分享此页 (https://twitter.com/intent/tweet?url=https%3A%2F%2Flearnxinyminutes.com%2Fdocs%2Fzh-cn%2Fdynamic-programming-cn%2F&text=Learn+X+in+Y+minutes%2C+where+X%3DDynamic+Programming)

X分钟速成Y (/)

其中 Y=Dynamic Programming

源代码下载： dynamic-programming-cn.html.markdown (/docs/files/dynamic-programming-cn.html.markdown)

动态规划

简介

动态规划是一种实用的技巧，它可以用来解决一系列特定问题。它的思路很简单，如果你对某个给定的输入解决了一个问题，那么你可以保存已有信息，以避免重复计算，节约计算时间。

记住，只有那些没有办法记住历史的才被迫做更多的苦力。(Fibonacci就是一个显然的例子)

解决问题的方式

1. *自顶向下*: 利用分支策略分解问题。如果你已经解决过当前子问题了，那么就返回已有信息。如果当前子问题没有计算过，那么就对它进行计算。这样的方法很易于思考、很直观。这被称作“记忆化”。

2. *自底向上*: 首先分析问题，将问题分解为不同规模的问题，并决定它们的顺序，按顺序计算，直到解决给定规模的问题。这样的流程可以保证在解决较大的问题之前解决（它所依赖的）较小的问题。这种流程被称作“动态规划”。

动态规划的例子

最长上升子序列问题。给定S= {a[1] , a[2] , a[3], a[4], ..., a[n-1], a[n] }，求出一个子序列，使得对于所有在这个子序列中所有满足j<i的j与i，满足aj<ai。首先我们要讨论以原序列的第i个元素结尾的最长上升子序列dp[i]。那么答案是整个dp序列的最大值。考虑dp[i]，它的最后一个元素为a[i]。枚举它的倒数第二个元素a[j]，则a[j]<a[i]成立。则dp[i]就是所有这样的dp[j]的最大值加上1(最后一个元素)。这个算法具有*O(n^2)*的时间复杂度。

此算法的伪代码：

```
for i=0 to n-1
    dp[i]=0
    for j=0 to i-1
        if (a[i] >  a[j] and dp[i]<dp[j])
            LS[i] = LS[j]
    dp[i]=dp[i]+1
for i=0 to n-1
    if (largest < dp[i])
        largest = dp[i]
```

这个算法的复杂度可以通过将数组换为其他数据结构来优化，来获得*O(n * log n)*的时间复杂度。

同样的思路可以求出有向无环图上的最大路径。

一些著名的动态规划问题及其实现

- Floyd Warshall 算法 - 教程与C实现源码 (http://www.thelearningpoint.net/computer-science/algorithms-all-to-all-shortest-paths-in-graphs---floyd-warshall-algorithm-with-c-program-source-code)
- 整数背包问题 - 教程与C实现源码 (http://www.thelearningpoint.net/computer-science/algorithms-dynamic-programming---the-integer-knapsack-problem)
- 最长公共子序列问题 - 教程与C实现源码 (http://www.thelearningpoint.net/computer-science/algorithms-dynamic-programming---longest-common-subsequence)

在线资源

- codechef (https://www.codechef.com/wiki/tutorial-dynamic-programming)
- 洛谷 (https://www.luogu.org/problem/lists?name=&orderitem=pid&tag=3)

有建议？或者发现什么错误？在Github上开一个issue (https://github.com/adambard/learnxinyminutes-docs/issues/new)，或者你自己也可以写一个pull request！

原著Akashdeep Goel，并由0个好心人 (https://github.com/adambard/learnxinyminutes-docs/blame/master/zh-cn/dynamic-programming-cn.html.markdown)修改。

(https://creativecommons.org/licenses/by-sa/3.0/deed.en_US) © 2017 Akashdeep Goel (http://github.com/akashdeepgoel)

Translated by: EtaoinWu (https://github.com/EtaoinWu)