

# Last class

---

- Schema theorem
- Markov chain modeling
- Convergence
- Running time complexity
- Expectation and tail inequalities
- Example of running time analysis



南京大学  
人工智能学院

SCHOOL OF ARTIFICIAL INTELLIGENCE, NANJING UNIVERSITY



# Heuristic Search and Evolutionary Algorithms

## Lecture 10: Running Time Analysis of EAs

Chao Qian (钱超)

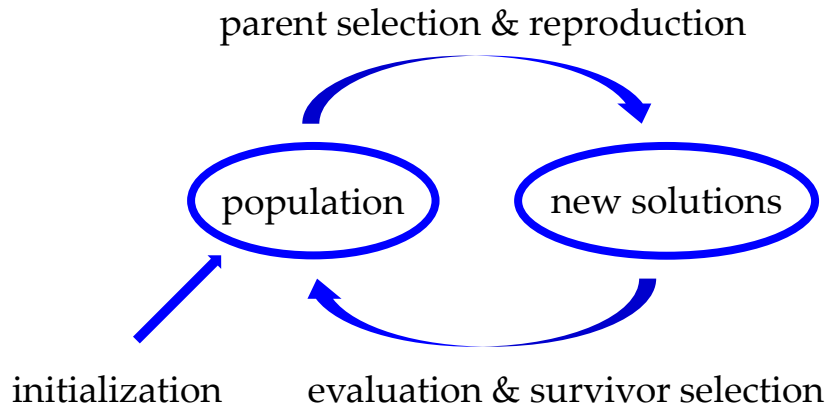
Associate Professor, Nanjing University, China

Email: [qianc@nju.edu.cn](mailto:qianc@nju.edu.cn)

Homepage: <http://www.lamda.nju.edu.cn/qianc/>

# Markov chain modeling

EA:

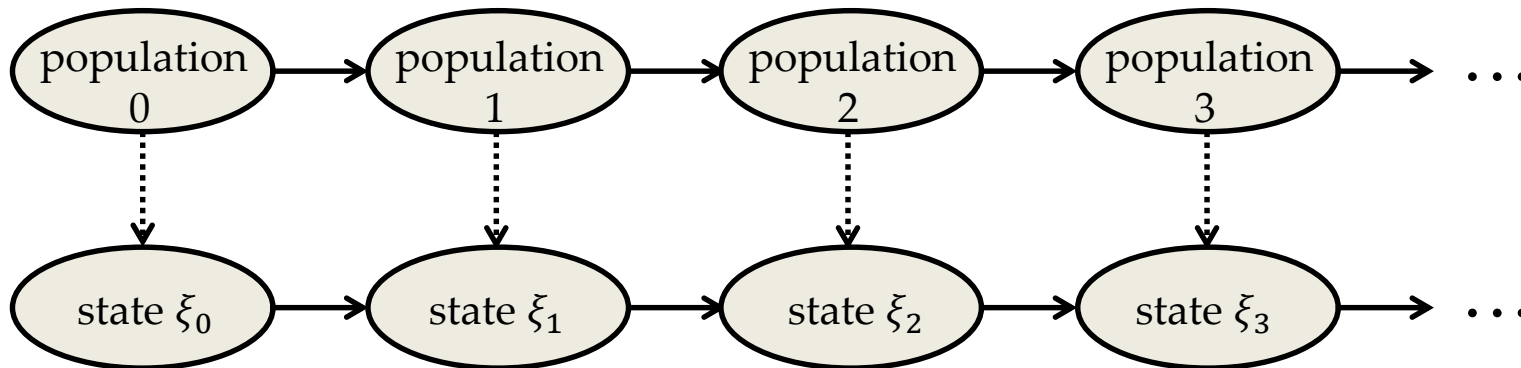


Running time analysis

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

a random variable

- $E[\tau]$
- $P(\tau \leq T)$



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

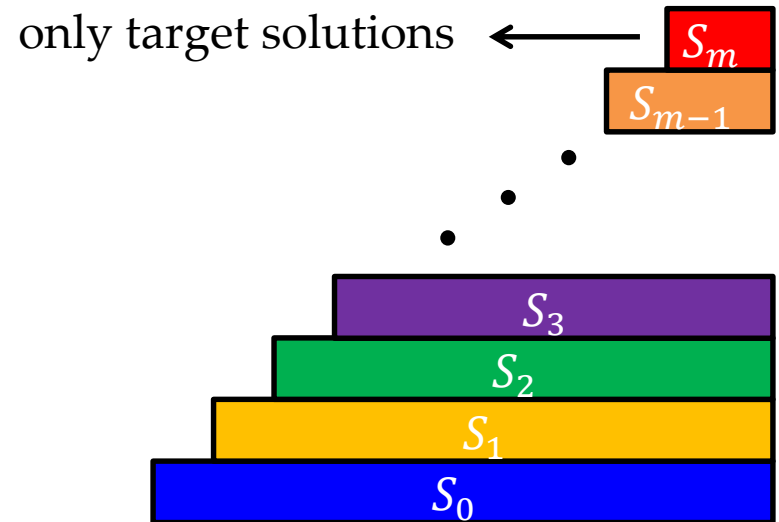
# Fitness level method

---

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

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

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



# Fitness level method

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

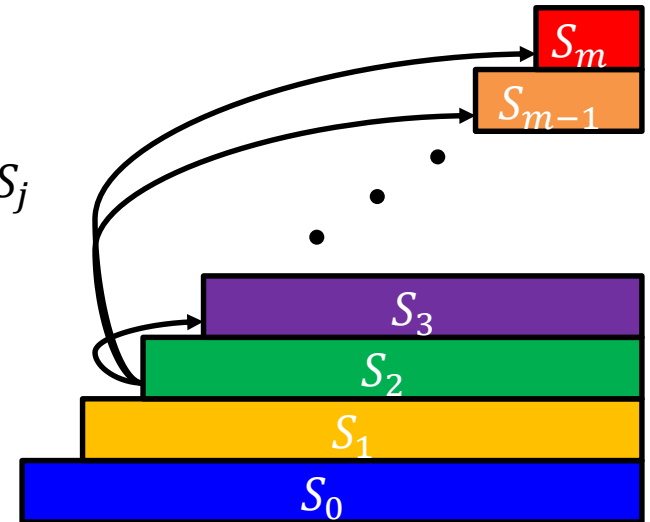
1. Divide the solution space  $S$  into  $m + 1$  subspaces  $S_0, S_1, \dots, S_m$

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

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

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

The best solution in  $\xi_t$   
belongs to  $S_i$



# Fitness level method

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

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

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

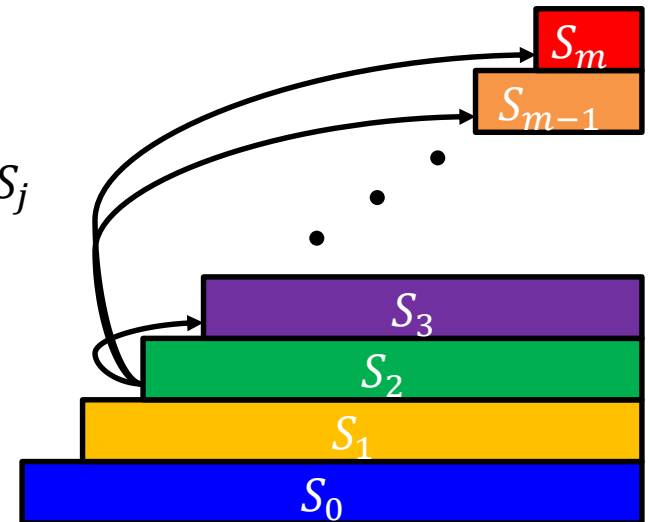
2. Bounds on the probability of leaving  $S_i$  to higher  $S_j$

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

### Expected running time

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

↓  
the initial distribution



# Fitness level method

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

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

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

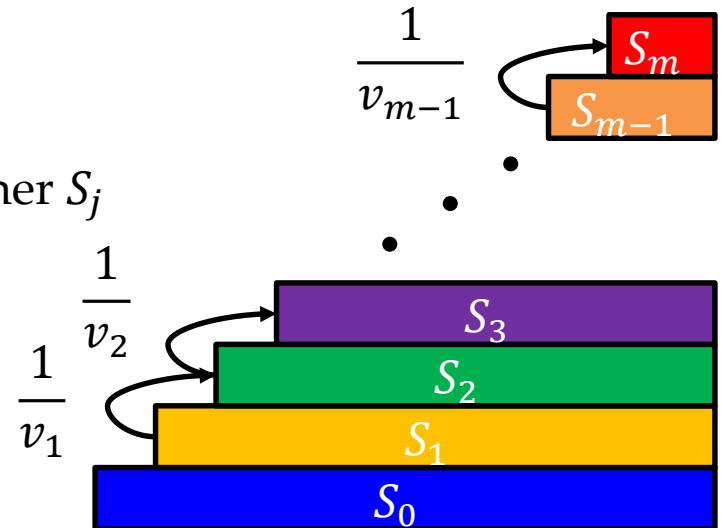
2. Bounds on the probability of leaving  $S_i$  to higher  $S_j$

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

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  $S_0, S_1, \dots, S_m$

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

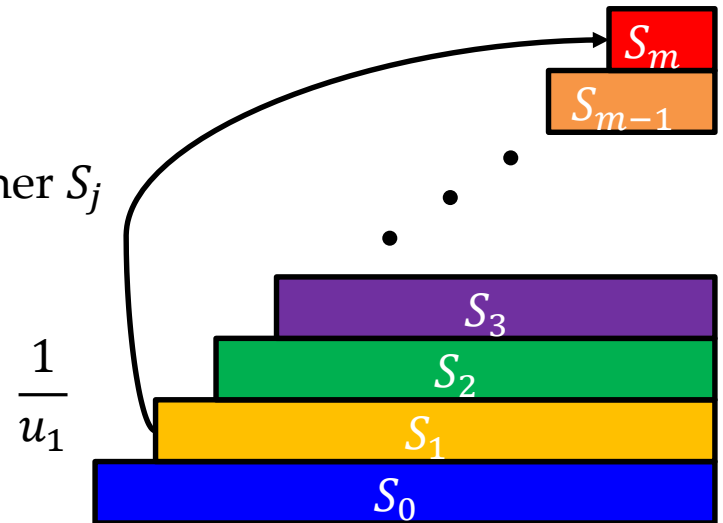
2. Bounds on the probability of leaving  $S_i$  to higher  $S_j$

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

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





# Application illustration: (1+1)-EA for OneMax

---

## (1+1)-EA:

Given a pseudo-Boolean function  $f$ :

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

Bit-wise mutation

## OneMax:

$$\max_{\mathbf{x} \in \{0,1\}^n} \left( \sum_{i=1}^n x_i \right) \longrightarrow \text{Count the number of 1-bits}$$

**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:

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

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

flip one of the  $n - i$  0-bits
keep the other bits unchanged

$$v_i = \frac{n-i}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1}$$

Upper bound on the expected running time:

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

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

# Refined fitness level method

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

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

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]:

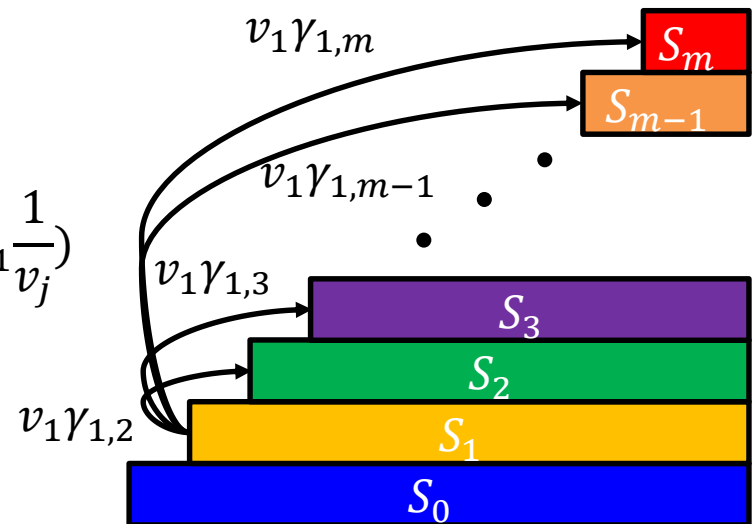
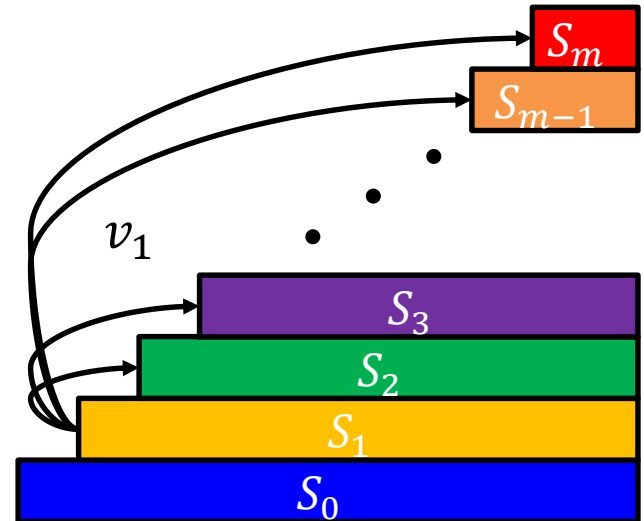
$$P(\xi_{t+1} \in S_j | \xi_t \in S_i) \geq v_i \cdot \gamma_{i,j}$$

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

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** with  $\chi = 1$

Similar for lower bound analysis



# Refined fitness level method

## The above two fitness level methods

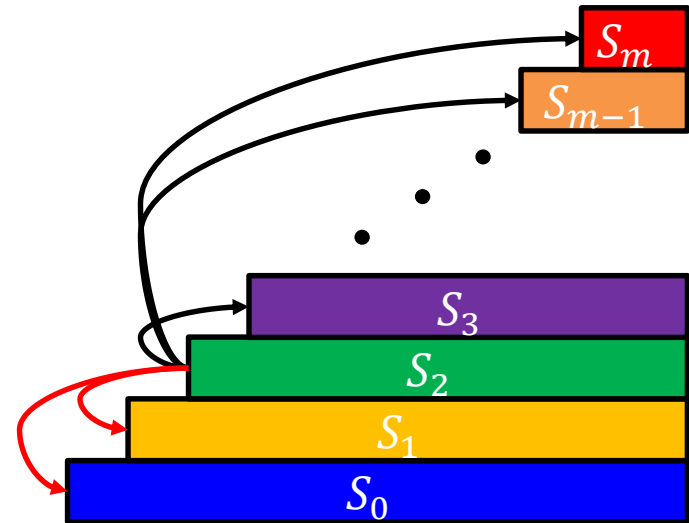
[Droste et al., TCS'02; Sudholt, TEC'13]

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

## Fitness level method for non-elitist EAs

[Dang & Lehre, Algorithmica'16]

- allow jumping to lower levels



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

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

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

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

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

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

# Additive drift analysis

## The basic idea [He & Yao, AIJ'01]:

1. Design a distance function  $V(x): \mathcal{X} \rightarrow \mathbb{R}$  to measure the distance from a state  $x \in \mathcal{X}$  to the target state space  $\mathcal{X}^*$

$$\bullet \forall x \in \mathcal{X} \setminus \mathcal{X}^*: V(x) > 0 \quad \bullet \forall x \in \mathcal{X}^*: V(x) = 0$$

2. Bounds on the expected drift in one step

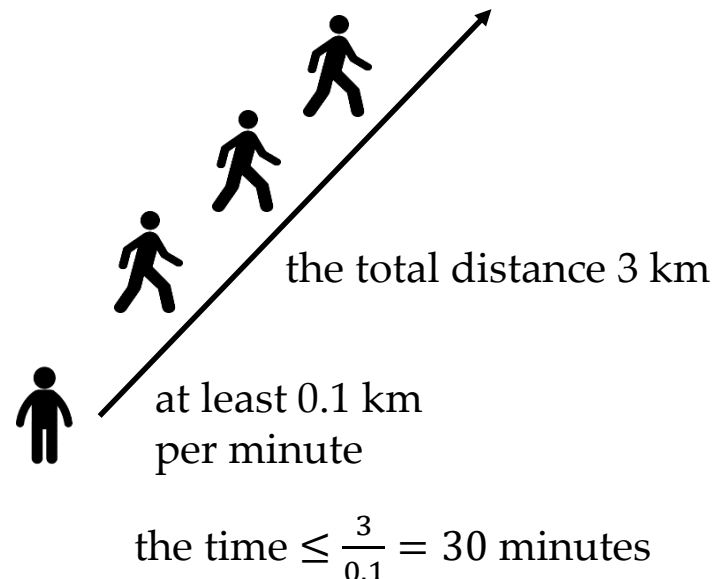
$$\bullet E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq c_l$$
$$\bullet E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \leq c_u$$

Expected running time

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

the initial distribution

The hotel



# Additive drift analysis

## The basic idea [He & Yao, AIJ'01]:

1. Design a distance function  $V(x): \mathcal{X} \rightarrow \mathbb{R}$  to measure the distance from a state  $x \in \mathcal{X}$  to the target state space  $\mathcal{X}^*$

$$\bullet \forall x \in \mathcal{X} \setminus \mathcal{X}^*: V(x) > 0 \quad \bullet \forall x \in \mathcal{X}^*: V(x) = 0$$

2. Bounds on the expected drift in one step

$$\bullet E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq c_l$$

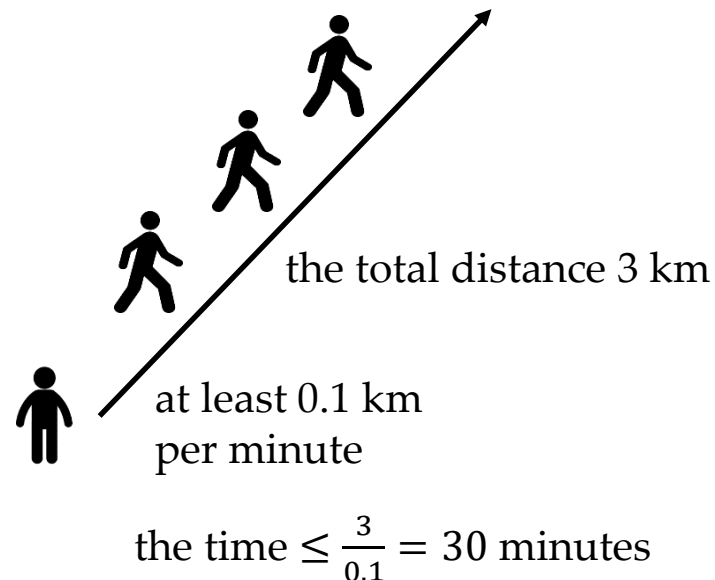
$$\bullet E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \leq c_u$$

### Expected running time

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

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

The hotel



# Application illustration: (1+1)-EA for LeadingOnes

---

## (1+1)-EA:

Given a pseudo-Boolean function  $f$ :

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

## LeadingOnes:

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

$LO(\mathbf{x})$   
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  $LO(x) = i$ :

$$\begin{aligned} & \mathbb{E}[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = x] \\ &= \mathbb{E}[LO(\xi_{t+1}) - i \mid \xi_t = x] \geq 1 \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^i \end{aligned}$$

$LO(x) = i \rightarrow \geq i + 1$  flip the first 0-bit keep the  $i$  leading 1-bits unchanged



# 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

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

keep the  $i$  leading 1-bits unchanged

$LO(x) = i \rightarrow \geq i + 1$       flip the first 0-bit       $c_l = \frac{1}{en}$

Upper bound on the expected running time:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{V(x)}{c_l} \leq \frac{n}{c_l} = n / \left(\frac{1}{en}\right) = en^2 \in O(n^2)$$

# Multiplicative drift analysis

**Additive** [He & Yao, AIJ'01]:

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

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

The hotel

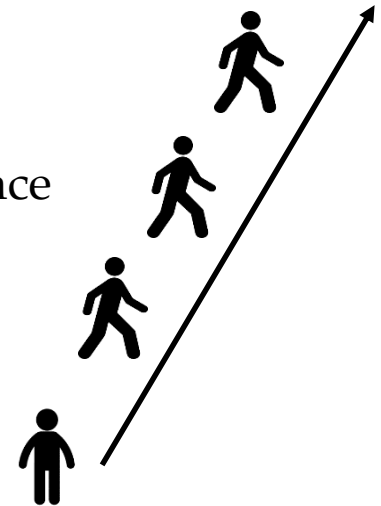


**Multiplicative** [Doerr et al., Algorithmica'12]:

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

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

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



# Multiplicative vs. Additive

---

For additive and multiplicative drift analysis, which one is stronger?

**Theorem.** [Doerr et al., *Algorithmica*'12] Multiplicative drift analysis can be proved by using **additive drift analysis**. **Stronger**

Multiplicative drift analysis:

$$E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq \delta \cdot V(\xi_t)$$

Construct a distance function  $U(x)$

$$E[U(\xi_t) - U(\xi_{t+1}) \mid \xi_t] \geq c_l \rightarrow \text{not depend on } \xi_t$$

Upper bound:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{1 + \ln(V(x)/V_{min})}{\delta}$$

Apply additive drift analysis

# Proof

---

Multiplicative drift analysis:

$$E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq \delta \cdot V(\xi_t)$$



Upper bound:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{1 + \ln(V(x)/V_{min})}{\delta}$$

Main idea:

- Design the distance function  $U(x) = \begin{cases} 0 & \text{if } V(x) = 0 \\ 1 + \ln(V(x)/V_{min}) & \text{Otherwise} \end{cases}$
- The drift from a state  $\xi_t \notin \mathcal{X}^*$  (i.e.,  $V(\xi_t) > 0$  or  $U(\xi_t) > 0$ ):

$$\text{If } \xi_{t+1} \in \mathcal{X}^*, \text{ then } U(\xi_t) - U(\xi_{t+1}) = 1 + \ln(V(\xi_t)/V_{min}) \geq 1 = \frac{V(\xi_t) - V(\xi_{t+1})}{V(\xi_t)}$$

$$\text{If } \xi_{t+1} \notin \mathcal{X}^*, \text{ then } U(\xi_t) - U(\xi_{t+1}) = \ln(V(\xi_t)/V(\xi_{t+1})) \geq \frac{V(\xi_t) - V(\xi_{t+1})}{V(\xi_t)}$$

# Proof

---

$$\ln(V(\xi_t)/V(\xi_{t+1})) \geq \frac{V(\xi_t) - V(\xi_{t+1})}{V(\xi_t)}$$

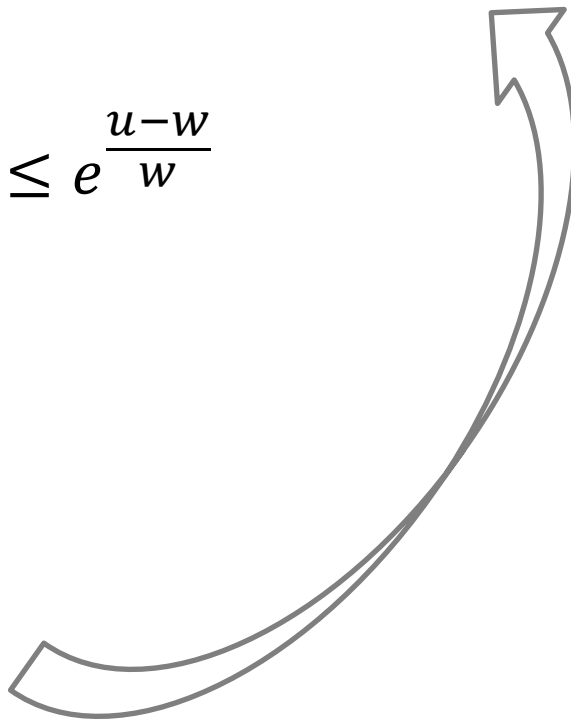
$$\frac{u}{w} = 1 + \frac{u-w}{w} \leq e^{\frac{u-w}{w}}$$



$$\ln \frac{u}{w} \leq \frac{u-w}{w}$$



$$\ln \frac{w}{u} \geq \frac{w-u}{w}$$

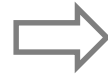


# Proof

---

Multiplicative drift analysis:

$$E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq \delta \cdot V(\xi_t)$$



Upper bound:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{1 + \ln(V(x)/V_{min})}{\delta}$$

Main idea:

- Design the distance function  $U(x) = \begin{cases} 0 & \text{if } V(x) = 0 \\ 1 + \ln(V(x)/V_{min}) & \text{Otherwise} \end{cases}$

- The drift from a state  $\xi_t \notin \mathcal{X}^*$  (i.e.,  $V(\xi_t) > 0$  or  $U(\xi_t) > 0$ ):

$$U(\xi_t) - U(\xi_{t+1}) \geq \frac{V(\xi_t) - V(\xi_{t+1})}{V(\xi_t)}$$

- The expected drift from a state  $\xi_t \notin \mathcal{X}^*$  (i.e.,  $V(\xi_t) > 0$  or  $U(\xi_t) > 0$ ):

$$E[U(\xi_t) - U(\xi_{t+1}) \mid \xi_t] \geq \frac{E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t]}{V(\xi_t)}$$

# Proof

Multiplicative drift analysis:

$$E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \geq \delta \cdot V(\xi_t)$$



Upper bound:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{1 + \ln(V(x)/V_{min})}{\delta}$$

Main idea:

- Design the distance function  $U(x) = \begin{cases} 0 & \text{if } V(x) = 0 \\ 1 + \ln(V(x)/V_{min}) & \text{Otherwise} \end{cases}$
- The expected drift from a state  $\xi_t \notin \mathcal{X}^*$  (i.e.,  $V(\xi_t) > 0$  or  $U(\xi_t) > 0$ ):

$$E[U(\xi_t) - U(\xi_{t+1}) \mid \xi_t] \geq \frac{E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t]}{V(\xi_t)} \geq \delta$$


Additive drift analysis [He & Yao, AIJ'01]:

$$E[U(\xi_t) - U(\xi_{t+1}) \mid \xi_t] \geq c_l \quad \text{Upper bound: } \sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{U(x)}{c_l}$$

# Multiplicative vs. Additive

---

For additive and multiplicative drift analysis, which one is stronger?

**Theorem.** [Doerr et al., *Algorithmica*'12] Multiplicative drift analysis can be proved by using additive drift analysis.  **Stronger**

Multiplicative drift analysis can be easier to be used when the expected drift under a natural distance function is proportional to the current distance

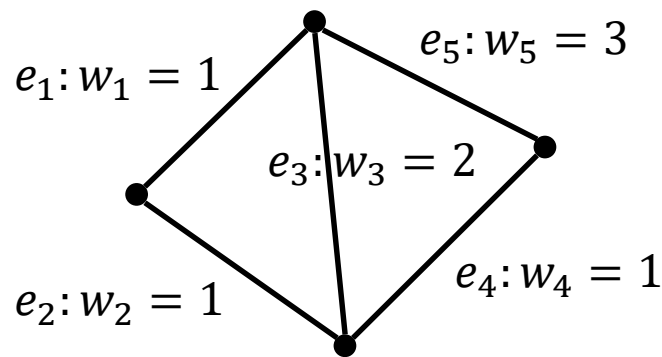


# Application illustration: (1+1)-EA for MST

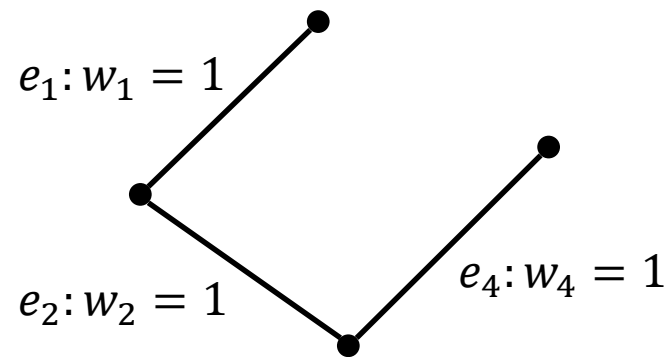
---

## Minimum spanning tree (MST):

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



The original graph



The minimum spanning tree

# Application illustration: (1+1)-EA for MST

**(1+1)-EA:** Given a pseudo-Boolean function  $f$ :

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

**Solution representation:**  $\mathbf{x} \in \{0,1\}^m \leftrightarrow$  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 maximum  
edge weight

$\min$

$f(\mathbf{x}) =$

$(c(\mathbf{x}) - 1)$

$\cdot w_{ub} +$

$\sum_{i:x_i=1} w_i$

the sum of  
edge weights

$w_{ub} = n^2 \cdot w_{max}$

to make a subgraph with less connected components better

the number of  
connected components

# 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  $\longrightarrow c(\mathbf{x}) = 1$
- (2) obtain a minimum spanning tree

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

- $c(\mathbf{x})$  cannot increase

• at least  $c(\mathbf{x}) - 1$  edges, the insertion of which can decrease  $c(\mathbf{x})$  by 1

• the probability of decreasing  $c(\mathbf{x})$  by 1 is at least  $\frac{c(\mathbf{x})-1}{m} \left(1 - \frac{1}{m}\right)^{m-1}$

flip one of those  $c(\mathbf{x}) - 1$  0-bits keep the other bits unchanged

# 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  $\longrightarrow c(\mathbf{x}) = 1$
- (2) obtain a minimum spanning tree

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

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

the probability of decreasing  $c(\mathbf{x})$  by 1 is at least  $\frac{c(\mathbf{x})-1}{m} \left(1 - \frac{1}{m}\right)^{m-1}$

fitness level method  $\implies$  The expected running time upper bound:

$$\sum_{c(\mathbf{x})=n}^2 \frac{em}{c(\mathbf{x}) - 1} \in O(m \log n)$$

# Proof

---

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

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

Using multiplicative drift analysis:

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

$$\begin{aligned} E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] &= V(\mathbf{x}) - E[V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] = f(\mathbf{x}) - E[f(\xi_{t+1}) \mid \xi_t = \mathbf{x}] \\ &\geq f(\mathbf{x}) - \left( \sum_{i=1}^{m-(n-1)} f(\mathbf{y}^i) \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^n f(\mathbf{z}^i) \cdot \frac{1}{m^2} \left(1 - \frac{1}{m}\right)^{m-2} + (1 - \dots) f(\mathbf{x}) \right) \end{aligned}$$

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  $(f(\mathbf{x}) - f_{opt}) / (m + 1)$

# Proof

---

- to analyze  $f(\mathbf{x}) \rightarrow f_{opt}$  the weight of a minimum spanning tree

Using multiplicative drift analysis:

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

$$\begin{aligned} E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] &= V(\mathbf{x}) - E[V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] = f(\mathbf{x}) - E[f(\xi_{t+1}) \mid \xi_t = \mathbf{x}] \\ &\geq f(\mathbf{x}) - \left( \sum_{i=1}^{m-(n-1)} f(\mathbf{y}^i) \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^n f(\mathbf{z}^i) \cdot \frac{1}{m^2} \left(1 - \frac{1}{m}\right)^{m-2} + (1 - \dots) f(\mathbf{x}) \right) \end{aligned}$$

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  $(f(\mathbf{x}) - f_{opt}) / (m + 1)$

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

# Proof

---

- to analyze  $f(\mathbf{x}) \rightarrow f_{opt}$  the weight of a minimum spanning tree

Using multiplicative drift analysis:

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

$$\begin{aligned} \mathbb{E}[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] &= V(\mathbf{x}) - \mathbb{E}[V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] = f(\mathbf{x}) - \mathbb{E}[f(\xi_{t+1}) \mid \xi_t = \mathbf{x}] \\ &\geq f(\mathbf{x}) - \left( \sum_{i=1}^{m-(n-1)} f(\mathbf{y}^i) \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^n f(\mathbf{z}^i) \cdot \frac{1}{m^2} \left(1 - \frac{1}{m}\right)^{m-2} + (1 - \dots) f(\mathbf{x}) \right) \end{aligned}$$

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  $(f(\mathbf{x}) - f_{opt}) / (m + 1)$

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

# Proof

---

- to analyze  $f(\mathbf{x}) \rightarrow f_{opt}$  the weight of a minimum spanning tree

Using multiplicative drift analysis:

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

$$\begin{aligned} \mathbb{E}[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] &= V(\mathbf{x}) - \mathbb{E}[V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] = f(\mathbf{x}) - \mathbb{E}[f(\xi_{t+1}) \mid \xi_t = \mathbf{x}] \\ &\geq f(\mathbf{x}) - \left( \sum_{i=1}^{m-(n-1)} f(\mathbf{y}^i) \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^n f(\mathbf{z}^i) \cdot \frac{1}{m^2} \left(1 - \frac{1}{m}\right)^{m-2} + (1 - \dots) f(\mathbf{x}) \right) \end{aligned}$$

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  $(f(\mathbf{x}) - f_{opt}) / (m + 1)$

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



# Proof

---

- to analyze  $f(\mathbf{x}) \rightarrow f_{opt}$  the weight of a minimum spanning tree

Using multiplicative drift analysis:

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

$$\begin{aligned} \mathbb{E}[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] &= V(\mathbf{x}) - \mathbb{E}[V(\xi_{t+1}) \mid \xi_t = \mathbf{x}] = f(\mathbf{x}) - \mathbb{E}[f(\xi_{t+1}) \mid \xi_t = \mathbf{x}] \\ &\geq f(\mathbf{x}) - \left( \sum_{i=1}^{m-(n-1)} f(\mathbf{y}^i) \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right)^{m-1} + \sum_{i=1}^n f(\mathbf{z}^i) \cdot \frac{1}{m^2} \left(1 - \frac{1}{m}\right)^{m-2} + (1 - \dots) f(\mathbf{x}) \right) \end{aligned}$$

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  $(f(\mathbf{x}) - f_{opt}) / (m + 1)$

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

# 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:

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

proportional to  
the current distance

Upper bound on the expected running time:

$$\sum_{x \in \mathcal{X}} \pi_0(x) \cdot \frac{1 + \ln(V(x)/V_{min})}{V(x)} \leq em(m+1)(1 + \ln(mw_{max})) \in O(m^2(\log n + \log w_{max}))$$

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

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

# Negative drift

## Simplified negative drift theorem for proving exponential lower bounds [Oliveto & Witt, Algorithmica'11]

**Theorem 4** (Simplified Drift Theorem) *Let  $X_t, t \geq 0$ , be the random variables describing a Markov process over a finite state space  $S \subseteq \mathbb{R}_0^+$  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  $[a, b]$  in the state space, two constants  $\delta, \varepsilon > 0$  and, possibly depending on  $\ell := b - a$ , a function  $r(\ell)$  satisfying  $1 \leq r(\ell) = o(\ell/\log(\ell))$  such that for all  $t \geq 0$  the following two conditions hold:*

1.  $E(\Delta_t(i)) \geq \varepsilon$  for  $a < i < b$ ,
2.  $\text{Prob}(\Delta_t(i) \leq -j) \leq \frac{r(\ell)}{(1+\delta)^j}$  for  $i > a$  and  $j \in \mathbb{N}_0$ .

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

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

The probability of a drift  
towards the target  
decreases exponentially

Exponential running time

negative drift



very unlikely



# Negative drift

---

## Simplified negative drift theorem for proving exponential lower bounds [Oliveto & Witt, Algorithmica'11]

**Theorem 4** (Simplified Drift Theorem) *Let  $X_t, t \geq 0$ , be the random variables describing a Markov process over a finite state space  $S \subseteq \mathbb{R}_0^+$  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  $[a, b]$  in the state space, two constants  $\delta, \varepsilon > 0$  and, possibly depending on  $\ell := b - a$ , a function  $r(\ell)$  satisfying  $1 \leq r(\ell) = o(\ell/\log(\ell))$  such that for all  $t \geq 0$  the following two conditions hold:*

1.  $E(\Delta_t(i)) \geq \varepsilon$  for  $a < i < b$ ,
2.  $\text{Prob}(\Delta_t(i) \leq -j) \leq \frac{r(\ell)}{(1+\delta)^j}$  for  $i > a$  and  $j \in \mathbb{N}_0$ .

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

a constant

relax

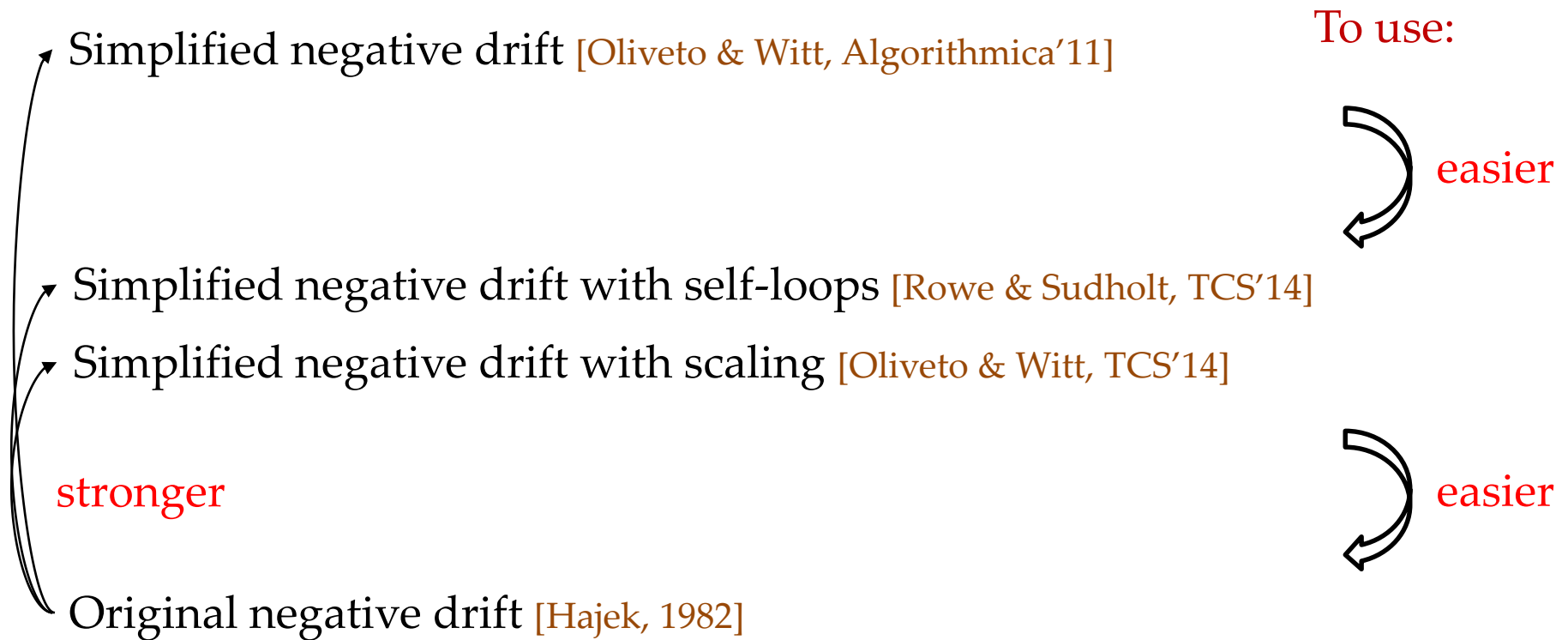
Simplified negative drift with self-loops [Rowe & Sudholt, TCS'14]

Simplified negative drift with scaling [Oliveto & Witt, TCS'14]

# Negative drift

---

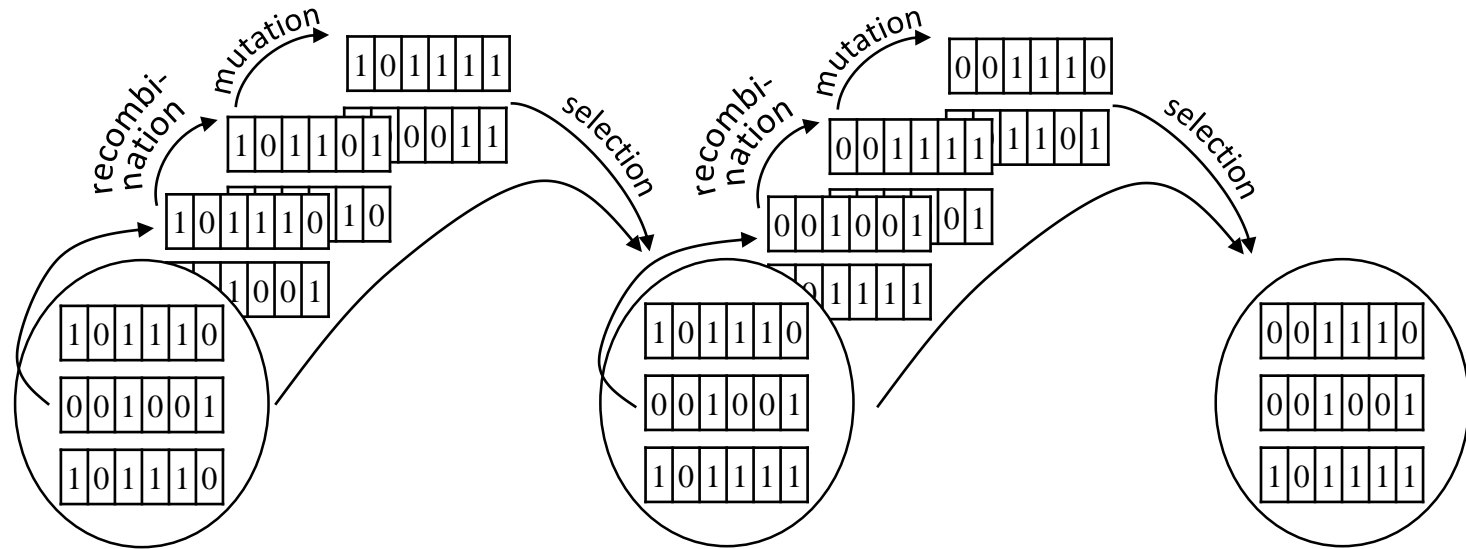
For proving exponential lower bounds on the running time of EAs:



# Switch analysis

Hard to be analyzed directly

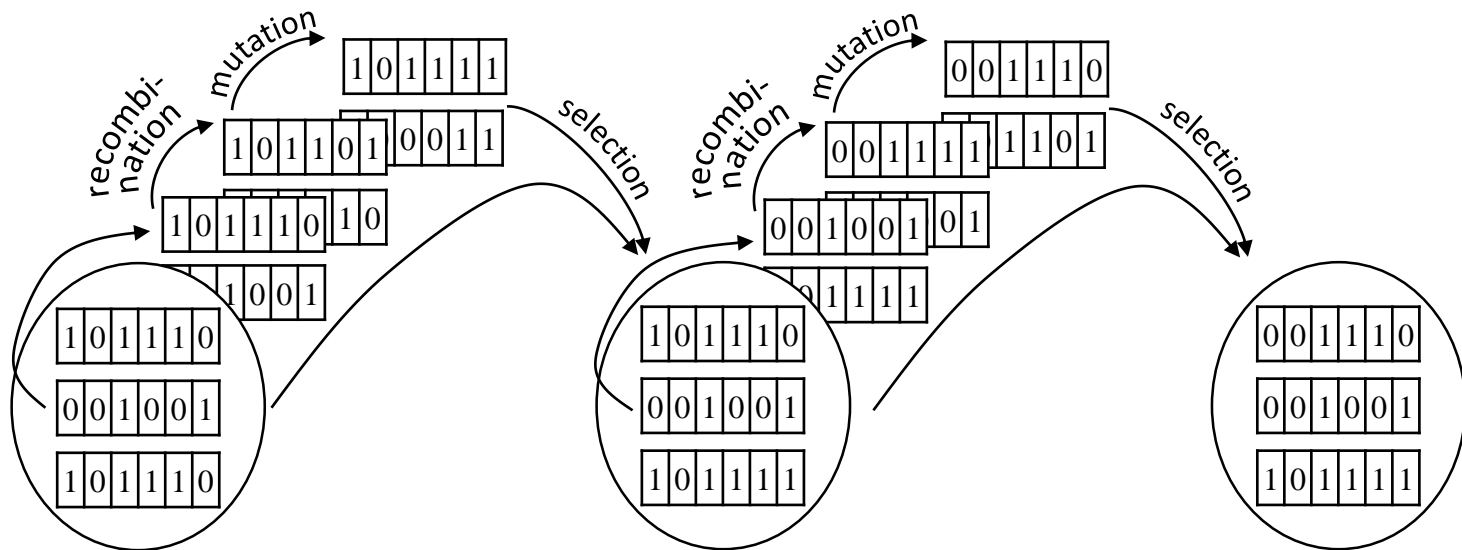
EA with recombination



# Switch analysis

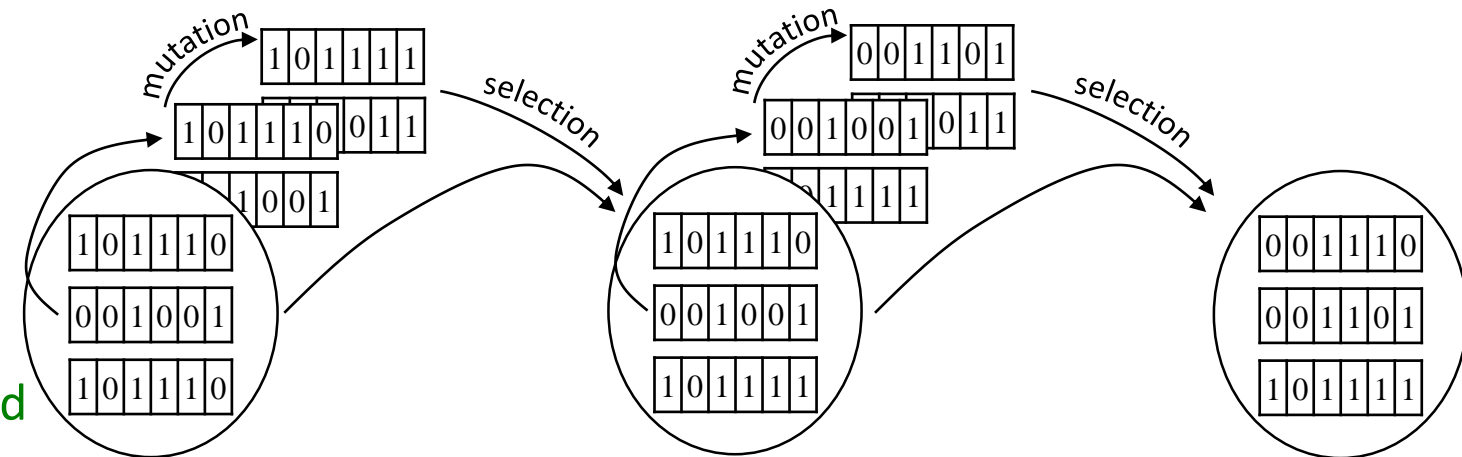
Hard to be analyzed directly

EA with recombination



EA without recombination

Easier to be analyzed





# Switch analysis

**Simplify the analysis**

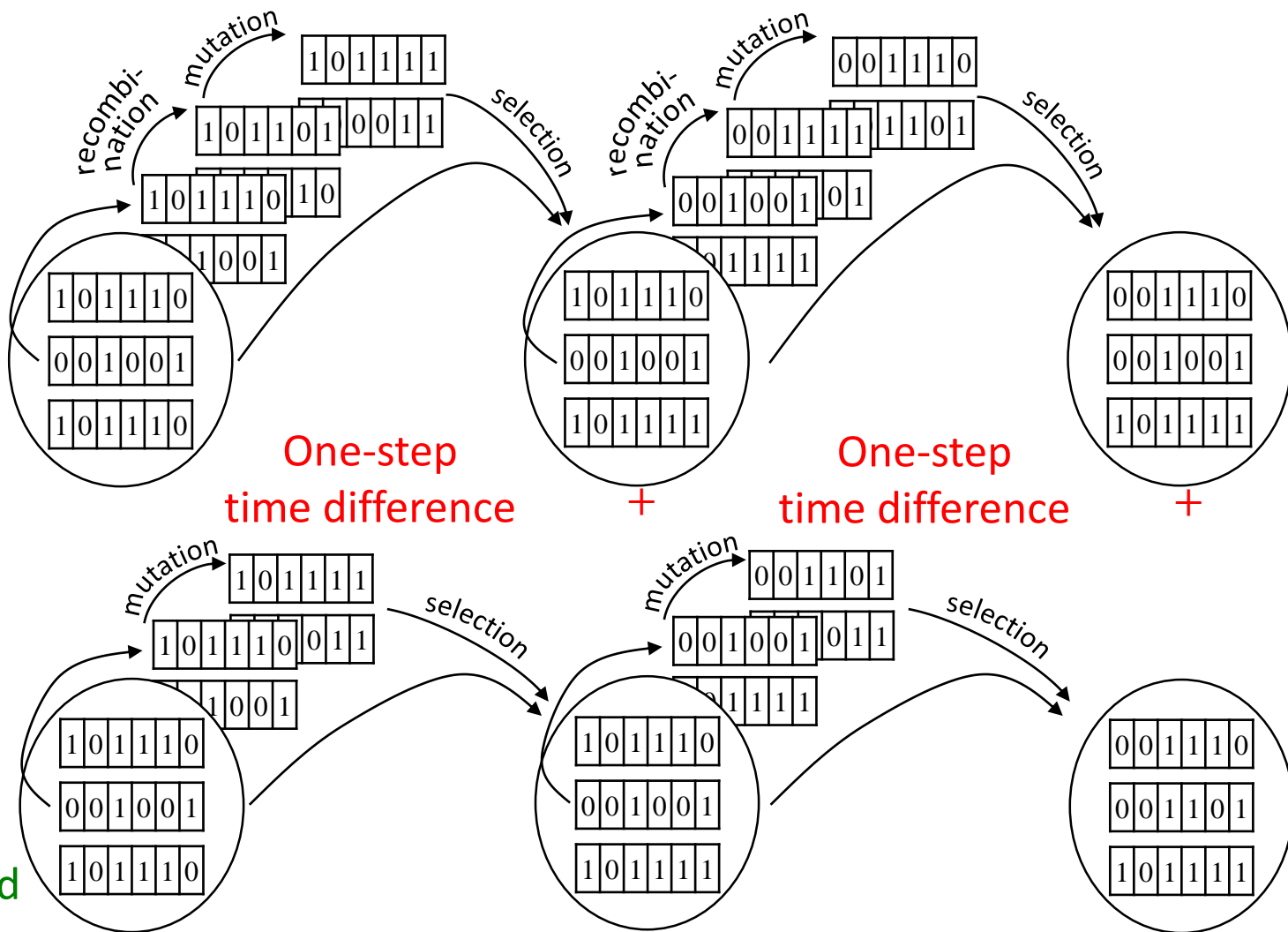
Hard to be analyzed directly

EA with recombination

Compare

EA without recombination

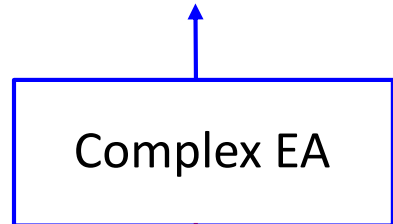
Easier to be analyzed



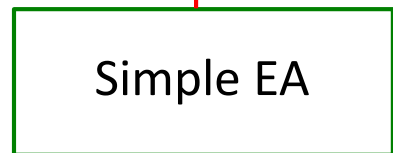
# Switch analysis

Simplify the analysis

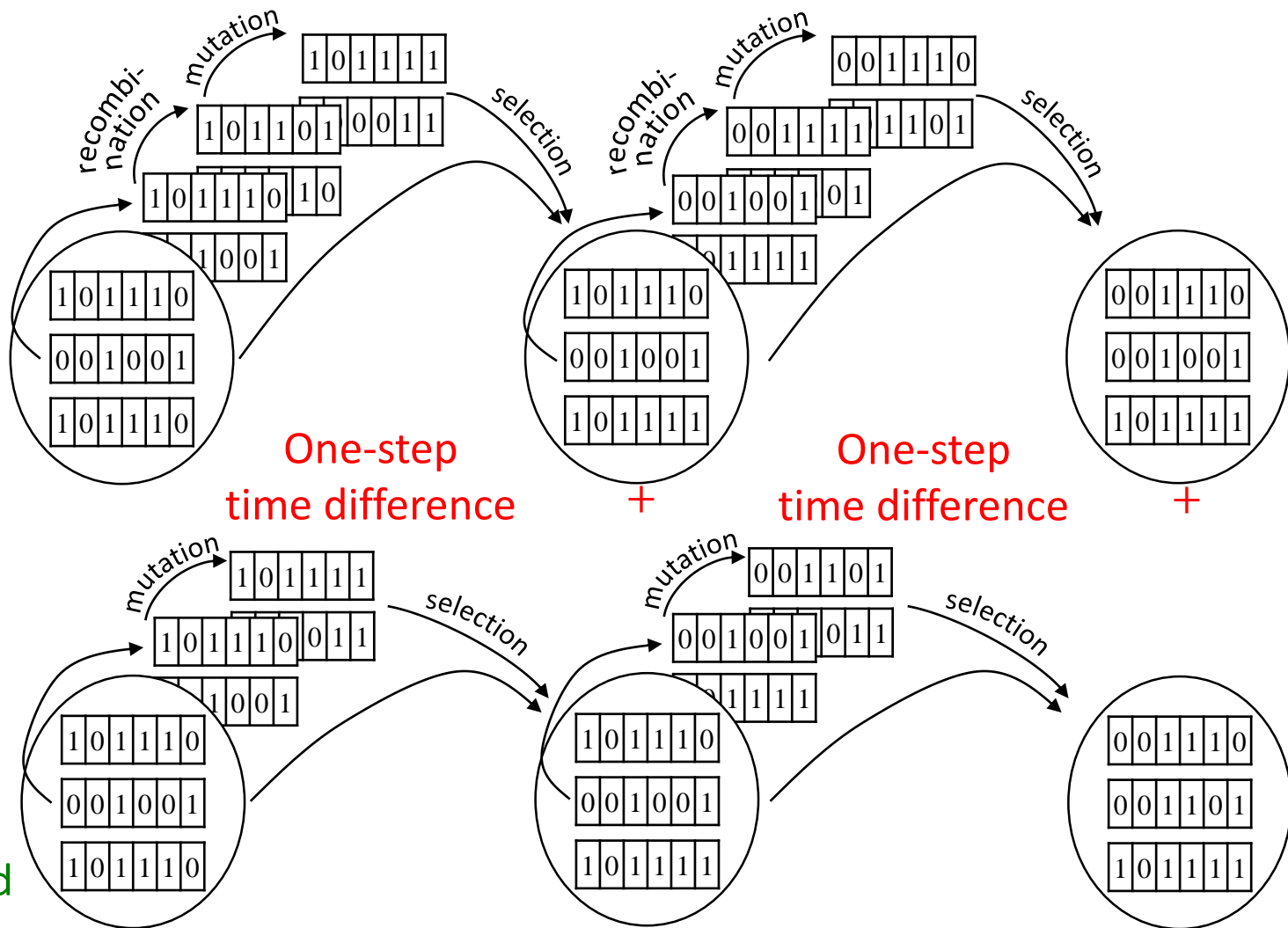
Hard to be analyzed directly



Compare

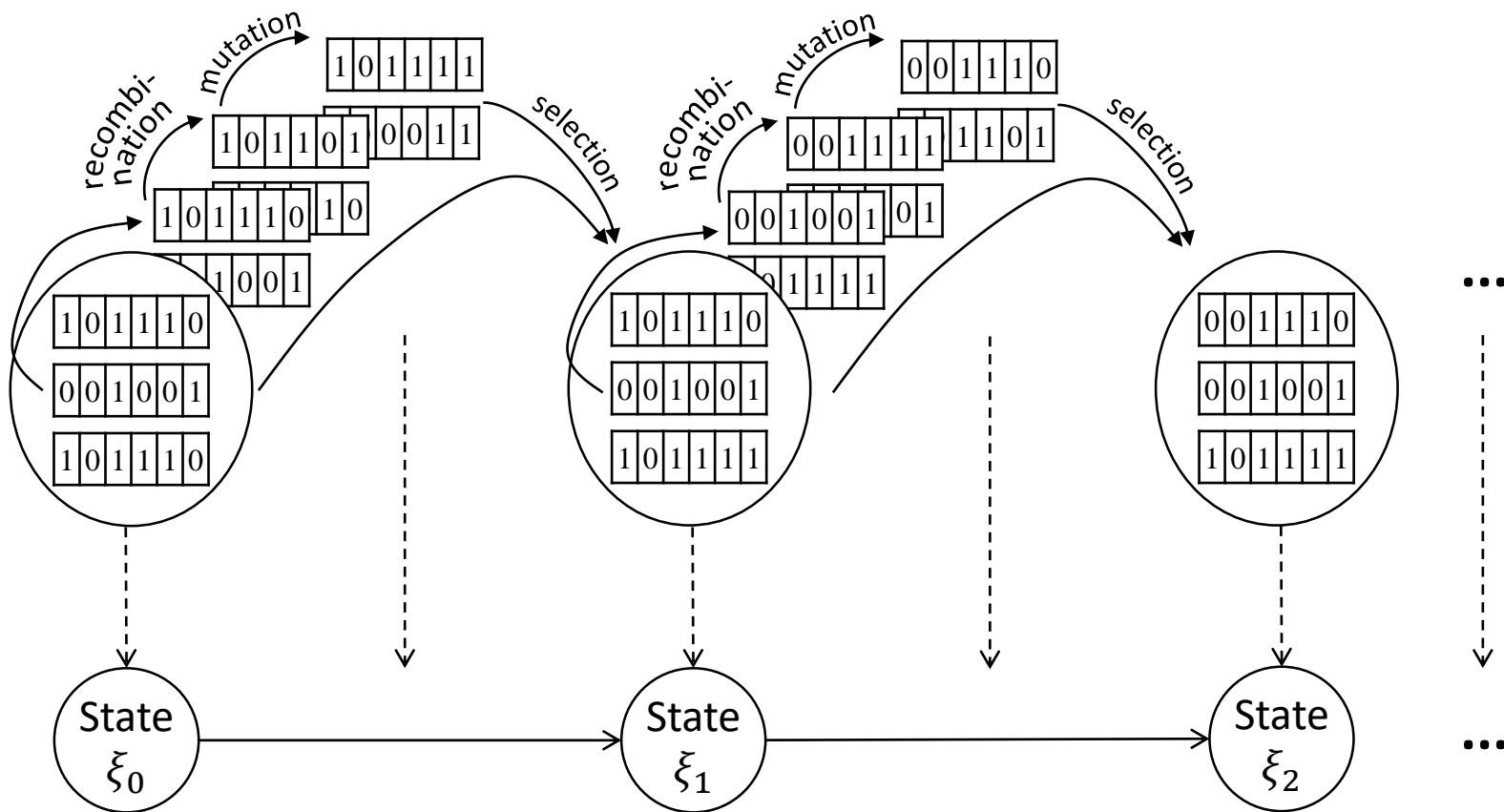


Easier to be analyzed



# Switch analysis

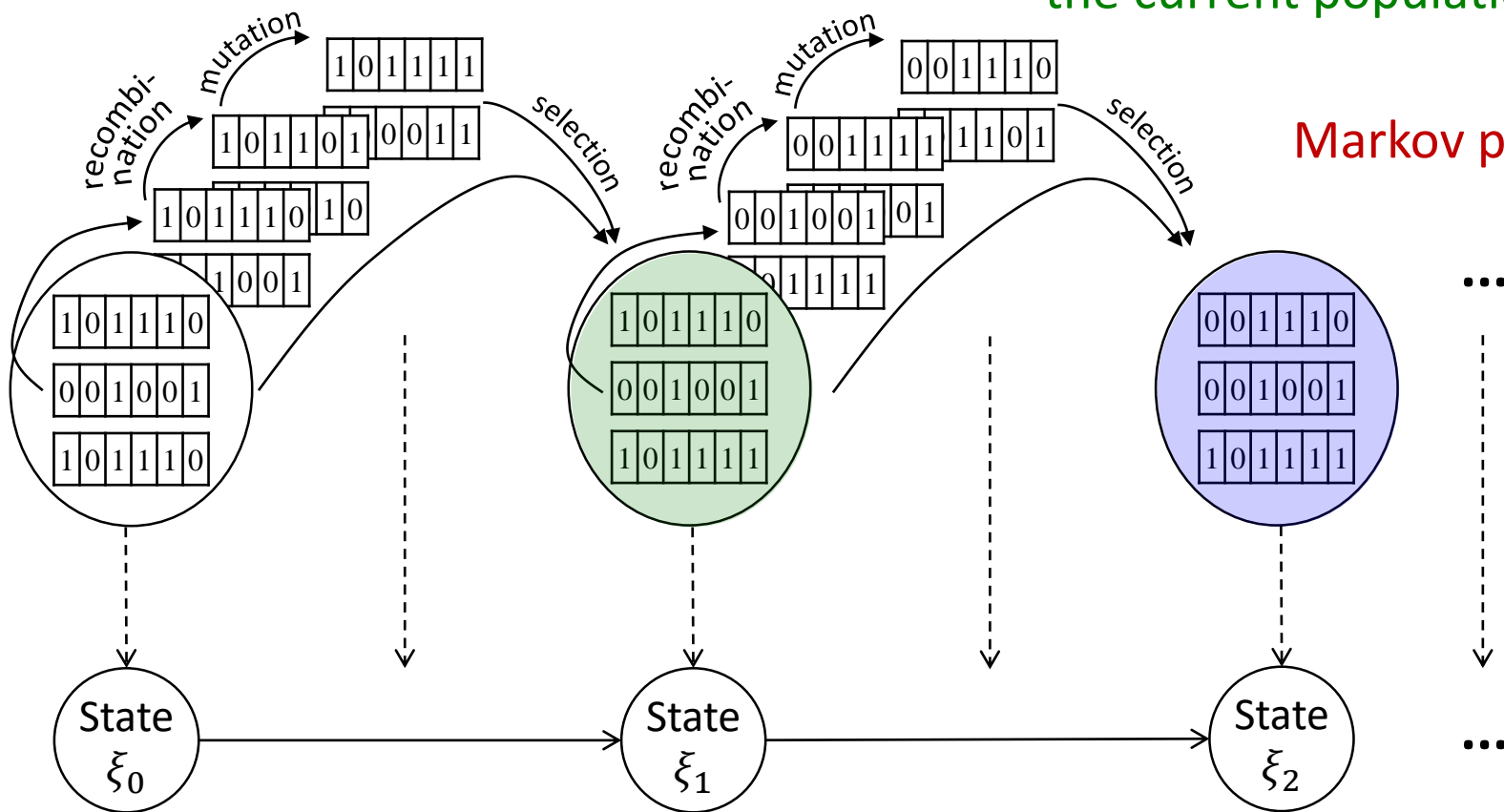
Model an EA process as a Markov chain



# Switch analysis

Model an EA process as a Markov chain

The generation of the next population only depends on the current population

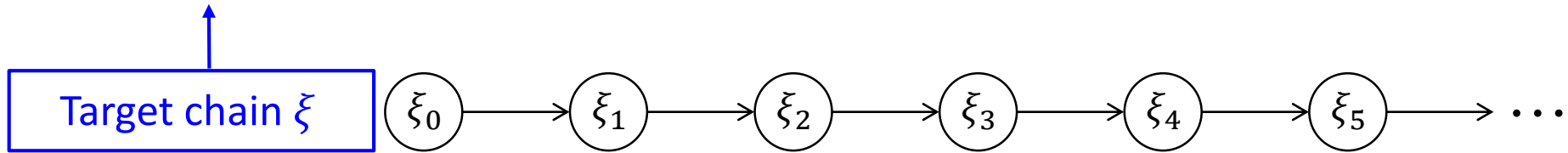


# Switch analysis

---

**Main idea** [Yu, Qian & Zhou, TEC'15]:

Hard to be analyzed directly

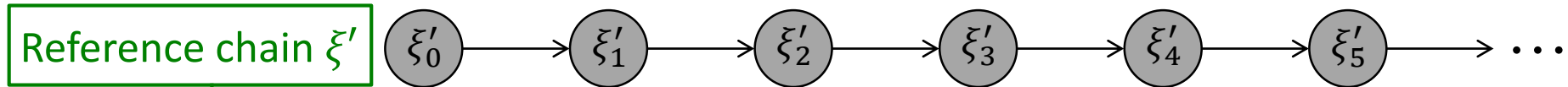
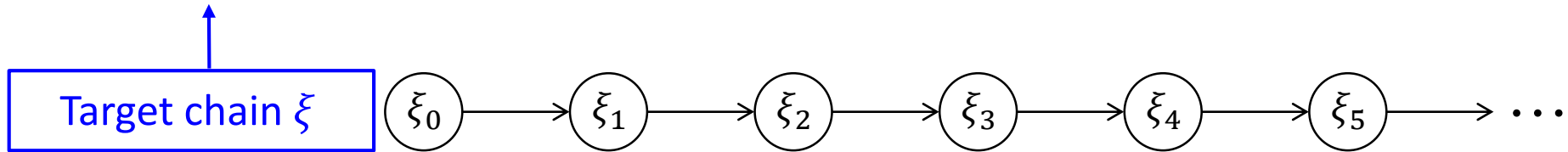


# Switch analysis

---

**Main idea** [Yu, Qian & Zhou, TEC'15]:

Hard to be analyzed directly



Easier to be analyzed

# Switch analysis

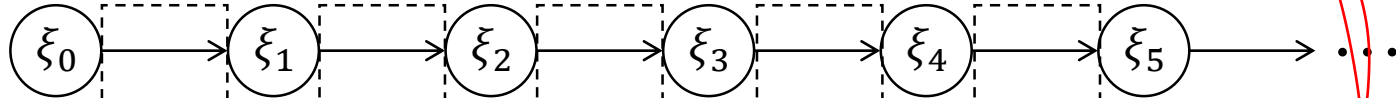
**Main idea** [Yu, Qian & Zhou, TEC'15]:

Hard to be analyzed directly

Expected running time

$$\mathbb{E}[\tau]$$

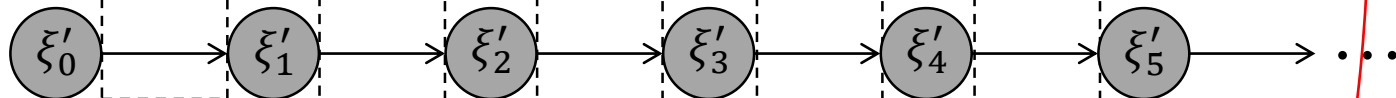
Target chain  $\xi$



One-step time difference

$$\rho_0 + \rho_1 + \rho_2 + \rho_3 + \rho_4 + \dots$$

Reference chain  $\xi'$



Easier to be analyzed

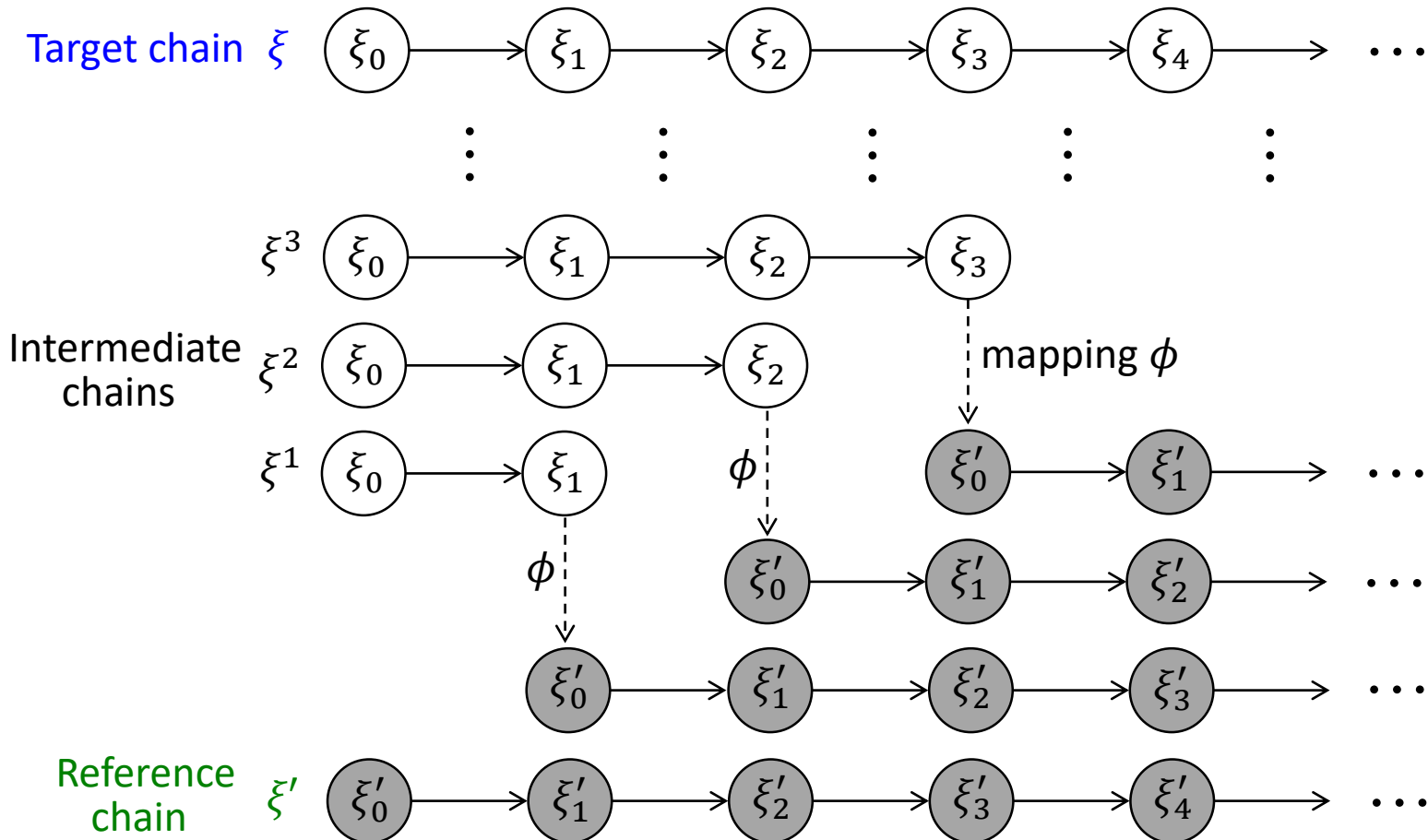
Expected running time

$$\mathbb{E}[\tau']$$

How to estimate one-step time difference  $\rho_t$ ?

# Switch analysis

How to estimate one-step time difference  $\rho_t$ ?

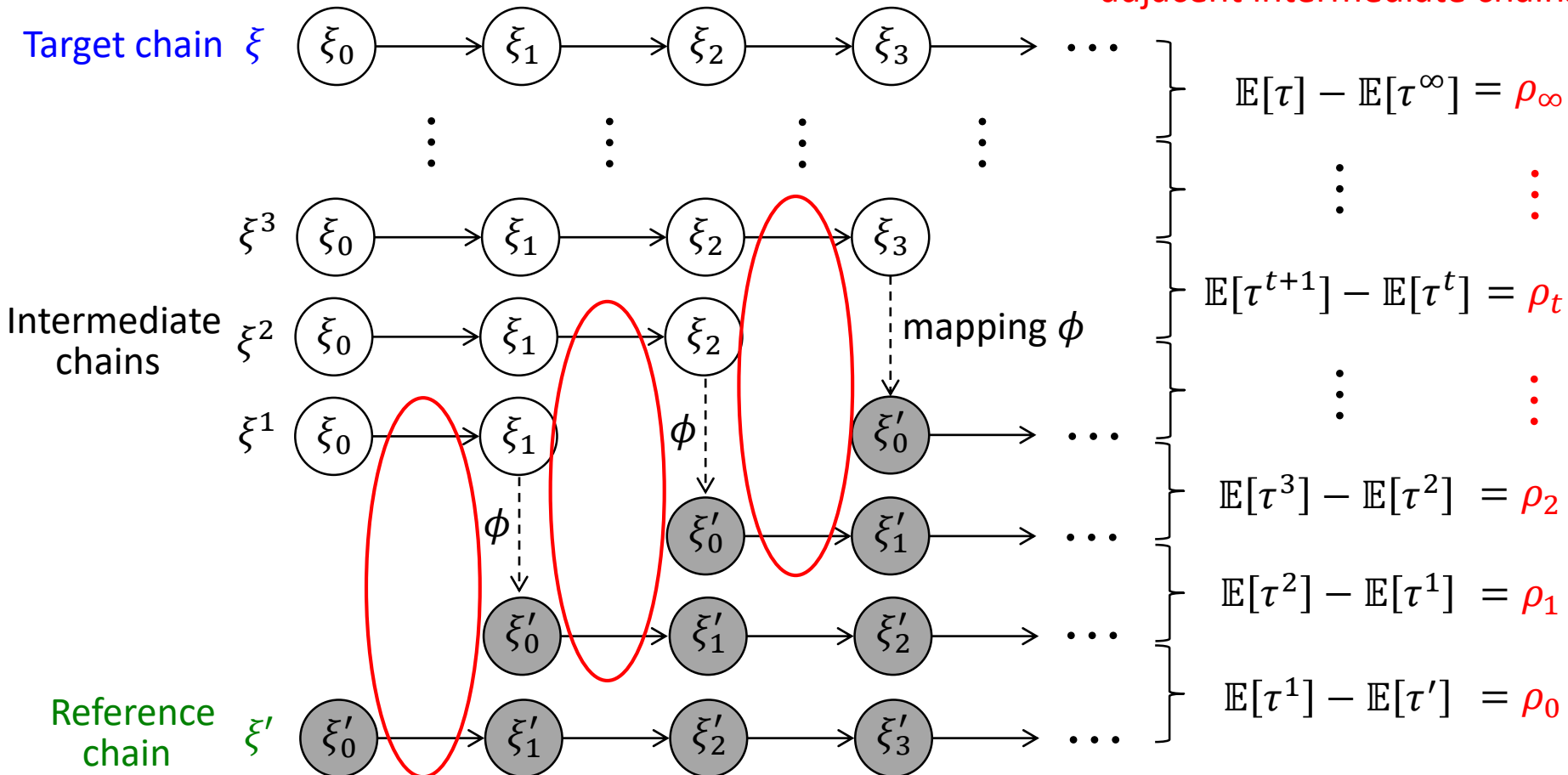




# Switch analysis

How to estimate one-step time difference  $\rho_t$ ?

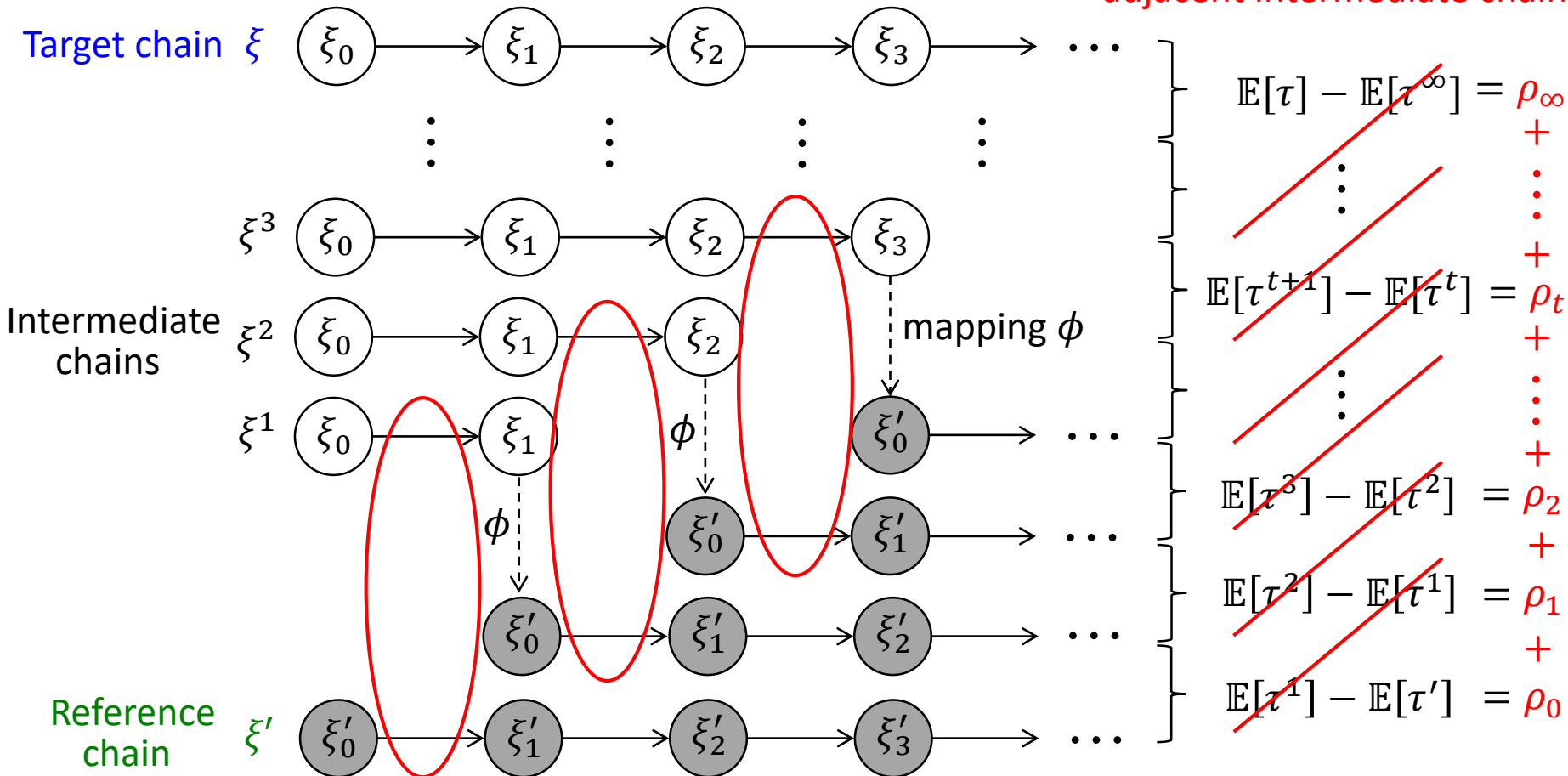
Time difference between adjacent intermediate chains



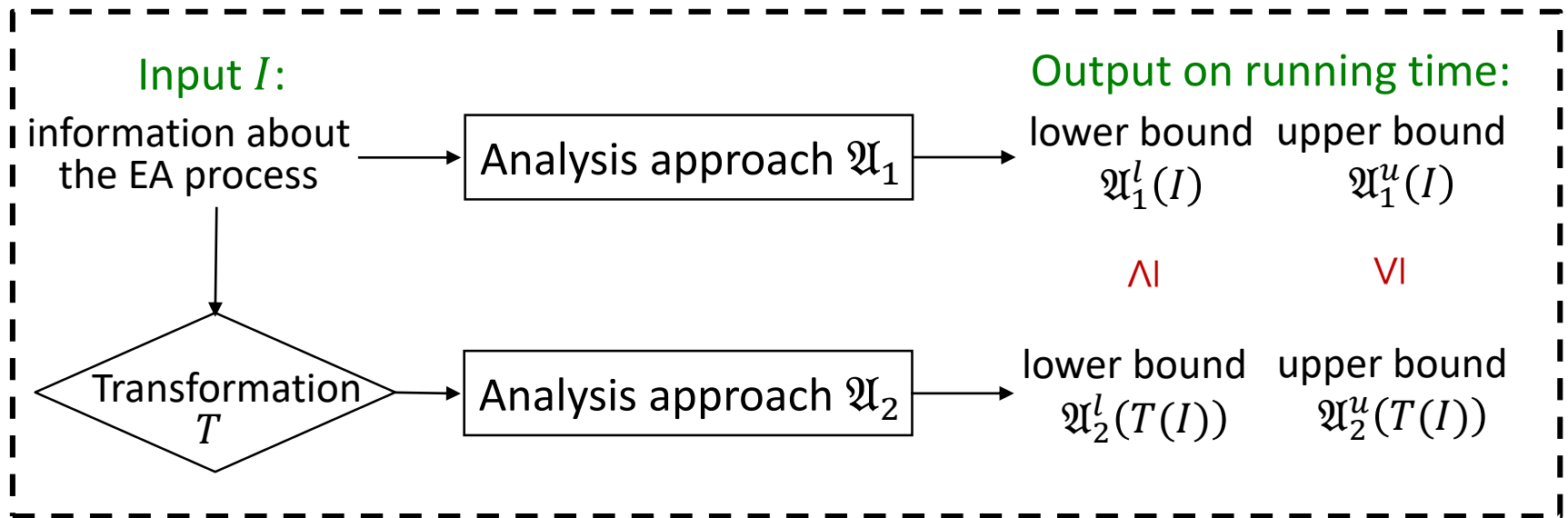
# Switch analysis

How to estimate one-step time difference  $\rho_t$ ?

Time difference between adjacent intermediate chains



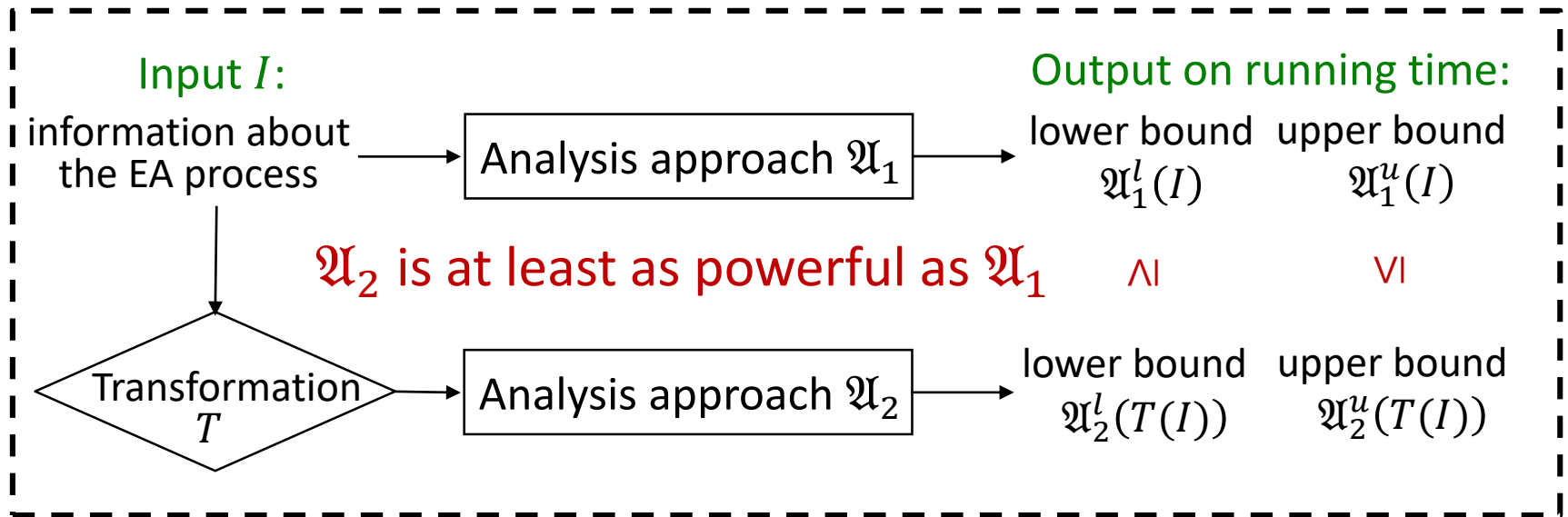
# Ability of switch analysis



$\mathfrak{A}_2$  is at least as powerful as  $\mathfrak{A}_1$

$\mathfrak{A}_2$  can derive at least the same tight bound as  $\mathfrak{A}_1$   
while requiring no more information

# Ability of switch analysis



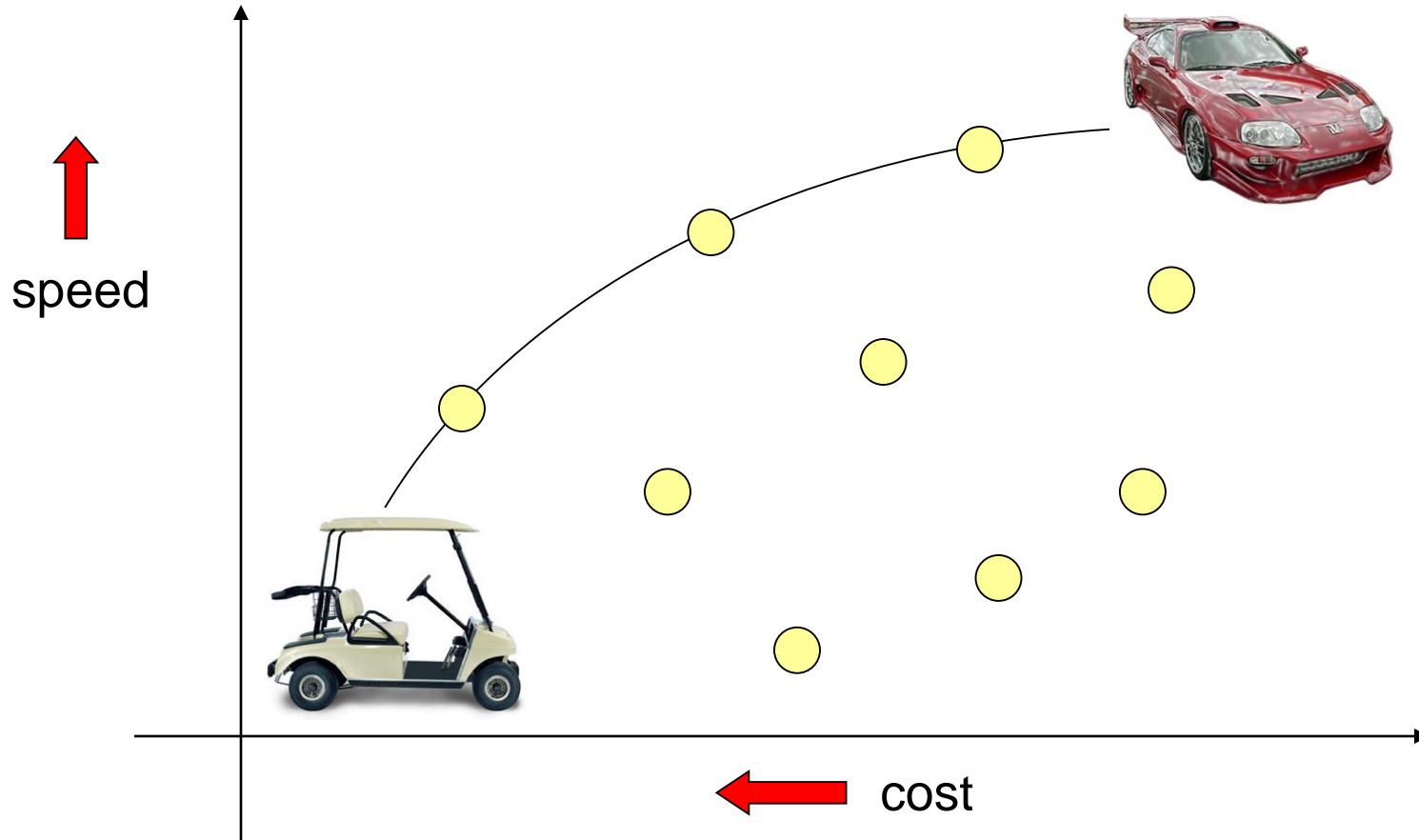
Switch analysis is at least as powerful as

- Fitness level
- Drift analysis

Each one can be viewed as switch analysis with a specific reference chain

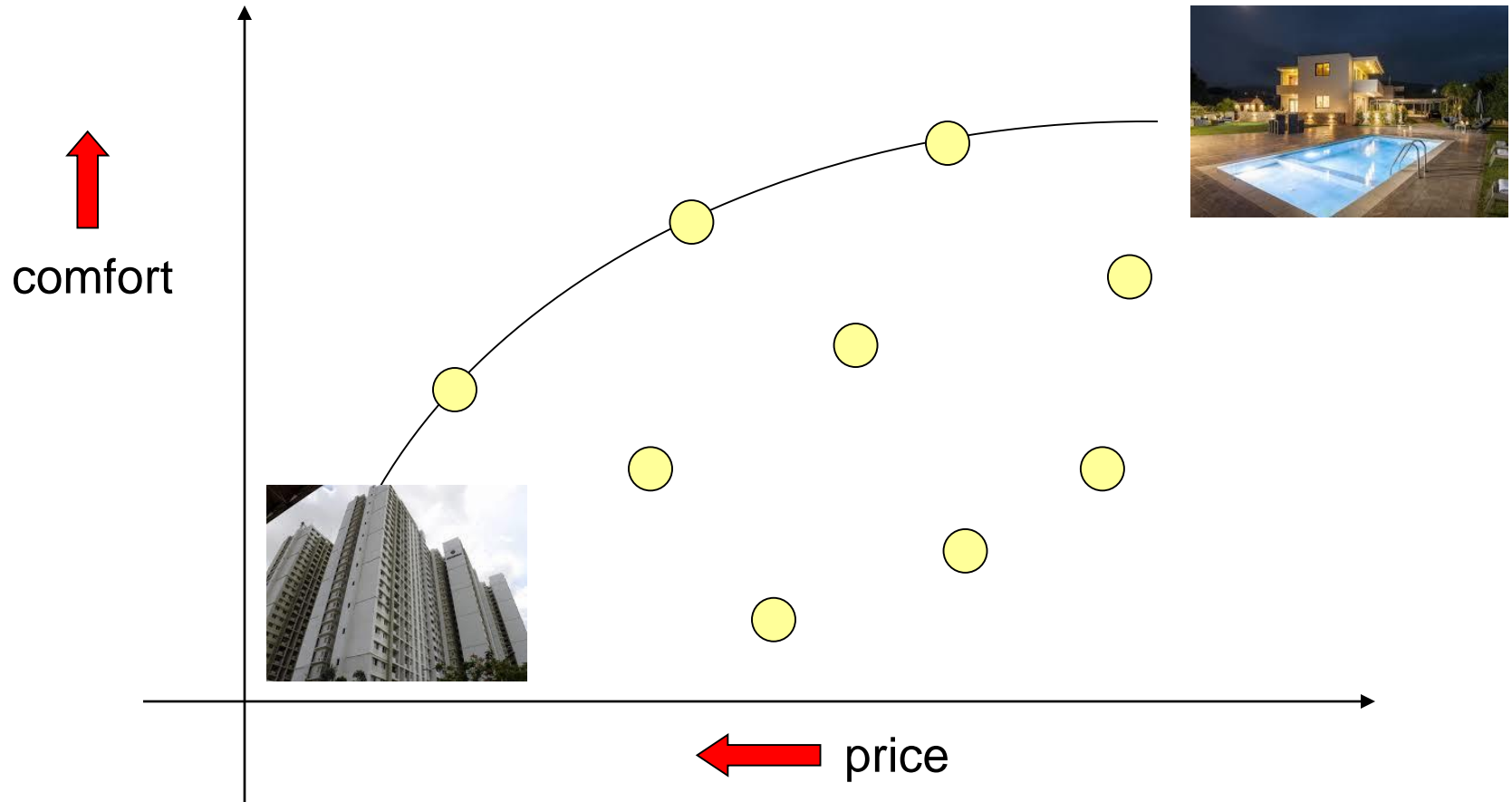
# Example of multi-objective optimization

---



# Example of multi-objective optimization

---



# Running time analysis tools

---

- Fitness level
  - Original fitness level
  - Refined fitness level
- Drift analysis
  - Additive drift
  - Multiplicative drift
  - Negative drift
- Switch analysis

When EAs are applied to solve multi-objective optimization problems, can these approaches be easily used to analyze their running time?

# Multi-objective optimization

---

**Multi-objective optimization:** optimize multiple objectives (which are usually conflicting) simultaneously

$$\max_{x \in \mathcal{X}} (f_1(x), f_2(x), \dots, f_m(x))$$

- $x$  **weakly dominates**  $y$ , denoted as  $x \succcurlyeq y$ , if

$$\forall i \in \{1, 2, \dots, m\}: f_i(x) \geq f_i(y)$$

- $x$  **dominates**  $y$ , denoted as  $x \succ y$ , if

$$\forall i \in \{1, 2, \dots, m\}: f_i(x) \geq f_i(y) \text{ and } \exists i \in \{1, 2, \dots, m\}: f_i(x) > f_i(y)$$

- $x$  is **incomparable** with  $y$ , if neither  $x \succcurlyeq y$  nor  $y \succcurlyeq x$



# Multi-objective optimization

---

**Multi-objective optimization:** optimize multiple objectives (which are usually conflicting) simultaneously

$$\max_{x \in \mathcal{X}} (f_1(x), f_2(x), \dots, f_m(x))$$

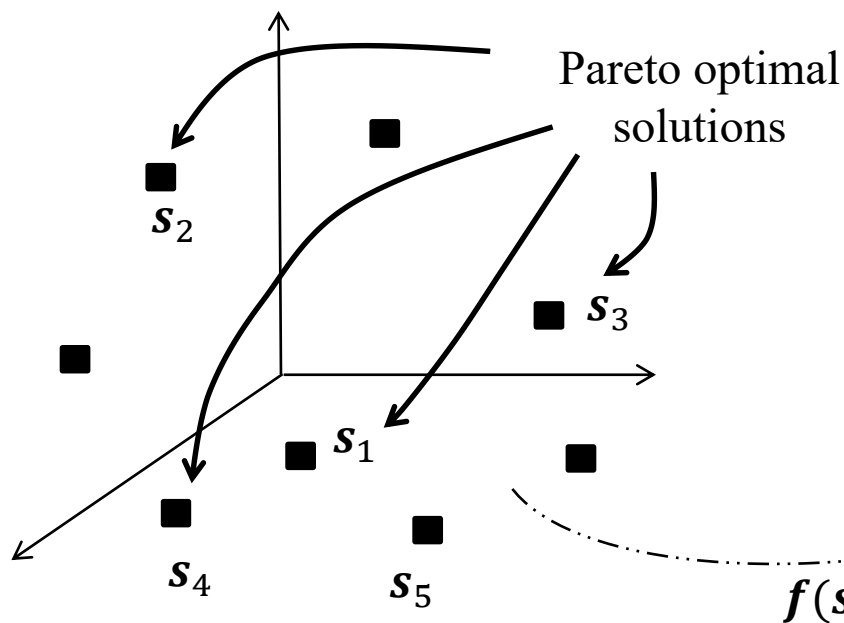
A solution is **Pareto optimal** if no other solution dominates it

The collection of objective vectors of all Pareto optimal solutions is called the **Pareto front**

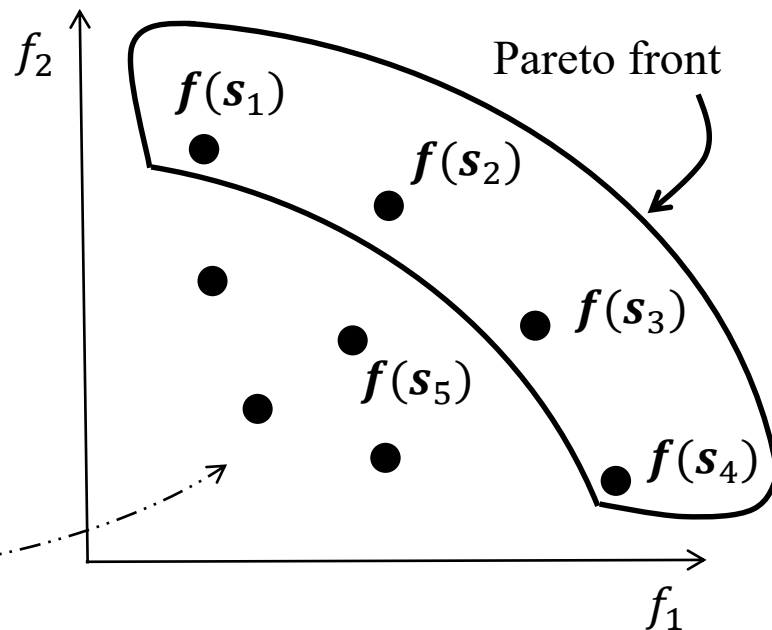
**The goal** of multi-objective optimization is to find a set of solutions whose objective vectors cover the Pareto front

# Example of multi-objective optimization

Solution Space



Objective Space



$$s_2 \succcurlyeq s_5$$

$$s_2 \succ s_5$$

$s_2$  is incomparable with  $s_3$

Assume the solution space contains only these 8 solutions

# Example of running time analysis

---

**GSEMO:** Given a pseudo-Boolean function vector  $f$ :

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

**LOTZ:**

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

Count the number of leading 1-bits

Count the number of trailing 0-bits

The Pareto set: 00 ... 00, 10 ... 00, ..., 11 ... 10, 11 ... 11

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

# 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 set {00 ... 00, 10 ... 00, ..., 11 ... 10, 11 ... 11}

**The analysis of phase (1):** the first objective  $\sum_{i=1}^n \prod_{j=1}^i x_j$

- consider the largest LO value in the population, which never decreases
- select the solution with the largest LO value, and only flip its first 0 bit
- the probability is at least  $\frac{1}{n+1} \frac{1}{n} (1 - \frac{1}{n})^{n-1}$

because the population size is no larger than  $n + 1$ ,  
and a solution is uniformly selected at random

# 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 set {00 ... 00, 10 ... 00, ..., 11 ... 10, 11 ... 11}

**The analysis of phase (1):** the first objective  $\sum_{i=1}^n \prod_{j=1}^i x_j$

- consider the largest LO value in the population, which never decreases
- select the solution with the largest LO value, and only flip its first 0 bit
- the probability is at least  $\frac{1}{n+1} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1}$

# Proof

---

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

Main idea:

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

- (1) obtain the Pareto optimal solution  $11 \dots 11$
- (2) obtain the Pareto set  $\{00 \dots 00, 10 \dots 00, \dots, 11 \dots 10, 11 \dots 11\}$

## The analysis of phase (1):

- consider the largest LO value in the population, which never decreases
- select the solution with the largest LO value, and only flip its first 0 bit
- the probability is at least  $\frac{1}{n+1} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1}$

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

# Proof

---

## The analysis of phase (2):

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

the probability is at least  $\frac{1}{n+1} \frac{1}{n} \left(1 - \frac{1}{n}\right)^{n-1}$

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

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

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

# Switch analysis for multi-objective optimization

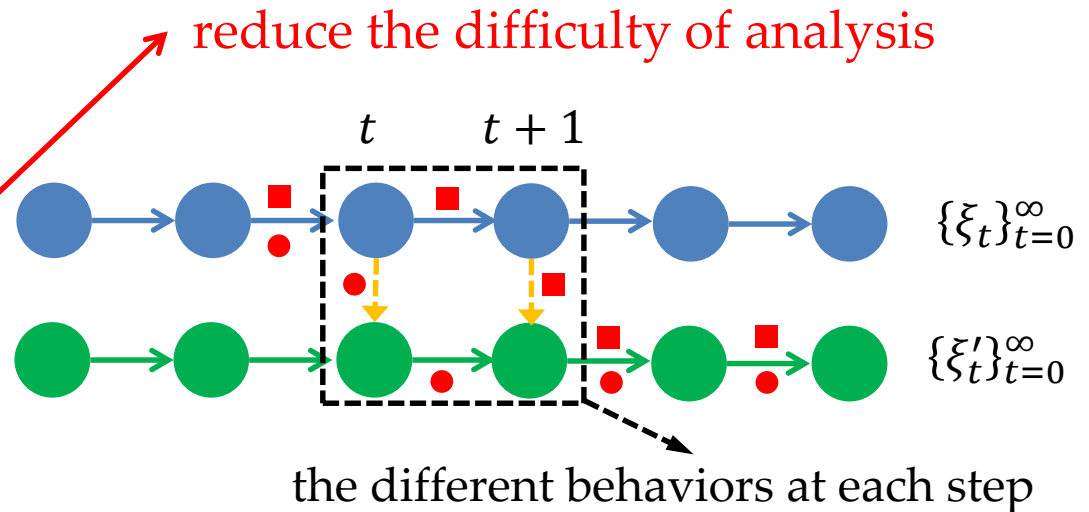
**Theorem:**  $\xi \in \mathcal{X}$  modeling an EA solving a multi-objective optimization problem, a well-defined function  $h_{\alpha,c}: \mathcal{X} \rightarrow \mathbb{N}_0$  and a Markov chain  $\xi' \in \mathcal{Y} = \{0,1\}^r$  with  $\mathcal{Y}^* = \{1^r\}$  such that  $\forall x \notin \mathcal{X}^*, \forall t \geq 0$ ,

$$\begin{aligned} & \sum_{i \in [r]} \mathbb{P}(\min\{h(\xi_{t+1}), r\} = i | \xi_t = x) \mathbb{E}[\tau' | \xi'_0 = 1^i 0^{r-i}] \\ & \leq \sum_{y \in \mathcal{Y}} \mathbb{P}(\xi'_1 = y | \xi'_0 = 1^{h(x)} 0^{r-h(x)}) \mathbb{E}[\tau' | \xi'_1 = y] + \delta, \\ \Rightarrow \mathbb{E}[\tau | \xi_0 = x_0] & \leq \mathbb{E}[\tau' | \xi'_0 = 1^{\min\{h(x_0), r\}} 0^{r-\min\{h(x_0), r\}}] / (1 - \delta) \end{aligned}$$

Basic idea [Bian et al., IJCAI'18]:

Given EA solving the given multi-objective problem

A Markov chain modeling a reference single-objective evolutionary process





# Application of switch analysis

Application [Bian, Qian and Tang, IJCAI'18]:

GSEMO	Problem	Previous result	Our result
Bi-objective	LOTZ	$O(n^3)$ [Giel, CEC'03]	$\leq 6n^3$
	COCZ	$O(n^2 \log n)$ [Qian et al., AIJ'13]	$\leq 3n^2 \log n$
Many-objective	$m$ COCZ	$O(n^{m+1})$ [Laumanns et al., TEC'04]	$O(n^m)$ for $m > 4$ , $O(n^3 \log n)$ for $m = 4$
Approximate analysis	WOMM	—	$1/n$ -approximation: $O(n^2 (\log_l n + \log_l(w_n/w_1)))$

→ gives the leading constants

→ is asymptotically tighter than



**L. Thiele**

Member of  
Academia  
Europaea

Switch analysis is general and powerful

# Running time analysis tools

---

- Fitness level
  - Original fitness level
  - Refined fitness level
- Drift analysis
  - Additive drift
  - Multiplicative drift
  - Negative drift
- Switch analysis

# 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]

(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 et al., TEC'09]

(1+ $\lambda$ )-EA linear function  $O(\lambda n + n \log n)$  [Doerr & Kunnemann, TCS'15]  
vertex cover *exponential* on bipartite graphs [Oliveto et al., 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];  
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  $O(m^3 w_{\min}(|C| + \log n + \log w_{\max}))$   
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]

LOTZ  $\Theta(n^2)$ ; COCZ  $\Theta(n \log n)$ ; bi-objective minimum spanning tree

REMO       $O\left(m^2 n w_{\min}(|C| + \frac{\log n + \log w_{\max}}{n w_{\min}} - N_{gc}(1 - \frac{1}{m}))\right)$  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]

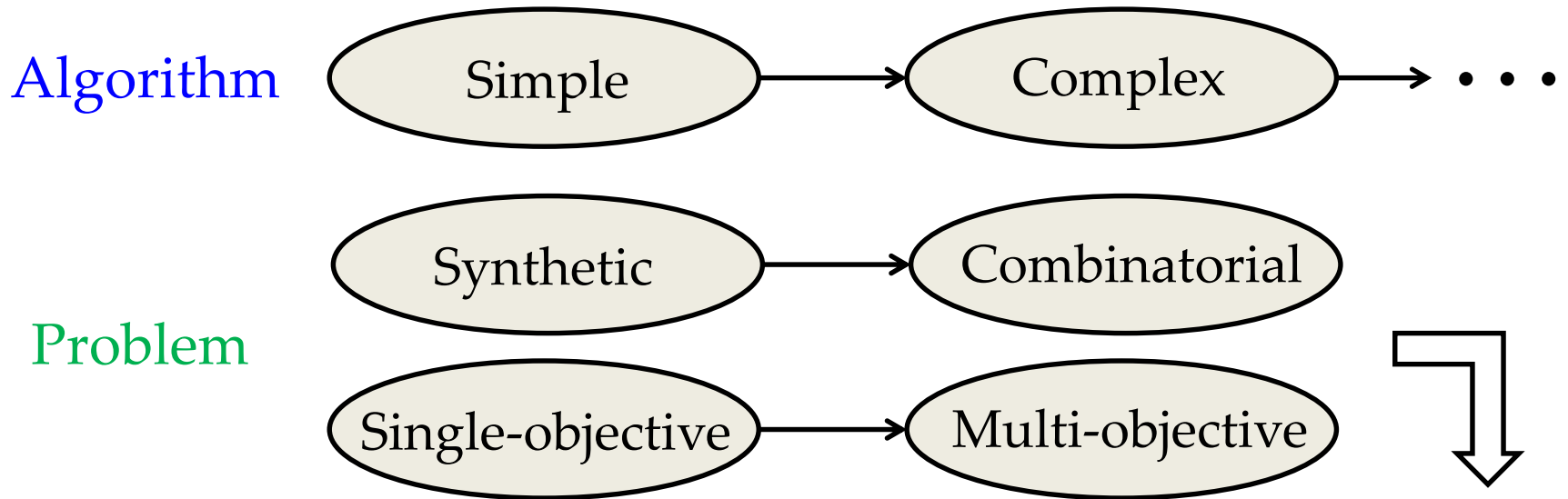
NSGA-II      LOTZ  $O(n^3)$ ; OneMinMax  $O(n^2 \log n)$   
[Zheng et al., AAAI'22; Bian & Qian, PPSN'22]

# Theoretical analysis of EAs

---



## Running time analysis



Environments with uncertainty

# Theory community

---



Ingo Wegener (1950-2008), TU Dortmund, Germany

Stefan Droste, Thomas Jansen, Ingo Wegener:

A Rigorous Complexity Analysis of the (1 + 1) Evolutionary Algorithm for Separable Functions with Boolean Inputs. *Evolutionary Computation* 6(2): 185-196 (1998)

University of Sheffield: Pietro Oliveto

UK University of Birmingham: Per Kristian Lehre  
Aberystwyth University: Thomas Jansen

Germany University of Potsdam: Tobias Friedrich  
University of Passau: Dirk Sudholt

France Ecole Polytechnique: Benjamin Doerr  
Sorbonne University: Carola Doerr

Switzerland ETH Zürich: Johannes Lengler

Denmark Technical University of Denmark: Carsten Witt

Australia University of Adelaide: Frank Neumann

## China

Nanjing University

Sun Yat-sen University

Harbin Institute of Technology,  
Shenzhen

Southern University of  
Science and Technology

# Summary

---

- Fitness level
  - Original fitness level
  - Refined fitness level
- Drift analysis
  - Additive drift
  - Multiplicative drift
  - Negative drift
- Switch analysis
- Results of running time analysis

# References

---

- C. Bian and C. Qian. Better running time of the non-dominated sorting genetic algorithm II (NSGA-II) by using stochastic tournament selection. In: *Proceedings of the 17th International Conference on Parallel Problem Solving from Nature (PPSN'22)*, Dortmund, Germany, 2022, pages 428-441
- C. Bian, C. Qian and K. Tang. A general approach to running time analysis of multi-objective evolutionary algorithms. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, Stockholm, Sweden, 2018, pages 1405-1411
- D. C. Dang and P. K. Lehre. Runtime analysis of non-elitist populations: From classical optimisation to partial information. *Algorithmica*, 2016, 75(3): 428–461
- B. Doerr, D. Johannsen and C. Winzen. Multiplicative drift analysis. *Algorithmica*, 2012, 64: 673-697
- 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



# References

---

- O. Giel. Expected runtimes of a simple multiobjective evolutionary algorithm. In *Proceedings of the 2003 IEEE Congress on Evolutionary Computation (CEC'03)*, Canberra, Australia, 2003, pages 1918–1925
- B. Hajek. Hitting time and occupation time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 1982, 14(3): 502–525
- J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. *Artificial Intelligence*, 2001, 127(1): 57-85
- 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
- F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science*, 2007, 378(1): 32-40
- P. S. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. *Algorithmica*, 2011, 59(3): 369-386

# References

---

- 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
- 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
- D. Sudholt. A new method for lower bounds on the running time of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 2013, 17(3): 418-435
- 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
- W. Zheng, Y. Liu and B. Doerr. A first mathematical runtime analysis of the non-dominated sorting genetic algorithm II (NSGA-II). In: *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI'22)*. Virtual, 2022, pages 10408-10416

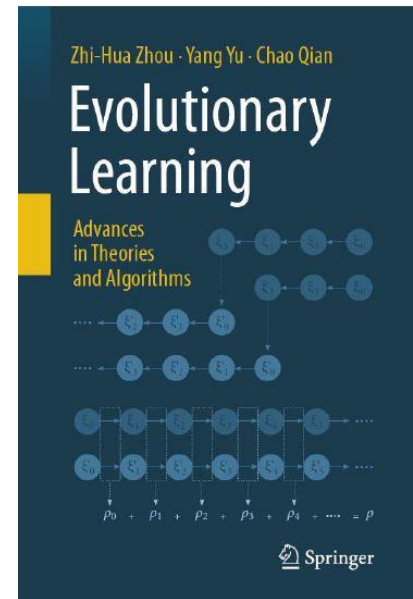
# Reading books



## Bioinspired Computation in Combinatorial Optimization

Algorithms and Their Computational Complexity

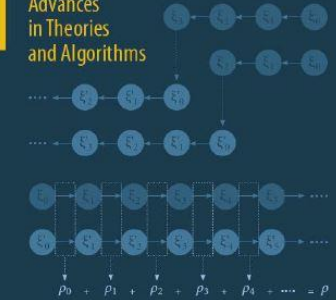
Authors: **Neumann, Frank, Witt, Carsten**



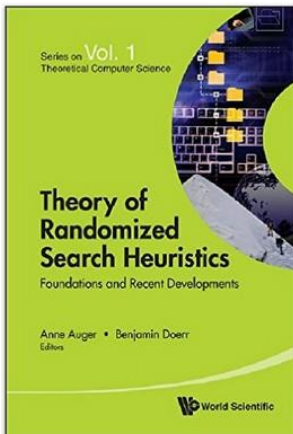
Zhi-Hua Zhou · Yang Yu · Chao Qian

## Evolutionary Learning

Advances  
in Theories  
and Algorithms



Springer



## Theory of Randomized Search Heuristics

Foundations and Recent Developments

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

# Assignment - 3

---

## Task:

- Apply fitness level method to analyze the upper bound on the expected running time of (1+1)-EA solving LeadingOnes
- Apply multiplicative and additive drift analysis to analyze the upper bound on the expected running time of (1+1)-EA solving OneMax
- Analyze the upper bound on the expected running time of GSEMO solving COCZ

Deadline: Dec. 17

详见课程网站