Introduction to

# Algorithm Design and Analysis

[01] Model of Computation

Jingwei Xu
https://ics.nju.edu.cn/~xjw/
Institute of Computer Software
Nanjing University

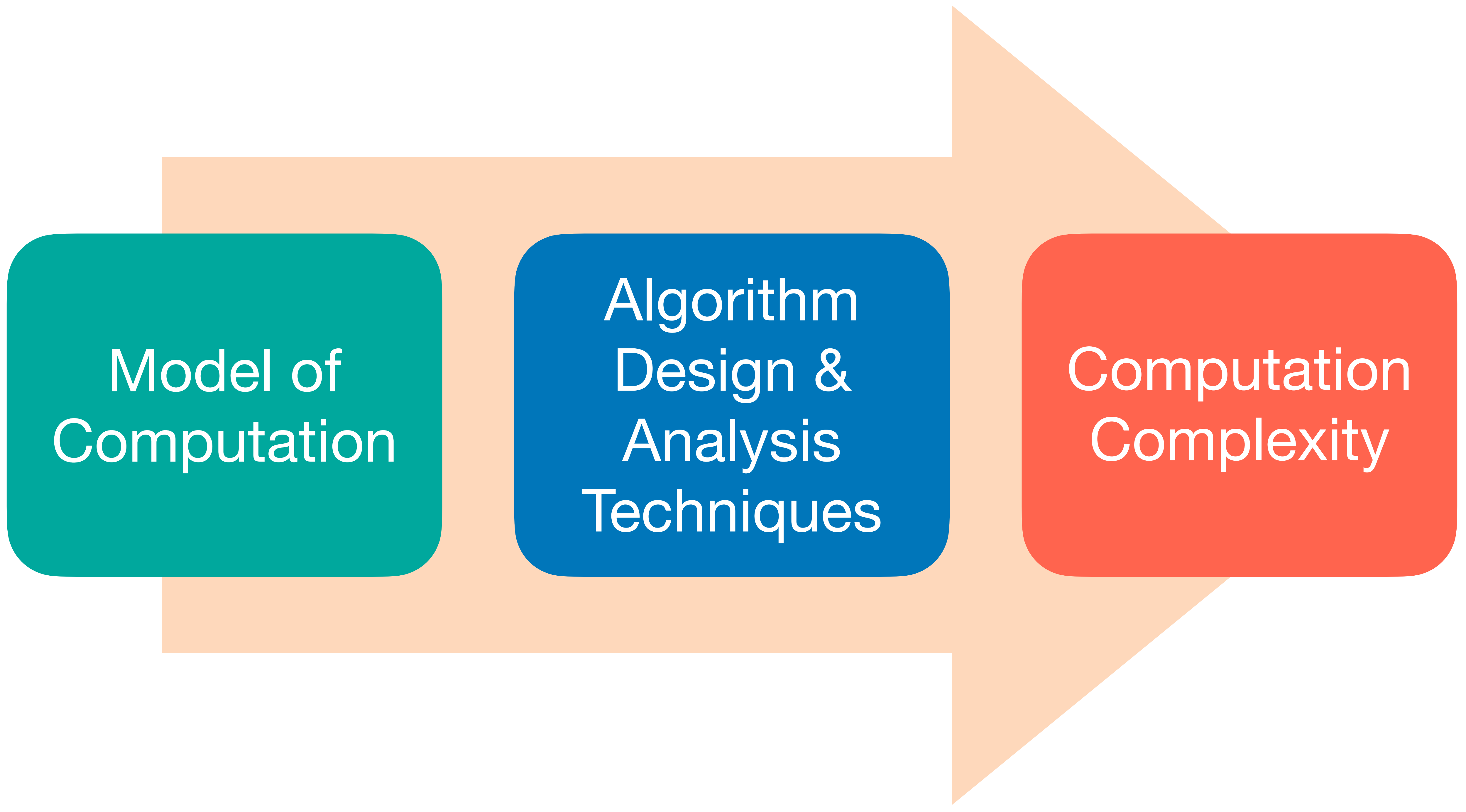
# Jingwei Xu（徐经纬）

- **Research: Intelligent Software**
  - Large Language Models
    - Algorithms & Infrastructure
- **Office:** 计算机系1022
- **Email:** jingweix@nju.edu.cn
- **homepage:** https://ics.nju.edu.cn/~xjw/

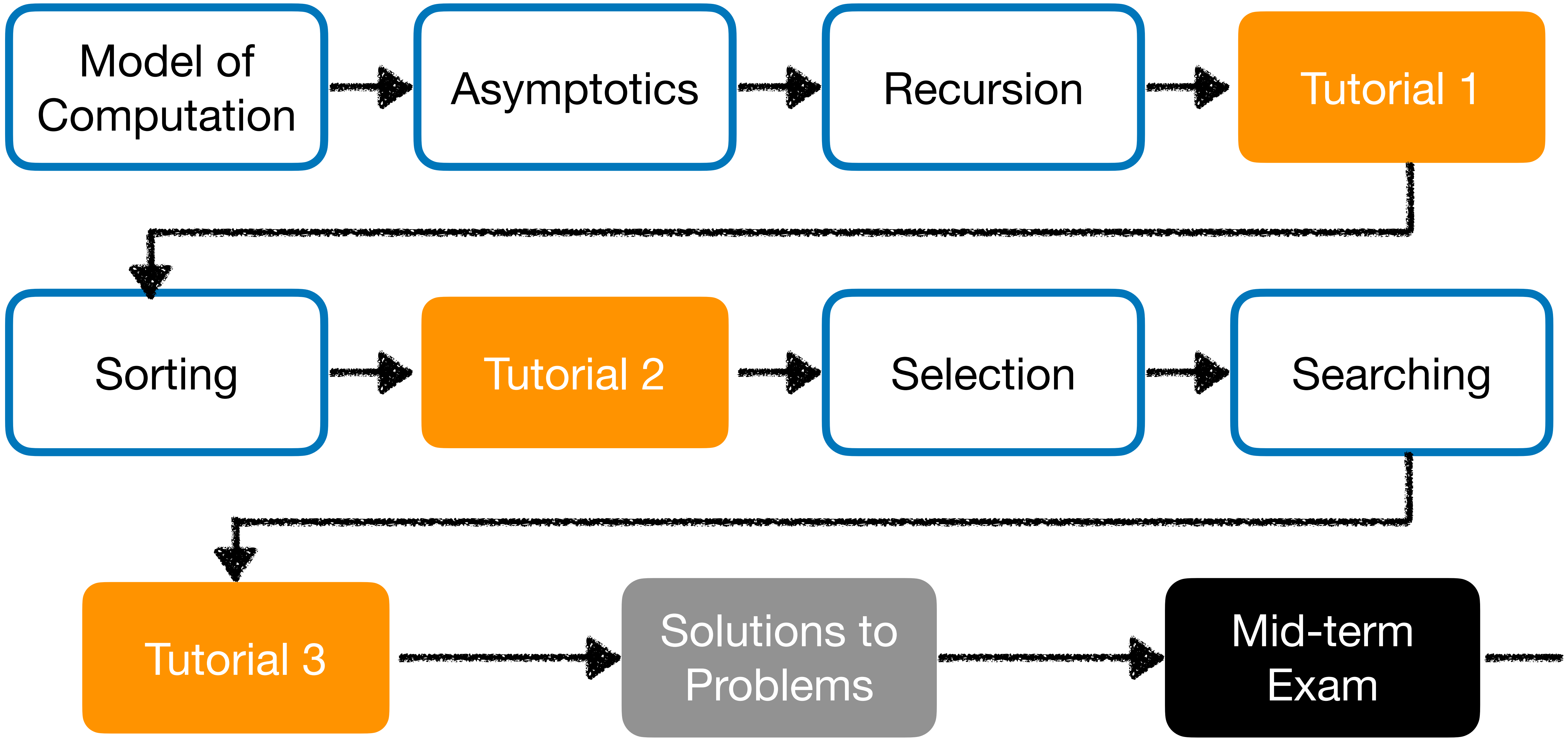
# Course Information

- Syllabus
- Textbook
- Website

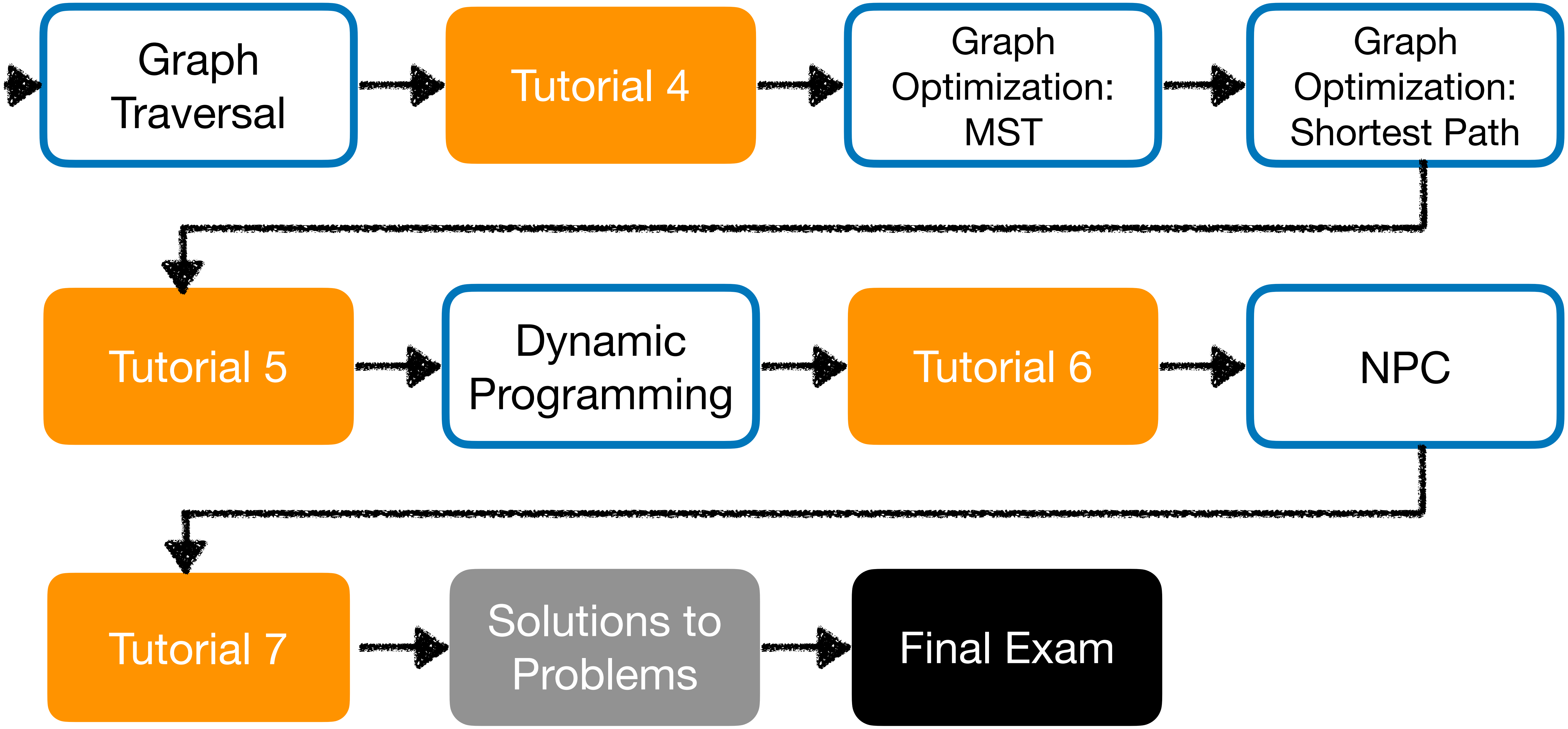
# Syllabus


Model of Computation
Algorithm Design & Analysis Techniques
Computation Complexity

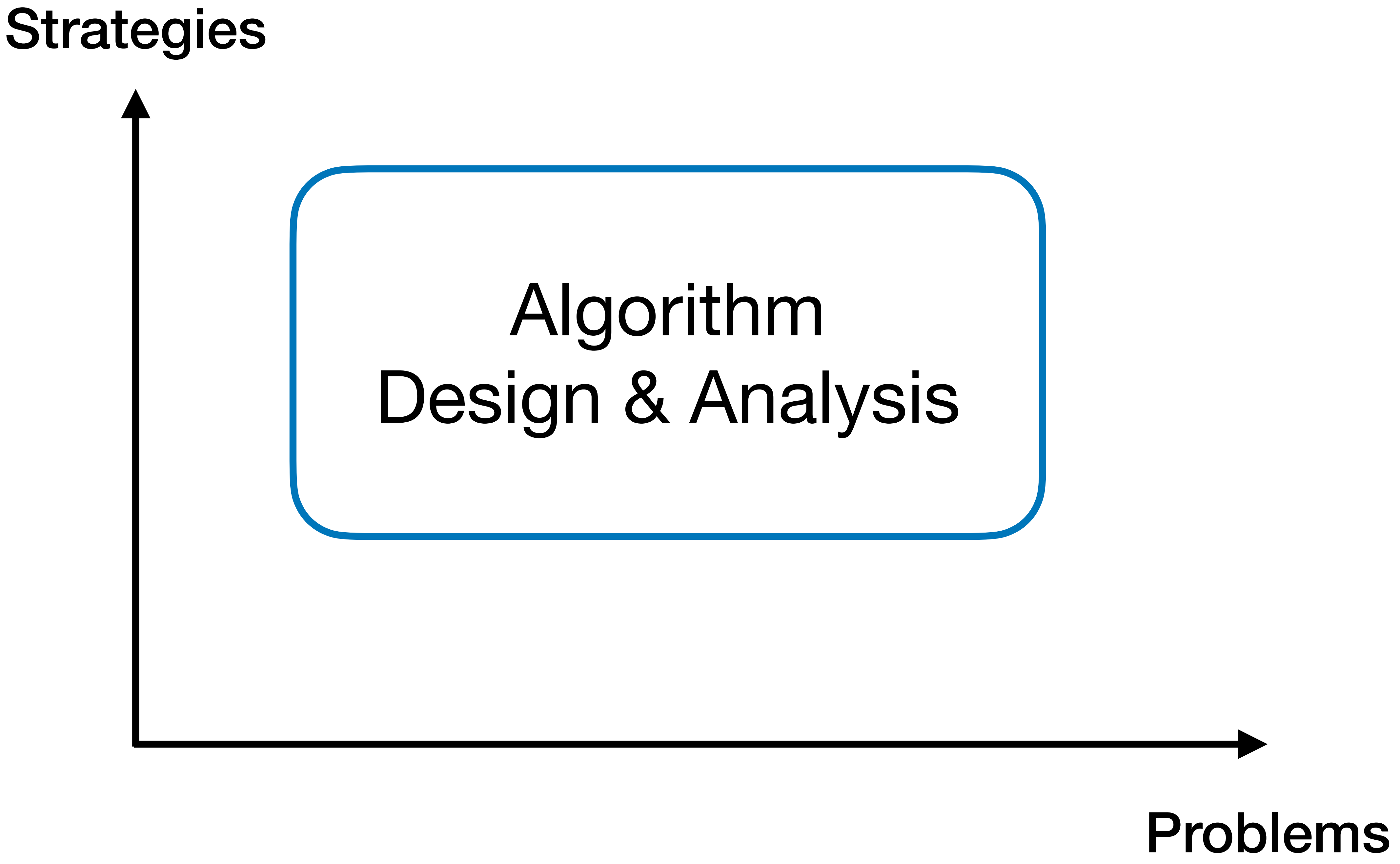
# Syllabus


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
NPC
Tutorial 7
Solutions to Problems
Final Exam

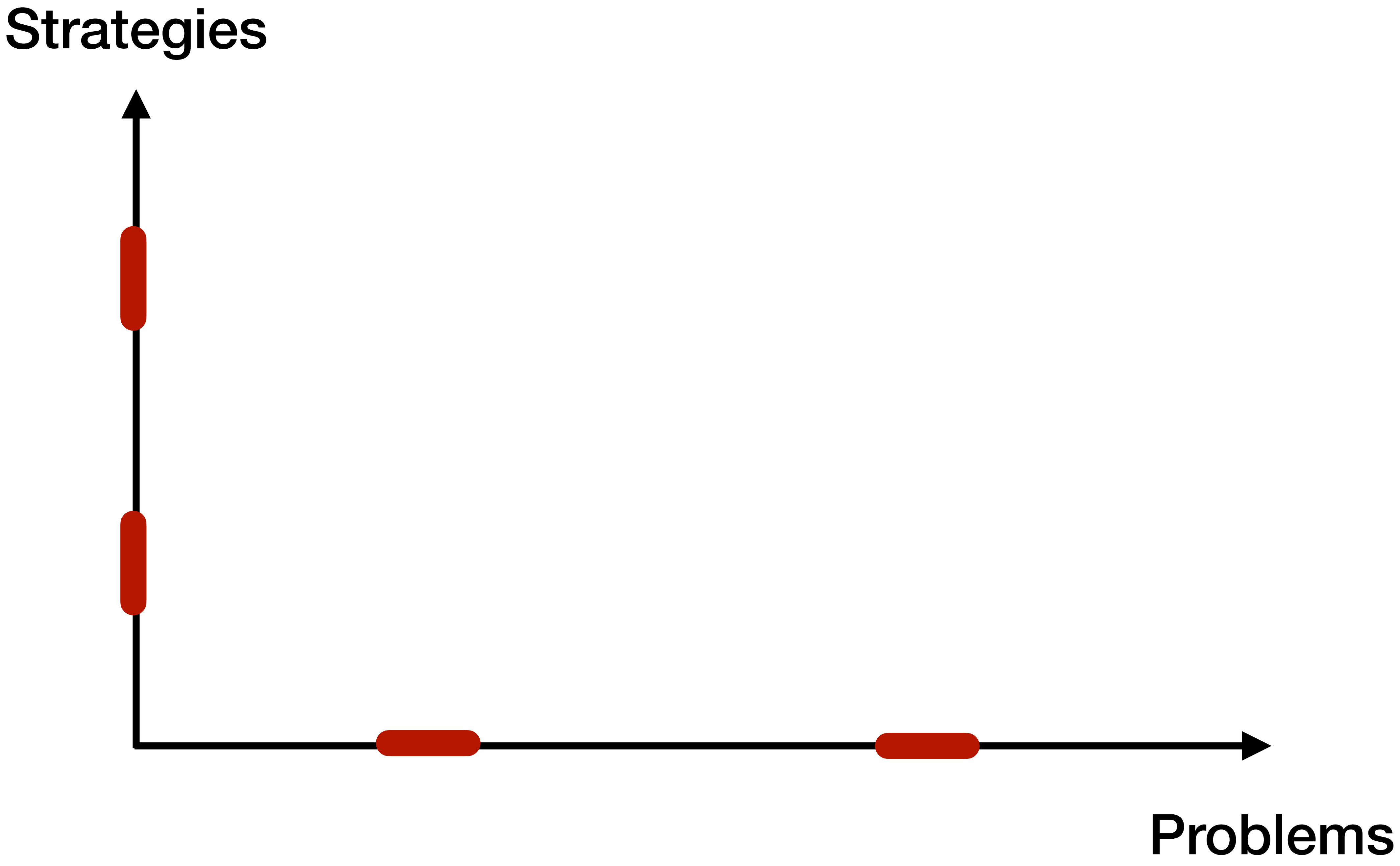
# Syllabus



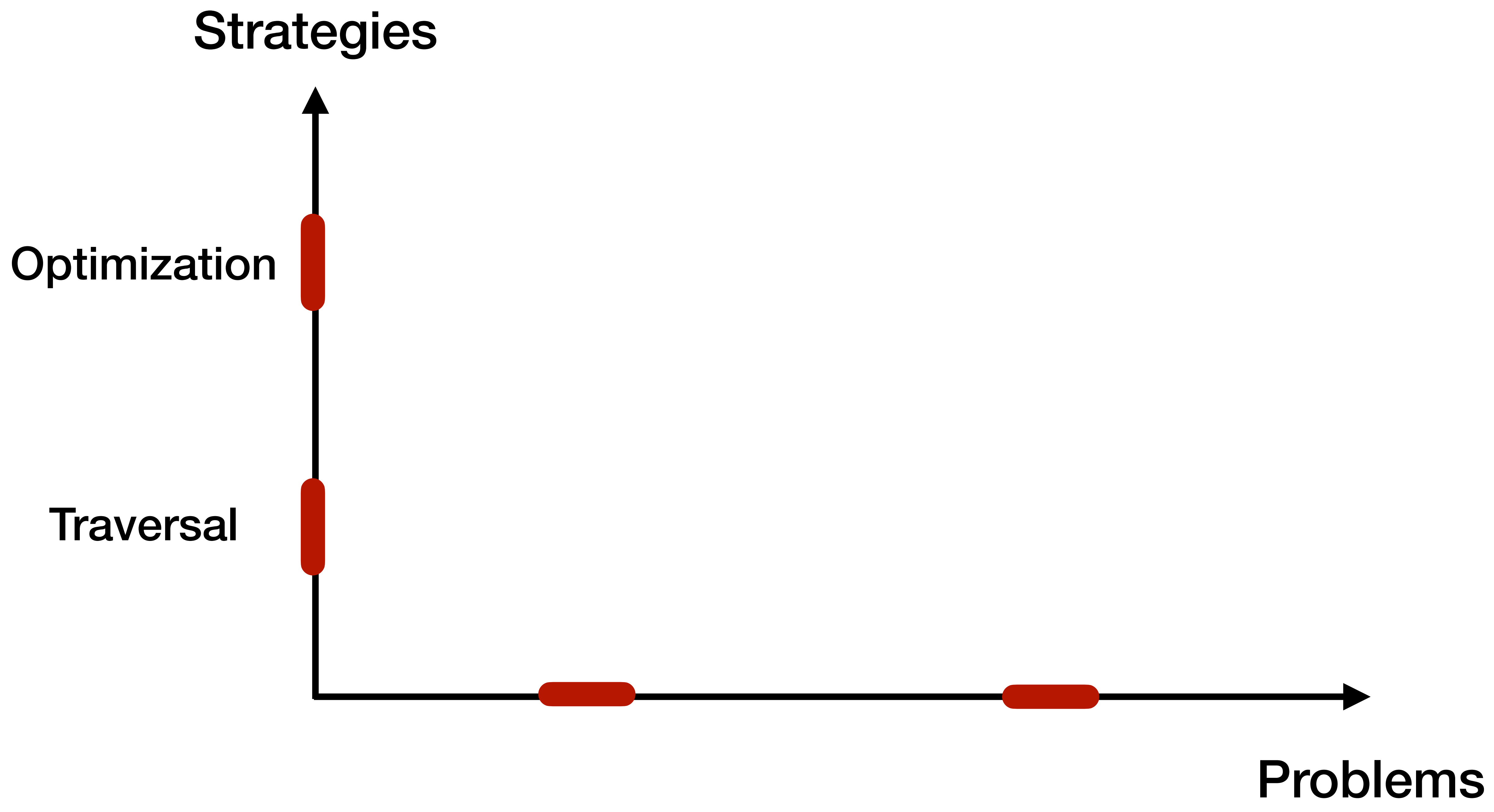
# Syllabus

Strategies
Problems

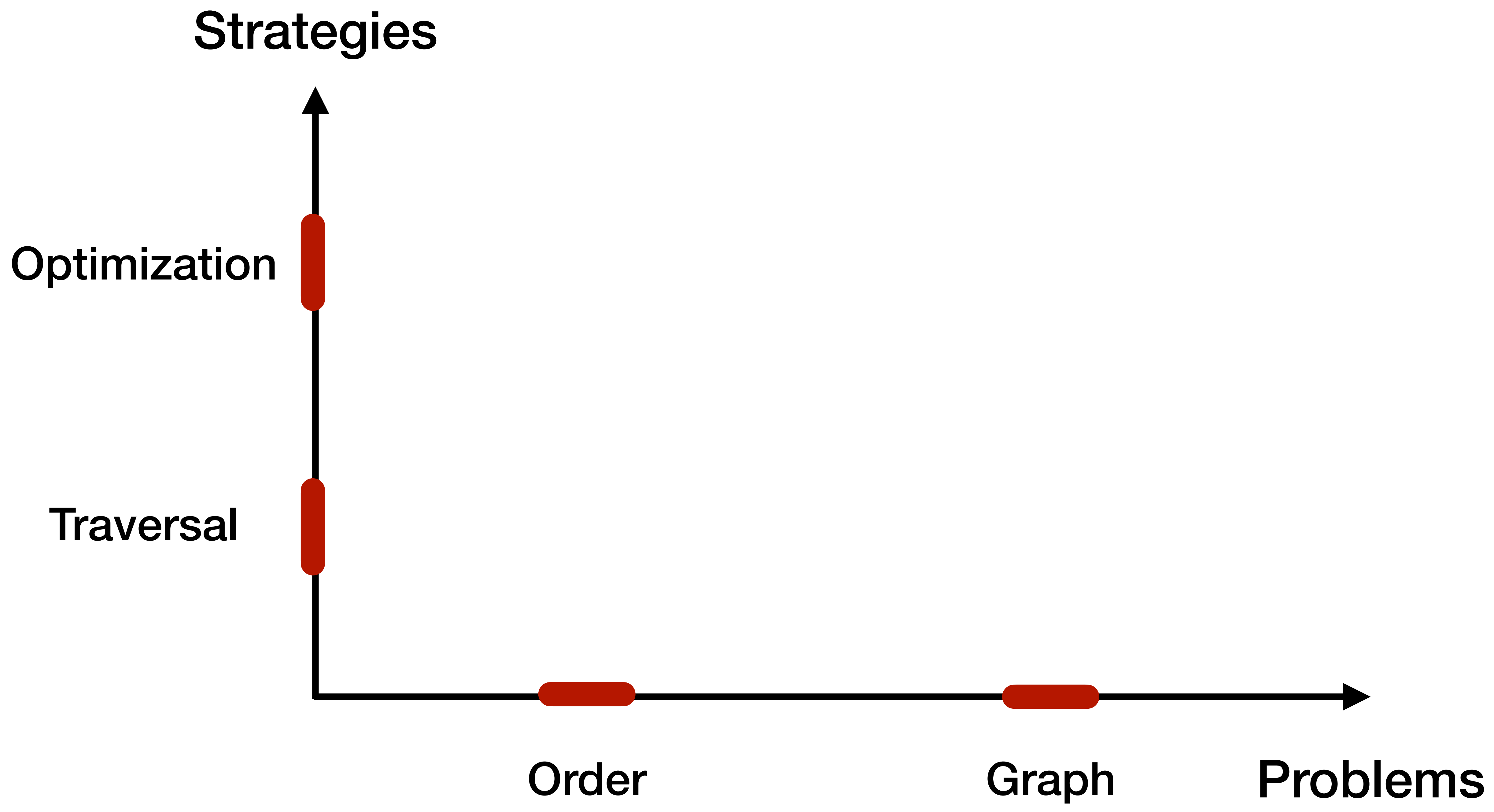
# Syllabus



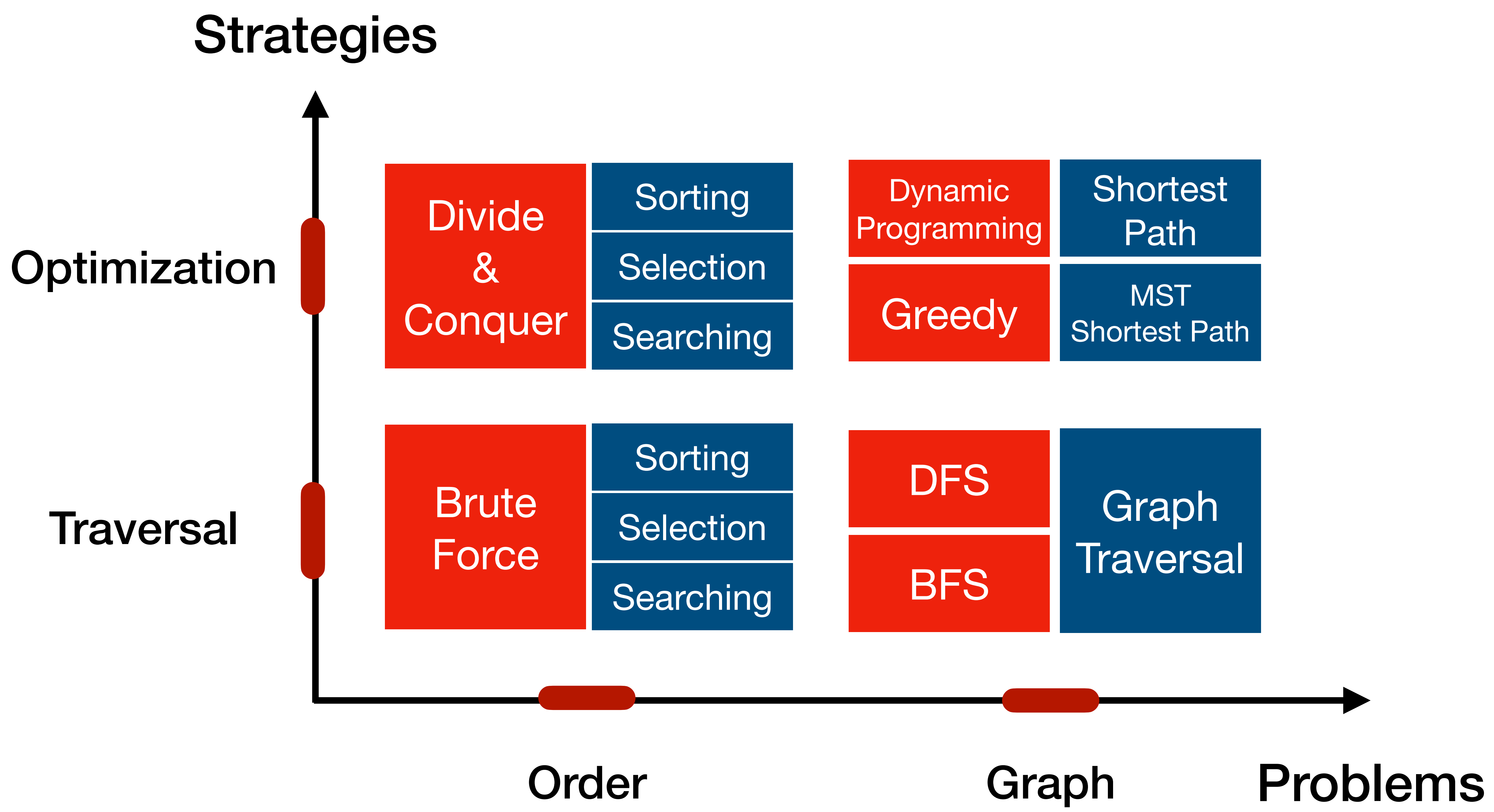
# Syllabus



# Syllabus



# Syllabus


Strategies
Optimization
Traversal
Divide & Conquer
Sorting
Selection
Searching
Dynamic Programming
Shortest Path
Greedy
MST
Shortest Path
Brute Force
Sorting
Selection
Searching
DFS
BFS
Graph Traversal
Order
Graph
Problems

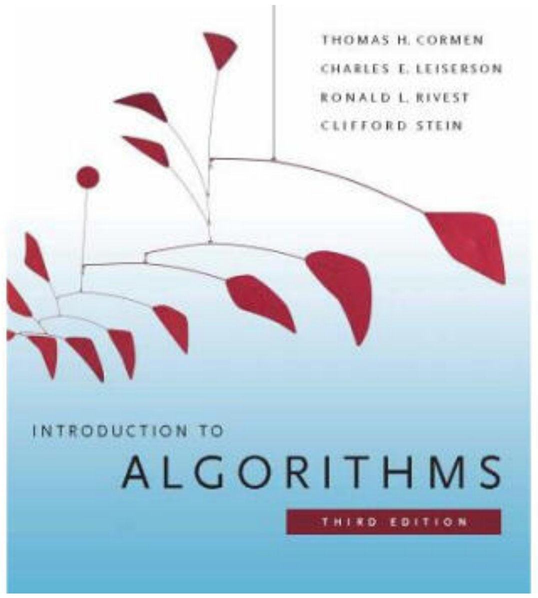
# Textbooks

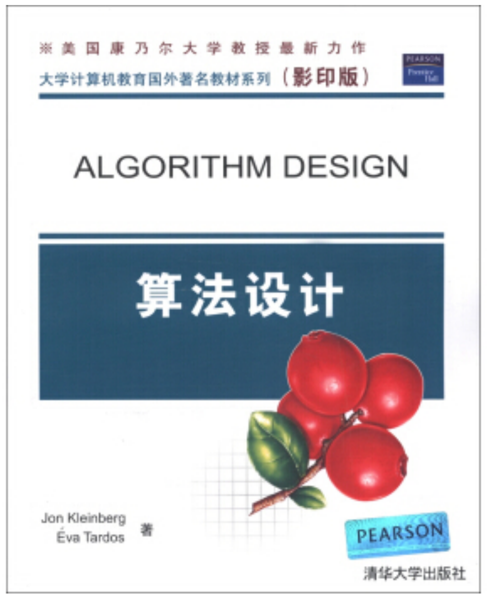
- **Course outline: LADA**
  - Lectures on Algorithm Design & Analysis (slides)
- **Course contents**
  - Algorithm Design and Analysis

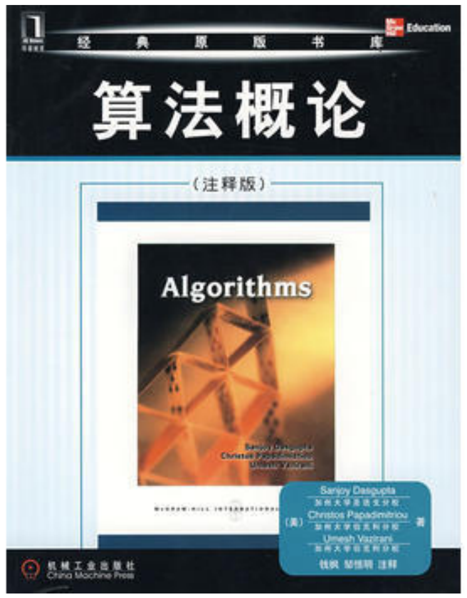
More info about this book:
https://zhuanlan.zhihu.com/p/24150569

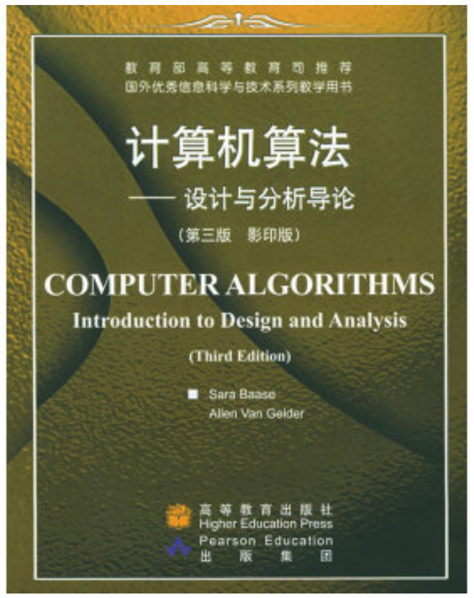
# Textbooks

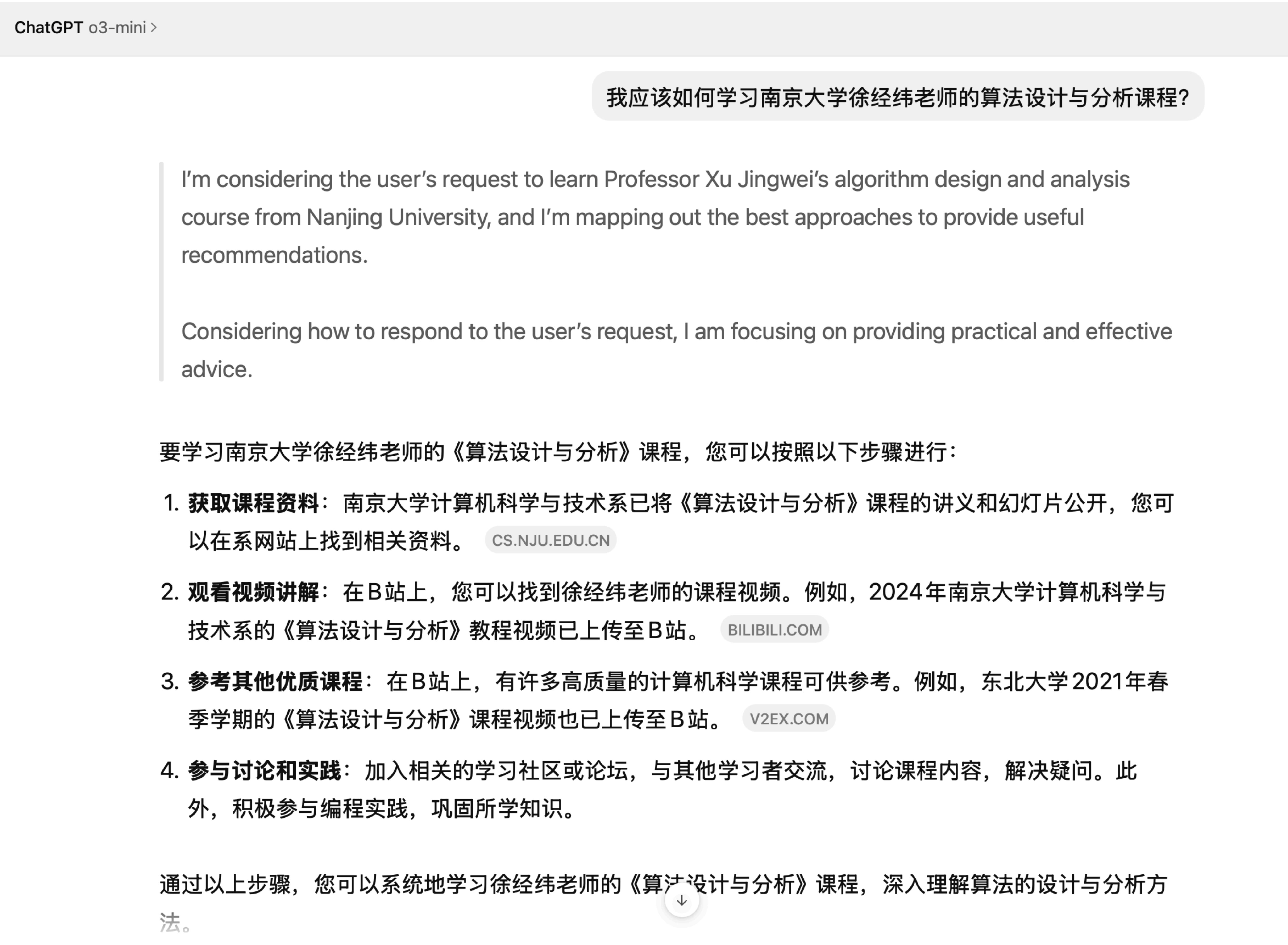
- **Further reading**
  - Introduction to Algorithms (CLRS) **https://walkccc.me/CLRS/**
  - Algorithms
  - Algorithm Design
  - Computer Algorithms*











ChatGPT o3-mini >

我应该如何学习南京大学徐经纬老师的算法设计与分析课程?

> I'm considering the user's request to learn Professor Xu Jingwei's algorithm design and analysis course from Nanjing University, and I'm mapping out the best approaches to provide useful recommendations.
>
> Considering how to respond to the user's request, I am focusing on providing practical and effective advice.

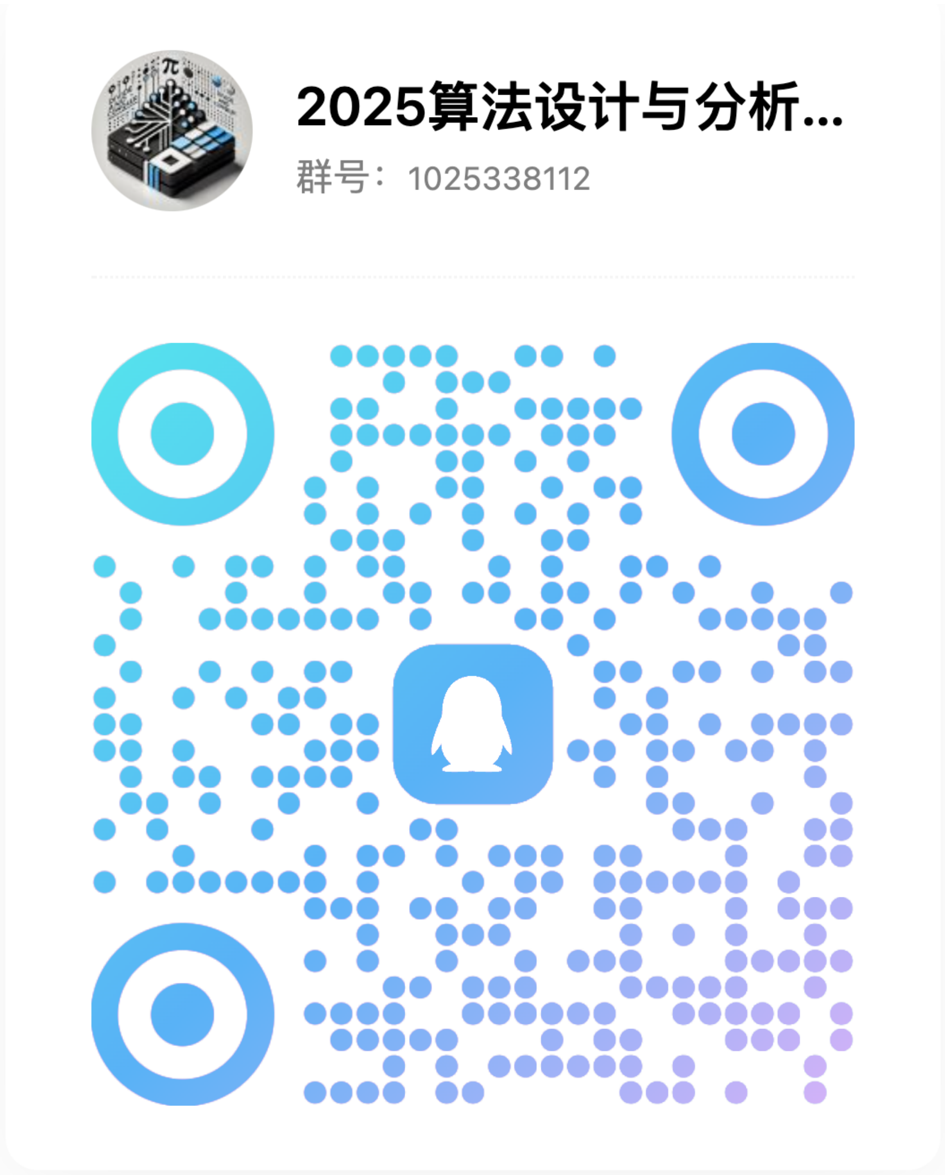
要学习南京大学徐经纬老师的《算法设计与分析》课程，您可以按照以下步骤进行：

1. **获取课程资料**：南京大学计算机科学与技术系已将《算法设计与分析》课程的讲义和幻灯片公开，您可以在系网站上找到相关资料。 CS.NJU.EDU.CN
2. **观看视频讲解**：在B站上，您可以找到徐经纬老师的课程视频。例如，2024年南京大学计算机科学与技术系的《算法设计与分析》教程视频已上传至B站。 BILIBILI.COM
3. **参考其他优质课程**：在B站上，有许多高质量的计算机科学课程可供参考。例如，东北大学2021年春季学期的《算法设计与分析》课程视频也已上传至B站。 V2EX.COM
4. **参与讨论和实践**：加入相关的学习社区或论坛，与其他学习者交流，讨论课程内容，解决疑问。此外，积极参与编程实践，巩固所学知识。

通过以上步骤，您可以系统地学习徐经纬老师的《算法设计与分析》课程，深入理解算法的设计与分析方法。

# Course Websites

- **算法QQ群**
  - 1025338112
- **Online Judge平台**
  - Coming soon



# Criteria of Assessment

- 作业 20%
    - 纸质作业
    - OJ平台
- 期中考试 20%
- 期末考试 60%

## Algorithm Design & Analysis

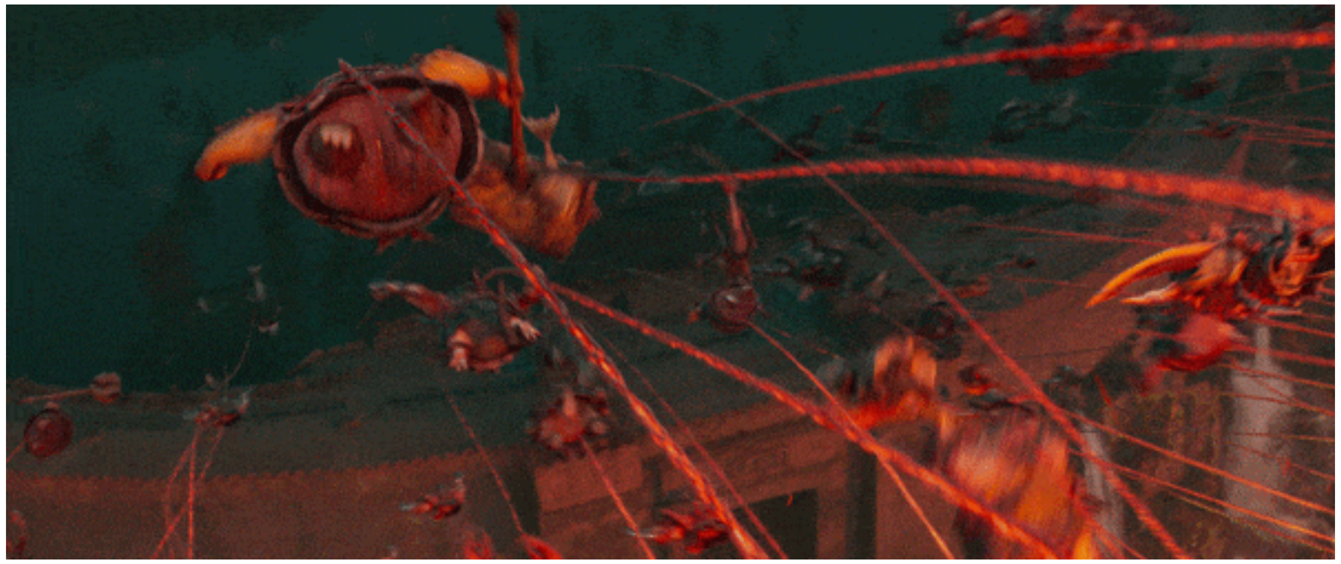
- **Algorithm - the spirit of computing**
  - Model of computation
- **Algorithm by example**
  - Greatest common divisor
  - Sequential search

## Algorithm Design & Analysis

- **Algorithm design & analysis**
  - Design: correctness
  - Analysis: worst-case / average-case cost

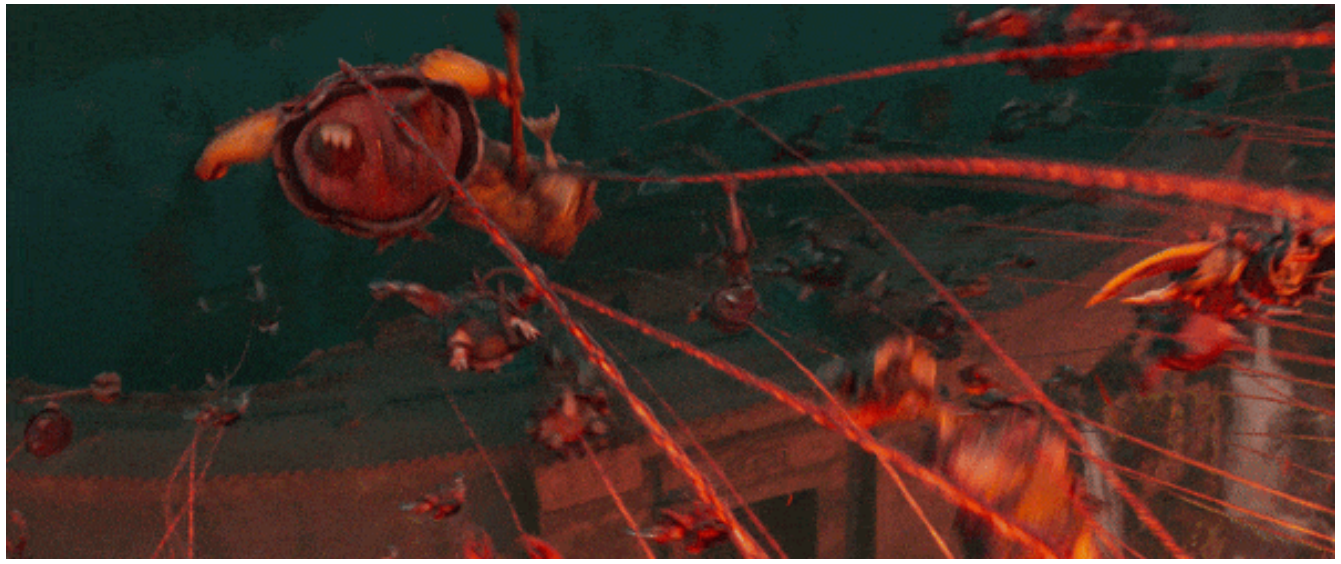
# Computer and Computing

- **Problem 1: why the computer seems to be able to do anything?**
  - Scientific computing, document processing, computer vision, movies, computer games, …



# Computer and Computing

- **Problem 1: why the computer seems to be able to do anything?**
    - Scientific computing, document processing, computer vision, movies, computer games, …



如何在南京大学计算机学院获得 GPA 5.0 的高分？

Reasoned for a couple of seconds

> I'm curious about achieving a GPA of 5.0 at Nanjing University's School of Computer Science. This involves deep analysis, strategic planning, and consistent effort, as consistent top scores indicate thorough understanding and mastery of the coursework.
>
> To achieve a GPA of 5.0 at Nanjing University's School of Computer Science, I'll delve into effective study habits, staying organized, maintaining motivation, and seeking guidance from instructors and fellow students.

要在南京大学计算机学院获得 GPA 5.0 的高分，除了需要扎实的学术基础外，还需要良好的学习策略和时间管理能力。以下是一些建议，帮助你在学术生涯中取得优异成绩：

1. **基础知识扎实**：

- 确保你对课程的基础概念和知识点有深入的理解。计算机科学的许多课程都建立在前期的知识基础上，打好基础是关键。

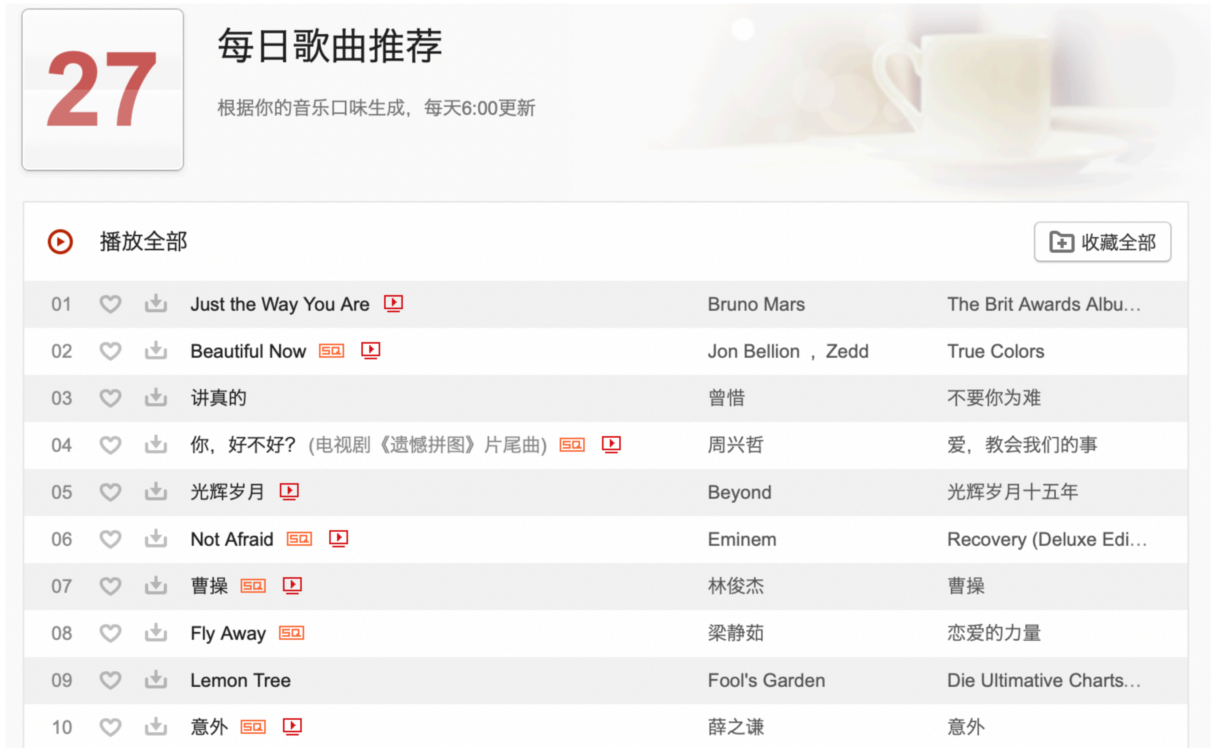
2. **课前预习**：

- 课前预习能够帮助你提前了解课程内容和难点，课堂上更容易跟上教授的讲解，提高学习效率。

3. **主动参与课堂讨论**：

# Computer and Computing

- **Problem 2: what can / cannot be efficiently done by a computer?**
  - Manage millions of songs v.s. music composition



27

每日歌曲推荐

根据你的音乐口味生成，每天6:00更新

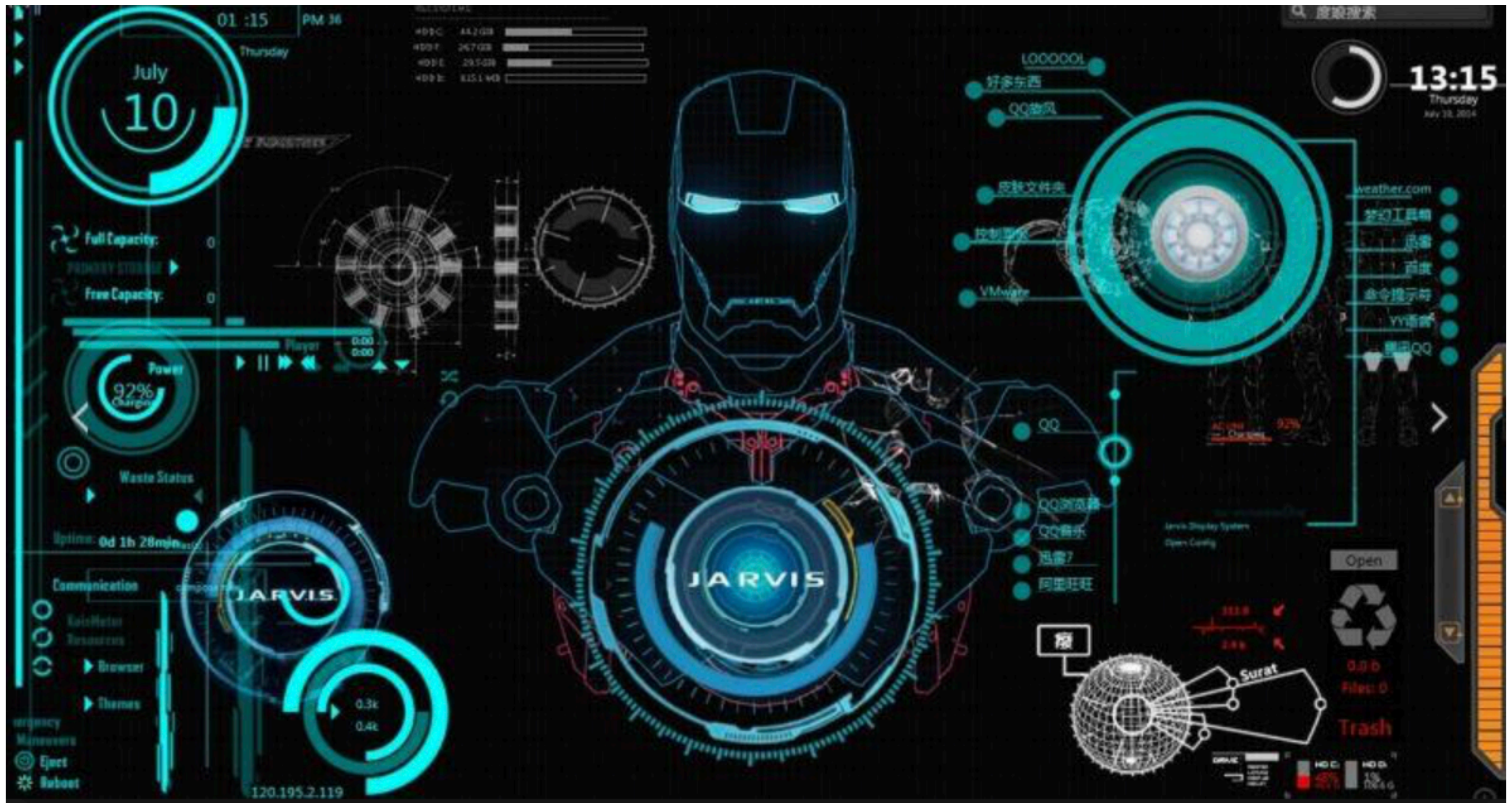
播放全部

收藏全部

| | | | |
|---|---|---|---|
| 01 | Just the Way You Are | Bruno Mars | The Brit Awards Albu… |
| 02 | Beautiful Now | Jon Bellion ，Zedd | True Colors |
| 03 | 讲真的 | 曾惜 | 不要你为难 |
| 04 | 你，好不好？（电视剧《遗憾拼图》片尾曲） | 周兴哲 | 爱，教会我们的事 |
| 05 | 光辉岁月 | Beyond | 光辉岁月十五年 |
| 06 | Not Afraid | Eminem | Recovery (Deluxe Edi… |
| 07 | 曹操 | 林俊杰 | 曹操 |
| 08 | Fly Away | 梁静茹 | 恋爱的力量 |
| 09 | Lemon Tree | Fool's Garden | Die Ultimative Charts… |
| 10 | 意外 | 薛之谦 | 意外 |

## Computing in Everyday Life

# Computing in Everyday Life

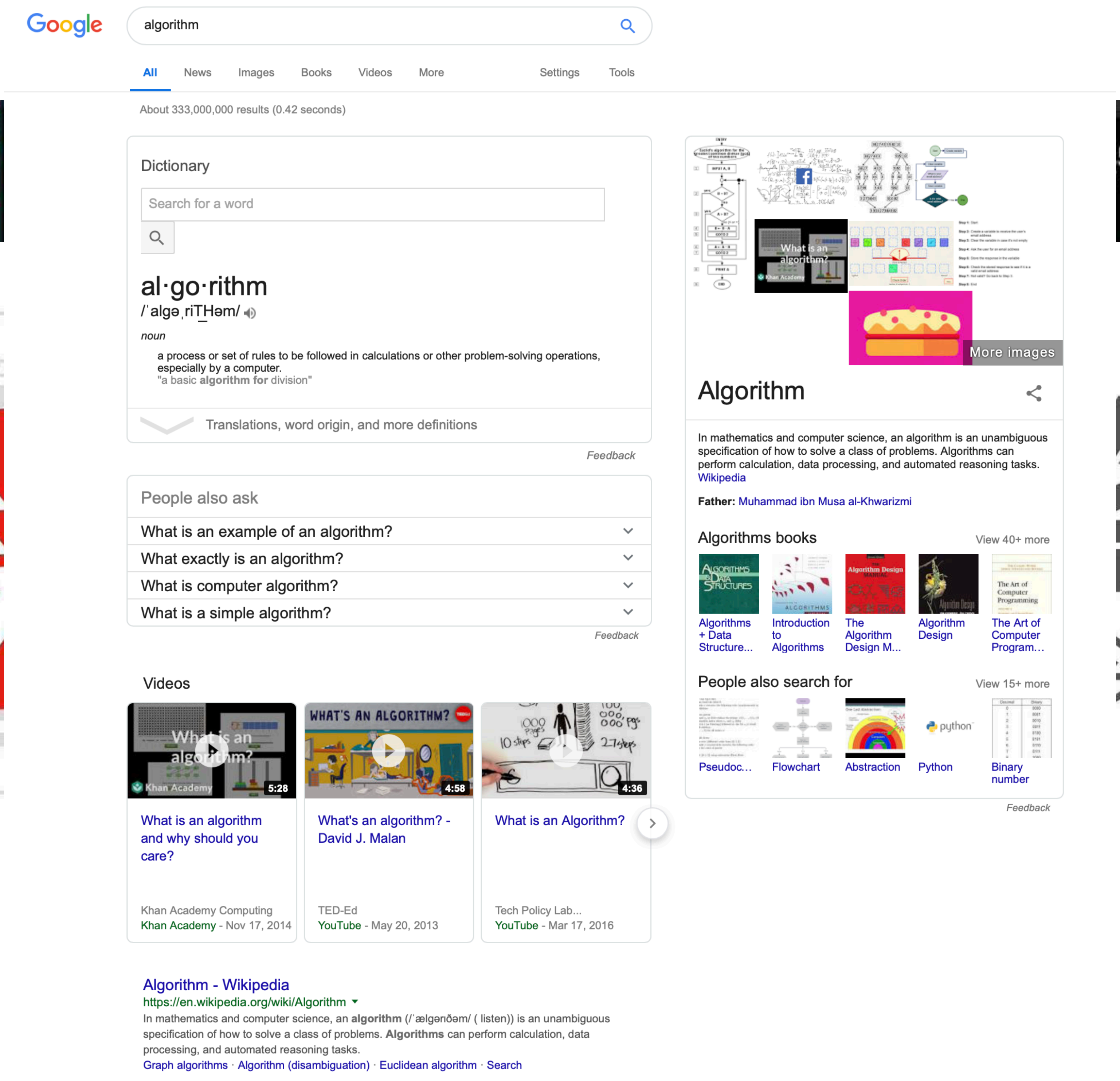


# Computing in Everyday Life

今日头条

你关心的 才是头条

# Computing in Everyday Life



Google

algorithm

All News Images Books Videos More Settings Tools

About 333,000,000 results (0.42 seconds)

Dictionary

Search for a word

al·go·rithm

/ˈalgəˌriT͟Həm/

*noun*

a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
"a basic **algorithm for** division"

Translations, word origin, and more definitions

Feedback

People also ask

What is an example of an algorithm?

What exactly is an algorithm?

What is computer algorithm?

What is a simple algorithm?

Feedback

Videos

| What is an algorithm and why should you care? | What's an algorithm? - David J. Malan | What is an Algorithm? |
|---|---|---|
| 5:28 | 4:58 | 4:36 |
| Khan Academy Computing | TED-Ed | Tech Policy Lab... |
| Khan Academy - Nov 17, 2014 | YouTube - May 20, 2013 | YouTube - Mar 17, 2016 |

Algorithm - Wikipedia
https://en.wikipedia.org/wiki/Algorithm
In mathematics and computer science, an **algorithm** (/ˈælɡərɪðəm/ ( listen)) is an unambiguous specification of how to solve a class of problems. **Algorithms** can perform calculation, data processing, and automated reasoning tasks.
Graph algorithms · Algorithm (disambiguation) · Euclidean algorithm · Search

More images

Algorithm

In mathematics and computer science, an algorithm is an unambiguous specification of how to solve a class of problems. Algorithms can perform calculation, data processing, and automated reasoning tasks. Wikipedia

**Father:** Muhammad ibn Musa al-Khwarizmi

Algorithms books — View 40+ more

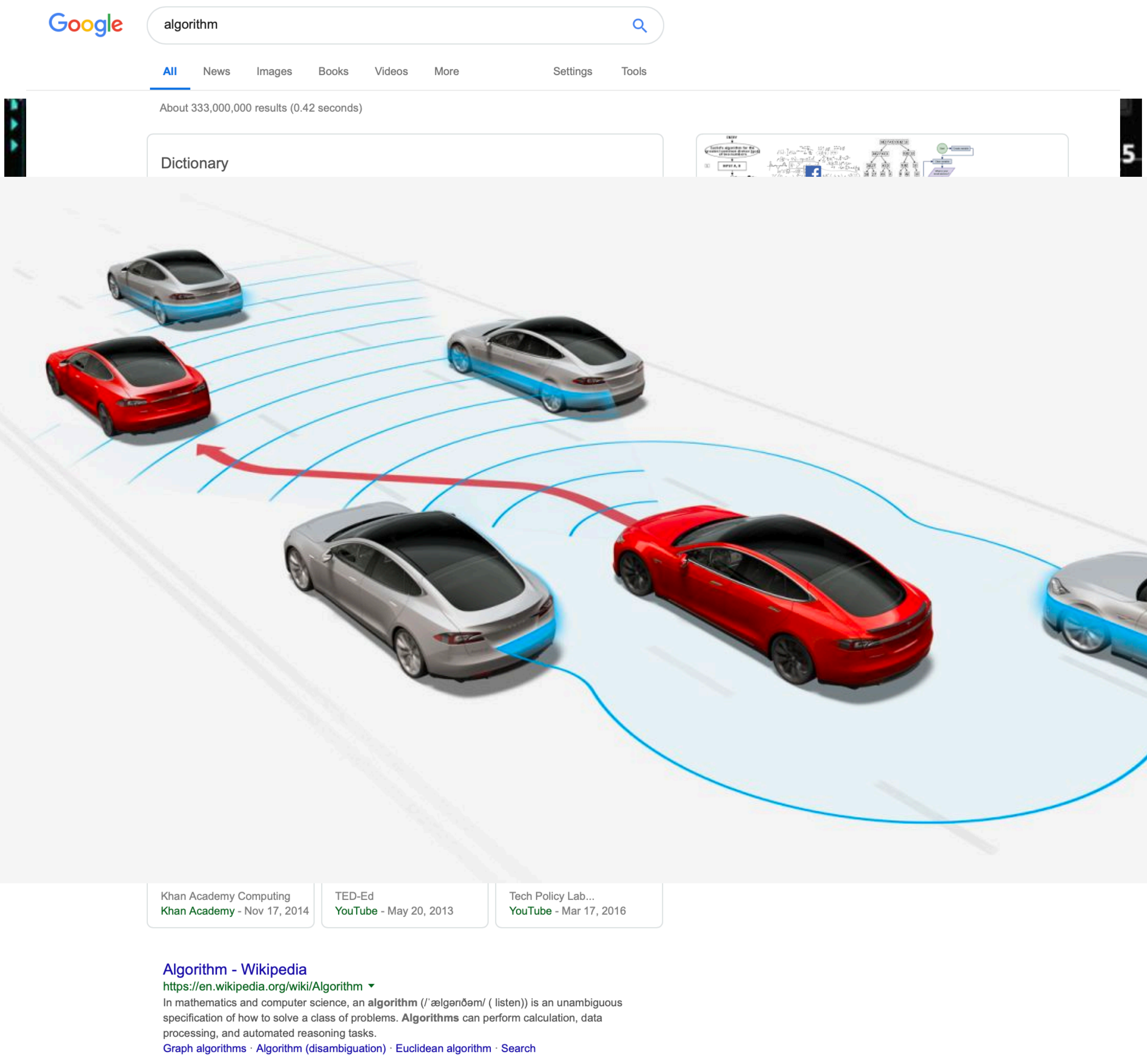
Algorithms + Data Structure... · Introduction to Algorithms · The Algorithm Design M... · Algorithm Design · The Art of Computer Program...

People also search for — View 15+ more

Pseudoc... · Flowchart · Abstraction · Python · Binary number

Feedback

# Computing in Everyday Life



algorithm

All News Images Books Videos More Settings Tools

About 333,000,000 results (0.42 seconds)

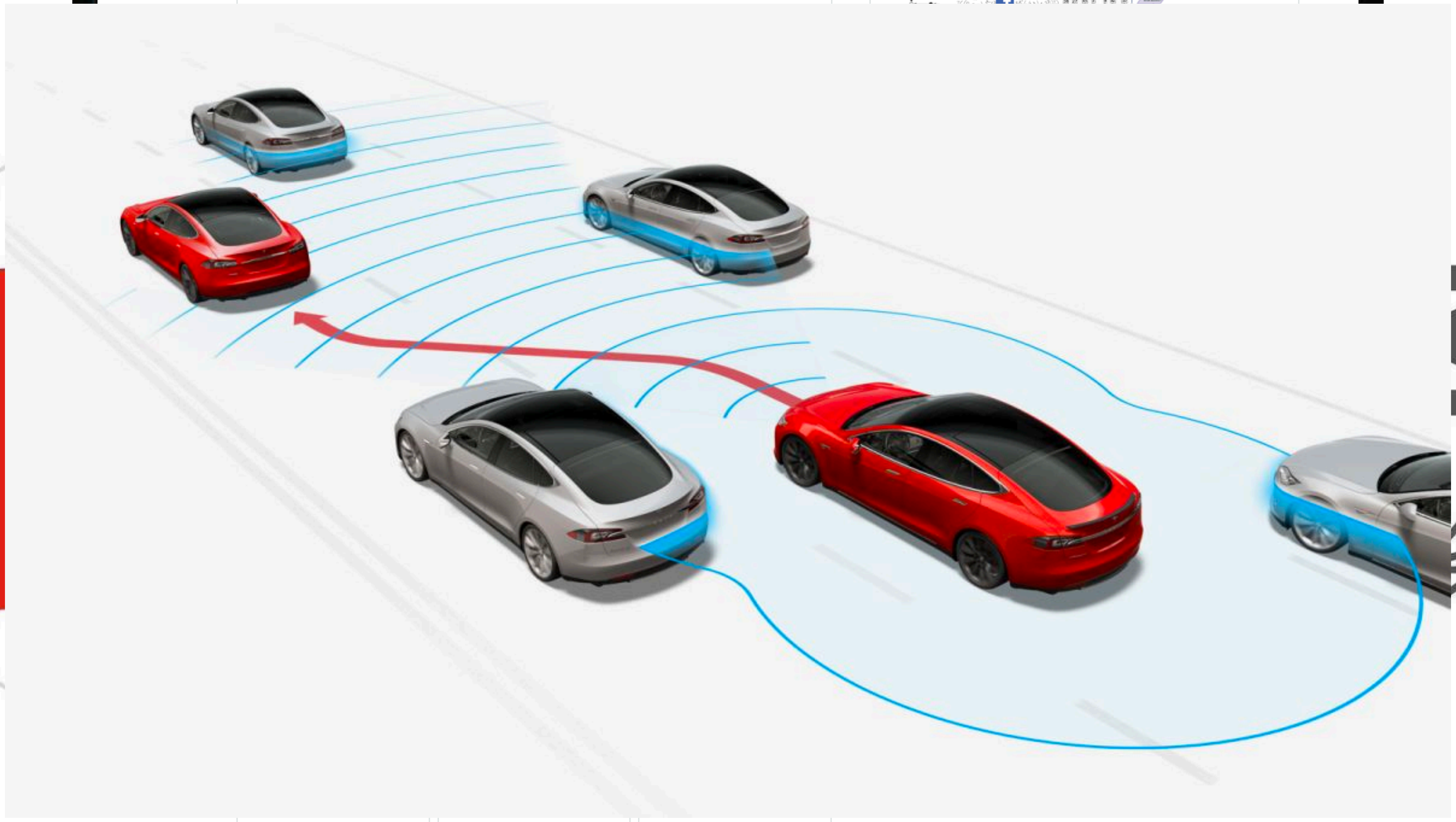
Dictionary

Khan Academy Computing
Khan Academy - Nov 17, 2014

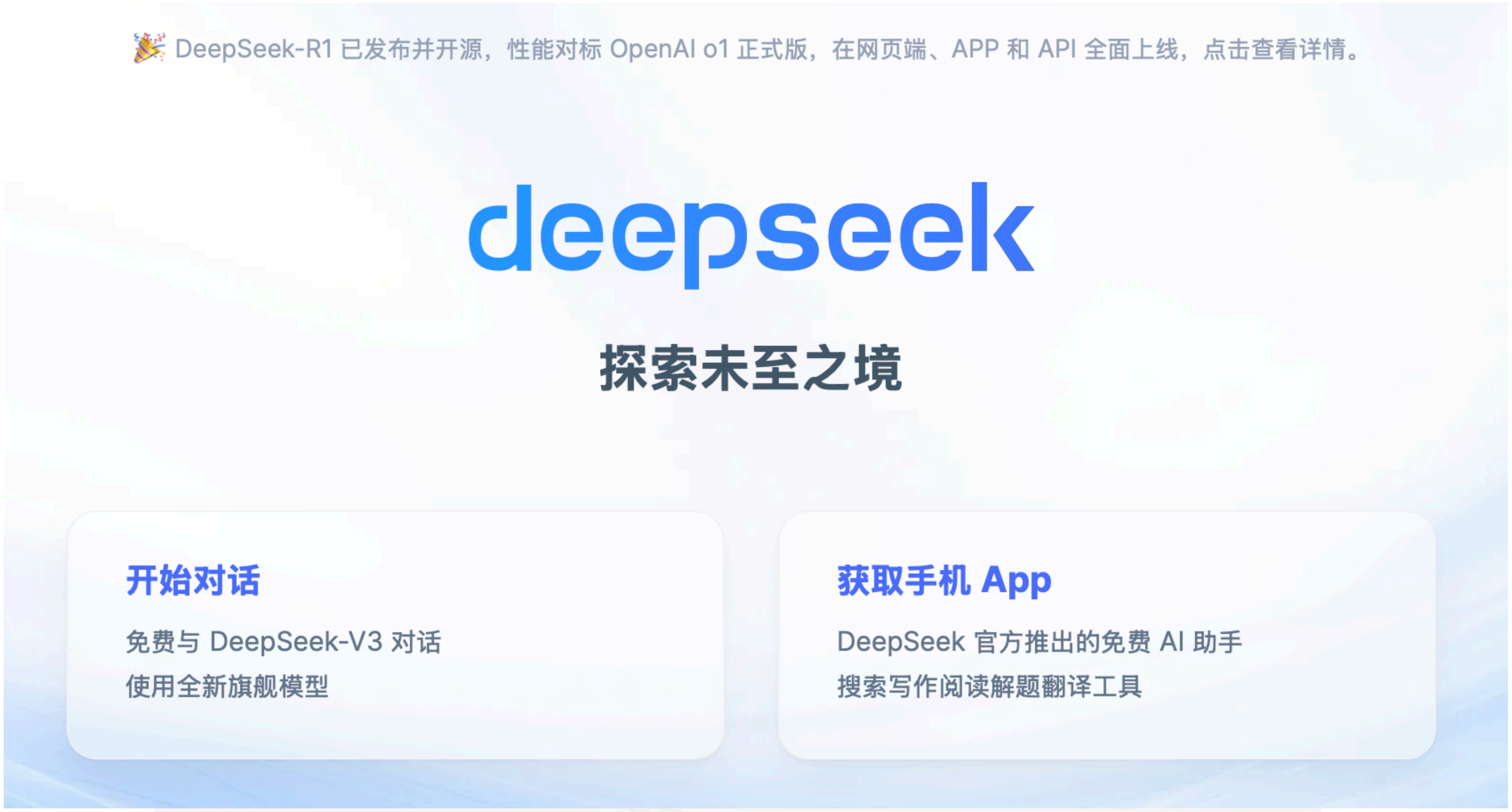
TED-Ed
YouTube - May 20, 2013

Tech Policy Lab...
YouTube - Mar 17, 2016

Algorithm - Wikipedia
https://en.wikipedia.org/wiki/Algorithm
In mathematics and computer science, an **algorithm** (/ˈælɡərɪðəm/ ( listen)) is an unambiguous specification of how to solve a class of problems. **Algorithms** can perform calculation, data processing, and automated reasoning tasks.
Graph algorithms · Algorithm (disambiguation) · Euclidean algorithm · Search



# Computing in Everyday Life



DeepSeek-R1 已发布并开源，性能对标 OpenAI o1 正式版，在网页端、APP 和 API 全面上线，点击查看详情。

deepseek

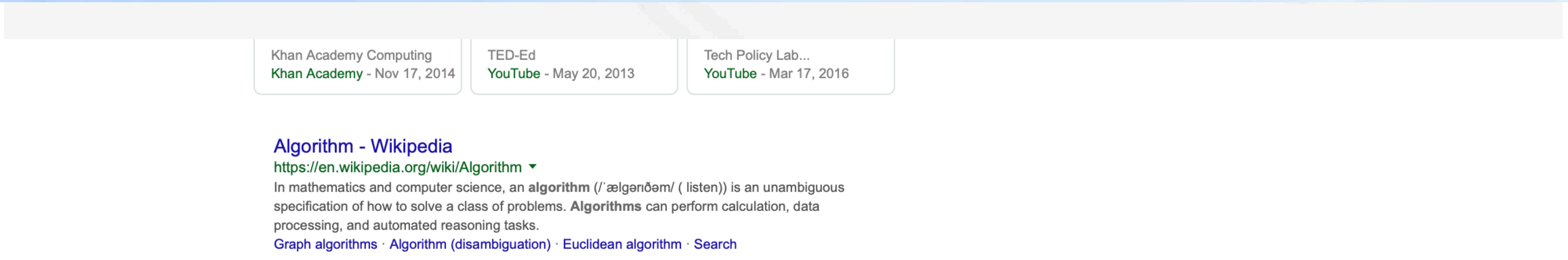
探索未至之境

**开始对话**

免费与 DeepSeek-V3 对话
使用全新旗舰模型

**获取手机 App**

DeepSeek 官方推出的免费 AI 助手
搜索写作阅读解题翻译工具



Khan Academy Computing
Khan Academy - Nov 17, 2014

TED-Ed
YouTube - May 20, 2013

Tech Policy Lab...
YouTube - Mar 17, 2016

Algorithm - Wikipedia
https://en.wikipedia.org/wiki/Algorithm
In mathematics and computer science, an **algorithm** (/ˈælgərɪðəm/ ( listen)) is an unambiguous specification of how to solve a class of problems. **Algorithms** can perform calculation, data processing, and automated reasoning tasks.
Graph algorithms · Algorithm (disambiguation) · Euclidean algorithm · Search

# Algorithm

今日头条
你关心的 才是头条

# Algorithm

🎉 DeepSeek-R1 已发布并开源，性能对标 OpenAI o1 正式版，在网页端、APP 和 API 全面上线，点击查看详情。

deepseek

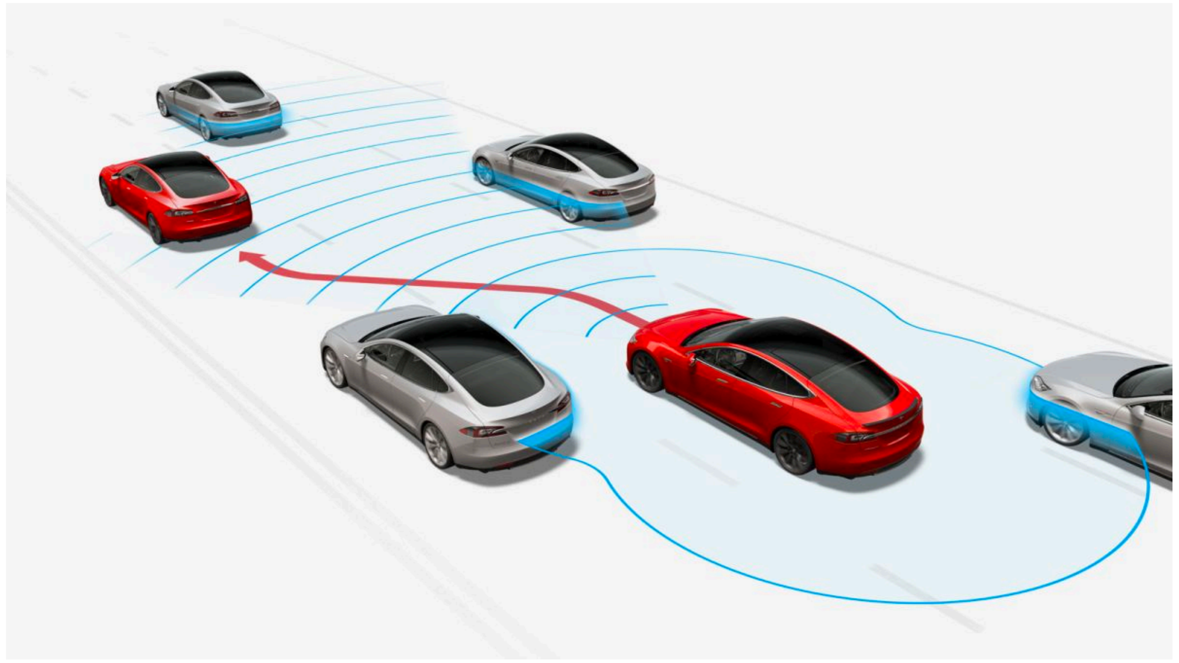
探索未至之境

**开始对话**

免费与 DeepSeek-V3 对话
使用全新旗舰模型

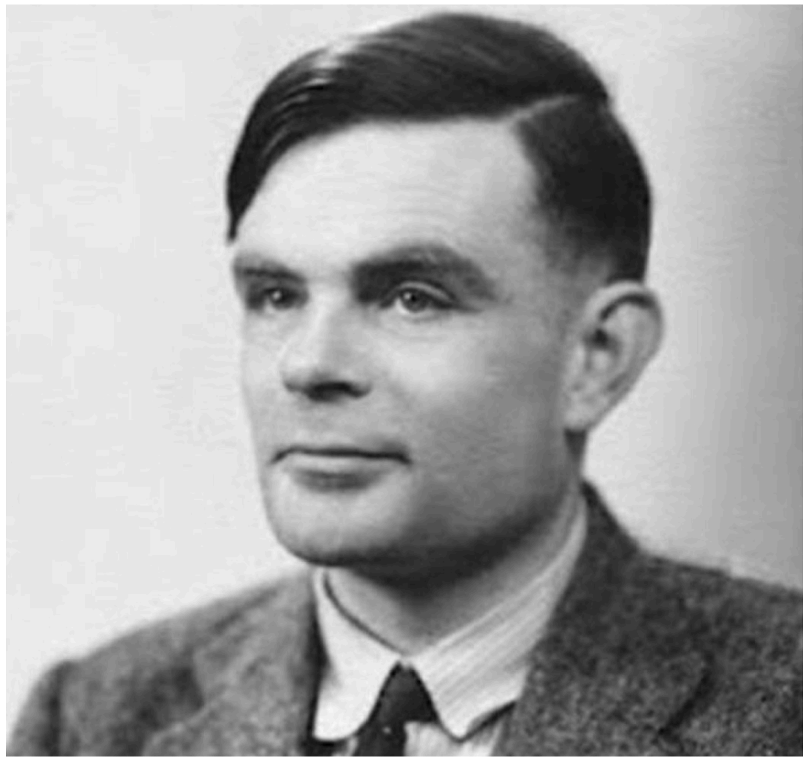
**获取手机 App**

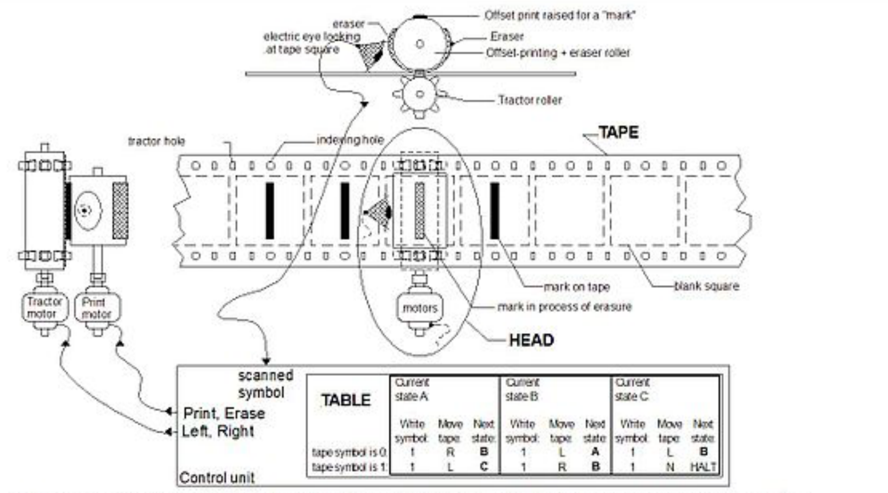
DeepSeek 官方推出的免费 AI 助手
搜索写作阅读解题翻译工具



# Computer and Computing

- **Computing**
  - Encoding everything into ‘0’s and ‘1’s
  - Operations over ‘1’s and ‘0’s
  - Decoding the ‘1’s and ‘0’s
- **Turing machine**
  - An abstract/logical computer



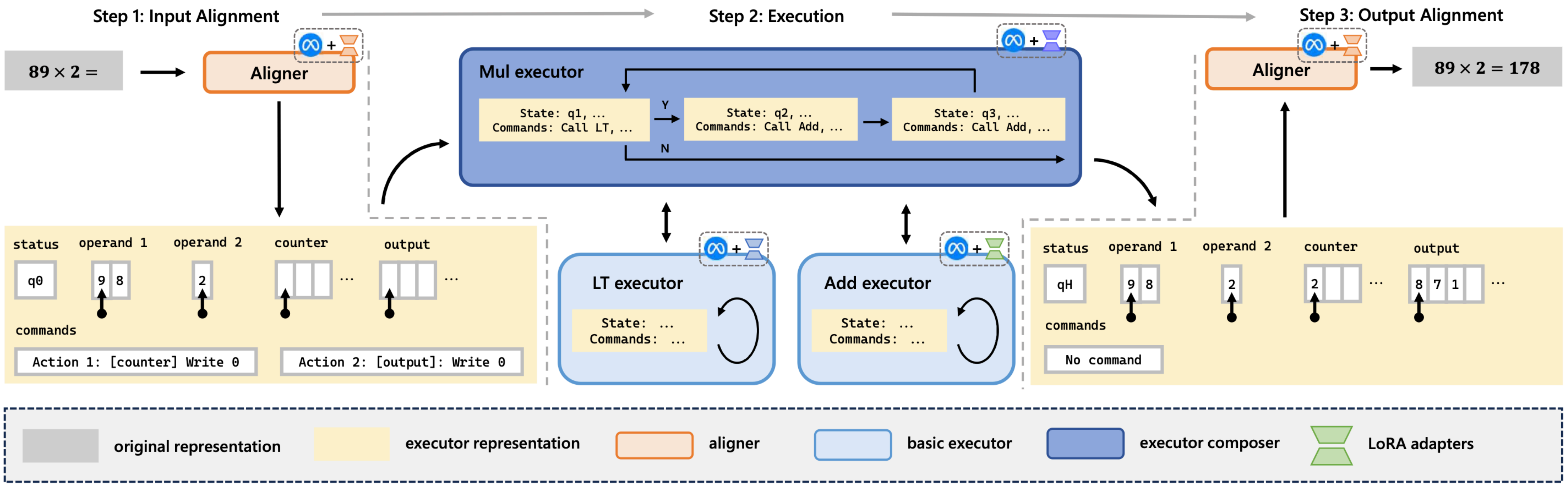


| TABLE | Current state A | | | Current state B | | | Current state C | | |
|---|---|---|---|---|---|---|---|---|---|
| | Write symbol: | Move tape: | Next state: | Write symbol: | Move tape: | Next state | Write symbol: | Move tape: | Next state |
| tape symbol is 0: | 1 | R | **B** | 1 | L | **A** | 1 | L | **B** |
| tape symbol is 1: | 1 | L | **C** | 1 | R | **B** | 1 | N | HALT |

A fanciful mechanical Turing machine's TAPE and HEAD. The TABLE instructions might be on another "read only" tape, or perhaps on punch-cards. Usually a "finite state machine" is the model for the TABLE.

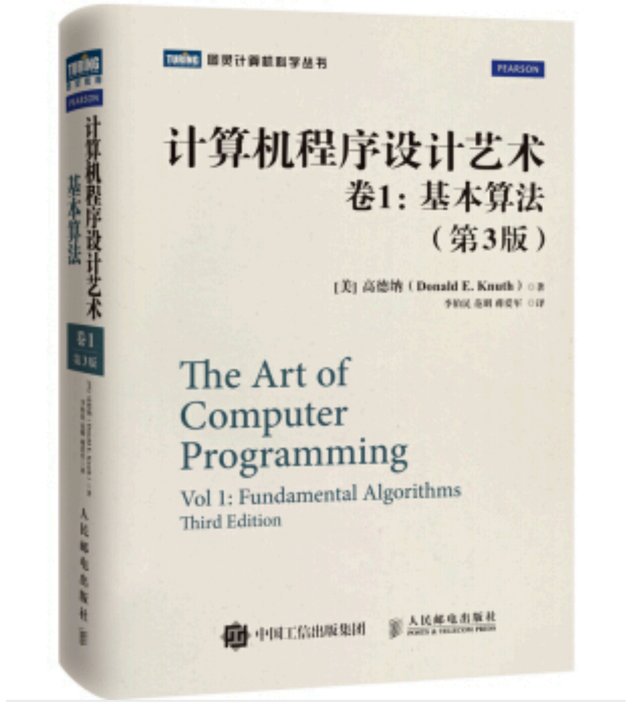
# Executing Arithmetic: Fine-Tuning Large Language Models as Turing Machines

**Junyu Lai  Jiahe Xu  Yao Yang  Yunpeng Huang  Chun Cao & Jingwei Xu** *
National Key Laboratory for Novel Software Technology, Nanjing University
{junyu_lai,jiahexu,yangyao,hyp}@smail.nju.edu.cn
{caochun,jingweix}@nju.edu.cn



# 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


计算机程序设计艺术
卷1：基本算法
（第3版）
The Art of Computer Programming
Vol 1: Fundamental Algorithms
Third Edition

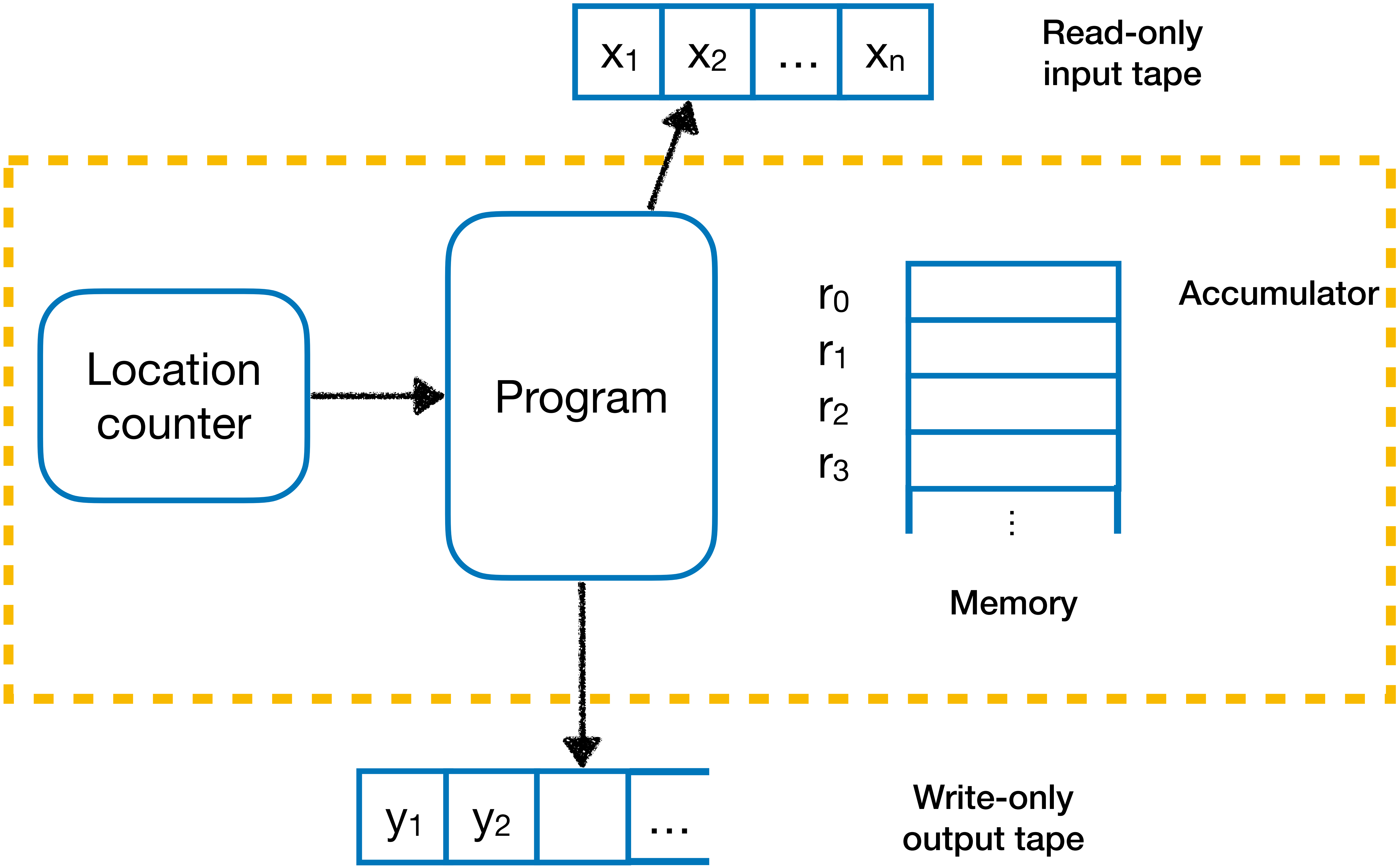
# Algorithm

- **Essential issues**
  - Model of computation
  - Algorithm design
  - Algorithm analysis

# 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 simple operation
  - Unlimited memory

**Tradeoff: accuracy v.s. ease of use**

# 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**
  - 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 r=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 search**

```
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 $n \le N_0$, Euclid(m, n) is correct;
  - Induction
    - Euclid(m, $N_0$+1) = Euclid($N_0$+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

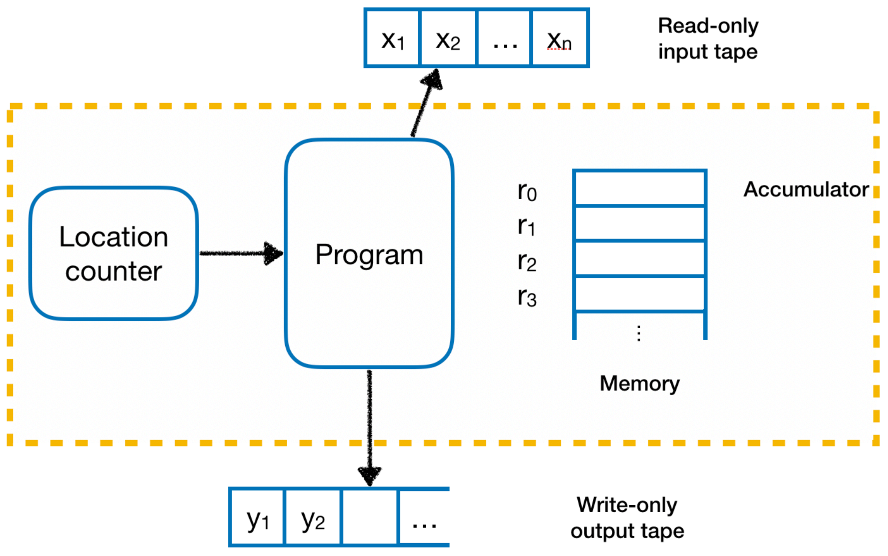
- **Criteria**
  - Performance metrics
- **Worst case**
  - Best case?
- **Average case**
  - Average cost?
- **Advanced topics**
  - Lower bound, optimality, …

# 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

RAM Model

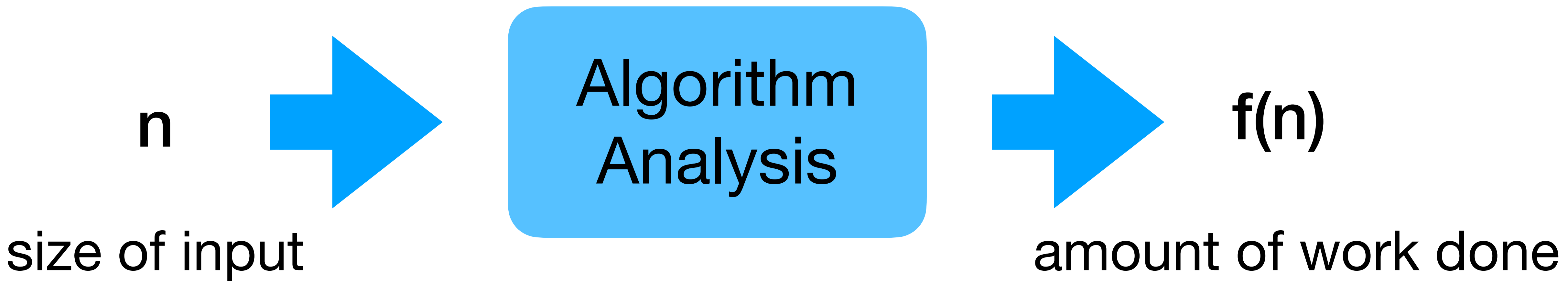


- Criteria
  - Critical operation
  - How many critical operation are conducted
- For example

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

# Algorithm Analysis

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



# Worst-case Complexity

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

# Average-case Complexity

- **A(n)**
  - Weighted average

$$A(n) = \sum_{I \in D(n)} Pr(I) f(I)$$

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

# Average-case Cost of SeqSearch

- **Case 1: K is in E[ ]**
    - Assumptions:
        - Assuming that K is in E[ ]
        - Assuming no same entries in E[ ]
        - Each possible input appears with equality (thus, K in the ith location with probability 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

$$\begin{aligned} A(n) &= Pr(succ)A_{succ}(n) + Pr(fail)A_{fail}(n) \\ &= q\frac{n+1}{2} + (1-q)n \end{aligned}$$

**How to make reasonable assumptions?**

# Algorithm Analysis

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

Thank you!
Q & A