Theoretical Foundation

Yang Yu and Zhi-Hua Zhou
LAMDA Group
National Key Laboratory for Novel Software Technology
Nanjing University, China

Road map

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

on performance in real-world situations

Theoretical studies

Focus on abstract and mathematical aspects of EAs

Develop solid, rigorous, and reliable knowledge

- empirical studies are limited to the experimented cases
- overcome experiment difficulties
- derive provable conclusions

Particularly for EAs

- when to use them
- what are their merits and drawbacks?
- how different configurations affect their performance?
- design better EAs

...

from rules of thumb to well understood heuristics

Conventional algorithm analysis

Problem → Algorithm → Measurement

Sorting	Quick Sort	average time complexity $O(n \log n)$
Shortest Path	Dijkstra's algorithm	average time complexity $O(\|V\|^2)$
Linear Programming	Simplex	worst case time complexity: exponential smoothed complexity: polynomial

Time complexity

What about an algorithm sorts (5,4,2,8,9) in 3 steps?

measured in a class of problem instances

e.g. all possible arrays of 5 numbers

average complexity

worst case complexity

measure the growing rate as the problem size increases

e.g. $2n^2$

asymptotical notation $O(n^2)$

Time complexity

asymptotical notation

$$f(n) \in O(g(n)) : \quad \exists c, n_0 > 0 \text{ such that } \forall n \geq n_0 : f(n) \leq cg(n)$$

$$f(n) \in \Omega(g(n)) : \quad \exists c, n_0 > 0 \text{ such that } \forall n \geq n_0 : f(n) \geq cg(n)$$

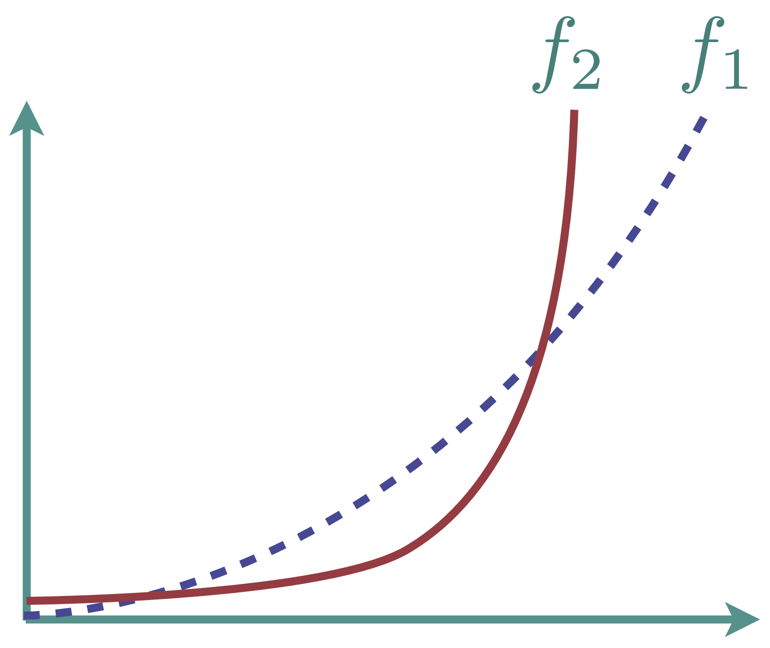
$$f(n) \in \Theta(g(n)) : \quad f(n) \in O(g(n)) \text{ and } f(n) \in \Omega(g(n))$$

e.g.,

$$f_1(n) = 1000n^2 \in \Theta(n^2)$$

$$f_2(n) = 0.01 \cdot 2^n \in \Theta(2^n)$$

$$f_1(n) \in O(f_2) \text{ and } f_2 \in \Omega(f_1)$$



But for EAs...

nature phenomena

Problem

Algorithm

Measurement

problem unknown

not designed with knowledge of problems

theoretical understanding is even more important

Local dynamics

-- how the population changes in steps

Schemata Theory [Holland, 75]

consider a binary solution space $\{0, 1\}^5 =$

00000	01000	10000	11000
00001	01001	10001	11001
00010	01010	10010	11010
00011	01011	10011	11011
00100	01100	10100	11100
00101	01101	10101	11101
00110	01110	10110	11110
00111	01111	10111	11111

a schema is a template with "#"= "any"

a schema defines a subspace

e.g.

01#1# order 3

#1#1# order 2

###1# order 1

how the population size changes in a schema/subspace?

Local dynamics

-- how the population changes in steps

$m(H_k, t)$: population size in the subspace H_k with order k

basic idea:

$$E[m(H_k, t+1)] = (1 - P(\text{leaving from } H_k))m(H_k, t) + P(\text{coming to } H_k)(m - m(H_k, t))$$

example: [Holland, 1975]

probability of passing the selection

probability of using the mutation

$$E[m(H_k, t+1)] > \frac{f(H_k)}{\bar{f}}(1 - kP_m - P_c P_d(H_k))m(H_k, t)$$

probability of using the crossover

probability that the crossover disrupts a solution

- *higher order schema are easier broken*
- *implicitly parallelism*

Local dynamics

Useful in:

- analyzing local/immediate schema changes
- assisting deriving intuitive guidances

local properties do not automatically tell the global results

Unanswered questions:

- does an EA converge?
- how fast an EA converges?
- ...

Road map

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

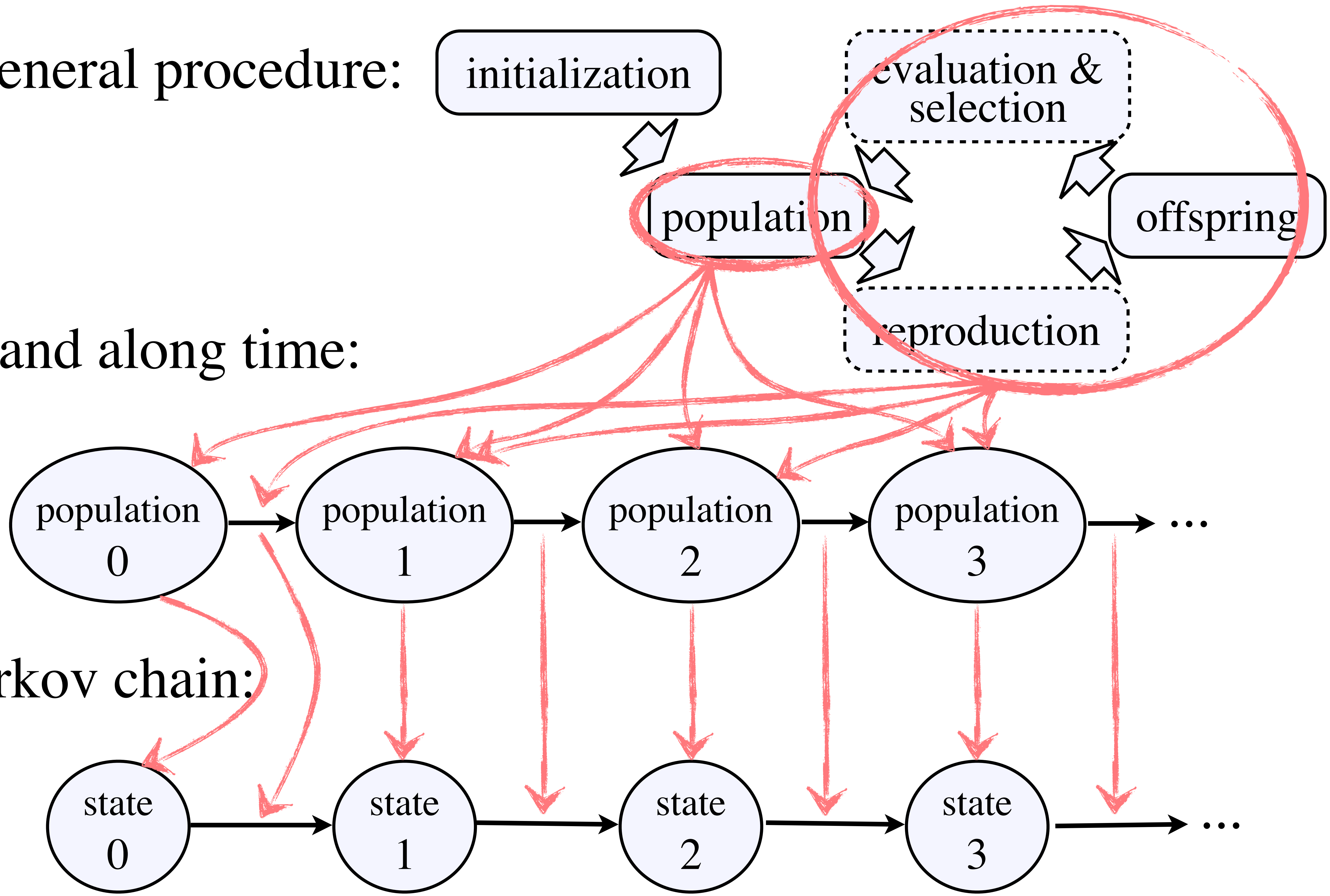
on performance in real-world situations

Markov chain modeling

A general procedure:

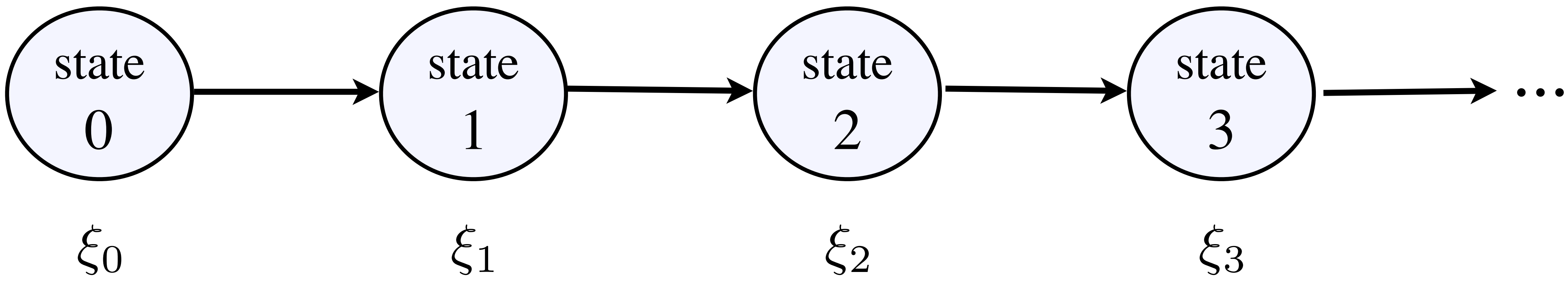
expand along time:

Markov chain:



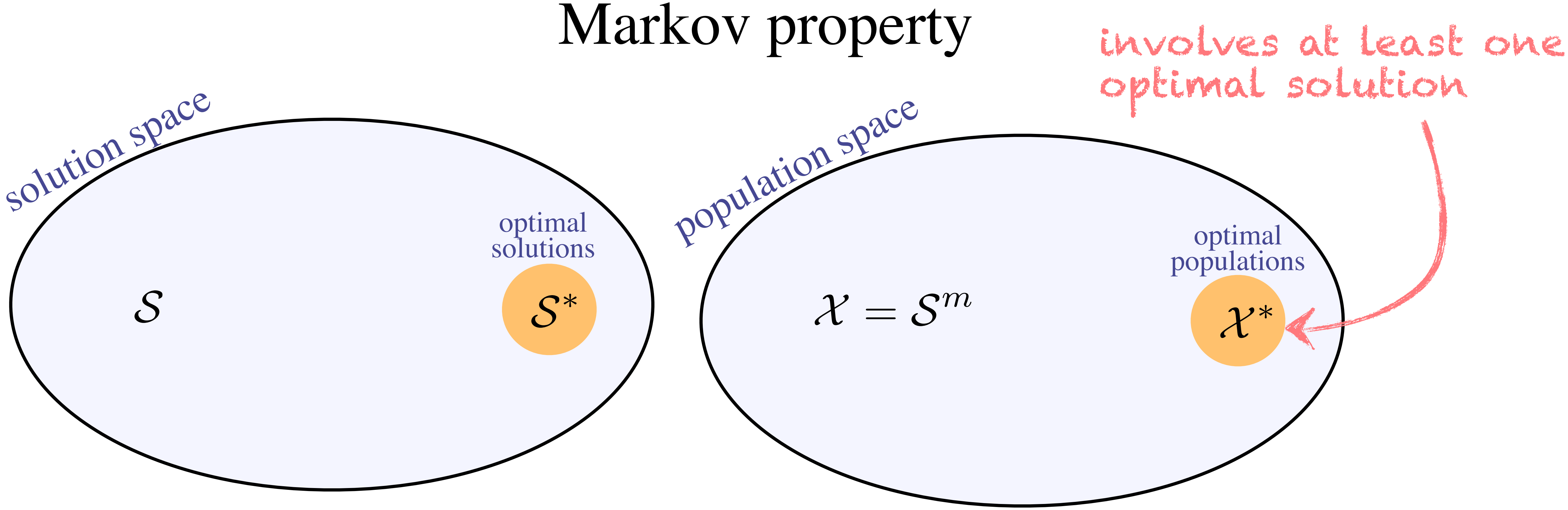
Markov chain modeling

Markov chain:



$$P(\xi_3 \mid \xi_2, \xi_1, \xi_0) = P(\xi_3 \mid \xi_2)$$

Markov property



Convergence

Does an EA converge to the global optimal solutions?

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

Considered as *closed*:

gain of optimality in one step

loss of optimality in one step

Theorem: (discrete version derived from [He & Yu, 01])

Let ξ be a Markov chain. Define

$$\alpha_t = \underbrace{\sum_{x \notin X^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x)} - \underbrace{\sum_{x \in X^*} P(\xi_{t+1} \notin \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x)}.$$

Then ξ converges to $\mathcal{X}^*$ if and only if α satisfies:

$$P(\xi_0 \in \mathcal{X}^*) + \sum_{t=0}^{+\infty} \alpha_t = 1$$

Convergence

Does an EA converge to the global optimal solutions?

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

Considered as *closed*:

An EA that

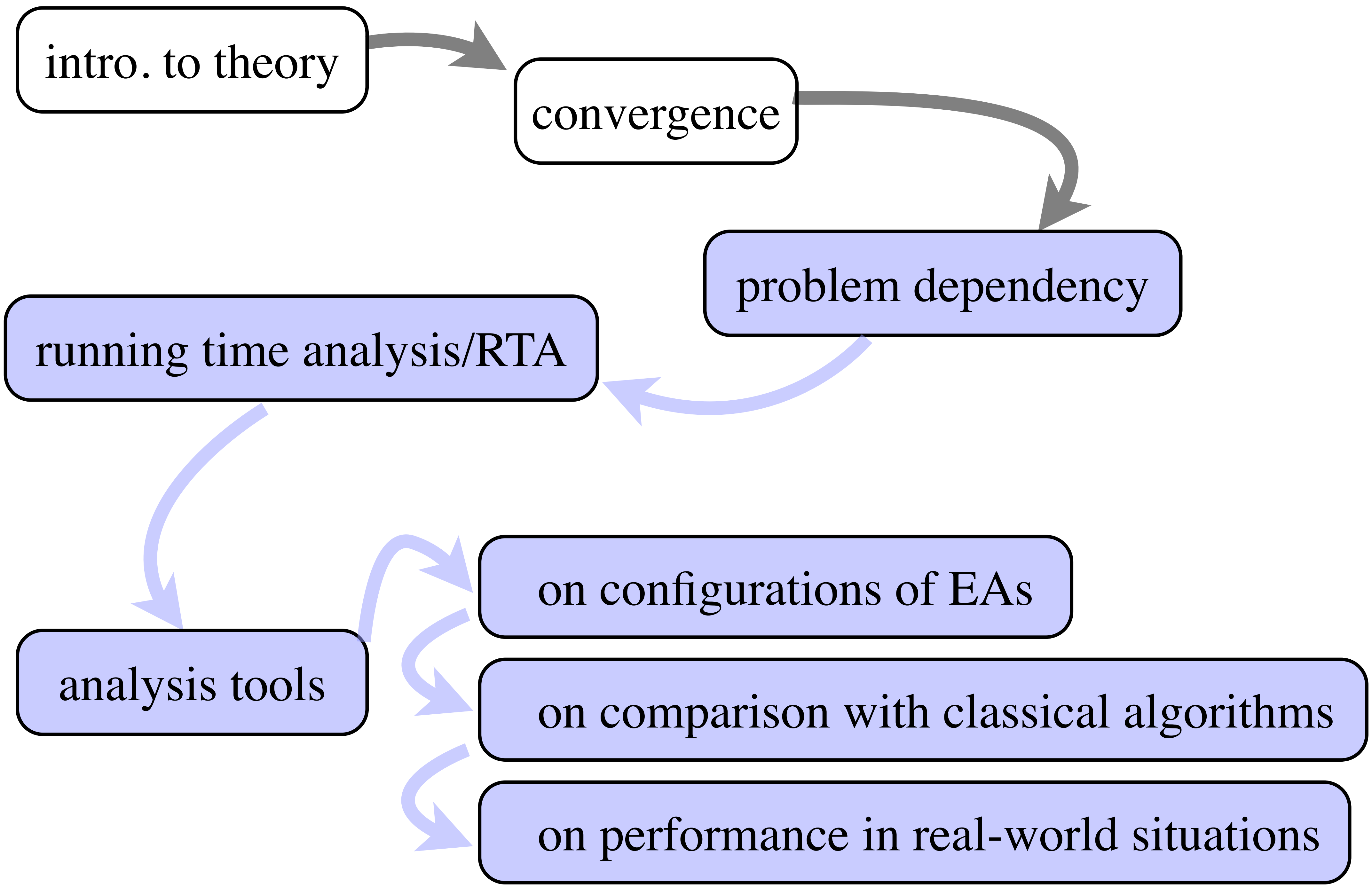
1. uses global operators gain of optimality > 0

2. preserves the best solution loss of optimality = 0

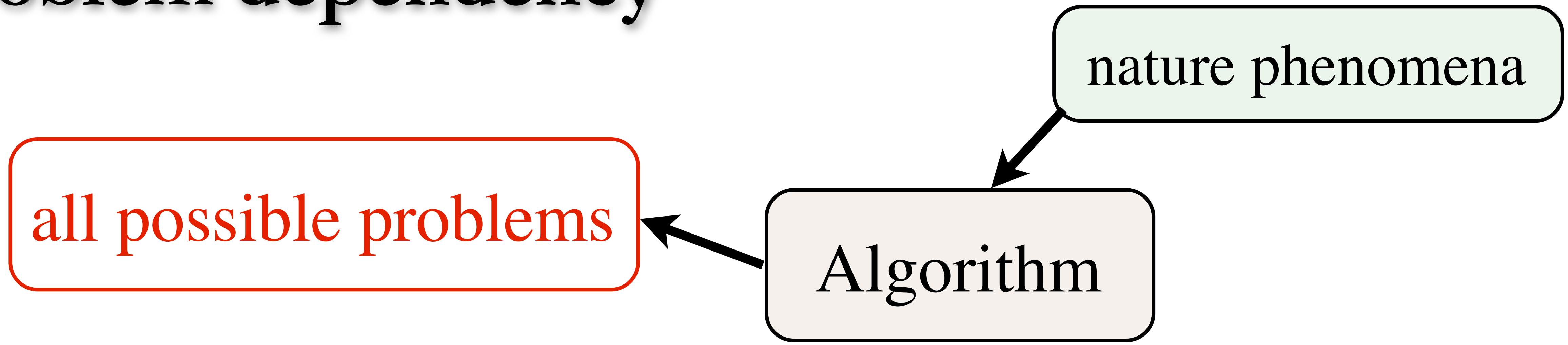
always converges to the optimal solutions

But life is limited! How fast does it converge?

Road map


intro. to theory
convergence
problem dependency
running time analysis/RTA
analysis tools
on configurations of EAs
on comparison with classical algorithms
on performance in real-world situations

Problem dependency



an EA A, objective f, m solutions

arbitrary measure of the objective values of the m solutions:

$$\Phi(\boldsymbol{y}_m \mid f, m, A)$$

Over all objectives $f : \mathcal{X}^m \rightarrow \{1, 2, \ldots, Y\}^m$

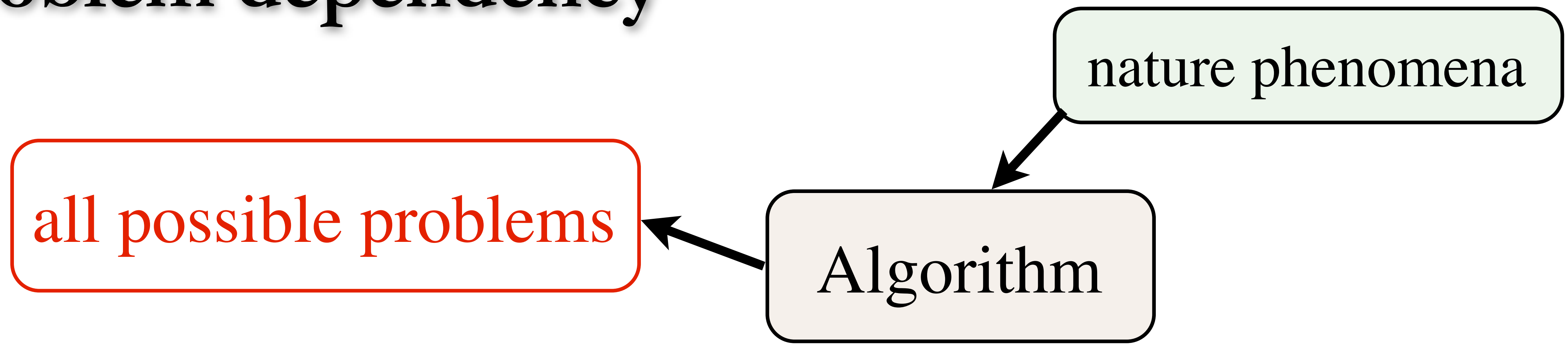
$$\sum_f I[k = \Phi(\boldsymbol{y}_m | f, m, A)] = \sum_f I[k = \Phi(f(A(m)))] = \sum_f \sum_{\boldsymbol{y}_m} I[k = \Phi(\boldsymbol{y}_m)] I[\boldsymbol{y}_m = f(A(m))]$$

$$= \sum_{\boldsymbol{y}_m} I[k = \Phi(\boldsymbol{y}_m)] \sum_f I[\boldsymbol{y}_m = f(A(m))] = \sum_{\boldsymbol{y}_m} I[k = \Phi(\boldsymbol{y}_m)] Y^{|\mathcal{X}|-m}$$

all algorithms have the same average performance

[Wolpert & Macready, 97]

Problem dependency

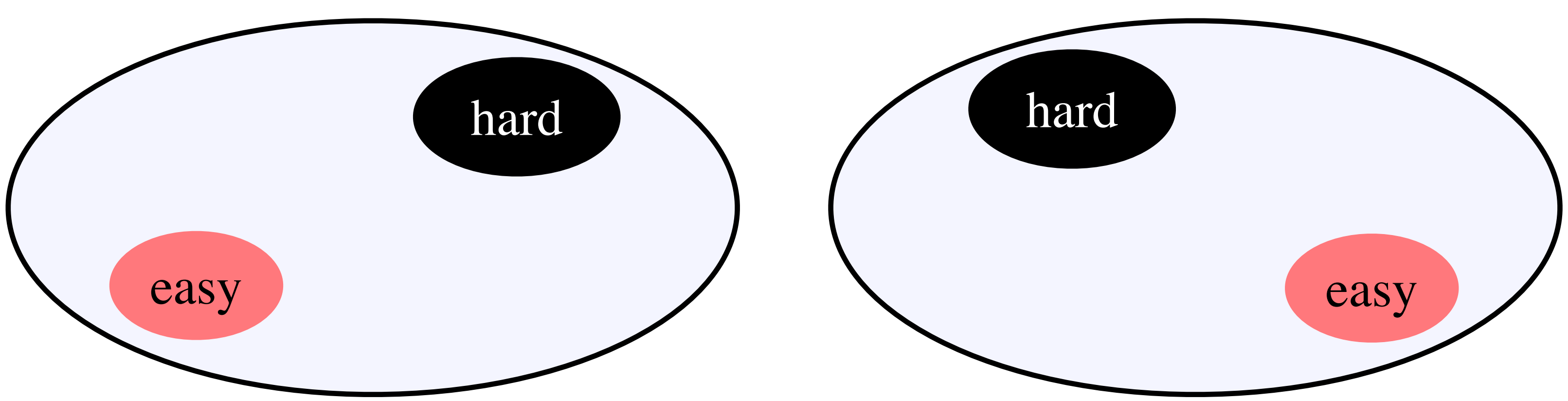


If:

- problem size n: the number of solutions is $\exp(n)$
- an EA with population size $\mathrm{poly}(n)$

then

$$\text{average time complexity } \Omega\left(\frac{\exp(n)}{\mathrm{poly}(n)}\right)$$ [Yu & Zhou, 08]



Road map

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

on performance in real-world situations

Examples in simple cases

Problem ← Algorithm → Measurement

Problem	Algorithm	Measurement
OneMax Linear Pseudo-Boolean Functions LeadingOnes LongPath ...	(1+1)-EA	Expected Running Time (ERT)

A simple EA: (1+1)-EA

An extremely simplified EA
missing some features of *real* EAs

no population

(1+1)-EA

1: $s \leftarrow$ a randomly drawn solution from $\mathcal{X}$
2: **for** t=1,2,... **do**
3: $\quad s' \leftarrow mutate(s)$
4: $\quad$ **if** $f(s') \geq f(s)$ **then**
5: $\quad\quad s \leftarrow s'$
6: $\quad$ **end if**
7: $\quad$ **terminate** if meets a stopping criterion
8: **end for**

one-bit mutation
randomly choose one bit and change its value

bitwise mutation
change every bit with some probability (e.g. $\frac{1}{n}$)

no crossover

for maximization, allow neutral changes

find an optimal solution

Running time analysis

Running time of an EA:

the number of ***solutions evaluated*** until reaching an optimal solution of the given problem for the ***first time***

the most time consuming step

may meet many times

Running time analysis:

running time with respect to the ***problem size*** (e.g. n)

computational complexity

the expected running time/ERT

e.g. $O(n^2)$ expected running time

ERT with high probability

e.g. $O(n \ln n)$ expected running time with probability at least $1 - \frac{1}{2^n}$

Probing problem

OneMax Problem: $\arg\max_{x \in \{0,1\}^n} \sum_{i=1}^{n} x_i$

count the number of 1 bits

fitness: $f(x) = \sum_{i=1}^{n} x_i$

EAs do not have the knowledge of the problems

only able to call $f(x)$

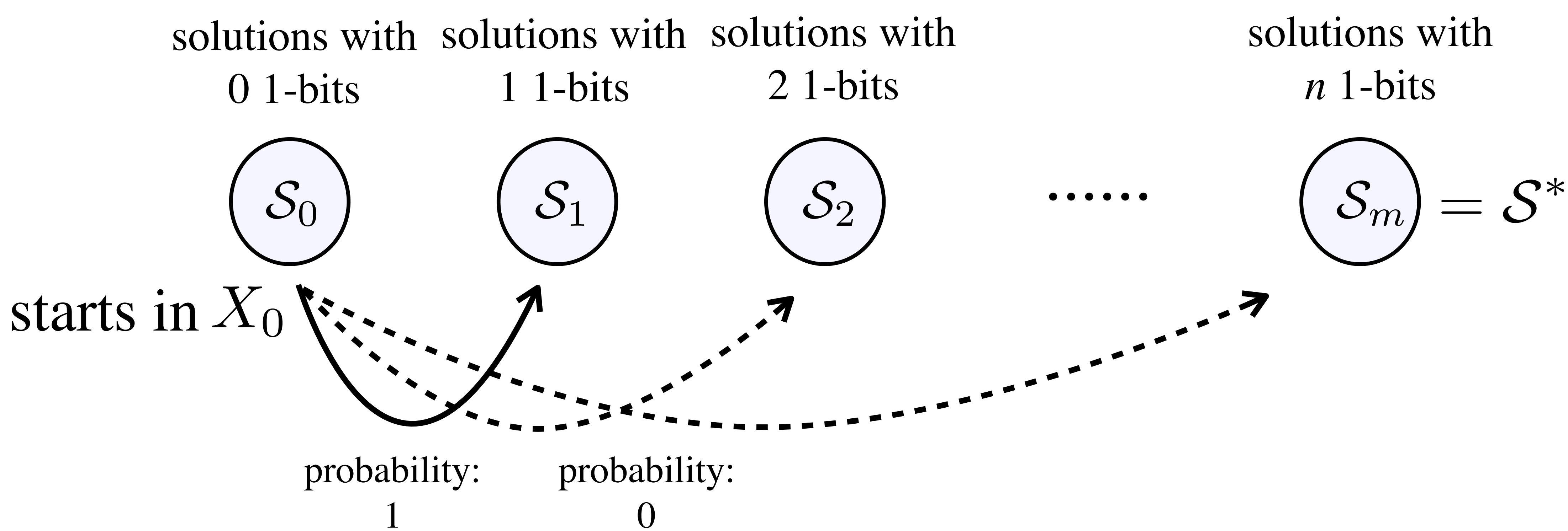
no difference with any other functions $f : \{0,1\}^n \to \mathbb{R}$

not only optimizing the problem, but also guessing the problem

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same f value

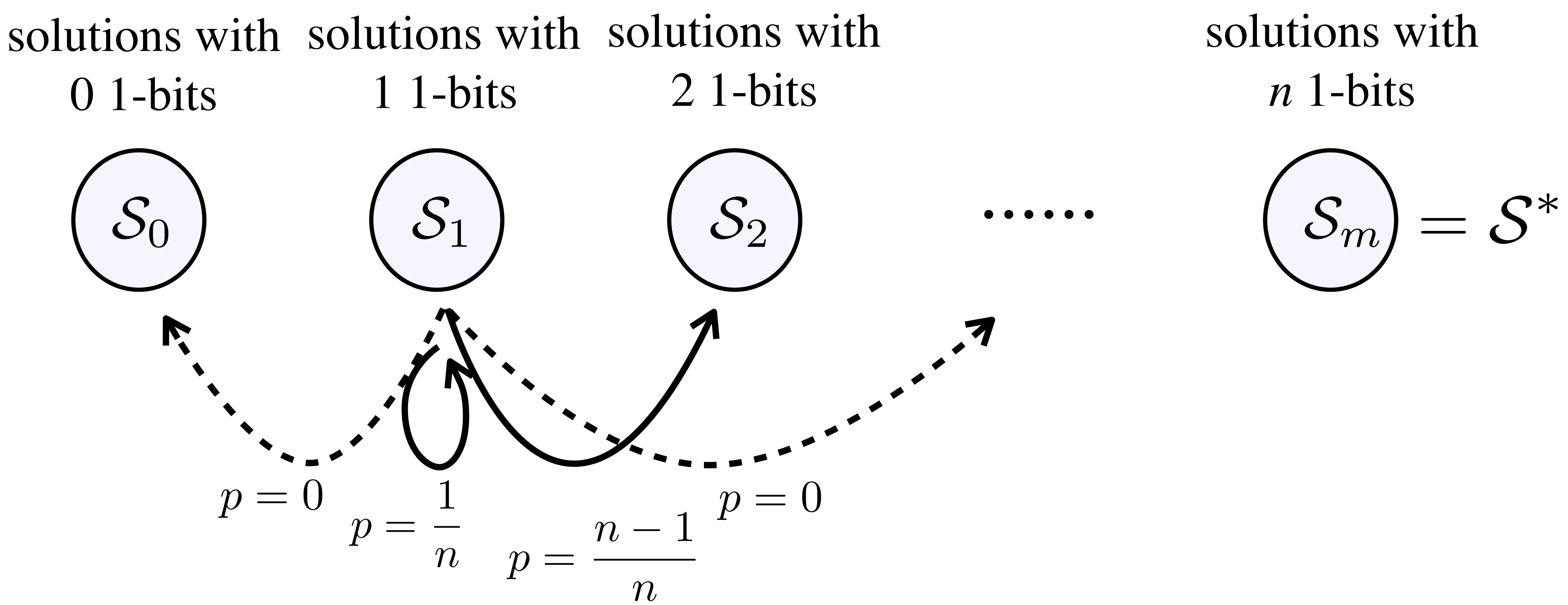


(1+1)-EA with one-bit mutation (Randomized Local Search):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same f value

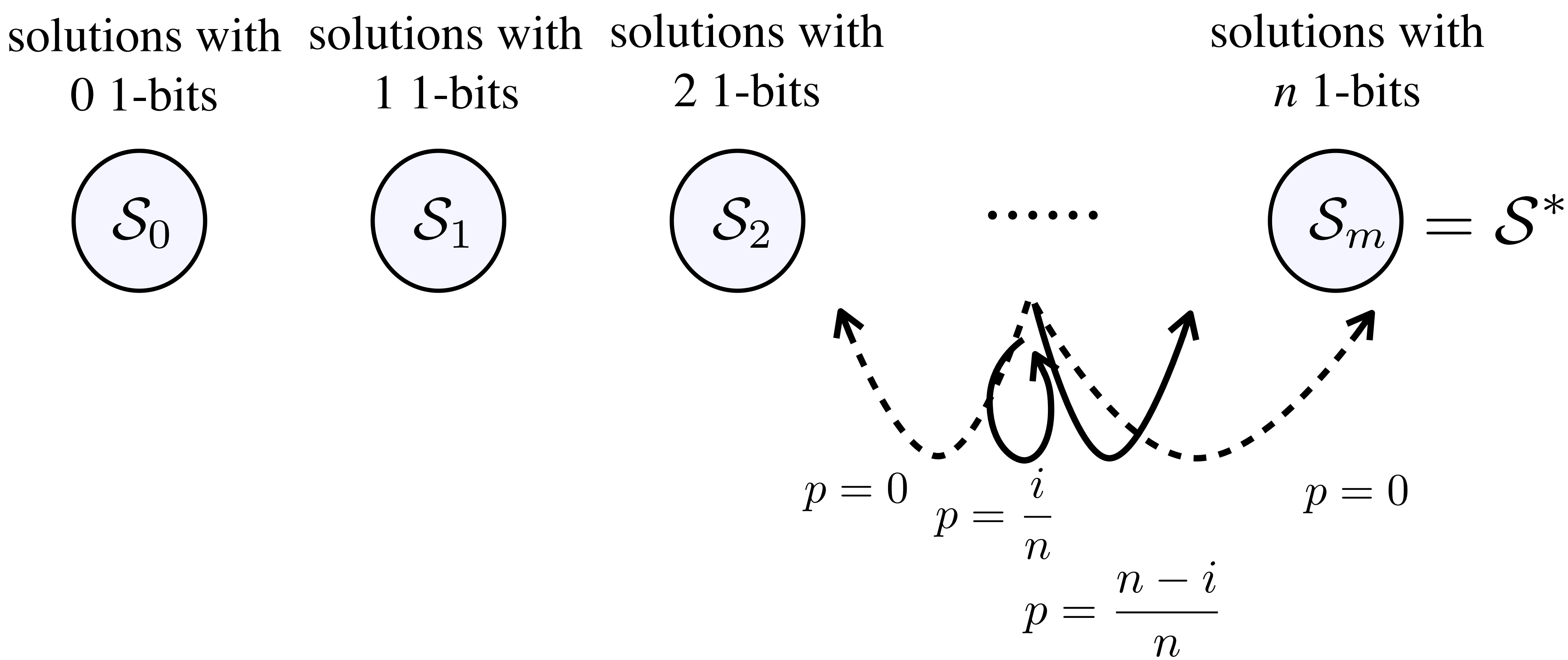


(1+1)-EA with <u>one-bit mutation</u> (Randomized Local Search):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same f value



(1+1)-EA with <u>one-bit mutation</u> (Randomized Local Search):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same *f* value

	solutions with 0 1-bits	solutions with 1 1-bits	solutions with 2 1-bits		solutions with *n* 1-bits
	$\mathcal{S}_0$	$\mathcal{S}_1$	$\mathcal{S}_2$	$\cdots\cdots$	$\mathcal{S}_m = \mathcal{S}^*$
probability of transition	$p=1$	$p=\frac{n-1}{n}$		$p=\frac{n-i}{n}$	$p=\frac{1}{n}$
expected #steps the transition happens	1	$\frac{n}{n-1}$		$\frac{n}{i}$	$\frac{n}{1}$

(1+1)-EA with <u>one-bit mutation</u> (Randomized Local Search):

ERT of (1+1)-EA in OneMax

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

(1+1)-EA with <u>one-bit mutation</u> (Randomized Local Search):

expected #steps the transition happens

$$\underbrace{1 \quad \frac{n}{n-1} \quad \cdots \quad \frac{n}{i} \quad \frac{n}{1}}$$

summed up $\sum_{i=1}^{n} \frac{n}{i} = nH_n \sim n \ln n$

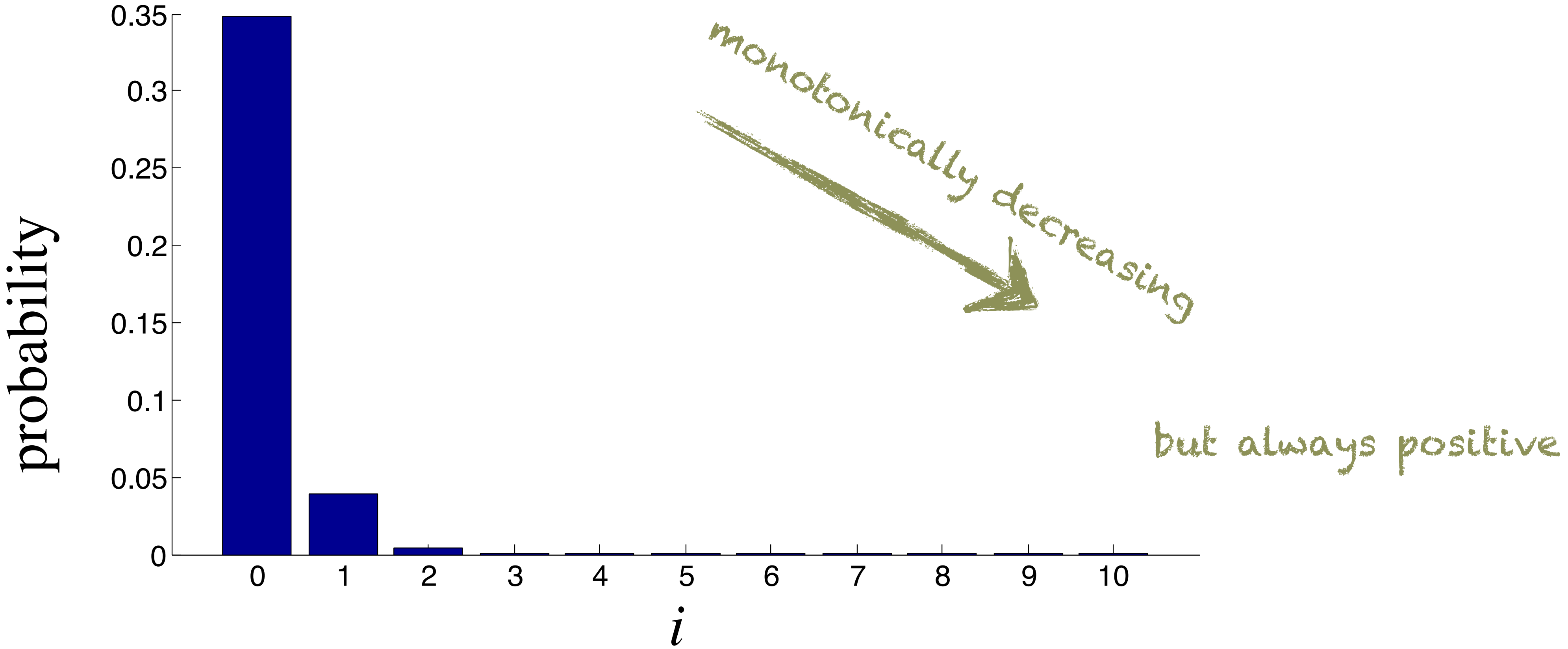
expected running time upper bound $O(n \ln n)$

ERT of (1+1)-EA in OneMax

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

(1+1)-EA with bitwise mutation (flip each bit with probability $\frac{1}{n}$):

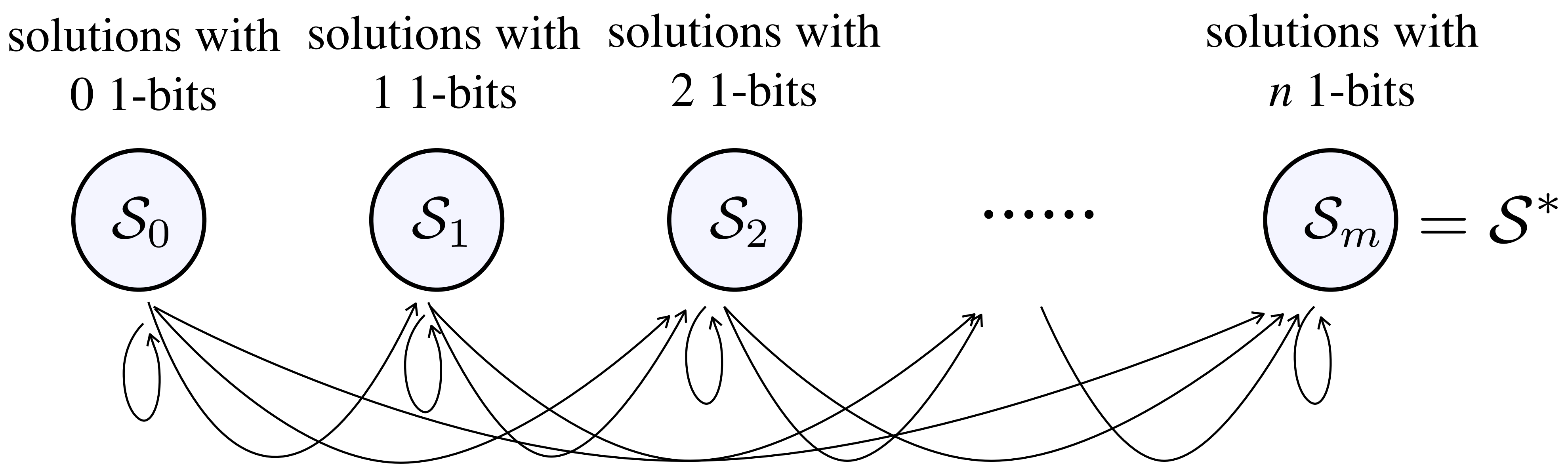
the probability of flipping i particular bits: $(\frac{1}{n})^i(1-\frac{1}{n})^{n-i}$



ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same *f* value



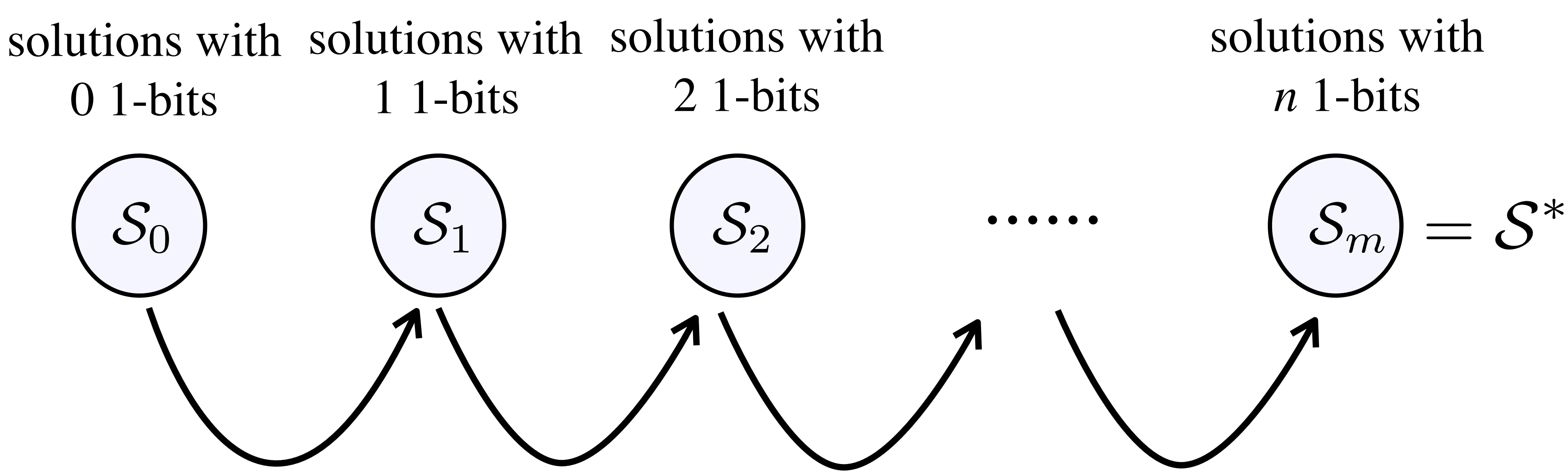
many transitions

(1+1)-EA with <u>bitwise mutation</u> (flip each bit with probability $\frac{1}{n}$):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same f value



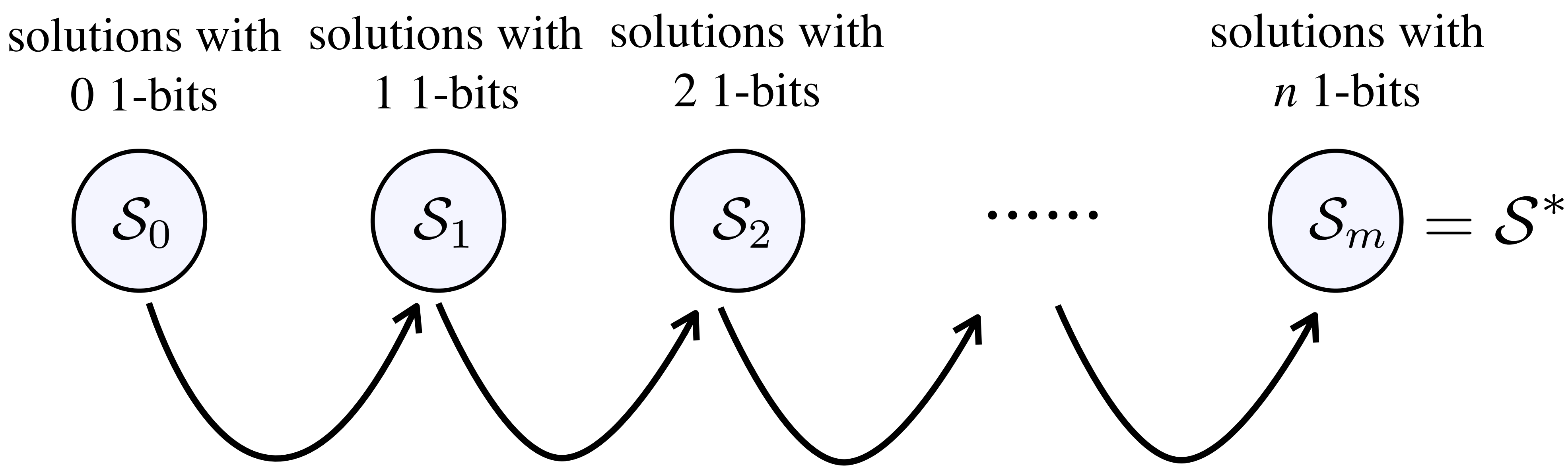
an upper bound: a path visits all subspaces

(1+1)-EA with bitwise mutation (flip each bit with probability $\frac{1}{n}$):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same f value



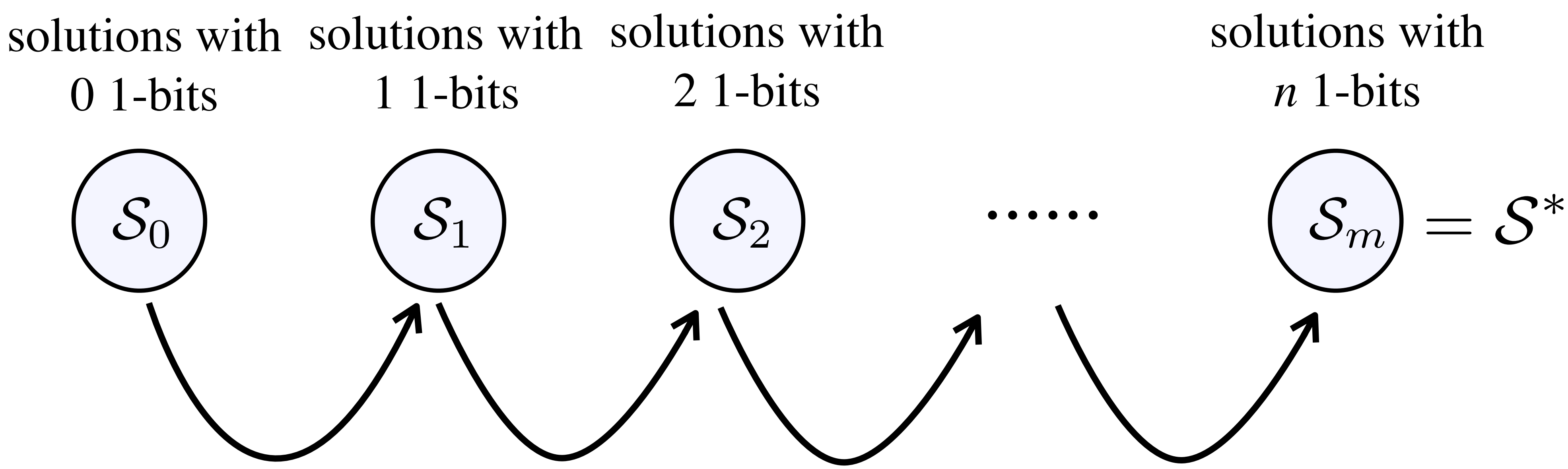
$$p = n(\frac{1}{n})(\frac{n-1}{n})^{n-1}$$

(1+1)-EA with <u>bitwise mutation</u> (flip each bit with probability $\frac{1}{n}$):

ERT of (1+1)-EA in OneMax

$$\text{OneMax:} \quad f(x) = \sum_{i=1}^{n} x_i$$

the solutions with the same number of 1-bits share the same *f* value



$$p = n(\frac{1}{n})(\frac{n-1}{n})^{n-1}$$

$$p \geq \binom{n-1}{1}(\frac{1}{n})(\frac{n-1}{n})^{n-1}$$

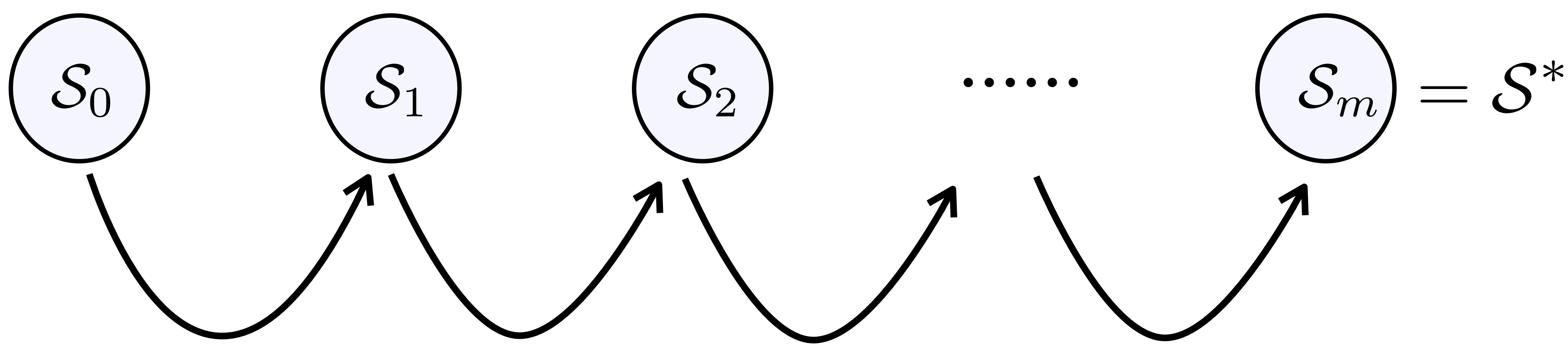
(1+1)-EA with <u>bitwise mutation</u> (flip each bit with probability $\frac{1}{n}$):

ERT of (1+1)-EA in OneMax

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

the solutions with the same number of 1-bits share the same *f* value

solutions with 0 1-bits | solutions with 1 1-bits | solutions with 2 1-bits | solutions with n 1-bits



$p = n(\frac{1}{n})(\frac{n-1}{n})^{n-1}$

$p \geq \binom{n-1}{1}(\frac{1}{n})(\frac{n-1}{n})^{n-1}$

$p \geq \binom{n-i}{1}(\frac{1}{n})(\frac{n-1}{n})^{n-1}$

(1+1)-EA with bitwise mutation (flip each bit with probability $\frac{1}{n}$):

ERT of (1+1)-EA in OneMax

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

(1+1)-EA with bitwise mutation (flip each bit with probability $\frac{1}{n}$):

probability of transition $\quad p \geq \binom{n-i}{1}(\frac{1}{n})(\frac{n-1}{n})^{n-1}$

expected #steps the transition happens $\quad \leq \frac{1}{n-i} \cdot n \cdot (1 + \frac{1}{n-1})^{n-1} \sim \frac{1}{n-i} \cdot n \cdot e$

summed up $\quad \sum_{i=0}^{n-1} \frac{en}{i} = enH_n \quad \sim en \ln n$

ERT upper bound $\quad O(n \ln n)$

ERT of (1+1)-EA in Linear Pseudo-Boolean Functions

Linear Pseudo-Boolean Functions: $\arg\max_{x \in \{0,1\}^n} \sum_{i=1}^{n} w_i x_i$

of which OneMax is a special case

where $w_i (\neq 0)$ are the weights

ERT of (1+1)-EA:

$$\Theta(n \ln n)$$ [Droste, et al. 98]

specially designed algorithm takes $\Theta(n)$ steps:

when not allowed to access the weight directly,

test every bit independently: $2n$ steps

recall that the EA does not have the knowledge about the problem

only a factor of $\ln n$ is paid for guessing the problem

Examples in simple cases

Problem ← Algorithm → Measurement

OneMax

Linear Pseudo-Boolean Functions

LeadingOnes

LongPath

...

(1+1)-EA

Expected Running Time (ERT)

probing problems help disclose properties of EAs

but EAs will not be used to solve these problems in practice

Road map

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

on performance in real-world situations

General analysis tools

running time analysis is commonly problem specific

going to derive the ERT of an EA in a problem

"where to look"

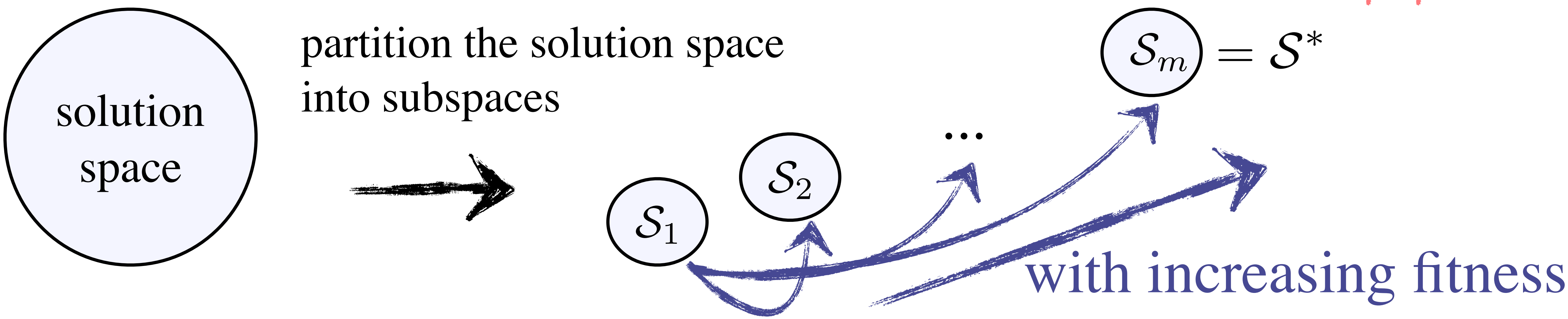
"what to calculate"

need a guide to tell *what to look* and *what to follow* to accomplish the analysis

- Fitness Level Method
- Drift Analysis
- Convergence-rate Based Method

Fitness Level Method [Wegener, 02]

a population is treated as the best solution in the population



Then calculate:

1. initialization probability of being in each subspace $\pi_0(\mathcal{S}_i)$
2. bounds of progress probability for $x \in \mathcal{S}_i$:

$$v_i \leq P(\xi_{t+1} \in \cup_{j=i+1}^{m} \mathcal{S}_j \mid \xi_t = x)$$
$$u_i \geq P(\xi_{t+1} \in \cup_{j=i+1}^{m} \mathcal{S}_j \mid \xi_t = x)$$

the ERT is then upper bounded by:

$$\sum_{1 \leq i \leq m-1} \pi_0(\mathcal{S}_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j}$$

and lower bounded by:

$$\sum_{1 \leq i \leq m-1} \pi_0(\mathcal{S}_i) \cdot \frac{1}{u_i}$$

Example in OneMax

partition

solutions with 0 1-bit	solutions with 1 1-bit	solutions with 2 1-bits		solutions with n 1-bits
$\mathcal{S}_1$	$\mathcal{S}_2$	$\mathcal{S}_3$	......	$\mathcal{S}_m = \mathcal{S}^*$

initialization distribution: $\pi_0(\mathcal{S}_i) = \dfrac{\binom{n}{i}}{2^n}$

progress probability for $x \in \mathcal{S}_i$:

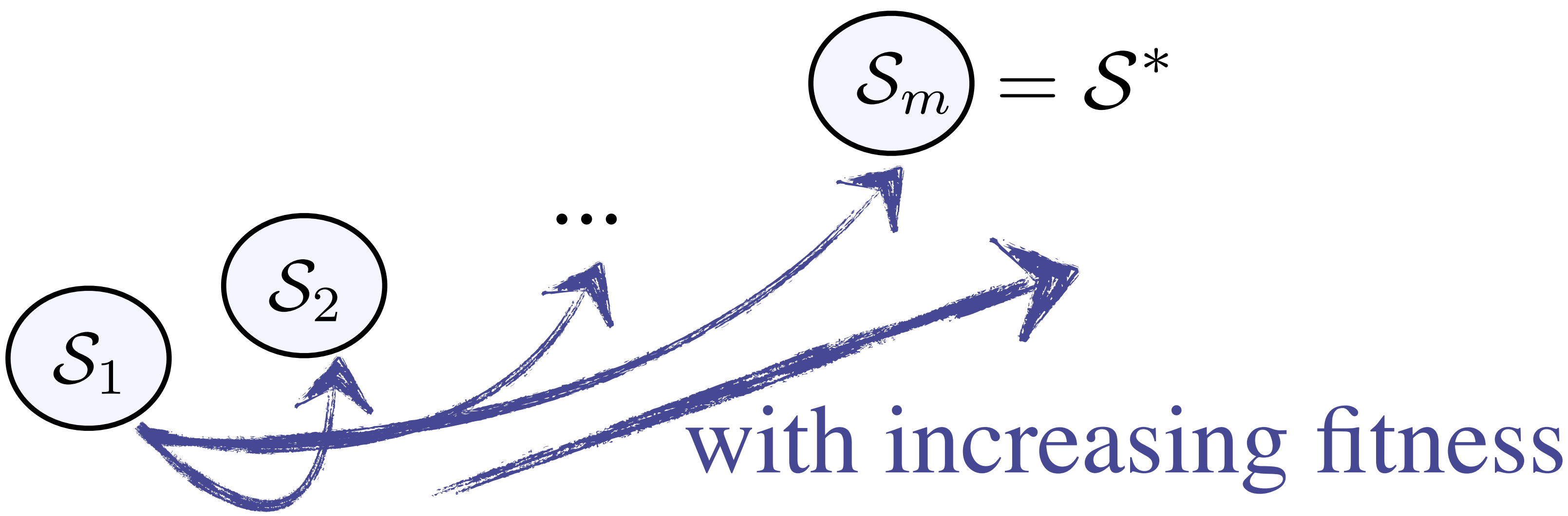
a lower bound: flipping one 0-bit but no 1-bits: $\binom{n-i}{1}(\frac{1}{n})(\frac{n-1}{n})^{n-1}$

ERT: $$\sum_{1 \le i \le m-1} \pi_0(\mathcal{S}_i) \cdot \sum_{j=i}^{m-1} \frac{1}{v_j} \le \pi_0(\mathcal{S}_0) \sum_{j=1}^{m-1} \frac{1}{v_j} \quad \in O(n \ln n)$$

Variants of Fitness Level Method

The fitness level method has been extended to derive tighter ERT bounds, by incorporating distribution of the transitions.

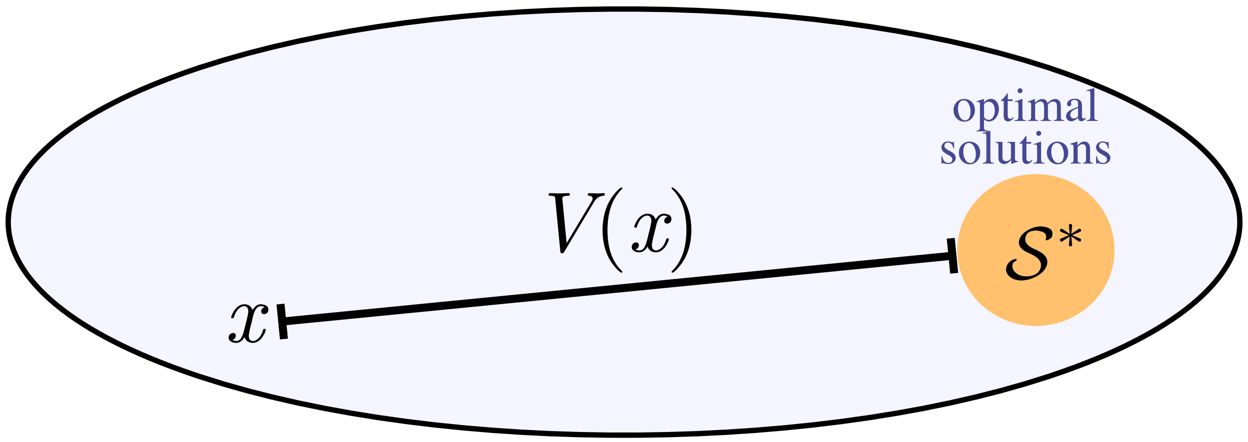
[Sudholt, 10] [Sudholt, 13]

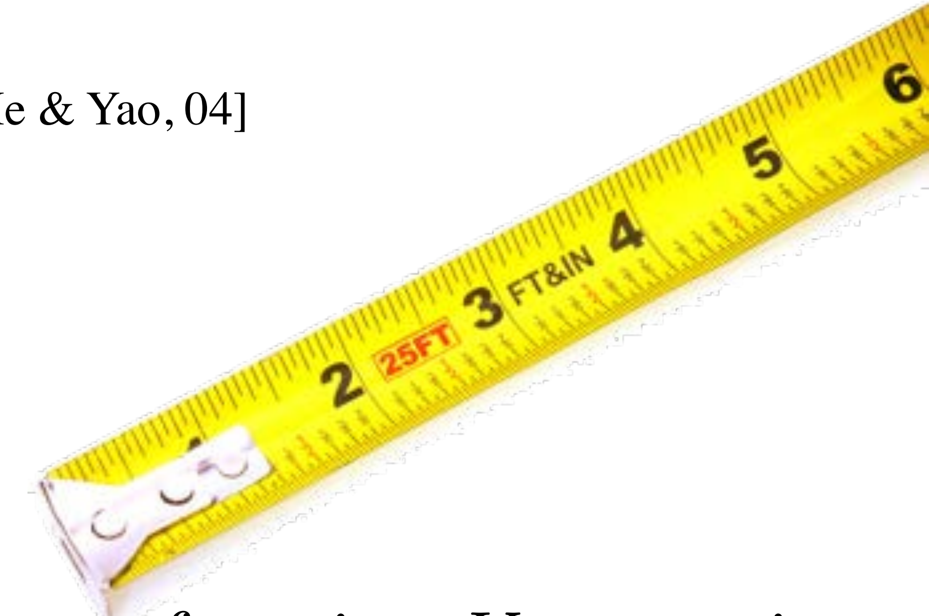


Incorporating tail bounds for sharp results. [Witt, 13]

Drift Analysis

[Hajek, 82][Sasaki & Hajek, 88][He & Yao, 01][He & Yao, 04]





distance function V measuring "distance" of a solution to optimal solutions. $V(x^*)=0$

Then calculate:

1. initialization probability of solutions $\pi_0(x)$
2. bounds of progress distance for every step:

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

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

the ERT is then upper bounded by:

$$\sum_{x\in\mathcal{X}} \pi_0(x)V(x)/c_l$$

and lower bounded by:

$$\sum_{x\in\mathcal{X}} \pi_0(x)V(x)/c_u$$

most simplified version

Example in LeadingOnes

LeadingOnes Problem: $\arg\max_{x\in\{0,1\}^n} \sum_{i=1}^{n}\prod_{j=1}^{i} x_i$

count the number of leading 1-bits

fitness: $f(x) = \sum_{i=1}^{n}\prod_{j=1}^{i} x_i$

$f(11011111) = 2$

Distance function: $V(x) = n - f(x)$

distance of optimal solutions is zero

The drift: $E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] =$

$$I(V(\xi_t) > V(\xi_{t+1}))E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] +$$

$$I(V(\xi_t) < V(\xi_{t+1}))E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] + \leftarrow \text{zero}$$

$$I(V(\xi_t) = V(\xi_{t+1}))E[V(\xi_t) - V(\xi_{t+1}) \mid \xi_t] \leftarrow \text{zero}$$

Only need to care the expected progress:

$\underbrace{11\ldots1}_{\text{keep}}\underset{\text{flip}}{0}\ldots\ldots$

probability of making progress >=
probability of increasing at least one leading 1-bit

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

Example in LeadingOnes

LeadingOnes Problem: $\arg\max_{x \in \{0,1\}^n} \sum_{i=1}^{n} \prod_{j=1}^{i} x_i$

fitness: $f(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_i$

count the number of leading 1-bits
f(11011111) = 2

Distance function: $V(x) = n - f(x)$

distance of optimal solutions is zero

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

ERT is then upper bounded as

$$\sum_{x \in \mathcal{X}} \frac{\pi_0(x) V(x)}{\frac{1}{en}} \leq \frac{V((00\ldots0))}{\frac{1}{en}} = \frac{n}{\frac{1}{en}} \in O(n^2)$$

the exact running time is approximate $0.86n^2$ [Böttcher, et al., 10]

Variants of Drift Analysis

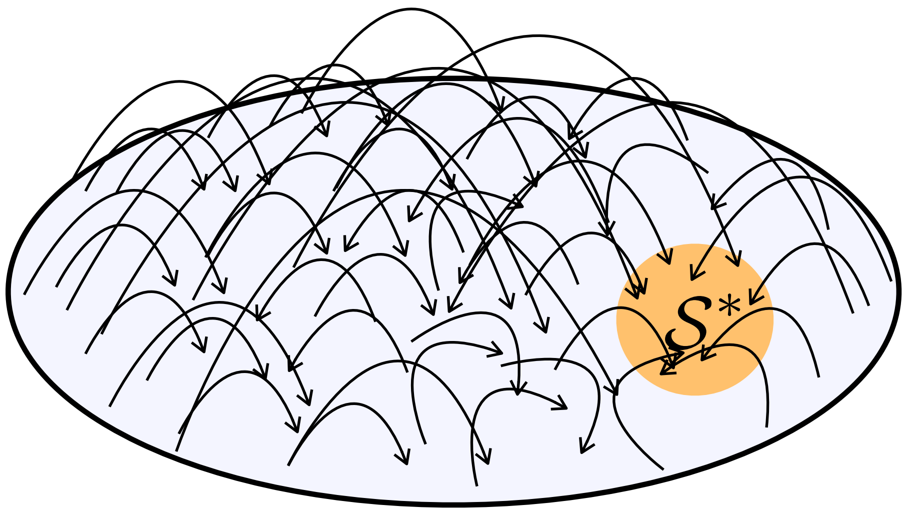
Other forms of drift analysis for better usability

[Happp, et al., 08] [Doerr, et al., 12] [Doerr & Goldberg, 13]

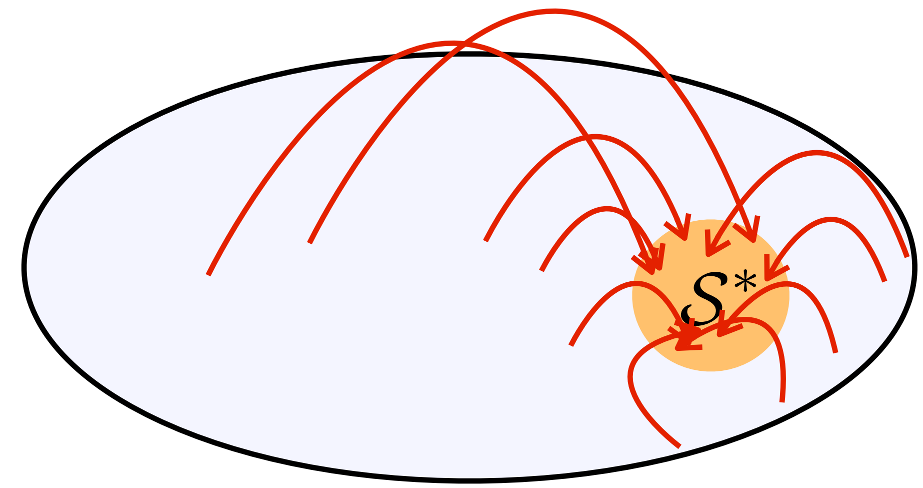
Incorporate tail bounds for sharp results

[Oliveto & Witt, 08] [Lehre & Witt, 13]

Convergence-rate Based Method [Yu & Zhou, 08]



only care about the reach at the optima



Then calculate:

bounds of getting optima for every step:

$$\alpha_t \leq \sum_{x \notin \mathcal{X}^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x \mid \xi_t \notin \mathcal{X}^*)$$

$$\beta_t \geq \sum_{x \notin \mathcal{X}^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x \mid \xi_t \notin \mathcal{X}^*)$$

the ERT is then upper bounded by:

$$\alpha_0 + \sum_{t=2}^{+\infty} t\alpha_{t-1} \prod_{i=0}^{t-2} (i - \alpha_i)$$

and lower bounded by:

$$\beta_0 + \sum_{t=2}^{+\infty} t\beta_{t-1} \prod_{i=0}^{t-2} (i - \beta_i)$$

Example in Trap

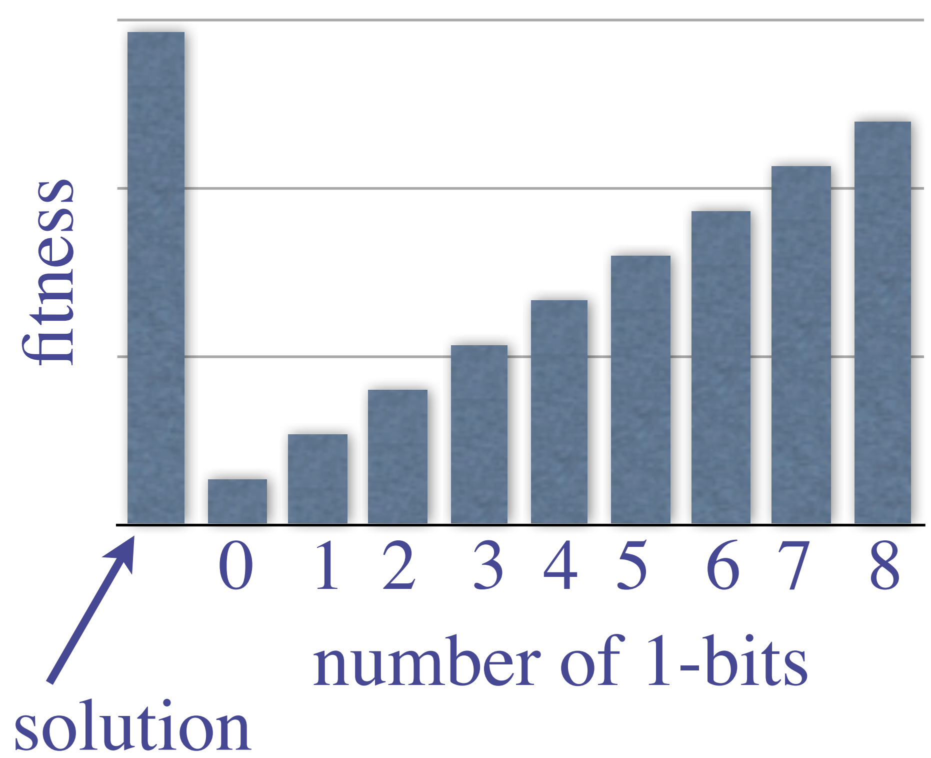
Trap Problem: $\arg\max_{x\in\{0,1\}^n} \sum_{i=1}^{n} w_i x_i$

$$\sum_{i=1}^{n} w_i x_i \leq C$$

constraint counting of 1-bits

where $w_1 = w_2 = \ldots = w_{n-1} > 1, w_n = C = 1 + \sum_{i=1}^{n-1}$

fitness: $f(x) = I[\sum_{i=1}^{n} w_i x_i \leq C] \sum_{i=1}^{n} w_i x_i - C$



for any solution with i bits different to the optimal solution

$$P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) = (\frac{1}{n})^i (1 - \frac{1}{n})^{n-i}$$

Example in Trap

Trap Problem:

fitness: $f(x) = I[\sum_{i=1}^{n} w_i x_i \leq C] \sum_{i=1}^{n} w_i x_i - C$

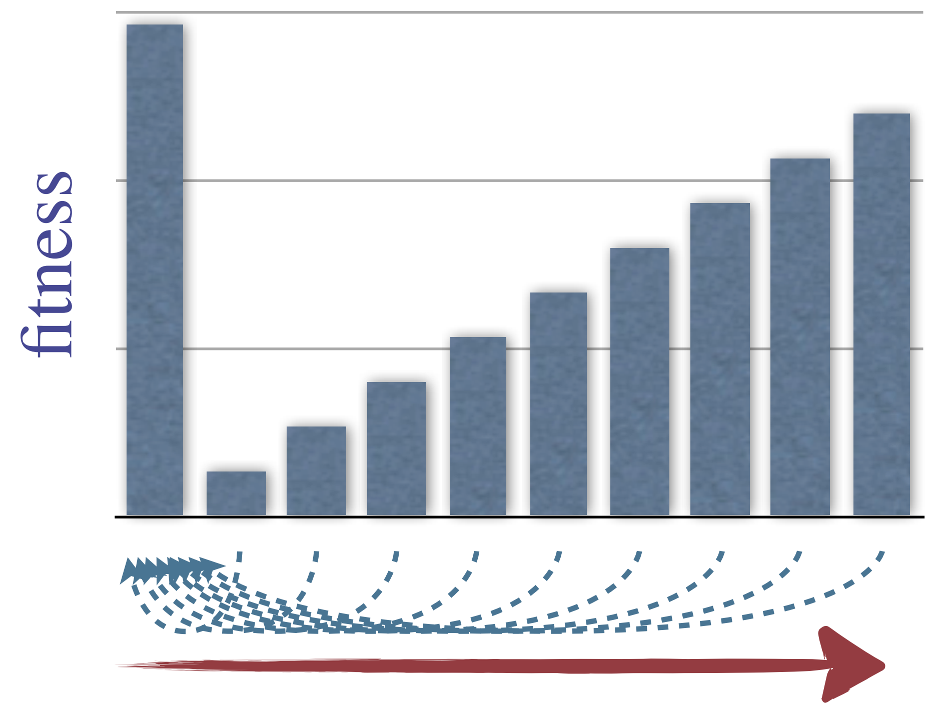
At the first step:

$$\sum_{x \notin \mathcal{X}^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x \mid \xi_t \notin \mathcal{X}^*)$$

$$= \sum_{i=0}^{n-1} \sum_{x \in \mathcal{X}_i} P(\xi_1 \in \mathcal{X}^* \mid \xi_0 = x) P(\xi_0 = x)$$

$$= \sum_{i=0}^{n-1} \binom{n}{i} \left(\frac{1}{n}\right)^{n-i} (1 - \frac{1}{n})^i \frac{1}{2^n}$$

$$= (1 - (\frac{n-1}{n})^n) \frac{1}{2^n} \sim \frac{e-1}{e} \frac{1}{2^n}$$



the distribution moves toward the wrong direction

In the later steps:

$$\sum_{x \notin \mathcal{X}^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x \mid \xi_t \notin \mathcal{X}^*) \leq \frac{e-1}{e} \frac{1}{2^n}$$

Example in Trap

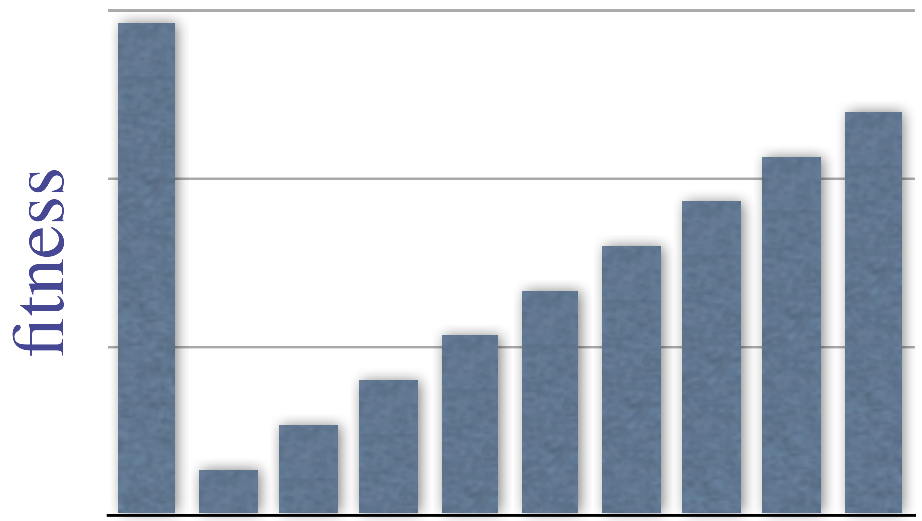
Trap Problem:

fitness: $f(x) = I[\sum_{i=1}^{n} w_i x_i \leq C] \sum_{i=1}^{n} w_i x_i - C$

$$\sum_{x \notin \mathcal{X}^*} P(\xi_{t+1} \in \mathcal{X}^* \mid \xi_t = x) P(\xi_t = x \mid \xi_t \notin \mathcal{X}^*) \leq \frac{e-1}{e} \frac{1}{2^n} = \beta_t$$

ERT is lower bounded by

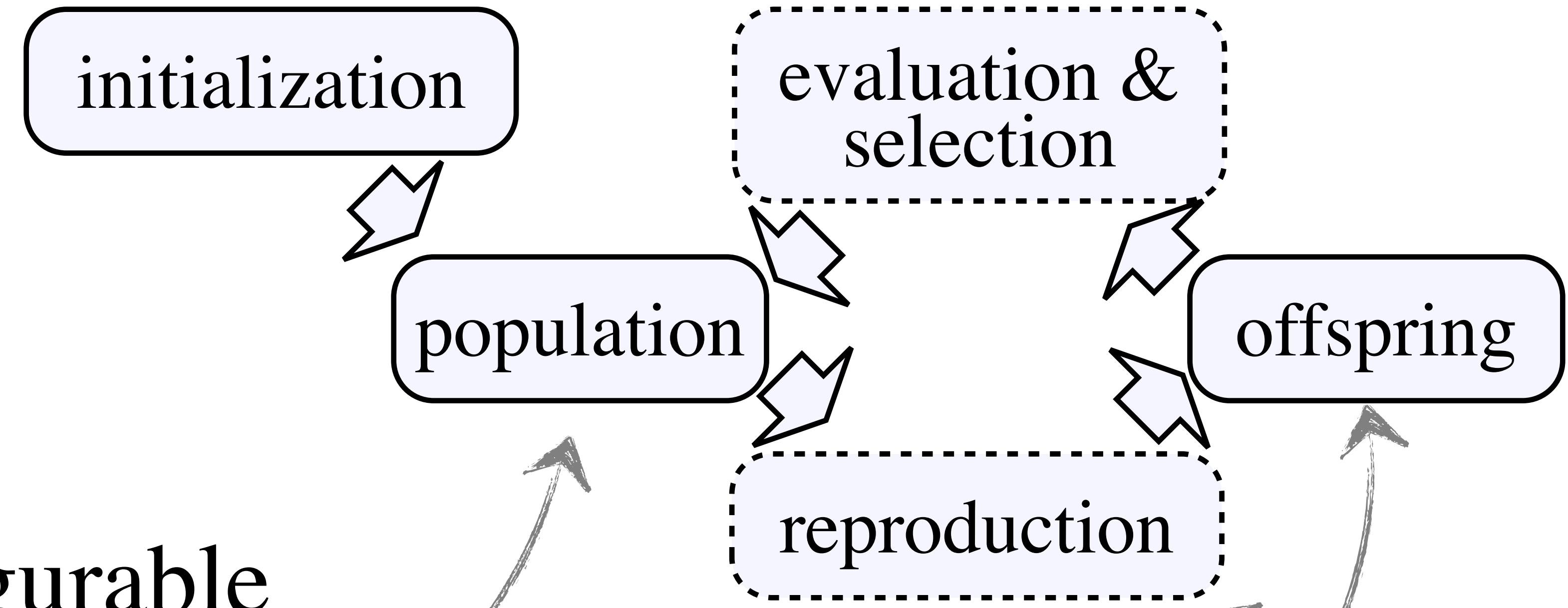
$$\beta_0 + \sum_{t=2}^{+\infty} t \beta_{t-1} \prod_{i=0}^{t-2} (i - \beta_i) = \frac{e}{e-1} 2^n \quad \in \Omega(2^n)$$



Road map

- intro. to theory
- convergence
- problem dependency
- running time analysis/RTA
- analysis tools
- on configurations of EAs
- on comparison with classical algorithms
- on performance in real-world situations

On configurations of EAs



A lot of parameters configurable

Is it good to maintain a population instead of a single solution?

Is it good to employ crossover for reproduction?

two characterizing features of EAs

On the effect of population

EAs maintaining a population of solutions:

(1+λ)-EA

1: $s \leftarrow$ a randomly drawn solution from $\mathcal{X}$
2: **for** t=1,2,... **do**
3: $Pop \leftarrow$ call $mutate(s)$ λ times
4: $s' \leftarrow$ the best solution in Pop
5: **if** $f(s') \geq f(s)$ **then**
6: $s \leftarrow s'$
7: **end if**
8: **terminate** if meets a stopping criterion
9: **end for**

(μ+1)-EA

1: $Pop = \{s_1, s_2, \ldots, s_\mu\} \leftarrow \mu$ randomly drawn solutions
2: **for** t=1,2,... **do**
3: $s \leftarrow$ select from Pop with probability proportional to the fitness
4: $s' \leftarrow mutate(s)$
5: $Pop \leftarrow$ select μ solutions from $Pop \cup s'$ with probability proportional to the fitness while keeping the best solution
6: **terminate** if meets a stopping criterion
7: **end for**

1 solution
λ offspring

μ solutions
1 offspring

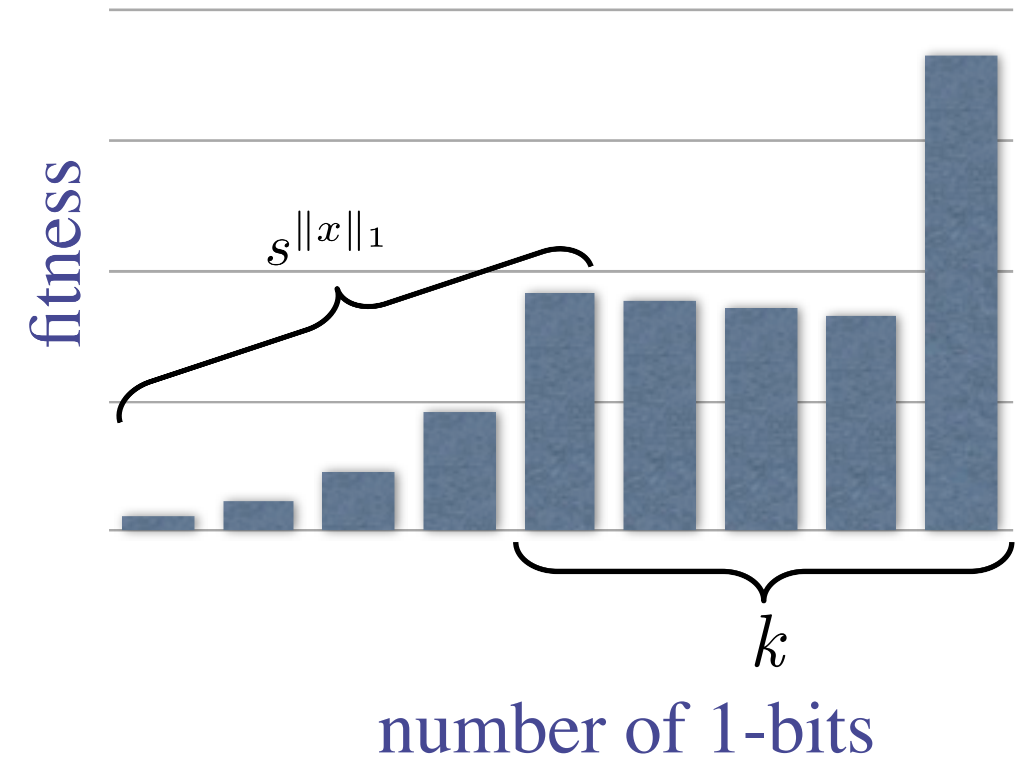
selection probability proportional to the fitness. fitness scale matters!

and also (N+N)-EA

On the effect of population

Can maintaining a population be beneficial?

[Jansen & Wegener, 01]: $\text{SJump}_{k,s}$ problem



Considering $k = \log n / \log\log n$, $s = n^2$

For (1+1)-EA

trapped at the local optimum

ERT: $O(n^{\log n/\log\log n})$

For (μ+1)-EA with $\mu = n$

probabilistic selection

spreads in the flat area

ERT: $O(n^{3/2})$

from super-polynomial to polynomial

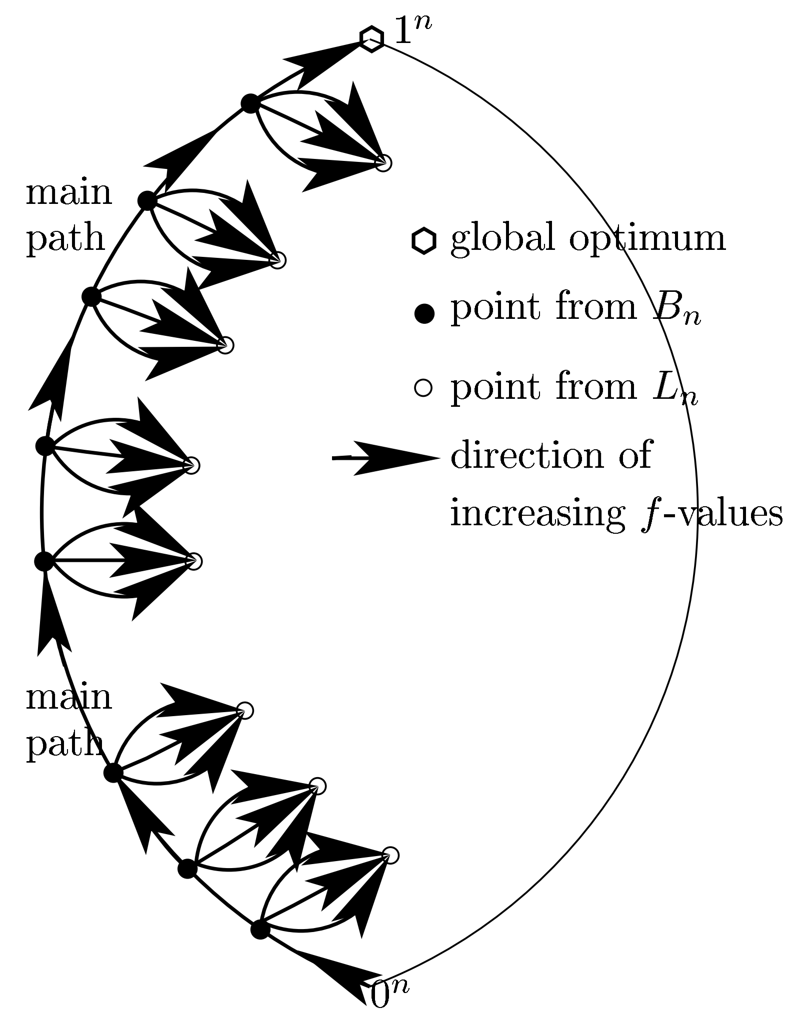
"parent population" with probabilistic selection helps spreading solutions

[Witt, 08]: from exponential to polynomial in an artificial problem

On the effect of population

Can maintaining a population be beneficial?

[Jansen, et al., 05]: SufSamp Problem



SufSamp [Jansen, et al., 05]

For (1+1)-EA

trapped at the local optimum

ERT: $n^{O(1)}$ with probability $2^{-\Omega(\sqrt{n}\log n)}$

For (1+λ)-EA with $\lambda = c \cdot n$

looking around before taking a step

follow the global path

ERT: $O(c^2 n^3)$

from super-polynomial to polynomial

"offspring population" enforces local search

On the effect of population

Is population always beneficial?

In OneMax problem:

known (1+1)-EA ERT upper bound $O(n \ln n)$

(μ+1)-EA ERT lower bound $\Omega(\sqrt{\mu} n \ln n + \mu n)$ [Storch, 08]

In LeadingOnes problem:

known (1+1)-EA ERT upper bound $O(n^2)$

(μ+1)-EA ERT lower bound $\Omega(\mu n \log n + n^2)$ [Witt, 06]

Similar results also found for

(1+λ)-EA [Jansen, et al., 05]

and (N+N)-EA [Chen, et al., 09]

in simple problems, population is not necessary

On the effect of population

Can population be harmful?

[Chen et al., 12]: TrapZeros Problem

For (1+1)-EA

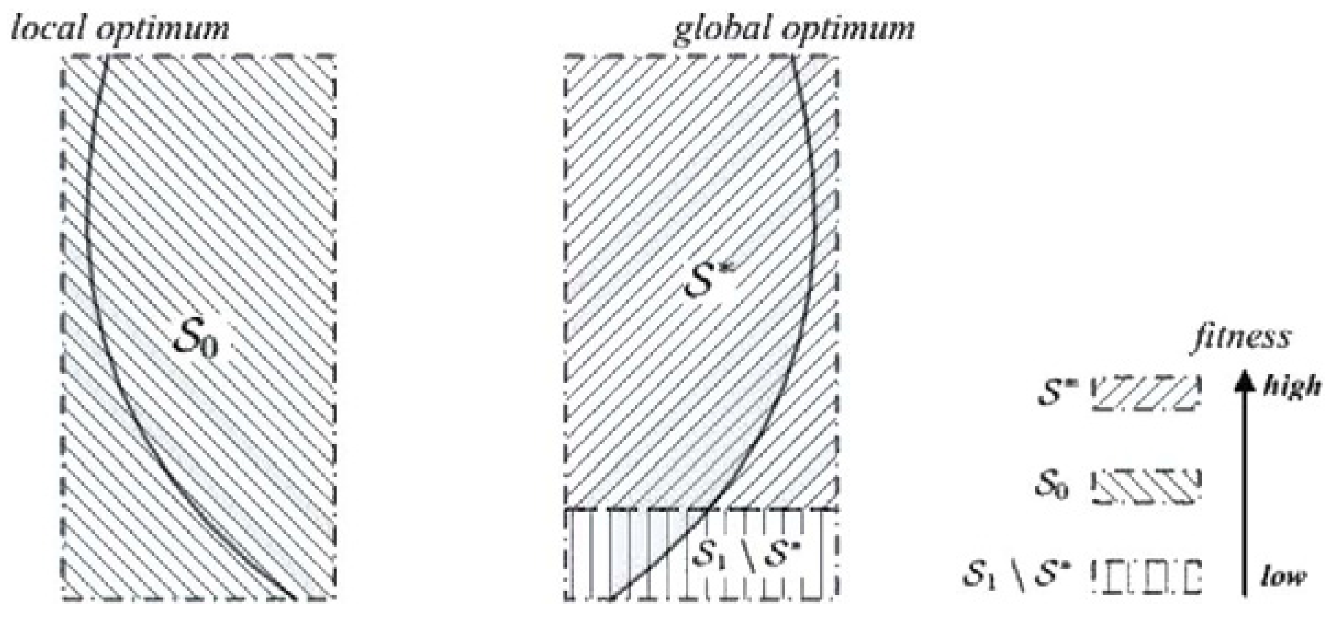
ERT: $O(n^2)$
with probability $\frac{1}{4} - O(\frac{\ln^2 n}{n})$

For (N+N)-EA

with N>1 and $N \in O(\ln n)$
ERT: $O(n^2)$
with probability $\frac{1}{poly(n)}$

For (N+N)-EA

with $N \in \Omega(n/\ln n)$
ERT is super-polynomial with an overwhelming probability



TrapZeros [Chen et al., 12]

too much selection pressure leads to over greedy

On the effect of crossover

to apply crossover, the EA has to maintain a population

(μ+1)-EA

$Pop = \{s_1, s_2, \ldots, s_\mu\} \leftarrow \mu$ randomly drawn solutions
for t=1,2,... **do**
  $s \leftarrow$ select from Pop with probability proportional to the fitness
  $s' \leftarrow mutate(s)$
  $Pop \leftarrow$ select μ solutions from $Pop \cup s'$ with probability proportional to the fitness while keeping the best solution
  terminate if meets a stopping criterion
end for

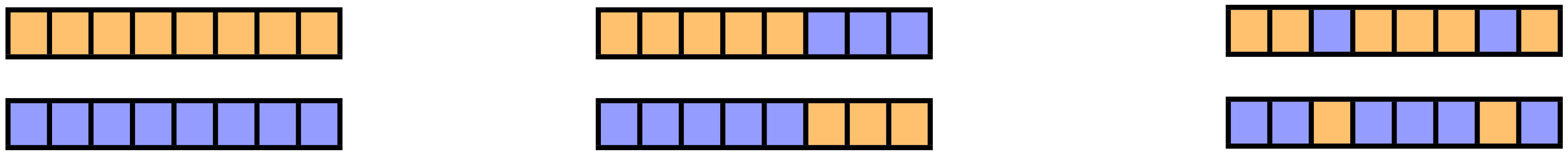
apply the crossover with a probability

(μ+1)-EA with crossover

$Pop = \{s_1, s_2, \ldots, s_\mu\} \leftarrow \mu$ randomly drawn solutions
for t=1,2,... **do**
  if within probability p_c **then**
    $s_1, s_2 \leftarrow$ select from Pop with probability proportional to the fitness
    $s \leftarrow$ a random outcome of $crossover(s_1, s_2)$
  else
    $s \leftarrow$ select from Pop with probability proportional to the fitness
  end if
  $s' \leftarrow mutate(s)$
  $Pop \leftarrow$ select μ solutions from $Pop \cup s'$ with probability proportional to the fitness while keeping the best solution
  terminate if meets a stopping criterion
end for

On the effect of crossover

crossover: operating on pairs of solutions



two solutions | one-point crossover exchange a part | uniform crossover exchange each bit with a prob.

irregularity of crossover

mutation: directly related to Hamming distance

crossover: ? distance

(11110000) + (11000011): generate 8 different outcomes

(11110000) + (11100001): generate 2 different outcomes

quadratic dynamic system [Rabani, et al., 98]

$$P(x) = \sum_{w,v,y} P(y)P(v)\Big(\frac{1}{2}P((x,w) \mid (y,v)) + \frac{1}{2}P((w,x) \mid (y,v))\Big)$$

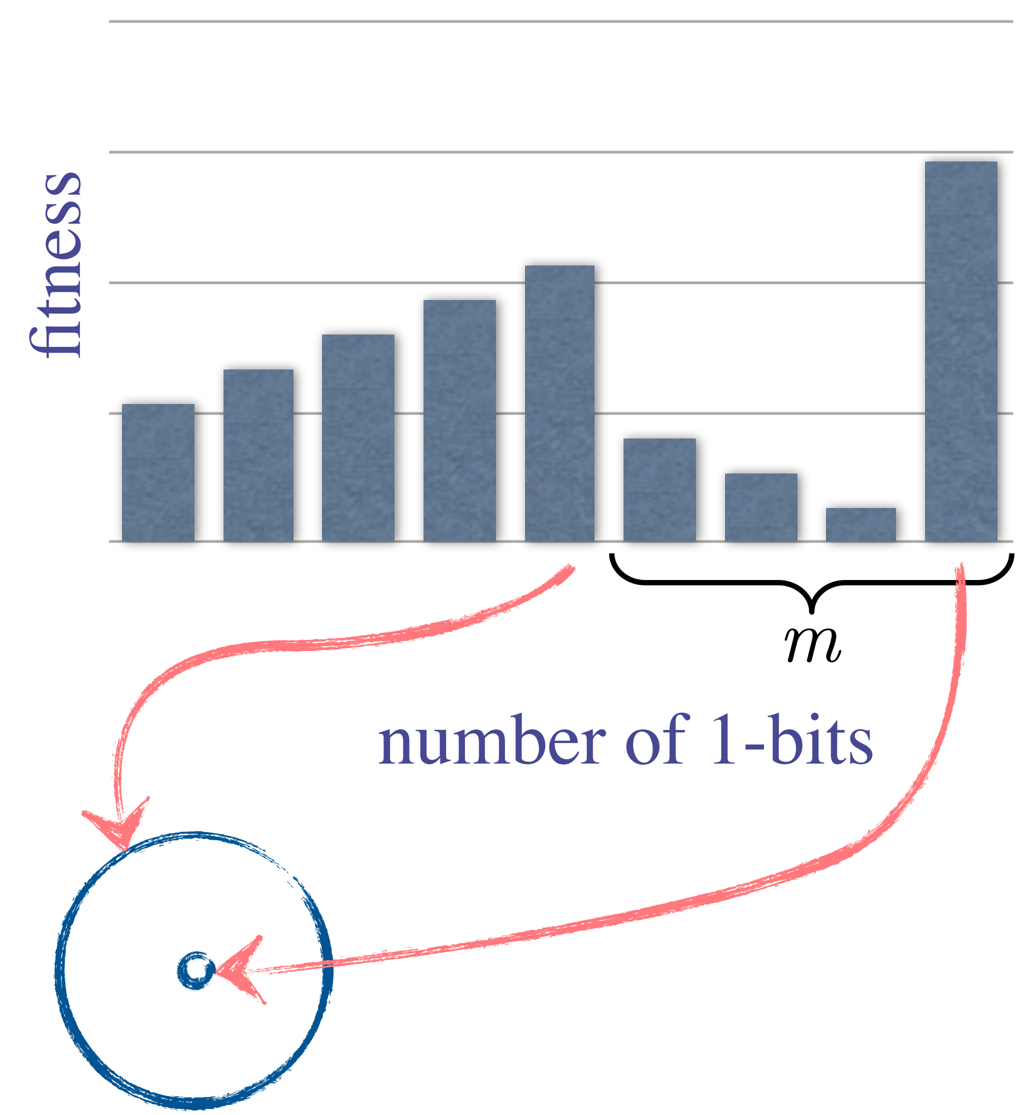
compare with that of Markov chain:

$$P(x) = \sum_{y} P(y)P(x \mid y)$$

studies without mutation or with pseudo-population

[Watson, 01] [Dietzfelbinger, et al., 03] [Kötzing, et al., 11]

On the effect of crossover



Can crossover be beneficial?

[Jansen & Wegener, 02]: Jump$_{n,m}$ Problem

Considering $m = \lceil \log n \rceil$

For (1+1)-EA

trapped at the local optimum

ERT: $\Theta(n^{\lceil \log n \rceil} + n \log n)$

For (μ+1)-EA with $\mu = \lceil \log^3 n \rceil$, small enough p_c, and avoid replicates

ERT: $O(n^3 \log n)$

from super-polynomial to polynomia

[Kötzing, et al., 11]: the results hold when without mutation after crossover

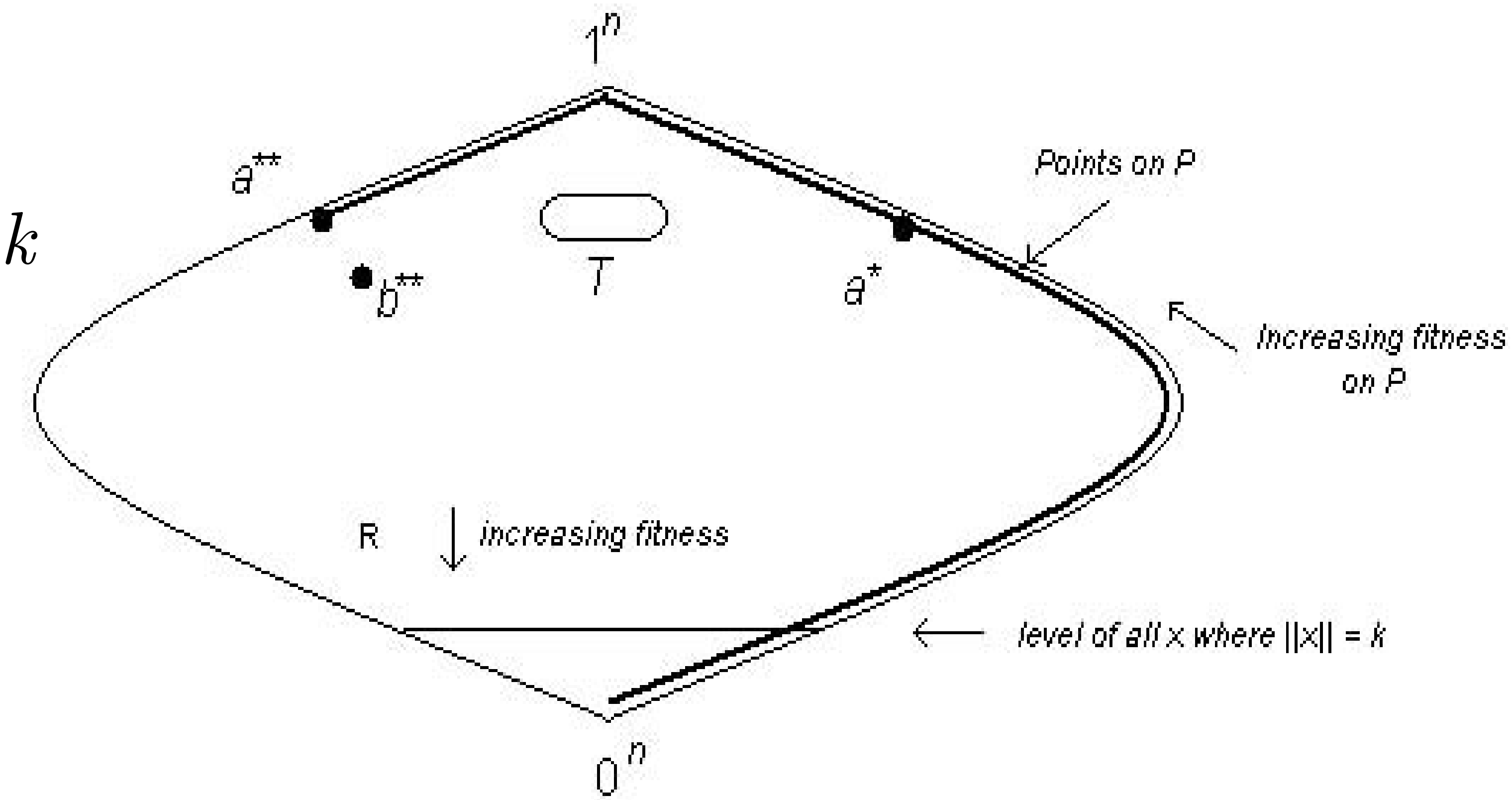
[Jansen & Wegener, 05]: Similar results in Real Royal Road Problem

On the effect of crossover

Can crossover be harmful?

[Richter, 08]: Ignoble Trails Problem

$||a^{**}, b^{**}|| = k$



For (2+1)-EA without crossover, ERT is $O(n^k)$

For (2+1)-EA with uniform crossover, ERT is exponential

Multi-objective optimization

optimizes multiple objectives simultaneously

$$\arg\max_x \boldsymbol{f}(x)$$

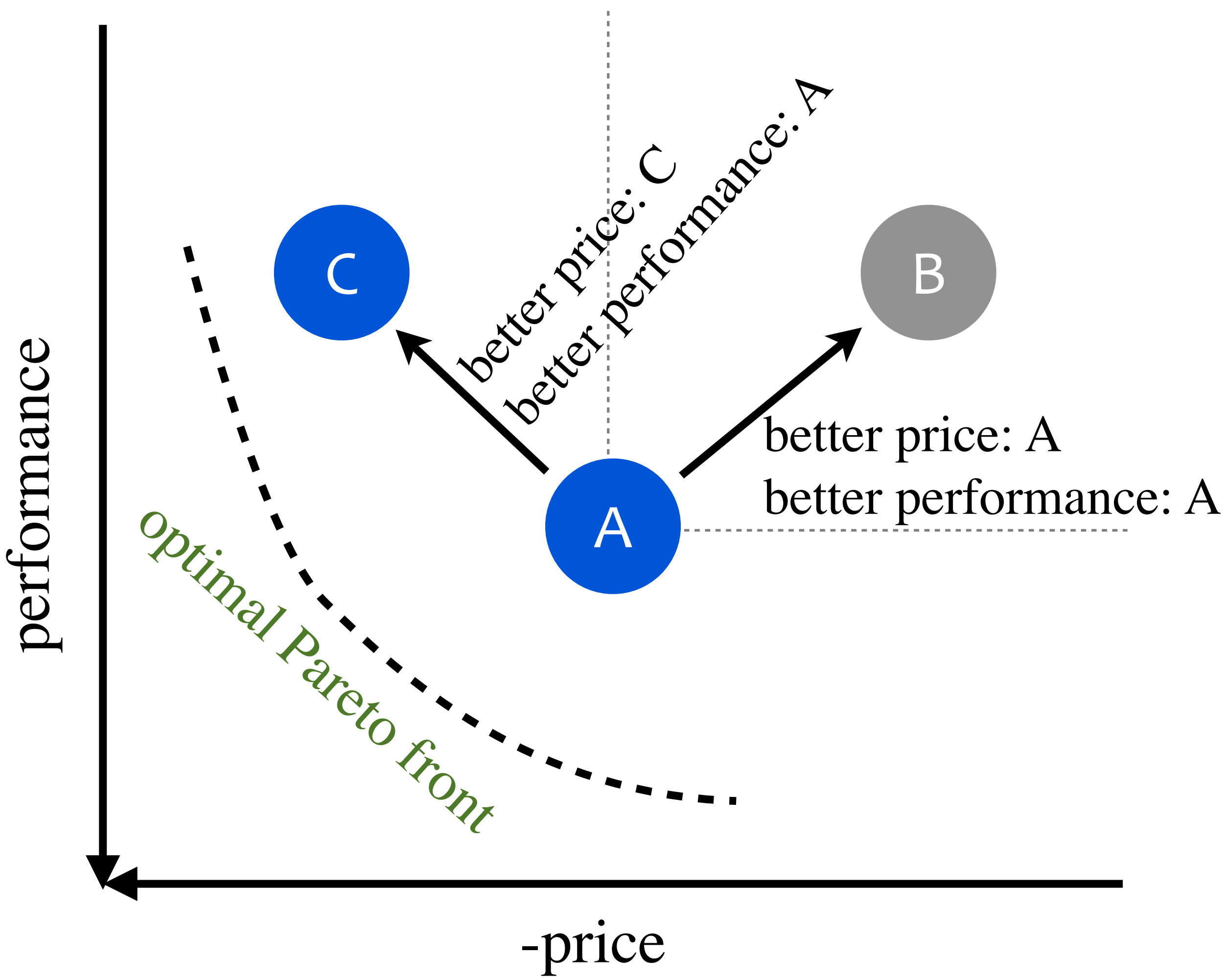
$$= \arg\max_x (f_1(x), \ldots, f_k(x))$$

[Laumanns, et al., 02]

A Simple Multi-objective EA (SEMO)

1: $Pop = \{s\} \leftarrow$ a randomly drawn solution
2: **for** t=1,2,... **do**
3: $s \leftarrow$ randomly select from Pop
4: $s' \leftarrow mutate(s)$
5: **if** $\nexists s'' \in Pop$ such that s'' dominates s' **then**
6: remove solutions in Pop that are dominated by s'
7: add s' into Pop
8: **end if**
9: **terminate** if meets a stopping criterion
10: **end for**

naturally maintain a population



A dominates B $f_{perf}(A) > f_{perf}(B)$
$f_{-price}(A) > f_{-price}(B)$

A and B are non-dominated $f_{perf}(A) < f_{perf}(C)$
$f_{-price}(C) > f_{-price}(A)$

On the effect of crossover

Can crossover be beneficial for multi-objective optimization?

[Neumann & Theile, 10]

crossover helps jump *gaps* in multi-criteria all-pairs-shortest-path problem

[Qian, et al., 11]

crossover helps fill the optimal Pareto front by recombining diverse solutions on the front, in COCZ and LOTZ problems

Can crossover be harmful for multi-objective optimization?

currently no evidence

On the effect of crossover

Other studies:

[Fischer & Wegener, 05]: studied crossover in Ising ring problems

[Sudholt, 05]: studied crossover in Ising tree problems

[Yu, et al., 10]: studied crossover in LeadingOnes problem

[Neumann, et al., 11]: studied crossover for parallel EAs

Road map

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

on performance in real-world situations

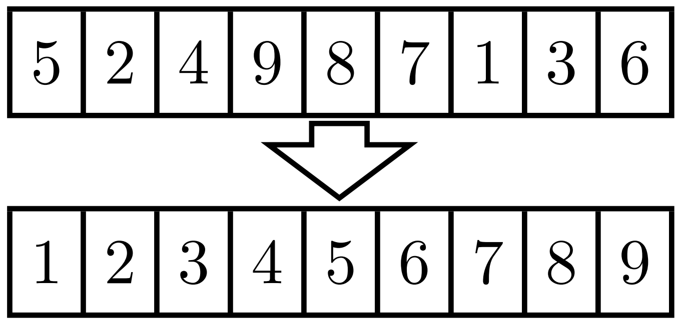
On comparison with classical algorithms

Sorting

Given: a sequence of numbers

Find: the sequence ordered ascendantly

complexity: $\Theta(n \ln n)$

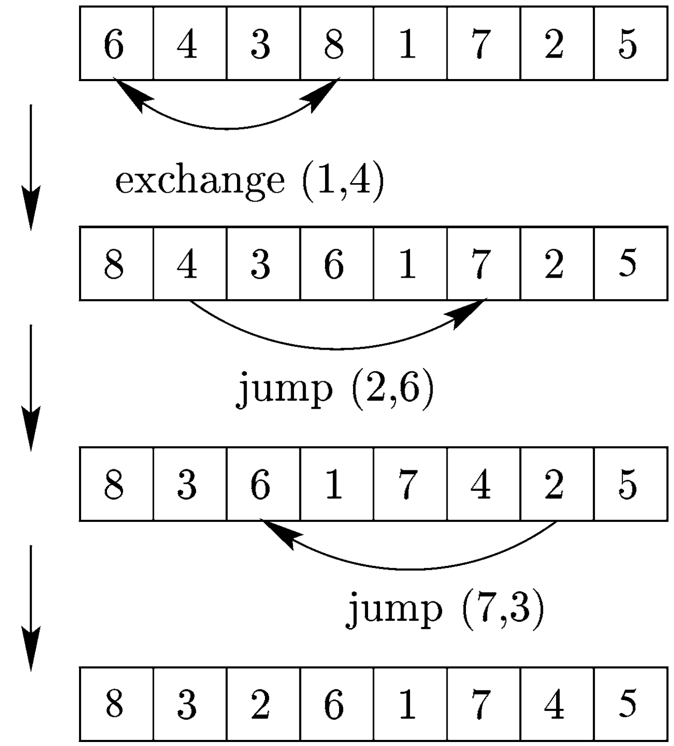


[Scharnow, et al., 04]:

Representation: an array of the numbers

Mutation: common mutation is not suitable;
exchange and *jump* operators

Fitness:
counting the number of sorted pairs
$O(n)$



Examples of mutations [Scharnow, et al., 04]

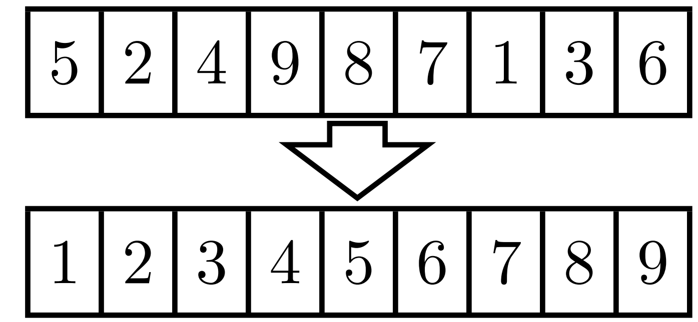
ERT of (1+1)-EA: $\Theta(n^2 \ln n)$

n^2 factor exploration, many redundant actions

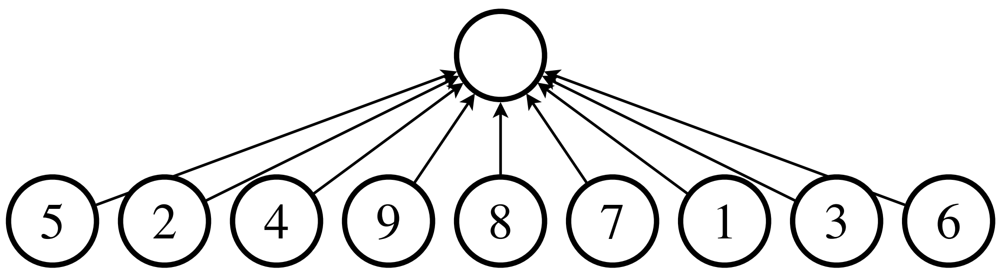
On comparison with classical algorithms

Sorting

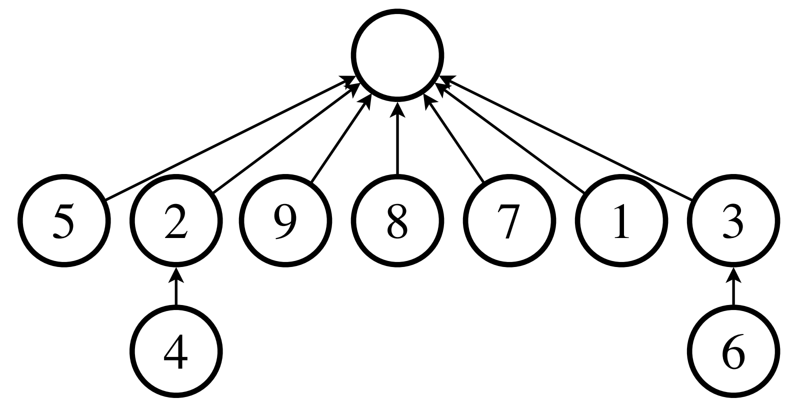
Given: a sequence of numbers
Find: the sequence ordered ascendantly
complexity: $\Theta(n \ln n)$

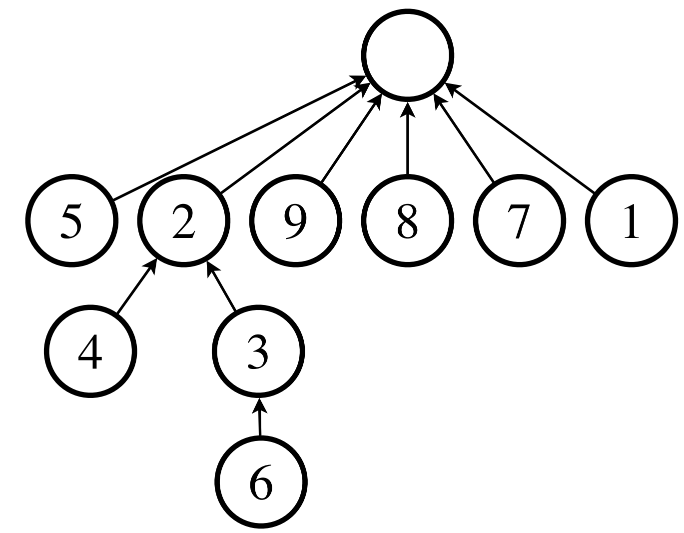


[Doerr & Happ, 08]: Directed tree representation



initial solution





Mutation: making two sibling nodes as parent-child
Fitness: count of corrected ordered pairs and strongly punish incorrectness
$O(1)$

ERT of (1+1)-EA: $O(n^2)$ $\Omega(n \ln n)$
empirical estimated ERT is in the order of $n \ln n$

On comparison with classical algorithms

Shortest Path (single source)

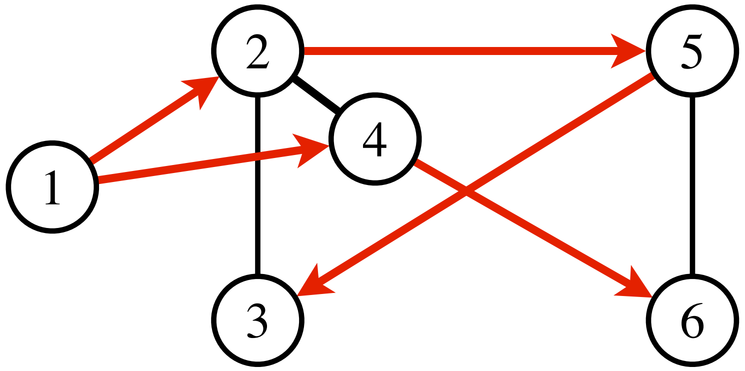
Given: a graph sequence of numbers

Find: the sequence ordered ascendantly

complexity: Dijkstra's algorithm $O(|V|^2)$

[Scharnow, et al., 04]:

Representation: an array indicating the predecessors of the index vertex



Mutation: randomly change the predecessor of some nodes

$$x = (\ 1\ 5\ 1\ 2\ 4\ )$$

index: 1 2 3 4 5 6

Fitness: multi-objectives, each objective measuring the path length from the source to a vertex

(1+1)-EA accepts solutions superior in all objectives

ERT of (1+1)-EA: $O(|V|^2 \max\{\ln|V|, \ell\})$ (ℓ is the radius *w.r.t*. the source)

[Doerr, et al., 11b]

On comparison with classical algorithms

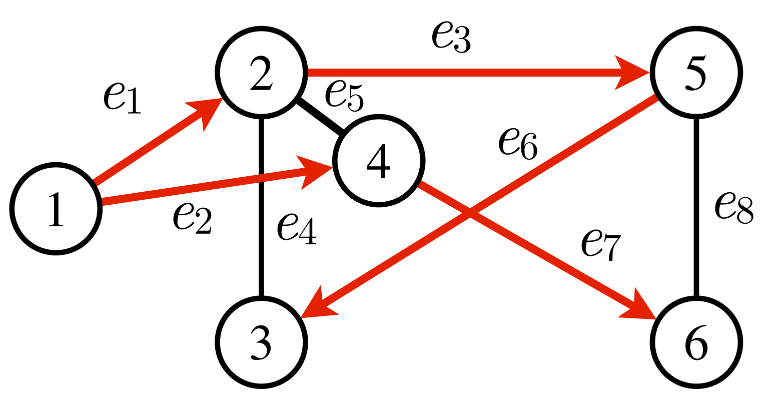
Shortest Path (single source)

Given: a graph sequence of numbers

Find: the sequence ordered ascendantly

complexity: Dijkstra's algorithm with Fibonacci heap

$$O(|E| + |V| \ln |V|)$$



$$x = (e_1 \;\; e_2 \;\; e_3 \;\; e_6 \;\; e_7)$$

[Doerr & Johannsen, 10]: Edge-based representation

Representation: an array indicating the selected edges

Mutation: replace a randomly chosen edge with another edge sharing the same end-vertex

Fitness: multi-objective, an objective measure the path length from the source to a node

ERT of (1+1)-EA: $O(|E| \max\{\ln |V|, \ell\})$ (ℓ is the radius *w.r.t*. the source)

On comparison with classical algorithms

	By EAs	By classical algorithms
All Pairs Shortest Path	$O(\|V\|^3 \ln \|V\|)$ [Doerr, et al., 13]	$\Theta(\|V\|^3)$ Floyd–Warshall algorithm
Maximum Matching	$O(\|E\|^{2\lceil 1/\epsilon \rceil})$ $(1+\epsilon)$-approximate [Giel & Wegener, 03]	$O(\sqrt{\|V\|}\|E\|)$ Hopcroft–Karp algorithm
Minimum Spanning Tree	$O(\|E\|^2(\ln \|V\| + \ln w_{\max}))$ [Neumann & Wegener, 07]	$O(\|E\| \cdot a(\|E\|, \|V\|))$ Chazelle's algorithm

...

On comparison with classical algorithms

[Doerr, et al., 11]: **EAs can do dynamic programming**

optimal substructures
overlapping subproblems

state space: $\mathcal{S}$ state transition func.: $\mathcal{F}_1, \ldots, \mathcal{F}_n$ consistency functions: $H_1, \ldots, H_n$

DP problem:

$$\mathcal{S}_0 \xrightarrow[H_1]{F \in \mathcal{F}_1} \cdots \longrightarrow \mathcal{S}_{i-1} \xrightarrow[H_i]{F \in \mathcal{F}_i} \mathcal{S}_i \longrightarrow \cdots \xrightarrow[H_n]{F \in \mathcal{F}_n} \mathcal{S}_n$$

contains initial states

contains states transited from the predecessor $\mathcal{S}_{i-1}$ by a function in $\mathcal{F}_i$, and the feasibility is checked by H_i

DP algorithm:

$$\mathcal{T}_0 \in \mathcal{S}_0 \longrightarrow \cdots \longrightarrow \mathcal{T}_i \in \mathcal{S}_i \rightarrow \mathcal{T}_{i-1} \in \mathcal{S}_{i-1} \longrightarrow \cdots \longrightarrow \mathcal{T}_n \in \mathcal{S}_n$$

single source shortest path:

- state space: a sequence of vertices with length at most n, and starts with s (source)
- initial states: $\{s\}$
- state transition functions: each function adds a vertex to the given sequence
- consistency: return feasible if the sequence is a path

On comparison with classical algorithms

[Doerr, et al., 11]: **EAs can do dynamic programming**

optimal substructures
overlapping subproblems

state space: $\mathcal{S}$ state transition func.: $\mathcal{F}_1, \ldots, \mathcal{F}_n$ consistency functions: $H_1, \ldots, H_n$

DP problem:

$$\mathcal{S}_0 \xrightarrow[H_1]{F \in \mathcal{F}_1} \cdots \longrightarrow \mathcal{S}_{i-1} \xrightarrow[H_i]{F \in \mathcal{F}_i} \mathcal{S}_i \longrightarrow \cdots \xrightarrow[H_n]{F \in \mathcal{F}_n} \mathcal{S}_n$$

contains initial states

contains states transited from the predecessor $\mathcal{S}_{i-1}$ by a function in $\mathcal{F}_i$, and the feasibility is checked by H_i

DP algorithm:

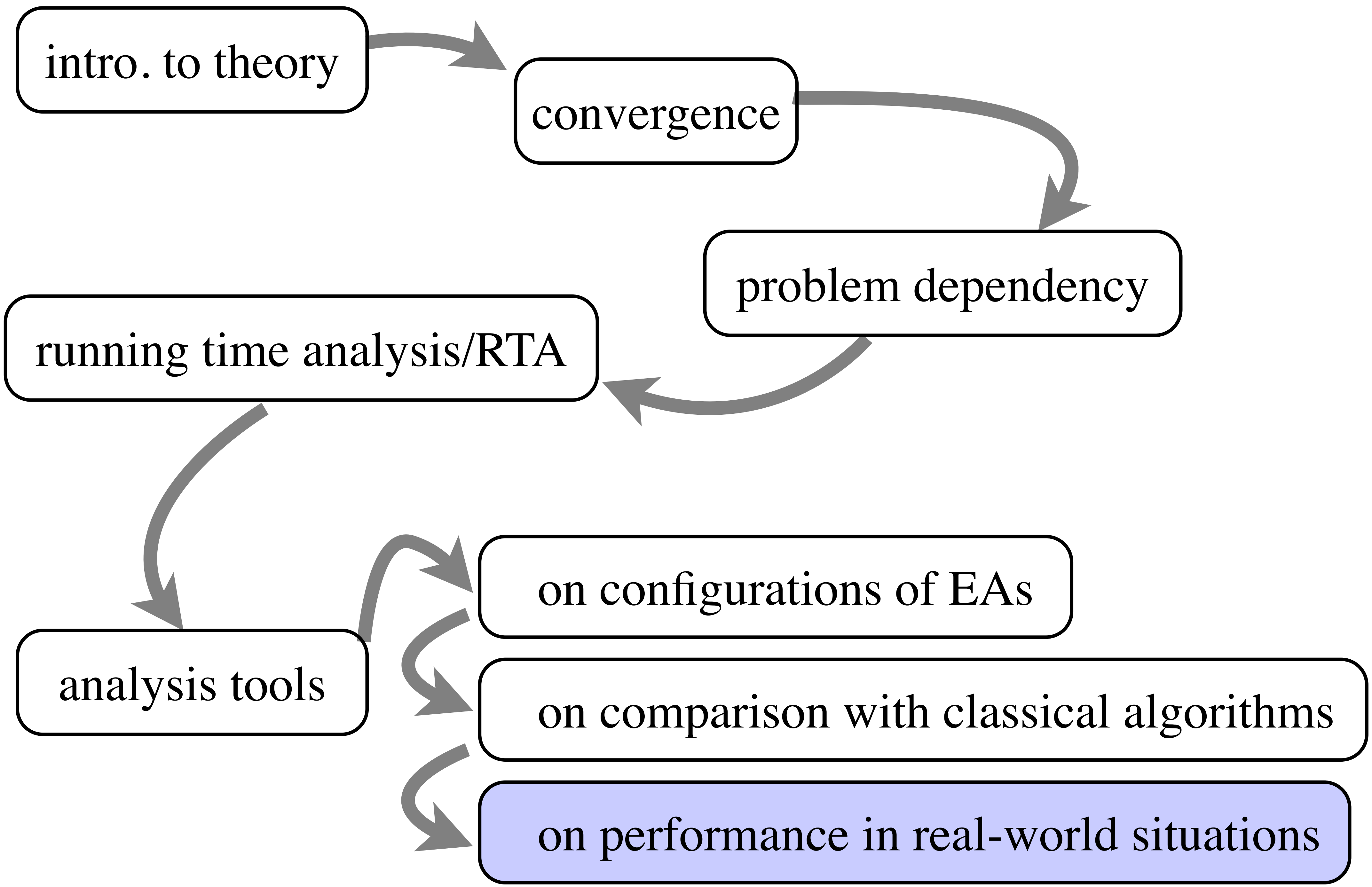
$$\mathcal{T}_0 \in \mathcal{S}_0 \longrightarrow \cdots \longrightarrow \mathcal{T}_i \in \mathcal{S}_i \rightarrow \mathcal{T}_{i-1} \in \mathcal{S}_{i-1} \longrightarrow \cdots \longrightarrow \mathcal{T}_n \in \mathcal{S}_n$$

EAs can be configured to solve a DP problem with ERT:

$$O(|\mathcal{S}_0| + n \cdot \log(\sum_{i=0}^{n} |\mathcal{T}_i|) \cdot \sum_{i=1}^{n} |\mathcal{T}_{i-1}| \cdot |\mathcal{F}_i|)$$

single source shortest path: $O(n^4 \ln n)$
all pairs shortest path: $O(n^5 \ln n)$

Road map


intro. to theory
convergence
problem dependency
running time analysis/RTA
analysis tools
on configurations of EAs
on comparison with classical algorithms
on performance in real-world situations

On real-world performance

EAs are expected to be applied in hard problems

- problems with unknown formulae

  properties about problem classes

- problems hard to solve (NP-hard)

  analysis in NP-hard problems

On properties about problem classes

[Fournier & Teytaud, 11]:

with the variable of problem class complexity
for evolutionary strategies
give lower bounds of the particular convergence rate

[Qian, et al., 12]:

in pseudo-boolean function class
for (1+1)-EA
identify the easiest and the hardest problem cases

...

In NP-hard problems

Approximation ratio

for minimization, in every problem instance let s be the solved solution and s^* be an optimal solution

approximation ratio is the largest value of $\frac{f(s)}{f(s^*)}$ over all problem instances

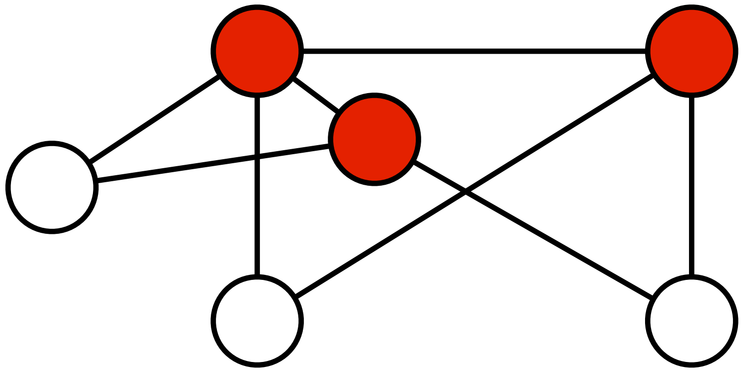
no smaller than 1, the smaller the better

usually consider the achieved ratio within polynomial ERT

In NP-hard problems

Minimum Vertex Cover (MVC) problem

to minimize the number of vertices covering all edges

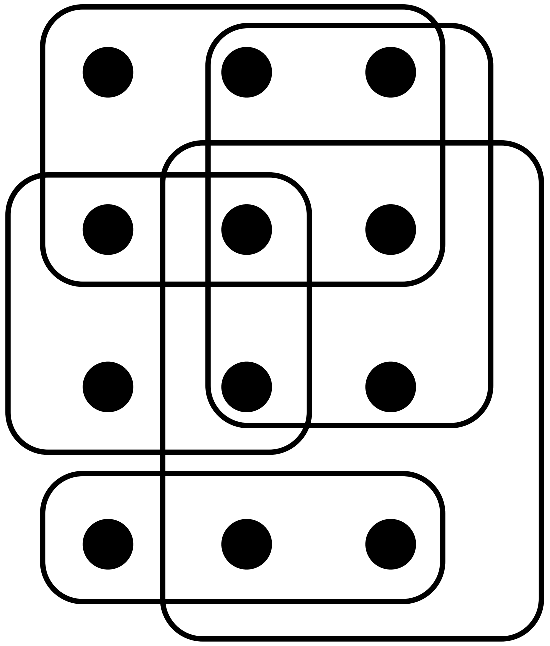


$2-$approximation by maximum matching

can not be approximated within a factor ≈ 1.36

Minimum Set Cover (MSC) problem

to minimize the number of sets covering all elements (uniweighted)

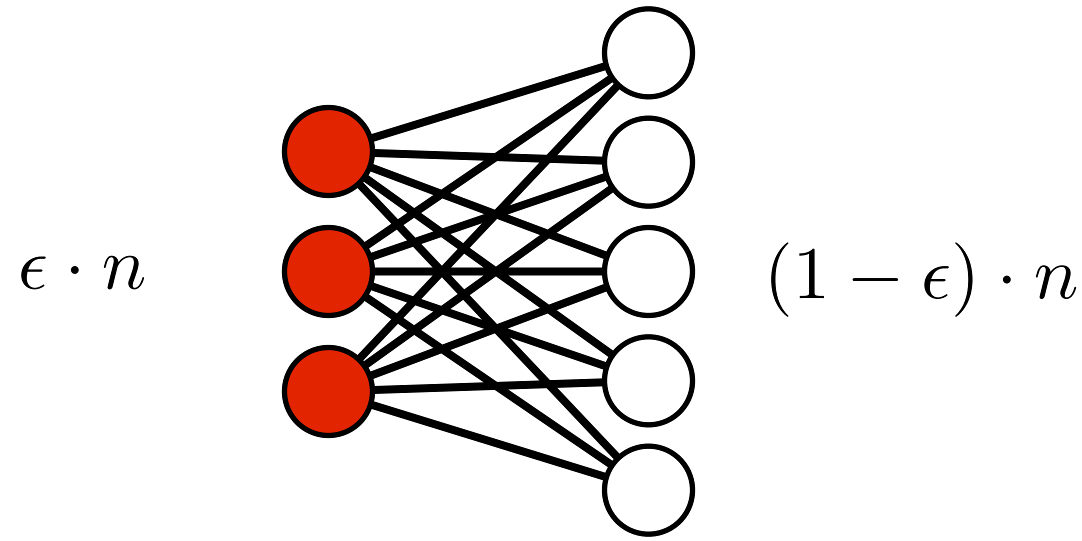


to minimize the total weight of a collection of sets covering all elements (general)

$\ln n-$approximation by the greedy algorithm, and is asymptotically tight

(1+1)-EA in MVC problem

The ERT of (1+1)-EA achieving an approximate ratio better than $\frac{(1-\epsilon)}{\epsilon}$ is exponential $\forall \epsilon > 0$ [Friedrich, et al., 10]



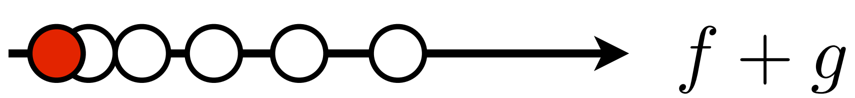
approximation ratio: $\frac{(1-\epsilon)}{\epsilon}$

Further investigations:
[Oliveto, et al., 09] studied (1+1)-EA in several instances of MVC problem
[Friedrich, et al., 09] studied hybrid (1+1)-EA with the greedy algorithm and the maximum matching algorithm

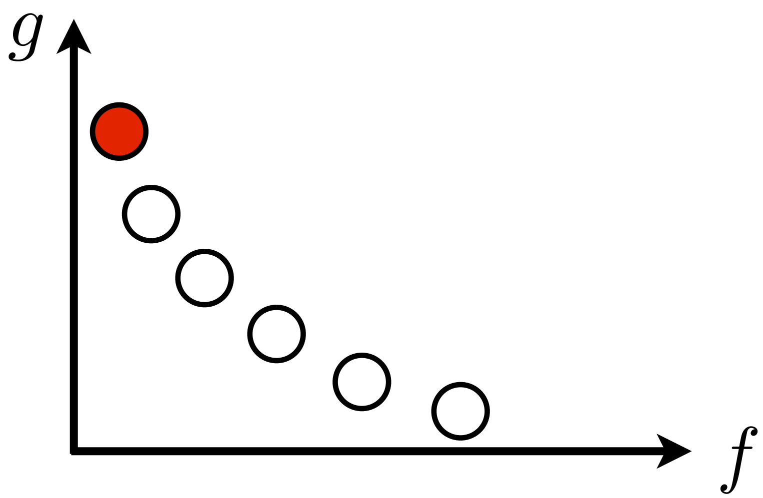
Multi-objective reformulation

1. Convert a single objective optimization problem to a multi-objective optimization problem by extracting/adding auxiliary functions
2. Solve the multi-objective optimization problem
3. Convert the obtained Pareto set back for the single objective problem

$$\arg\min_{\boldsymbol{x}} f(x) + g(x)$$



$$\arg\min_{\boldsymbol{x}} \Big(f(x), g(x) \Big)$$



For MVC problem

single objective:

$$\arg\min[\text{number of selected vertices}] + \lambda \cdot [\text{number of uncovered edges}]$$

multi-objective:

$$\arg\min([\text{number of selected vertices}], [\text{number of uncovered edges}])$$

Multi-objective reformulation

[Scharnow, et al., 04] first disclosed that multi-objective reformulation may be helpful in solving Shortest Path problem.
It is then confirmed by studies (e.g. [Neumann & Wegener, 07b] in shortest path and spanning tree problems)

[Friedrich, et al., 10]: by the multi-objective reformulation with SEMO,

1. solve the Minimum Vertex Cover bipartite instance in polynomial time
2. obtain $\ln n-$approximate solutions for the (general) Minimum Set Cover problem in polynomial time

[Laumanns, et al., 02]

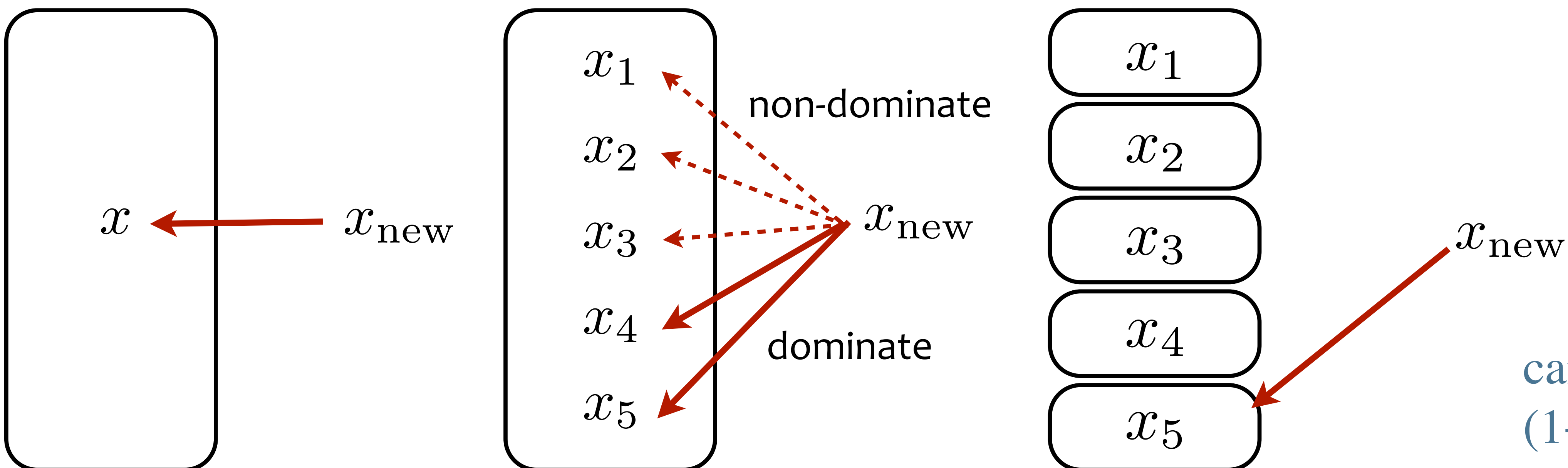
A Simple Multi-objective EA (SEMO)

```
Pop = {s} ← a randomly drawn solution
for t=1,2,... do
    s ← randomly select from Pop
    s' ← mutate(s)
    if ∄s'' ∈ Pop such that s'' dominates s' then
        remove solutions in Pop that are dominated by s'
        add s' into Pop
    end if
    terminate if meets a stopping criterion
end for
```

A unified framework

[Yu, et al., 12] proposed a unified framework for both single- and multi-objective EAs

isolation function: isolates the competition among solutions



(1+1)-EA | multi-objective EA | by isolations

can be configured as (1+1)-EA or a multi-objective EA

q isolations, c gap, r_i increase of objective value

EAs can finds $(\sum_{i=0}^{q-1} r_i)$-approximate solutions in $O(q^2 n^c)$ time

should not too many isolations (q is polynomial in n)

should not too large variation is needed (c is constant)

A unified framework

[Yu, et al., 12] proposed a unified framework for both single- and multi-objective EAs

isolation function: isolates the competition among solutions

EAs can finds $(\sum_{i=0}^{q-1} r_i)$-approximate solutions in $O(q^2 n^c)$ time

Applications:

- **simulate the greedy algorithm**
  finds H_n -approximate solutions in $O(mn^2)$ time in general MSC problem

- **exceed the greedy algorithm**
  finds $(H_k - \frac{k-1}{8k^9})$-approximate solutions in $O(m^{k+1}n^2)$ time for k-set cover problem

1/*k*-approximate solutions for b-matching, maximum profit scheduling and maximum asymmetric TSP problems (*k*-extensible systems) [Mestre, 06]

In NP-Hard problems

Minimum Vertex Cover fixed-parameter complexity [Kratsch & Neumann, 13]

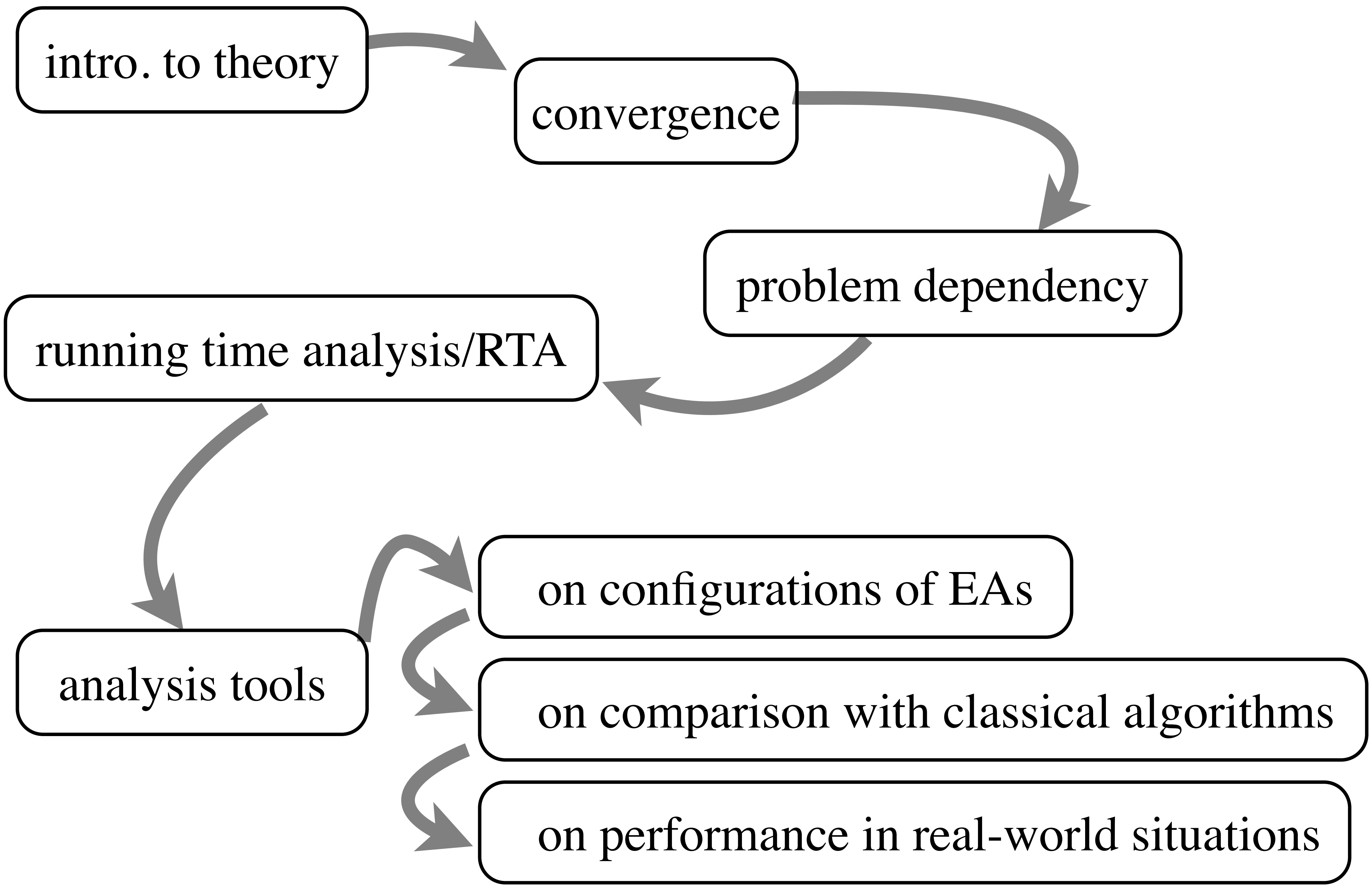
Spanning Forest [Neumann & Laumanns, 06]

Minimum Multicuts [Neumann & Reichel, 08]

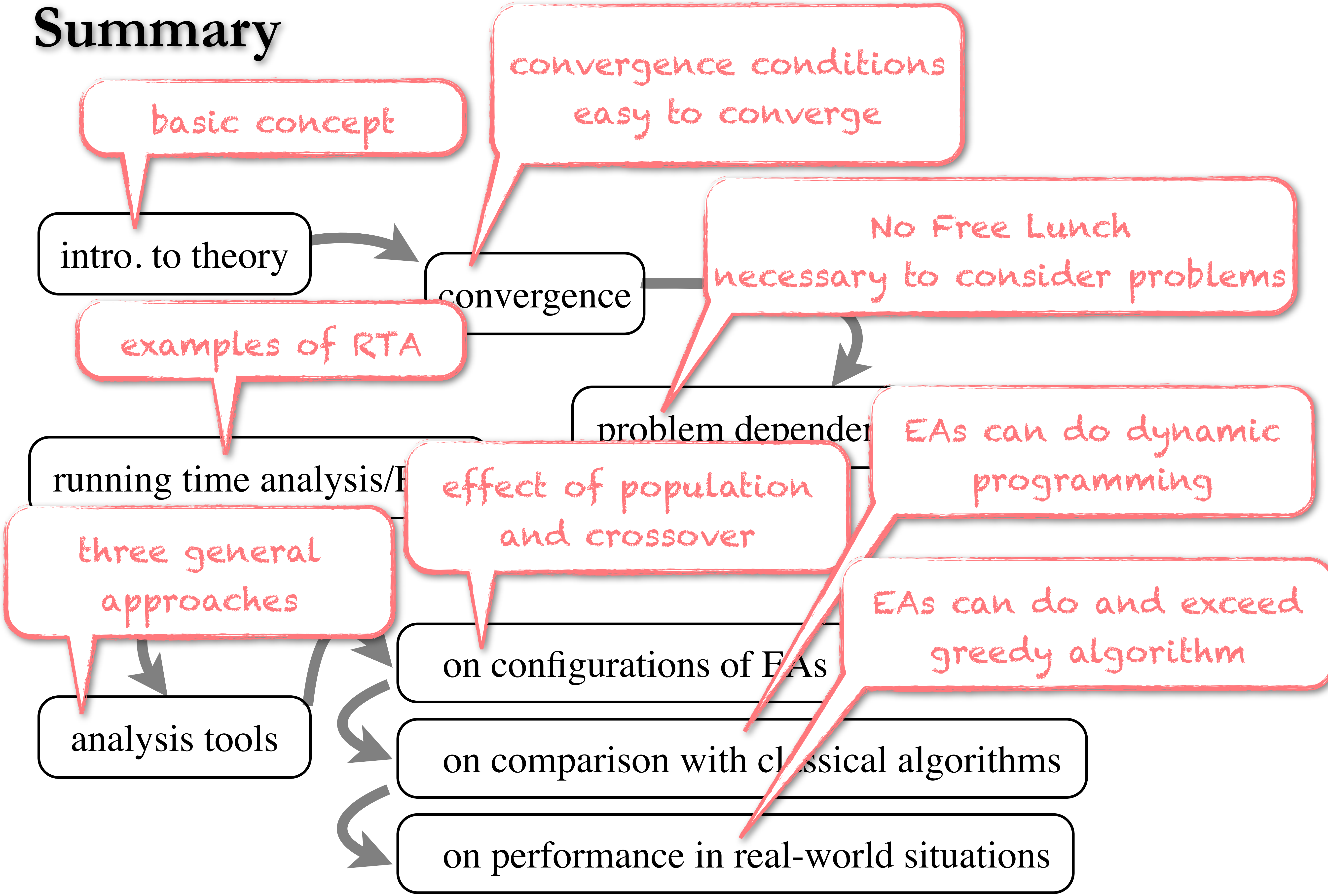
Traveling Salesman [Kötzing, et al., 12][Sutton & Neumann, 12]

...

Summary


intro. to theory
convergence
problem dependency
running time analysis/RTA
analysis tools
on configurations of EAs
on comparison with classical algorithms
on performance in real-world situations

Summary


basic concept
intro. to theory
convergence conditions
easy to converge
convergence
No Free Lunch
necessary to consider problems
problem dependency
examples of RTA
running time analysis/RTA
three general approaches
analysis tools
effect of population
and crossover
on configurations of EAs
EAs can do dynamic
programming
EAs can do and exceed
greedy algorithm
on comparison with classical algorithms
on performance in real-world situations

Summary

a lot of open problems

a fast growing research area

intro. to theory

convergence

problem dependency

running time analysis/RTA

analysis tools

on configurations of EAs

on comparison with classical algorithms

on performance in real-world situations

Available books on EA theory

F. Neumann, C. Witt.
Bioinspired Computation in Combinatorial Optimization – Algorithms and Their Computational Complexity.
Springer-Verlag, Berlin, Germany, 2010.

A. Auger and B. Doerr. *Theory of Randomized Search Heuristics - Foundations and Recent Developments*.
World Scientific, Singapore, 2011.

...

Major venues of theoretical work on EAs

Major journals:

- Artificial Intelligence (Elsevier)
- Algorithmica (Springer)
- Evolutionary Computation (MIT Press)
- Theoretical Computer Science (Elsevier)
- IEEE Trans. on Evolutionary Computation (IEEE)
- ...

Major conferences:

- PPSN (International Conference on Parallel Problem Solving From Nature, bi-annual, even year)
- GECCO (International Conference on Genetic and Evolutionary Computation, annual)
- FOGA (International Workshop on Foundations of Genetic Algorithms, bi-annual, odd year)
- CEC (IEEE Conference on Evolutionary Computation, annual)
- ...

Reference

[Böttcher, et al., 10] S. Böttcher, B. Doerr and F. Neumann. Optimal Fixed and Adaptive Mutation Rates for the LeadingOnes Problem. In: Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10), pages 1-10, Kraków, Poland, 2010.

[Chen, et al., 09] T. Chen, J. He, G. Sun, G. Chen and X. Yao. A new approach for analyzing average time complexity of population-based evolutionary algorithms on unimodal problems. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 39(5):1092-1106, 2009.

[Chen, et al., 12] T. Chen, K. Tang, G. Chen and X. Yao. A large population size can be unhelpful in evolutionary algorithms. Theoretical Computer Science, 436:54-70. 2012.

[Dietzfelbinger, et al., 03] M. Dietzfelbinger, B. Naudts, C. Van Hoyweghen, and I. Wegener. The analysis of a recombi- native hill-climber on H-IFF. IEEE Transactions on Evolutionary Computation, 7(5):417-423, 2003.

[Doerr & Goldberg, 13] B. Doerr and L. A. Goldberg. Adaptive drift analysis. Algorithmica, 65:224-250, 2013.

[Doerr & Happ, 08] B. Doerr, and E. Happ. Directed trees: A powerful representation for sorting and ordering problems. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC'08), Hong Kong, China, 2008, pp.3606-3613.

[Doerr & Johannsen, 10] B. Doerr and D. Johannsen. Edge-based representation beats vertex-based representation in shortest path problems. In: Proceedings of the 12th ACM Conference on Genetic and Evolutionary Computation (GECCO'10), Portland, OR, 2010, pp.759-766.

[Doerr, et al., 11] B. Doerr, A. V. Eremeev, F. Neumann, M. Theile, C. Thyssen. Evolutionary algorithms and dynamic programming. Theoretical Computer Science 412(43): 6020-6035, 2011.

[Doerr, et al., 11b] B. Doerr, E. Happ, and C. Klein. Tight analysis of the (1+1)-EA for the single source shortest path problem. Evolutionary Computation 19(4): 673-691, 2011.

[Doerr, et al., 12] B. Doerr, D. Johannsen, and C. Winzen. Multiplicative drift analysis. Algorithmica, 64:673-697, 2012.

[Doerr, et al., 13] B. Doerr, D. Johannsen, T. Kötzing, F. Neumann, and M. Theile. More effective crossover operators for the all-pairs shortest path problem. Theoretical Computer Science, 471: 12-26, 2013.

[Droste, et al., 98] S. Droste, T. Jansen, and I. Wegener. A rigorous complexity analysis of the (1 + 1) evolutionary algorithm for separable functions with boolean inputs. Evolutionary Computation, 6(2):185-196, 1998.

[Fischer & Wegener, 05] S. Fischer and I. Wegener. The one-dimensional Ising model: mutation versus recombination. Theoretical Computer Science, 344(2-3):208-225, 2005.

[Fournier & Teytaud, 11] H. Fournier, O. Teytaud. Lower bounds for comparison based evolution strategies using VC-dimension and sign patterns. Algorithmica, 59:387-408, 2011.

Reference

[Friedrich, et al., 09] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt, Analyses of simple hybrid algorithms for the vertex cover problem, Evolutionary Computation 17 (1): 3-19, 2009.

[Friedrich, et al., 10] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt. Approximating covering problems by randomized search heuristics using multi-objective models. Evolutionary Computation, 18(4):617-633, 2010.

[Giel & Wegener, 03] O. Giel, I. Wegener. Evolutionary algorithms and the maximum matching problem. In: Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS'03), 415-426, 2003.

[Hajek, 82] B. Hajek. Hitting-time and occupation-time bound implied by drift analysis with applications. Advances in Applied Probability, 14(3):502-525, 1982.

[Happ, et al., 08] E. Happ, D. Johannsen, C. Klein, and F. Neumann. Rigorous analyses of fitness-proportional selection for optimizing linear functions. In: Proceedings of the 10th ACM Conference on Genetic and Evolutionary Computation (GECCO'08), Atlanta, GA, 2008, pp.953-960.

[He & Yao, 01] J. He and X. Yao. Drift analysis and average time complexity of evolutionary algorithms. Artificial Intelligence, 127(1): 57-85, 2001.

[He & Yao, 04] J. He and X. Yao. A study of drift analysis for estimating computation time of evolutionary algorithms. Natural Computing, 3(1): 21-35, 2004.

[He & Yu, 01] J. He and X. Yu. Conditions for the convergence of evolutionary algorithms. Journal of Systems Architecture, 47(7): 601-612, 2001.

[Holland, 75] J. Holland, Adaptation in Natural and Artificial Systems, The MIT Press, 1975.

[Jansen & Wegener, 01] T. Jansen and I. Wegener. On the utility of populations in evolutionary algorithms. In: Proceedings of the 3rd ACM Conference on Genetic and Evolutionary Computation (GECCO'01), San Francisco, CA, 2001, pp.1034-1041.

[Jansen & Wegener, 02] T. Jansen and I. Wegener. The analysis of evolutionary algorithms -- A proof that crossover really can help. Algorithmica, 34(1): 47-66, 2002.

[Jansen & Wegener, 05] T. Jansen and I. Wegener. Real royal road functions -- where crossover provably is essential. Discrete Applied Mathematics, 149(1-3): 111-125, 2005.

[Jansen, et al., 05] T. Jansen, K. Jong and I. Wegener. On the choice of the offspring population size in evolutionary algorithms. Evolutionary Computation, 13(4): 413-440, 2005.

[Kötzing, et al., 11] T. Kötzing, D. Sudholt, and M. Theile. How crossover helps in pseudo-boolean optimization. In: Proceedings of the 13th ACM Conference on Genetic and Evolutionary Computation (GECCO'11), Dublin, Ireland, 2011, pp.989-996.

Reference

[Kötzing, et al., 12] T. Kötzing, F. Neumann, H. Röglin, C. Witt. Theoretical analysis of two ACO approaches for the traveling salesman problem. Swarm Intelligence 6(1): 1-21, 2012.

[Kratsch & Neumann, 13] S. Kratsch, and F. Neumann: Fixed-parameter evolutionary algorithms and the vertex cover problem. Algorithmica 65(4): 754-771, 2013.

[Laumanns, et al., 02] M. Laumanns, L. Thiele, E. Zitzler, E. Welzl, and K. Deb, Running time analysis of multi-objective evolutionary algorithms on a simple discrete optimization problem, in: Proceedings of the 7th International Conference on Parallel Problem Solving from Nature (PPSN'02), London, UK, 2002, pp. 44-53.

[Lehre & Witt, 13] P. K. Lehre and C. Witt. General drift analysis with tail bounds. ArXiv:1307.2559, 2013.

[Mestre, 06] J. Mestre. Greedy in Approximation Algorithms. In: Proceedings of the 14th Annual European Symposium on Algorithms, Zurich, Switzerland, 2006, pp.528-539.

[Neumann & Laumanns, 06] F. Neumann, M. Laumanns, Speeding up approximation algorithms for NP-hard spanning forest problems by multi-objective optimization, in: Proceedings of the 7th Latin American Symposium on Theoretical Informatics, Valdivia, Chile, 2006, pp. 745-756.

[Neumann & Reichel, 08] F. Neumann, J. Reichel, Approximating minimum multicuts by evolutionary multi-objective algorithms, in: Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN'08), Dortmund, Germany, 2008, pp. 72-81.

[Neumann & Theile, 10] 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), pages 667-676, Krakow, Poland, 2010.

[Neumann & Wegener, 07] F. Neumann and I. Wegener. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. Theoretical Computer Science 378:32-40, 2007.

[Neumann & Wegener, 07b] F. Neumann, I. Wegener, Can single-objective optimization profit from multiobjective optimization? in: J. Knowles, D. Corne, K. Deb (Eds.), Multiobjective Problem Solving from Nature - From Concepts to Applications, Springer, Berlin, Germany, 2007, pp. 115-130.

[Neumann, et al., 11] F. Neumann, P. S. Oliveto, G. Rudolph, and D. Sudholt. On the effectiveness of crossover for migration in parallel evolutionary algorithms. In: Proceedings of the 13th ACM Conference on Genetic and Evolutionary Computation (GECCO'11), pages 1587-1594, Dublin, Ireland, 2011.

[Oliveto & Witt, 08] P. Oliveto and C. Witt. Simplified drift analysis for proving lower bounds in evolutionary computation. In: Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN'08), pages 82-91, Dortmund, Germany, 2008.

Reference

[Oliveto, et al., 09] P. Oliveto, J. He and X. Yao. Analysis of the (1 + 1)-EA for finding approximate solutions to vertex cover problems, IEEE Transactions on Evolutionary Computation 13(5):1006-1029, 2009.

[Qian, et al., 11] C. Qian, Y. Yu, and Z.-H. Zhou. An analysis on recombination in multi-objective evolutionary optimization. In: Proceedings of the 13th ACM Conference on Genetic and Evolutionary Computation (GECCO'11), pages 2051-2058, Dublin, Ireland, 2011.

[Qian, et al., 12] C. Qian, Y. Yu, Z.-H. Zhou. On algorithm-dependent boundary case identification for problem classes. In: Proceedings of the 12th International Conference on Parallel Problem Solving from Nature (PPSN'12), Taormina, Italy, 2012, pp.62-71.

[Rabani, et al., 98] Y. Rabani, Y. Rabinovich, and A. Sinclair. A computational view of population genetics. Random Structures and Algorithms, 12(4):313-334, 1998.

[Richter, 08] J. Richter, A. Wright, and J. Paxton. Ignoble trails -- where crossover is provably harmful. In: Proceedings of the 10th International Conference on Parallel Problem Solving from Nature (PPSN'08), pages 92-101, Dortmund, Germany, 2008.

[Sasaki & Hajek, 88] G. Sasaki and B. Hajek. The time complexity of maximum matching by simulated annealing. Journal of the ACM, 35(2):387-403, 1988.

[Scharnow, et al., 04] J. Scharnow, K. Tinnefeld, and I. Wegener. The analysis of evolutionary algorithms on sorting and shortest paths problems. Journal of Mathematical Modelling and Algorithms, 3(4):349-366, 2004.

[Storch, 08] T. Storch. On the choice of the parent population size. Evolutionary Computation, 16(4):557-578, 2008.

[Sudholt, 05] D. Sudholt. Crossover is provably essential for the ising model on trees. In: Proceedings of the 7th ACM Conference on Genetic and Evolutionary Computation (GECCO'05), Washington DC, 2005, pp.1161-1167.

[Sudholt, 10] D. Sudholt. General lower bounds for the running time of evolutionary algorithms. In: Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10), Krakow, Poland, 2010, pp.124-133.

[Sudholt, 13] D. Sudholt. A new method for lower bounds on the running time of evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 17(3): 418-435, 2013.

[Sutton & Neumann, 12] A. M. Sutton, Neumann. A Parameterized Runtime Analysis of Evolutionary Algorithms for the Euclidean Traveling Salesperson Problem. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12), Toronto, Canada, 2012.

[Watson, 01] R. A. Watson. Analysis of recombinative algorithms on a non-separable building block problem. In: W. N. Martin and W. M. Spears, editors, Foundations of Genetic Algorithms 6, . Morgan Kaufmann, San Francisco, 2001, pp.69-89.

[Wegener, 02] I. Wegener. Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In: M. M. Ruhul A. Sarker and X. Yao, editors, Evolutionary Optimization. Kluwer, 2002.

Reference

[Witt, 06] C. Witt. Runtime analysis of the (μ+1) EA on simple pseudo-Boolean functions. Evolutionary Computation, 14(1): 65-86, 2006.

[Witt, 08] C. Witt. Population size versus runtime of a simple evolutionary algorithm. Theoretical Computer Science, 403(1): 104-120, 2008.

[Witt, 13] C. Witt. The fitness level method with tail bounds. ArXiv:1307.4274, 2013.

[Wolpert & Macready, 97] D. Wolpert, and W. G. Macready: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1:67-82, 1997.

[Yu & Zhou, 08] Y. Yu and Z.-H. Zhou. A new approach to estimating the expected first hitting time of evolutionary algorithms. Artificial Intelligence, 172(15): 1809-1832, 2008.

[Yu, et al., 10] Y. Yu, C. Qian, and Z.-H. Zhou. Towards analyzing recombination operators in evolutionary search. In: Proceedings of the 11th International Conference on Parallel Problem Solving from Nature (PPSN'10) Part I, Krakow, Poland, 2010, pp.144-153.

[Yu, et al., 12] 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.