Introduction - Dynamic Programming

动态规划介绍

动态规划（Dynamic Programming，DP）是运筹学（线性规划、网络流等问题也属于运筹学）中的一个问题分支，用于求解最优解。

DP 模型基本上是一种递归方程，把问题的各阶段联系起来，保证每个阶段的最优可行解对于整个问题既是最优的也是可行的。一般来说递归方程的结构对于初学者并不“合乎逻辑”，“难以理解”，最好的做法是做一些适当的计算来理解方程的正确性。运筹学中将动态规划问题分为*确定性*动态规划和*随机性*动态规划。在我们的大学生计算机算法中，只考虑确定性动态规划问题。

DP 计算的基本特性：

(1) 每个阶段所做的计算都是该阶段可行路径的函数，并且只针对该阶段；

(2) 当前阶段仅仅连接到紧接着的上一阶段，与再前面的阶段无关。这种连接是以最短距离小结的形式表示出上一阶段的输出；

递归公式：$x_i = f(x_{i-1})$，其中x_0为初始的常数，i从 1 开始增加。在这个方程中，x_i是系统在阶段i的状态，它由前一个状态x_{i-1}计算得到，而不需要考虑更前一个状态x_{i-2}，而后一个状态x_{i+1}也只需要x_i的信息。状态拆分使得我们只需要根据当前状态就可以对后一个状态作出最优决策，而不需要重新考察所有前面阶段所做的决策。递归公式也称作状态转移方程。

最优性原则：对以后阶段所做出的未来决策会产生一个最优策略，它与前面各阶段所采用的策略无关。

DP 模型具有 3 个基本要素：

(1) 定义*阶段*；

(2) 定义每个阶段的*可选方案*；

(3) 定义每个阶段的*状态*；

在这 3 个要素中，状态的定义往往是最微妙的：是什么样的关系把各阶段联系在一起；在当前阶段作出可行决策，又不用重新考察前面阶段所做的决策，需要什么信息。

实际编程中，状态通常会存储在数组中，对于长度为 n 的数组，其范围不再是$[0, n-1]$，而会专门把 0 空出来，范围是$[1, n]$，0 这个部分一般用来存储固定的初始值。

在本书中，我们将 DP 问题分为 4 个部分：

(1) 线性 DP；

(2) 背包问题；

(3) 区域 DP；

(4) 树型 DP；