle swarm optimization



Purpose: find optimum of objective/fitness function $f : D \to \mathbb{R}$



- study the (expected) optimization time
- started in late 1980s/early 1990s with analysis of SA
- using/adapting techniques from classical randomized algorithms
- nowadays body of results and methods for the analysis available, especially in combinatorial optimization

Contents

Challenge

Very innocently looking settings surprisingly hard to analyze
since EAs (and other RSHs) were not made for analysis

Aim for today

Show selected (state-of-the-art) results and techniques;
discuss challenges and opportunies of approach

1. Introducing an important technique: drift analysis
2. An application to "weighted hillclimbing" with EAs: linear functions
3. Back to classical TCS?

Drift Analysis: Idea

- EAs **are** randomized algorithms $\rightarrow$ analyze their "running time" $=$ time to reach optimum (or some approximation).

Drift Analysis: Idea

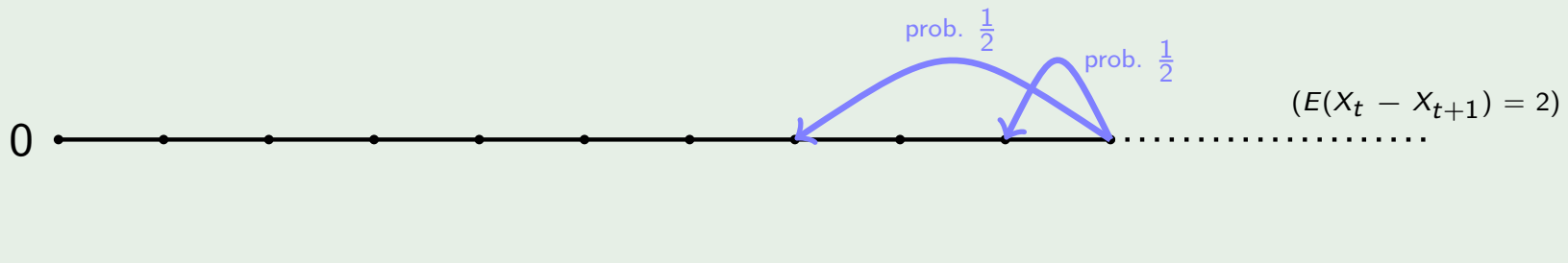
- EAs **are** randomized algorithms $\rightarrow$ analyze their "running time" $=$ time to reach optimum (or some approximation).
- EAs induce very complex stochastic processes.
- In addition, want to analyze a complex random variable related to a global property (a first hitting time).
- Can rarely understand the global behavior completely.

Drift Analysis: Idea

- EAs **are** randomized algorithms $\rightarrow$ analyze their "running time" $=$ time to reach optimum (or some approximation).
- EAs induce very complex stochastic processes.
- In addition, want to analyze a complex random variable related to a global property (a first hitting time).
- Can rarely understand the global behavior completely.
- But can roughly understand what happens from one step to the next – a local property.
- Drift analysis is the tool to translate the one-step local behavior into a global statement on the running time.

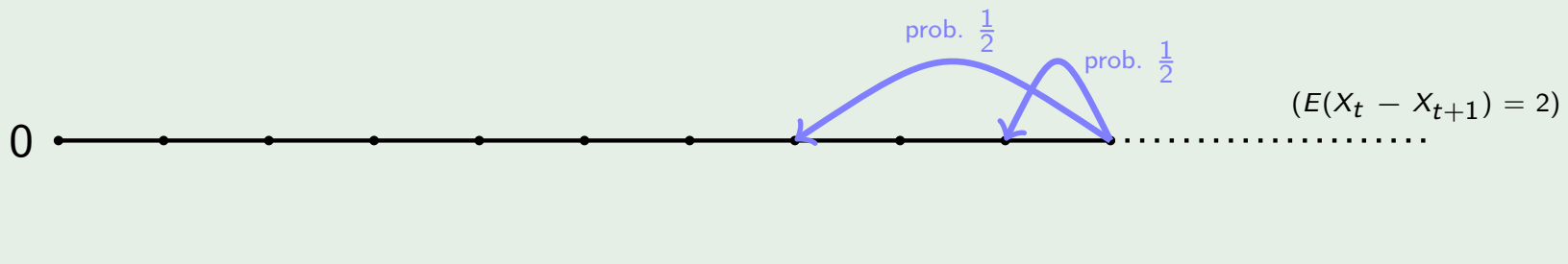
A Simple Scenario: Additive Drift

- You start off at distance $X_0 = n$ (discrete r.v.) from optimum.
- In each step, reduce distance by at least d *in expectation*: $E(X_t - X_{t+1} \mid X_t > 0) \geq d$ – maybe sometimes more or less.
- Find (or bound) the expected time to reach 0: $E(T_0)$?



A Simple Scenario: Additive Drift

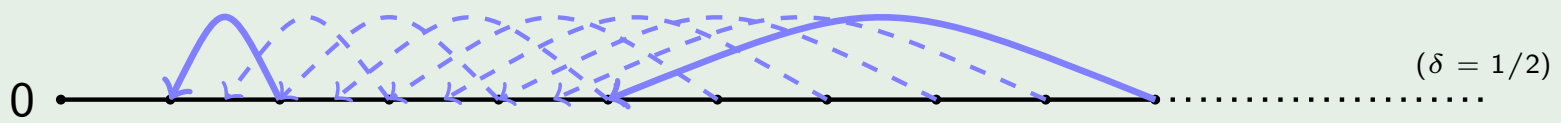
- You start off at distance $X_0 = n$ (discrete r.v.) from optimum.
- In each step, reduce distance by at least d *in expectation*: $E(X_t - X_{t+1} \mid X_t > 0) \geq d$ – maybe sometimes more or less.
- Find (or bound) the expected time to reach 0: $E(T_0)$?



- The obvious answer is correct: $E(T_0) \leq \frac{n}{d}$.
- Analogously $E(X_t - X_{t+1} \mid X_t > 0) \leq d \Rightarrow E(T_0) \geq \frac{n}{d}$.
- Scenario is called additive drift. Brought into theory of EAs by He/Yao (2001).

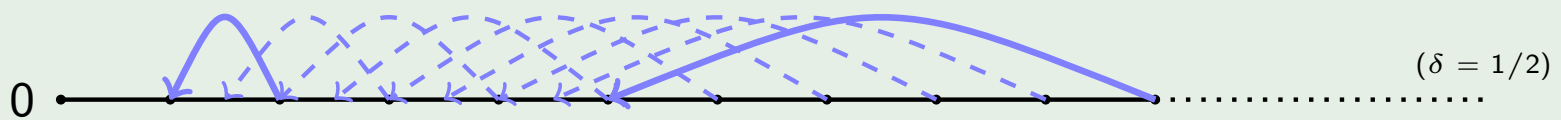
Multiplicative Drift

- Still $X_0 = n$. In each step, reduce distance by expected δ-factor of current one: $E(X_t - X_{t+1} \mid X_t = s) \geq \delta s$. Find $E(T_0)$.
- Example: $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. Your guess for $E(T_0)$?



Multiplicative Drift

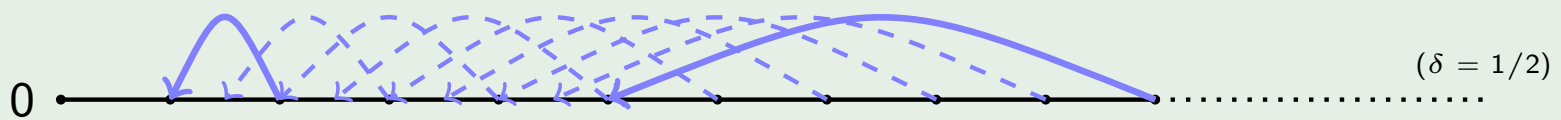
- Still $X_0 = n$. In each step, reduce distance by expected δ-factor of current one: $E(X_t - X_{t+1} \mid X_t = s) \geq \delta s$. Find $E(T_0)$.
- Example: $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. Your guess for $E(T_0)$?



- $E(T_0) \leq 2(\ln n + 1)$, in general $E(T_0) \leq \frac{(\ln X_0)+1}{\delta}$.
- Lower bound?

Multiplicative Drift

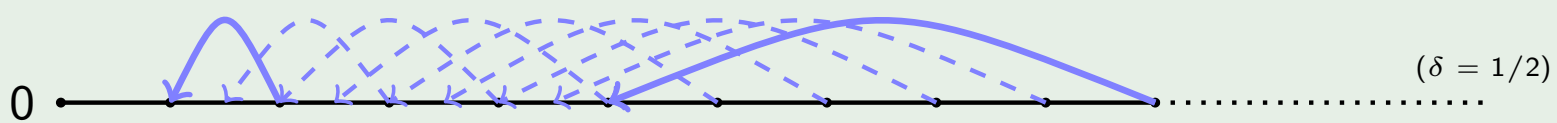
- Still $X_0 = n$. In each step, reduce distance by expected δ-factor of current one: $E(X_t - X_{t+1} \mid X_t = s) \geq \delta s$. Find $E(T_0)$.
- Example: $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. Your guess for $E(T_0)$?



- $E(T_0) \leq 2(\ln n + 1)$, in general $E(T_0) \leq \frac{(\ln X_0)+1}{\delta}$.
- Lower bound? Say $\Pr(X_{t+1} = 0) = \Pr(X_{t+1} = X_t) = \frac{1}{2}$. Then still $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. However, $E(T_0) = 2$.
- Matching lower bound ($E(T_0) \geq \frac{\ln X_0}{\delta}$) requires concentration of the one-step progress – often met in EAs.

Multiplicative Drift

- Still $X_0 = n$. In each step, reduce distance by expected δ-factor of current one: $E(X_t - X_{t+1} \mid X_t = s) \geq \delta s$. Find $E(T_0)$.
- Example: $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. Your guess for $E(T_0)$?



- $E(T_0) \leq 2(\ln n + 1)$, in general $E(T_0) \leq \frac{(\ln X_0)+1}{\delta}$.
- Lower bound? Say $\Pr(X_{t+1} = 0) = \Pr(X_{t+1} = X_t) = \frac{1}{2}$. Then still $E(X_{t+1} \mid X_t = s) = \frac{s}{2}$. However, $E(T_0) = 2$.
- Matching lower bound ($E(T_0) \geq \frac{\ln X_0}{\delta}$) requires concentration of the one-step progress – often met in EAs.
- Scenario is called multiplicative drift – more typical than additive drift in analysis of EAs.
- Due to Doerr/Johannsen/Winzen and Doerr/Goldberg (2010).

A Warm-up Example Using Multiplicative Drift

An extremely simple EA: (1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x independently with probability $\frac{1}{n}$.
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

A Warm-up Example Using Multiplicative Drift

An extremely simple EA: (1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x independently with probability $\frac{1}{n}$.
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

An extremely simple problem

$\textsc{OneMax}(x_1, \ldots, x_n)$ (in fact: "OneMin") $= x_1 + \cdots + x_n$

A Warm-up Example Using Multiplicative Drift

An extremely simple EA: (1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x independently with probability $\frac{1}{n}$.
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

An extremely simple problem

$\textsc{OneMax}(x_1, \ldots, x_n)$ (in fact: "OneMin") $= x_1 + \cdots + x_n$

Analysis:

- Say $X_t = \textsc{OneMax}(x) = s$, where $s \in \{1, \ldots, n\}$.
- s ways of flipping a single one-bit, each prob. $\frac{1}{n}\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{1}{en}$.
- $E(X_t - X_{t+1} \mid X_t = s) \geq \frac{s}{en} \Rightarrow \delta = \frac{1}{en}$ in multiplicative drift theorem
- $E(T_0) \leq \frac{\ln(X_0)+1}{\delta} \leq \frac{\ln(n)+1}{1/(en)} = en \ln n + en = O(n \log n)$.

Further Applications of Drift Analysis

- Bounds on the running time of population-based EAs, e. g.

The Simple GA (SGA)

1. Create population P of μ randomly chosen individuals.
2. $C := \emptyset$.
3. While $|C| < \mu$ do
   - **Fitness-proportional selection:** Select two parents x' and x'' from P proportional to their fitness.
   - **Uniform crossover:** Create an offspring x by setting each bit $x_i = x'_i$ with probability $1/2$ and $x_i = x''_i$ otherwise, for $1 \leq i \leq n$.
   - **Standard Bit Mutation:** Flip each bit x_i of x with probability $1/n$.
   - $C := C \cup \{x\}$.
4. Set $P := C$ and go to 2.

- EAs in combinatorial optimization (minimum spanning trees, maximum matchings, makespan scheduling . . .)
- EAs in noisy optimization (drift methods really crucial here)
- . . .

Linear Functions: Generalized OneMax

Aim

Case study in drift analysis: bounds for linear functions.
Increases understanding of an innocently looking problem.

Impact

- Prove optimality of parameter setting from practice.
- Prove optimality of (1+1) EA within a larger class of EAs.

The Setting

Task: minimize linear pseudo-Boolean $f : \{0,1\}^n \to \mathbb{R}$

$f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ w. l. o. g. $0 < w_1 \leq \cdots \leq w_n$

The Setting

Task: minimize linear pseudo-Boolean $f : \{0,1\}^n \to \mathbb{R}$

$f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ w. l. o. g. $0 < w_1 \leq \cdots \leq w_n$

Randomized Local Search (RLS)

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping exactly one bit in x, chosen uniformly at random.
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

The Setting

Task: minimize linear pseudo-Boolean $f : \{0,1\}^n \to \mathbb{R}$

$f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ w. l. o. g. $0 < w_1 \leq \cdots \leq w_n$

(1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x indep'ly with prob. p (default $p = \frac{1}{n}$).
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

The Setting

Task: minimize linear pseudo-Boolean $f : \{0,1\}^n \to \mathbb{R}$

$f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ w. l. o. g. $0 < w_1 \leq \cdots \leq w_n$

(1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x indep'ly with prob. p (default $p = \frac{1}{n}$).
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

Expected time for minimization? Can't we just reuse OneMax result?
Or use another classical method (coupon collector problem)?

The Setting

Task: minimize linear pseudo-Boolean $f : \{0,1\}^n \to \mathbb{R}$

$f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ w. l. o. g. $0 < w_1 \leq \cdots \leq w_n$

(1+1) EA

1. Choose $x \in \{0,1\}^n$ uniformly at random.
2. Repeat
   - Create y by flipping each bit in x indep'ly with prob. p (default $p = \frac{1}{n}$).
   - If $f(y) \leq f(x)$, set $x := y$.
3. Until happy.

Expected time for minimization? Can't we just reuse OneMax result?
Or use another classical method (coupon collector problem)?

Yes, BUT only for RLS $\to \leq n \ln n + O(n)$ iterations expected

What Makes Analysis of (1+1) EA Difficult

Many light bits exchanged for few heavy ones, $(0, \ldots, 0, 1)$ might be improved to $(1, \ldots, 1, 0)$. How does this affect running time?

What Makes Analysis of (1+1) EA Difficult

Many light bits exchanged for few heavy ones, $(0, \ldots, 0, 1)$ might be improved to $(1, \ldots, 1, 0)$. How does this affect running time?

Some prior work

Expected running time of (1+1) EA with $p = \frac{1}{n}$ on linear function

- $O(n^2)$ (not too difficult)
- $O(n \ln n)$ (Droste, Jansen, Wegener 1998/2002)
- $\leq (1 + o(1))2.02en \ln n$ (Jägersküpper 2008/2011)
- $\leq (1 + o(1))1.39en \ln n$ (Doerr, Johannsen, Winzen 2010)
- $\geq en \ln n - O(n)$ (DJW 2010 + Doerr, Fouz, W. 2011).

What Makes Analysis of (1+1) EA Difficult

Many light bits exchanged for few heavy ones, $(0, \ldots, 0, 1)$ might be improved to $(1, \ldots, 1, 0)$. How does this affect running time?

Some prior work

Expected running time of (1+1) EA with $p = \frac{1}{n}$ on linear function

- $O(n^2)$ (not too difficult)
- $O(n \ln n)$ (Droste, Jansen, Wegener 1998/2002)
- $\leq (1 + o(1))2.02en \ln n$ (Jägersküpper 2008/2011)
- $\leq (1 + o(1))1.39en \ln n$ (Doerr, Johannsen, Winzen 2010)
- $\geq en \ln n - O(n)$ (DJW 2010 + Doerr, Fouz, W. 2011).

Main new result (W. 2012/13)

Running time bound $(1 \pm o(1))\frac{e^c}{c} n \ln n$ if $p = \frac{c}{n}$; optimal for $p = \frac{1}{n}$, which is an often recommended mutation probability in practice.

What Makes Analysis of (1+1) EA Difficult

Many light bits exchanged for few heavy ones, $(0,\ldots,0,1)$ might be improved to $(1,\ldots,1,0)$. How does this affect running time?

Some prior work

Expected running time of (1+1) EA with $p = \frac{1}{n}$ on linear function

- $O(n^2)$ (not too difficult)
- $O(n \ln n)$ (Droste, Jansen, Wegener 1998/2002)
- $\leq (1 + o(1))2.02en \ln n$ (Jägersküpper 2008/2011)
- $\leq (1 + o(1))1.39en \ln n$ (Doerr, Johannsen, Winzen 2010)
- $\geq en \ln n - O(n)$ (DJW 2010 + Doerr, Fouz, W. 2011).

Main new result (W. 2012/13)

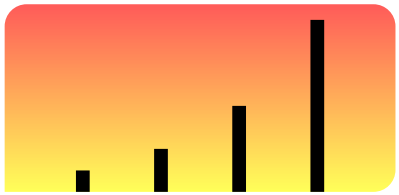
Running time bound $(1 \pm o(1))\frac{e^c}{c} n \ln n$ if $p = \frac{c}{n}$; optimal for $p = \frac{1}{n}$, which is an often recommended mutation probability in practice.

Note: $\left(\left(1-\frac{c}{n}\right)^{n-1}\binom{n}{1}\frac{c}{n}\right)^{-1} \approx \frac{e^c}{c}$ is waiting time for single-bit flip.

Discussion of Distance Function

$\textsc{OneMax}(x_1, \ldots, x_n)$
$= x_1 + \cdots + x_n$

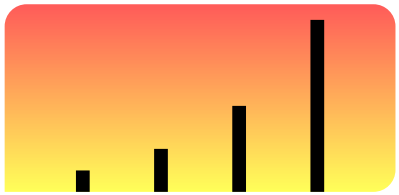
$\textsc{BinVal}(x_1, \ldots, x_n) =$
$x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n$



Discussion of Distance Function

$\textsc{OneMax}(x_1, \ldots, x_n)$
$= x_1 + \cdots + x_n$

$\textsc{BinVal}(x_1, \ldots, x_n) =$
$x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n$



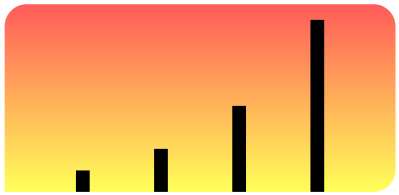
- $\textsc{OneMax}$: 0-bits never lost $\rightarrow$ distance (measured in 1-bits) non-increasing $\rightarrow$ multiplicative drift $\rightarrow$ $en(\ln n + 1)$.

  $\textsc{BinVal}$: can lose almost all 0-bits.

Discussion of Distance Function

$\textsc{OneMax}(x_1, \ldots, x_n)$
$= x_1 + \cdots + x_n$

$\textsc{BinVal}(x_1, \ldots, x_n) =$
$x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n$



- $\textsc{OneMax}$: 0-bits never lost $\rightarrow$ distance (measured in 1-bits) non-increasing $\rightarrow$ multiplicative drift $\rightarrow$ $en(\ln n + 1)$.

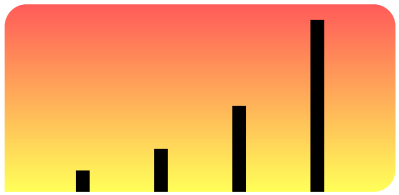
  $\textsc{BinVal}$: can lose almost all 0-bits.

- Consider $\Delta(\text{1-bits}) := (\text{no. 1-bits time } t - \text{no. 1-bits time } t+1)$:

Discussion of Distance Function

$\textsc{OneMax}(x_1, \ldots, x_n)$
$= x_1 + \cdots + x_n$

$\textsc{BinVal}(x_1, \ldots, x_n) =$
$x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n$



- $\textsc{OneMax}$: 0-bits never lost $\rightarrow$ distance (measured in 1-bits) non-increasing $\rightarrow$ multiplicative drift $\rightarrow$ $en(\ln n + 1)$.

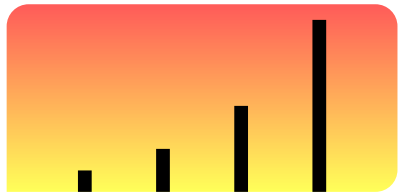
  $\textsc{BinVal}$: can lose almost all 0-bits.

- Consider $\Delta(\text{1-bits}) := (\text{no. 1-bits time } t - \text{no. 1-bits time } t+1)$:
  - in general random
  - possibly negative (= bad)
  - but positive expectation $\geq 1 - (n-1)\frac{1}{n} \geq \frac{1}{n}$ (additive drift)

Discussion of Distance Function

$\textsc{OneMax}(x_1, \ldots, x_n) = x_1 + \cdots + x_n$

$\textsc{BinVal}(x_1, \ldots, x_n) = x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n$



- $\textsc{OneMax}$: 0-bits never lost $\rightarrow$ distance (measured in 1-bits) non-increasing $\rightarrow$ multiplicative drift $\rightarrow$ $en(\ln n + 1)$.

  $\textsc{BinVal}$: can lose almost all 0-bits.

- Consider $\Delta(\text{1-bits}) := (\text{no. 1-bits time } t - \text{no. 1-bits time } t+1)$:
  - in general random
  - possibly negative (= bad)
  - but positive expectation $\geq 1 - (n-1)\frac{1}{n} \geq \frac{1}{n}$ (additive drift)

- Problem: $\frac{1}{n}$ not sufficient for our purpose, gives only $O(n^2)$.

- Need different "potential/distance function" w. r. t. search points $\rightarrow$ distance from optimum is not measured in no. 1-bits.

Finding the Right Potential Function

From now on: $f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ arbitrary.
No. one-bits usually not a good potential function.

Finding the Right Potential Function

From now on: $f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ arbitrary.
No. one-bits usually not a good potential function.

Classical potential functions

- Could try $X_t := f$-value at time t, which is a weighted sum of 1-bits; leads to good multiplicative drift, however $\ln(X_0)$ might be huge.
- Potential function should "compress" steeply increasing weights.
- Previous work: $X_t := \sum_{i=n/2+1}^{n} 2x(t)_i + \sum_{i=1}^{n/2} x(t)_i$ or similar

Finding the Right Potential Function

From now on: $f(x_1, \ldots, x_n) = w_1 x_1 + \cdots + w_n x_n$ arbitrary.
No. one-bits usually not a good potential function.

Classical potential functions

- Could try $X_t := f$-value at time t, which is a weighted sum of 1-bits; leads to good multiplicative drift, however $\ln(X_0)$ might be huge.
- Potential function should "compress" steeply increasing weights.
- Previous work: $X_t := \sum_{i=n/2+1}^{n} 2x(t)_i + \sum_{i=1}^{n/2} x(t)_i$ or similar

Adaptive potential functions

- Idea: do not compress all f in the same way.
- If the linear function at hand is OneMax, why not use OneMax?
- If it is BinVal, can we use more slowly increasing weights?
- First time used by Doerr and Goldberg (2010).

New Potential Function Proves Upper Bound

Define $g(x) = g_1 x_1 + \cdots + g_n x_n$ by $g_1 := 1 = \gamma_1$ and

$$g_i := \min\left\{ \underbrace{\left(1 + \frac{(\ln \ln n)p}{(1-p)^{n-1}}\right)^{i-1}}_{=:\gamma_i}, g_{i-1} \cdot \frac{w_i}{w_{i-1}} \right\} \text{ for } 2 \leq i \leq n.$$

New Potential Function Proves Upper Bound

Define $g(x) = g_1 x_1 + \cdots + g_n x_n$ by $g_1 := 1 = \gamma_1$ and

$$g_i := \min\left\{ \underbrace{\left(1 + \frac{(\ln \ln n)p}{(1-p)^{n-1}}\right)^{i-1}}_{=:\gamma_i}, g_{i-1} \cdot \frac{w_i}{w_{i-1}} \right\} \text{ for } 2 \leq i \leq n.$$

Properties

- g includes the mutation probability p.
- If f has "steep" coefficients (e. g., BinVal), then $g_i = \gamma_i$.
- In "flat" regions, we reproduce f (with some scaling).

New Potential Function Proves Upper Bound

Define $g(x) = g_1 x_1 + \cdots + g_n x_n$ by $g_1 := 1 = \gamma_1$ and

$$g_i := \min\left\{ \underbrace{\left(1 + \frac{(\ln\ln n)p}{(1-p)^{n-1}}\right)^{i-1}}_{=:\gamma_i}, g_{i-1} \cdot \frac{w_i}{w_{i-1}} \right\} \text{ for } 2 \le i \le n.$$

Properties

- g includes the mutation probability p.
- If f has "steep" coefficients (e. g., BinVal), then $g_i = \gamma_i$.
- In "flat" regions, we reproduce f (with some scaling).
- Crucial: $E(X_t - X_t \mid X_t = s) \ge (1 - o(1)) \cdot s \cdot p \cdot (1-p)^{n-1}$ (X_t is g-value at time t).

Further Results

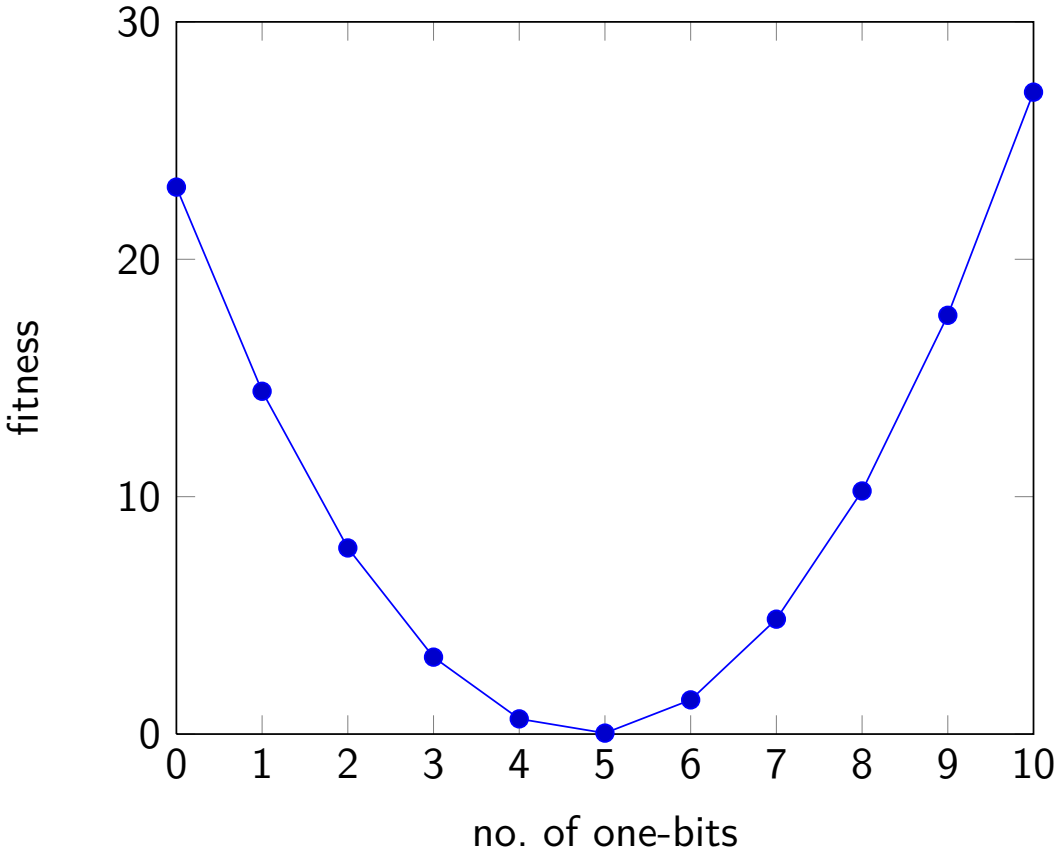
- OneMax: stochastically smallest optimization time within class
- Tight lower bounds using multiplicative drift
- Phase transition from polynomial to superpolynomial in regime $p = \Theta((\ln n)/n)$
- (1+1) EA as optimal mutation-based, population-based algorithm

for $t := 0 \to \mu - 1$ **do**
  create $x_t \in \{0,1\}^n$ uniformly at random.
end for
repeat
  select $x \in \{x_0, \ldots, x_t\}$ according to t and $f(x_0), \ldots, f(x_t)$.
  create x_{t+1} by flipping each bit in x indep'ly with probability p.
  $t := t + 1$.
until happy

Recent Developments

- Is expected running time the right measure of efficiency?
- Where is drift analysis going?
- Is drift analysis useful for other purposes than analysis of RSHs?

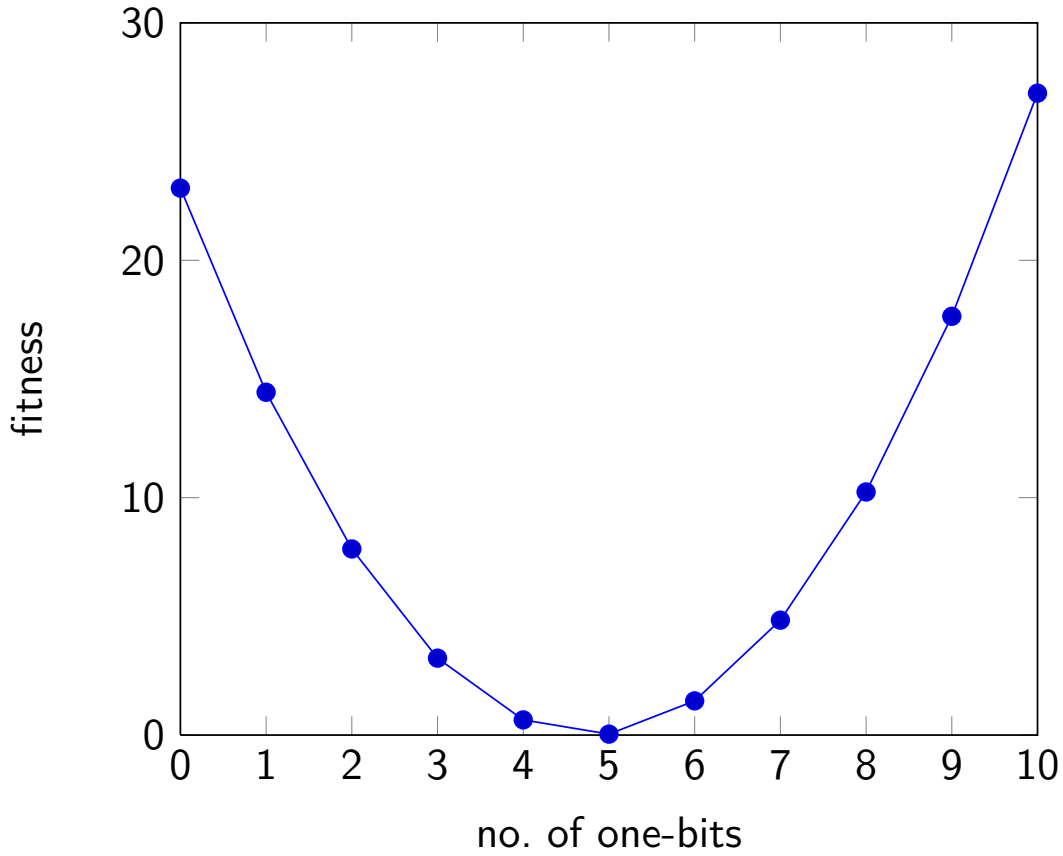
Why Expected Time is Not Enough
(1+1) EA on DISTANCE (DJW 2002)



Each with prob. $\approx \frac{1}{2}$:

- local optimum reached, time n^n to escape
- global optimum found in $O(n \log n)$ steps

Why Expected Time is Not Enough
(1+1) EA on DISTANCE (DJW 2002)



Each with prob. $\approx \frac{1}{2}$:

- local optimum reached, time n^n to escape
- global optimum found in $O(n \log n)$ steps

Expected running time $\Omega(n^n)$ for (1+1) EA but function is not really difficult (expected 2 restarts sufficient)

$\rightarrow$ results on the distribution of running time required.

Sharp Concentration of Measure

Classical example

X number of 1-bits in uniformly random $x \in \{0,1\}^n$. Then (Chernoff)

$$\Pr(|X - E(X)| \geq \delta\sqrt{n}) \leq 2e^{-\delta^2/3}.$$

$\rightarrow$ deviation in lower-order term exponentially unlikely.

Sharp Concentration of Measure

Classical example

X number of 1-bits in uniformly random $x \in \{0,1\}^n$. Then (Chernoff)

$$\Pr(|X - E(X)| \geq \delta\sqrt{n}) \leq 2e^{-\delta^2/3}.$$

$\rightarrow$ deviation in lower-order term exponentially unlikely.

Similar results for running time?

Known: for running time T of (1+1) EA on OneMax:
$\Pr(T > E(T) + \delta n) \leq e^{-\delta/e}$ (by multiplicative drift analysis).

No corresponding lower tail ($\Pr(T < E(T) - \delta n)$) known before.

Sharp Concentration of Measure

Classical example

X number of 1-bits in uniformly random $x \in \{0,1\}^n$. Then (Chernoff)

$$\Pr(|X - E(X)| \geq \delta\sqrt{n}) \leq 2e^{-\delta^2/3}.$$

$\rightarrow$ deviation in lower-order term exponentially unlikely.

Similar results for running time?

Known: for running time T of (1+1) EA on OneMax:
$\Pr(T > E(T) + \delta n) \leq e^{-\delta/e}$ (by multiplicative drift analysis).

No corresponding lower tail ($\Pr(T < E(T) - \delta n)$) known before.

Will also get lower tails with new theorems. Running times are identified as "almost deterministic", despite algorithms being heavily randomized.

Sharp Concentration of Running Times

Drift analysis with tail bounds (Lehre and W., 2014)

- Suppose we know (bound on) $h(s) = E(X_t - X_{t+1} \mid X_t = s)$.
- Consider potential function $g(s) = \frac{1}{h(1)} + \int_1^s \frac{1}{h(x)}\,\mathrm{d}x$
- Bound $E(e^{-\lambda(g(s)-g(X_{t+1}))} \mid X_t = s) \leq \beta < 1$ for some $\lambda > 0$.
- Then $\Pr(T_0 > t) \leq \beta^t \cdot e^{\lambda g(X_0)}$.
- Analogous bound on $\Pr(T_0 < t)$ available.

Sharp Concentration of Running Times

Drift analysis with tail bounds (Lehre and W., 2014)

- Suppose we know (bound on) $h(s) = E(X_t - X_{t+1} \mid X_t = s)$.
- Consider potential function $g(s) = \frac{1}{h(1)} + \int_1^s \frac{1}{h(x)}\,\mathrm{d}x$
- Bound $E(e^{-\lambda(g(s)-g(X_{t+1}))} \mid X_t = s) \leq \beta < 1$ for some $\lambda > 0$.
- Then $\Pr(T_0 > t) \leq \beta^t \cdot e^{\lambda g(X_0)}$.
- Analogous bound on $\Pr(T_0 < t)$ available.

Implications within theory of RSHs

- For (1+1) EA on OneMax: $\Pr(|T - en\ln n| \geq \delta n) \leq e^{-\Omega(\delta)}$.
- Also valid for linear functions!
- Similar results for other benchmark functions (e. g., expected $\Theta(n^2)$, deviation $\delta n^{3/2}$ has prob. $e^{-\Omega(\delta)}$.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$. Consider cycles, e. g., $1 \to 5 \to 8 \to 3 \to 1$.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$. Consider cycles, e. g., $1 \to 5 \to 8 \to 3 \to 1$.
- How to determine $T(n)$, the number of cycles?
- Recurrence $T(n) = 1 + T(h(n))$, where $h(n) \sim \text{Unif}(0, \ldots, n-1)$.

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$. Consider cycles, e. g., $1 \to 5 \to 8 \to 3 \to 1$.
- How to determine $T(n)$, the number of cycles?
- Recurrence $T(n) = 1 + T(h(n))$, where $h(n) \sim \text{Unif}(0, \ldots, n-1)$.
- Karp: $\Pr(T(n) > \log_2(n+1) + \delta) \leq 2^{-\delta+1}$

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$. Consider cycles, e. g., $1 \to 5 \to 8 \to 3 \to 1$.
- How to determine $T(n)$, the number of cycles?
- Recurrence $T(n) = 1 + T(h(n))$, where $h(n) \sim \text{Unif}(0, \ldots, n-1)$.
- Karp: $\Pr(T(n) > \log_2(n+1) + \delta) \leq 2^{-\delta+1}$
- For comparison, $E(T(n)) = \ln n + \Theta(1)$;

Drift Analysis and Probabilistic Recurrences

Scenario

- Consider a recurrrence $T(n) = 1 + T(h(n))$, where $h(n) \leq n$ random and $E(h(n))$ known; $T(0) = 0$.
- Karp (JACM 1994) bounds $\Pr(T(n) \geq u(n) + \delta)$, where $u(n)$ "solution" of $T(n) = 1 + T(E(h(n)))$ (de-randomized recurrence).
- Recurrence is a stochastic process: $X_0 = n$, $X_{t+1} = h(X_t)$.
- Can be treated by drift analysis.

Application to random combinatorial structure

- Draw random permutation π of $\{1, \ldots, n\}$. Consider cycles, e. g., $1 \to 5 \to 8 \to 3 \to 1$.
- How to determine $T(n)$, the number of cycles?
- Recurrence $T(n) = 1 + T(h(n))$, where $h(n) \sim \text{Unif}(0, \ldots, n-1)$.
- Karp: $\Pr(T(n) > \log_2(n+1) + \delta) \leq 2^{-\delta+1}$
- For comparison, $E(T(n)) = \ln n + \Theta(1)$; note: $\log_2 n \approx 1.44 \ln n$.

Sharp Tail Bound on No. of Cycles

New result

Using drift analysis with tail bounds, we get

$\Pr(T(n) < (1-\delta)(\ln n)) \leq e^{-\frac{\delta^2}{4}(1-o(1))\ln n}$ and

$\Pr(T(n) > (1+\delta)((\ln n)+1)) \leq e^{-\frac{\min\{\delta,\delta^2\}}{6}\ln n}$.

Compared to Karp's technique, need knowledge of distribution of $h(n)$, not only the expectation.

Sharp Tail Bound on No. of Cycles

New result

Using drift analysis with tail bounds, we get

$\Pr(T(n) < (1-\delta)(\ln n)) \leq e^{-\frac{\delta^2}{4}(1-o(1))\ln n}$ and

$\Pr(T(n) > (1+\delta)((\ln n)+1)) \leq e^{-\frac{\min\{\delta,\delta^2\}}{6}\ln n}$.

Compared to Karp's technique, need knowledge of distribution of $h(n)$, not only the expectation.

Obvious topics for future research

- $T(n) = a(n) + T(h(n))$ for $a(n) > 1$
- More than one recursive call?

Summary

Drift analysis for the analysis of EAs

- Introduced additive and multiplicative drift.
- Applied drift analysis to bound running time of (1+1) EA on linear functions:
  - Tight bounds up to lower-order terms.
  - Structural insight: multiple-bit flips do neither help or harm.
  - Practically used mutation probability is theoretically optimal.
  - "Greedy" (1+1) EA is optimal mutation-based, population-based EA.
  - Adaptive multiplicative drift analysis is very powerful technique, but finding the "right" potential function can be difficult.
- Drift analysis can yield sharp-concentration results.
- Looking into EAs from CS perspective may be promising: techniques developed here may be useful for problems from classical TCS.

Summary

Drift analysis for the analysis of EAs

- Introduced additive and multiplicative drift.
- Applied drift analysis to bound running time of (1+1) EA on linear functions:
  - Tight bounds up to lower-order terms.
  - Structural insight: multiple-bit flips do neither help or harm.
  - Practically used mutation probability is theoretically optimal.
  - "Greedy" (1+1) EA is optimal mutation-based, population-based EA.
  - Adaptive multiplicative drift analysis is very powerful technique, but finding the "right" potential function can be difficult.
- Drift analysis can yield sharp-concentration results.
- Looking into EAs from CS perspective may be promising: techniques developed here may be useful for problems from classical TCS.

Thank you for your attention!