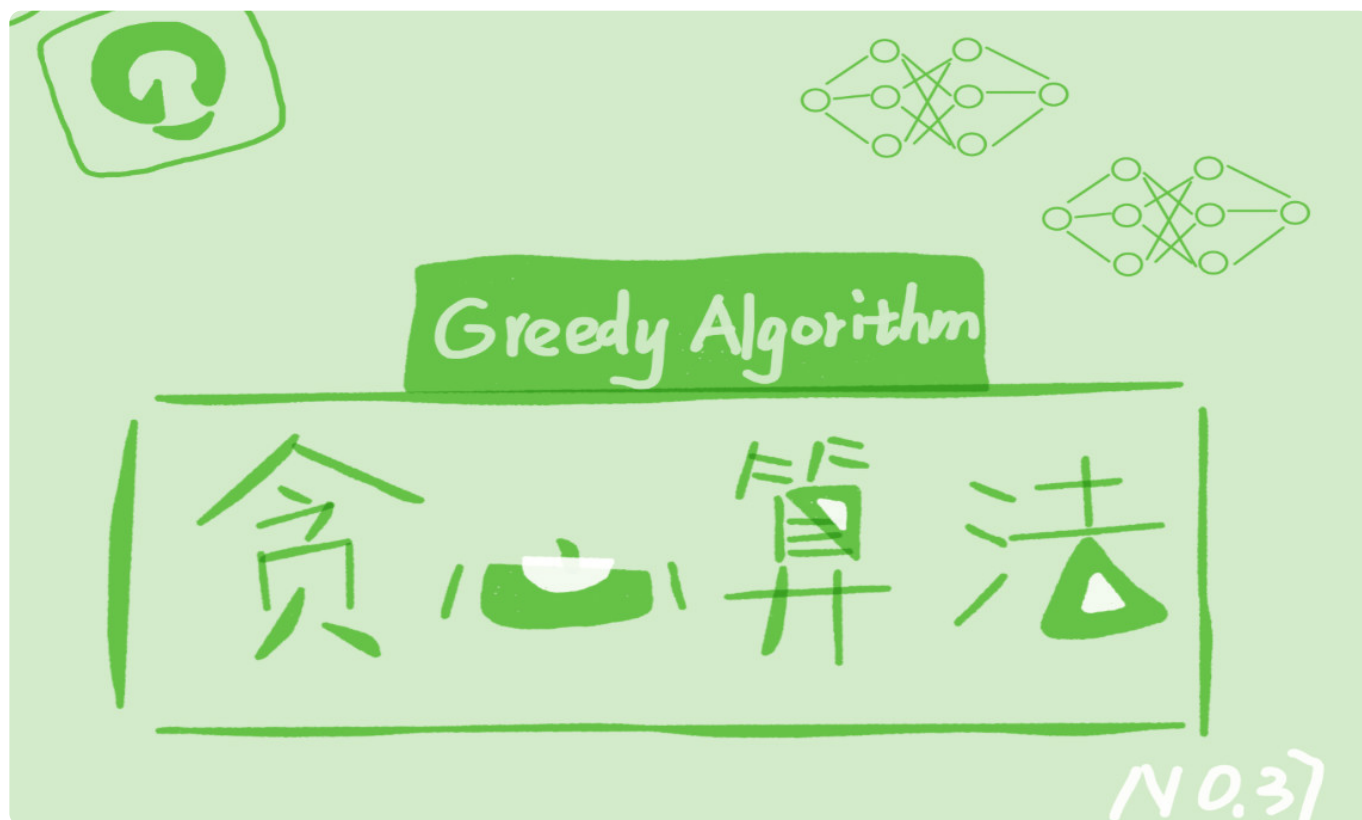


37 | 贪心算法：如何用贪心算法实现Huffman压缩编码？

2018-12-17 王争

数据结构与算法之美

[进入课程 >](#)



讲述：修阳

时长 15:40 大小 14.36M



基础的数据结构和算法我们基本上学完了，接下来几节，我会讲几种更加基本的算法。它们分别是贪心算法、分治算法、回溯算法、动态规划。更加确切地说，它们应该是算法思想，并不是具体的算法，常用来指导我们设计具体的算法和编码等。

贪心、分治、回溯、动态规划这 4 个算法思想，原理解释起来都很简单，但是要真正掌握且灵活应用，并不是件容易的事情。所以，接下来的这 4 个算法思想的讲解，我依旧不会长篇大论地去讲理论，而是结合具体的问题，让你自己感受这些算法是怎么工作的，是如何解决问题的，带你在问题中体会这些算法的本质。我觉得，这比单纯记忆原理和定义要更有价值。

今天，我们先来学习一下贪心算法（greedy algorithm）。贪心算法有很多经典的应用，比如霍夫曼编码（Huffman Coding）、Prim 和 Kruskal 最小生成树算法、还有 Dijkstra 单源最短路径算法。最小生成树算法和最短路径算法我们后面会讲到，所以我们今天讲下霍夫曼编码，看看它是如何利用贪心算法来实现对数据压缩编码，有效节省数据存储空间的。

如何理解“贪心算法”？

关于贪心算法，我们先看一个例子。

假设我们有一个可以容纳 100kg 物品的背包，可以装各种物品。我们有以下 5 种豆子，每种豆子的总量和总价值都各不相同。为了让背包中所装物品的总价值最大，我们如何选择在背包中装哪些豆子？每种豆子又该装多少呢？

物品	总量 (kg)	总价值 (元)
黄豆	100	100
绿豆	30	90
红豆	60	120
黑豆	20	80
青豆	50	75

实际上，这个问题很简单，我估计你一下子就能想出来，没错，我们只要先算一算每个物品的单价，按照单价由高到低依次来装就好了。单价从高到低排列，依次是：黑豆、绿豆、红豆、青豆、黄豆，所以，我们可以往背包里装 20kg 黑豆、30kg 绿豆、50kg 红豆。

这个问题的解决思路显而易见，它本质上借助的就是贪心算法。结合这个例子，我总结一下贪心算法解决问题的步骤，我们一起来看看。

第一步，当我们看到这类问题的时候，首先要联想到贪心算法：针对一组数据，我们定义了限制值和期望值，希望从中选出几个数据，在满足限制值的情况下，期望值最大。

类比到刚刚的例子，限制值就是重量不能超过 100kg，期望值就是物品的总价值。这组数据就是 5 种豆子。我们从中选出一部分，满足重量不超过 100kg，并且总价值最大。

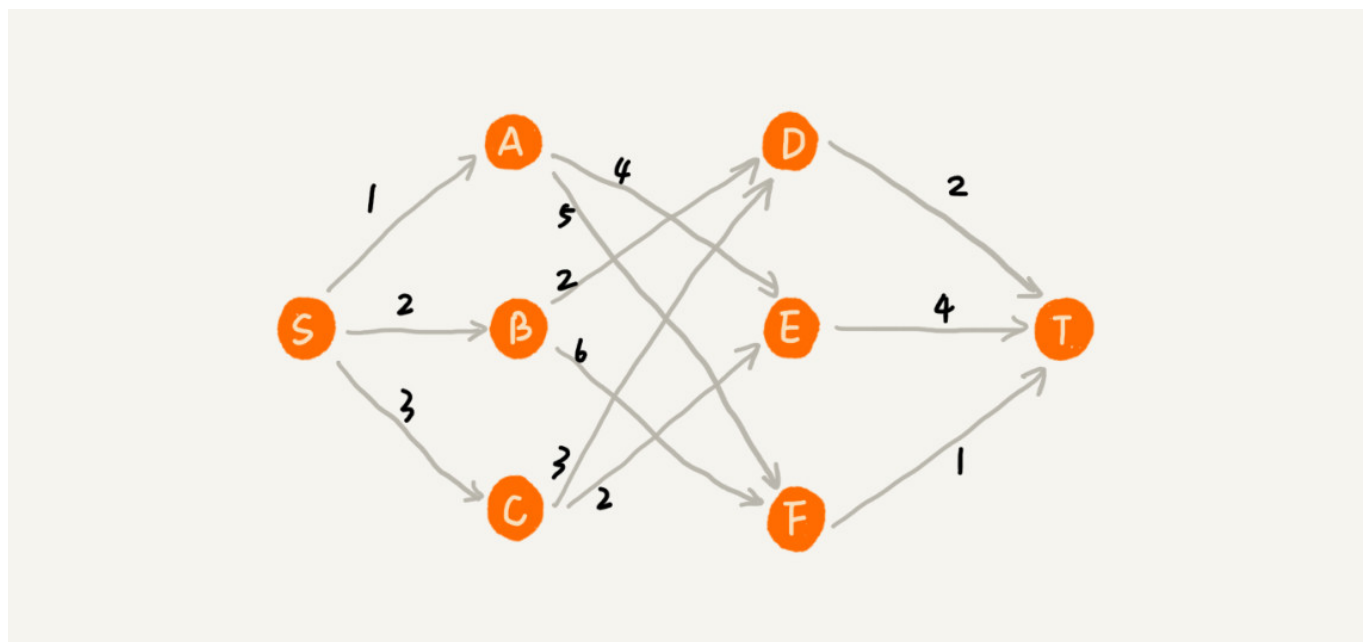
第二步，我们尝试看下这个问题是否可以用贪心算法解决：每次选择当前情况下，在对限制值同等贡献量的情况下，对期望值贡献最大的数据。

类比到刚刚的例子，我们每次都从剩下的豆子里面，选择单价最高的，也就是重量相同的情况下，对价值贡献最大的豆子。

第三步，我们举几个例子看下贪心算法产生的结果是否是最优的。大部分情况下，举几个例子验证一下就可以了。严格地证明贪心算法的正确性，是非常复杂的，需要涉及比较多的数学推理。而且，从实践的角度来说，大部分能用贪心算法解决的问题，贪心算法的正确性都是显而易见的，也不需要严格的数学推导证明。

实际上，用贪心算法解决问题的思路，并不总能给出最优解。

我来举一个例子。在一个有权图中，我们从顶点 S 开始，找一条到顶点 T 的最短路径（路径中边的权值和最小）。贪心算法的解决思路是，每次都选择一条跟当前顶点相连的权最小的边，直到找到顶点 T。按照这种思路，我们求出的最短路径是 S->A->E->T，路径长度是 $1+4+4=9$ 。



但是，这种贪心的选择方式，最终求的路径并不是最短路径，因为路径 S->B->D->T 才是最短路径，因为这条路径的长度是 $2+2+2=6$ 。为什么贪心算法在这个问题上不工作了？

在这个问题上，贪心算法不工作的主要原因是，前面的选择，会影响后面的选择。如果我们第一步从顶点 S 走到顶点 A，那接下来面对的顶点和边，跟第一步从顶点 S 走到顶点 B，是完全不同的。所以，即便我们第一步选择最优的走法（边最短），但有可能因为这一步选择，导致后面每一步的选择都很糟糕，最终也就无缘全局最优解了。

贪心算法实战分析

对于贪心算法，你是不是还有点懵？如果死抠理论的话，确实很难理解透彻。掌握贪心算法的关键是多练习。只要多练习几道题，自然就有感觉了。所以，我带着你分析几个具体的例子，帮助你深入理解贪心算法。

1. 分糖果

我们有 m 个糖果和 n 个孩子。我们现在要把糖果分给这些孩子吃，但是糖果少，孩子多（ $m < n$ ），所以糖果只能分配给一部分孩子。

每个糖果的大小不等，这 m 个糖果的大小分别是 $s_1, s_2, s_3, \dots, s_m$ 。除此之外，每个孩子对糖果大小的需求也是不一样的，只有糖果的大小大于等于孩子的对糖果大小的需求的时候，孩子才得到满足。假设这 n 个孩子对糖果大小的需求分别是 $g_1, g_2, g_3, \dots, g_n$ 。

我的问题是，如何分配糖果，能尽可能满足最多数量的孩子？

我们可以把这个问题抽象成，从 n 个孩子中，抽取一部分孩子分配糖果，让满足的孩子的个数（期望值）是最大的。这个问题的限制值就是糖果个数 m 。

我们现在来看看如何用贪心算法来解决。对于一个孩子来说，如果小的糖果可以满足，我们就没必要用更大的糖果，这样更大的就可以留给其他对糖果大小需求更大的孩子。另一方面，对糖果大小需求小的孩子更容易被满足，所以，我们可以从需求小的孩子开始分配糖果。因为满足一个需求大的孩子跟满足一个需求小的孩子，对我们期望值的贡献是一样的。

我们每次从剩下的孩子中，找出对糖果大小需求最小的，然后发给他剩下的糖果中能满足他的最小的糖果，这样得到的分配方案，也就是满足的孩子个数最多的方案。

2. 钱币找零

这个问题在我们的日常生活中更加普遍。假设我们有 1 元、2 元、5 元、10 元、20 元、50 元、100 元这些面额的纸币，它们的张数分别是 c_1 、 c_2 、 c_5 、 c_{10} 、 c_{20} 、 c_{50} 、 c_{100} 。我们现在要用这些钱来支付 K 元，最少要用多少张纸币呢？

在生活中，我们肯定是先用面值最大的来支付，如果不够，就继续用更小一点面值的，以此类推，最后剩下的用 1 元来补齐。

在贡献相同期望值（纸币数目）的情况下，我们希望多贡献点金额，这样就可以让纸币数更少，这就是一种贪心算法的解决思路。直觉告诉我们，这种处理方法就是最好的。实际上，要严谨地证明这种贪心算法的正确性，需要比较复杂的、有技巧的数学推导，我不建议你花太多时间在上面，不过如果感兴趣的话，可以自己去研究下。

3. 区间覆盖

假设我们有 n 个区间，区间的起始端点和结束端点分别是 $[l_1, r_1]$ ， $[l_2, r_2]$ ， $[l_3, r_3]$ ，.....， $[l_n, r_n]$ 。我们从这 n 个区间中选出一部分区间，这部分区间满足两两不相交（端点相交的情况不算相交），最多能选出多少个区间呢？

区间: $[6, 8]$ $[2, 4]$ $[3, 5]$ $[1, 5]$ $[5, 9]$ $[8, 10]$

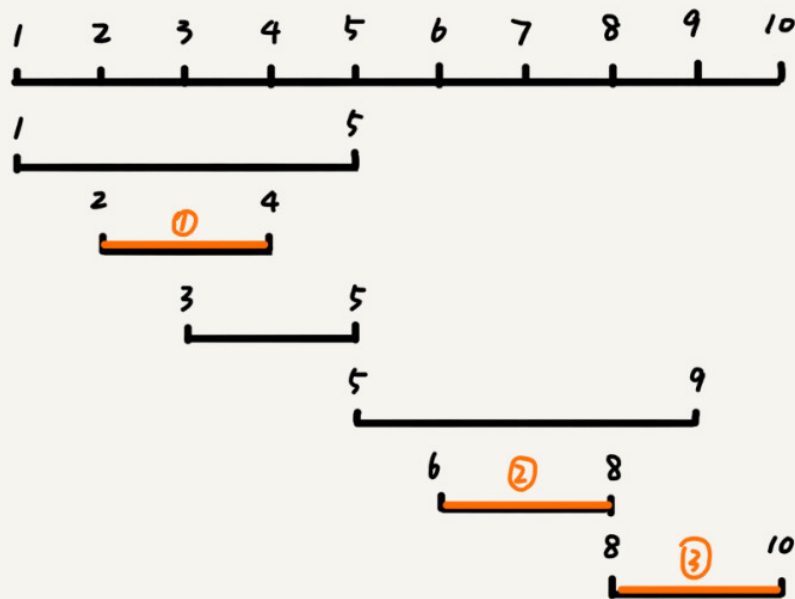
不相交区间: $[2, 4]$ $[6, 8]$ $[8, 10]$

这个问题的处理思路稍微不是那么好懂，不过，我建议你最好能弄懂，因为这个处理思想在很多贪心算法问题中都有用到，比如任务调度、教师排课等等问题。

这个问题的解决思路是这样的：我们假设这 n 个区间中最左端点是 l_{\min} ，最右端点是 r_{\max} 。这个问题就相当于，我们选择几个不相交的区间，从左到右将 $[l_{\min}, r_{\max}]$ 覆盖上。我们按照起始端点从小到大的顺序对这 n 个区间排序。

我们每次选择的时候，左端点跟前面的已经覆盖的区间不重合的，右端点又尽量小的，这样可以让剩下的未覆盖区间尽可能的大，就可以放置更多的区间。这实际上就是一种贪心的选

择方法。




解答开篇

今天的内容就讲完了，我们现在来看开篇的问题，如何用贪心算法实现霍夫曼编码？

假设我有一个包含 1000 个字符的文件，每个字符占 1 个 byte (1byte=8bits)，□存储这 1000 个字符就一共需要 8000bits，那有没有更加节省空间的存储方式呢？

假设我们通过统计分析发现，这 1000 个字符中只包含 6 种不同字符，假设它们分别是 a、b、c、d、e、f。而 3 个二进制位 (bit) 就可以表示 8 个不同的字符，所以，为了尽量减少存储空间，每个字符我们用 3 个二进制位来表示。那存储这 1000 个字符只需要 3000bits 就可以了，比原来的存储方式节省了很多空间。不过，还有没有更加节省空间的存储方式呢？

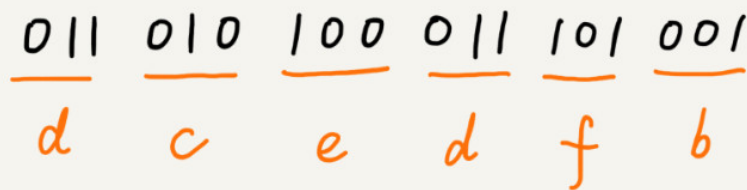
 复制代码

```
1 a(000)、b(001)、c(010)、d(011)、e(100)、f(101)
```

霍夫曼编码就要登场了。霍夫曼编码是一种十分有效的编码方法，广泛用于数据压缩中，其压缩率通常在 20% ~ 90% 之间。

霍夫曼编码不仅会考察文本中有多少个不同字符，还会考察每个字符出现的频率，根据频率的不同，选择不同长度的编码。霍夫曼编码试图用这种不等长的编码方法，来进一步增加压缩的效率。如何给不同频率的字符选择不同长度的编码呢？根据贪心的思想，我们可以把出现频率比较多的字符，用稍微短一些的编码；出现频率比较少的字符，用稍微长一些的编码。

对于等长的编码来说，我们解压缩起来很简单。比如刚才那个例子中，我们用 3 个 bit 表示一个字符。在解压缩的时候，我们每次从文本中读取 3 位二进制码，然后翻译成对应的字符。但是，霍夫曼编码是不等长的，每次应该读取 1 位还是 2 位、3 位等等来解压缩呢？这个问题就导致霍夫曼编码解压缩起来比较复杂。为了避免解压缩过程中的歧义，霍夫曼编码要求各个字符的编码之间，不会出现某个编码是另一个编码前缀的情况。



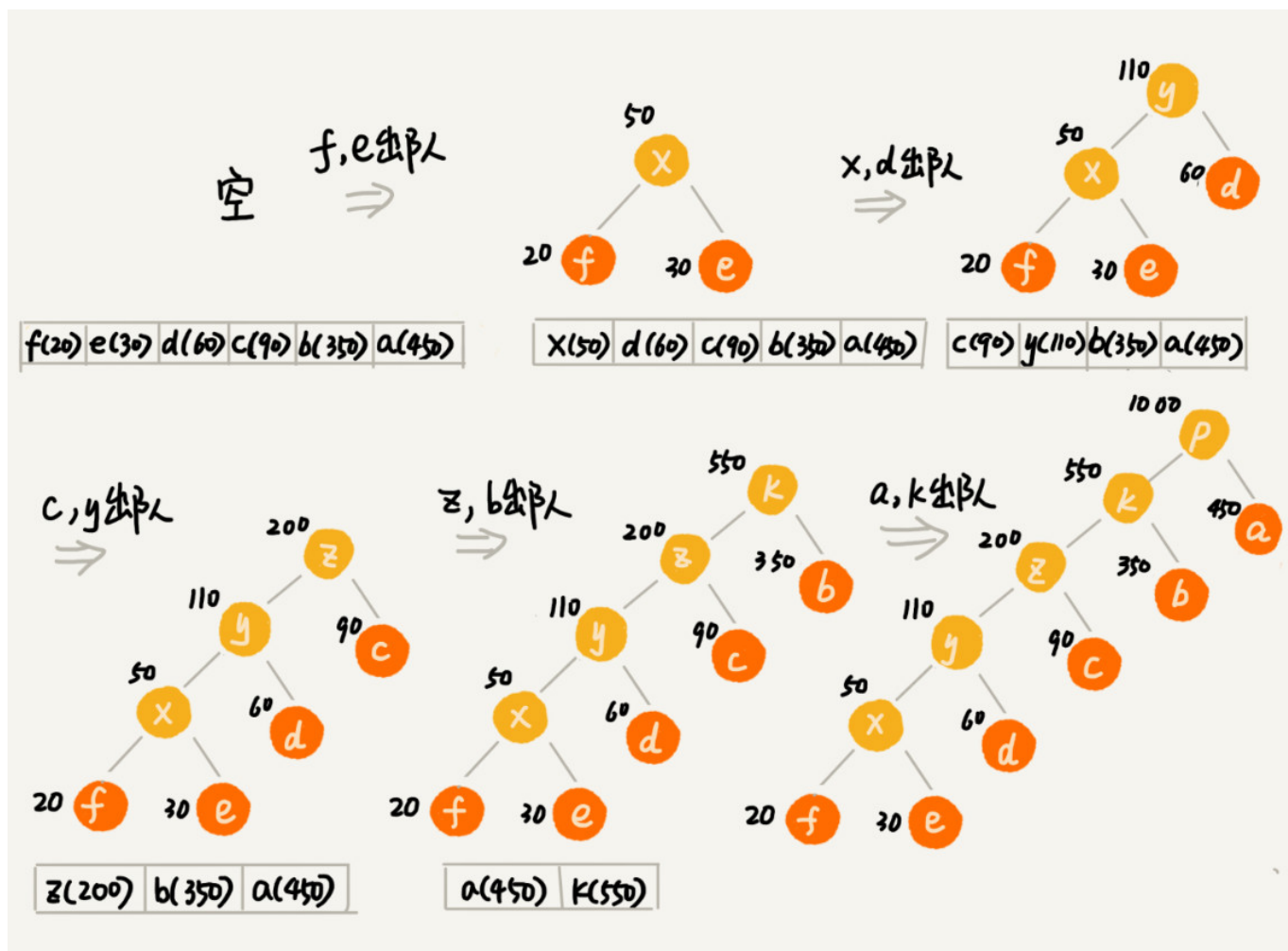
011	010	100	011	101	001
d	c	e	d	f	b

假设这 6 个字符出现的频率从高到低依次是 a、b、c、d、e、f。我们把它编码下面这个样子，任何一个字符的编码都不是另一个的前缀，在解压缩的时候，我们每次会读取尽可能长的可解压的二进制串，所以在解压缩的时候也不会歧义。经过这种编码压缩之后，这 1000 个字符只需要 2100bits 就可以了。

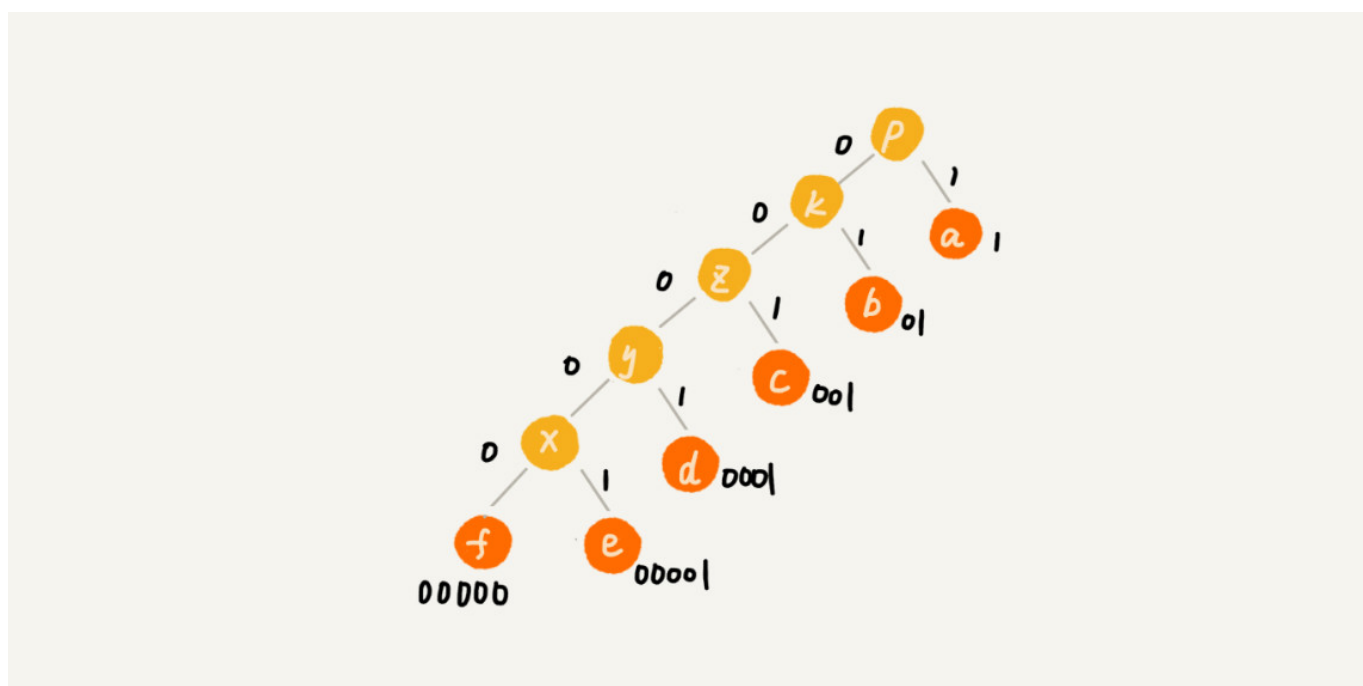
字符	出现频率	编码	总=进制位数
a	450	1	450
b	350	01	700
c	90	001	270
d	60	0001	240
e	30	00001	150
f	20	00000	100

尽管霍夫曼编码的思想并不难理解，但是如何根据字符出现频率的不同，给不同的字符进行不同长度的编码呢？这里的处理稍微有些技巧。

我们把每个字符看作一个节点，并且辅带着把频率放到优先级队列中。我们从队列中取出频率最小的两个节点 A、B，然后新建一个节点 C，把频率设置为两个节点的频率之和，并把这个新节点 C 作为节点 A、B 的父节点。最后再把 C 节点放入到优先级队列中。重复这个过程，直到队列中没有数据。



现在，我们给每一条边加上画一个权值，指向左子节点的边我们统统标记为 0，指向右子节点的边，我们统统标记为 1，那从根节点到叶节点的路径就是叶节点对应字符的霍夫曼编码。



内容小结

今天我们学习了贪心算法。

实际上，贪心算法适用的场景比较有限。这种算法思想更多的是指导设计基础算法。比如最小生成树算法、单源最短路径算法，这些算法都用到了贪心算法。**从我个人的学习经验来讲，不要刻意去记忆贪心算法的原理，多练习才是最有效的学习方法。**

贪心算法的最难的一块是如何将要解决的问题抽象成贪心算法模型，只要这一步搞定之后，贪心算法的编码一般都很简单。贪心算法解决问题的正确性虽然很多时候都看起来是显而易见的，但是要严谨地证明算法能够得到最优解，并不是件容易的事。所以，很多时候，我们只需要多举几个例子，看一下贪心算法的解决方案是否真的能得到最优解就可以了。

课后思考

1. 在一个非负整数 a 中，我们希望从中移除 k 个数字，让剩下的数字值最小，如何选择移除哪 k 个数字呢？
2. 假设有 n 个人等待被服务，但是服务窗口只有一个，每个人需要被服务的时间长度是不同的，如何安排被服务的先后顺序，才能让这 n 个人总的等待时间最短？

欢迎留言和我分享，也欢迎点击“[请朋友读](#)”，把今天的内容分享给你的好友，和他一起讨论、学习。



数据结构与算法之美

为工程师量身打造的数据结构与算法私教课

王争

前 Google 工程师



新版升级：点击「[请朋友读](#)」，10位好友免费读，邀请订阅更有**现金**奖励。

上一篇 36 | AC自动机：如何用多模式串匹配实现敏感词过滤功能？

下一篇 38 | 分治算法：谈一谈大规模计算框架MapReduce中的分治思想

精选留言 (76)

写留言



cirno

2018-12-17

87

1、由最高位开始，比较低一位数字，如高位大，移除，若高位小，则向右移一位继续比较两个数字，直到高位大于低位则移除，循环k次，如：

4556847594546移除5位-》455647594546-》45547594546-》4547594546-》
4447594546-》444594546

...

展开



开心小毛

2018-12-31

36

找零问题不能用贪婪算法，即使有面值为一元的币值也不行：考虑币值为100，99和1的币种，每种各一百张，找396元。

动态规划可求出四张99元，但贪心算法解出需三张一百和96张一元。

展开

作者回复: 是的



Jalyn

2018-12-17

18

想知道目前没掉队的有多少 哈哈

展开

作者回复: 慢慢学 不着急



feifei

2018-12-20

14

1, 在一个非负整数 a 中, 我们希望从中移除 k 个数字, 让剩下的数字值最小, 如何选择移除哪 k 个数字呢?

对于此题, 我的求解思路是每次选择数据的最高位的数据值进行移除, 这样我们每次选择的移除的数值都是最大的, 剩下的数值也是最小的。

比如, 数据5892, 将数据拆成5000,800,90,2, 然后使用大顶堆来进行存储, 然后每次移...

展开



Alexis何春...

2019-01-12

11

留言里feifei说的两种解决思路都是错的, 给的链接也失效了.... 老师可以回复一下防止误导后来的同学呀!

以及没有看出来霍夫曼算法和贪心算法有什么联系, 求详细讲解



luxinfeng

2018-12-17

9

老师能详细讲讲区间覆盖这个问题的选择过程么?

展开



您的好...

2018-12-17

6

给大家提个醒, 货币找零问题如果没有C1货币的话得用动态规划去解, 如果出现{C2, C7, C10}货币找零11块的时候使用greedy就会出现找不开的情况。。。有C1就不会出现找不开的情况且多个C1可以构成任何面值, 所以这种情况下使用greedy是对的!

(leetcode322题调了一下午的路过。。。)

展开



qinggeouy...

2019-01-19

5

1、在一个非负整数 a 中, 希望从中移除 k 个数字, 让剩下的数字值最小, 如何选择移除哪 k 个数字呢?

整数 a , 由若干位数字组成, 移除 k 个数字后的值最小。从高位开始移除: 移除高位数字比它低位数字大的那个; K 次循环。...

展开



睡痴儿

2019-01-22

4

第一个题，可以反着来想。给定另一个数组，怎么从中原本的中挑出 $n-k$ 个，使其值最小。首先第一位必须要是最小的一个，但是因为有序，所以只能是从0到 $n-1-k$ 个中挑一个最小的，下标为 m 。以后依次类推，从 m 到 $n-k$ 中挑一个最小的。如果有相等的情况，以下标小的为准。

第二个，就是从小到大排序即可。

展开



小美

2018-12-17

4

老师 区间覆盖的问题，(1-5) 和 (2-4) 中为什么选(2-4) 方便老师解释下吗，贪心不能全局最优 用贪心 如何在这个问题上全局最优呢

展开

作者回复: 24让剩下的没有被覆盖的区间最大 如果你选15 那我完全可以用24替代15 这样子 只有更好 不会更差



蒋礼锐

2018-12-17

3

第一个问题不知道0可不可以被保留在最高位，如果可以的话那么应该每次移除该整数的非零最高位，比如909090， k 为2的话，最小的值应该是0090，如果0不能在最高位，就贪心算法就不能得到最优解了，跟之前的加权图一样，因为决策会相互影响。

第二个问题假设5个人，时间分别是1-5-3-4-2分钟，等待的时间是每个人等待时间的总...

展开



发飙的蜗牛

2019-02-28

2

个人觉得分糖果不能使用贪心算法，如果使用可以加个条件一个孩子只能分一个糖果，因为可能存在孩子的满意度大于最小的糖果，但是最小的两个的和刚好满足。这个情况贪心就无法满足了。

作者回复: 哈哈，你很细心，是的，一个孩子最多一个糖果



www.xnsms...

2019-01-06

👍 2

霍夫曼编码，用一个树来避免某个字符的编码是另一个字符编码的前缀，真的是好巧妙



Jerry银银

2018-12-19

👍 2

打个卡，我还在

展开 ∨



白若

2018-12-17

👍 2

思考题(自己的想法，不知道对不对，希望老师能给我评论。)

1.从前往后两两比较，若前数大于等于后数，选择移除。如果一轮下来没达到k个数，就移除最后的m个数，m为k-已选出个数。

2.时间越短位置越靠前。

展开 ∨



Neil

2019-03-22

👍 1

通过局部最优，达到全局最优

展开 ∨



乐凡

2019-03-21

👍 1

老师，我觉得那个霍夫曼编码可能有点缺陷，就是不同字符在字符串中出现的频率很接近，那如果还是按照给每个字符用不同长度的二进制码表示，很有可能会比前面用相同二进制码表示耗费的空间多（亲自测过）。我觉得每一个字符在字符串中出现的频率需要满足一定的大小差距，这样使用空间才会比使用相同数量的二进制码更少。这个大小差距点的公式不太好算，笨一点的办法就是用两种方式比较下占用空间大小。

展开 ∨

作者回复: 给个你的测试用例我看看



bingo

2019-02-22

👍 1

@吴：wiki上的哈夫曼树是标准的生成步骤，老师这里举的例子是一种特殊情况，哈夫曼树构建的一般性方法在本科的教程上就写的很通俗了。

我用wiki里的值举个例子吧：原始集合的值是[2,3,4,4,5,7]

第一步：从原始集合中取出最小的两个值并将这两个值从原始集合中剔除，这两个最小的值相加得到一个新的值并加入原始集合，这两个小值作为这个新值的树叶，新值当然就...
展开

作者回复: 🐼啊 就喜欢你这么极致的人！



bingo

2019-02-22

👍 1

@spark。贪心算法是利用局部最优求解全局最优（求出来的是否正确要靠自己判断，所以适合那些局部最优加起来就是全局最优的问题）

本例中霍夫曼编码要解决的问题就是让总文本使用编码后变得占内存最小，你看，求最值，这不就是求全局最优解嘛！

下面就用我这不太靠谱的语言表达能力再写写老师的“这 1000 个字符只需要 2100bits”...
展开



0:o2

2019-01-27

👍 1

1.既然移除k个数字，不管移除哪个，剩余的长度应该都是最小的。

从高位向低位，依次移除9->8->7.....

例如 98975432，第一次移除后是8975432，第二次移除后是875432，第三次移除后是75432。。。

2.从等待时间最短的开始，依次递增

展开

