第八章　动态规划

8.1　练习

Problem 8.1.1
试给出算法10.3[1]中的$p(i,j)$ 如何经过$\Theta(n)$ 的预处理后，每次调用时计算的代价能降为$\Theta(1)$。

Problem 8.1.2
完成line breaking完整的动态规划算法[2]。算法的输出不仅包括penalty的值还需要有一个数组$lastWord$，其中$lastWord[L]$ 记录了第L行的最后一个单词的下标。（当$lastWord[L] = n$，整个段落在第L 行结束。）

Problem 8.1.3 (最大和子串问题)
对于一个包含负值的数字串A[1...n]，要找到它的一个子串$A[i...j](0 \le i \le j \le n)$，使得在A的所有子串中，A[i...j]的和最大。请使用动态规划策略设计一个$O(n)$ 的算法。

Problem 8.1.4 (整数子集合问题)
给定一个自然数集合$A = \{s_1, s_2, \cdots, s_n\}$和自然数$S$。判断是否存在$A$ 的子集，其中元素的和恰好是S。给出一个动态规划算法来解决该问题。

Problem 8.1.5 (任务调度问题)
给定一组任务$S = \{a_1, a_2, \cdots, a_n\}$，每个任务$a_i$的持续时间是$[s_i, f_i)$。由于每个任务都需要独占一个公共的资源，所以我们需要找出一个最大的mutually compatible 的任务子集合。请用动态规划策略设计一个任务调度算法（注：本题实际也可以使用贪心算法求解）。

Problem 8.1.6 (All-Pair Shortest Path)
给定图，求图中所有点对之间的最短路径。请从动态规划策略的角度，重新解读Floyd-Warshall算法。

8.2　问题

Problem 8.2.1 (最长公共子序列问题)
给定一个序列，其子序列的定义是：将序列中的一个或者多个元素去掉而形成的序列。子序列是给定序列中部分元素，依照原来的出现顺序组成的序列。形式化描述为，给定一个序列$X =< x_1, x_2, \cdots, x_m >$，另一个序列$Z =< z_1, z_2, \cdots, z_k >$ 是X的子序列，当存在一个严格递增

[1]请参考Sara Baase and Allen Van Gelder. Computer Algorithms-Introduction to Design and Analysis(算法设计与分析)
[2]请参考Sara Baase and Allen Van Gelder. Computer Algorithms-Introduction to Design and Analysis(算法设计与分析)10.5 章

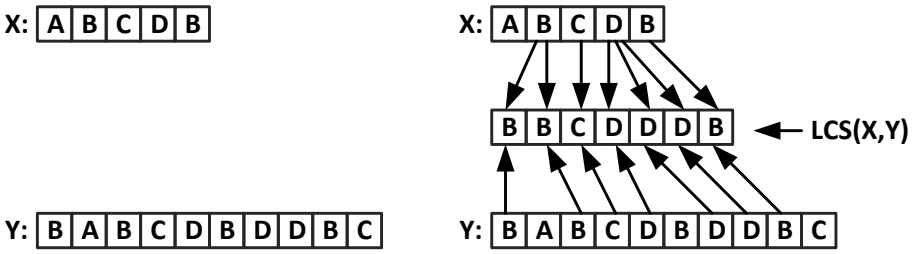
的X的下标的序列$i_1, i_2, \cdots, i_k$，对于所有的$j = 1, 2, \cdots, k$都有$x_{i_j} = z_j$。例如，$Z =< B, C, D, B >$是$X =< A, B, C, B, D, A, B >$ 的子序列，其对应的下标序列为$< 2, 3, 5, 7 >$。

给定两个序列X和Y，当序列Z同时是X和Y的子序列时，我们说Z是X 和Y 的公共子序列。针对最长公共子序列问题，给定两个序列X和Y，找到X和Y 的具有最大长度的公共子序列。

Problem 8.2.2

请给出求解下面两个问题的算法并简要说明算法的正确性并给出算法的时间复杂度。

(a) 下面是基于最长子序列的问题。给定两个字符串$X = \langle x_1, x_2, ..., x_m \rangle$和$Y = \langle y_1, y_2, ..., y_n \rangle$。请给出计算最长子序列长度的算法，在最长子序列中$X$中的字符可以重复出现但是$Y$中的不可以。算法只需要给出长度不需要具体的子序列。



(b) 除了X, Y两个字符串之外，给定一个正整数k。问题和(a)中的一样，但是子序列中X 中的字符重复出现的次数不超过k，Y中的字符不可重复出现。

例如，假设X, Y如(a)中的图所示，$k = 2$，那么在最长子序列中只有'D'可以出现两次而不是三次。

Problem 8.2.3

请设计高效的算法找到字符串$T[1...n]$中前向和后向相同的最长的连续子串的长度。前向和后向的子串不能够重叠。下面是几个例子。

- 给定输入字符串**ALGORITHM**，你的算法返回0。
- 给定输入字符串**RECURRSION**，你的算法返回1，子串是**R**。
- 给定输入字符串**REDIVIDE**，你的算法返回3，子串是**EDI** （前向和后向的字符串不能够重叠）。
- 给定输入字符串**DYNAMICPROGRAMMINGMANYTIMES**，你的算法返回4，子串是**YNAM**

Problem 8.2.4

令$A[1...m]$和$B[1...n]$是两个任意的序列。A, B的公共超序列（Supersequence）是指A, B皆为它的子序列。请描述一个高效算法找到A, B的最短公共超序列，算法最终只需给出长度。

Problem 8.2.5

两个字符串X, Y的$shuffle$是将X, Y中的字符插入到一个新的字符串中，并且X, Y 中的字符仍然保持原来的顺序。例如，'banana' 和'ananas'通过下面几种方式形成'bananaananas'。

$$\text{banana}_{\text{ananas}} \quad \text{ban}_{\text{ana}}\text{ana}_{\text{nas}} \quad \text{b}_{\text{an}}\text{an}_{\text{a}}\text{a}_{\text{na}}\text{na}_{\text{s}}$$

字符串'prodgyrnamammiincg'和'dyprongarmammicing'都是'dynamic' 和'programming' 的$shuffle$：

$$\mathrm{pro^{d}g^{y}r^{nam}ammi^{i}n^{c}g} \quad \mathrm{dy_{pro}n_{g}a_{r}m_{amm}ic_{ing}}$$

给定三个字符串$A[1...m], B[1...n], c[1...m+n]$，描述并分析算法判断$C$ 是否是A, B 的$shuffle$。

Problem 8.2.6

这个问题中我们考虑最长公共子序列问题的两个变体。给定三个序列X, Y, Z，它们的长度分别为$|X| = m, |Y| = n, |Z| = k$，想要知道序列$X$和$Y$ 是否可以合并成为一个新的序列Z并且不改变其中任何一个序列中元素的相对顺序。例如，$X = \langle ABC\rangle$，$Y = \langle BACA\rangle$可以合并成为$Z = \langle ABBACCA\rangle$，（$Z = \langle A_x, B_x, B_y, A_y, C_x, C_y, A_y\rangle$），但是如果$Z = \langle ABCBAAC\rangle$，$X, Y$就不能合成$Z$。显然有$k = m + n$，否则$A, B$ 一定不能合成为Z。

(a) $Farnsworth$教授声称他找到了下面这个简单的算法来解决这个问题。首先，计算X 和Z的LCS（最长子序列），令Z' 是从Z 中将LCS 中的元素删除得到序列。如果$Y = Z'$，则答案为是，否则为否。如果算法正确，请证明，否则请给出反例。

(b) 请给出解决这个判定问题的复杂度为$O(nm)$的算法。（无论$Farnsworth$教授的算法是否正确，这里都不可以使用）

(c) 现在将这个判定问题转化为一个优化问题，从X, Y, Z中删除数目最少的元素使得合并成立。例如，$X = \langle ABC\rangle$，$Y = \langle BACA\rangle$，$Z = \langle ABCBAAC\rangle$，如果从$Y$ 中删除最后一个元素，从Z中删除除第一个元素以外的所有元素，X 和Y就可以合并成为Z。优化问题的解就是2。算法不仅需要得到最少的元素删除数目还需要输出所删除的元素集合，时间复杂度是$O(mnk)$ （这里$k = m + n$不再是必要条件）。

Problem 8.2.7

给定数组$A[1...n]$，数组中元素的值可以是正数，负数或者0。要找到它的子数组$A[i...j]$，使得$A[i...j]$中各个元素的乘积最大。

(a) 假设A中的元素皆为正数，请给出相应的算法。

(b) 假设A中的元素有正，有负，请给出相应的算法。

Problem 8.2.8

给定包含n个字符的字符串$s[1\ldots n]$。该字符串可能来自于一本年代久远的书籍，只是由于纸张朽烂的缘故文档中所有的标点符号都不见了（因此该字符串看起来就像这样：“itwasthebestoftimes...”）。现在您希望在字典的帮助下重建这个文档。在此字典表示为一个布尔函数$dict(\cdot)$，对于任意的字符串w，

$$dict(w) = \begin{cases} true & w\text{是合法的单词} \\ false & \text{其他情况} \end{cases}$$

(a) 请给出一个动态规划算法，判断$s[\cdot]$是否能重建为由合法单词组成的序列。假设调用$dict$每次只需一个单位的时间，该算法运行时间应该不超过$O(n^2)$。

(b) 若$s[\cdot]$是由合法单词组成的，请在算法中输出对应的单词序列。

Problem 8.2.9

当字符串颠倒和原字符串相同时，我们称字符串为回文。例如，**I**，**DEED**，**RACECAR**。

(a) 描述并分析找到给定字符串的满足回文条件的最长子序列的算法，算法最终只需给出长度即可。例如，**MAHDYNAMICPROGRAMZLETMESHOWYOUTHEM** 的最长回文子序列是**MHYMRORMYHM**，所以算法最终得到的结果是11。

(b) 任何一个字符串都可以拆分为一组回文。例如，字符串**BUBBASEESABANANA**（'Bubba sees a banana.'）可以根据下面的步骤分解为回文。

BUB+BASEESAB+ANANA

B+U+BB+A+SEES+ABA+NAN+A

B+U+BB+A+SEES+A+B+NANA

B+U+B+B+A+S+E+E+S+A+B+N+A+N+A

请描述并分析算法对给定字符串可以拆分为的最少的回文数量。例如，给定的输入字符串**BUBBASEESABANANA**，你的算法给出的结果应该是3。

Problem 8.2.10

某种字符串处理语言提供了一个将字符串一分为二的基本操作。由于该操作需要拷贝原来的字符串，因此对于长度为n 的串，无论在其什么位置进行分割，都需要花费n个单位的时间。现在假设我们要将一个字符串分割成多段，具体的分割次序会对总的运行时间产生影响。例如，如果要在位置3和位置10 分割一个长度为20的串，首先在位置3进行分割产生的总代价为$20 + 17 = 37$，而首先在位置10 进行分割产生的总代价为$20 + 10 = 30$。

请给出一个动态规划算法，对于给出了m个分割位置的长度为n的字符串，计算完成所有分割的最小代价。

Problem 8.2.11

(a) 给定数量无限的面值分别为$x_1, x_2, \ldots, x_n$的硬币，我们希望将价格v兑换成零钱。也即，我们希望找出一堆总值恰好为v 的硬币。这有时候是不可能的：例如，如果硬币只有5和10 两种面值，则我们可以兑换15 却不能兑换12。请给出一个针对如下问题的$O(nv)$ 的动态规划算法：输入：$x_1, \cdots, x_n$；v。问题：能否用面值分别为$x_1, \cdots, x_n$的硬币兑换价格v？

(b) 考虑以上零钱兑换问题的一个变型：您有面值$x_1, x_2, \ldots, x_n$ 的硬币，希望兑换的价格为v，但是每种面值的硬币最多只能使用一次！举例来说，如果硬币面值为$1, 5, 10, 20$，则可以兑换的价格包括$16 = 1 + 15$ 和$31 = 1 + 10 + 20$，但是无法兑换40 （因为20不能用两次）。输入：正整数$x_1, x_2, \ldots, x_n$；以及另一个整数v。输出：是否能够仅使用面值分别为$x_1, x_2, \ldots, x_n$中的硬币最多一次，兑换价格v？请说明如何在$O(nv)$时间内求解该问题。

(c) 考虑以上零钱兑换问题的另一个变型。给定无限多的面值$x_1, x_2, \ldots, x_n$的硬币，我们希望用其中最多k枚硬币兑换价格v。即，我们需要找到不超过k 枚的硬币，使其总面值为v。这也可能是无法实现的：例如，若面值为5 和10，$k = 6$，则我们将可以兑换55，但却不能兑换65。请给出以下问题的动态规划算法：输入：$x_1, x_2, \ldots, x_n$；k；v。问题：是否有可能用不超过k枚面值分别为$x_1, x_2, \ldots, x_n$ 的硬币兑换价格v？

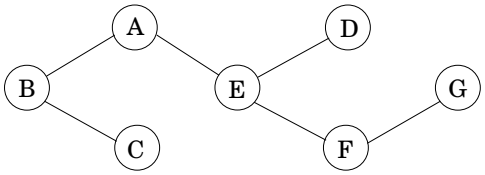
Problem 8.2.12

图$G=(V,E)$的一个顶点覆盖S是V的子集，满足：E中的每条边都至少有一个端点属于S。请给出如下任务的一个线性时间的算法：

输入：无向树$T=(V,E)$。

输出：T的最小覆盖的大小。

例如，对如下的树，可能的顶点覆盖包括$\{A,B,C,D,E,F,G\}$和$\{A,C,D,F\}$，不包括$\{C,E,F\}$。最小顶点覆盖的大小为3，对应集合$\{B,E,G\}$。



Problem 8.2.13

序列对齐。新的基因被发现后，为了了解其功能，一个常用的方法是查找已知基因中与之相似的基因。两个基因之间的相似度是由它们可以相互对齐的程度决定的。为了对它进行量化，考虑基于字母表$\Sigma=\{A,C,G,T\}$定义的字符串。设两个基因（字符串）分别为$x=ATGCC$，$y=TACGCA$。x和y的一个对齐是指将它们都横向书写，然后将其中相同的字符进行匹配的方式，例如：

$$\begin{array}{cccccccc} - & A & T & - & G & C & C \\ T & A & - & C & G & C & A \end{array}$$

其中："–"表示一个"空隙"；串中的所有字符必须保持原有顺序，且每列必须包含来自至少一个字符串的某个字符。对齐的分值基于一个$(|\Sigma|+1)\times(|\Sigma|+1)$ 的评分矩阵δ定义，其中包含额外的一行和一列对应于空隙的情况。举例来说，此前列举的对齐方式分值如下：

$$\delta(-,T)+\delta(A,A)+\delta(T,-)+\delta(-,C)+\delta(G,G)+\delta(C,C)+\delta(C,A)$$

请给出一个动态规划算法，以两个字符串$x[1\ldots n],y[1\ldots m]$以及评分矩阵$\delta$为输入，返回分值最高的对齐。要求运行时间为$O(mn)$。

Problem 8.2.14

现在考虑带权重版本的课程表问题。不同的课程学分不同（学分和课程的时间无关）。你的目标是选择一组互不冲突的课程使得得到的学分最多。

(a) 现有贪心算法是按照课程结束时间的先后顺序总是选择最先结束的课程，请证明这个算法并不能够总是得到最优解。

(b) 请给出在$O(n^2)$时间内解决该问题的算法。

(c) 是否可以用动态规划的方法解决此问题。

Problem 8.2.15

假设您准备开始一次长途旅行。以0英里为跑点，一路上共有n座旅店，距离起点的英里数分别为$a_1<a_2<\cdots<a_n$。旅途中，您只能在这些旅店停留，当然在哪里停留完全由您决定。最后一座旅店（a_n）是您的终点。

理想情况下，您每天可以行进200英里，不过考虑到旅店间的实际距离，有时候可能还达不到这么远. 如果您某天走了x英里，那么您将受到$(200-x)^2$ 的惩罚。您需要计划好行程，以使得总的惩罚– 每天所受惩罚的总和最小。请给出一个高效的算法，用于确定一路上最优的停留位置序列。

Problem 8.2.16

Yuckdonald公司计划沿着Quaint Valley高速公路修建一系列酒店。n个可能的选址位于一条直线上，它们距离高速起点的距离依次（按照升序）为$m_1, m_2, \cdots, m_n$。限制条件如下：

- 在每个地址上最多修建一座酒店。在位置i建设酒店可能带来的利润为p_i，其中$i = 1, 2, \cdots, n$，$p_i > 0$。
- 两个酒店间至少间隔k英里，其中k为正整数。

请给出一个高效的算法，计算修建这些酒店最多能获得的利润总额。

Problem 8.2.17

$Vankin'smile$是一个在$n \times n$的格子上玩的纸牌游戏。玩家一开始将纸牌放在任意的一个格子上，然后在每一轮中只可以将纸牌向下或者向右移动一个格子，当纸牌被玩家移动超出边界时游戏结束。每一个格子有一个数值，可以是正数，负数或者0。一开始，玩家的得分为0；只要玩家的牌未超过边界，就将当前所处格子的数值加到分数中，这个游戏的目标就是使得得分越高越好。

例如，给定下面的格子，玩家一开始将纸牌放置在第二行上的8对应的格子，然后将纸牌向下移动，向下移动，向右移动，向下移动，游戏结束。他最终可以得到8-6+7-3+4=10 分，但这并不是可以得到的最高得分。

-1	7	-8	10	-5
-4	-9	**8**	-6	0
5	-2	**-6**	-6	7
-7	4	7	**-3**	-3
7	1	-6	**4**	-9

请描述并分析高效算法来得到$n \times n$个格子上玩$Vankin'smile$所能得到的最高分数。

Problem 8.2.18

一个送披萨的男孩有一系列的订单要送。这些订单的目的地用坐标$\{p_1, p_2, ..., p_n\}$来表示，$p_i = (x_i, y_i)$。假设x坐标是严格递增的，即$x_1 < x_2 < ... < x_n$。披萨店的位置在p_1，送外买的路线满足两个约束条件：首先按照x坐标递增的顺序送外卖；然后按照x递减的顺序送外卖并且最终回到披萨店。（见图8.1）假设已有函数$dist(i, j)$ 用来计算p_i 和p_j 之间的距离，请给出计算最短外卖路线代价的算法。（不要求求解最短路径，只需要计算代价即可）

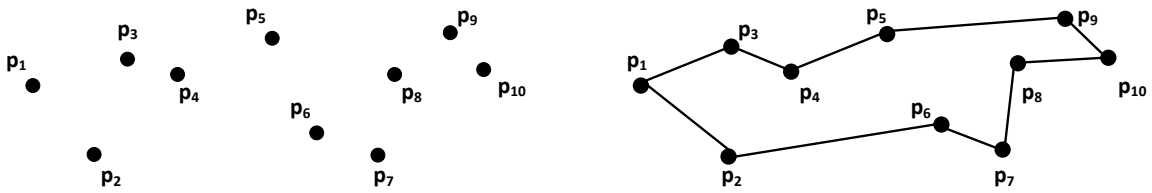


图 8.1: 问题的输入输出图示