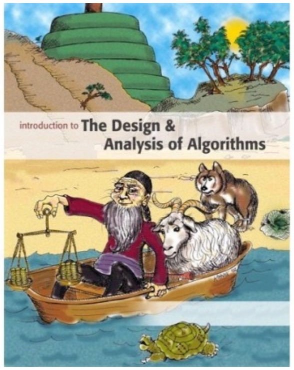
Introduction to

Algorithm Design and Analysis

[1] Model of Computation



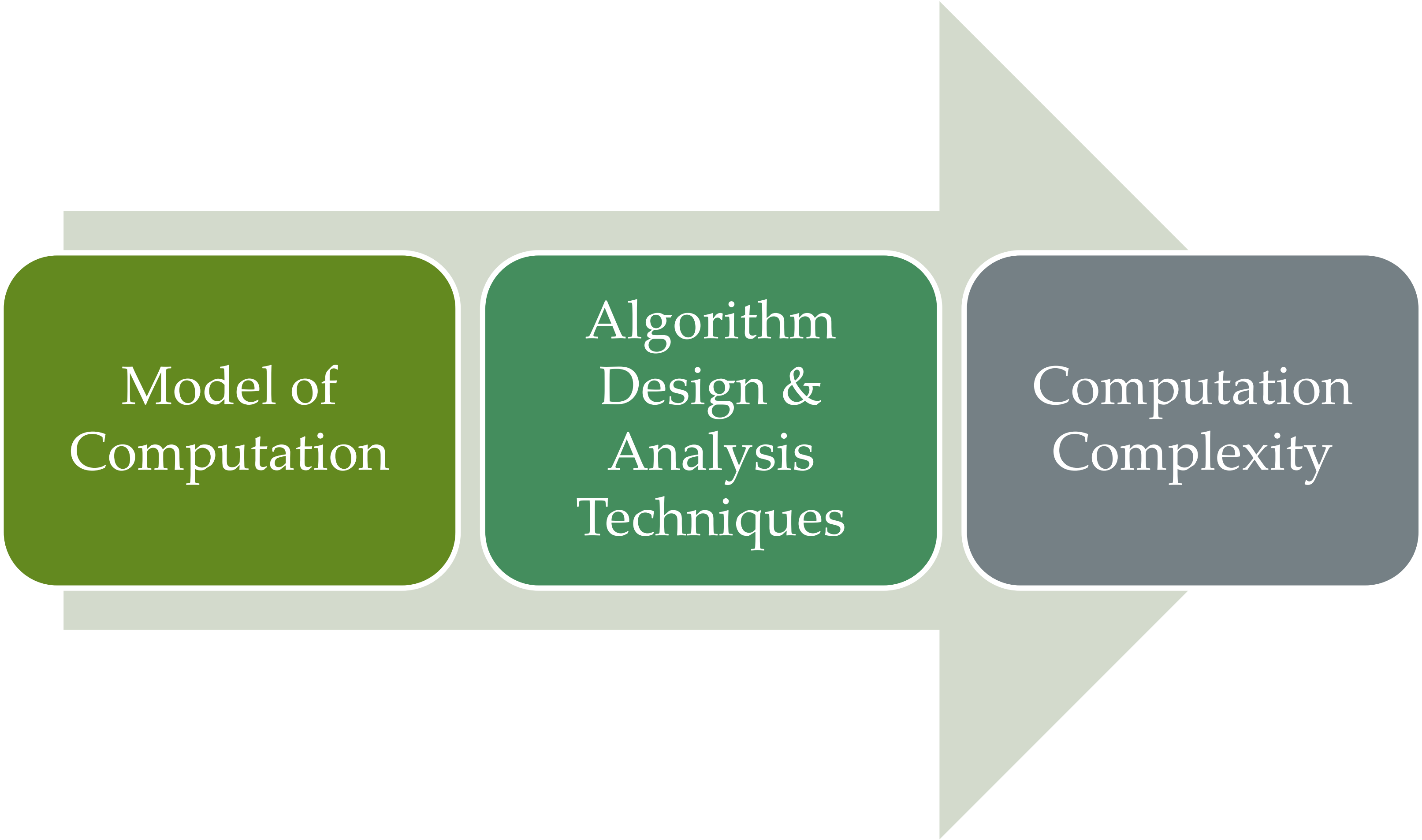
Yu Huang

http://cs.nju.edu.cn/yuhuang
Institute of Computer Software
Nanjing University

Course Information

- Syllabus
- Textbook
- Website

Syllabus

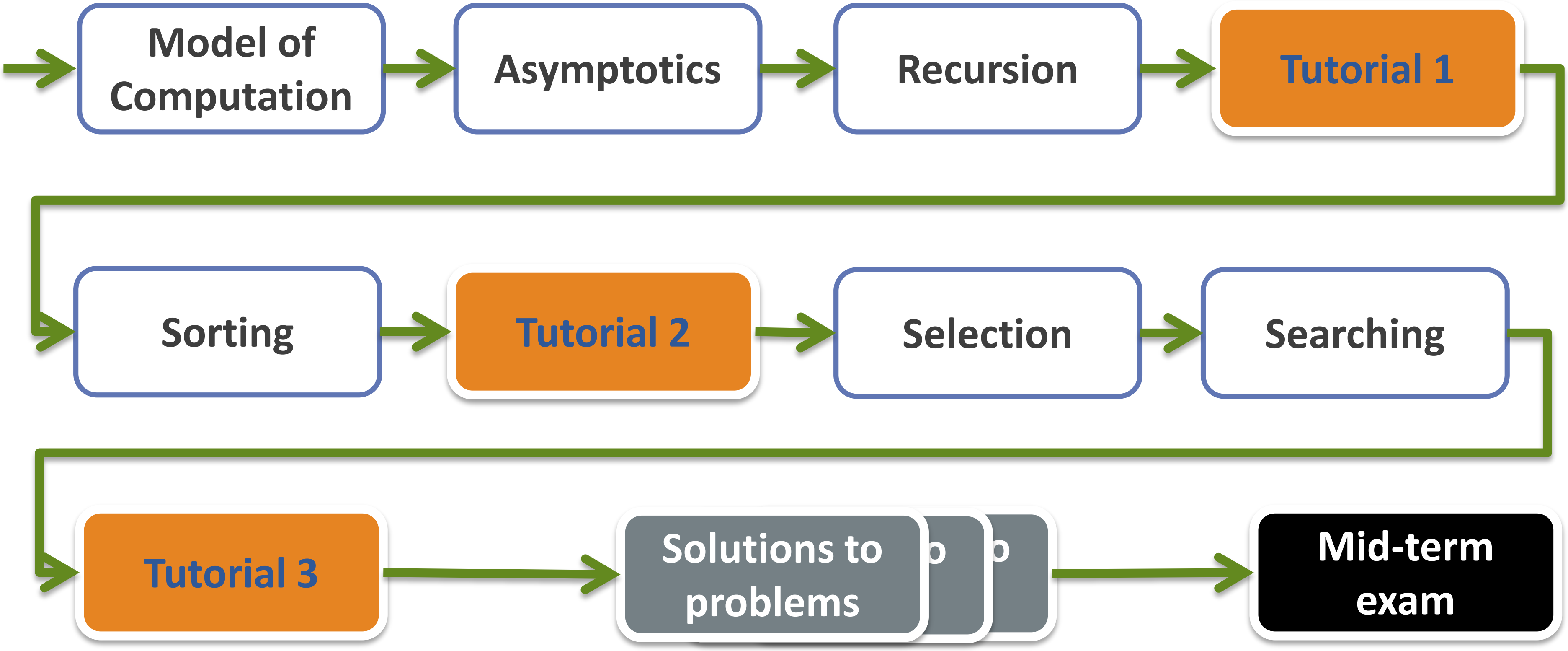


Model of Computation

Algorithm Design & Analysis Techniques

Computation Complexity

Syllabus



Model of Computation

Asymptotics

Recursion

Tutorial 1

Sorting

Tutorial 2

Selection

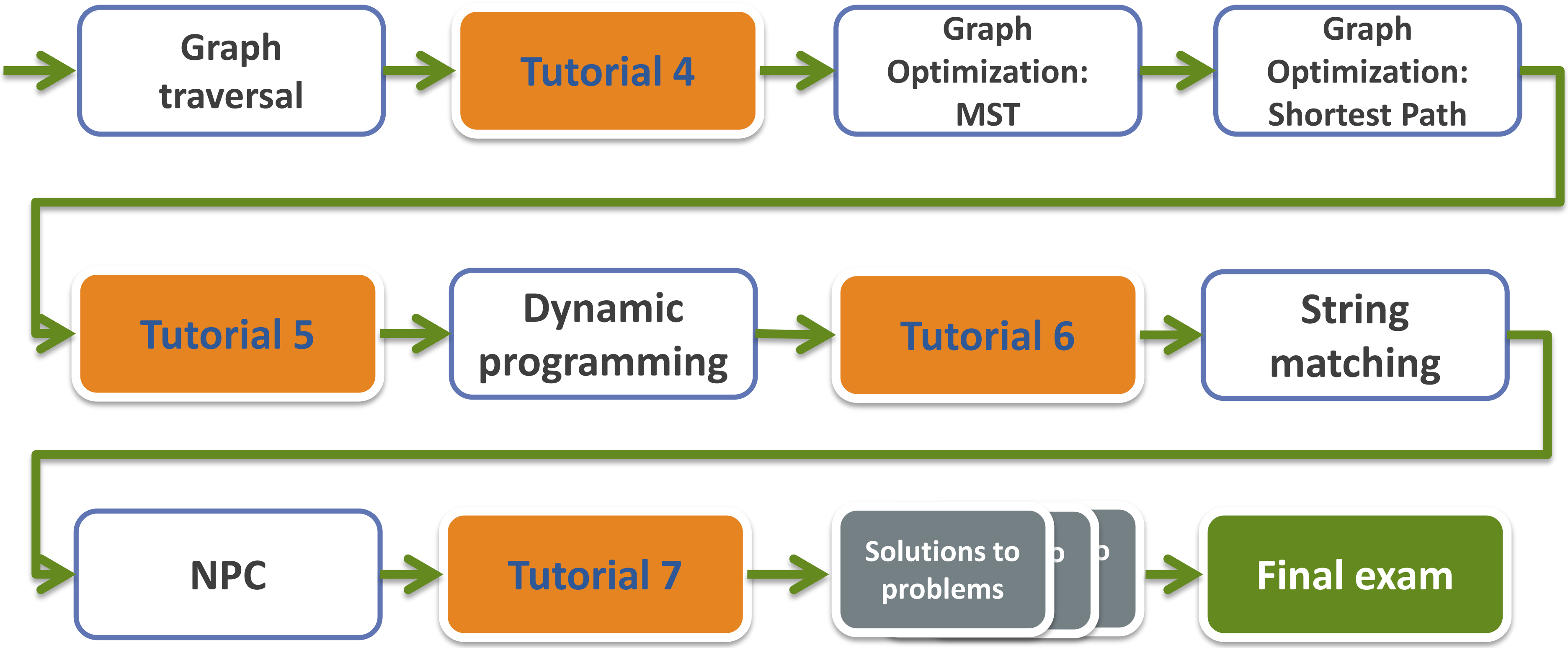
Searching

Tutorial 3

Solutions to problems

Mid-term exam

Syllabus



Graph traversal

Tutorial 4

Graph Optimization: MST

Graph Optimization: Shortest Path

Tutorial 5

Dynamic programming

Tutorial 6

String matching

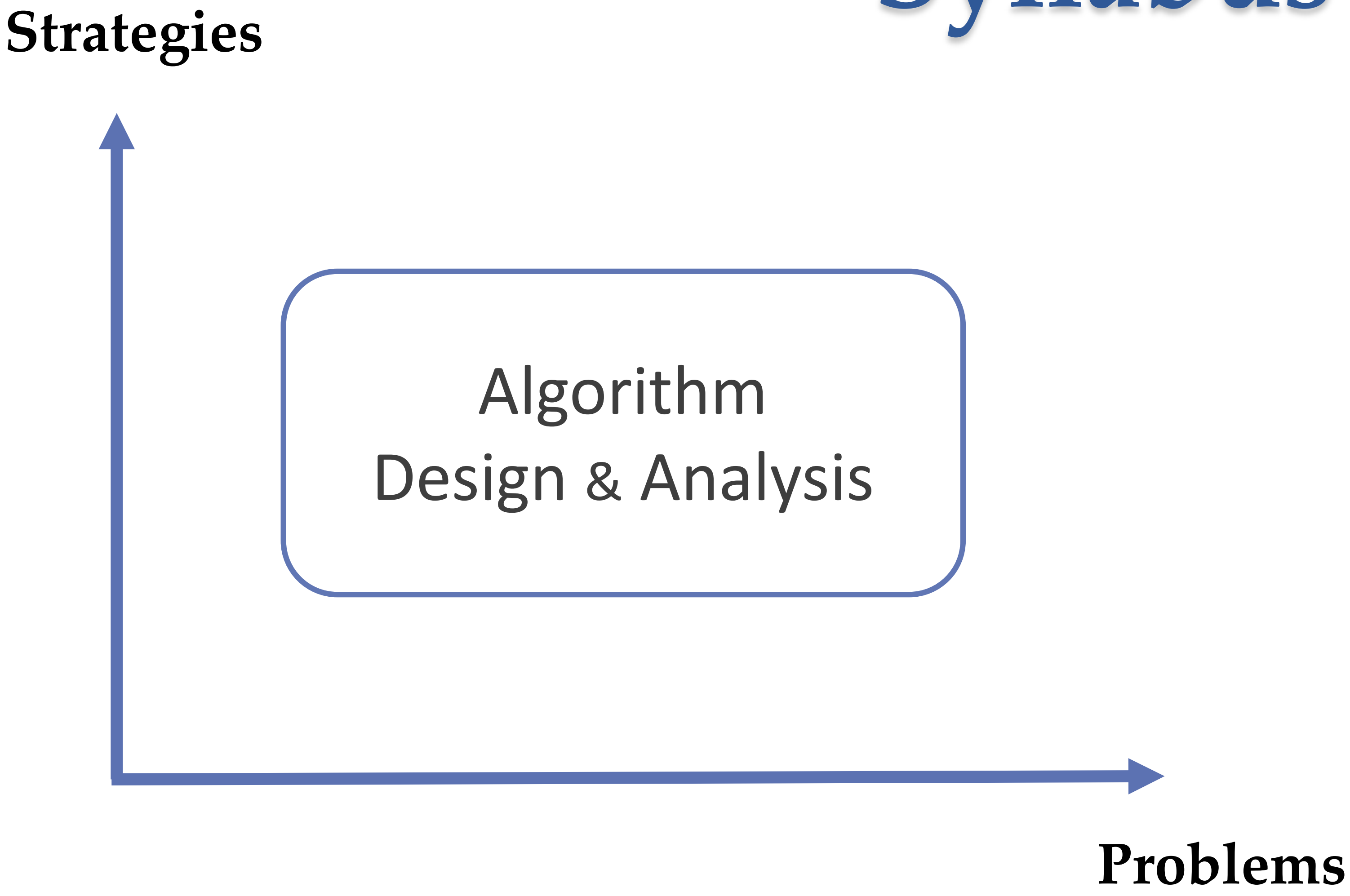
NPC

Tutorial 7

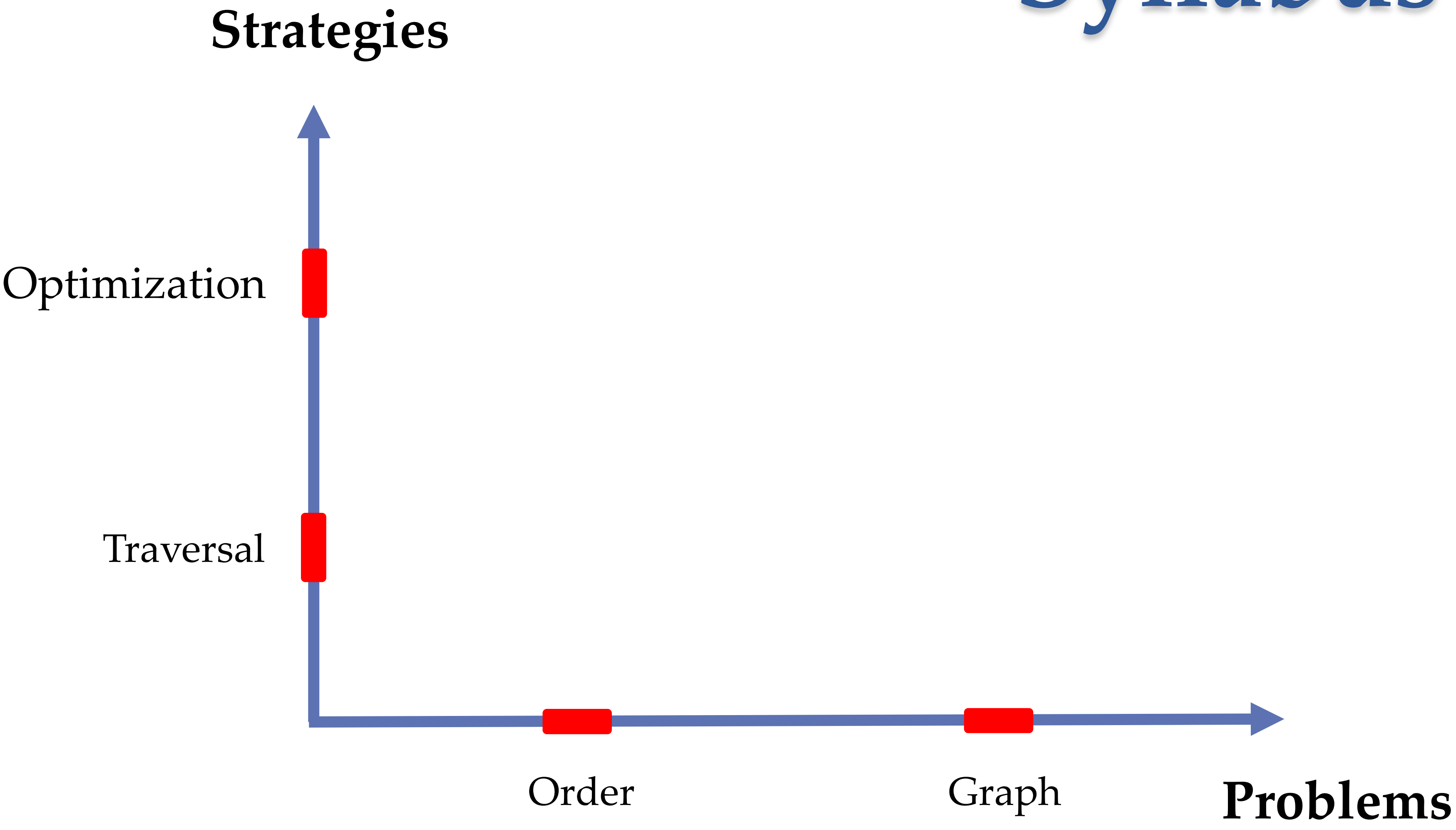
Solutions to problems

Final exam

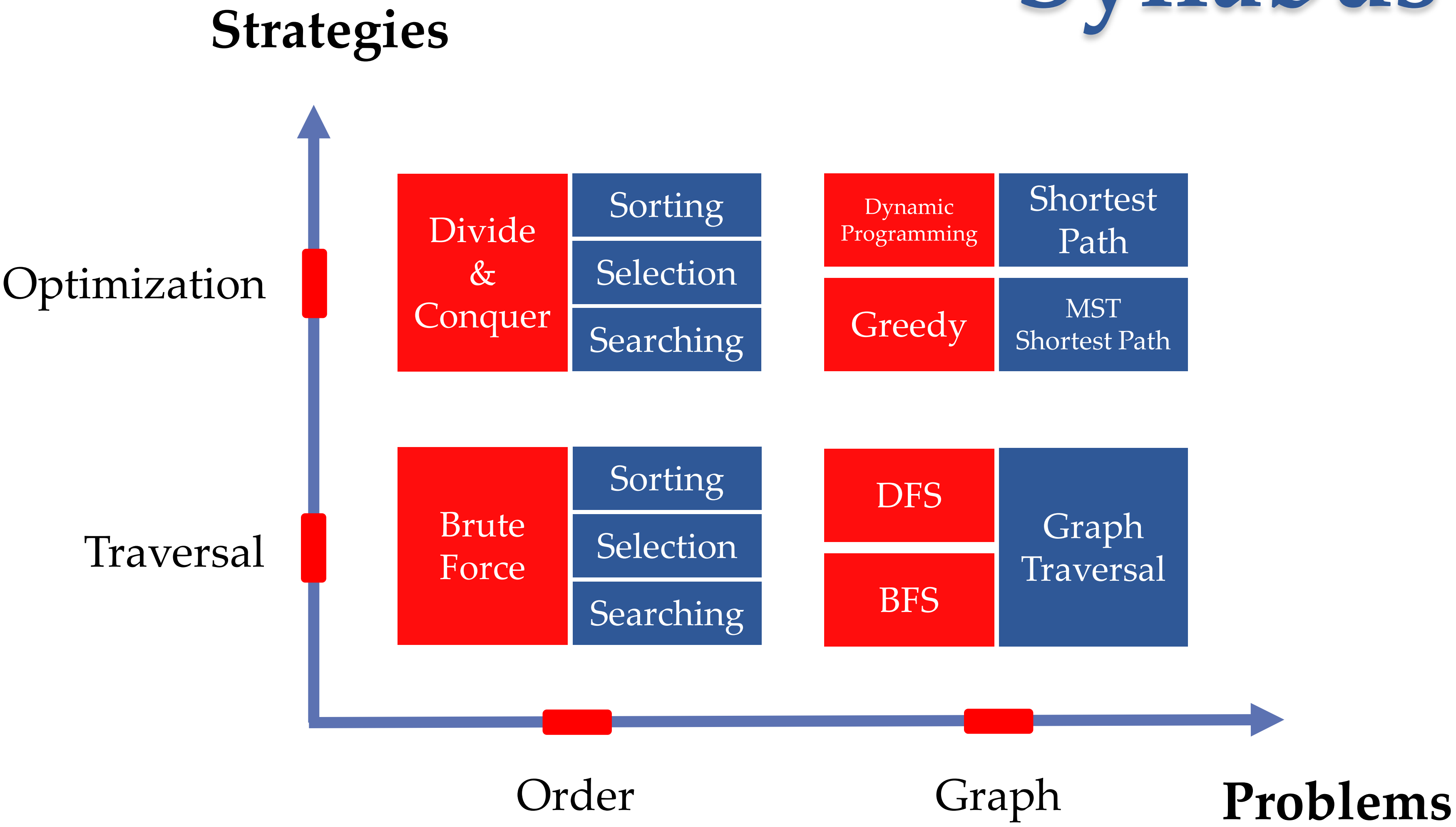
Syllabus



Syllabus



Syllabus



Strategies

Optimization

Divide & Conquer

Sorting

Selection

Searching

Dynamic Programming

Shortest Path

Greedy

MST
Shortest Path

Traversal

Brute Force

Sorting

Selection

Searching

DFS

BFS

Graph Traversal

Order

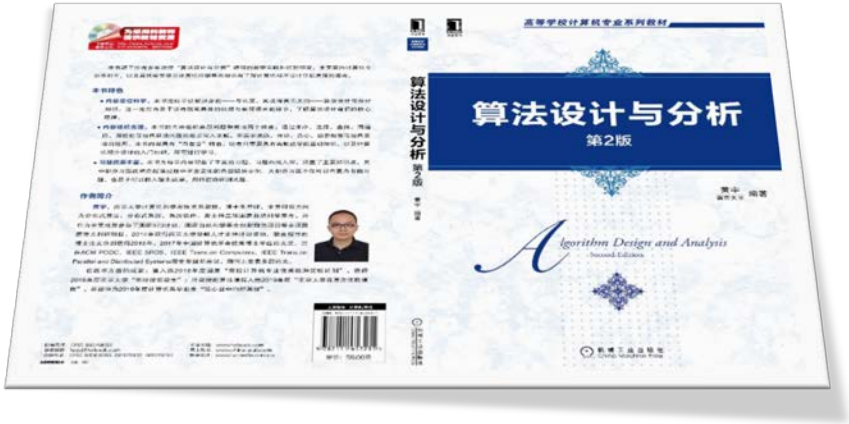
Graph

Problems

Textbooks

本页与视频略有不同，
以此pdf为准

- Course outline: LADA
  - Lectures on Algorithm Design & Analysis (slides)
- Course contents
  - Algorithm Design and Analysis

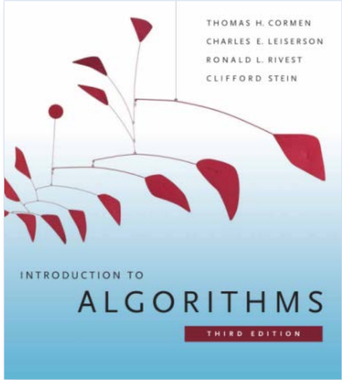


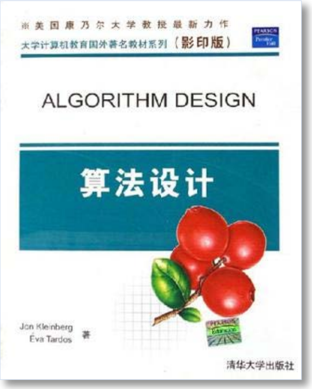
More info about the book:
https://zhuanlan.zhihu.com/p/24150569

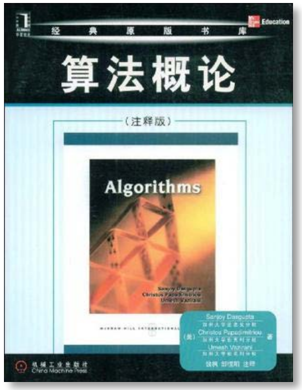
Textbooks

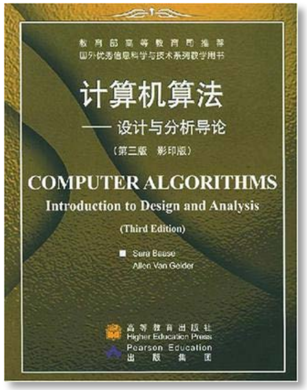
- **Further reading**
  - o Introduction to Algorithms
  - o Algorithm Design
  - o Algorithms
  - o Computer Algorithms*

See the "douban list" for more info:
http://book.douban.com/doulist/1155824/









Algorithm – Design & Analysis

- **Algorithm - the spirit of computing**
  - Model of computation
- **Algorithm by example**
  - Greatest common divisor
  - Sequential search
- **Algorithm design & analysis**
  - Correctness
  - Worst-case / average-case cost analysis

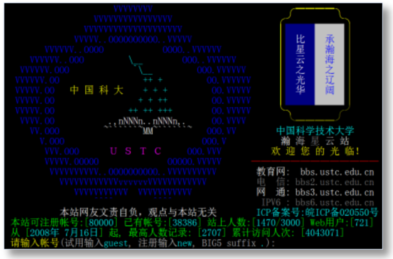
Computer and Computing

- **Problem 1**
  - o Why the computer seems to be able to do anything?
    - Scientific computing, document processing, computer games, ebooks, movies, computer games, …

IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. XX, NO. XX, XXX XXXX 1

Runtime Detection of the Concurrency Property in Asynchronous Pervasive Computing Environments

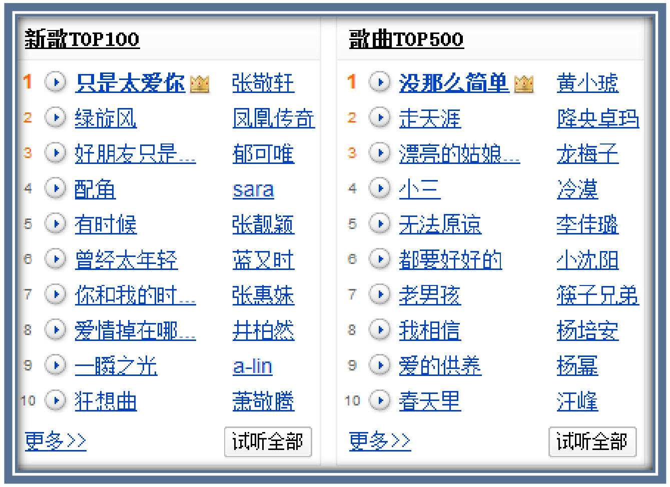
Yu Huang, Yiling Yang, Jiannong Cao, Xiaoxing Ma, Xianping Tao, and Jian Lu



```
中国科学技术大学
瀚 海 星 云 站
欢 迎 您 的 光 临!
教育网:  bbs.ustc.edu.cn
电  信: bbs2.ustc.edu.cn
网  通: bbs3.ustc.edu.cn
 IPV6 : bbs6.ustc.edu.cn
本站网友文责自负, 观点与本站无关        ICP备案号:皖ICP备020550号
本站可注册帐号:[80000] 已有帐号:[38386] 站上人数:[1470/3000] Web用户:[721]
从 [2008年 7月16日] 起, 最高人数记录: [2707] 累计访问人次: [4043071]
请输入帐号(试用输入guest, 注册输入new, BIG5 suffix .):
```

Computer and Computing

- **Problem 2**
  - What can / cannot be efficiently done by a computer?
    - Manage millions of songs vs. music composition

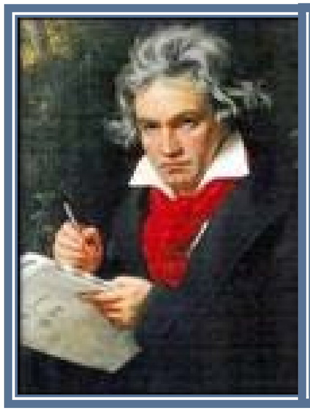


新歌TOP100		
1	只是太爱你	张敬轩
2	绿旋风	凤凰传奇
3	好朋友只是...	郁可唯
4	配角	sara
5	有时候	张靓颖
6	曾经太年轻	蓝又时
7	你和我的时...	张惠妹
8	爱情掉在哪...	井柏然
9	一瞬之光	a-lin
10	狂想曲	萧敬腾

更多>> 试听全部

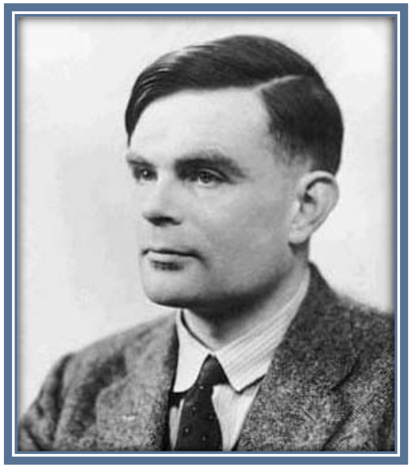
歌曲TOP500		
1	没那么简单	黄小琥
2	走天涯	降央卓玛
3	漂亮的姑娘...	龙梅子
4	小三	冷漠
5	无法原谅	李佳璐
6	都要好好的	小沈阳
7	老男孩	筷子兄弟
8	我相信	杨培安
9	爱的供养	杨幂
10	春天里	汪峰

更多>> 试听全部

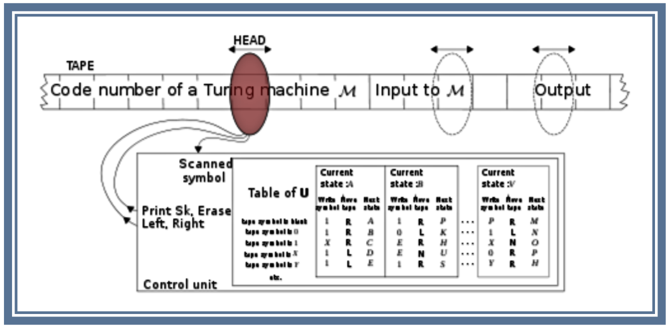


Computer and Computing

- **Computing**
  - o Encoding everything into `0's and `1's
  - o Operations over '1's and '0's
  - o Decoding the '1's and '0's

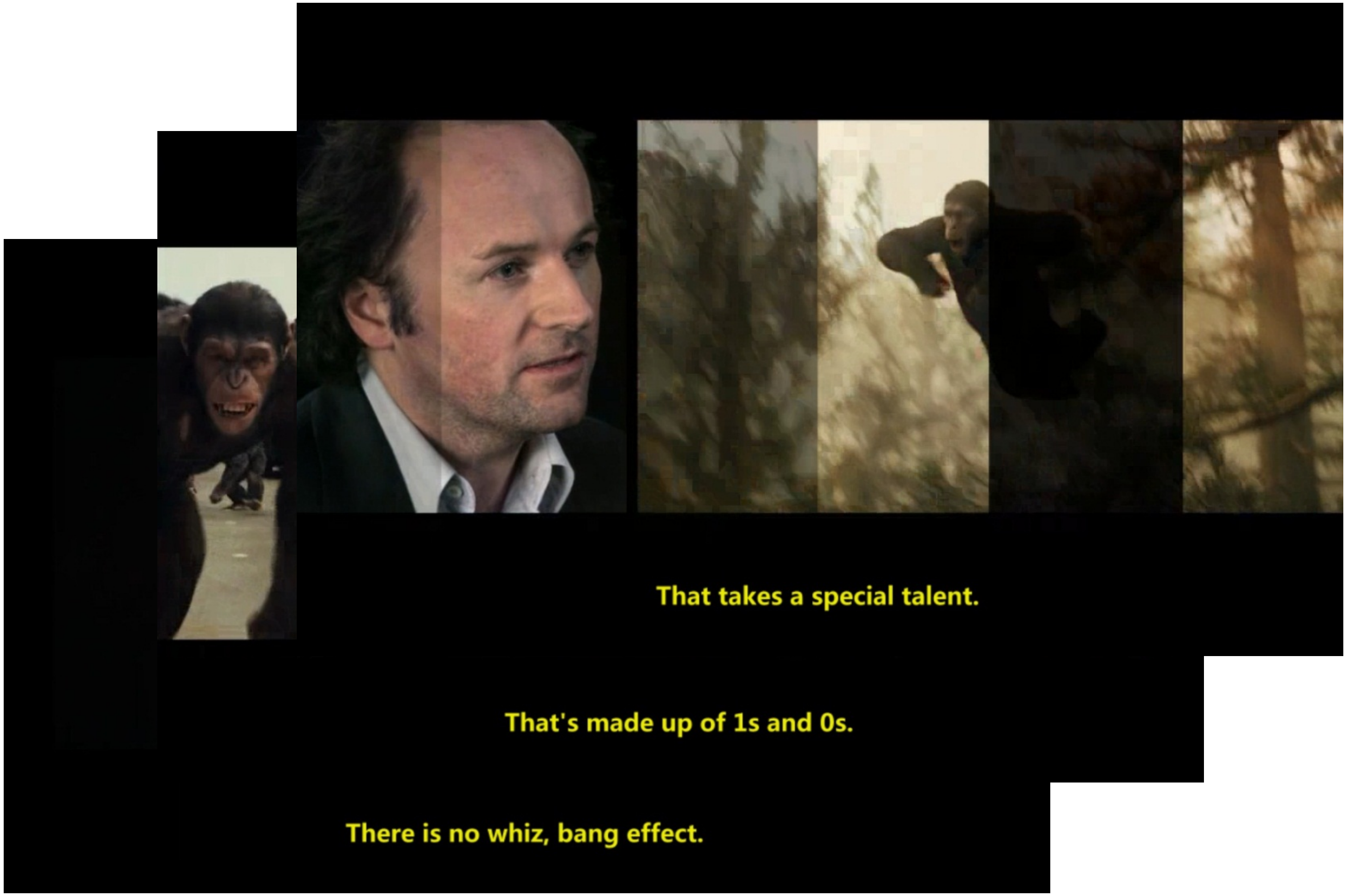


- **Turing machine**
  - o An abstract/logical computer



	Current state :A			Current state :B				Current state :N		
	Write symbol	Move tape	Next state	Write symbol	Move tape	Next state		Write symbol	Move tape	Next state
tape symbol is blank	1	R	*A*	1	R	*P*	...	*P*	R	*M*
tape symbol is 0	1	R	*B*	0	L	*K*	...	1	L	*N*
tape symbol is 1	*X*	R	*C*	*E*	R	*H*	...	*X*	N	*O*
tape symbol is *X*	1	L	*D*	*E*	N	*U*	...	0	R	*P*
tape symbol is *Y*	1	L	*E*	1	R	*S*	...	*Y*	R	*H*
etc.										

Computing in Everyday Life

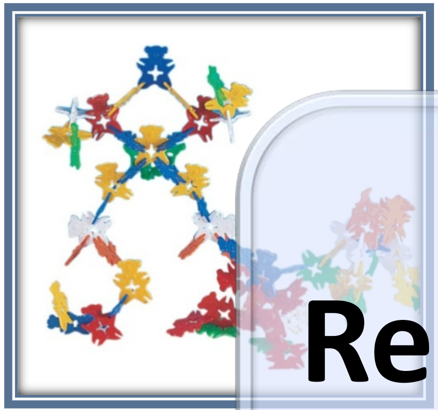


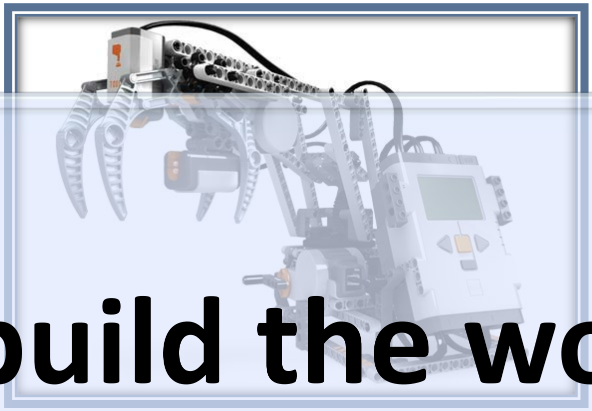
That takes a special talent.

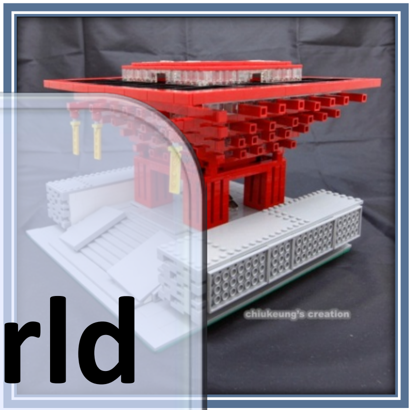
That's made up of 1s and 0s.

There is no whiz, bang effect.

Algorithm



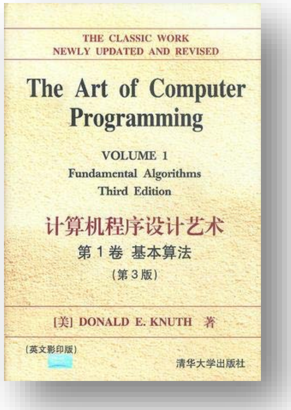




Rebuild the world with 0s and 1s

Algorithm

- **Algorithm is the spirit of computing**
  - To solve a specific problem (so called an *algorithmic problem*)
  - Combination of basic operations
    - in a precise and elegant way
- **Essential issues**
  - Model of computation
  - Algorithm design
  - Algorithm analysis



THE CLASSIC WORK
NEWLY UPDATED AND REVISED

The Art of Computer
Programming

VOLUME 1
Fundamental Algorithms
Third Edition

计算机程序设计艺术
第 1 卷 基本算法
（第 3 版）

[美] DONALD E. KNUTH 著

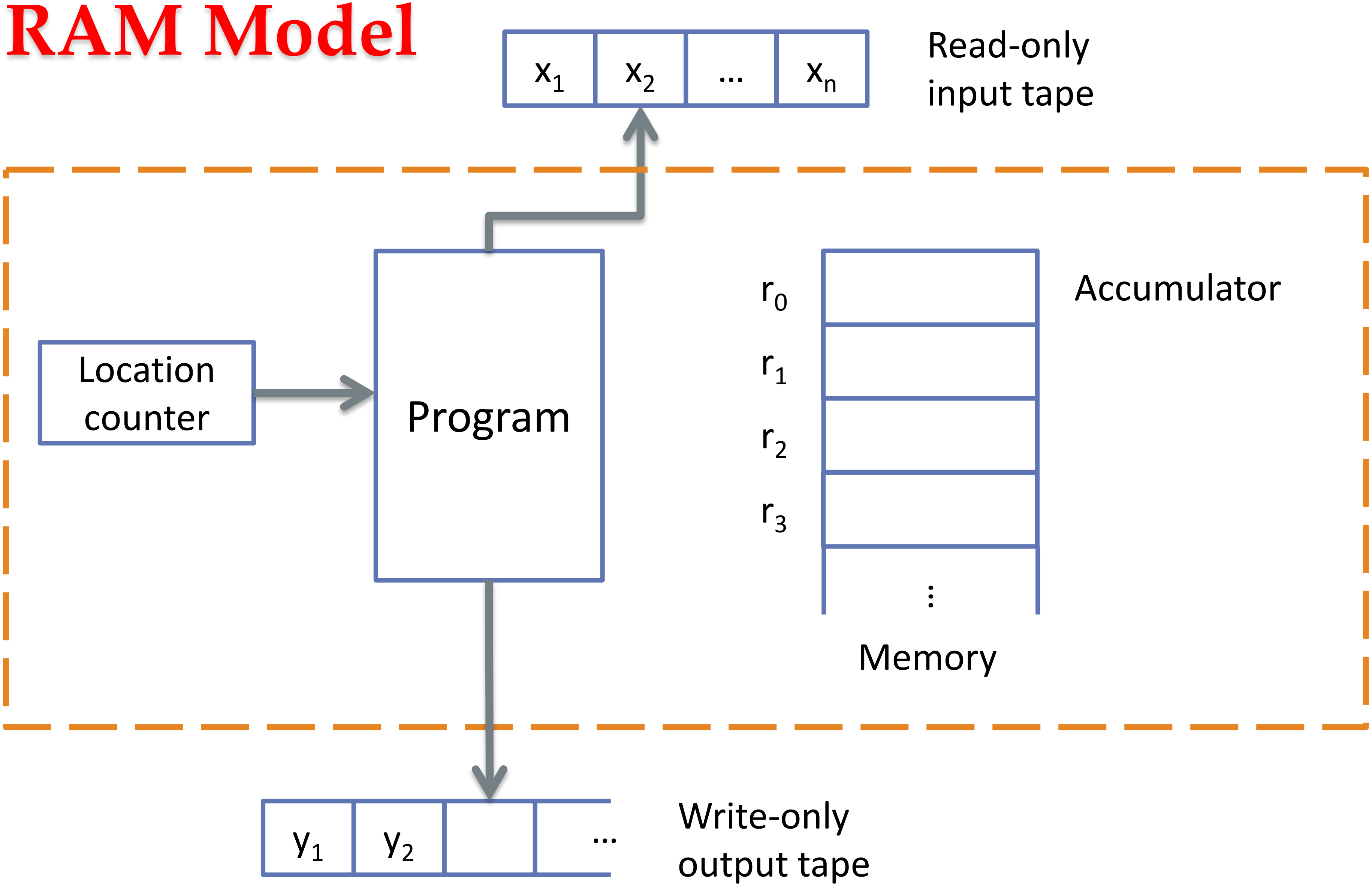
（英文影印版）

清华大学出版社

Model of Computation

- **Problems**
  - Why the algorithms we learn can run almost everywhere?
  - Why the algorithms we learn can be implemented in any language?
- **Machine- and language- independent algorithms, running on an abstract machine**
  - Turing machine: over-qualify
  - RAM model: simple but powerful

RAM Model


x_1
x_2
…
x_n
Read-only input tape
Location counter
Program
r_0
r_1
r_2
r_3
⋮
Accumulator
Memory
y_1
y_2
…
Write-only output tape

The RAM Model of Computation

- **Each *simple operation* takes one time step**
  - E.g., key comparison, +/-, memory access, …
- **Non-simple operations should be decomposed**
  - Loop
  - Subroutine
- **Memory**
  - Memory access is a simple operation
  - Unlimited memory

Tradeoff:
accuracy
vs.
ease of use

Further Reading

“哼，你让他们成楔形攻击队形不就行了？”秦始皇轻蔑地看着冯•诺伊曼。牛顿不知从什么地方掏出六面小旗．三白三黑，冯•诺伊曼接过来分给三名士兵，每人一白一黑，说：“白色代表0，黑色代表1。好，现在听我说，出，你转身看着入1和入2，如果他们都举黑旗，你就举黑旗，其他的情况你都举白旗，这种情况有三种：入l白，入2黑；入l黑，入2白；入1、入2都是白。”

“不需要，我们组建一千万个这样的门部件，再将这些部件组合成一个系统，这个系统就能进行我们所需要的运算，解出那些预测太阳运行的微分方程。这个系统，我们把它叫做……嗯，叫做……”

“计算机。”汪淼说。

“啊——好！”冯•诺伊曼对汪淼竖起一根指头，“计算机，这个名字好，整个系统实际上就是一部庞大的机器，是有史以来最复杂的机器！”

刘慈欣，《三体、牛顿、冯•诺依曼、秦始皇、三日连珠》，《三体》第一部

To Create an Algorithm

- **Algorithm design**
  - Composition of simple operations, to solve an algorithmic problem
- **Algorithm analysis**
  - Amount of work done / memory used
    - In the worst/average case
  - Advanced issues
    - Optimality, approximation ratio, …

Algorithm by Example

- **Algorithmic Problem 1**
  - Find the greatest common divisor of two non-negative integers *m* and *n*

- **Algorithmic Problem 2**
  - Is a specific key *K* stored in array E[1..n]?

Probably the Oldest Algorithm

- **Euclid Algorithm**

Problem

- Find the greatest common divisor of two non-negative integers *m* and *n*

Specification

Input: non-negative integer m, n
Output: gcd(m, n)

Euclid algorithm

```
[E1] n divides m, the remainder -> r
[E2] if r = 0 then return n
[E3] n -> m; r-> n; goto E1
```

Euclid algorithm – recursive version

```
Euclid(m,n)
[E1] if n=0 then return m
[E2] else return Euclid(n, m mod n)
```

Sequential Search

Problem

- Search an array for a specific key

Specification

Input: K, E[1..n]

Output: Location of K (1,2,…,n; -1: K is not in E[])

Sequential searchEuclid algorithm

```
Int seqSearch(int[] E, int n, int K)
    int ans, index;
    ans=-1;
    for (index=1; index<=n; index++)
        if (K==E[index])
            ans=index;
            break;
    Return ans;
```

Algorithm Design

- **Criteria**
  - Defining correctness
- **Main challenge**
  - For proving correctness
- **Our strategy**
  - Mathematical induction
  - …

Specification

Input: non-negative integer m, n
Output: gcd(m, n)

Main challenge

- The output is **always** correct, for **any** legal input.
- Infinite possible inputs

Mathematical induction

- Weak principle
- Strong principle

For Your Reference

- Mathematical induction

The Weak Principle of Mathematical Induction

- If the statement p(b) is true and the statement p(n-1) => p(n) is true for all n>b, then p(n) is true for all integers n>=b.

The Strong Principle of Mathematical Induction

- If the statement p(b) is true, and the statement {p(b) and p(b+1) and … and p(n-1) => p(n)} is true, for all n>b, then p(n) is true for all integers n>=b.

Correctness of the Euclid Algorithm

- **Induction on n**
  - Base case
    - n = 0: for any m, Euclid(m, 0) = m;
    - n = 1: for any m, Euclid(m, 1) = 1;
    - n = 2: …
  - Assumption
    - For any n≤N_0, Euclid(m, n) is correct;
  - Induction
    - Euclid(m, N_0+1) = Euclid(N_0+1, m mod (N_0+1));

gcd(m, N_0+1) = gcd(N_0+1, m mod (N_0+1))

Notes on Induction

“Notes on Structured Programming”, E.W. Dijkstra

> I have mentioned **mathematical induction** explicitly, because it is the only pattern of reasoning that I am aware of, that eventually enables us to cope with loops and recursive procedures.

Algorithm Analysis

- **Criteria**
  - Performance metrics
- **Worst case**
  - Best case?
- **Average case**
  - Average cost?
- **Advanced topics**
  - Lower bound, optimality, …

Algorithm Analysis

- **How to measure**
  - Not too general
    - Giving essential indication in comparison of algorithms
  - Not too precise
    - Machine independent
    - Language independent
    - Programming paradigm independent
    - Implementation independent

Algorithm Analysis

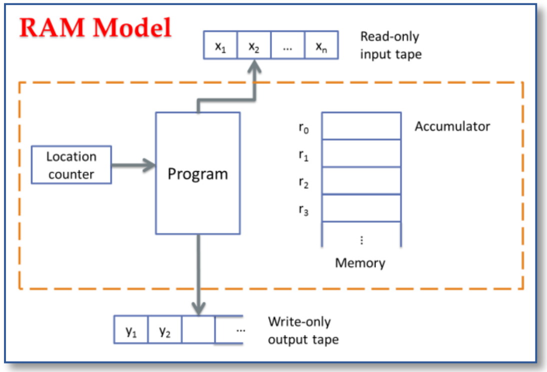
After two years of writing my book, I realized that its novelty was quantitatively determining how good a program is. I didn't just want to say one program was better than another. I wanted to say one is 13.8% better than another, and explain how to compare them.

Author A would talk about algorithm A, and author B would talk about his competing algorithm B. And author A never wrote about algorithm B, and author B never wrote about algorithm A. Also, authors A and B used different computers. **As a neutral journalist**, I explained both from one point of view. Asking, "How good is an algorithm really?" is a fun problem. That's the analysis of algorithms.

D.E. Knuth
"The Computer Scientist Who Can't Stop Telling Stories"
Quanta Magazine

Algorithm Analysis

- **Criteria**
  - Critical operation
  - How many critical operation are conducted

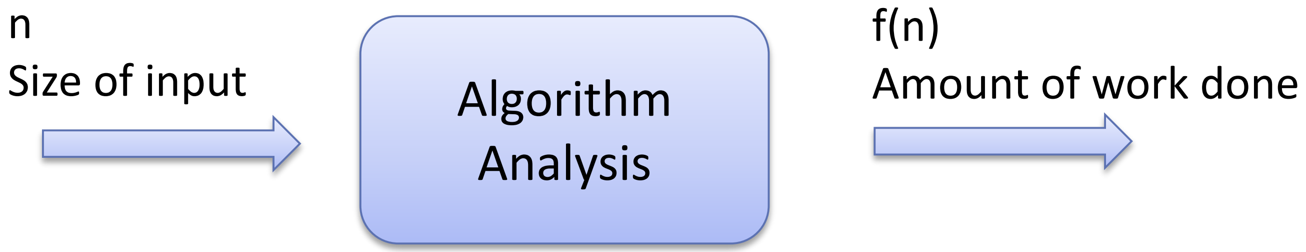


- **For example**

Algorithmic problem	Critical operation
Sorting, selection, searching String matching	Comparison (of keys)
Graph traversal	Processing a node/edge
Matrix multiplication	Multiplication

Algorithm Analysis

- **Amount of work done**
  - o usually depends on size of the input
  - o usually does not depend on size of the input only



n

Size of input

Algorithm Analysis

f(n)

Amount of work done

Worst-case Complexity

- **W(n)**
  - o Upper bound of cost
    - For any possible input
  - o $W(n) = \max_{I \in D_n} f(I)$

Average-case Complexity

- **A(n)**
  - o Weighted average
  - o $A(n) = \sum_{I \in D(n)} \Pr(I)\, f(I)$
- **A special case**
  - o Average cost
    - Total cost of all inputs, averaged over the input size
  - o $Average(n) = \frac{1}{|D(n)|} \sum_{I \in D(n)} f(I)$

Average-case Cost of *SeqSearch*

- **Case 1: K is in E[]**
  - Assumptions:
    1. Assuming that K is in E[]
    2. Assuming no same entries in E[]
    3. Each possible input appears with equality (thus, K in the i^{th} location with probability $\frac{1}{n}$)

  - $A_{succ}(n) = \sum_{i=0}^{n-1} \Pr(I_i|succ)\, t(I_i)$

    $= \sum_{i=0}^{n-1} \frac{1}{n}(i+1)$

    $= \frac{n+1}{2}$

Average-case Cost of *SeqSearch*

- **Case 2: K may (or may not) be in E[]**
  - Assume that K is in E[] with probability q
  - $A(n) = \Pr(succ)\, A_{succ}(n) + \Pr(fail)\, A_{fail}(n)$

$$= q\frac{n+1}{2} + (1-q)n$$

How to make reasonable assumptions?

Algorithm Analysis

- **Advanced topics**
  - o Lower bound (Selection)
  - o Optimality (Greedy, DP)
  - o Computation complexity
  - o Approximate / online / randomized algorithms

Thank you!

Q & A

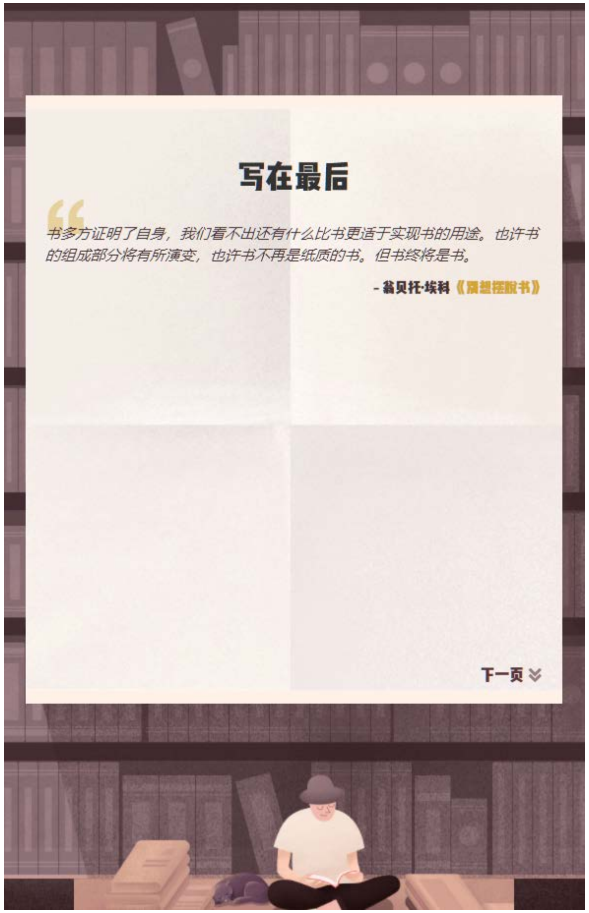
Yu Huang

http://cs.nju.edu.cn/yuhuang

Appendix

也许计算的设备会变得天翻地覆，但是算法终将是算法

– 黄宇 《算法设计与分析讲义》

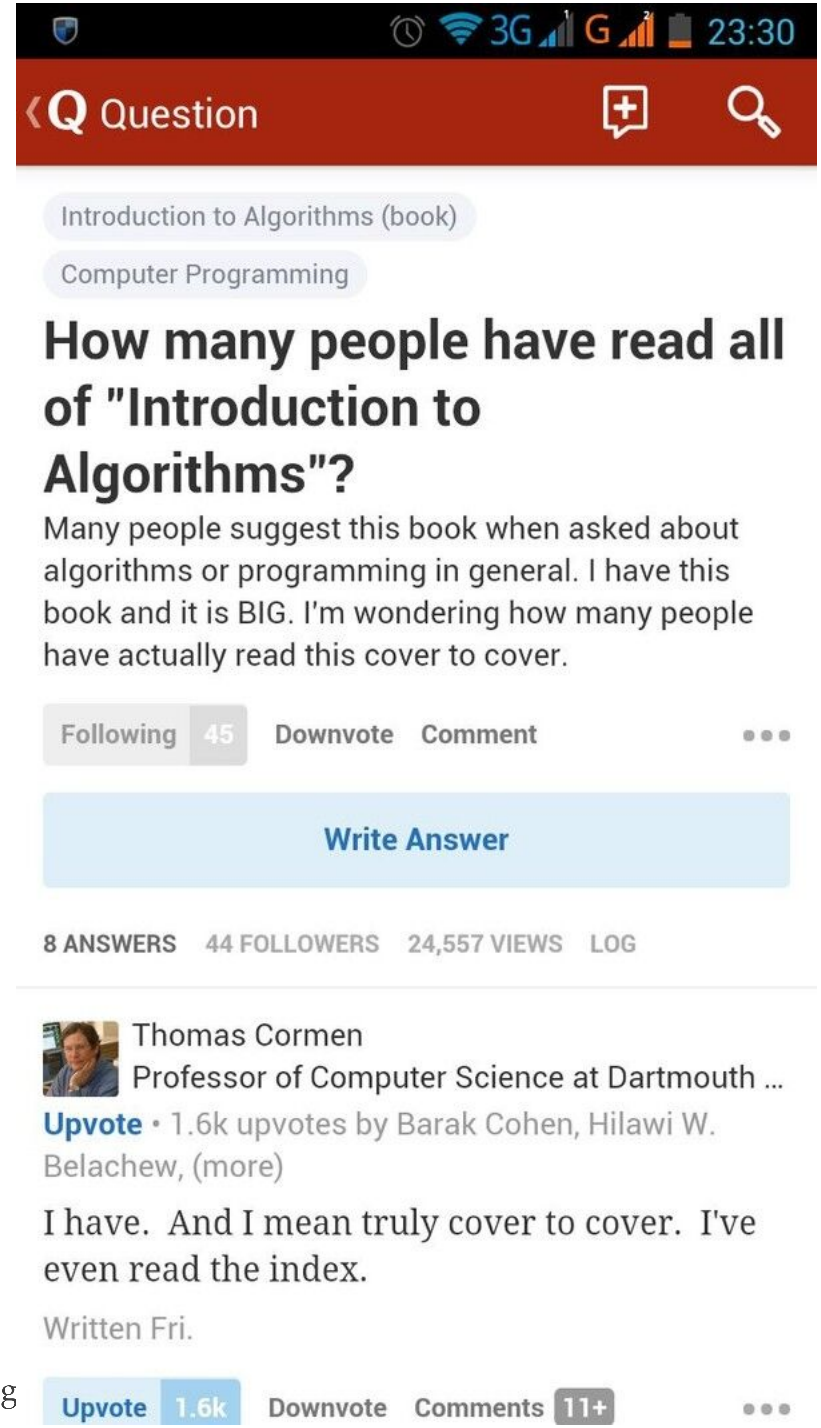


写在最后

“书多方证明了自身，我们看不出还有什么比书更适于实现书的用途。也许书的组成部分将有所演变，也许书不再是纸质的书。但书终将是书。

– 翁贝托·埃科《别想摆脱书》

Appendix



Introduction to Algorithms (book)

Computer Programming

How many people have read all of "Introduction to Algorithms"?

Many people suggest this book when asked about algorithms or programming in general. I have this book and it is BIG. I'm wondering how many people have actually read this cover to cover.

Thomas Cormen
Professor of Computer Science at Dartmouth ...

Upvote • 1.6k upvotes by Barak Cohen, Hilawi W. Belachew, (more)

I have. And I mean truly cover to cover. I've even read the index.

Written Fri.