37|贪心算法：如何用贪心算法实现Huffman压缩编码？

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。

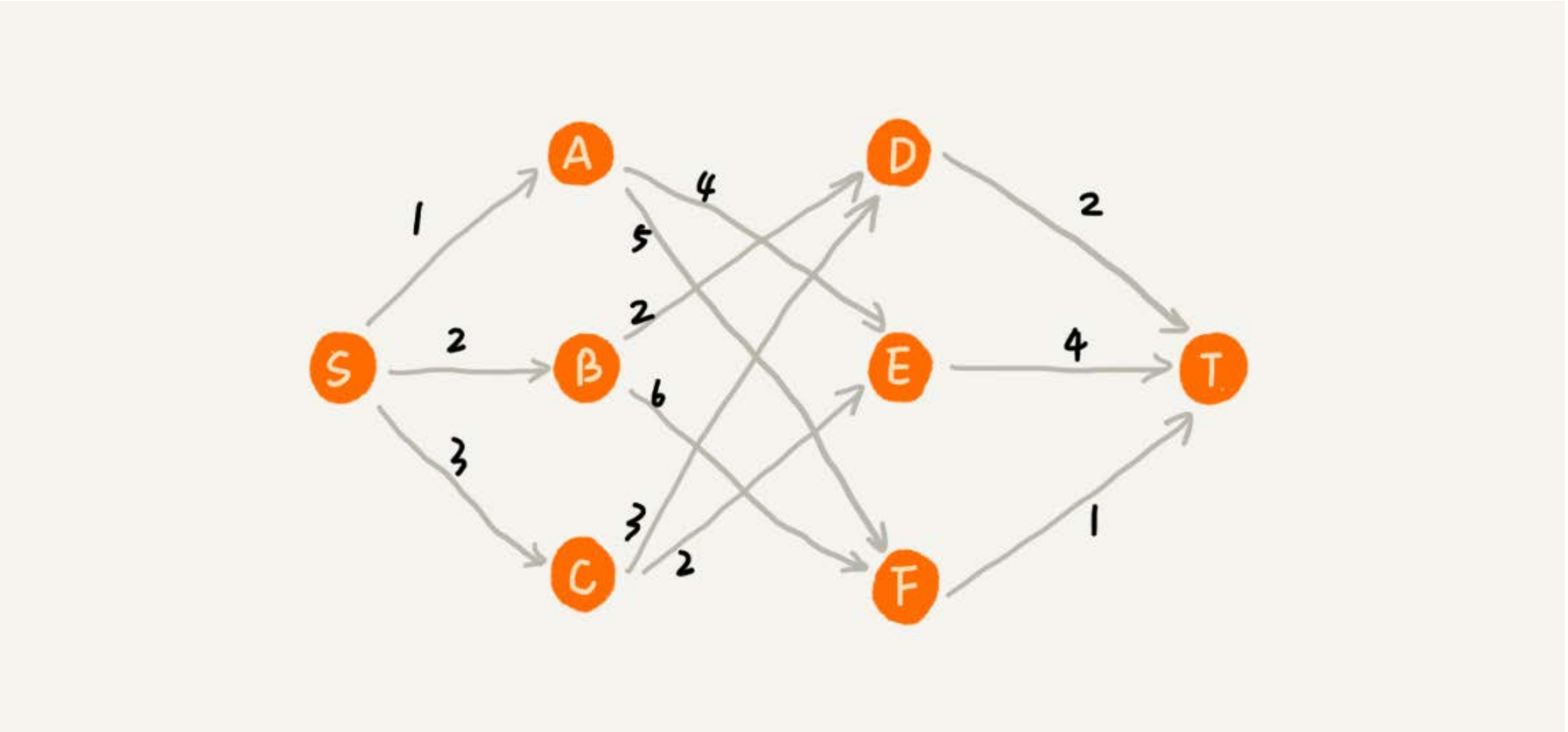
第二步，我们尝试看下这个问题是否可以用贪心算法解决：每次选择当前情况下，在对限制值同等贡献量的情况下，对期望值贡献最大的数据。

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呢？

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s1，s2，s3，……，sm。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g1，g2，g3，……，gn。

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

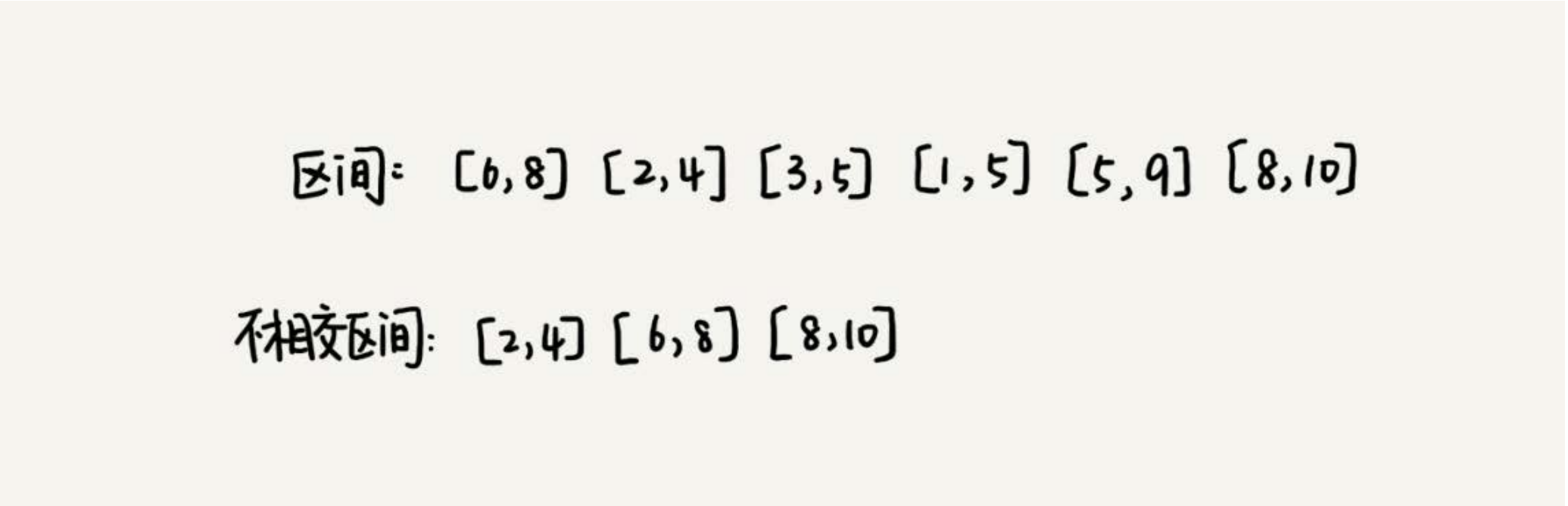
这个问题在我们的日常生活中更加普遍。假设我们有1元、2元、5元、10元、20元、50元、100元这些面额的纸币，它们的张数分别是c1、c2、c5、c10、c20、c50、c100。我们现在要用这些钱来支付K元，最少要用多少张纸币呢？

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

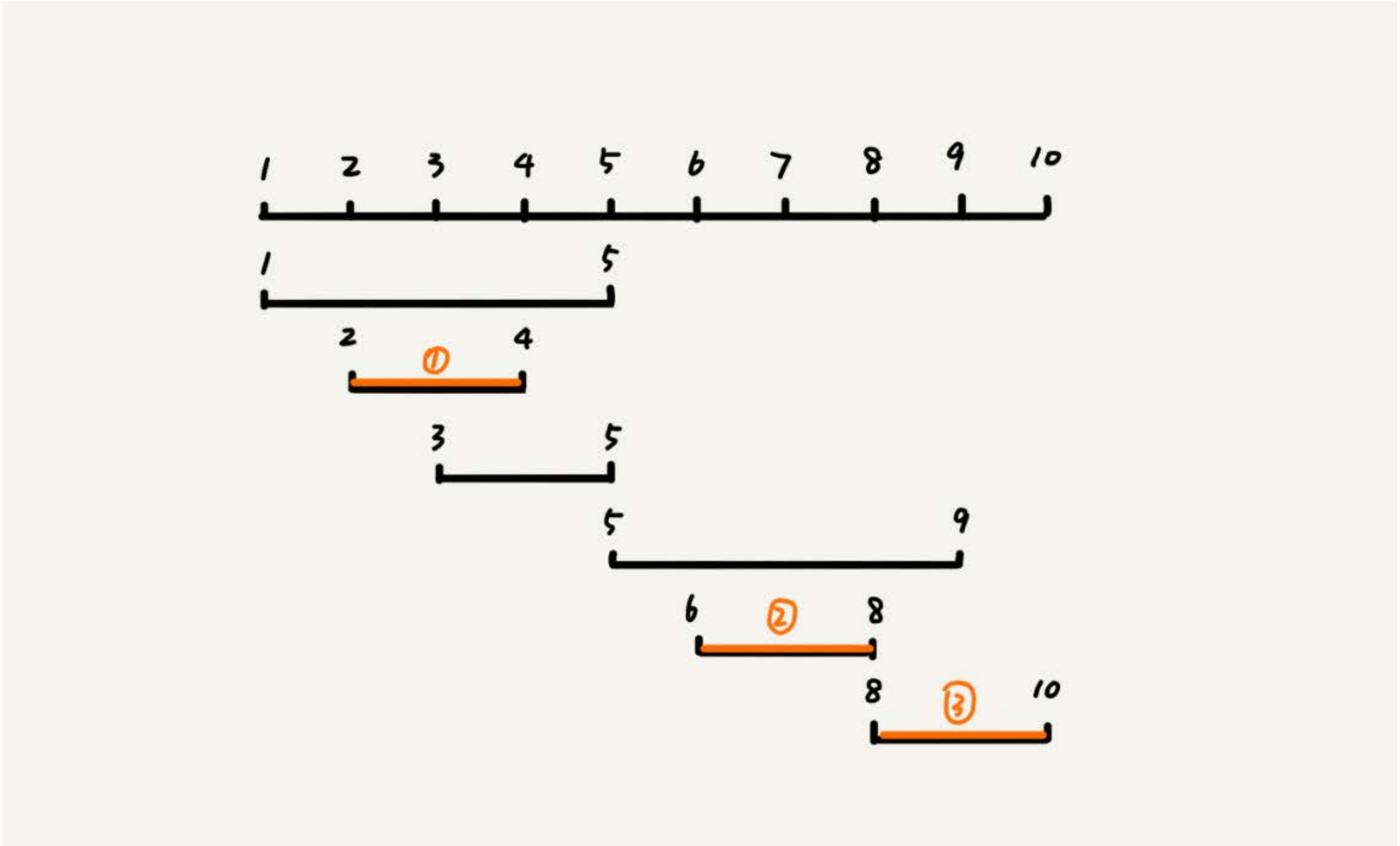
假设我们有n个区间，区间的起始端点和结束端点分别是[l1, r1]，[l2, r2]，[l3, r3]，……，[ln, rn]。我们从这n个区间中选出一部分区间，这部分区间满足两两不相交（端点相交的情况不算相交），最多能选出多少个区间呢？



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lmin，最右端点是rmax。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[lmin, rmax]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？

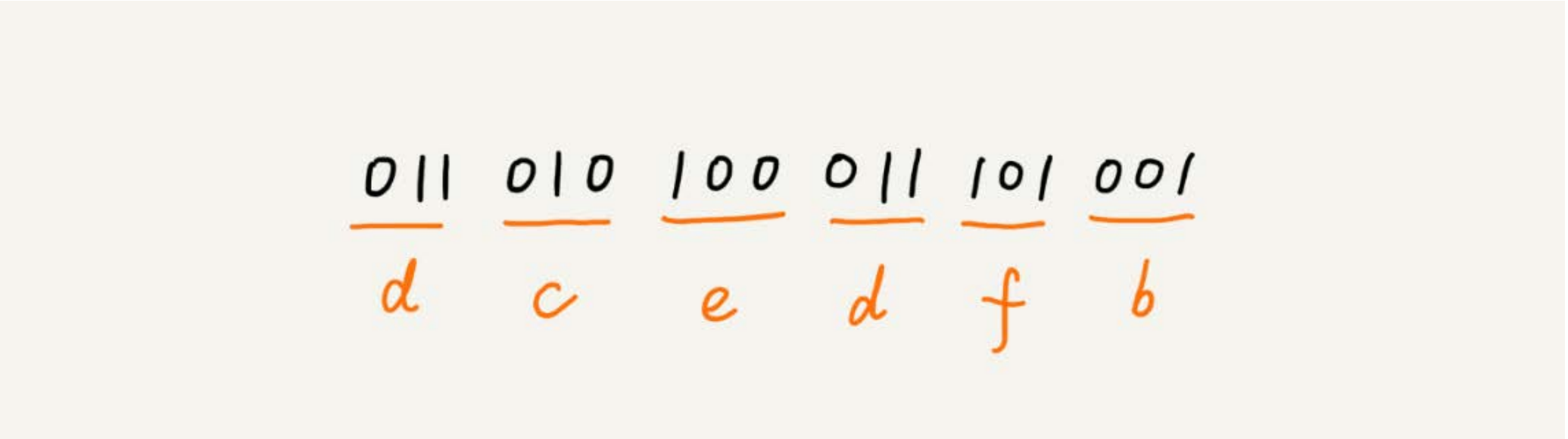
假设我们通过统计分析发现，这1000个字符中只包含6种不同字符，假设它们分别是a、b、c、d、e、f。而3个二进制位（bit）就可以表示8个不同的字符，所以，为了尽量减少存储空间，每个字符我们用3个二进制位来表示。那存储这1000个字符只需要3000bits就可以了，比原来的存储方式节省了很多空间。不过，还有没有更加节省空间的存储方式呢？

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。

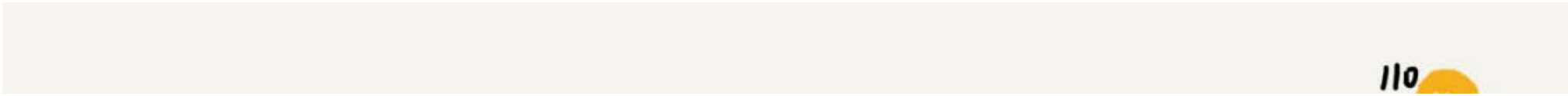


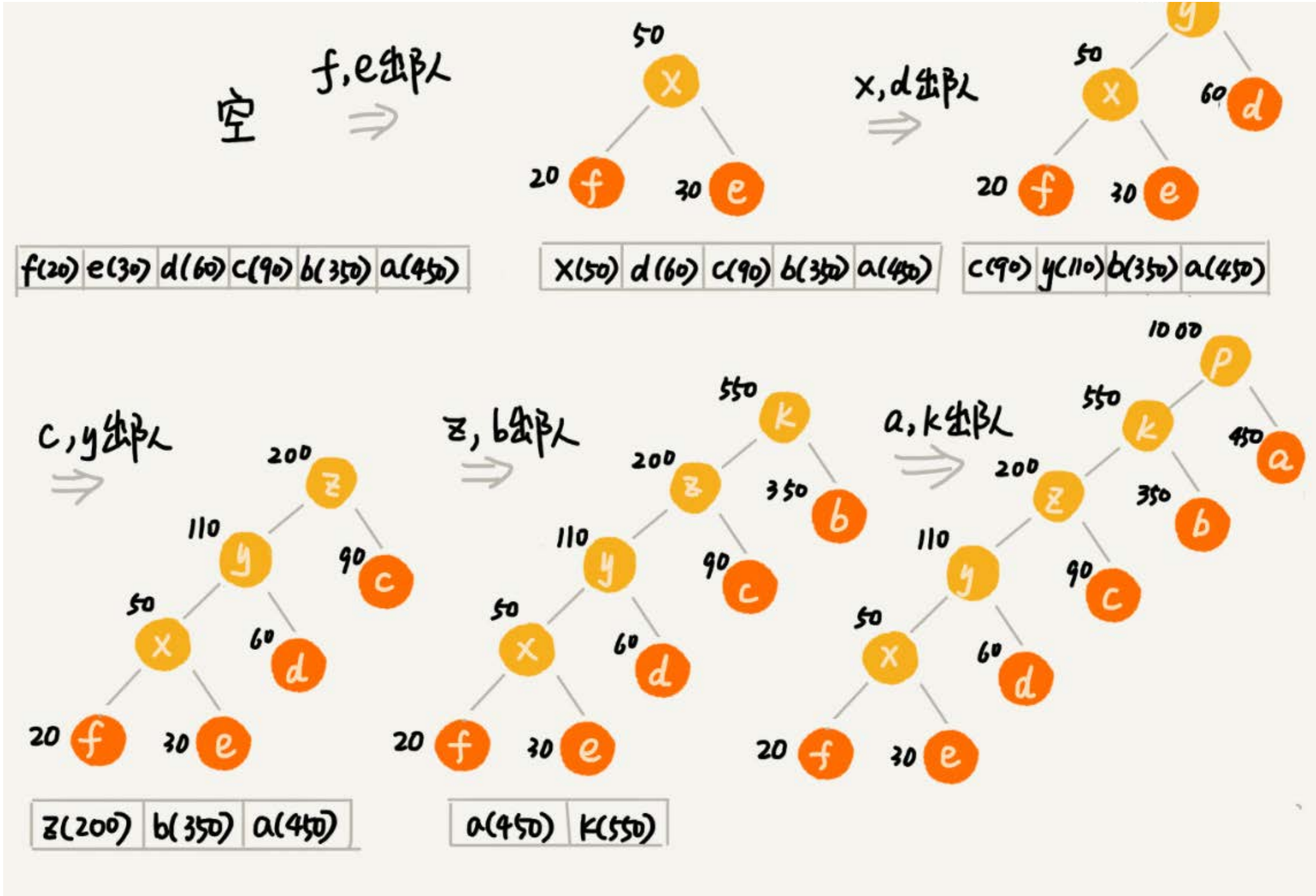
假设这6个字符出现的频率从高到低依次是a、b、c、d、e、f。我们把它们编码下面这个样子，任何一个字符的编码都不是另一个的前缀，在解压缩的时候，我们每次会读取尽可能长的可解压的二进制串，所以在解压缩的时候也不会歧义。经过这种编码压缩之后，这1000个字符只需要2100bits就可以了。

字符	出现频率	编码	总二进制位数
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。

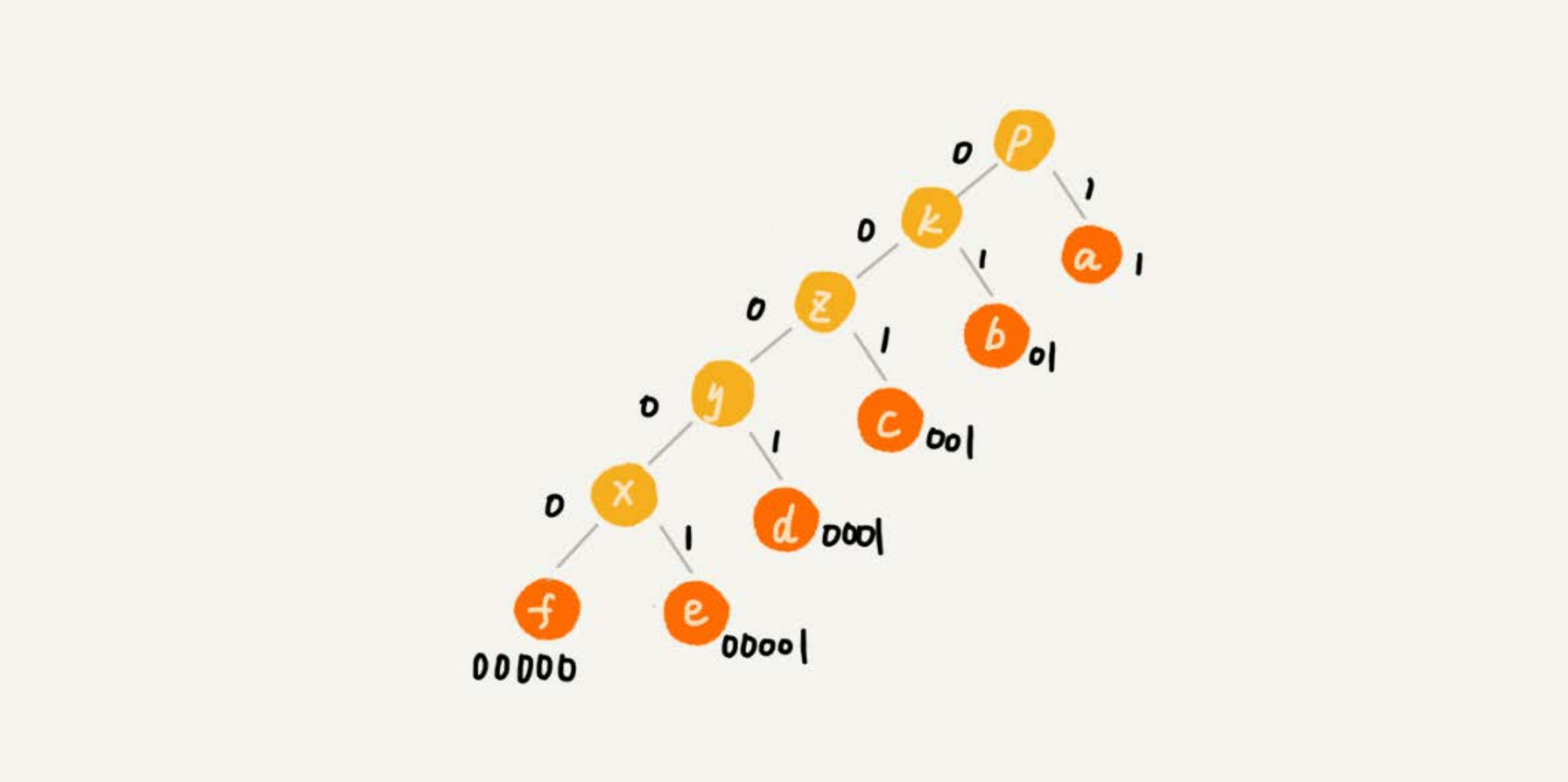
我们把每个字符看作一个节点，并且辅带着把频率放到优先级队列中。我们从队列中取出频率最小的两个节点A、B，然后新建一个节点C，把频率设置为两个节点的频率之和，并把这个新节点C作为节点A、B的父节点。最后再把C节点放入到优先级队列中。重复这个过程，直到队列中没有数据。





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。

精选留言：

- cirno 2018-12-17 11:04:31

  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

2、由等待时间最短的开始服务 [37赞]

- luxinfeng 2018-12-17 11:12:54
  老师能详细讲讲区间覆盖这个问题的选择过程么？ [7赞]

- Jalyn 2018-12-17 08:34:45
  想知道目前没掉队的有多少 哈哈 [7赞]

  作者回复2018-12-17 09:27:39
  慢慢学 不着急

- 开心小毛 2018-12-31 10:01:10
  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。 [6赞]

  作者回复2019-01-02 10:06:45
  是的

- feifei 2018-12-20 08:20:06
  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，然后每次移除大顶堆中堆顶最大的元素。

  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？
  对于此问题，我的求解思路是让待时间最长的来安排先后顺序
  比如，现在有3个人，a、b、c,a等待了10分钟，b待待了20分钟，c待待了30分钟
  同样使用大顶堆来进行存储等待时间，堆顶的元素就是当前等待时间最长的人
  然后每次从堆拿出堆顶元素的人来进行服务，这样就可以让这n个人的总的等待时间最短。

  对于学习的贪心算法，老师虽然只进行了理论讲解，但我看完了老师所讲的，我对贪心算法的理解有了一定的认识，我就试着把贪心算法的内容中涉及的东西，都翻译成了代码，
  感觉收获良多，也把这个分享给童鞋，希望对他们有帮助。

1，这是第一个示例，背包中豆子的最大价值的问题
https://github.com/kkzfl22/datastruct/tree/master/src/main/java/com/liujun/algorithm/greedyAlgorithm/case1

2，这是孩子分糖果的问题
https://github.com/kkzfl22/datastruct/tree/master/src/main/java/com/liujun/algorithm/greedyAlgorithm/case2

3，这是钱币支付的问题
https://github.com/kkzfl22/datastruct/tree/master/src/main/java/com/liujun/algorithm/greedyAlgorithm/case3

4，这是区间覆盖的问题
https://github.com/kkzfl22/datastruct/tree/master/src/main/java/com/liujun/algorithm/greedyAlgorithm/case4

5，这是霍夫漫编码的实现
https://github.com/kkzfl22/datastruct/tree/master/src/main/java/com/liujun/algorithm/greedyAlgorithm/huffman

欢迎大家与我交流，也欢迎老师给我指正，谢谢 [5赞]

- 您的好友William 2018-12-17 14:04:52
  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。。。） [4赞]

- 白若 2018-12-17 12:50:40
  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。 [2赞]

- 小美 2018-12-17 10:32:41
  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 [2赞]

  作者回复2018-12-18 10:02:44
  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 09:32:12
  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和即单个服务时间*剩余等待人数。不管现在服务的是谁，剩余等待的人数是不会变的，所以只需要找单个服务时间最小的，即按服务时间数升序服务即1-2-3-4-5，总的时间为1*4+2*3+3*2+4*1=20

就像春运取票，你如果是去取售票机上买票的话，后面排队着急的人会说，让我先取吧，我直接取很快。这样集体时间效率最高，但是对单个个体来说就不一定了，比如之前那个要买票的。 [2赞]

- qinggeouye 2019-01-19 13:13:47
  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。

  也可以用 Top K 排序，求出 K 个最大的数字，移除。

  2、假设有 n 个人等待被服务，但是窗口只有一个，每个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？

  每个人需要被服务的时间不一样，但所有人加起来总的被服务时间是固定的。

  题意是求 n 个人总的等待时间，每个人在被服务之前，所经过的等待时间是不同的。

  而当前被服务的人所需的服务时间，会累加到剩下的那些等待被服务人的等待时间上。

  要使 n 个人总的等待时间最短，那么每次安排服务时间最短的那个人被服务：堆排序（小顶堆）。

  另外，@Alexis何春光 的留言，第一句话表示赞同。 [1赞]

- Alexis何春光 2019-01-12 02:04:54
  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 [1赞]

- 观弈道人 2019-01-03 18:51:56
  感觉追求最优解的算法都可以称为贪心算法了，比如哈夫曼编码 [1赞]

- 新丁 2018-12-19 20:07:39
  贪心算法并不一定能获得最优解！学习做事也一样不能总看眼前的最优选择，坚定目标，踏踏实实的往下学，只有回过头来总结才能证明你是否获得了最优解 [1赞]

- luxinfeng 2018-12-17 10:56:45

  思考题第一题：从非负整数a的第一位开始，如果这个位置上的数比后k个位置上的数小，则保留，否则删除该数并令k=k-1；到非负整数的第二位时，如果该位置上的数比后k个位置上的数小，则保留，否则删除该数并令k=k-1，以此类推.....至k=0时结束。

  思考题第二题：按照所需服务时间由小到大的序列进行服务。

  [1赞]

- 鹏程万里 2019-01-29 16:32:42

  贪心算法：就是每一步操作对想要的结果“贡献”最大！

- 0:o2 2019-01-27 23:28:11

  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

- 睡痴儿 2019-01-22 21:57:25

  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，但是因为有顺序，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。

- 随心而至 2019-01-18 08:26:22

  贪心，每一步都选择目前最优的选择

  问题1；在前2k中，每次都从数字中拿出最大值，直到取了k次

  问题2: 被服务者需要时间优先

- luo 2019-01-15 10:50:17

  贪心算法思考题：

  1.数字去除k位使数值最小

  146579238 移除4个数据 14657

  14523 移除最大？

  416723897 4必须移除

  16723

  12389

  12387 位数越高数据要求越小,限定条件是移除k位数字，期望值移除后后去数值最小，前k+1位必须是获取最小的放在第一位，比如移除4位则前四位最小的

  1 +1 4 3 6723 2,2

是 ，一之前有几位数据则移除几位，下标从最小的这个数据开始 ，在比较剩余移除位数 比如现在移除 剩余移除 位，在 比较最小为 前面需要移除2位，剩余移除位数为1,38中3最小不移除，89 8最小不移除，97 7最小移除 剩余为12387 位最小数据。

- cw 2019-01-13 09:29:26

  1.从高位开始，如果高位小于低位则比对低位，如果高位大则直接删除高位。

  2.等待时间短开始服务