Heuristic Search and Evolutionary Algorithms

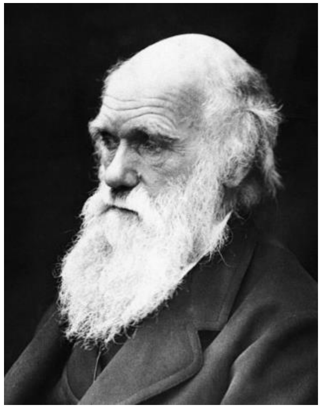
Evolutionary Algorithms
– Origins, Components and Applications

Compiled By :
Dr. Diya Vadhwani

- Hill-climbing search
- Simulated annealing
- Local beam search
- Local search for continuous spaces
- Evolutionary algorithms

Local search

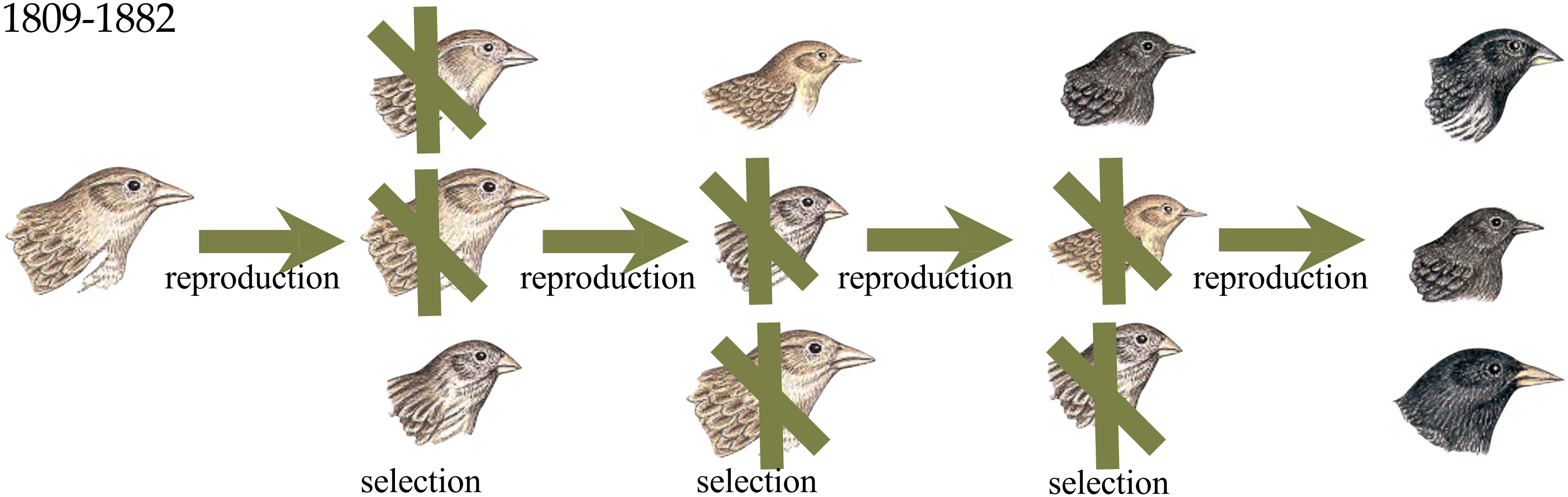
Biological evolution



Charles Darwin
1809-1882

C. Darwin, after collecting abundant evidence, developed a theory about how species evolve

reproduction with variation + nature selection



Optimization

With the development of computing technology

Curious researchers started to implement Darwin’s theory of evolution in computer, and found connections to ***optimization***

Optimization:

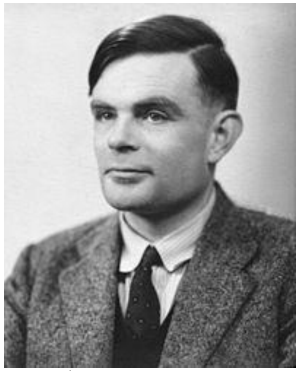
how to put as much stuff as possible into a fixed size container?

Formally: $\arg\max_{x \in \mathcal{X}} f(x)$ every x is an arrangement of objects
f counts the number of objects in the container

Evolutionary optimization

In 1950, Turing described how evolution might be used for his optimization:

building intelligent machine



Alan Turing
1912-1954

”We have thus divided our problem into two parts. The child programme and the education process. These two remain very closely connected. We cannot expect to find a good child machine at the first attempt. One must experiment with teaching one such machine and see how well it learns. One can then try another and see if it is better or worse. There is an obvious connection between this process and evolution, by the identifications

Structure of the child machine = Hereditary material

Changes of the child machine = Mutations

Judgment of the experimenter = Natural selection”

(The last equation swapped)

[A. M. Turing. Computing machinery and intelligence. Mind 49: 433-460, 1950.]

The origins

Genetic Algorithms (GA)

for optimization in discrete domains

[J. H. Holland. *Outline for a logical theory of adaptive systems*. JACM, 1962]

J. H. Holland
1929-2015

University of Michigan

Evolutionary Strategies (ES)

for optimization in continuous domains

[I. Rechenberg. *Cybernetic solution path of an experimental problem*. 1965]

I. Rechenberg
1934-2021

Technical University of Berlin

Evolutionary Programming (EP)

for optimizing finite state machines (agents)

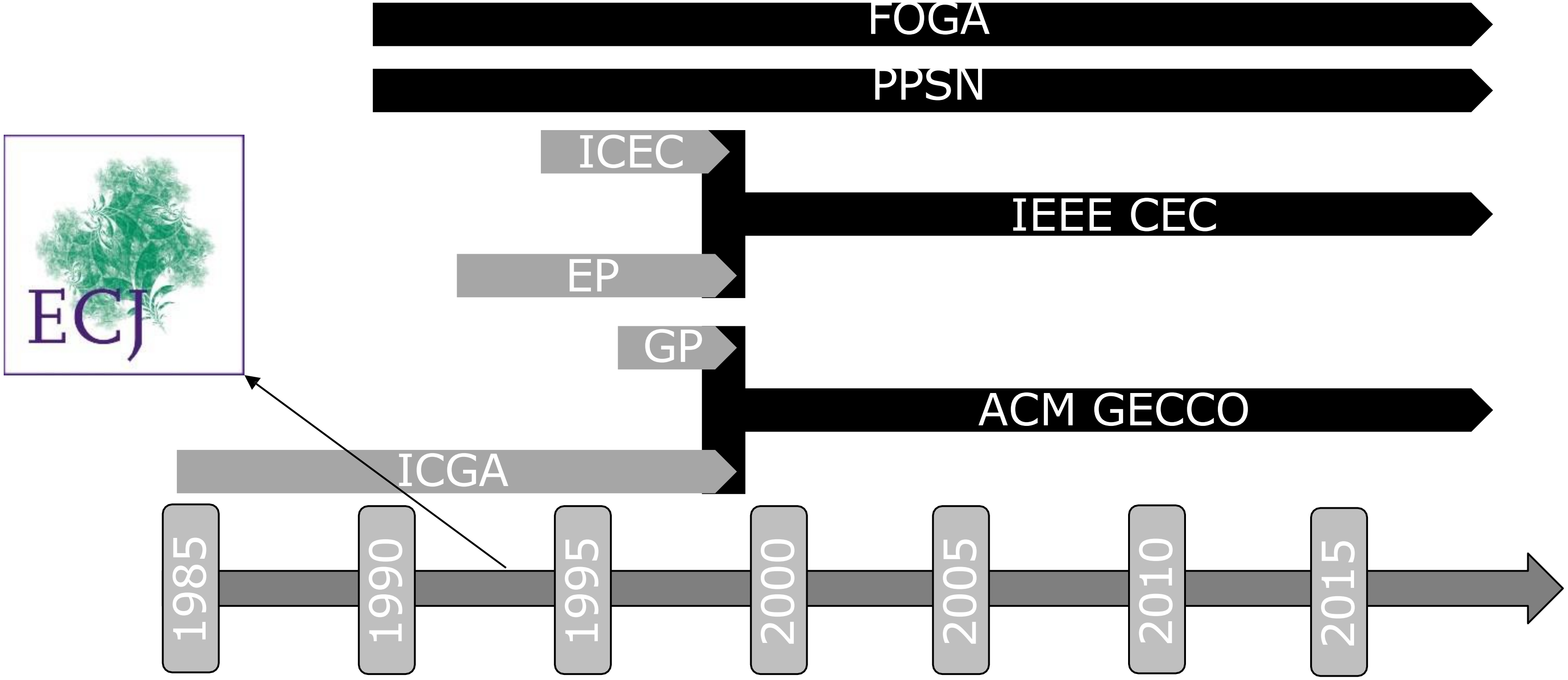
[L. J. Fogel, A. J. Owens, M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. 1966]

L. J. Fogel
1928-2007

University of California, Los Angeles

The origins

The research of GA, ES and EP was done independently from 1960s to 1980s, and unified to one field "Evolutionary Computation" in 1990s



Main conferences and journals

Four main conferences

- IEEE Congress on Evolutionary Computation (CEC)
- ACM Conference on Genetic and Evolutionary Computation (GECCO)
- International Conference on Parallel Problem Solving from Nature (PPSN)
- ACM Conference on Foundations of Genetic Algorithms (FOGA)

Three main journals

- Evolutionary Computation Journal (ECJ, MIT Press, 1993)
- IEEE Trans. on Evolutionary Computation (TEvC)
- ACM Trans. on Evolutionary Learning and Optimization (TELO)

Evolutionary algorithms

Genetic Algorithms (GA)
for optimization in discrete domains

[J. H. Holland. *Outline for a logical theory of adaptive systems*. JACM, 1962]

Evolutionary Strategies (ES)
for optimization in continuous domains

[I. Rechenberg. *Cybernetic solution path of an experimental problem*. 1965]

Evolutionary Programming (EP)
for optimizing finite state machines

[L. J. Fogel, A. J. Owens, M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. 1966]

Other variants:
- Genetic Programming
- Differential Evolution
- ...

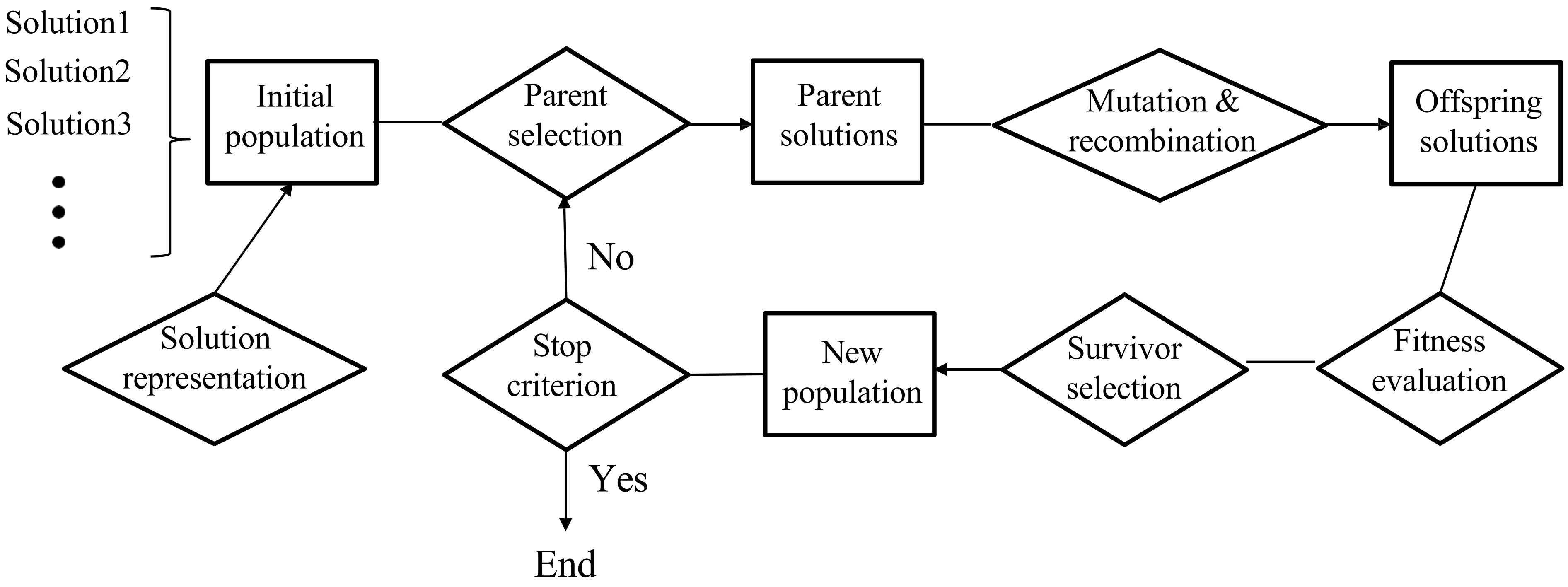
Other heuristics inspired from nature:
- Ant Colony Optimization
- Particle Swarm Optimization
- ...

Evolutionary algorithms (EAs)

Evolutionary algorithms

EAs share a common routine

for $\arg\max_{x} f(x)$



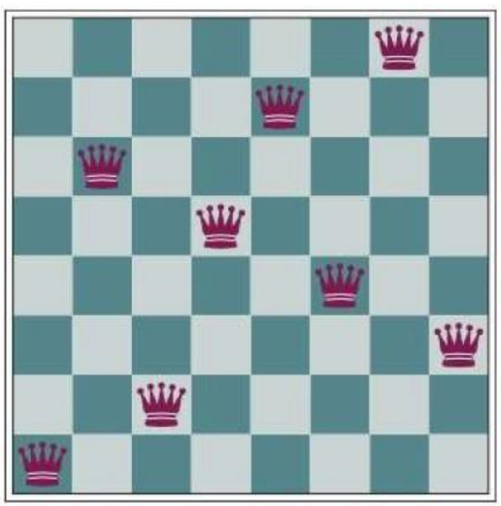
Components - representation

Representation: provides code for candidate solutions that can be manipulated by a computer

encoding & decoding

phenotype: object in original problem context

genotype: code to denote that object



Integer vector

1	6	2	5	7	4	8	3

Binary vector

000101001100110011111010

different

Permutation

1	6	2	5	7	4	8	3

Components – fitness

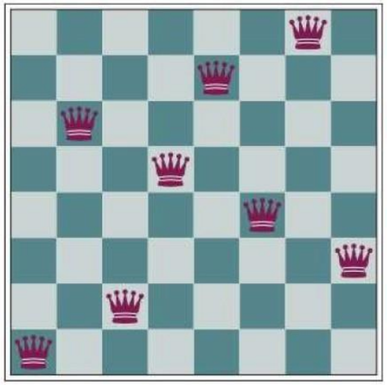
Fitness: represents the task to solve, or the requirements (can be seen as "the environment") to adapt to

Fitness evaluation assigns a single real-value to each phenotype which forms the basis for selection

Example:

$$\arg\max_x x^2$$

Fitness: x^2



Fitness: number of nonattacking pairs of queens

Components - population

Population: holds the candidate solutions of the problem, which is a multiset of genotypes

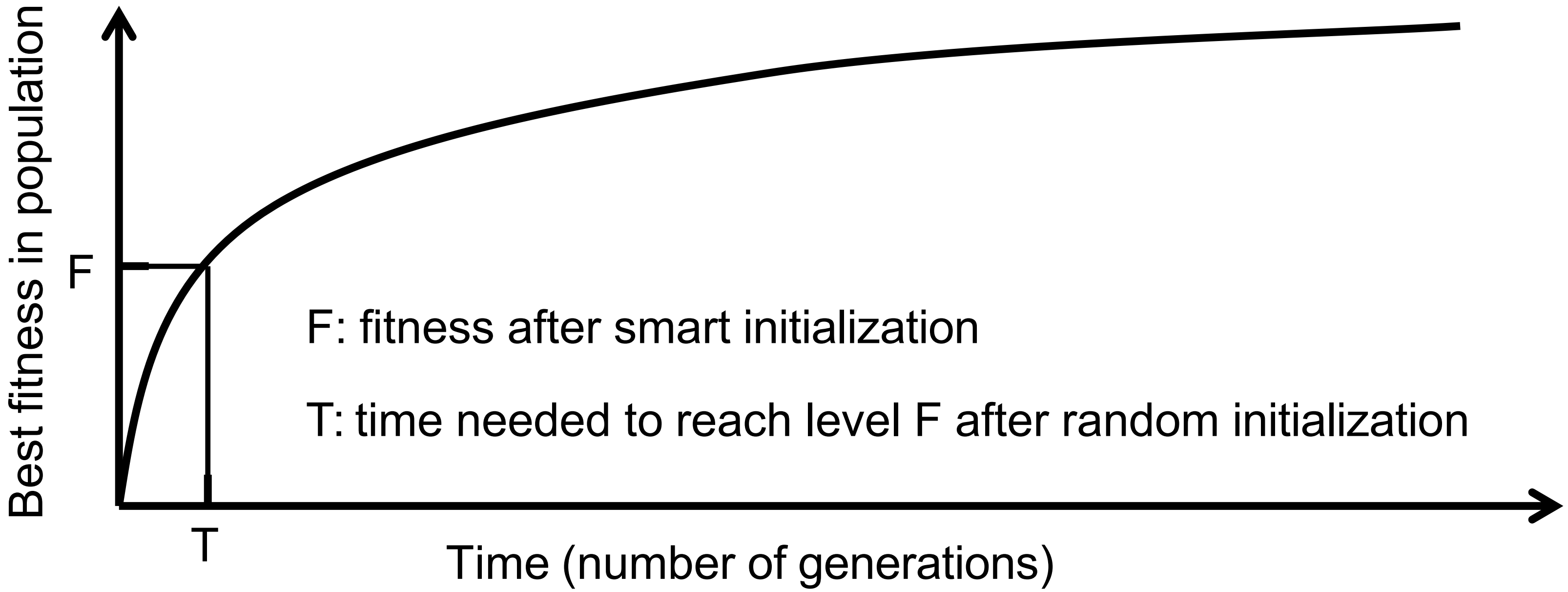
Size of population: the number of contained genotypes

Diversity of population: the number of different fitnesses / phenotypes / genotypes present

Components - initialization

Initialization: generates the genotypes in the initial population

- generates the genotypes randomly
- includes existing solutions, or uses problem-specific heuristics, to seed the population



Components – parent selection

Parent selection: selects genotypes to undergo variation

Usually probabilistic

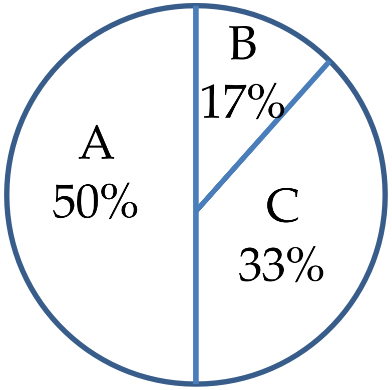
- high quality genotypes more likely to be selected than low quality
- even worst in the current population usually has non-zero probability of being selected

Example: fitness proportional selection

fitness(A) = 3

fitness(B) = 1

fitness(C) = 2



Components – variation

Variation: generates new (offspring) genotypes

- Mutation: causes small, random variance of one parent

Parent	1	0	1	1	1	0	0	0
Offspring	1	0	1	1	0	0	0	0

- Recombination/crossover: merges information from parents into offspring

Parent1	1	0	1	1	1	0	0	0
Parent2	0	0	1	0	1	0	1	0
Offspring1	1	0	1	1	1	0	1	0
Offspring2	0	0	1	0	1	0	0	0

Components – survivor selection

Survivor selection: selects genotypes from parents and offspring to form the next population

Often deterministic

- Fitness based : e.g., rank parents and offspring, and select the top segment
- Age based: make as many offspring as parents and delete all parents

Example:

Parents	Offspring
fitness(A) = 3	fitness(D) = 4
fitness(B) = 1	fitness(E) = 1.5
fitness(C) = 2	fitness(F) = 1

Fitness based: A, C, D

Age based: D, E, F

Components – stop criterion

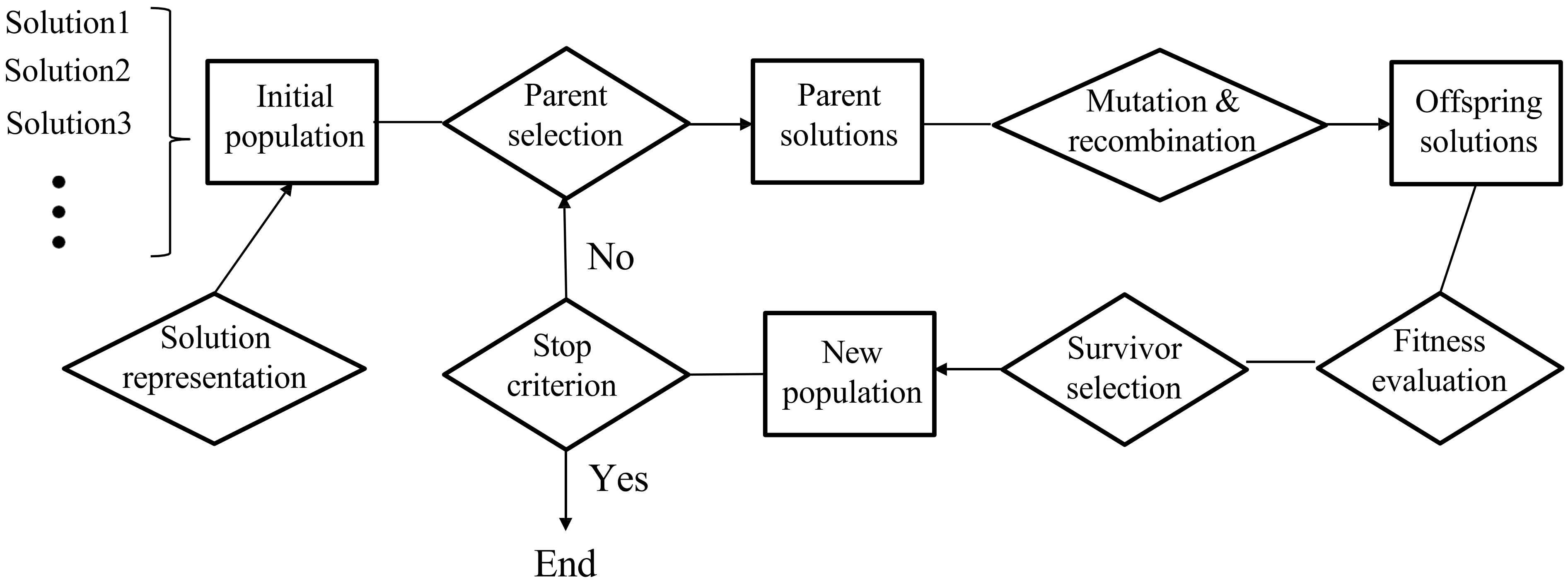
Anytime behavior of EAs

Stop criteria:

- Reaching some (known/hoped for) fitness
- Reaching some maximum allowed number of generations
- Reaching some specified number of generations without fitness improvement
- Reaching some minimum level of population diversity

Evolutionary algorithms

EAs share a common routine   for $\arg\max_x f(x)$



Need to design each component of EAs

Evolutionary algorithms

Genetic Algorithms (GA)
for optimization in discrete domains

[J. H. Holland. *Outline for a logical theory of adaptive systems*. JACM, 1962]

Binary representation

Evolutionary Strategies (ES)
for optimization in continuous domains

[I. Rechenberg. *Cybernetic solution path of an experimental problem*. 1965]

Evolutionary Programming (EP)
for optimizing finite state machines

[L. J. Fogel, A. J. Owens, M. J. Walsh. *Artificial Intelligence through Simulated Evolution*. 1966]

Real-valued representation

Genetic Programming (GP)
for optimizing computer programs

[J. R. Koza. *Genetic Programming*.1992]

Tree representation

Example illustration - max x^2

The problem: $\arg max_{x \in \{0,1,\ldots,31\}} \; x^2$

Fitness function f: x^2

Solution representation: binary vector of length 5

For example, $x = 15$ can be represented by 01111

Genotype no.	Initial population	x value	Fitness $f(x) = x^2$	Selection prob. p_i	Expected count	Actual count
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1

1170

Population size = 4, randomly generated

Parent selection: $p_i = f(i) / \sum_{j \in P} f(j)$

Parent solutions

Expected Value =N.pi

Example illustration - max x^2

Genotype no.	Parent solutions	Crossover point	Offspring after xover	Flipped bits	Offspring after mutation
1	0 1 1 0|1	4	0 1 1 0 0	1	1 1 1 0 0
2	1 1 0 0|0	4	1 1 0 0 1	none	1 1 0 0 1
2	1 1|0 0 0	2	1 1 0 1 1	none	1 1 0 1 1
4	1 0|0 1 1	2	1 0 0 0 0	3	1 0 1 0 0

One-point crossover:

Select one point randomly, and exchange the parts after the point

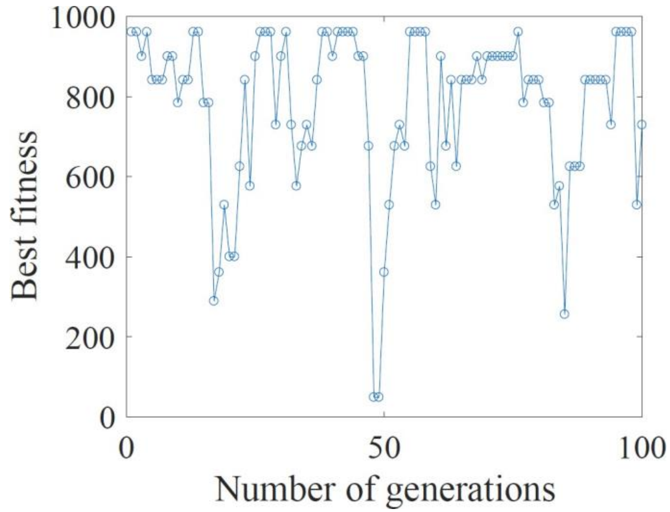
Bit-wise mutation:

Flip each bit of a solution with prob. $1/n$ where $n = 5$

Example illustration - max x^2

Initial population	x value	Fitness $f(x) = x^2$	Offspring after mutation	x value	Fitness $f(x) = x^2$	Next population
0 1 1 0 1	13	169	1 1 1 0 0	28	784	1 1 1 0 0
1 1 0 0 0	24	576	1 1 0 0 1	25	625	1 1 0 0 1
0 1 0 0 0	8	64	1 1 0 1 1	27	729	1 1 0 1 1
1 0 0 1 1	19	361	1 0 1 0 0	18	324	1 0 1 0 0

Curve change of the best fitness



Fitness evaluation

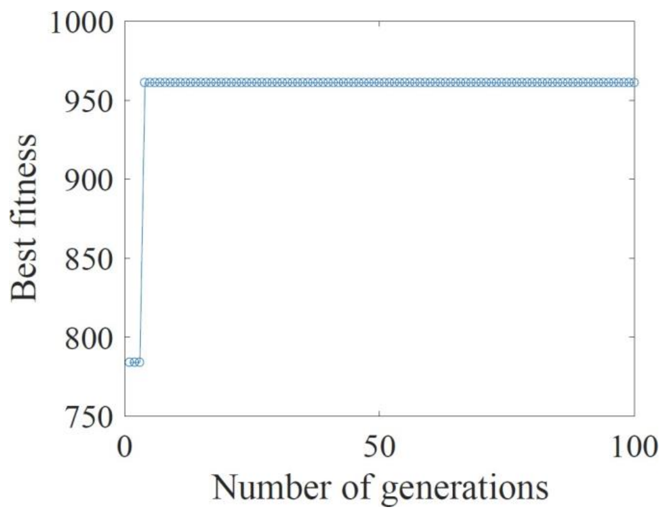
Age based survival selection:

Use the offspring directly to form the next population

Example illustration - max x^2

Initial population	x value	Fitness $f(x) = x^2$	Offspring after mutation	x value	Fitness $f(x) = x^2$	Next population
0 1 1 0 1	13	169	1 1 1 0 0	28	784	1 1 1 0 0
1 1 0 0 0	24	576	1 1 0 0 1	25	625	1 1 0 0 1
0 1 0 0 0	8	64	1 1 0 1 1	27	729	1 1 0 1 1
1 0 0 1 1	19	361	1 0 1 0 0	18	324	1 1 0 0 0

Curve change of the best fitness



Fitness based survival selection:

Select the best four genotypes from the current population and offspring

Problem Solving using GA

maximize $f(x) = x^2$ for $x \in \{0, \ldots, 31\}$ with **5-bit** chromosomes, pop size $N = 4$.

I'll follow one clean run: **roulette-wheel (SUS) selection**, **1-point crossover at bit 3**, **bit-wise mutation** (small).

Generation 0 (given)

geno	bits	x	f(x)=x²
1	01101	13	169
2	11000	24	576
3	01000	8	64
4	10011	19	361

$\sum f = 1170$ → selection probs $p_i = f_i / \sum f = [0.14, 0.49, 0.06, 0.31]$.

Selection (SUS, step = 1/N = 0.25, start r = 0.05): pointers at 0.05, 0.30, 0.55, 0.80 hit genotypes **[1,2,2,4]** → parent pool

$$[01101,\ 11000,\ 11000,\ 10011].$$

Crossover (1-point after bit 3; pair parents in order):

- Pair A: 011|01 × 110|00 → **01100 (12)**, **11001 (25)**
- Pair B: 110|00 × 100|11 → **11011 (27)**, **10000 (16)**

Mutation: assume none triggers (small p_m).

Generation 1: $[01100 = 12,\ 11001 = 25,\ 11011 = 27,\ 10000 = 16]$

Fitness $[144, 625, 729, 256]$; $\sum = 1754$; $p \approx [0.08, 0.36, 0.42, 0.15]$.

Generation 1 → 2

Selection (same SUS): pointers 0.05, 0.30, 0.55, 0.80 → **[1,2,3,3]**

parents $[01100,\ 11001,\ 11011,\ 11011]$.

Crossover (cut after bit 3):

- Pair A: 011|00 × 110|01 → **01101 (13)**, **11000 (24)**
- Pair B: 110|11 × 110|11 → **11011 (27)**, **11011 (27)**

Generation 2: $[13, 24, 27, 27]$ with fitness $[169, 576, 729, 729]$; $\sum = 2203$.

Generation 2 → 3 (a mutation creates the optimum)

Selection (SUS): pointers 0.05, 0.30, 0.55, 0.80 → **[1,2,3,4]**

parents $[13, 24, 27, 27]$.

Crossover (cut after bit 3):

- Pair A: 011|01 × 110|00 → **01100 (12)**, **11001 (25)**
- Pair B: 110|11 × 110|11 → **11011 (27)**, **11011 (27)**

Mutation (bit-wise, small p_m): suppose one 11011 flips the 3rd bit (0→1) → **11111 (31)**.

Generation 3: $[12, 25, 31, 27]$ with fitness $[144, 625, 961, 729]$; $\sum = 2459$.

Now the **best chromosome is 11111** → $x = 31$. Subsequent selection rapidly fills the population with 31 (the true maximizer), since $31^2 = 961$ dominates.

Example illustration - knapsack

Knapsack problem: given n items, each with a weight w_i and a value v_i, to select a subset of items maximizing the sum of values while keeping the summed weights within some capacity W_{max}

$$\arg max_{\boldsymbol{x}\in\{0,1\}^n} \sum_{i=1}^{n} v_i x_i \quad s.t. \sum_{i=1}^{n} w_i x_i \le W_{max}$$

$x_i = 1$: the i-th item is included

Solution representation

Genotype: binary vector of length n

1	1	0	1	1	0	1	1

Phenotype: binary vector of length n

1	1	0	1	1	0	1	1

Decoding: scan from left to right, and keep the value 1 if the summed weight does not exceed W_{max}

Example illustration - knapsack

Knapsack: $\arg max_{x \in \{0,1\}^n} \sum_{i=1}^{n} v_i x_i \quad s.t. \sum_{i=1}^{n} w_i x_i \le W_{max}$

Solution representation

$x_i = 1$: the i-th item is included

Genotype: binary vector of length n

1	1	0	1	1	0	1	1

Phenotype: binary vector of length n

1	1	0	1	1	0	1	1

Decoding: scan from left to right, and keep the value 1 if the summed weight does not exceed W_{max}

Example: v_i: 4,2,6,10,4,3,7,2; w_i: 2,3,3,8,6,5,7,1; $W_{max} = 25$

Genotype: 11011011 $\Rightarrow$ Phenotype: 11011001

Fitness function f: the sum of values, i.e., $\sum_{i=1}^{n} v_i x_i$

Example illustration - knapsack

Population size	500
Initialization	Random
Parent selection	Tournament selection with size 2
Recombination	One-point crossover
Recombination prob.	70%
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	500
Survival selection	Age based
Termination condition	No improvement in last 25 generations

Select two solutions from the population uniformly at random, and choose the better one as a parent solution

Example illustration - knapsack

Population size	500
Initialization	Random
Parent selection	Tournament selection with size 2
Recombination	One-point crossover
Recombination prob.	70%
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	500
Survival selection	Age based
Termination condition	No improvement in last 25 generations

Select one point randomly, and exchange the parts of the parents after that point

Example illustration - knapsack

Population size	500
Initialization	Random
Parent selection	Tournament selection with size 2
Recombination	One-point crossover
Recombination prob.	70%
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	500
Survival selection	Age based
Termination condition	No improvement in last 25 generations

Flip each bit of a solution with probability $1/n$

Example illustration - knapsack

Population size	500
Initialization	Random
Parent selection	Tournament selection with size 2
Recombination	One-point crossover
Recombination prob.	70%
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	500
Survival selection	Age based
Termination condition	No improvement in last 25 generations

The 500 offspring form the next population directly

Example illustration - knapsack

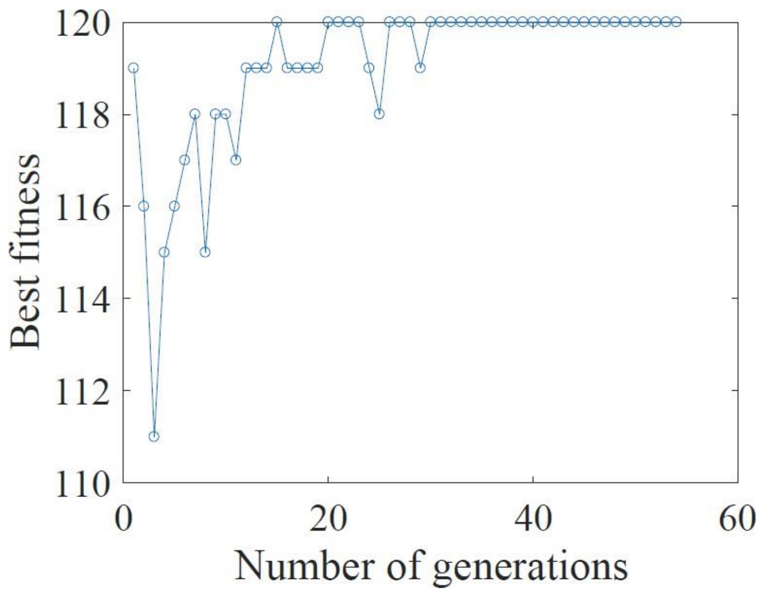
Population size	500
Initialization	Random
Parent selection	Tournament selection with size 2
Recombination	One-point crossover
Recombination prob.	70%
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	500
Survival selection	Age based
Termination condition	No improvement in last 25 generations

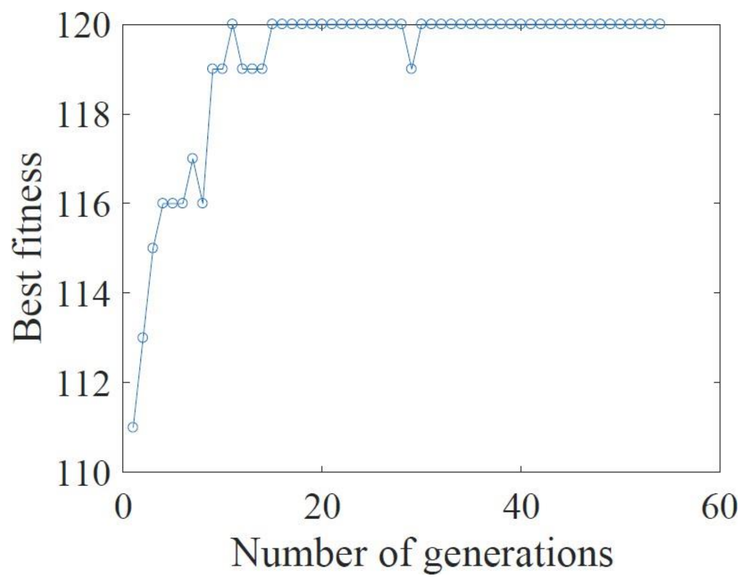
Example illustration - knapsack

Example: $n = 20, W_{max} = 100$

v	4	18	1	16	5	9	3	19	7	13	10	6	5	1	2	17	12	12	2	15
w	6	11	6	12	16	14	4	16	11	18	2	3	7	7	19	16	12	12	9	18

Run 1: Randomized Run 2:

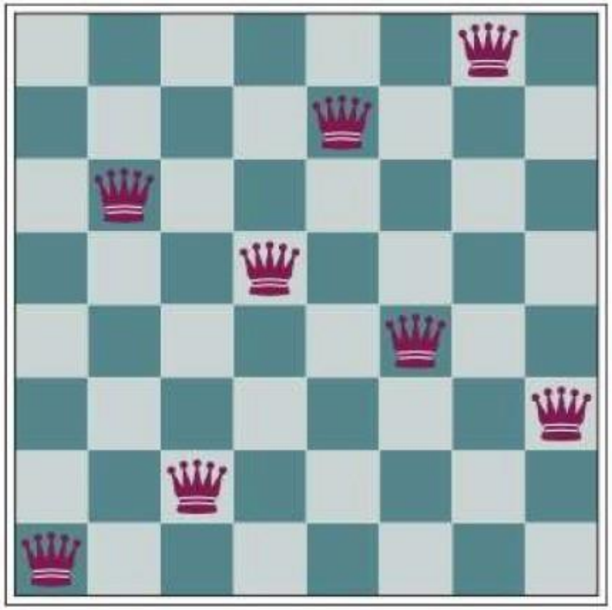




Example illustration - 8-queens

8-queens problem: to place eight queens on a chessboard such that no queen attacks any other

Fitness function f: number of nonattacking pairs of queens



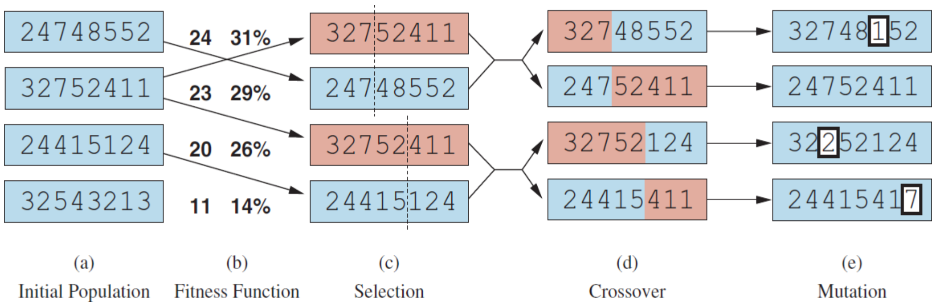
Solution representation

Integer vector

1	6	2	5	7	4	8	3

position of the queen on each column

Example illustration - 8-queens



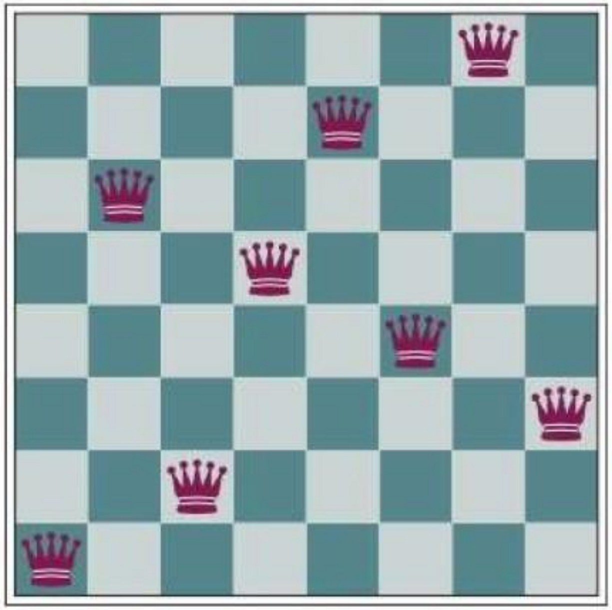
How about another setup?

Representation	Integer vector
Population size	4
Initialization	Random
Parent selection	Fitness proportional
Recombination	One-point crossover
Mutation	Bit-wise mutation
Mutation prob.	$1/n$
Number of offspring	4
Survival selection	fitness based
Termination condition	Reach the best fitness

Example illustration - 8-queens

8-queens problem: to place eight queens on a chessboard such that no queen attacks any other

Fitness function *f*: number of nonattacking pairs of queens



Solution representation

Permutation

1	6	2	5	7	4	8	3

position of the queen on each column

Genotype space is smaller than that of integer representation, but still contains the optimum

Example illustration - 8-queens

Representation	Permutation
Population size	100
Initialization	Random
Parent selection	Best 2 out of random 5
Recombination	Cut-and-crossfill crossover
Mutation	Swap
Mutation prob.	80%
Number of offspring	2
Survival selection	Fitness based
Termination condition	Reach the best fitness or 10,000 fitness evaluations

Select five solutions from the population uniformly at random, and choose the best two as the parent solutions

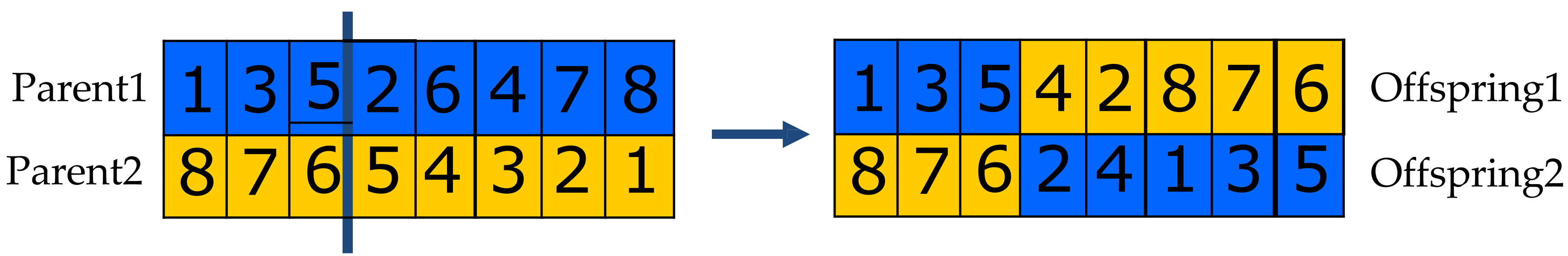
Example illustration - 8-queens

Representation	Permutation
Population size	100
Initialization	Random
Parent selection	Best 2 out of random 5
Recombination	Cut-and-crossfill crossover
Mutation	Swap
Mutation prob.	80%
Number of offspring	2
Survival selection	Fitness based
Termination condition	Reach the best fitness or 10,000 fitness evaluations

Example illustration - 8-queens

Cut-and-crossfill crossover:

1. Select a crossover point randomly;
2. Cut both parents into two segments at this point;
3. Copy the first segment of parent 1 into offspring 1 and the first segment of parent 2 into offspring 2;
4. Scan parent 2 after the crossover point and fill the second segment of offspring 1 with values from parent 2, skipping those that it already contains
5. Do the same for parent 1 and offspring 2



Example illustration - 8-queens

Representation	Permutation
Population size	100
Initialization	Random
Parent selection	Best 2 out of random 5
Recombination	Cut-and-crossfill crossover
Mutation	Swap
Mutation prob.	80%
Number of offspring	2
Survival selection	Fitness based
Termination condition	Reach the best fitness or 10,000 fitness evaluations

Swap values of two randomly chosen positions

1	3	5	2	6	4	7	8

→

1	3	7	2	6	4	5	8

Example illustration - 8-queens

Representation	Permutation
Population size	100
Initialization	Random
Parent selection	Best 2 out of random 5
Recombination	Cut-and-crossfill crossover
Mutation	Swap
Mutation prob.	80%
Number of offspring	2
Survival selection	Fitness based
Termination condition	Reach the best fitness or 10,000 fitness evaluations

Remove the worst two from the population and two offspring

Example illustration - 8-queens

Representation	Permutation
Population size	100
Initialization	Random
Parent selection	Best 2 out of random 5
Recombination	Cut-and-crossfill crossover
Mutation	Swap
Mutation prob.	80%
Number of offspring	2
Survival selection	Fitness based
Termination condition	Reach the best fitness or 10,000 fitness evaluations

Example illustration - 8-queens

Average of 100 independent runs

Setup 1:

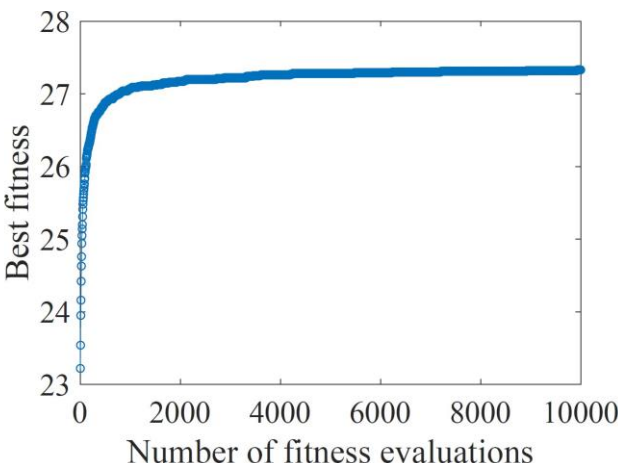
Setup 2:

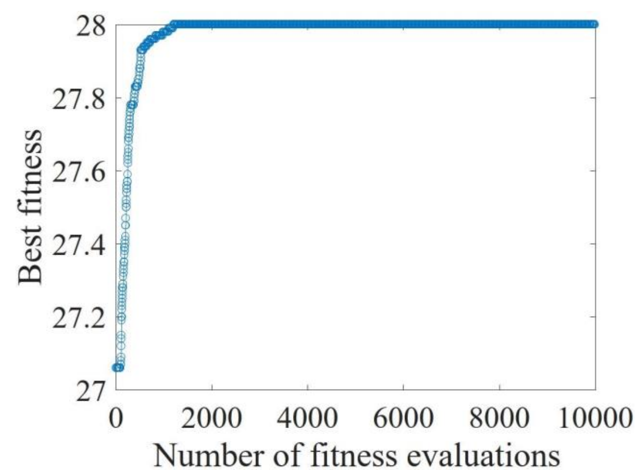
The number of fitness evaluations

7094

270

Curve change of the best fitness





The setup of components has a large influence on the performance

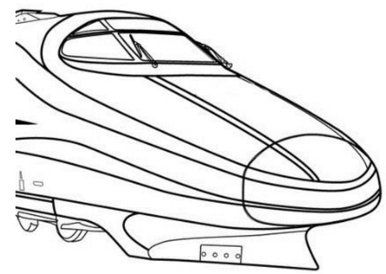
Application: High-speed train head design

Problem: optimize the efficiency of the train head

extremely hard to apply traditional optimization methods

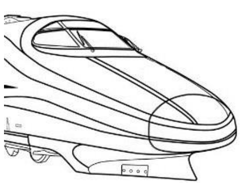
Representation:

parameterize

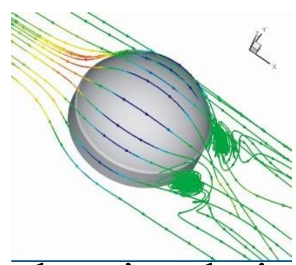


represented as a vector of parameters

Fitness:



x_i

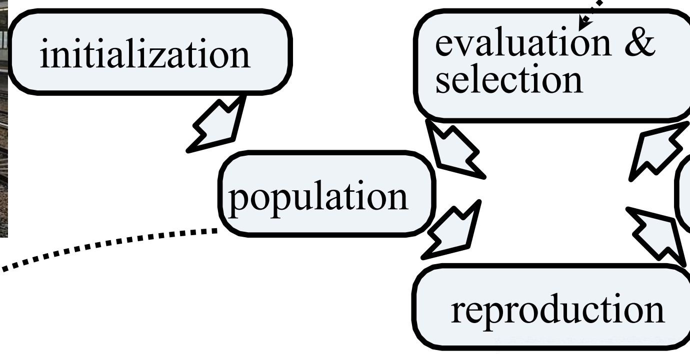


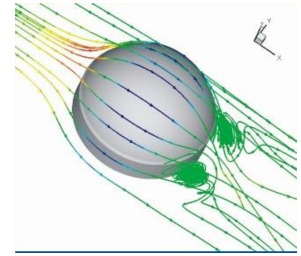
test by simulation

$f(x_i)$

Application: High-speed train head design

Series 700

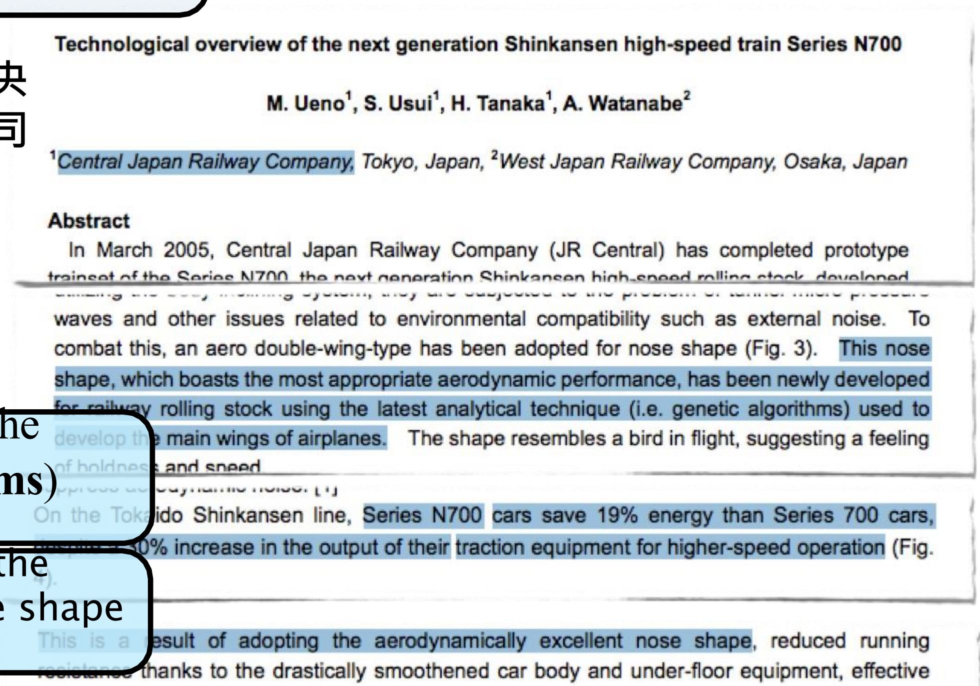




Series N700

节省 19% 能耗

日本中央
铁路公司



Technological overview of the next generation Shinkansen high-speed train Series N700

M. Ueno[1], S. Usui[1], H. Tanaka[1], A. Watanabe[2]

[1]*Central Japan Railway Company, Tokyo, Japan,* [2]*West Japan Railway Company, Osaka, Japan*

Abstract

In March 2005, Central Japan Railway Company (JR Central) has completed prototype trainset of the Series N700, the next generation Shinkansen high-speed rolling stock, developed

waves and other issues related to environmental compatibility such as external noise. To combat this, an aero double-wing-type has been adopted for nose shape (Fig. 3). This nose shape, which boasts the most appropriate aerodynamic performance, has been newly developed for railway rolling stock using the latest analytical technique (i.e. genetic algorithms) used to develop the main wings of airplanes. The shape resembles a bird in flight, suggesting a feeling of boldness and speed.

On the Tokaido Shinkansen line, Series N700 cars save 19% energy than Series 700 cars, despite a 30% increase in the output of their traction equipment for higher-speed operation (Fig. 4).

This is a result of adopting the aerodynamically excellent nose shape, reduced running resistance thanks to the drastically smoothened car body and under-floor equipment, effective

this nose ... has been newly developed ... using the latest analytical technique (i.e. **genetic algorithms**)

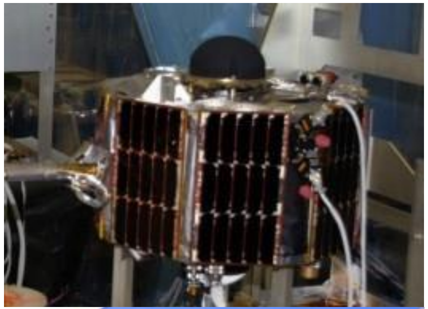
N700 cars save **19%** energy ... **30%** increase in the output... This is a result of adopting the ... nose shape

http://www.lamda.nju.edu.cn/qianc/

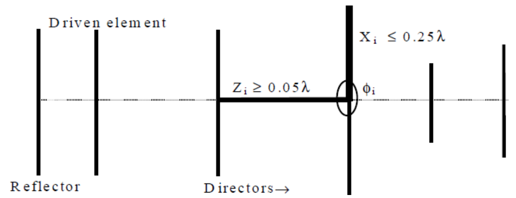
Application: Antenna design

Problem: optimize the efficiency of the antenna

extremely hard to apply traditional optimization methods



Representation:

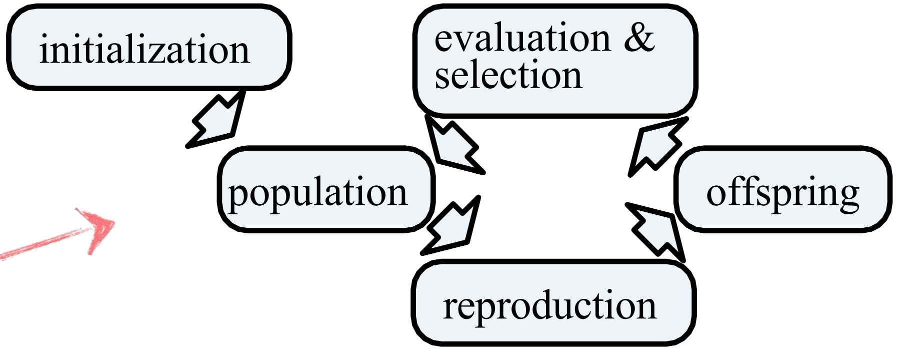


a sequence of operators
forward, rotate-x
rotate-y, rotate-z

Fitness by simulation test

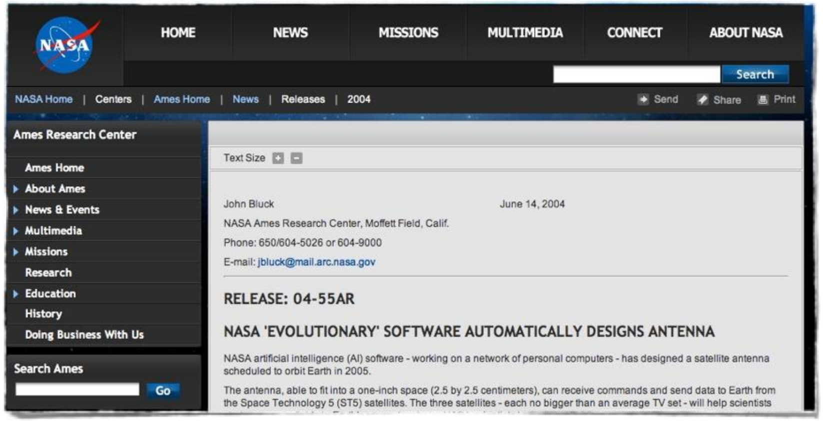
easy to test a given solution

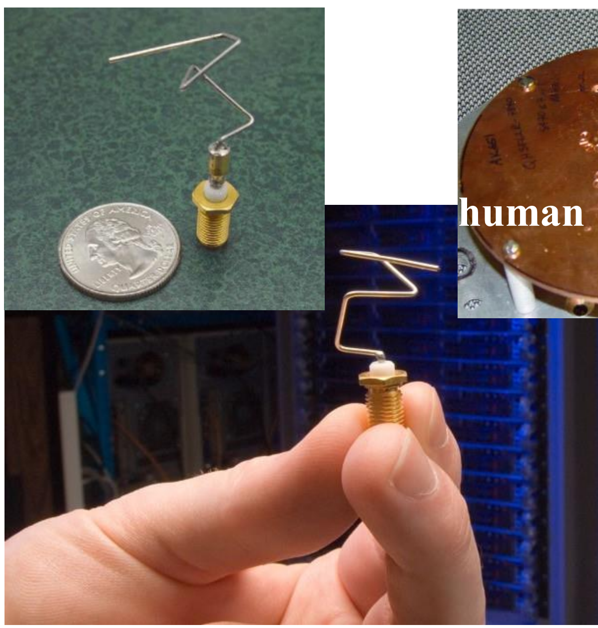
use EAs!

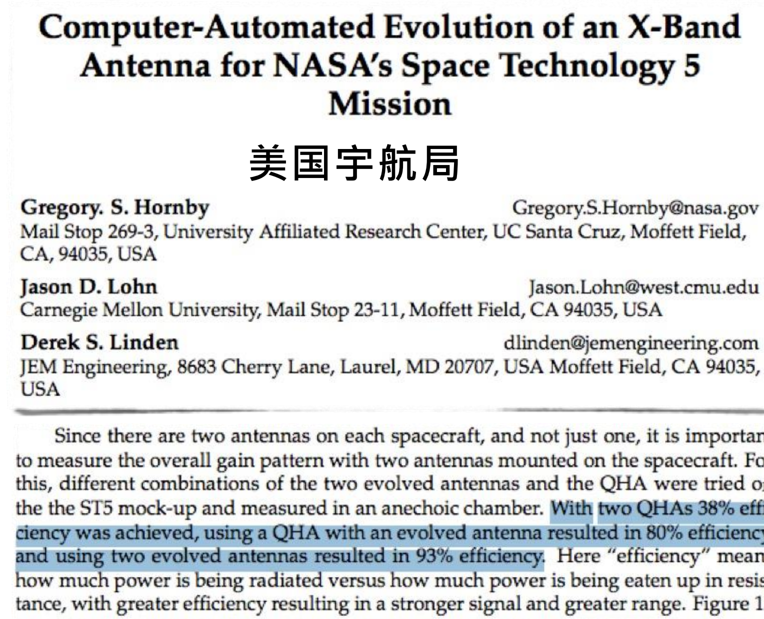


Application: Antenna design

功效由 38% 提升至 93%







Computer-Automated Evolution of an X-Band Antenna for NASA's Space Technology 5 Mission

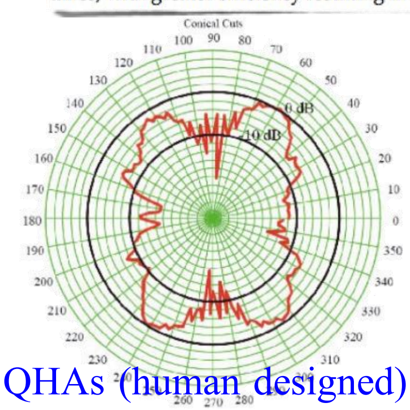
美国宇航局

Gregory. S. Hornby Gregory.S.Hornby@nasa.gov
Mail Stop 269-3, University Affiliated Research Center, UC Santa Cruz, Moffett Field, CA, 94035, USA

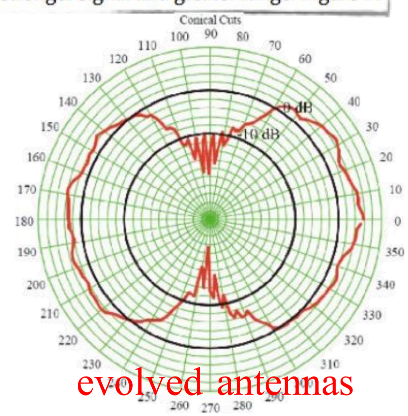
Jason D. Lohn Jason.Lohn@west.cmu.edu
Carnegie Mellon University, Mail Stop 23-11, Moffett Field, CA 94035, USA

Derek S. Linden dlinden@jemengineering.com
JEM Engineering, 8683 Cherry Lane, Laurel, MD 20707, USA Moffett Field, CA 94035, USA

Since there are two antennas on each spacecraft, and not just one, it is important to measure the overall gain pattern with two antennas mounted on the spacecraft. For this, different combinations of the two evolved antennas and the QHA were tried on the the ST5 mock-up and measured in an anechoic chamber. With two QHAs 38% efficiency was achieved, using a QHA with an evolved antenna resulted in 80% efficiency, and using two evolved antennas resulted in 93% efficiency. Here "efficiency" means how much power is being radiated versus how much power is being eaten up in resistance, with greater efficiency resulting in a stronger signal and greater range. Figure 11



QHAs (human designed)
38% efficiency



evolved antennas
93% efficiency

http://www.lamda.nju.edu.cn/qianc/

Application: Engine design

One genetic algorithm discovered a design for a gas turbine that went on to become the engine for the Boeing 777, made by General Electric.

It is almost 1 percent more efficient in its use of fuel than previous engines. In a mature field like gas turbines, 1 percent is a windfall.

[Newsweek, 1995]

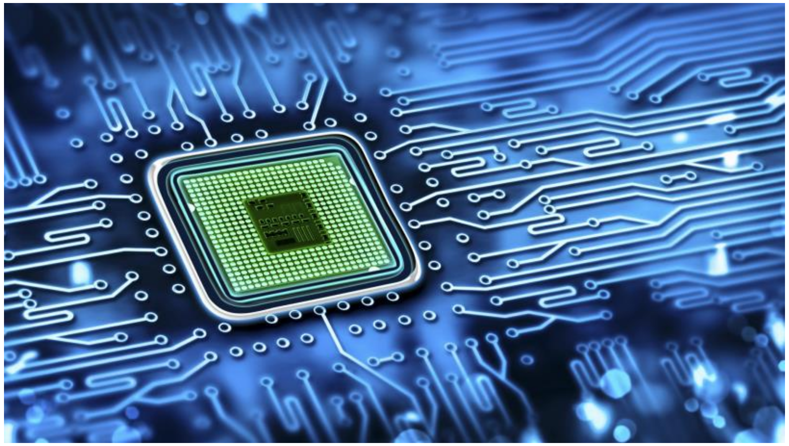
From https://www.inf.ed.ac.uk/teaching/courses/nat/slides/GA_cutout.pdf

Application: Chip design

Texas Instruments wanted to design a computer chip on the smallest piece of silicon possible.

A genetic algorithm came up with a circuit design that took up 18 percent less space, using a strategy of cross connections that no human had thought of.

"A genetic algorithm will usually come up with something very different from what a human would," says Illinois's Goldberg. "Then you have this 'aha' moment." [Newsweek, 1995]



From https://www.inf.ed.ac.uk/teaching/courses/nat/slides/GA_cutout.pdf

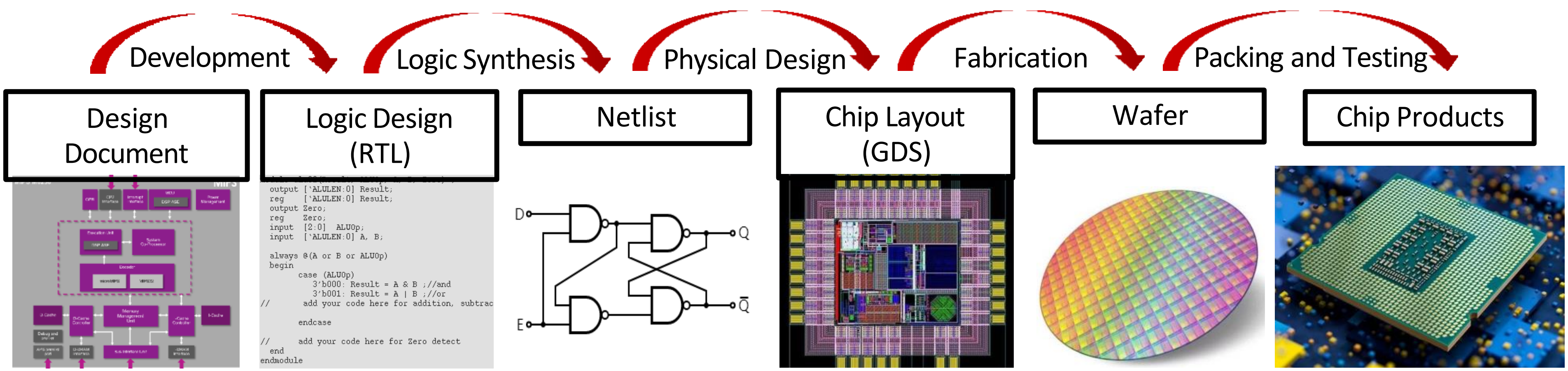
Application: Chip design

Function design and verification: Design the RTL and verify the functions. (Document -> RTL)

Logic synthesis: Mapping the RTL design into netlist. (RTL -> Netlist)

Physical design: Design the physical layout according to netlist by EDA tools. (Netlist -> GDS)

Chip manufacturing: Fabricate the chip from GDS layout by photolithography. (GDS -> Product)



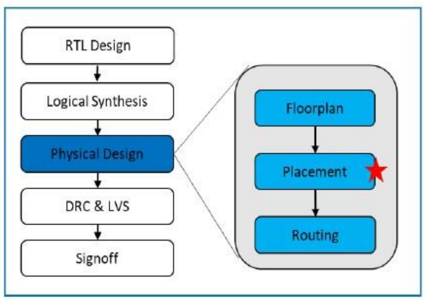
Application: Chip design

难题4：布局布线优化技术

出题组织：海思/诺亚方舟实验室　接口专家：许思源 xusiyuan520@huawei

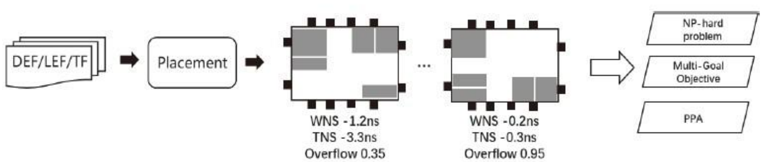
技术背景

芯片布局起着承上（逻辑综合）启下（布线）的作用，是现代超大规模集成电路设计物理设计流程中的一步，显著影响芯片设计的迭代周期。由于芯片布局被认为是NP-Complete的问题，因此如何快速生成高质量的芯片布局是布局问题的重要挑战。



Chip Design Flow　　Auto Placement and Route (APR) Flow

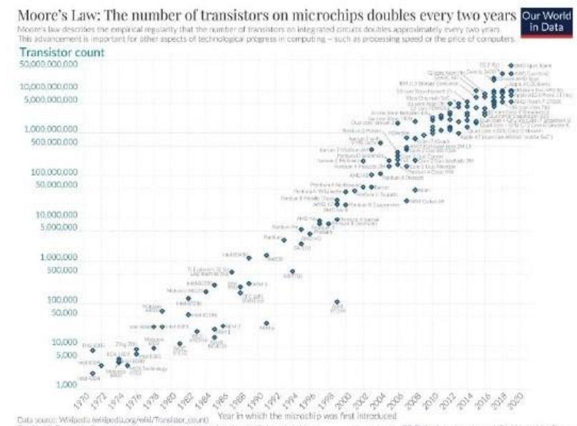
芯片布局问题可以描述为给定标准单元和宏单元的大小和连接关系，给出约束(如没有元件重叠)和优化目标(时序，拥塞，功耗，面积等)，基于特定策略确定每个单元的位置。



难题5：超高维空间多目标黑盒优化技术

出题组织：海思/诺亚方舟实验室　接口专家：胡守博 hushoubo@huawei.

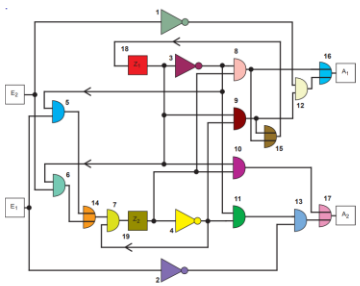
技术背景



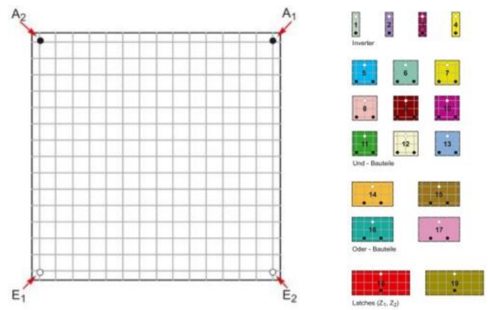
[Reference: https://en.wikipedia.org/wiki/Moore%27s_law#/media/File:Moore's_Law_Transistor_Count_1970-2020.png]

- 随着芯片制程日益演进，芯片微架构设计空间呈现几何增长导致设计空间爆炸的问题，因此无法通过穷举的方式确定最优的架构空间参数配置。
- 传统的最优参数选取依靠架构师选取有潜力的参数配置进行仿真确定，但随着设计空间的膨胀，专家经验的局限性导致人工结果距最优配置差距越来越大。
- 现有自动化最优参数寻优方案主要依赖黑盒优化算法（如贝叶斯优化，MCTS 等），但实际任务中的超高维设计空间寻优仍未很好解决，是当前主要挑战之一。

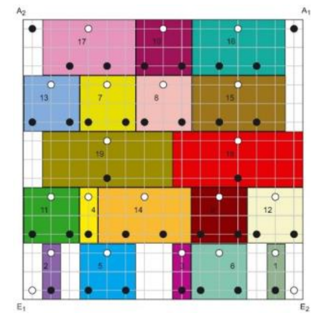
Application: Chip design



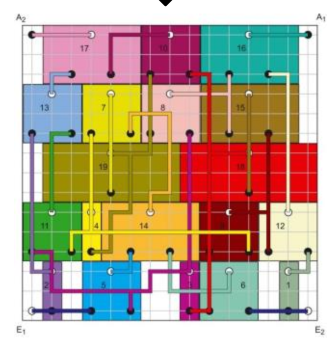
Netlist



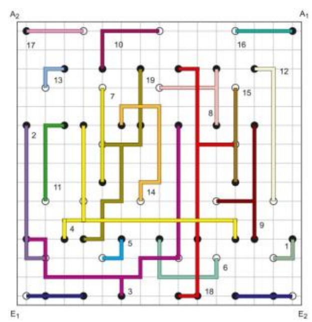
Canvas and modules



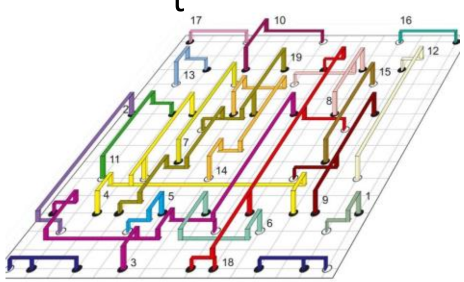
Placement



Routing



Routing result

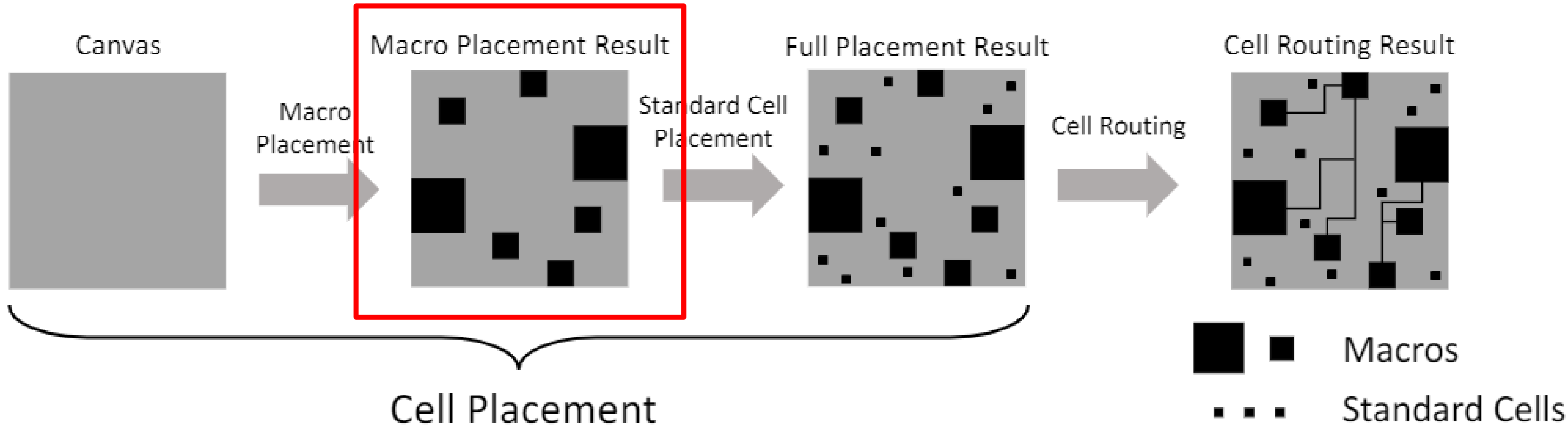


3D-vision

Figures are from http://www.or.uni-bonn.de/~vygen/files/buda.pdf

Application: Chip design

Macro Placement: an important task in chip floorplanning, which tries to determine the positions of all macros with the aim of optimizing PPA



Objective: power, HPWL, congestion …

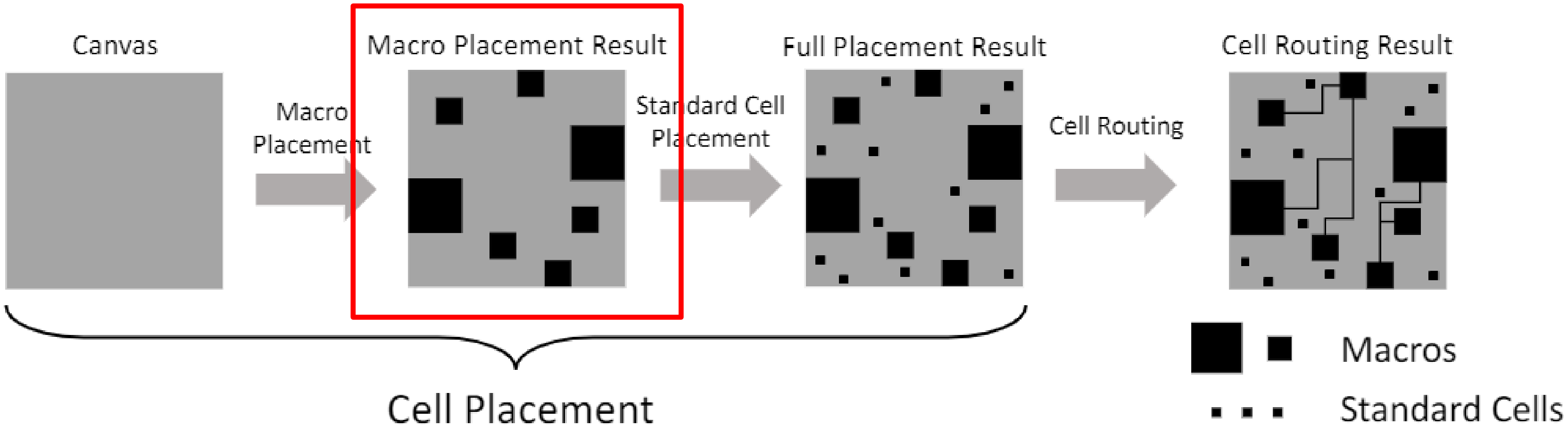
Constraint: non-overlapping

#macros: thousands

Black-box
Expensive
Multi-objective
High-dimension

Application: Chip design

Macro Placement: an important task in chip floorplanning, which tries to determine the positions of all macros with the aim of optimizing PPA



Existing approaches:

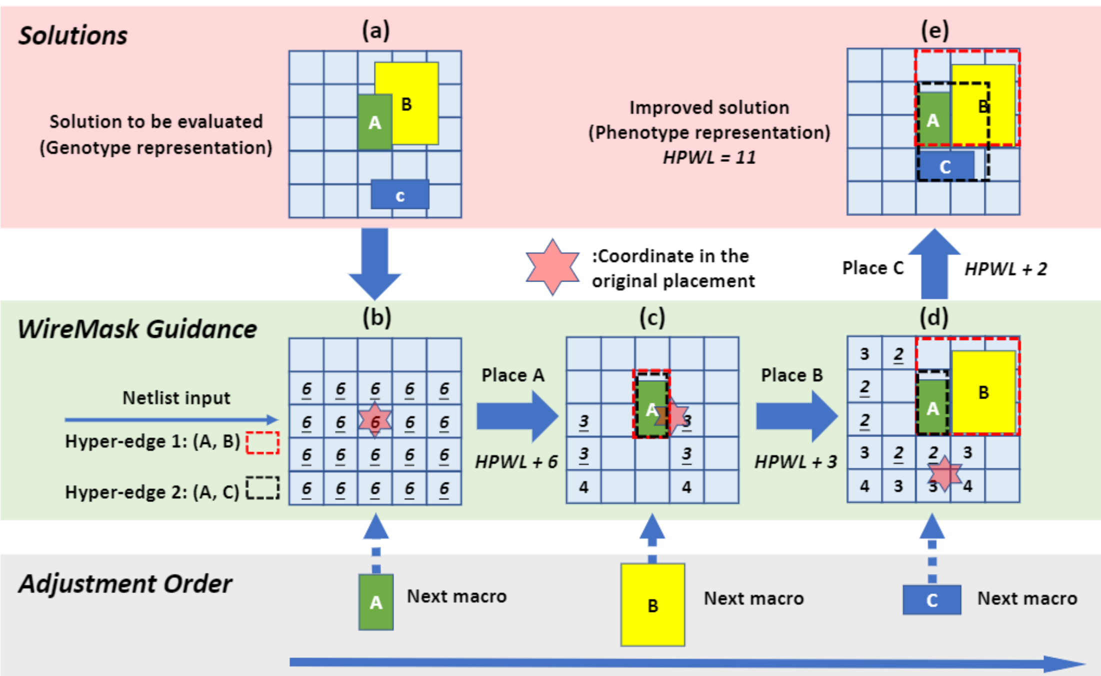
- Packing-based methods — *poor scalability*
- Analytical methods — *overlapping*
- Grid-based RL methods — *under-exploration*

Application: Chip design

Macro Placement by Wire-Mask-Guided Black-Box Optimization

Yunqi Shi, Ke Xue, Lei Song, Chao Qian. NeurIPS 2023

Greedy improvement guided by wire mask



Genotype-Phenotype Mapping

Improve efficiency,
Keep non-overlapping

Any black-box optimization algorithm

Good exploration

Application: Chip design

Comparison with state-of-the-art methods

Table 1: HPWL values ($\times 10^5$) obtained by ten compared methods on seven chips. Each result consists of the mean and standard deviation of five runs. The best (smallest) mean value on each chip is bolded. The symbols '+', '−' and '≈' indicate the number of chips where the result is significantly superior to, inferior to, and almost equivalent to WireMask-EA, respectively, according to the Wilcoxon rank-sum test with significance level 0.05.

Method	Type	adaptec1	adaptec2	adaptec3	adaptec4	bigblue1	bigblue3	bigblue4 ($\times 10^7$)	$+/-/\approx$	Avg. Rank
SP-SA [30]	Packing	18.84 ± 4.62	117.36 ± 8.73	115.48 ± 7.56	120.03 ± 4.25	5.12 ± 1.43	164.70 ± 19.55	25.49 ± 2.73	0/7/0	6.86
NTUPlace3 [10]	Analytical	26.62	321.17	328.44	462.93	22.85	455.53	48.38	0/7/0	9.00
RePlace [11]	Analytical	16.19 ± 2.10	153.26 ± 29.01	111.21 ± 11.69	37.64 ± 1.05	2.45 ± 0.06	119.84 ± 34.43	11.80 ± 0.73	1/6/0	5.28
DREAMPlace [25]	Analytical	15.81 ± 1.64	140.79 ± 26.73	121.94 ± 25.05	**37.41 ± 0.87**	2.44 ± 0.06	107.19 ± 29.91	12.29 ± 1.64	1/6/0	4.86
Graph [29]	RL	30.10 ± 2.98	351.71 ± 38.20	358.18 ± 13.95	151.42 ± 9.72	10.58 ± 1.29	357.48 ± 47.83	53.35 ± 4.06	0/7/0	9.00
DeepPR [13]	RL	19.91 ± 2.13	203.51 ± 6.27	347.16 ± 4.32	311.86 ± 56.74	23.33 ± 3.65	430.48 ± 12.18	68.30 ± 4.44	0/7/0	8.86
MaskPlace [23]	RL	6.38 ± 0.35	73.75 ± 6.35	84.44 ± 3.60	79.21 ± 0.65	2.39 ± 0.05	91.11 ± 7.83	11.07 ± 0.90	0/7/0	4.28
WireMask-RS	Ours	6.13 ± 0.05	59.28 ± 1.48	60.60 ± 0.45	62.06 ± 0.22	2.19 ± 0.01	62.58 ± 2.07	**8.20 ± 0.17**	0/5/2	2.57
WireMask-BO	Ours	6.07 ± 0.14	59.17 ± 3.94	61.00 ± 2.08	63.86 ± 1.01	2.14 ± 0.03	67.48 ± 6.49	8.62 ± 0.18	0/3/4	2.86
WireMask-EA	Ours	**5.91 ± 0.07**	**52.63 ± 2.23**	**57.75 ± 1.16**	58.79 ± 1.02	**2.12 ± 0.01**	**59.87 ± 3.40**	8.28 ± 0.25	–	**1.43**

packing

analytical

RL

Our methods

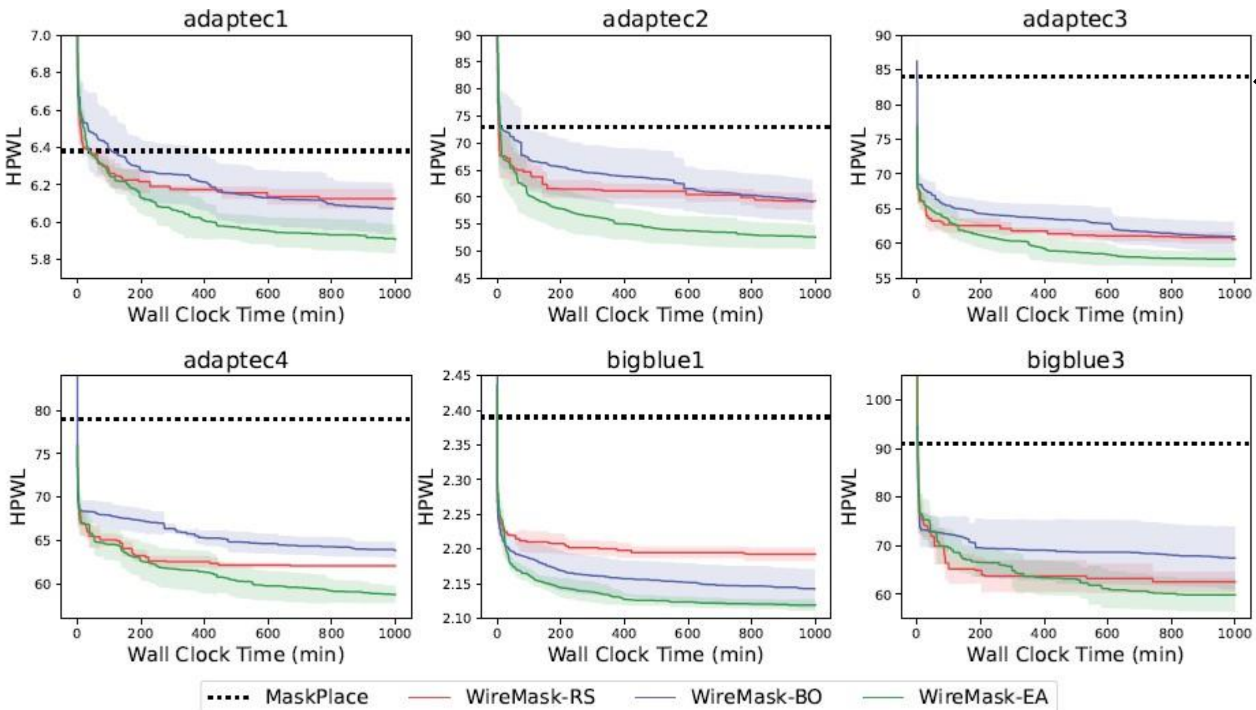
Nature 2021

WireMask-EA achieves the best average rank, and performs the best on 5 out of the 7 chips

WireMask-EA is significantly better than any previous method on at least 6 out of the 7 chips

Application: Chip design

Our methods surpass the SOTA method MaskPlace with an average of 8 minutes



MaskPlace [Lai et al., NeurIPS'22]

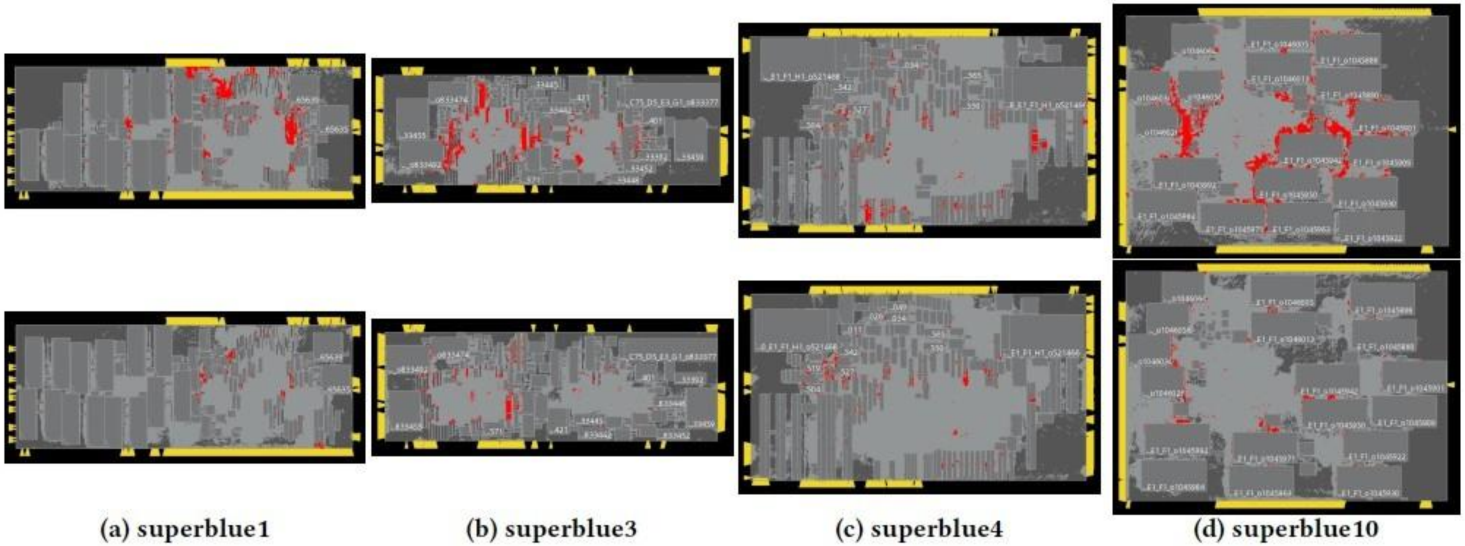
200-minute convergence

Figure 4: HPWL ($\times 10^5$) vs. wall clock time of WireMask-BBO, where the shaded region represents the standard error derived from 5 independent runs.

Application: Chip design

Application: Chip design

Comparison on congestion



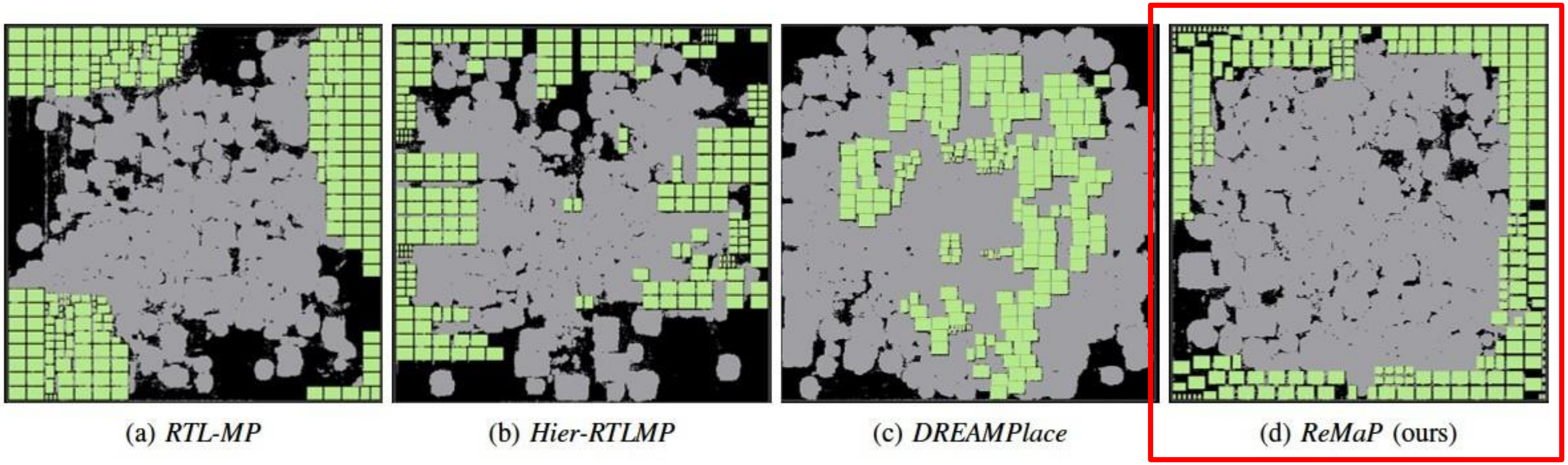
Multiple-DMP
[Lin et al., TCAD'20]

Our method:
less congested

Figure 5: Placement layouts and congestions of Multiple-DMP (top row) and Hybro-WireMask (bottom row) on the ICCAD 2015 benchmarks, superblue1, superblue3, superblue4, and superblue10. The congestion results are obtained by ***Cadence Innovus***, where red points indicate the congestion critical regions.

Application: Chip design

Comparison on regularity



Our method: better regularity

[Shi, Lin, Xu, Kai, Xue, Yuan, Qian, and Zhou, DAC 2025]

Application: Chip design

Comparison on timing metrics

Benchmark	DREAMPlace* [20]		DREAMPlace 4.0* [18]		Differentiable-TDP† [12]		Distribution-TDP§ [19]		Ours	
	TNS	WNS	TNS	WNS	TNS	WNS	TNS	WNS	TNS	WNS
superblue1	-262.44	-18.87	-85.03	-14.10	-74.85	-10.77	-42.10	-9.26	**-17.44**	**-7.75**
superblue3	-76.64	-27.65	-54.74	-16.43	-39.43	-12.37	-26.59	-12.19	**-20.40**	**-11.82**
superblue4	-290.88	-22.04	-144.38	-12.78	-82.92	**-8.49**	-123.28	-8.86	**-82.88**	-9.17
superblue5	-157.82	-48.92	-95.78	-26.76	-108.08	-25.21	-70.35	-31.64	**-62.18**	**-24.65**
superblue7	-141.55	-19.75	-63.86	**-15.22**	-46.43	**-15.22**	-95.89	-17.24	**-43.52**	**-15.22**
superblue10	-731.94	-26.10	-768.75	-31.88	**-558.05**	**-21.97**	-691.10	-25.86	-558.14	-23.08
superblue16	-453.57	-17.71	-124.18	-12.11	-87.03	-10.85	-55.99	-12.21	**-22.90**	**-8.63**
superblue18	-96.76	-20.29	-47.25	-11.87	-19.31	-7.99	-19.23	**-5.25**	**-16.16**	-6.92
Average Ratio	6.90	2.07	2.75	1.40	2.00	1.09	1.68	1.11	**1.00**	**1.00**

Average improvement:

40.5% in total negative slack (TNS)

8.3% in worst negative slack (WNS)

[Shi, Xu, Kai, Lin, Xue, Yuan, and Qian, DATE 2025, Best Paper Award]

Application: Chip design

Dear Authors,

As every year, DATE is honouring the authors of one excellent paper of each of its tracks D,A,T,E with a best paper award (BPA).
As the DATE 2025 Awards Chair and in the name of the Programme Chair Theocharis Theocharides and the Track Chairs of the D,A,T,E tracks, it is my great pleasure to let you know that your paper

Timing-Driven Global Placement by Efficient Critical Path Extraction

has been selected by an independent committee to receive the BPA for Track D of DATE 2025 in Lyon, France.

Congratulations!

获芯片设计自动化领域
顶级国际会议**DATE’25**
最佳论文奖

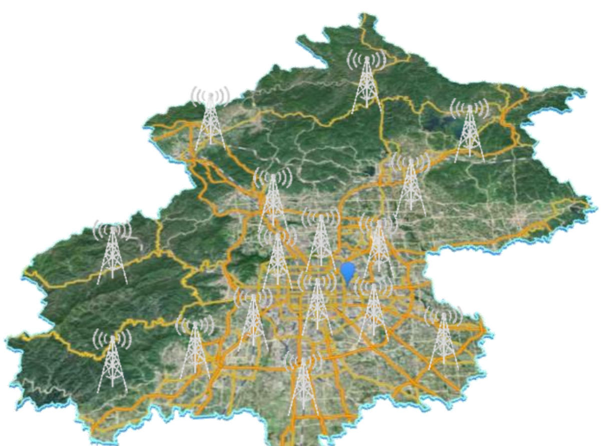
[Shi, Xu, Kai, Lin, Xue, Yuan, and Qian, DATE 2025, Best Paper Award]

Application: Wireless network optimization

华为部署了大量的无线网络系统

线网部署

问题规模大
数万小区，**千万参数**

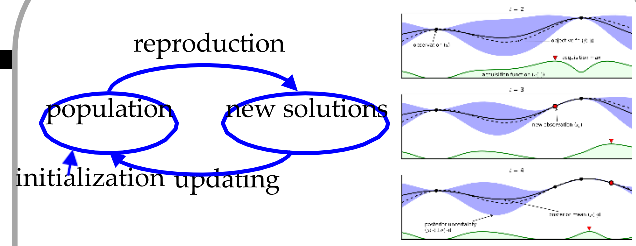


业务种类多

参数下发

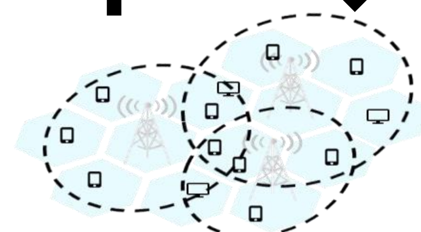
模型建立

参数优化



评估

优化



Digital Twin 模拟器

无线网络参数优化问题极其重要

Application: Wireless network optimization

难题5：[优化决策]大规模复杂网络中多参数耦合、多目标竞争下快速寻优

出题组织：公共开发部　　接口专家: 赖卓航 laizhuohang@huawei.com

揭榜挂帅难题

技术背景

无线网络优化，是指通过对网络参数进行优化调整，保证网络质量与用户体验最优。由于无线网络存在物理传播环境复杂、网络状态动态变化、多制式/频谱/硬件共存、网络指标相互冲突等特点，无线网络的参数优化问题一般具备以下特征：

- **参数规模大/优化维度高：** 网络小区个数多（可达10W个），每个小区有100+可调整的网络参数，参数间关联关系复杂，导致需同时优化大量网络参数（如：可能高达1000W个）
- **多目标优化：** 网络优化可能需要同时达成多个目标，如同时优化网络的信号覆盖强度、信号干扰、用户感知速率等，且优化目标之间可能存在冲突。优化时需同时考虑多个目标的综合最优
- **寻优效率要求高：** 由于网络状态随时间动态变化，需对网络进行频繁调整以适应动态环

技术诉求

大规模黑盒参数优化问题高效求解　设计面向无线网络优化场景的高性能大规模黑盒参数优化方案，使
法端到端优化时长≤ 1 min(@准实时场景)， ≤ 15 min(@非实时场景)，目标数3-10个，对象数1W-1
100-400个寻优参数/对象

http://www.lamda.nju.edu.cn/qianc/

Application: Protein Sequence Optimization

The Nobel Prize in Chemistry 2018

© Nobel Media AB. Photo: A. Mahmoud

Frances H. Arnold

Prize share: 1/2

© Nobel Media AB. Photo: A. Mahmoud

George P. Smith

Prize share: 1/4

© Nobel Media AB. Photo: A. Mahmoud

Sir Gregory P. Winter

Prize share: 1/4

The Nobel Prize in Chemistry 2018 was divided, one half awarded to Frances H. Arnold "for the directed evolution of enzymes", the other half jointly to George P. Smith and Sir Gregory P. Winter "for the phage display of peptides and antibodies."

"Evolution—the adaption of species to different environments—has created an enormous diversity of life. **Frances Arnold has used the same principles – genetic change and selection – to develop proteins that solve humankind's chemical problems. In 1993, Arnold conducted the first directed evolution of enzymes, which are proteins that catalyze chemical reactions.** The uses of her results include more environmentally friendly manufacturing of chemical substances, such as pharmaceuticals, and the production of renewable fuels."

Application: Biological evolution

南京大學 NANJING UNIVERSITY | 新闻网 NJU NEWS 南京大学新闻中心主办

新闻关键字搜索

首 页 综合新闻 专题新闻 理论园地 讲话与部署 南雍号 媒

南大首创这条“高清曲线” 列入中国十大科技进展

在1月20日由中国科学院和中国工程院院士评选出的“**2020年中国十大科技进展**”中，南京大学樊隽轩、沈树忠领衔研发的“**全球第一条高精度的古生代3亿多年的海洋生物多样性变化曲线**”，名列榜单，成功入选。

首页 - 综合新闻 2020-01-17 作者：地球科学与工程学院 来源：地球科学与工程学院

《Science》刊登南京大学地球科学与工程学院研究成果：大数据和超算揭秘古生代海洋生物多样性演化

北京时间1月17日，国际权威期刊《Science》以研究长文的形式在线发表了南京大学、中国科学院南京地质古生物所樊隽轩教授、沈树忠院士等的论文“A high-resolution summary of Cambrian to Early Triassic marine invertebrate biodiversity ”。该研究利用古生物大数据、超算和遗传算法等全新的方法和手段，基于化石记录重现了生命演化历史，改变了当前对古生代海洋生物多样性演化的认知。

最近更新

如何让专利“活”起来？全国百所高校聚...
2020.10.15

扬子江生态文明创新中心首届理事会召开...
2020.10.15

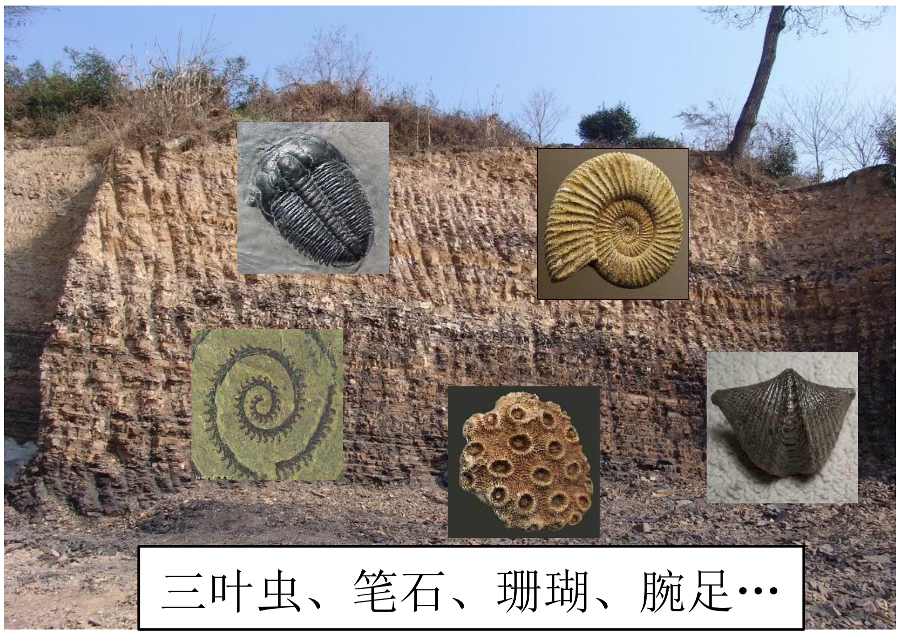
电子科学与工程学院启动“星火培优”学...
2020.10.15

我校举行“墨子杯”兵棋推演大赛校内选...

Application: Biological evolution

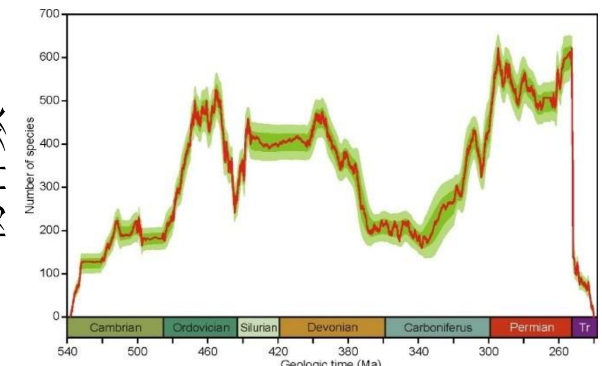
自然科学四大基础科学问题之一：生命起源与演化

地层剖面中海量化石记录数据



生物多样性变化曲线

物种数



地质时期

利用化石记录重现生命演化历史

序列优化问题：为不同物种的“首现”和“末现”事件排序，

使其与地层剖面中观测到的化石数据尽可能一致

Application: Biological evolution

简单例子

（西摩岛上采集的剖面 A）

两条剖面： **Seymour Is. Section A**

Seymour Is. Section F

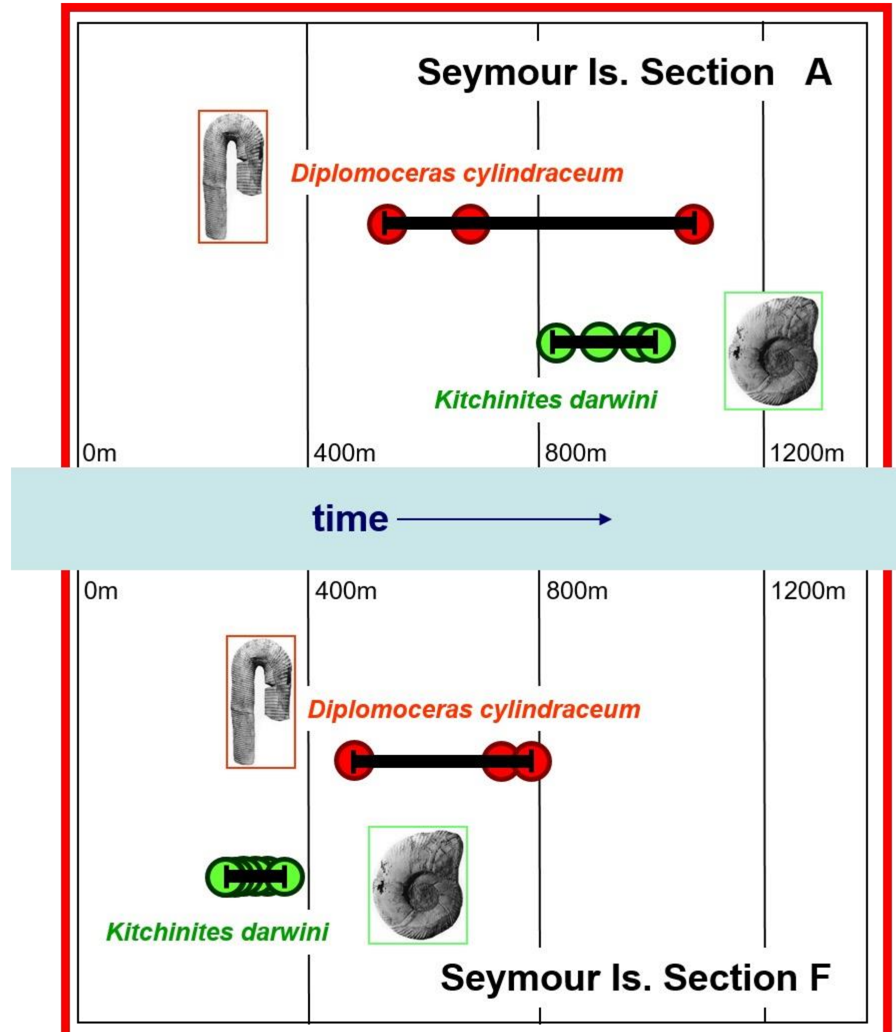
两个物种：  

四个事件：  首现、末现

 首现、末现

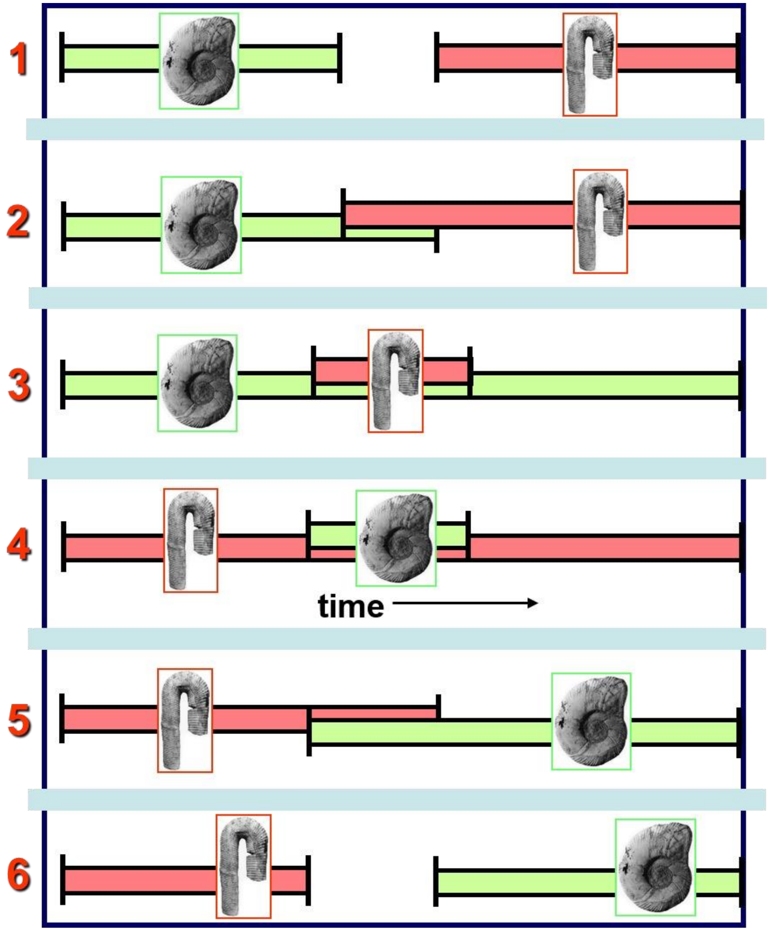
问题：为这四个事件排序，

使其与右图中观测到的数据尽可能一致



Application: Biological evolution

四个事件的所有可能序列（即所有可能的解）



. . green+ . . green- . . red+ . . red- . .

. . green+ . . red+ . . green- . . red- . .

. . green+ . . red+ . . red- . . green- . .

. . red+ . . green+ . . green- . . red- . .

. . red+ . . green+ . . red- . . green- . .

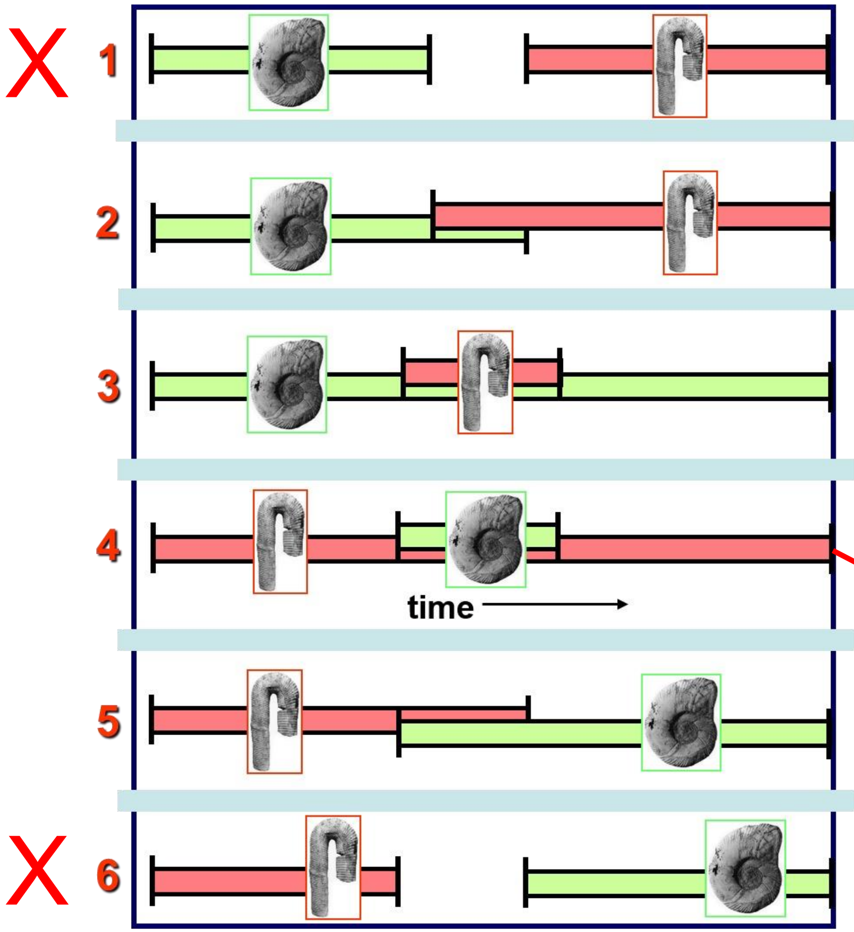
. . red+ . . red- . . green+ . . green- . .

\+ 首现

− 末现

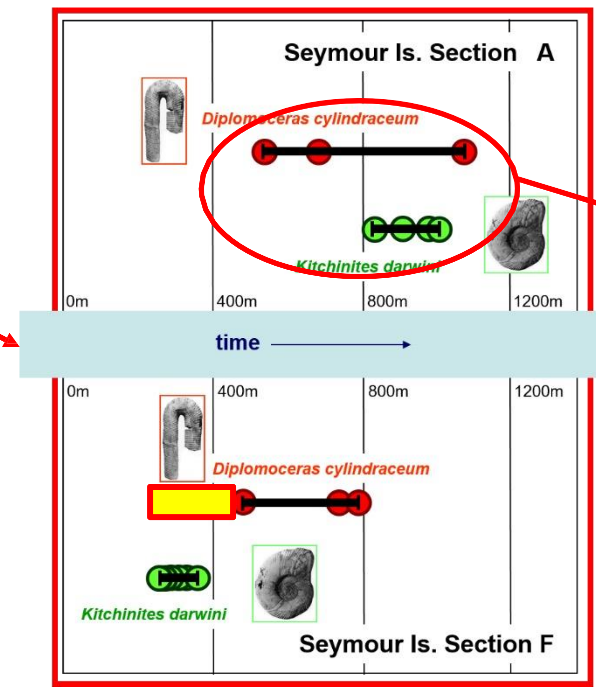
Application: Biological evolution

四个事件的所有可能序列（即所有可能的解）



目标：延限延展量　　越小越好

为使实际观测数据与序列保持一致，
对数据所需做的延展量



共生约束

Application: Biological evolution

四个事件的所有可能序列

目标：延限延展量　　越小越好

X 1

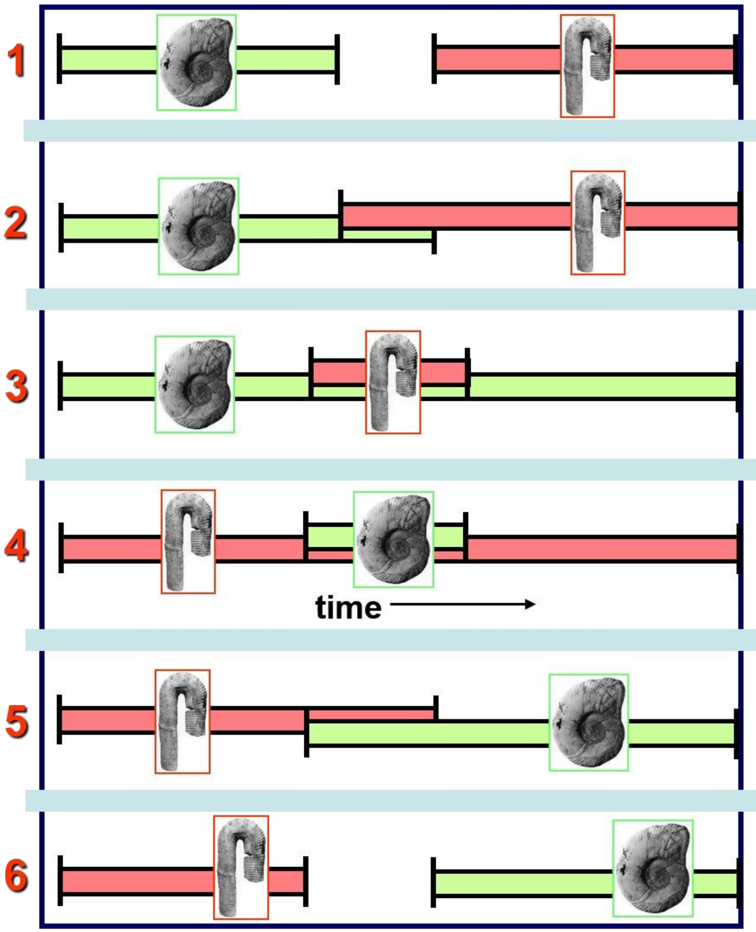
2

3

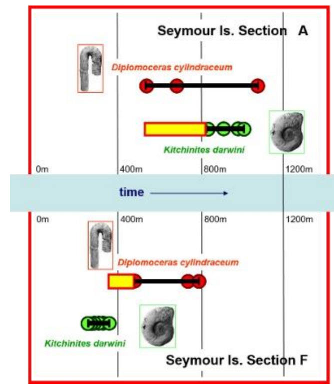
最优序列 4

5

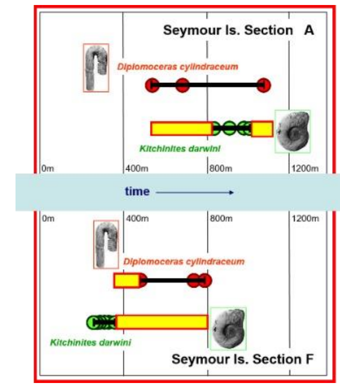
X 6



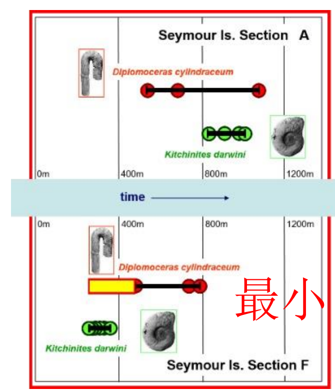
序列 2 的目标值



序列 3 的目标值

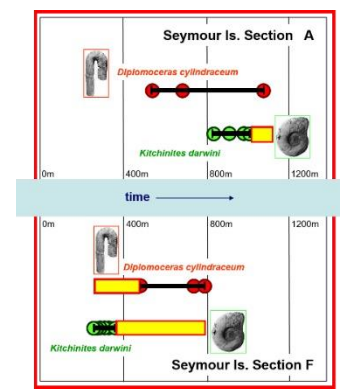


序列 4 的目标值



最小

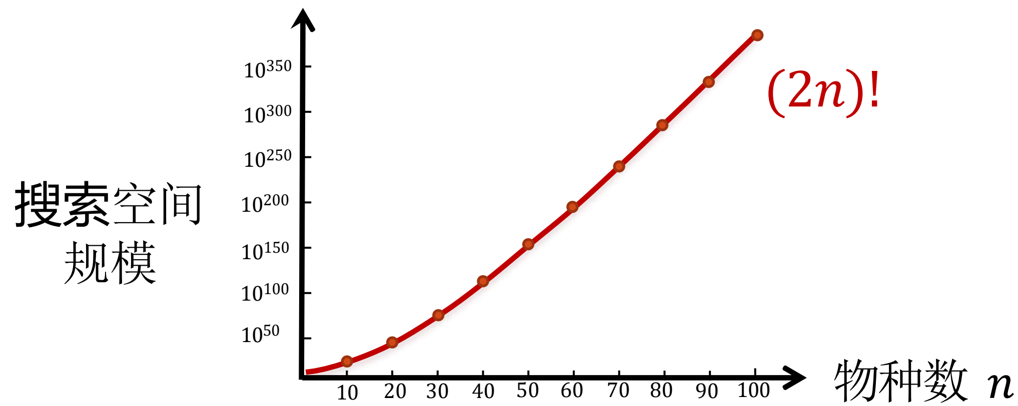
序列 5 的目标值



Application: Biological evolution

实际问题非常复杂

搜索空间规模
关于物种数呈指数级增长

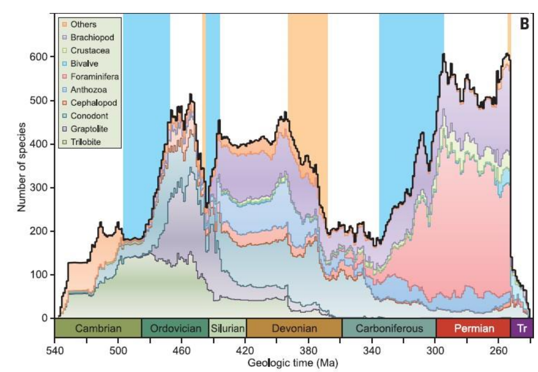


南大樊隽轩教授、沈树忠院士等人

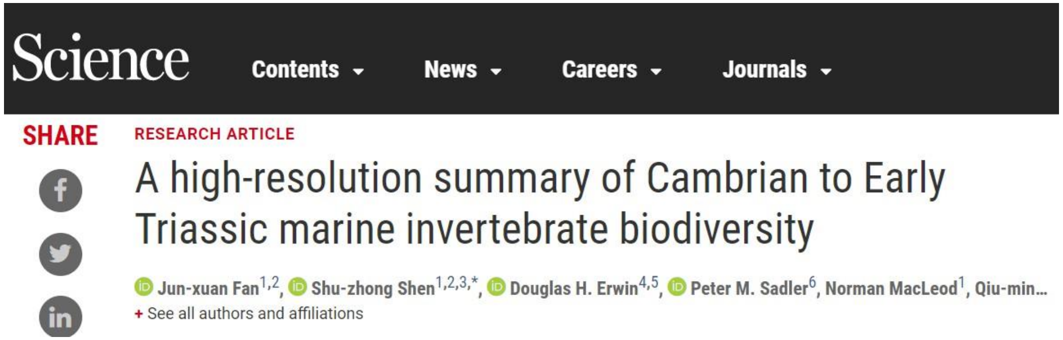
全球第一条高精度
海洋生物多样性变化曲线

中国的地层剖面数据
3122个剖面
11268个物种

模拟退火
“天河2号”
700万核时



Application: Biological evolution



Science Contents News Careers Journals

SHARE

RESEARCH ARTICLE

A high-resolution summary of Cambrian to Early Triassic marine invertebrate biodiversity

Jun-xuan Fan[1,2], Shu-zhong Shen[1,2,3,*], Douglas H. Erwin[4,5], Peter M. Sadler[6], Norman MacLeod[1], Qiu-min...

+ See all authors and affiliations

Science：“新的数据集和方法，推动整个演化生物学的变革”

Nature：“古生物学家以惊人的细节绘制地球3亿年历史”

2020年中国科学十大进展

Thanks J. Fan and X. Hou for providing the figures

南大樊隽轩教授、沈树忠院士等人

全球第一条高精度海洋生物多样性变化曲线

中国的地层剖面数据

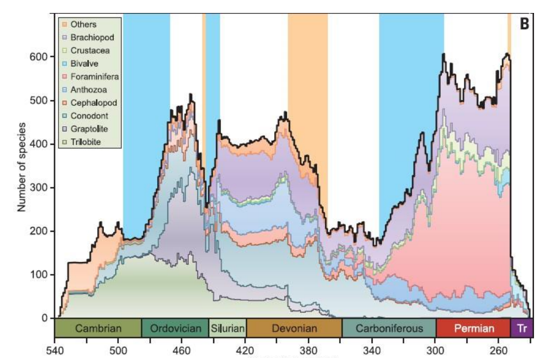
3122个剖面

11268个物种

模拟退火

“天河2号”

700万核时



Application: Biological evolution

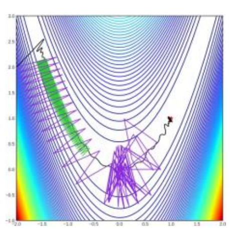
合作破解地球
生命演化的奥秘！

已有算法　　　　　　不适用于更大规模的数据

地层剖面数据	搜索空间规模	“天河2号”
中国：3122个剖面、11268个物种	22536!	700万核时
全世界：约8000个剖面、30000个物种	60000!	**不可计算**

合作提出针对该问题的演化算法

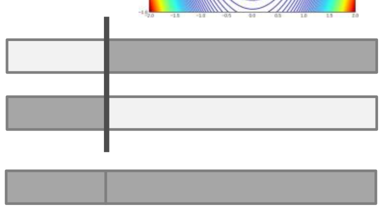
结合先验知识的初始化策略



自适应变异算子

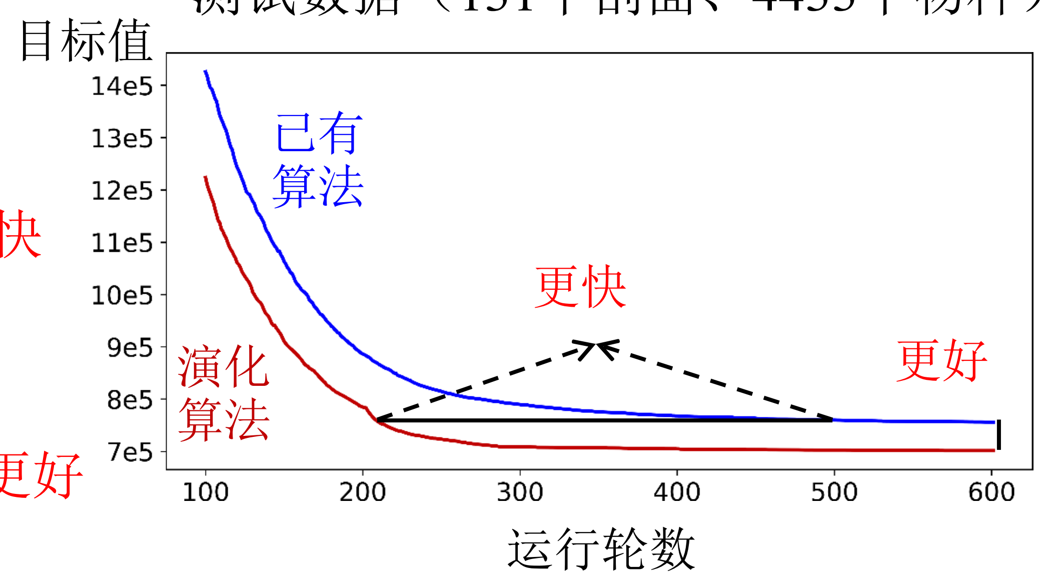
更快

交叉算子



更好

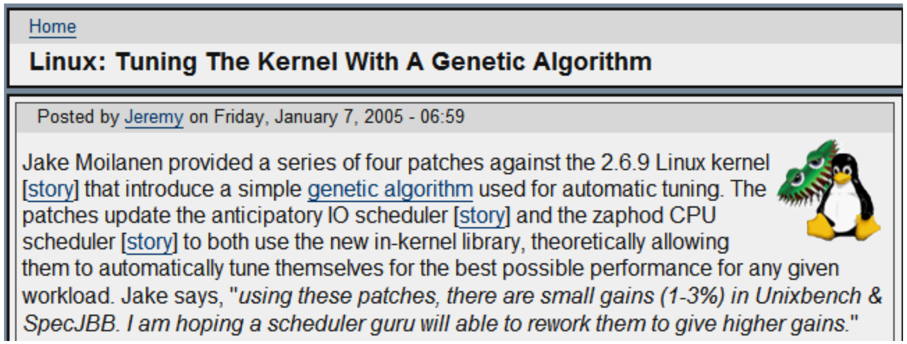
测试数据（131个剖面、4433个物种）



http://www.lamda.nju.edu.cn/qianc/

And more

optimizing operating systems:



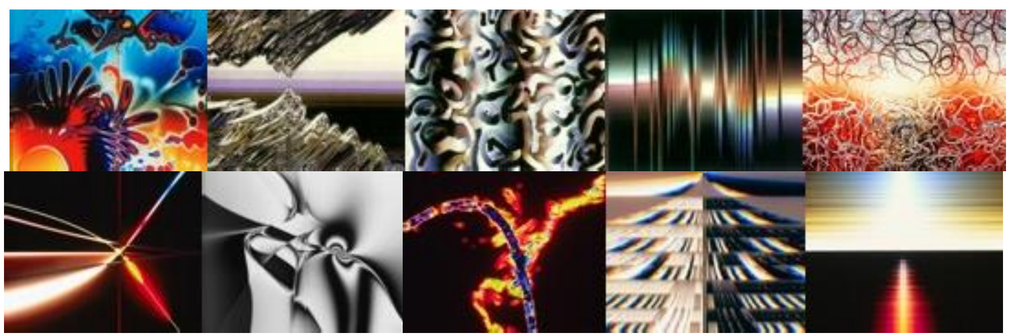
Home

Linux: Tuning The Kernel With A Genetic Algorithm

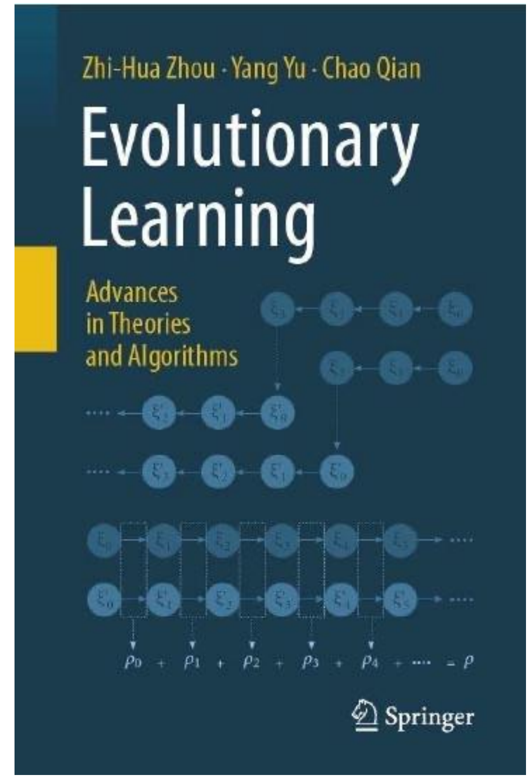
Posted by Jeremy on Friday, January 7, 2005 - 06:59

Jake Moilanen provided a series of four patches against the 2.6.9 Linux kernel [story] that introduce a simple genetic algorithm used for automatic tuning. The patches update the anticipatory IO scheduler [story] and the zaphod CPU scheduler [story] to both use the new in-kernel library, theoretically allowing them to automatically tune themselves for the best possible performance for any given workload. Jake says, "*using these patches, there are small gains (1-3%) in Unixbench & SpecJBB. I am hoping a scheduler guru will able to rework them to give higher gains.*"

interactive art design:



machine learning:

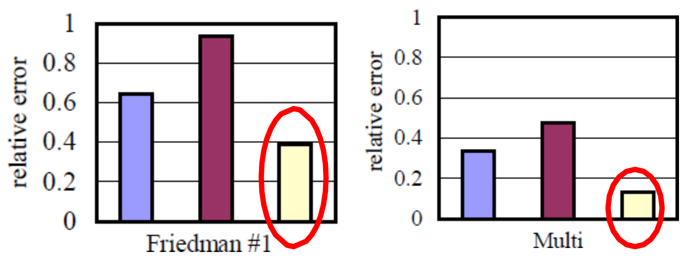


search-based software engineering

And more

Evolutionary learning has yielded encouraging empirical outcomes

Evolutionary selective ensemble



achieves smaller error by using fewer learners

[Zhou et al., AIJ'02]

Evolutionary neural architecture search

Study	Params.	C10+	C100+	Reachable?
Maxout (Goodfellow et al., 2013)	–	90.7%	61.4%	No
Network in Network (Lin et al., 2013)	–	91.2%	–	No
All-CNN (Springenberg et al., 2014)	1.3 M	92.8%	66.3%	Yes
Deeply Supervised (Lee et al., 2015)	–	92.0%	65.4%	No
Highway (Srivastava et al., 2015)	2.3 M	92.3%	67.6%	No
ResNet (He et al., 2016)	1.7 M	93.4%	72.8%†	Yes
Evolution (ours)	5.4 M	94.6%		N/A
	40.4 M		77.0%	
Wide ResNet 28-10 (Zagoruyko & Komodakis, 2016)	36.5 M	96.0%	80.0%	Yes
Wide ResNet 40-10+d/o (Zagoruyko & Komodakis, 2016)	50.7 M	96.2%	81.7%	No
DenseNet (Huang et al., 2016a)	25.6 M	96.7%	82.8%	No

achieves competitive performance to the hand-designed models

[Real et al., ICML'17]

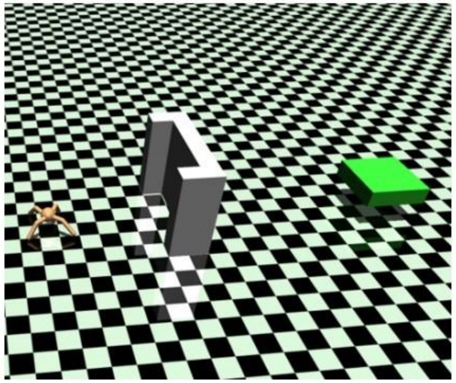
And more

Evolutionary learning has yielded encouraging empirical outcomes

Evolutionary reinforcement learning

Environment	EDO-CS	QD-RL	ME-ES	DvD-ES	CVT-ES	NSR-ES	Vanilla ES
HalfCheetahFwd	**4284**	2930	2700	-3419	3219	1346	-5543
HalfCheetahBwd	**6548**	6013	5953	6353	4672	5366	3911
AntFwd	**4617**	4291	4316	4507	3856	1737	1911
AntBwd	**4697**	4164	4123	3498	2958	3961	-851
Performance Ranking	**1**	3	3.5	3.75	4.75	5.25	6.75

achieves a set of policies with both high quality and diversity [Wang et al., ICLR'22]



(a) *AntWall-v0* environment

Evolutionary multitask learning

Model	imagenet2012	cifar100	cifar10
ViT L/16 fine-tuning (Dosovitskiy et al., 2021)	85.30	93.25	99.15
μ2Net after 5 task iterations	86.38	94.75	99.35
μ2Net after 10 task iterations	86.66	94.67	99.38
μ2Net cont. after adding VTAB-full tasks	**86.74**	94.67	99.41
μ2Net cont. after adding VDD tasks	**86.74**	94.74	99.43
μ2Net cont. after adding all 69 tasks	**86.74**	**94.95**	**99.49**

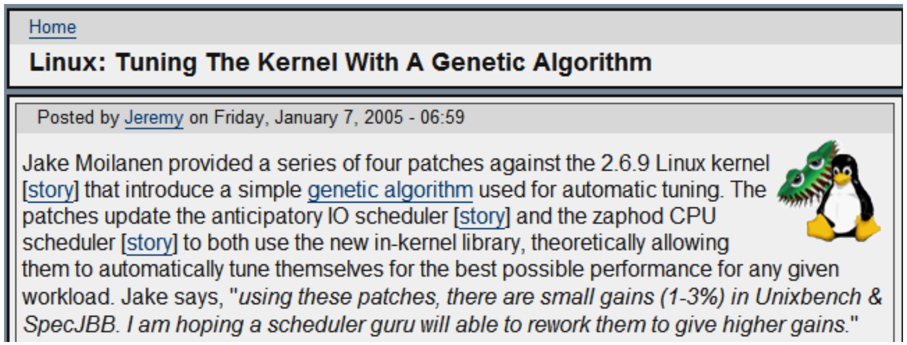
achieves competitive results on 69 public image classification tasks [Gesmundo & Dean, 2022]

better

SOTA: 99.40% [Touvron et al., ICCV'21]

And more

optimizing operating systems:

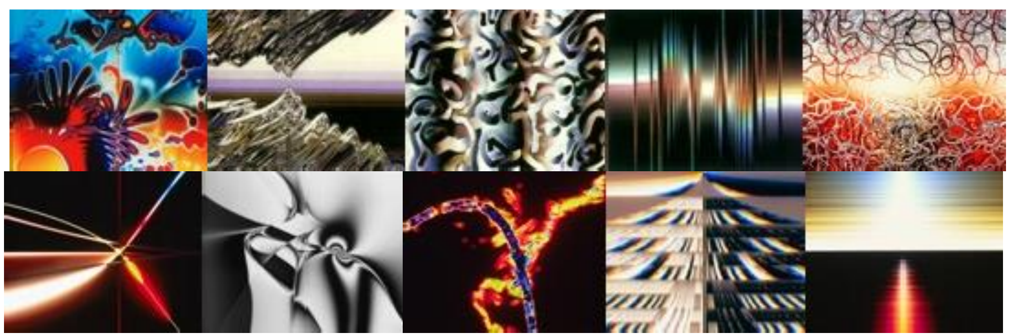

Home

Linux: Tuning The Kernel With A Genetic Algorithm

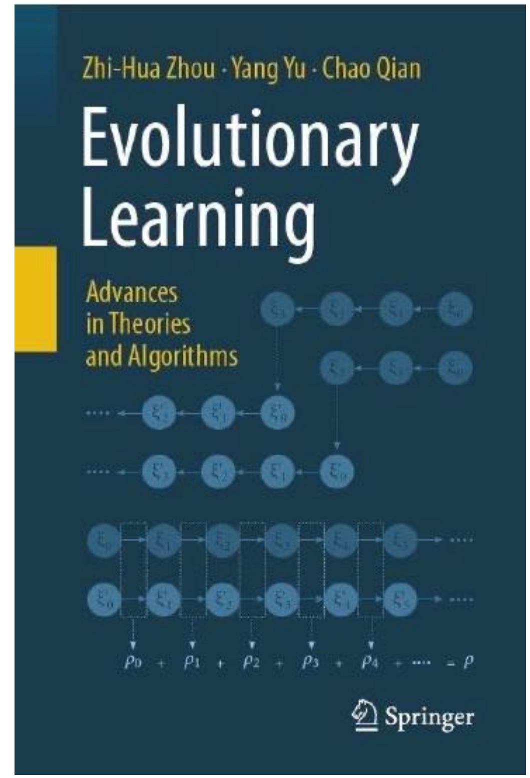
Posted by Jeremy on Friday, January 7, 2005 - 06:59

Jake Moilanen provided a series of four patches against the 2.6.9 Linux kernel [story] that introduce a simple genetic algorithm used for automatic tuning. The patches update the anticipatory IO scheduler [story] and the zaphod CPU scheduler [story] to both use the new in-kernel library, theoretically allowing them to automatically tune themselves for the best possible performance for any given workload. Jake says, "*using these patches, there are small gains (1-3%) in Unixbench & SpecJBB. I am hoping a scheduler guru will able to rework them to give higher gains.*"

interactive art design:



machine learning:



As long as solutions can be evaluated, EAs can be applied

Summary

- Evolutionary algorithms: Origins
- Evolutionary algorithms: Components
- Evolutionary algorithms: Applications

References

- K. A. De Jong. Evolutionary Computation – A Unified Approach. Chapter 2.
- A. E. Eiben and J. E. Smith. Introduction to Evolutionary Computing. Chapters 2-3.
- J. Fan, et al. A high-resolution summary of Cambrian to Early Triassic marine invertebrate biodiversity. Science, 367: 272–277, 2020
- Y. Shi, K. Xue, L. Song, and C. Qian. Macro placement by wire-mask-guided black-box optimization. NeurIPS'23.

References

- K. Xue, X. Lin, Y. Shi, S. Kai, S. Xu, and C. Qian. Escaping local optima in global placement. DAC'24 poster

- K. Xue., R.-T. Chen, X. Lin, Y. Shi, S. Kai, S. Xu, and C. Qian. Reinforcement learning policy as macro regulator rather than macro placer. NeurIPS'24.

- Y. Shi, S. Xu, S. Kai, X. Lin, K. Xue, M. Yuan, and C. Qian. Timing-driven global placement by efficient critical path extraction. DATE'25, **Best Paper Award**.

- Y. Shi, X. Lin, S. Xu, S. Kai, K. Xue, M. Yuan, C. Qian, and Z.-H. Zhou. ReMaP: Macro placement by recursively prototyping and periphery-guided relocating. DAC'25.