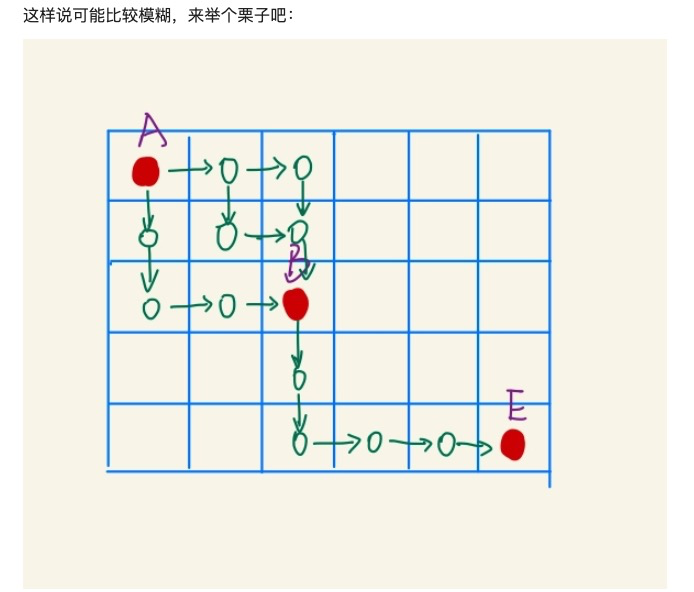
背包问题

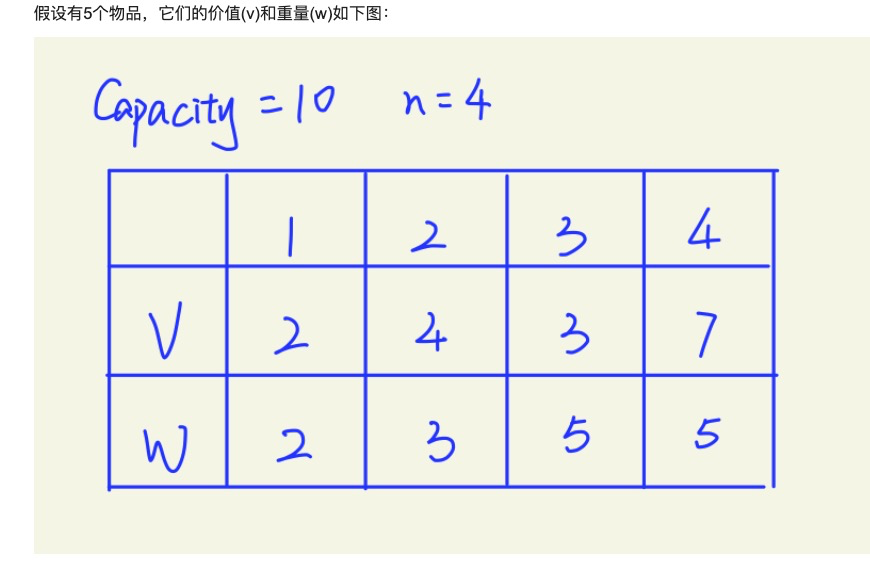
最优化原理指的最优策略具有这样的性质：不论过去状态和决策如何，对前面的决策所形成的状态而言，余下的诸决策必须构成最优策略。简单来说就是一个最优策略的子策略也是必须是最优的，而所有子问题的局部最优解将导致整个问题的全局最优。如果一个问题能满足最优化原理，就称其具有最优子结构性质。

这是判断问题能否使用动态规划解决的先决条件，如果一个问题不能满足最优化原理，那么这个问题就不适合用动态规划来求解。



1. 背包问题：

假设你是一名经验丰富的探险家，背着背包来到野外进行日常探险。天气晴朗而不燥热，山间的风夹杂着花香，正当你欣赏这世外桃源般的美景时，突然，你发现了一个洞穴，这个洞穴外表看起来其貌不扬，但凭借着惊为天人的直觉，这个洞穴不简单。



背包总容量为10，现在要从中选择物品装入背包中，要求物品的重量不能超过背包的容量，并且最后放在背包中物品的总价值最大。

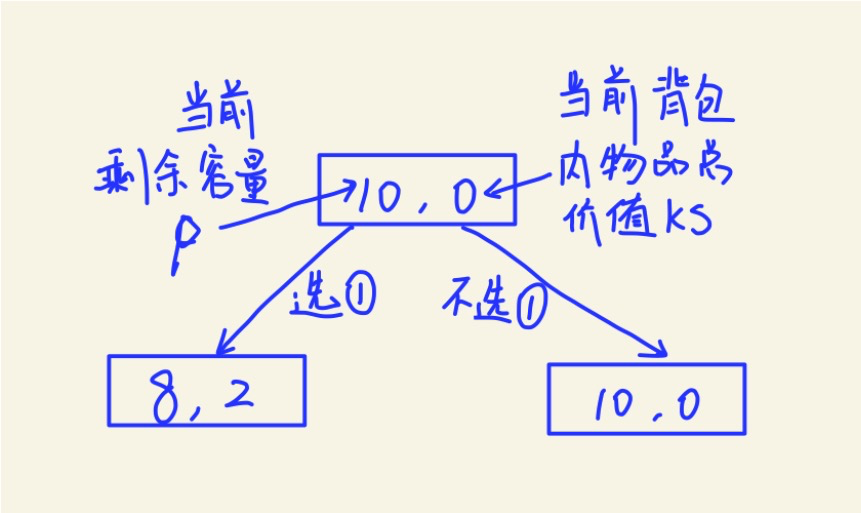
为什么叫做0/1背包呢？为什么不叫1/2背包，2/3背包？？？

仔细想想，这里每个物品只有一个，对于每个物品而言，只有两种选择，盘它或者不盘，盘它记为1，不盘记为0，我们不能将物品进行分割，比如只拿半个是不允许的。这就是这个问题被称为0/1背包问题的原因。

所以究竟选还是不选，这是个问题。

让我们先来体验一下将珠宝装入背包的感觉，为了方便起见，用xi代表第i个珠宝的选择（xi = 1 代表选择该珠宝，0则代表不选），vi代表第i个珠宝的价值，wi代表第i个珠宝的重量。于是我们就有了这样的限制条件：

我们的初始状态是背包容量为10，背包内物品总价值为0，接下来，我们就要开始做选择了。对于1号珠宝，当前容量为10，容纳它的重量2绰绰有余，因此有两种选择，选它或者不选。我们选择一个珠宝的时候，背包的容量会减少，但是里面的物品总价值会增加。就像下面这样：



这样就分出了两种情况，我们继续进行选择，如果我们选择了珠宝1，那么对于珠宝2，当前剩余容量为8，大于珠宝2的容量3，因此也有两种选择，选或者不选。

现在，我们得到了四个可能结果，我们每做出一个选择，就会将上面的每一种可能分裂成两种可能，后续的选择也是如此，最终，我们会得到如下的一张决策图：

这里被涂上色的方框代表我们的最终待选结果，本来应该有16个待选结果，但有三个结果由于容量不足以容纳下最后一个珠宝，所以就没有继续进行裂变。

然后，我们从这些结果中，找出价值最大的那个，也就是13，这就是我们的最优选择，根据这个选择，依次找到它的所有路径，便可以知道该选哪几个珠宝，最终结果是：珠宝4，珠宝2，珠宝1。

接下来，我们就来分析一下，如何将它扩展到一般情况。为了实现这个目的，我们需要将问题进行抽象并建模，然后将其划分为更小的子问题，找出递推关系式，这是分治思想中很重要的一步。

抽象问题，背包问题抽象为寻找组合（x1,x2,x3...xn，其中xi取0或1，表示第i个物品取或者不取），vi代表第i个物品的价值，wi代表第i个物品的重量，总物品数为n，背包容量为c。

建模，问题即求max(x1v1 + x2v2 + x3v3 + ... + xnvn)。

约束条件，x1w1 + x2w2 + x3w3 + ... + xnwn < c

定义函数KS(i,j)：代表当前背包剩余容量为j时，前i个物品最佳组合所对应的价值；

那这里的递推关系式是怎样的呢？对于第i个物品，有两种可能：

背包剩余容量不足以容纳该物品，此时背包的价值与前i-1个物品的价值是一样的，KS(i,j) = KS(i-1,j)

背包剩余容量可以装下该商品，此时需要进行判断，因为装了该商品不一定能使最终组合达到最大价值，如果不装该商品，则价值为：KS(i-1,j)，如果装了该商品，则价值为KS(i-1,j-wi) + vi，从两者中选择较大的那个，所以就得出了递推关系式：

对于这个问题的子问题，这里有必要详细说明一下。原问题是，将n件物品放入容量为c的背包，子问题则是，将前i件物品放入容量为j的背包，所得到的最优价值为KS(i,j)，如果只考虑第i件物品放还是不放，那么就可以转化为一个只涉及到前i-1个物品的问题。如果不放第i个物品，那么问题就转化为“前i-1件物品放入容量为j的背包中的最优价值组合”，对应的值为KS(i-1,j)。如果放第i个物品，那么问题就转化成了“前i-1件物品放入容量为j-wi的背包中的最优价值组合”，此时对应的值为KS(i-1,j-wi)+vi。

所以，就可以很容易的写出递归解法了：

