

计算机科学中的经典算法问题

Great Ideas in Computer Science

王晓茹
计算机学院
北京邮电大学

算法学习的脉络



学习进度条

计算机科学中的经典算法问题
Great Ideas in Computer Science

纳什均衡-博弈论

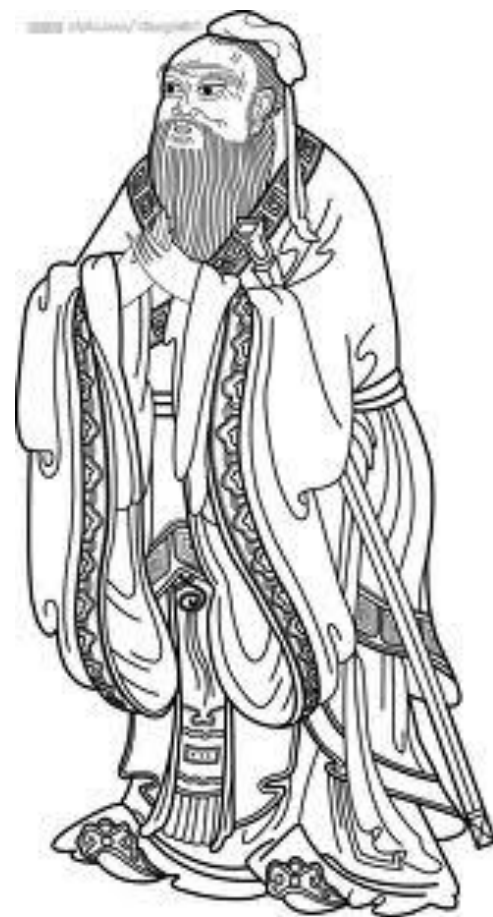
1. 《论语·宪问》

子曰：“骥不称其力，称其德也。”

或曰：“以德报怨，何如？”

子曰：“何以报德？”

以直报怨，以德报德。”



- **【注释】**

骥：千里马。古代称善跑的马为骥。

- **【译文】**

孔子说：“千里马值得称赞的不是它的气力，而是称赞它的品德。”

- **【译文】**

有人说：“用恩德来报答怨恨怎么样？”

- 孔子说：“用什么来报答恩德呢？应该用正直来报答怨恨，用恩德来报答恩德。”

- **【评析】**

孔子不同意“以德报怨”的做法，认为应当是“以直报怨”。

这是说，不以有旧恶旧怨而改变自己的公平正直，也就是坚持了正直，“以直报怨”对于个人道德修养极为重要。

1. 《论语·宪问》

子曰：“骥不称其力，称其德也。”

或曰：“以德报怨，何如？”

子曰：“**何以报德？以直报怨，以德报德。**”

如何**证明**孔子的说法是正确的？

请**计算**孔子的说法是正确的！



纳什均衡

- 是博弈论的一个重要概念，可用于解释多种实际问题，如经济学、政治学、生物学等。
- 数学家约翰纳什提出，他因此获得了**1994**年的诺贝尔经济学奖。
- 在经济领域、在国家竞争领域、在战争领域，在各个领域都在实践着纳什均衡的精髓。



纳什均衡-定义

- 假设有 n 个局中人参与博弈，给定其他人策略的条件下，每个局中人选择自己的最优策略（个人最优策略可能依赖于也可能不依赖于他人的策略），从而使自己收益最大化。
- 所有局中人策略构成一个策略组合（**Strategy Profile**）。

纳什定理：只要参与博弈的几方策略是有限的，那么就一定存在一种**平衡状态**，大家都会采用这种平衡策略，而没有单方面改变策略的动力。

这种平衡状态就是**纳什均衡**。

囚徒困境



prisoner's dilemma

- 1950年，由就职于兰德公司的梅里尔·弗勒德（Merrill Flood）和梅尔文·德雷希尔（Melvin Dresher）拟定出相关困境的理论，后来由顾问艾伯特·塔克（Albert Tucker）以囚徒方式阐述，并命名为“囚徒困境”。
- 经典的囚徒困境如下：



警察对两个人分别审讯 并且告诉他们政策

- 若一人**认罪**并作证检控对方（相关术语称“**背叛**”对方），而对方保持沉默，此人将即时获释，沉默者将判监10年。
- 若二人都保持沉默（相关术语称互相“**合作**”），则二人同样判监半年。
- 若二人都**互相检举**（互相“**背叛**”），则二人同样判监2年。

如何分析

	甲沉默（合作）	甲认罪（背叛）
乙沉默（合作）	二人同服刑半年	甲即时获释；乙服刑10年
乙认罪（背叛）	甲服刑10年；乙即时获释	二人同服刑2年

甲方
评估

沉默-半年
或10年

认罪-获释
或2年

乙方
评估

沉默-半年
或10年

认罪-获释
或2年

- 非合作博弈论中的假设

- 囚徒困境假定每个参与者（即“囚徒”）都是利己的，即都寻求最大自身利益，而不关心另一参与者的利益。

- 参与者都是理性的。

- 参与者某一策略所得利益，如果在任何情况下都比其他策略要低的话，此策略称为“严格劣势”，理性的参与者绝不会选择。

- 另外，没有任何其他力量干预个人决策，参与者可完全按照自己意愿选择策略。

● 纳什均衡

- 二人面对的情况一样，所以二人的理性思考都会得出相同的结论——选择认罪（背叛）。
- 认罪（背叛）是两种策略之中的**支配性**策略。
- 这场博弈中唯一可能达到的**纳什均衡**，就是双方参与者都认罪（背叛）对方，结果二人同样服刑**2**年。

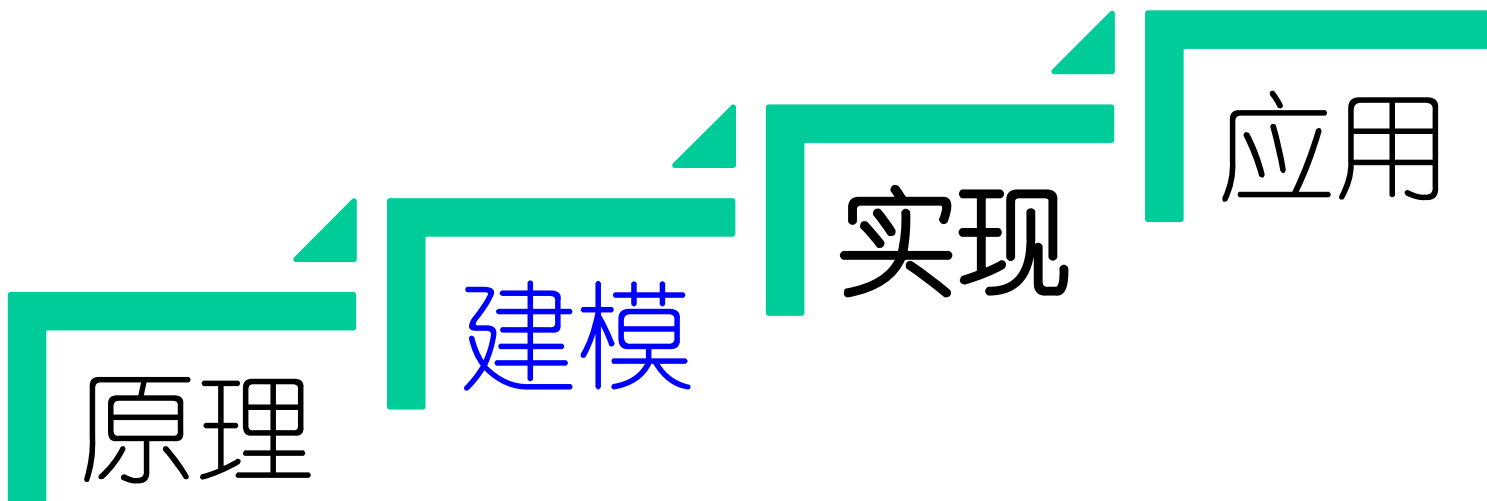
● 帕累托最优解决方案

- 以全体利益而言，如果两个参与者都合作保持沉默，两人都只会被判刑半年，总体利益更高，结果也比两人背叛对方、判刑**2**年的情况较佳。

● “困境”所在

- 两个囚徒都**理性地**选择背叛，结果二人判监均比合作为高，总体利益较合作为低。
- 博弈的纳什均衡，显然不是顾及团体利益的**帕累托最优**解决方案。
- **非零和**博弈中，帕累托最优和纳什均衡是互相冲突的。

算法学习的脉络



学习进度条

练练思维

- 经济学上，市场的供给和需求是平衡的，而且在平衡的时候,请大家讨论：

商品的价格VS成本的关系

- 囚徒困境下：
商品的价格=商家的成本

如何打破这种平衡，实现价格锁定？

重复的囚徒困境

- 罗伯特·阿克塞尔罗德在其著作《合作的进化》中，探索了经典囚徒困境情景的一个扩展，并把它称作“重复的囚徒困境”。
- 在这个博弈中，参与者必须反复地选择他们彼此相关的策略，并且记住他们以前的对抗。
- 阿克塞尔罗德邀请全世界的学术同行来设计计算机策略，并在一个重复囚徒困境竞赛中互相竞争。参赛的程序的差异广泛地存在于这些方面：算法的复杂性、最初的对抗、宽恕的能力等等。
- 阿克塞尔罗德发现，当这些对抗被每个选择不同策略的参与者一再重复了很长时间之后，从利己的角度来判断，最终“贪婪”策略趋向于减少，而比较“利他”策略更多地被采用。他用这个博弈来说明，通过自然选择，一种利他行为的机制可能从最初纯粹的自私机制进化而来。

难题的解决-计算机模拟

- 为解决“囚徒困境”难题，美国曾组织竞赛，要求参赛者根据“重复囚徒困境”（双方不止一次相遇，“背叛”可能在以后遭到报复）来设计程序。
- 将程序输入计算机反复互相博弈，以最终得分评估优劣（双方合作各得3分；双方背叛各得1分；一方合作一方背叛，合作方得0分，背叛方得5分）；
 - 有些程序采用“随机”对策；
 - 有些采用“永远背叛”对策；
 - 有些采用“永远合作”对策；
- 最佳确定性策略被认为是“以牙还牙”，这是阿纳托尔·拉波波特（Anatol Rapoport）开发并运用到锦标赛中的方法。它是所有参赛程序中最简单的，只包含了四行BASIC语言，并且赢得了比赛。这个策略只不过是在重复博弈的开头合作，然后，采取你的对手前一回合的策略。

“一报还一报” 策略

- 我方在第一次相遇时选择“合作”，之后就采取对方上一次的选项。这意味着：
- 在对方每一次背叛后，我方就“以牙还牙”，也背叛一次；
- 对方每一次合作后，我方就“以德报德”一次；
- **《论语·宪问》中，有人问孔子：“以德报怨，何如？”孔子反问：“何以报德？以直报怨，以德报德。”**
- 该策略有别于“善良”的“永远合作”或“邪恶”的“永远背叛”对策，及对方一旦“不忠”，我方就不再给机会，长久对抗的策略。

算法分析

- 如果你选择“永远背叛”策略，你或许会在第一局拿到最高分，但之后的各局可能都只能拿到低分，最后虽然可能“战胜”不少对手，但由于总分很低，最终难逃被淘汰出局的命运。
- 所以除非很难与对方再次相遇，不用担心其日后的反应，才可选择对抗与背叛；
- 而在长期互动、博弈的关系中，“一报还一报”是最佳策略：它是善意的，从不首先背叛；它不迂腐，不管过去相处多好，仍然对背叛有反应；它是宽容的，不因一次背叛而选择玉石俱焚。

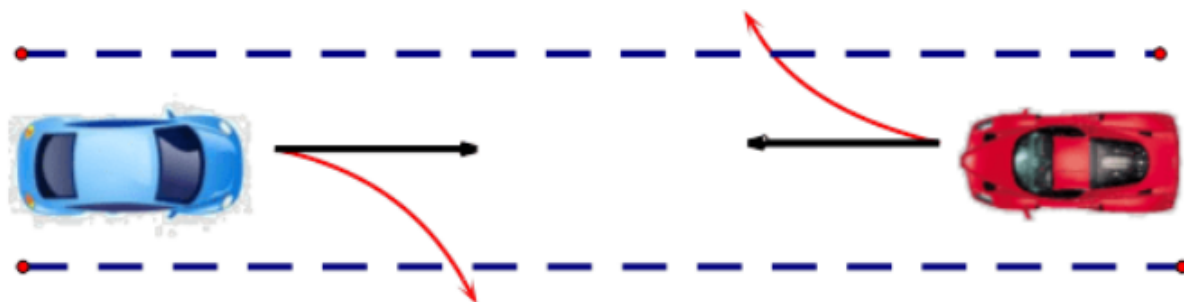
三字经：人之初，性本善

科学意义、理性光辉

- 一报还一报策略的胜利对人类和其他生物的合作行为的形成所具有的深刻含义是显而易见的。
- 《合作进化》一书中指出，一报还一报策略能导致社会各个领域的合作，包括在最无指望的环境中的合作。
 - 第一次世界大战中自发产生的“自己活，也让他人活”的原则。当时前线战壕里的军队约束自己不开枪杀伤人，只要对方也这么做。使这个原则能够实行的原因是，双方军队都已陷入困境数月，这给了他们相互适应的机会。
- 一报还一报的相互作用使得自然界即使没有智能也能产生合作关系。
 - 真菌从地下的石头中汲取养分，为海藻提供了食物，而海藻反过来又为真菌提供了光合作用；
 - 金蚁合欢树为一种蚂蚁提供了食物，而这种蚂蚁反过来又保护了该树；
 - 无花果树的花是黄蜂的食物，而黄蜂反过来又为无花果树传授花粉，将树种撒向四处。

练练思维：胆小鬼的博弈

罗素版的博弈是这样的：两个人在一条车道上相对着开车，每个人都可以随时打方向盘驶出车道，最先驶出车道的人就会被对方嘲笑为胆小鬼，而一直在车道上狂飙的人获胜。

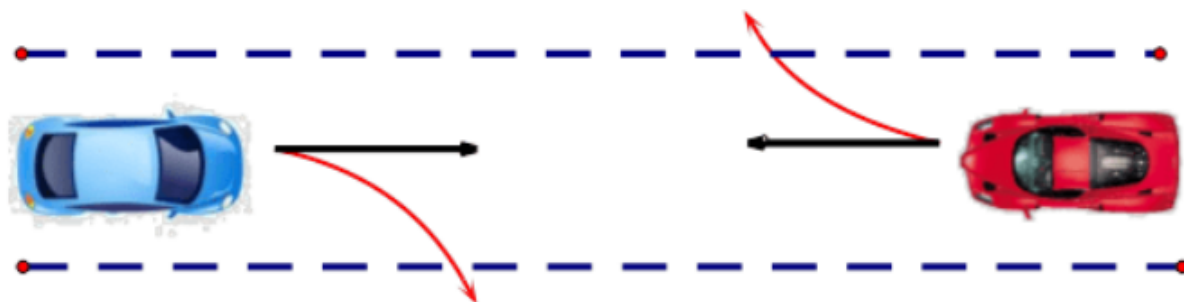


但是，如果两辆车都不驶出车道，最终两辆车就会撞在一起，同归于尽。

	 B示弱	 B死磕
 A示弱	2 2	1 3
 A死磕	3 1	-10 -10

练练思维：胆小鬼的博弈

罗素版的博弈是这样的：两个人在一条车道上相对着开车，每个人都可以随时打方向盘驶出车道，最先驶出车道的人就会被对方嘲笑为胆小鬼，而一直在车道上狂飙的人获胜。



但是，如果两辆车都不驶出车道，最终两辆车就会撞在一起，同归于尽。

A		 B示弱	 B死磕
	 A示弱	2 2	1 3
	 A死磕	3 1	-10 -10

纳什均衡点

双方都没有动力去改变自己的决策

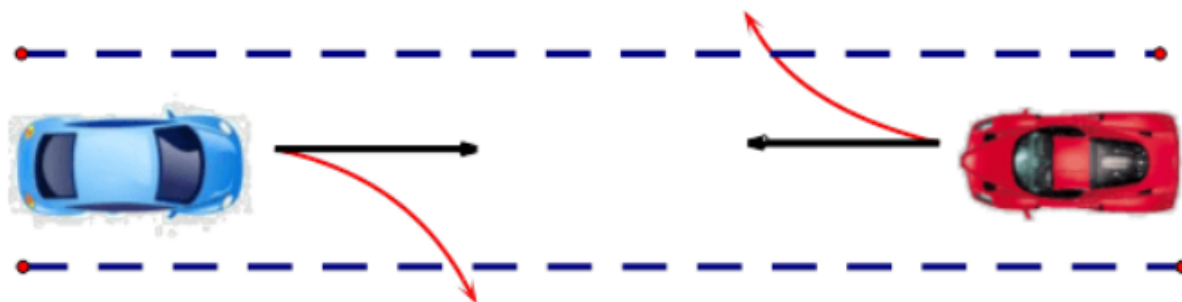
A		 B示弱	 B死磕
	 A示弱	2 2	1 3
	 A死磕	3 1	-10 -10

纳什均衡点

收益为1 3

练练思维：胆小鬼的博弈中，如何让对方成为胆小鬼？

罗素版的这个游戏是这样的：两个人在一条车道上相对着开车，每个人都可以随时打方向盘驶出车道，最先驶出车道的人就会被对方嘲笑为胆小鬼，而一直在车道上狂飙的人获胜。

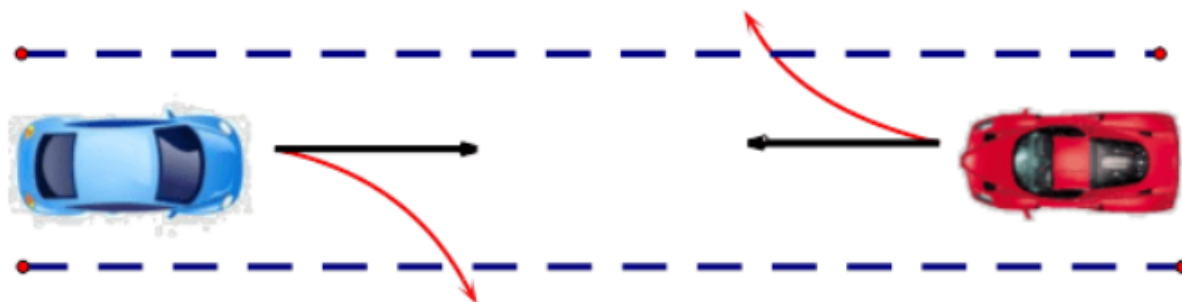


但是，如果两辆车都不驶出车道，最终两辆车就会撞在一起，同归于尽。



课后练习：用胆小鬼的博弈中解释古巴导弹危机中前苏联与美国的决策！

罗素版的游戏是这样的：两个人在一条车道上相对着开车，每个人都可以随时打方向盘驶出车道，最先驶出车道的人就会被对方嘲笑为胆小鬼，而一直在车道上狂飙的人获胜。



但是，如果两辆车都不驶出车道，最终两辆车就会撞在一起，同归于尽。



多参与者的博弈纳什均衡-三姬分金

●规则：

- 韩非子提出：有三个人A，B，C，按抽签顺序提议分金子方案，本题方案是依次A,B,C.
- 如果某人方案没有获得全体人员半数以上通过，不包括半数，则该提议人将被处死。

●假设：

- 聪明：如果获得最大利益。
- 理性：以自己的收益最大化为目标。
- 收益最大：金币最多，还要除掉对手。

●A是不是最危险的？

若只剩下B, C

- A已经出局
- B不管有什么提议, C都会除掉他。
 - 因为无论B有什么提议, C只要否定, B就不会获得半数以上通过。C就可以获得全部金币。
 - 因而, B清醒的认识到: 如果只剩下B和C, B必死。
 - B因而无条件支持A, 一定要保住A。
 - A也知道这个结果, 因而, A会给出这样的结论:

A的方案: A: 100, B: 0, C: 0

- 如果再多一个人M, M知道游戏规则, M先选, 因此, 顺序是M, A, B, C
- 分金方案是什么?



M的方案: $M:98, A:0, B:1, C:1$

- 如果把M杀掉了，A就会独得所有金币。
- M是具有先手优势，B和C没有决策力，但是低端人群，因而可以被拉拢，但收益很低。A是夹层，必杀。



谋略是否能你补实力的差距？

● 石头剪子布？



田忌赛马



田忌

VS



齐王



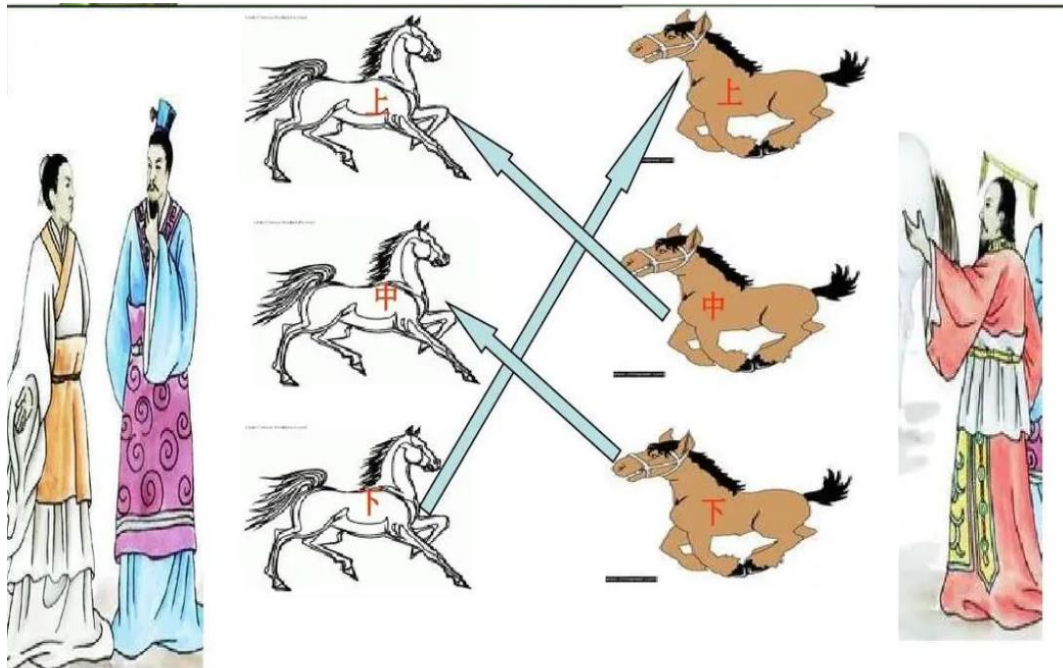
一等马



二等马



三等马



是否存在纳什均衡？

纳什定理：只要参与博弈的几方策略是有限的，那么就一定存在一种**平衡状态**，大家都会采用这种平衡策略，而没有单方面改变策略的动力。

这种平衡状态就是**纳什均衡**。

●纯策略下的纳什均衡

参与者最终走向了一个固定的策略集合，因而可以快速找到均衡点。

●混合策略的纳什均衡

无纯策略下，参与者为了让自己的收益最大化，也一定会以固定的频率，在几种策略之间切换；

参与博弈的每个人都会为每一种策略分配固定的频率，只有这样才能让自己的利益最大化。

混合策略的纳什均衡

- 石头剪子布？
- 石头剪刀布各占 $\frac{1}{3}$ 的频率
- 每人都以 $\frac{1}{3}$ 频率出石头剪子布，双方收益为0



谋略是否能你补实力的差距？

- 多轮情况下，田忌和齐王都是6中出马顺序，因此共36中策略。

得分		田忌					
		上中下	中下上	下上中	上下中	中上下	下中上
齐王	上中下	3, -3	1, -1	-1, 1	1, -1	1, -1	1, -1
	中下上	-1, 1	3, -3	1, -1	1, -1	1, -1	1, -1
	下上中	1, -1	-1, 1	3, -3	1, -1	1, -1	1, -1
	上下中	1, -1	1, -1	1, -1	3, -3	1, -1	-1, 1
	中上下	1, -1	1, -1	1, -1	-1, 1	3, -3	1, -1
	下中上	1, -1	1, -1	1, -1	1, -1	-1, 1	3, -3

那田忌就选下上中而获得胜利

得分		田忌					
		上中下	中下上	下上中	上下中	中上下	下中上
齐王	上中下	3, -3	1, -1	-1, 1	1, -1	1, -1	1, -1
	中下上	-1, 1	3, -3	1, -1	1, -1	1, -1	1, -1
	下上中	1, -1	-1, 1	3, -3	1, -1	1, -1	1, -1
	上下中	1, -1	1, -1	1, -1	3, -3	1, -1	-1, 1
	中上下	1, -1	1, -1	1, -1	-1, 1	3, -3	1, -1
	下中上	1, -1	1, -1	1, -1	1, -1	-1, 1	3, -3

齐王胜
齐王都有5/6的可能会获胜

计算机科学中的经典算法问题

Great Ideas in Computer Science

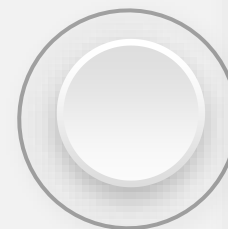
可计算问题

计算思维

计算思维是一种重要的思维方式，有了计算思维就会知道如何将一个问题进行抽象，变为让计算机可“理解”并可计算的模型。



用贝叶斯算法
讲《狼来了》的故事



PART TWO



Standing on Shoulders of Giants



给计算机“讲”《狼来了的故事》

AI学推理

狼来了

伊索寓言“狼来了的故事”讲的是一个小孩每天到山上放羊，山里有狼出没。第一天，他在山上喊“狼来了！狼来了！”，山下的村民闻声便去打狼，发现狼没有来；第二天仍是如此；第三天，狼真的来了，可无论小孩怎么喊叫，也没有人来救他，因为前两次他说了谎，人们不再相信他了。

这个故事告诉我们，不要撒谎！



现在，我们用**计算思维**来给计算机讲解这个故事

目的：让计算机通过计算的方式，学会了故事中的道理，也不去撒谎。



用贝叶斯给AI讲道理

AI学推理

在山的那边，海的那边，有一个小牧童…

记事件**B**为“**小牧童可信**”。

设村民对这个小牧童的印象为

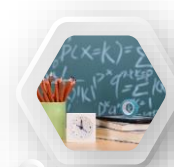
$$P(B) = 0.8, P(\bar{B}) = 0.2$$

先验概率

记事件**A**为“**小孩说谎**”

$$P(A|B) = 0.1, P(A|\bar{B}) = 0.5$$

似然概率



用贝叶斯给AI讲道理

故事继续

设村民对这个小牧童的印象为

$$P(B)=0.8, P(\bar{B})=0.2$$

先验概率

记事件A为“小孩说谎”

$$P(A|B)=0.1, P(A|\bar{B})=0.5$$

似然概率

如果，小孩撒了谎，村民对他的信任度变成了多少？

$$P(B|A)$$

后先验概率



用贝叶斯给AI讲道理

故事继续

设村民对这个小牧童的印象为

$$P(B) = 0.8, P(\bar{B}) = 0.2$$

记事件A为“小孩说谎”

$$P(A|B) = 0.1, P(A|\bar{B}) = 0.5$$

用贝叶斯公式求 $P(B|A)$

$$P(B|A) = \frac{P(B)P(A|B)}{P(B)P(A|B) + P(\bar{B})P(A|\bar{B})}$$

设阈值=0.2



用贝叶斯“学”道理

第一次撒谎后

第一次村民上山打狼，发现狼没有来，即小孩说了谎（A）。村民根据这个信息，对小孩的可信程度改变为（用贝叶斯公式）

$$\begin{aligned}P(B|A) &= \frac{P(B)P(A|B)}{P(B)P(A|B) + P(\bar{B})P(A|\bar{B})} \\&= \frac{0.8 \times 0.1}{0.8 \times 0.1 + 0.2 \times 0.5} = 0.444\end{aligned}$$

这表明村民上了一次当后，对这个小孩的可信程度由原来的0.8调整为**0.444**，也就是

$$P(B) = 0.444, P(\bar{B}) = 0.556$$



用贝叶斯“学”道理

第二次撒谎后

在此基础上，我们再用一次贝叶斯公式计算 $P(B|A)$

亦即这个小孩第二次说谎后，村民对他的可信程度改变为：

$$P(B|A) = \frac{0.444 \times 0.1}{0.444 \times 0.1 + 0.556 \times 0.5} = 0.138$$

这表明村民们经过两次上当，对这个小孩的可信程度已经从0.8下降到**0.138**，如此低的可信程度，村民听到第三次呼叫怎么再会上山打狼呢？

AI用数学公式，通过计算，“学会”了推理，懂得了道理！



用贝叶斯推理

烽火戏诸侯

练一练

司马迁《史记·周本纪》：
“褒姒不好笑，幽王欲其笑
万方，故不笑。

幽王为烽燧大鼓，有寇至则
举烽火。诸侯悉至，至而无
寇，褒姒乃大笑。”

西周烽燧

公元前779年，西周都城镇京，周幽王为博得美人一笑，利用烽火来戏耍诸侯。

哈哈，美人终于笑了。

烽燧

也叫“烽火台”，白天燃烟叫“烽”，夜间点火叫“燧”，是一种信息传递方式。烽火又叫“狼烟”，因为加了狼屎，就有浓烟。



烽火戏诸侯



计算机科学中的经典算法问题

Great Ideas in Computer Science

猜数问题

计算思维/ Computational Thinking

在一张白纸上，如何在最短时间里“得到”16个格子？

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

AI 如何算的快？

折半查找

Standing on Shoulders of Giants

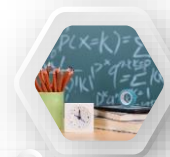


AI如何算的更快？

猜数游戏

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

小明心里想了一个数，这个数的范围是1~16，大家可以通过提问来猜这个数，小明的回答只能是**YES**或**NO**，如何快速猜到他心里想的这个数。



AI如何算的更快？

猜数游戏

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



[码上 \(bupt.edu.cn\)](https://bupt.edu.cn)
<https://ezcoding.bupt.edu.cn>



AI如何算的更快？

问问AI大模型



问答论坛



1对1辅导



教学管理



用户中心



使用帮助



关于我们



小明心里想了一个数，这个数的范...

C++2023211301Q-4070 · 提问于 2小时前

历史提问

公开回答

智能回答 · I

智能回答 · II

互动 (0)

多回答对比

知识提示

求助

由大模型给出的智能解答

特别提醒：码上采用大模型技术生成答案，不能保证完全正确，因此答案仅供参考
如有错误，请点击“此答案有问题”，并可向老师求助

问题回答

为了快速猜出小明心里想的数字，可以采用二分查找法 (Binary Search) 的思路。具体步骤如下：
1. 设定数字范围的最小值 min 为0，最大值 max 为100。
2. 计算当前范围的中间值 mid，即 $(\min + \max) / 2$ 。
3. 提问小明：“你心里想的数字是否大于或等于 mid？”
4. 如果小明回答YES，则将 min 更新为 mid + 1；如果回答NO，则将 max 更新为 mid - 1。
5. 重复步骤2至4，直到 min 大于 max，此时 mid 就是小明心里想的数字。
这种方法每一次提问都会将可能的数字范围减半，从而最多只需要提问7次（因为 $2^7 = 128 > 100$ ），就可以猜出一个0到100之间的数字。
这种二分查找的方式不仅适用于猜数字游戏，还广泛应用于计算机科学中的搜索算法。你可以尝试将这种思想应用到其他问题中，比如在一个有序数组中查找特定元素。
如果您有关于实现二分查找的具体代码问题，或是在理解这个过程中有任何疑问，都可以随时提问，我会提供进一步的帮助。

你可以尝试继续和小码聊聊...

Tips: 首次提问后不可对代码进行修改，如需修改请点击左侧的重新提问按钮

Powered by 北京邮电大学计算机学院/网络与交换技术全国重点实验室网络智能研究中心、科大讯飞股份有限公司、和讯数字科技有限公司

Since 2022, EZCoding 团队



AI如何算的更快？

问问AI大模型

C:\Users\yxyxy\Desktop\中国科技馆讲座\小明猜数-大模型2.exe

请在心中想一个1到16之间的数，我会尝试猜出它。
如果我猜的数比你想要的大，请输入NO；
如果我猜的数比你想要的小，请输入YES；
如果猜对了，请输入CORRECT。
你心里想的数是不是 8 ？ (YES/NO/CORRECT)：

用户中心



使用帮助



关于我们

```
#include <iostream>
#include <string>
```

```
int main() {
    int min = 0;
    int max = 100;
    int mid;
    std::string response;
```

```
    std::cout << "请在心中想一个0到100之间的数，我会尝试猜出它。";
```

```
    std::cout << "如果我猜的数比你想要的大，请输入NO； 如果比你想要的小，请输入YES； 如果猜对了，请输入CORRECT。";
```



你可以尝试继续和小码聊...

Tips: 首次提问后不可对代码进行修改，如需修改请点击左侧的重新提问按钮

历史提问

公开回答

多回答对比

知识点

求助

回复于 2小时前

色，而用户（假设为小明）需要在心中想一个数，然后对程序的猜测进行回应。



Powered by 北京邮电大学计算机学院/网络与交换技术全国重点实验室网络智能研究中心、科大讯飞股份有限公司、讯数数字科技有限公司
Since 2022, EZCoding 团队

AI如何算的更快？

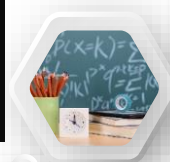
问问AI大模型

C:\Users\yyxyy\Desktop\中国科技馆讲座\小明猜数-大模型2.exe

请在心中想一个1到16之间的数，我会尝试猜出它。
如果我猜的数比你想要的大，请输入NO；
如果我猜的数比你想要的小，请输入YES；
如果猜对了，请输入CORRECT。
你心里想的数是不是 8 ？ (YES/NO/CORRECT) :

C:\Users\yyxyy\Desktop\中国科技馆讲座\小明猜数-大模型2.exe

请在心中想一个1到16之间的数，我会尝试猜出它。
如果我猜的数比你想要的大，请输入NO；
如果我猜的数比你想要的小，请输入YES；
如果猜对了，请输入CORRECT。
你心里想的数是不是 8 ？ (YES/NO/CORRECT) : NO
你心里想的数是不是 4 ？ (YES/NO/CORRECT) : NO
你心里想的数是不是 2 ？ (YES/NO/CORRECT) : CORRECT



AI如何算的更快？

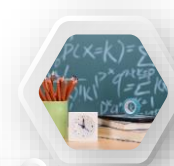
比一比

$O(n)$

假设：查找一个元素的时间是 1 毫秒

查找数量	顺序查找	折半查找
100 个元素	100 毫秒	7 毫秒
10,000 个元素	10 秒	14 毫秒
1000,000,000 个元素	11 天	30 毫秒

$O(\log n)$



计算机科学中的经典算法问题

Great Ideas in Computer Science

国王的婚姻

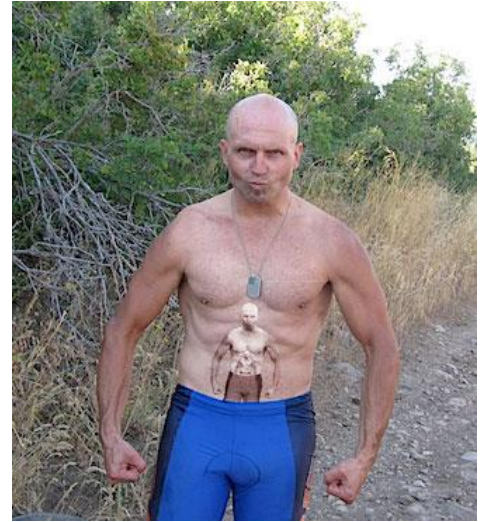
国王的婚姻

-- 洪加威

- 从前一位喜欢数学的国王向一位聪明漂亮的公主求婚。公主向他提出了一个问题：求出48 770 428 433 377 171的一个真因子。如果国王能在一天之内找到答案，她就接受他的求婚。
- 国王回去后立即开始逐个数地进行计算，他从早到晚，共算了三万多个数，最终还是没有结果。
- 国王向公主求情，公主将答案相告：223 092 827是它的一个真因子。国王很快就验证了这个数确能除尽48 770 428 433 377 171。
- 公主说：“我再给你一次机会。如果你仍然没有做到，那么你将是婚礼上的一位宾客。”

- ❑ 国王立即回国，并向时任宰相的大数学家求教，大数学家在仔细地思考后认为这个数为17位，则最小的一个真因子不会超过9位。
- ❑ 他给国王出了一个主意：按自然数的顺序给全国的老百姓每人编一个号发下去，等公主给出数目后，立即将它们通报全国，让每个老百姓用自己的编号去除这个数，除尽了立即上报，赏金万两。
- ❑ 接着，公主给出了新的17位数。
- ❑ 国王采用新方法之后很快找到了正确的答案。

● 并行处理



Recursion = :^)

谢谢聆听！