An Introduction on Theory of Evolutionary Optimization

Chao Qian (钱 超)
http://staff.ustc.edu.cn/~chaoqian/

UBRI, School of Computer Science and Technology
University of Science and Technology of China, China

第三届演化计算与学习研讨会ECOLE 2016

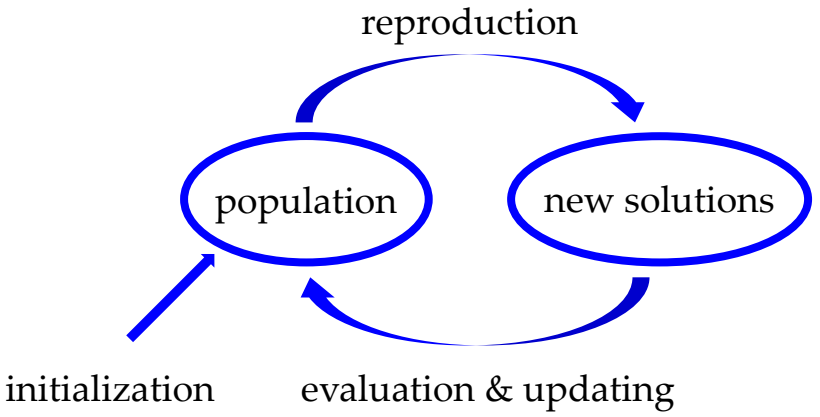
Outline

- **Introduction**
- Running time analysis approaches
- Running time analysis results
- Examples of running time analysis leading to improved design of EAs
- Summary

Evolutionary algorithms

Evolutionary algorithms (EAs) have been widely used in real applications

GA, ES, GP, PSO,
ACO, DE, ……



theoretical analysis: difficult

EAs are randomized and complex

- With the same input, the performed operations and the output can be different
- The designed reproduction operators and updating mechanisms can be complex

The theoretical foundation of EAs is still weak,

but important

Early theoretical studies

Schema theorem [Holland, 1975]

- Proposed to explain the behavior of genetic algorithms

$$\mathrm{E}[m(H,t+1)] \geq \frac{m(H,t)f(H)}{a_t}(1-p)$$

- Critiqued from several directions, and even wrong [Reeves & Rowe, 2002]
- Cannot explain the performance or limit behaviors of EAs

Convergence analysis [Rudolph, FI'98]

- Given unlimited time, can the algorithm find the optimum with probability 1?

$$lim_{t\to+\infty} P(\xi_t \in \mathrm{X}^*) = 1?$$

- Sufficient conditions:
  - ✓ There is a positive probability to reach any solution in the search space from any other solution (satisfied by most canonical EAs)
  - ✓ The algorithm keeps the best found solution (elitism)

Running time analysis

Convergence analysis

$lim_{t\to+\infty} P(\xi_t \in X^*) = 1$?

How fast does it converge?

Running time analysis

$\tau = \min\{t \geq 0 \mid \xi_t \in X^*\}$

The number of iterations until finding an optimal solution for the first time

Running time analysis

Convergence analysis

$lim_{t\to+\infty} P(\xi_t \in X^*) = 1$?

Running time analysis

$\tau = \min\{t \geq 0 \mid \xi_t \in X^*\}$

The leading theoretical aspect

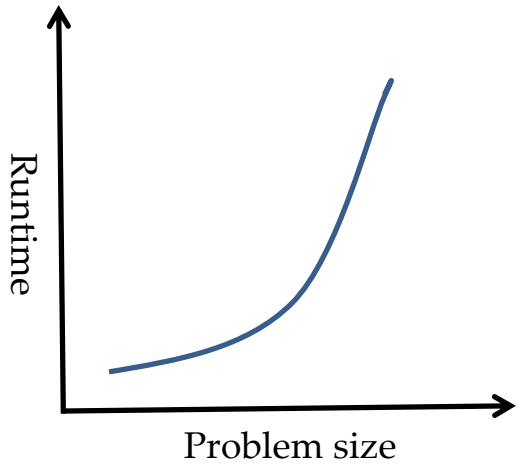
[Auger & Doerr, 2011; Neumann & Witt, 2012]

The number of iterations until finding an optimal solution for the first time

Running time complexity

- The number of iterations × the number of fitness evaluations in each iteration
- Usually grows with the problem size and expressed in asymptotic notations

  e.g., (1+1)-EA solving LeadingOnes: $O(n^2)$



Running time analysis

Convergence analysis

$lim_{t\to+\infty} P(\xi_t \in X^*) = 1$?

Running time analysis

$\tau = \min\{t \geq 0 \mid \xi_t \in X^*\}$

The leading theoretical aspect

[Auger & Doerr, 2011; Neumann & Witt, 2012]

The number of iterations until finding an optimal solution for the first time

A quick guide to asymptotic notations:

Let g and f be two functions defined on the real numbers.

- $g \in O(f)$: $\exists M > 0$ such that $g(x) \leq M \cdot f(x)$ for all sufficiently large x
- $g \in \Omega(f)$: $f \in O(g)$
- $g \in \Theta(f)$: $g \in O(f)$ and $g \in \Omega(f)$

$g \in O(f) \to g \leq f$

$g \in \Omega(f) \to g \geq f$

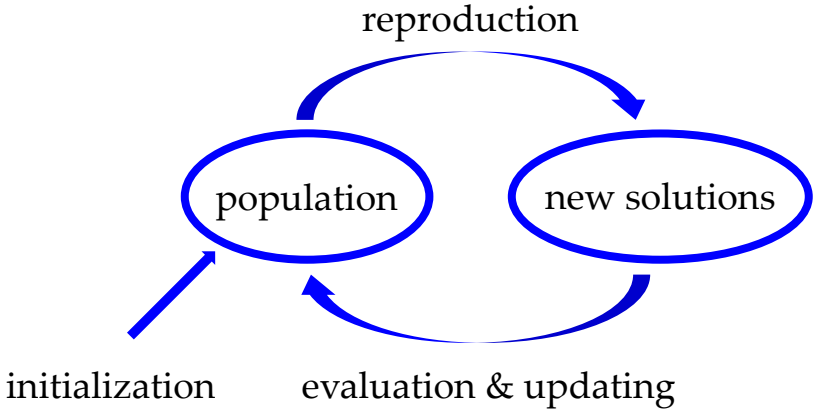
$g \in \Theta(f) \to g = f$

Outline

- Introduction
- **Running time analysis approaches**
- Running time analysis results
- Examples of running time analysis leading to improved design of EAs
- Summary

Markov chain modeling

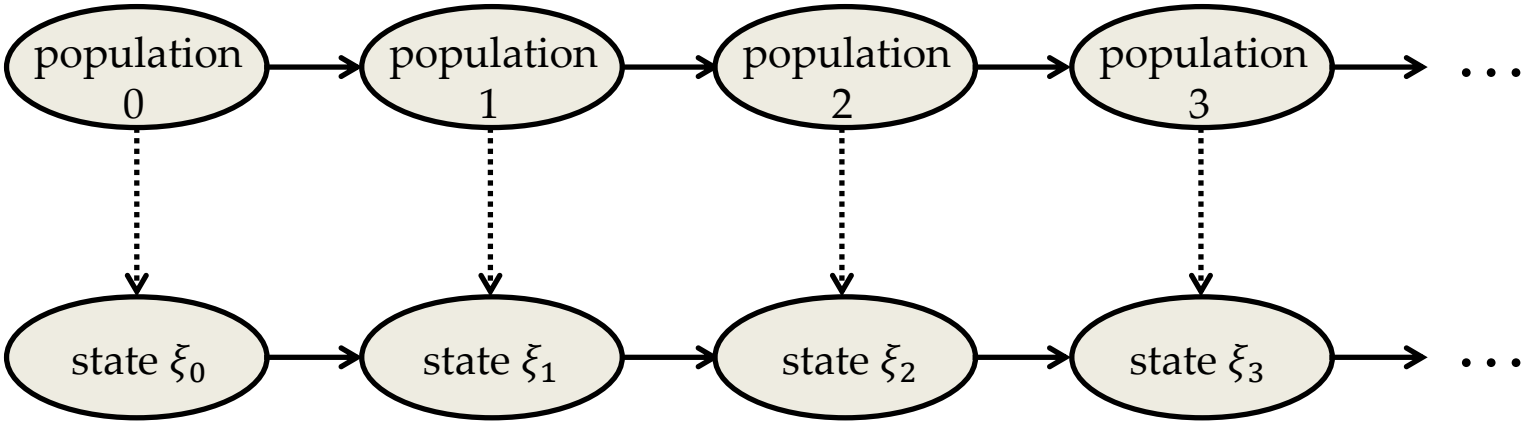
EA:



Running time analysis

$\tau = \min \{t \geq 0 \mid \xi_t \in X^*\}$

a random variable
- $\mathrm{E}[\tau]$
- $P(\tau \leq T)$



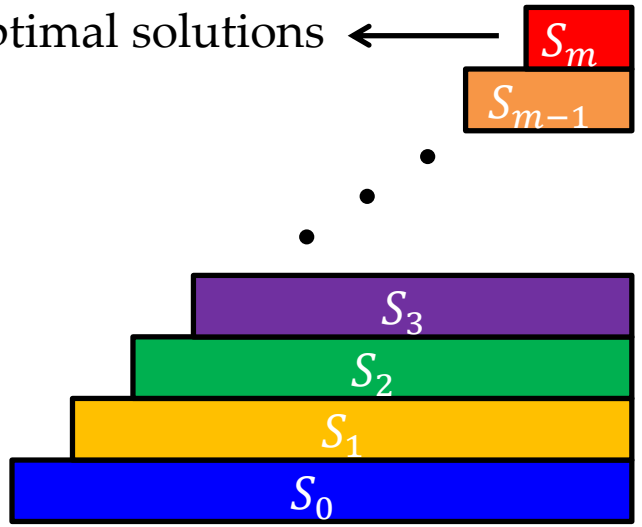
Markov chain: $P(\xi_t|\xi_{t-1}, \ldots, \xi_1, \xi_0) = P(\xi_t|\xi_{t-1})$

Fitness level method

The basic idea [Droste et al., TCS'02]:

1. Divide the solution space S into $m+1$ subspaces $S_0, S_1, \ldots, S_m$

- $\forall i \neq j: S_i \cap S_j = \emptyset,\ \bigcup_{i=0}^{m} S_i = S$
- $\forall i < j, x \in S_i, y \in S_j: f(x) < f(y)$



Fitness level method

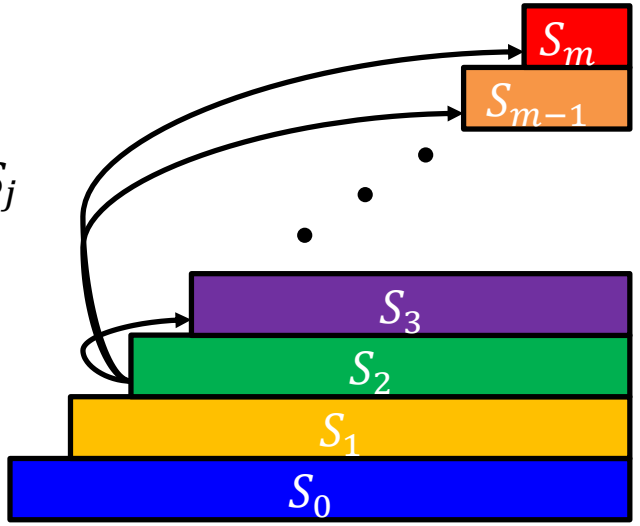
The basic idea [Droste et al., TCS'02]:

1. Divide the solution space S into $m+1$ subspaces $S_0, S_1, \ldots, S_m$

   - $\forall i \neq j: S_i \cap S_j = \emptyset,\ \bigcup_{i=0}^{m} S_i = S$
   - $\forall i < j, x \in S_i, y \in S_j: f(x) < f(y)$

2. Bounds on the probability of leaving S_i to higher S_j

   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \geq v_i$
   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \leq u_i$



Expected running time

Upper bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j}$

the initial distribution

Fitness level method

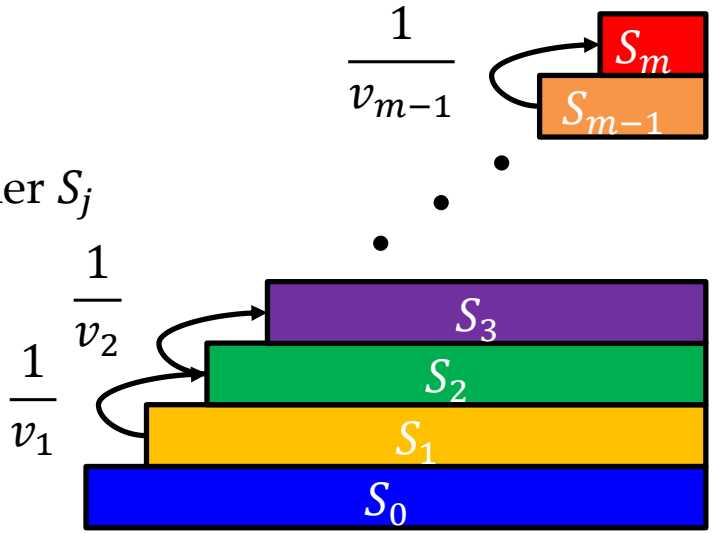
The basic idea [Droste et al., TCS'02]:

1. Divide the solution space S into $m+1$ subspaces $S_0, S_1, \ldots, S_m$

   - $\forall i \neq j: S_i \cap S_j = \emptyset,\ \bigcup_{i=0}^{m} S_i = S$
   - $\forall i < j, x \in S_i, y \in S_j: f(x) < f(y)$

2. Bounds on the probability of leaving S_i to higher S_j

   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \geq v_i$
   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \leq u_i$



Expected running time

Upper bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j}$

the initial distribution

Fitness level method

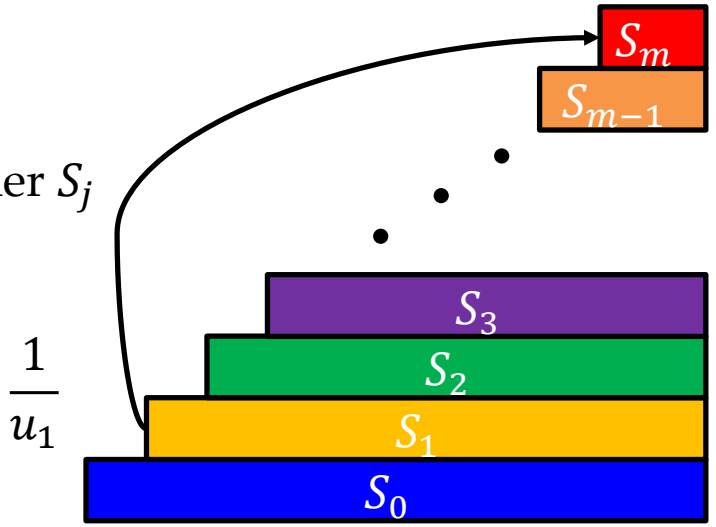
The basic idea [Droste et al., TCS'02]:

1. Divide the solution space S into $m+1$ subspaces $S_0, S_1, \ldots, S_m$

   - $\forall i \neq j: S_i \cap S_j = \emptyset,\ \bigcup_{i=0}^{m} S_i = S$
   - $\forall i < j, x \in S_i, y \in S_j: f(x) < f(y)$

2. Bounds on the probability of leaving S_i to higher S_j

   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \geq v_i$
   - $P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j | \xi_t \in S_i) \leq u_i$



Expected running time

Upper bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j}$

Lower bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \frac{1}{u_i}$

Example: (1+1)-EA for OneMax

(1+1)-EA:

Given a pseudo-Boolean function f:

1. $x :=$ randomly selected from $\{0,1\}^n$.
2. Repeat until some termination criterion is met
3. $\quad x' :=$ flip each bit of x with probability $1/n$.
4. $\quad$ if $f(x') \geq f(x)$
5. $\quad\quad x = x'$.

OneMax:

$\arg max_{x \in \{0,1\}^n} \sum_{i=1}^{n} x_i$ → Count the number of 1-bits

Theorem. [Droste et al., TCS'02] The expected running time of the (1+1)-EA solving the OneMax problem is $O(n \log n)$.

Proof

Theorem. [Droste et al., TCS'02] The expected running time of the (1+1)-EA solving the OneMax problem is $O(n \log n)$.

Main idea:

the number of 1-bits

- Divide the solution space $\{0,1\}^n$ into $S_0, S_1, \ldots, S_n$ with $S_i = \{x \in \{0,1\}^n \mid |x| = i\}$
- The probability of jumping to higher S_j from S_i is lower bounded by

$$P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j \mid \xi_t \in S_i) \geq \frac{n-i}{n} \cdot \left(1-\frac{1}{n}\right)^{n-1}$$

keep the other bits unchanged

flip one of the $n-i$ 0-bits

Upper bound on the expected running time:

$$\sum_{i=0}^{n-1} \pi_0(S_i) \cdot \sum_{j=i}^{n-1} \frac{1}{v_j} \leq \sum_{j=0}^{n-1} \frac{1}{v_j}$$

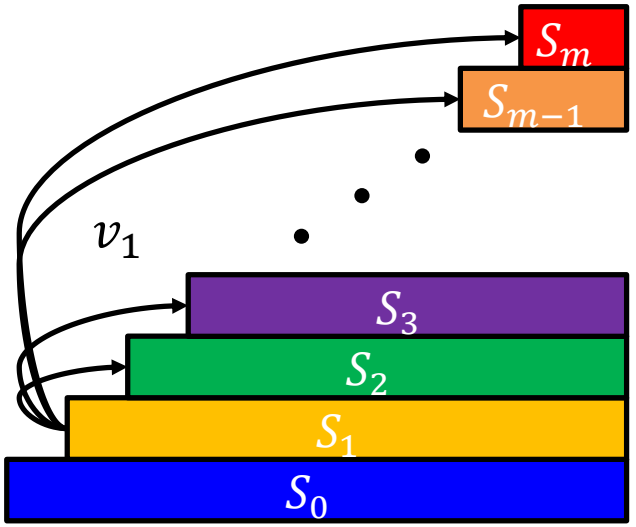
$$\leq \sum_{j=0}^{n-1} \frac{n}{n-j} \frac{1}{\left(1-\frac{1}{n}\right)^{n-1}} \leq en \sum_{j=1}^{n} \frac{1}{j} \in O(n \log n)$$

Refined fitness level method

Original [Droste et al., TCS'02]:

$$P(\xi_{t+1} \in \bigcup_{j=i+1}^{m} S_j \mid \xi_t \in S_i) \geq v_i$$

Upper bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j}$

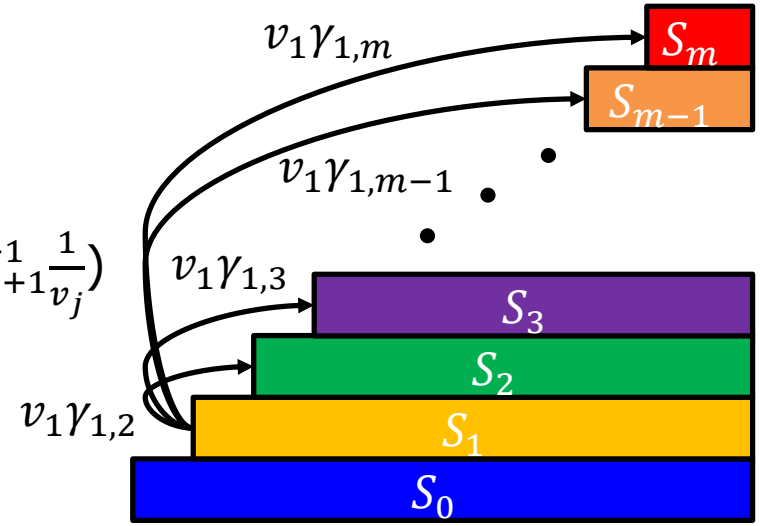


Refined [Sudholt, TEC'13]:

$$P(\xi_{t+1} \in S_j \mid \xi_t \in S_i) \geq v_i \cdot \gamma_{i,j}$$

$$\sum_{j=i+1}^{m} \gamma_{i,j} = 1 \qquad \gamma_{i,j} \leq \chi \sum_{k=j}^{m} \gamma_{i,k}$$

Upper bound: $\sum_{i=0}^{m-1} \pi_0(S_i) \cdot \left(\frac{1}{v_i} + \chi \sum_{j=i+1}^{m-1} \frac{1}{v_j}\right)$



Original is a specialization of refined

If $\chi = 1$, upper bound

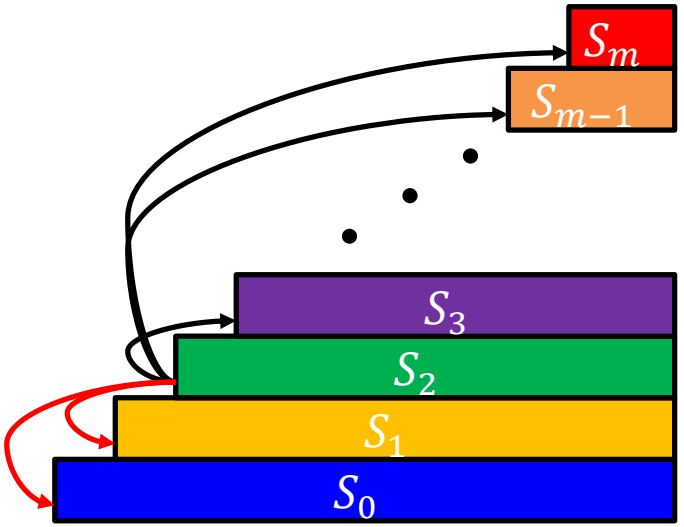
If $\chi = 0$, lower bound

Refined fitness level method

The above two fitness level methods

[Droste et al., TCS'02; Sudholt, TEC'13]

- Only consider jumping to higher levels
- Proposed for elitist EAs



The fitness level method for non-elitist EAs [Dang & Lehre, Algorithmica'16]

- allows jumping to lower levels

Theorem 8 *Given a function $f : \mathcal{X} \to \mathbb{R}$, and an f-based partition $(A_1, \ldots, A_{m+1})$, let T be the number of selection-variation steps until Algorithm 1 with a selection mechanism p_{sel} obtains an element in A_{m+1} for the first time. If there exist parameters $p_0, s_1, \ldots, s_m, s_* \in (0, 1]$, and $\gamma_0 \in (0, 1)$ and $\delta > 0$, such that*

(C1) $p_{\text{mut}}\left(y \in A_i^+ \mid x \in A_j\right) \geq s_j \geq s_*$ *for all* $j \in [m]$,

(C2) $p_{\text{mut}}\left(y \in A_j \cup A_j^+ \mid x \in A_j\right) \geq p_0$ *for all* $j \in [m]$,

(C3) $\beta(\gamma, P)\, p_0 \geq (1+\delta)\gamma$ *for all* $P \in \mathcal{X}^\lambda$ *and* $\gamma \in (0, \gamma_0]$,

(C4) $\lambda \geq \frac{2}{a} \ln\left(\frac{16m}{ac\varepsilon s_*}\right)$ *with* $a = \frac{\delta^2 \gamma_0}{2(1+\delta)}$, $\varepsilon = \min\{\delta/2, 1/2\}$ *and* $c = \varepsilon^4/24$,

then

$$\mathbf{E}[T] \leq \frac{2}{c\varepsilon}\left(m\lambda(1+\ln(1+c\lambda)) + \frac{p_0}{(1+\delta)\gamma_0}\sum_{j=1}^{m}\frac{1}{s_j}\right).$$

Drift analysis

The basic idea [Sasaki & Hajek, JACM'88; He & Yao, AIJ'01]:

1. Design a distance function $V(x): X \to \mathbb{R}$ to measure the distance from a state $x \in X$ to the optimal state space X^*
   - $\forall x \in X - X^*: V(x) > 0$
   - $\forall x \in X^*: V(x) = 0$

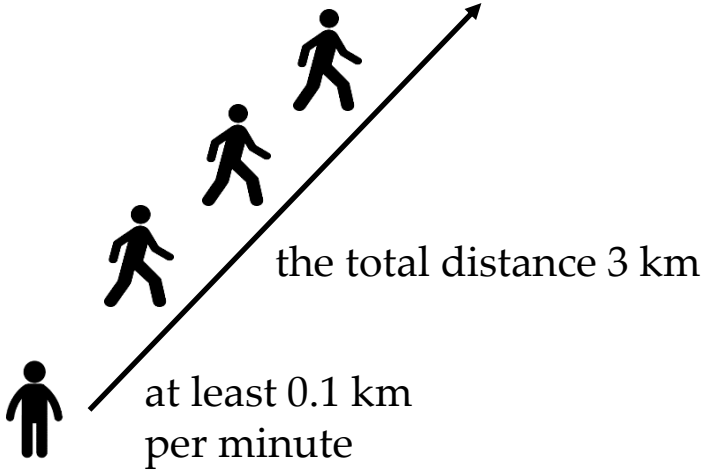
2. Bounds on the expected drift in one step
   - $\mathrm{E}[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq c_l$
   - $\mathrm{E}[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \leq c_u$

Expected running time

Upper bound: $\sum_{x \in X} \pi_0(x) \cdot \frac{V(x)}{c_l}$

the initial distribution

The hotel



the time $\leq \frac{3}{0.1} = 30$ minutes

Drift analysis

The basic idea [Sasaki & Hajek, JACM'88; He & Yao, AIJ'01]:

1. Design a distance function $V(x): X \to \mathbb{R}$ to measure the distance from a state $x \in X$ to the optimal state space X^*
   - $\forall x \in X - X^*: V(x) > 0$
   - $\forall x \in X^*: V(x) = 0$

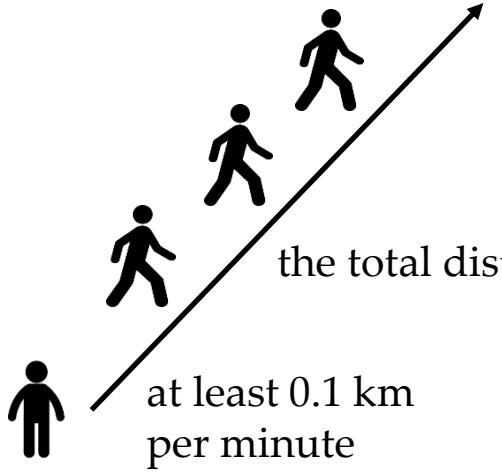
2. Bounds on the expected drift in one step
   - $\mathrm{E}[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq c_l$
   - $\mathrm{E}[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \leq c_u$

Expected running time

Upper bound: $\sum_{x \in X} \pi_0(x) \cdot \frac{V(x)}{c_l}$

Lower bound: $\sum_{x \in X} \pi_0(x) \cdot \frac{V(x)}{c_u}$

The hotel



the time $\leq \frac{3}{0.1} = 30$ minutes

Example: (1+1)-EA for LeadingOnes

(1+1)-EA:

Given a pseudo-Boolean function f:

1. $x :=$ randomly selected from $\{0,1\}^n$.
2. Repeat until some termination criterion is met
3. $\quad x' :=$ flip each bit of x with probability $1/n$.
4. $\quad$ if $f(x') \geq f(x)$
5. $\quad\quad x = x'$.

LeadingOnes:

$$\arg max_{x \in \{0,1\}^n} \left(\sum_{i=1}^{n} \prod_{j=1}^{i} x_j \right)$$

$LO(x)$

Count the number of consecutive 1-bits starting from the left

e.g., $f(11010) = 2$, $f(01111) = 0$

Theorem. [He & Yao, AIJ'01] The expected running time of the (1+1)-EA solving the LeadingOnes problem is $O(n^2)$.

Proof

Theorem. [He & Yao, AIJ'01] The expected running time of the (1+1)-EA solving the LeadingOnes problem is $O(n^2)$.

Main idea:

- Design the distance function $V(x) = n - LO(x)$

the number of leading 1-bits

- The expected drift from a solution x with $LO(x) = i$ is lower bounded by

$$E[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = x) \geq 1 \cdot \frac{1}{n} \cdot (1 - \frac{1}{n})^i$$

keep the i leading 1-bits unchanged

$LO(x) = i \rightarrow \geq i + 1$

flip the first 0-bit

Upper bound on the expected running time:

$$\sum_{x \in X} \pi_0(x) \cdot \frac{V(x)}{c_l} \leq \frac{n}{c_l}$$

$$\leq n \cdot n \cdot \frac{1}{(1-\frac{1}{n})^i} \leq en^2 \in O(n^2)$$

Multiplicative drift analysis

Original [He & Yao, AIJ'01]:

$\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq c_l$ → not depend on ξ_t

Upper bound: $\sum_{x \in X} \pi_0(x) \cdot \frac{V(x)}{c_l}$

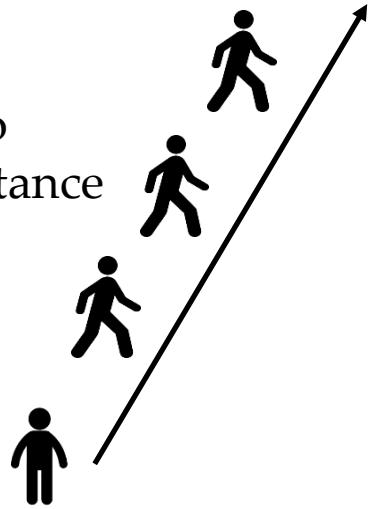
The hotel

Multiplicative [Doerr et al., Algorithmica'12]:

$\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq \delta \cdot V(\xi_t)$ proportional to the current distance

Upper bound: $\sum_{x \in X} \pi_0(x) \cdot \frac{1 + \log\left(V(x)/V_{min}\right)}{\delta}$

$\min\{V(x) \mid V(x) > 0\}$

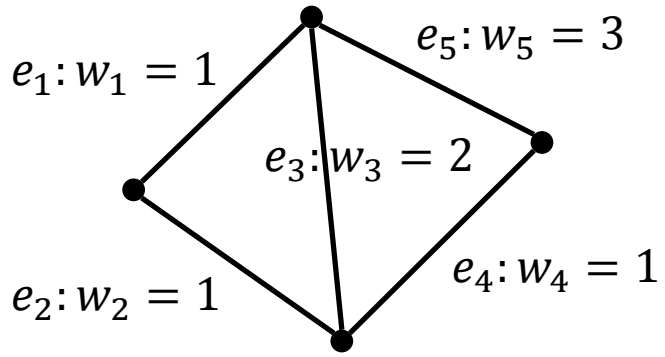


Multiplicative is not stronger than original

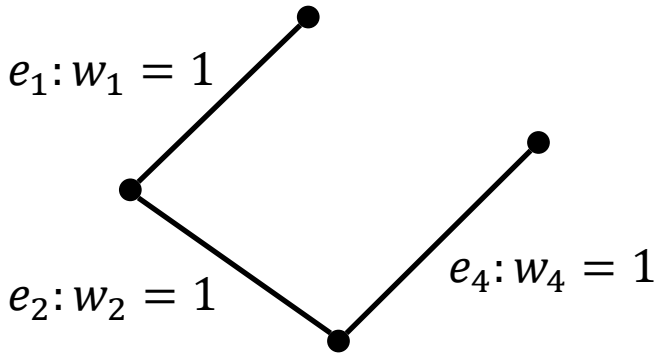
Example: (1+1)-EA for MST

Minimum spanning tree (MST):

- Given: an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive integer weights $w: E \to \mathbb{N}$
- The Goal: find a connected subgraph $E' \subseteq E$ with the minimum weight



The original graph



The minimum spanning tree

Example: (1+1)-EA for MST

(1+1)-EA:

Given a pseudo-Boolean function f:

1. $x :=$ randomly selected from $\{0,1\}^n$.
2. Repeat until some termination criterion is met
3. $x' :=$ flip each bit of x with probability $1/n$.
4. if $f(x') \geq f(x)$
5. $x = x'$.

Solution representation: $x \in \{0,1\}^m \leftrightarrow$ a subgraph

e.g., $\{e_1, e_2, e_4\} \rightarrow 11010$ $x_i = 1$ means that edge e_i is selected

Fitness function:

the number of connected components

$$\min \quad f(x) = (c(x) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$$

$w_{ub} = n^2 \cdot w_{max}$, to make a subgraph with less connected components better

Proof

Theorem. [Neumann & Wegener, TCS'07; Doerr et al., Algorithmica'12] The expected running time of the (1+1)-EA solving the MST problem is $O(m^2(\log n + \log w_{max}))$.

Main idea:

(1) obtain a connected subgraph $\longrightarrow$ $c(x) = 1$

(2) obtain a minimum spanning tree

The analysis of phase (1): $\min\ f(x) = (c(x) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$

- $c(x)$ cannot increase
- at least $c(x) - 1$ edges, the insertion of which can decrease $c(x)$ by 1

the probability of decreasing $c(x)$ by 1 is at least $\frac{c(x)-1}{m}(1-\frac{1}{m})^{m-1}$ $\Longrightarrow$ the expected steps for decreasing $c(x)$ by 1 is at most $\frac{em}{c(x)-1}$

The expected running time: $\sum_{c(x)=n}^{2} \frac{em}{c(x)-1} \in O(m \log n)$

Proof

The analysis of phase (2): $\min\ f(x) = (c(x) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$

- it will always be connected, i.e., $c(x) = 1$ always holds
- to analyze $f(x) \rightarrow f_{opt}$ the weight of a minimum spanning tree

Using multiplicative drift analysis:

- design the distance function: $V(x) = f(x) - f_{opt}$
- analyze the expected drift:

$$\mathrm{E}[\,V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = x] = V(x) - \mathrm{E}[\,V(\xi_{t+1}) \mid \xi_t = x] = f(x) - \mathrm{E}[\,f(\xi_{t+1}) \mid \xi_t = x]$$

$$\geq f(x) - \left(\sum_{i=1}^{m-(n-1)} f(y^i) \cdot \frac{1}{m}\left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^{n} f(z^i) \cdot \frac{1}{m^2}\left(1 - \frac{1}{m}\right)^{m-2} + (1 - \cdots) f(x)\right)$$

there exists a set of $m - (n-1)$ 1-bit flips and a set of n 2-bit flips such that the average weight decrease is at least $(f(x) - f_{opt})/(m+1)$

$$\geq \frac{1}{m}\left(1 - \frac{1}{m}\right)^{m-1} \frac{f(x) - f_{opt}}{m+1} \geq \frac{1}{em(m+1)} V(x)$$

Proof

The analysis of phase (2): $\min f(x) = (c(x) - 1) \cdot w_{ub} + \sum_{i:x_{i=1}} w_i$

- it will always be connected, i.e., $c(x) = 1$ always holds
- to analyze $f(x) \rightarrow f_{opt}$ the weight of a minimum spanning tree

Using multiplicative drift analysis:

- design the distance function: $V(x) = f(x) - f_{opt}$
- analyze the expected drift:

$$\mathrm{E}[\, V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = x] \geq \frac{1}{em(m+1)} V(x)$$

proportional to the current distance

Upper bound on the expected running time:

$$\sum_{x \in X} \pi_0(x) \cdot \frac{1 + \log (V(x)/V_{min})}{\delta} \leq em(m+1)(1 + \log (mw_{max}))$$

$$\in O(m^2(\log n + \log w_{max}))$$

$V(x) \leq mw_{max}$ $\quad$ $V_{min} \geq 1$

Proof

Theorem. [Neumann & Wegener, TCS'07; Doerr et al., Algorithmica'12] The expected running time of the (1+1)-EA solving the MST problem is $O(m^2(\log n + \log w_{max}))$.

Main idea:

(1) obtain a connected subgraph

(2) obtain a minimum spanning tree

The expected running time of phase (1): $O(m \log n)$

The expected running time of phase (2): $O(m^2(\log n + \log w_{max}))$

The total expected running time: $O(m^2(\log n + \log w_{max}))$

Simplified drift analysis

The simplified drift analysis theorem
for proving exponential lower bounds [Oliveto & Witt, Algorithmica'11]

Theorem 2 (Simplified Drift Theorem) *Let X_t, $t \geq 0$, be the random variables describing a Markov process over the state space $S := \{0, 1, \ldots, N\}$ and denote $\Delta_t(i) := (X_{t+1} - X_t \mid X_t = i)$ for $i \in S$ and $t \geq 0$. Suppose there exist an interval $[a, b]$ of the state space (of asymptotically growing length $b - a$) and three constants $\delta, \varepsilon, r > 0$ such that for all $t \geq 0$*

1. $E(\Delta_t(i)) \geq \varepsilon$ *for* $a < i < b$
2. $\text{Prob}(\Delta_t(i) = -j) \leq 1/(1+\delta)^{j-r}$ *for* $i > a$ *and* $j \geq 1$

then there is a constant $c^* > 0$ such that for $T^* := \min\{t \geq 0 \colon X_t \leq a \mid X_0 \geq b\}$ it holds $\text{Prob}(T^* \leq 2^{c^*(b-a)}) = 2^{-\Omega(b-a)}$.

The expected drift is negative,
i.e., away from the target in expectation

The probability of a drift
towards the target
decreases exponentially

a constant

Exponential running time

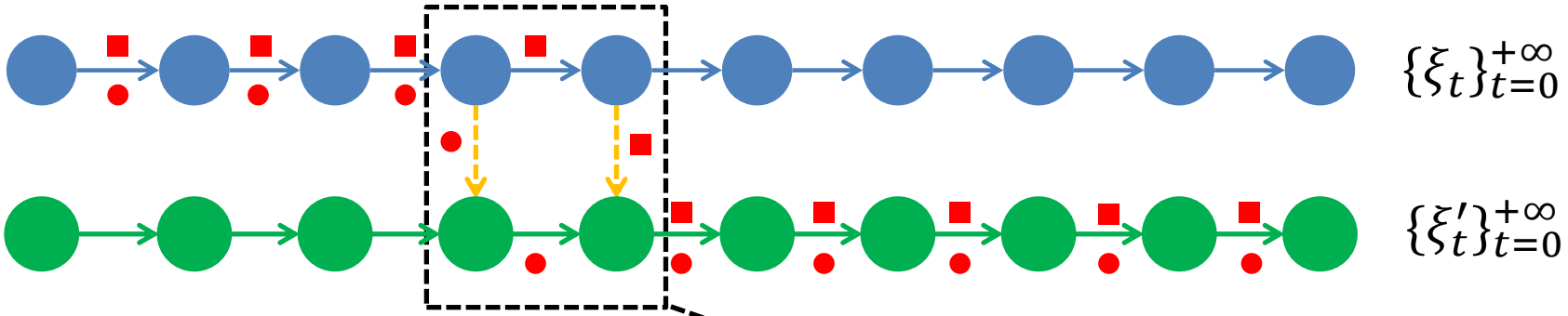
The simplified drift theorem with self-loops [Rowe & Sudholt, TCS'14]

The simplified drift theorem with scaling [Oliveto & Witt, TCS'14]

Switch analysis

The basic idea [Yu et al., TEC'15]:

Given EA on the given problem



Reference algorithm on the reference problem

investigate the different behaviors at each step

The expected running time:

$$\mathrm{E}[\tau] \leq (\geq)\ \mathrm{E}[\tau'] + \sum_{t=0}^{+\infty} \rho_t$$

The expected running time of $\{\xi'_t\}_{t=0}^{+\infty}$, easy to analyze

Multi-objective optimization

The task: optimize multiple objectives simultaneously

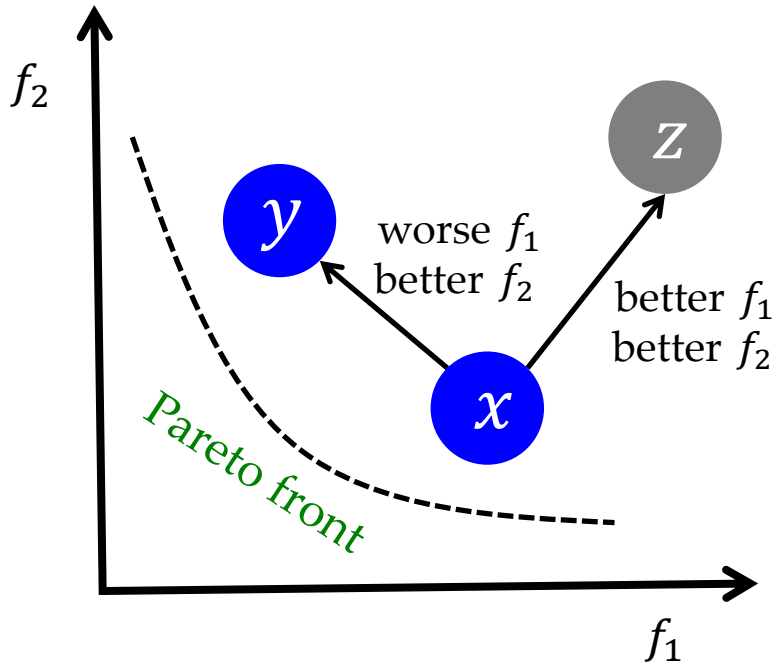
$$min_{x \in \mathrm{X}} \ (f_1(x), f_2(x), \ldots, f_m(x))$$

x dominates z:

$$f_1(x) < f_1(z) \ \wedge \ f_2(x) < f_2(z)$$

x incomparable y:

$$f_1(x) > f_1(y) \ \wedge \ f_2(x) < f_2(y)$$



Previous analysis approaches are not easy to be directly applied

Example: GSEMO for LOTZ

GSEMO: Given a pseudo-Boolean function vector $\boldsymbol{f}$:

1. $x :=$ randomly selected from $\{0,1\}^n$.
2. $P := \{x\}$.
3. Repeat until some termination criterion is met
4. Choose x from P uniformly at random.
5. $x' :=$ flip each bit of x with probability $1/n$.
6. if $\nexists\, z \in P$ such that $z \succ x'$
7. $P := (P - \{z \in P \mid x' \succcurlyeq z\}) \cup \{x'\}$.

Keep non-dominated solutions

LOTZ: $\arg max_{x\in\{0,1\}^n} \left(\sum_{i=1}^{n}\prod_{j=1}^{i}x_j,\ \sum_{i=1}^{n}\prod_{j=i}^{n}(1-x_j)\right)$

Count the number of leading 1-bits

Count the number of trailing 0-bits

The Pareto set: $00\ldots00,\ 10\ldots00,\ \ldots,\ 11\ldots10,\ 11\ldots11$.

The Pareto front: $(0,n),\ (1,n-1),\ \ldots,\ (n-1,1),\ (n,0)$.

Proof

Theorem. [Giel, CEC'03] The expected running time of the GSEMO solving the LOTZ problem is $O(n^3)$.

Main idea:

(1) obtain the Pareto optimal solution 11 ... 11

(2) obtain the Pareto front

The analysis of phase (1):

- select the solution with the largest LO value, and only flip its first 0 bit
- the probability: $\frac{1}{n+1}\frac{1}{n}(1-\frac{1}{n})^{n-1}$

the population size is not larger than $n+1$

Proof

Theorem. [Giel, CEC'03] The expected running time of the GSEMO solving the LOTZ problem is $O(n^3)$.

Main idea:

(1) obtain the Pareto optimal solution 11 ... 11

(2) obtain the Pareto front

The analysis of phase (1):

- select the solution with the largest LO value, and only flip its first 0 bit
- the probability: $\frac{1}{n+1}\frac{1}{n}(1-\frac{1}{n})^{n-1}$

Proof

Theorem. [Giel, CEC'03] The expected running time of the GSEMO solving the LOTZ problem is $O(n^3)$.

Main idea:

(1) obtain the Pareto optimal solution 11 ... 11

(2) obtain the Pareto front

The analysis of phase (1):

- select the solution with the largest LO value, and only flip its first 0 bit
- the probability: $\frac{1}{n+1}\frac{1}{n}(1-\frac{1}{n})^{n-1}$

the probability of increasing the largest LO value by 1 is at least $\frac{1}{en(n+1)}$

it is sufficient to increase n times

$\Rightarrow$ The expected running time:

$$n \cdot en(n+1) \in O(n^3)$$

Proof

The analysis of phase (2):

- the found Pareto optimal solutions will always be kept
- follow the path: $1^n \rightarrow 1^{n-1}0 \rightarrow \cdots \rightarrow 10^{n-1} \rightarrow 0^n$

the probability: $\frac{1}{n+1} \cdot \frac{1}{n}(1-\frac{1}{n})^{n-1}$

The expected running time: $n \cdot en(n+1) \in O(n^3)$

The expected running time of phase (1): $O(n^3)$

The total expected running time: $O(n^3)$

Running time analysis approaches

- Fitness level method
- Refined fitness level method
- Drift analysis
- Multiplicative drift analysis
- Switch analysis

Upper and lower bounds on the expected running time

- Simplified drift analysis

Exponential running time with high probability

More specific approaches for multi-objective EAs

Outline

- Introduction
- Running time analysis approaches
- **Running time analysis results**
- Examples of running time analysis leading to improved design of EAs
- Summary

Results in single-objective optimization

(1+1)-EA	linear function $\Theta(n \log n)$ [Droste et al., TCS'02] minimum spanning tree $O(m^2 \log(n + w_{max}))$ [Neumann & Wegener, TCS'07] partition $O(n^2)$ with $\frac{4}{3}$ approximation [Witt, STACS'05] vertex cover $e^{\Omega(n)}$ with arbitrary bad approximation [Oliveto, TEC'09]
(u+1)-EA	OneMax $O(un + n \log n)$; LeadingOnes $O(un \log n + n^2)$ [Witt, ECJ'06] maximum clique $O(un \log n)$ on sparse graphs [Storch, TCS'07] vertex cover $O(un \log n)$ on bipartite graphs [Oliveto, TEC'09]
(1+λ)-EA	linear function $O(\lambda n + n \log n)$ [Doerr & Kunnemann, TCS'15] vertex cover *exponential* on bipartite graphs [Oliveto, TEC'09]
(N+N)-EA	OneMax $O(Nn \log \log n + n \log n)$; LeadingOnes $O(Nn \log n + n^2)$ [Chen et al., TSMCB'09]

EDA [Chen et al., TEC'10]; ACO [Doerr et al., TCS'11]; PSO [Sudholt & Witt, TCS'10]; Memetic algorithms [Sudholt, TCS'09]; GP [Wagner et al., ECJ'15]

Results in multi-objective optimization

SEMO	LOTZ $\Theta(n^3)$; COCZ $O(n^2 \log n)$ [Laumanns et al., TEC'04]
	LOTZ $O(n^3)$ [Giel, CEC'03]; $\Omega(n^2/p)$ [Doerr et al., CEC'13]
GSEMO	bi-objective minimum spanning tree $O(m^3 w_{min}(\lvert C\rvert + \log n + \log w_{max}))$ with 2 approximation [Neumann, EJOR'07]
DEMO	multi-objective all-pairs-shortest-path $O(nP_{max}g)$ *with* $r^{3g \log n}$ approximation [Neumann & Theile, PPSN'10]
REMO	LOTZ $\Theta(n^2)$; COCZ $\Theta(n \log n)$; bi-objective minimum spanning tree $O\left(m^2 n w_{min}(\lvert C\rvert + \frac{\log n + \log w_{max}}{n w_{min}} - N_{gc}(1 - \frac{1}{m}))\right)$ with 2 approximation [Qian et al., AIJ'13]
MOEA/D	LOTZ $O(n^2 \log n)$; COCZ $\Theta(n \log n)$ [Li et al., TEC'16]

Single-objective optimization problems by multi-objective EAs

Single-objective better by multi-objective

Minimum spanning tree (MST):

- Given: an undirected connected graph $G = (V, E)$ on n vertices and m edges with positive integer weights $w: E \to \mathbb{N}$
- The Goal: find a connected subgraph $E' \subseteq E$ with the minimum weight

Fitness function: $\min \quad f(x) = (c(x) - 1) \cdot w_{ub} + \sum_{i:x_i=1} w_i$

$\Downarrow$

Bi-objective $\min \quad (c(x), \sum_{i:x_i=1} w_i)$

Single-objective: $O(m^2(\log n + \log w_{max}))$

Bi-objective: $O(mn(n + \log w_{max}))$

$\Rightarrow$ multi-objective better for dense graphs, e.g., $m = \Theta(n^2)$

More examples

Problem	Single-objective	Multi-objective
MST	$O(m^2(\log n + \log w_{max}))$	$O(mn(n + \log w_{max}))$ [Neumann & Wegener, GECCO'05]
Set cover	$exponential$	$O(mn(\log c_{max} + \log n))$ [Friedrich et al., ECJ'10]
Minimum cut	$exponential$	$O(Fm(\log c_{max} + \log n))$ [Neumann et al., Algorithmica'11]
Minimum LST	$\Omega(ku^k)$	$O(k^2 \log k)$ [Lai et al., TEC'14]
Minimum cost coverage	$exponential$	$O(Nn(\log n + \log w_{max} + N)$ [Qian et al., IJCAI'15]

Outline

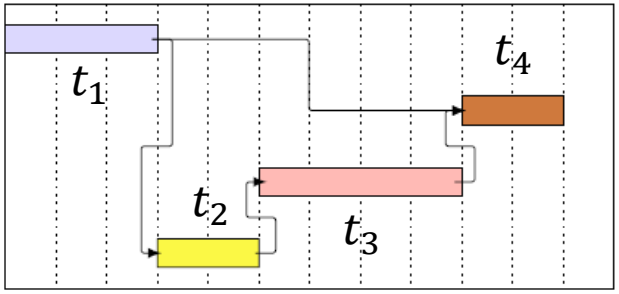
- Introduction
- Running time analysis approaches
- Running time analysis results
- **Examples of running time analysis leading to improved design of EAs**
- Summary

Project scheduling problem

A full software project scheduling process:

- identify project tasks
- identify task dependencies
- estimate resources for tasks
- allocate employees to tasks
- create project charts

To minimize the cost and completion time of the software project, while meeting all the constraints



Project Scheduling Problem

Project scheduling problem

The input:

- a set of employees $e_1, \dots, e_n$ with salaries $s_1, \dots, s_n$, and sets of skills $skill_1, \dots, skill_n$, respectively
- a set of tasks $t_1, \dots, t_m$ with required efforts $eff_1, \dots, eff_n$, and sets of required skills $req_1, \dots, req_m$, respectively
- a task precedence graph (TPG) – a directed graph with tasks as nodes and task precedence as edges

The objective: produce a schedule which minimizes the cost and time

$x_{i,j} \in \{0, \frac{1}{k}, \frac{2}{k}, \dots, 1\}$: the amount of dedication of e_i to t_j

Constraints:

- required skills: $req_j \subseteq \bigcup_{i=1}^{n} \{skill_i \mid x_{i,j} > 0\}$
- overwork: $max_{i,\tau} \{e_i^{work}(\tau)\} \leq 1$

the total dedication of employee i at time τ

Improved design inspired by theory

The overwork problem: $max_{i,\tau}\{e_i^{work}(\tau)\} \leq 1$

- Previous - repair

  If $M = max_{i,\tau}\{e_i^{work}(\tau)\} > 1, \quad x_{i,j} = x_{i,j}/M$

  divide dedications across the whole schedule

- Proposed – normalization [Minku et al., TSE'14]

  $x_{i,j} = x_{i,j}/\max(1, \sum_{t_l \in V'} x_{i,l})$

  divide dedications when necessary

Theoretical analysis:

Theorem 1. The expected running time of the (1+1)-EA with normalization on a linear schedule problem is $O((knm)^2)$.

Easy instances

Theorem 2. There exists a PSP instance where the (1+1)-EA with normalization needs at least exponential time.

Hard instances

Improved design inspired by theory

The proposed algorithm

[Minku et al., TSE'14]

Algorithm 3 Pop-EA for project scheduling

1: Initialise population P with μ candidate solutions.
2: **repeat**
3: Select λ parents from P using binary tournament selection.
4: **for** each pair of parents $x^{(1)}$ and $x^{(2)}$ **do**
5: With probability P_c, apply crossover between $x^{(1)}$ and $x^{(2)}$ to generate $x'^{(1)}$ and $x'^{(2)}$.
6: Otherwise, $x'^{(1)} \leftarrow x^{(1)}$ and $x'^{(2)} \leftarrow x^{(2)}$
7: Apply mutation to $x'^{(1)}$ and $x'^{(2)}$ using probability P_m.
8: $P \leftarrow P \cup \{x'^{(1)}, x'^{(2)}\}$.
9: **end for**
10: Select the μ best candidate solutions from P to survive for the next generation, based on the fitness function f.
11: **until** happy

The proposed Pop-EA performs the best

Experimental results

TABLE 8
Quality of Feasible Solution Measured by the Number of Employees (n) Multiplied by the Completion Time (time)

Benchmark 1: average $n \cdot$ time					
n	Pop-EA	(1+1) EA	RLS	(1+1) EA no-norm	GA [6]
5	**98.00**	98.04	98.19	113.96	109.40
10	**98.02**	98.06	98.17	129.68	112.70
15	**98.02**	98.07	98.22	128.06	115.95
20	**98.04**	98.08	98.19	129.54	117.60
Stdev	0.01	0.02	0.02	7.60	3.63

The averages were calculated considering only the runs in which a feasible solution was found The best values are in bold. Stdev is the standard deviation of the average values reported in the table.

TABLE 3
Average Ranking of Algorithms According to Fitness across Problem Instances

~ Avg Rank	Avg Rank	Std Deviation	Algorithm
1	1.1875	0.5708	Pop-EA
2	2.0833	0.3472	(1+1) EA
3	2.7292	0.6098	RLS
4	4.0000	0.0000	(1+1) EA no-norm

Smaller ranking values are better rankings.

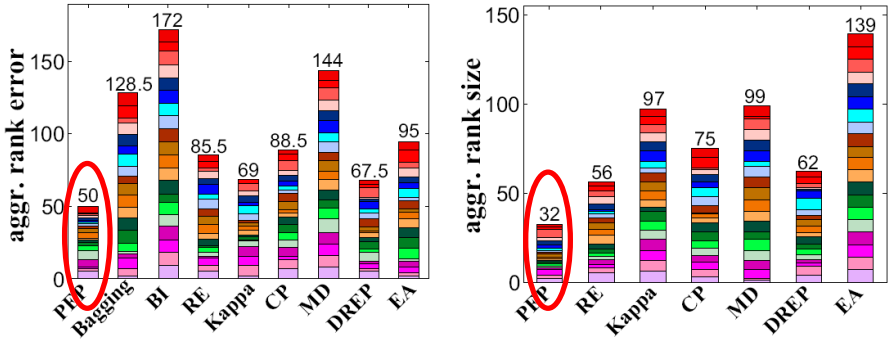
More examples

Theory: single-objective optimization can be solved better by multi-objective optimization

Ensemble pruning

[Qian et al., AAAI'15]

Theorem 1. *For any objective and any size, PEP within $O(n^4 \log n)$ expected optimization time can find a solution weakly dominating that generated by OEP at the fixed size.*



Subset selection

[Qian et al., NIPS'15; IJCAI'16]

Theorem 1. *For sparse regression, POSS with $E[T] \leq 2ek^2n$ and $I(\cdot) = 0$ (i.e., a constant function) finds a set S of variables with $|S| \leq k$ and $R^2_{Z,S} \geq (1 - e^{-\gamma_{\emptyset,k}}) \cdot OPT$.*

Data set	OPT	POSS	FR	FoBa	OMP	RFE	MCP
housing	.7437±.0297	.7437±.0297	.7429±.0300•	.7423±.0301•	.7415±.0300•	.7388±.0304•	.7354±.0297•
eunite2001	.8484±.0132	.8482±.0132	.8348±.0143•	.8442±.0144•	.8349±.0150•	.8424±.0153•	.8320±.0150•
svmguide3	.2705±.0255	.2701±.0257	.2615±.0260•	.2601±.0279•	.2557±.0270•	.2136±.0325•	.2397±.0237•
ionosphere	.5995±.0326	.5990±.0329	.5920±.0352•	.5929±.0346•	.5921±.0353•	.5832±.0415•	.5740±.0348•
sonar	–	.5365±.0410	.5171±.0440•	.5138±.0432•	.5112±.0425•	.4321±.0636•	.4496±.0482•
triazines	–	.4301±.0603	.4150±.0592•	.4107±.0600•	.4073±.0591•	.3615±.0712•	.3793±.0584•
coil2000	–	.0627±.0076	.0624±.0076•	.0619±.0075•	.0619±.0075•	.0363±.0141•	.0570±.0075•
mushrooms	–	.9912±.0020	.9909±.0021•	.9909±.0022•	.9909±.0022•	.6813±.1294•	.8652±.0474•
clean1	–	.4368±.0300	.4169±.0299•	.4145±.0309•	.4132±.0315•	.1596±.0562•	.3563±.0364•
w5a	–	.3376±.0267	.3319±.0247•	.3341±.0258•	.3313±.0246•	.3342±.0276•	.2694±.0385•
gisette	–	.7265±.0098	.7001±.0116•	.6747±.0145•	.6731±.0134•	.5360±.0318•	.5709±.0123•
farm-ads	–	.4217±.0100	.4196±.0101•	.4170±.0113•	.4170±.0113•	–	.3771±.0110•
POSS: win/tie/loss		–	12/0/0	12/0/0	12/0/0	11/0/0	12/0/0

Summary

- The theoretical foundation of EAs is weak, but important
- State-of-the-art running time analysis approaches
  - fitness level method
  - drift analysis
  - switch analysis
- Running time analysis results
  - single-objective optimization
  - multi-objective optimization
- Examples of improved design of EAs inspired by theoretical analysis

Schema theorem → Convergence → Running time

[Holland, 1975] [Rudolph, FI'98] [He & Yao, AIJ'01]

Running time analysis

Algorithm: Simple → Complex → • • •

Problem: Synthetic → Combinatorial → • • •

Single-objective → Multi-objective

Interesting directions

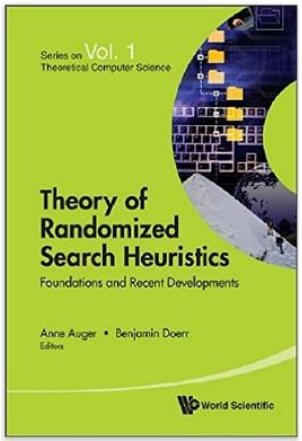
- Analysis on real EAs or real problems
- Running time analysis approaches for MOEAs
- Improved design of EAs by theory
- Analysis in noisy environments
- Analysis in continuous optimization
- ……

Reading books

Bioinspired Computation in Combinatorial Optimization

Algorithms and Their Computational Complexity

Authors: **Neumann**, Frank, **Witt**, Carsten



Theory of Randomized Search Heuristics

Foundations and Recent Developments

Edited by: **Anne Auger** (*INRIA, France*), **Benjamin Doerr** (*Max-Planck-Institut für Informatik, Germany*)

References

- B. Doerr, D. Johannsen and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 2012, 64: 673-697
- B. Doerr and L. A. Goldberg. Adaptive drift analysis. *Algorithmica*, 2013, 65: 224-250
- S. Droste, T. Jansen and I. Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 2002, 276(1-2): 51-81
- T. Friedrich, J. He, N. Hebbinghaus, F. Neumann and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. *Evolutionary Computation*, 2010, 18(4): 617-633
- J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 2001, 127(1): 57-85
- X. Lai, Y. Zhou, J. He and J. Zhang. Performance analysis of evolutionary algorithms for the minimum label spanning tree problem. *IEEE Transactions on Evolutionary Computation*, 2014, 18(6): 860-872.

References

- M. Laumanns, L. Thiele and E. Zitzler. Running time analysis of multiobjective evolutionary algorithms on pseudo-Boolean functions. *IEEE Transactions on Evolutionary Computation*, 2004, 8(2): 170-182

- Y. Li, Y. Zhou, Z.-H. Zhan and J. Zhang. A primary theoretical study on decomposition based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 2016, in press

- L. L. Minku, D. Sudholt and X. Yao. Improved evolutionary algorithm design for the project scheduling problem based on runtime analysis. *IEEE Transactions on Software Engineering*, 2014, 40(1): 83-102

- F. Neumann and I. Wegener. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing*, 2006, 5(3): 305-319

- F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 2007, 378(1): 32-40

- F. Neumann and M. Theile. How crossover speeds up evolutionary algorithms for the multi-criteria all-pairs-shortest-path problem. In: *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10)*, 2010, pages 667-676, Krakow, Poland

References

- F. Neumann, J. Reichel and M. Skutella. Computing minimum cuts by randomized search heuristics. *Algorithmica*, 2011, 59(3): 323-342

- P. S. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. *Algorithmica*, 2011, 59(3): 369-386

- P. S. Oliveto and C. Witt. On the runtime analysis of the simple genetic algorithm. *Theoretical Computer Science*, 2014, 545: 2-19

- C. Qian, Y. Yu and Z.-H. Zhou. An analysis on recombination in multi-objective evolutionary optimization. *Artificial Intelligence*, 2013, 204: 99-119

- C. Qian, Y. Yu and Z.-H. Zhou. Pareto ensemble pruning. In: *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*, 2015, pages 2935-2941, Austin, TX

- C. Qian, Y. Yu and Z.-H. Zhou. On constrained Boolean Pareto optimization. In: *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*, 2015, pages 389-395, Buenos Aires, Argentina

References

- C. Qian, Y. Yu and Z.-H. Zhou. Subset selection by Pareto optimization. In: *Advances in Neural Information Processing Systems 28 (NIPS'15)*, 2015, pages 1765-1773, Montreal, Canada

- C. Qian, J.-C. Shi, Y. Yu, K. Tang and Z.-H. Zhou. Parallel Pareto optimization for subset selection. In: *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, 2016, New York, NY

- J. E. Rowe and D. Sudholt. The choice of the offspring population size in the $(1,\lambda)$ evolutionary algorithm. *Theoretical Computer Science*, 2014, 545: 20-38

- Y. Yu, X. Yao and Z.-H. Zhou. On the approximation ability of evolutionary optimization with application to minimum set cover. *Artificial Intelligence*, 2012, 180-181: 20-33

- Y. Yu, C. Qian and Z.-H. Zhou. Switch analysis for running time analysis of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 2015, 19(6): 777-792

- Y. Yu and C. Qian. Running time analysis: Convergence-based analysis reduces to switch analysis. In *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC'15)*, 2015, pages 2603-2610, Sendai, Japan