chap 2组合优化问题的求解方法

求解方法

一般面对组合优化问题有3个方向：

1.设计多项式时间算法（对于P问题）

2.对复杂的问题，看看能不能在指数时间求最优解，或者在多项式时间内求近似解，研究特殊可解性

3.证明该问题为NP-难问题

能在指数时间内求解最优解的方法：

整数规划法，分支定界法，动态规划法等等

能在多项式时间内求解近似解的方法

近似算法，近似方案，启发式算法

（贪婪算法，线性规划松弛，局部搜索算法等）

整数规划

整数规划也是NP Hard

组合优化很多问题都可以用整数规划来描述，建立整数规划的主要步骤有**确定决策变量**，**给出目标函数**，**列出约束条件**

整数规划求解也是NP-Hard，以下给出几种问题的整数规划形式：

背包问题

现在有n件物品，满足对于物品j，价值为p_j,大小为w_j，背包的容量为C。我们要求选择若干物品放入背包，使得在放入背包的物品大小之和不超过背包容量的前提下，使得放入背包的物品价值之和尽量大。

因此决策变量为

$x_j=\begin{cases}1,\text{放入第j种物品},\\0,其它\end{cases}$

因此写成整数规划可以是

$$max \sum_{j=1}^{n} p_j * x_j$$

$$s.t. \sum_{j=1}^{n} w_j x_j \leq C$$

$$x_j = 0,1, j = 1, \ldots, n$$

指派问题：

注意指派问题是一个P问题。

记决策向量为$x=(x_{11},x_{12},..,x_{nn})^T$,满足以下条件

1. 一个人需要指派1项任务
2. 一项任务需要1个人完成

因此，我们的整数规划可以写成：

$$min\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}x_{ij}$$
$$s.t.\sum_{j=1}^{n}x_{ij}=1,\forall i=1,\dots,n$$
$$\sum_{i=1}^{n}x_{ij}=1,\forall j=1,\dots,n$$
$$0\le x_{ij}\le 1,i,j=1,\dots,n$$

如果矩阵的所有子式都是$0,+-1$，那么该矩阵就是全幺模矩阵。

系数矩阵为全幺模矩阵的整数规划的松弛线性规划的最优解必然为整数解

TSP问题的整数规划

决策变量为$x_{ij}=\begin{cases}1,\text{离开城市i到达城市j},\\0,其它\end{cases}$

那么整数规划问题可以写成：

$$min\sum\sum c_{ij}x_{ij}$$
$$s.t.\sum_{i}x_{ij}=1,i=1,\dots,n$$
$$\sum_{j}x_{ij}=1,j=1,\dots,n$$
$$x_{ij}=0,1,i,j=1,..,n$$

但是这个显然是有问题的，也就是说其中可能出现了子环游，比如

$$x_{13}=x_{35}=x_{51}=x_{24}=x_{42}=1$$

这样也满足条件，但是它不是一个环游。可以通过增加n-1个实决策变量$u_2,...,u_n$与$(n-1)^2$个约束：

$$(n-1)x_{ij}+u_i-u_j\le n-2,i,j=2,3,\dots,n$$

来消除。为什么呢？

证明如果解有子环游，那么这个约束就无法被满足

假设解含有子环游，则必然有一个子环游$(i_1,...,i_k)$不经过城市1.

如果子环游有1个城市，那么$x_{ii}=1$,同时$(n-1)x_{ii}+u_i-u_i=n-1>n-2$

如果子环游至少有2个城市，那么存在$i_{k+1}=i_1,x_{i_ji_{j+1}}=1$.

那么我们有此时

$\sum_{j=1}^k((n-1)x_{i_ji_{j+1}}+u_{i_j}-u_{i_{j+1}})=\sum((n-1)+u_{i_j}-u_{i_{j+1}})$

因为有$i_{k+1}=i_1$,因此此时后面的加就是一个环，那么有$k(n-1)>k(n-2)$

但是应该是小的，因此成立

因为u_i可以取任意常数，因此我们可以对任意的可行环游，如果城市i是第k个经过的城市，那么我们可以取$u_i=k$,而同时x_{ij}的取值与前面定义一样。

此时我们有约束

$(n-1)x_{i,j}+u_i-u_j\leq n-2,i,j=2,3....,n$

动态规划与最优化原理：

什么是动态规划呢?

动态规划是求解多阶段决策优化问题的一种数学方法与算法思想。

基本思路是将需要求解的实例转换为规模较小的实例，然后利用递推关系导出两者最优解之间的关系，然后从初始条件出发，逐步求最优解。

最优化原理

一个过程的最优决策具有以下性质：无论初始状态s_0以及初始决策x_1应该如何确定，之后的决策$(x_2^*,...,x_n^*)$对于以x_1^*所造成的状态s_1^*为初始状态的后面过程而言，必然为最优策略。

这就是说对于一个问题，我们可以先将其用优化x_1的小实例构造，然后一步一步解决后面的最优问题，因为它可以逐步求最优解。

因此，一个n阶段过程的最优策略可以这样构成：首先求出以初始决策x_1所造成的状态s_1为初始状态的$n-1$阶段子过程最优策略，然后在附加第一阶段效益的情形下，从所有可能的初始决策中得到的解中选择最优解。

背包问题的动态规划解法（普通意义下的NP-C问题）

依次考虑所有的物品，每个物品对应动态规划问题的一个阶段，决策为放入不放入

构造一系列实例 **I(k,w),0<=k<=n,0<=w<=C**，其中背包容量是w，物品数是k（排序包含原实例中前k个物品）

记$C^*(k,w)$为当前实例的$I(k,w)$的最优解，原实例的最优值为$C^*(n,C)$

那么我们的递推关系可以怎么写：

假如我们有k个物品，而容量为w，那么此时如果说第k个物品的权重$w_k>w$，那么这个结果就和$C^*(k-1,w)$相同。

如果说可以放进背包呢？那当前的最优解有2种可能，一种是没有放进去，那么最优解就是$C^*(k-1,w)$，如果放进去了，那么剩余容量就是$C^*(k-1,w-w_k)$，也就是说，可以放入的物品必然是$I(k-1,w-w_k)$的最优解中放入的物品。因此我们的最优解必然在$max(C^*(k-1,w),C^*(k-1,w-w_k)+p_k)$这个过程中。

因此，对于当前情况$C^*(k,w)$可以进行如下的推导：

\begin{cases}C^*(k-1,w),if\ w_k>w,\\max\{C^*(k-1,w),C^*(k-1,w-w_k)+p_k\},el

同时，这个递归过程的初始条件满足

$C^*(0,w)=0,w=0,...,C,C^*(k,0)=0,k=0,...,n$

因此经过循环就可求到*I(n,C)*

动态规划DPS的时间复杂度为$O(nC)$,背包问题的实例规模是$n+\lceil log_2L\rceil$,L为最大数。因此DPS是关于实力规模与最大数的多项式时间算法，是伪多项式时间算法（只要最大数不是很大）

背包问题是普通意义下的NP-完全问题（因为存在伪多项式时间算法）

自己跑一遍：

层层递归即可，跑完。

动态规划DPV：

（是求一个最小值的东西）

问题：

记录$y(k,q)$为在前k个物品中，选出若干物品，使得其价值之和恰好为q的前提下，选出物品大小之和所能取到的最小值。

如果价值之和恰为q的物品子集不存在，那么我们让$y(k,q)=C+1$

那么我们显然它的递推关系为：

$$
y(k,q)=\begin{cases}y(k-1,q),if\ p_k>q,\\min\{y(k-1,q),y(k-1,q-p_k)+w_k\},else\
$$

时间 c'c复杂度一样为最大数的多项式，也就是说为$O(nP)$,是伪多项式时间算法

背包问题的分支定界法

实例求解过程可以用一个高度为n的二叉树进行描述，二叉树的每一层对应一个物品，最后一层的每一个节点对应一个解。每个二叉树出两个分枝，连接两个子节点。这两个子节点分别对应着该节点下一层所对应的物品放不放入背包。

如果我们只是分支的话，大概有2^n条可行解。这个显然是一个指数时间算法。因此我们需要一个剪纸的过程。

减枝：

1.该枝不存在可行解（也就是放入的物品大小超过了容量，后面都减掉）

2.**断定该枝不存在最优解**（应该如何判断呢？）

对于下界问题：

任意一个已获得可行解的目标值都是最优值的下界。

对于每一枝，求该枝所有可行解目标值的上界。如果上界不大于下界，那么该枝不存在更好可行解（可以减掉了）

一般上界如何求呢？一般是求该枝对应的整数规划的松弛线性规划的最优值。

松弛线性规划问题：

如果我们将物品按照价值密度的非增顺序排列，然后一个一个放就可以了。

最后一个数目可以取$x_j^*=(C-\sum_{i=1}^{j-1}w_i)/w_j$

背包的最优解可能因为不放入物品j，而放入之后大小适中的物品，也可能因为放入物品j而不得不取出之前的部分物品。

因此此时我们可以对该分支进行定界。

背包问题的下界

因此分枝定界法，从根节点开始，按照深度优先或广度优先看每个节点。对于每个节点，计算上界与下界，如果下界大于当前下界，那么修正当前下界。

如果所有节点都检查完毕而且不能再分支了，那么当前下界就是实例的最优值。

近似算法，启发式算法

最坏情况界：

最坏情况界的证明方法：

上界，就是最坏情况界不超过r：

对任意实例I,满足$C^A(I)\leq rC^*(I)$

对很多优化问题，不一定能得到简单表达式。那么我们需要给出一些容易计算的下界，使得$C^*(I)\geq C^*_{LB}(I)$,然后再证明$C^A(I)\leq rC^*_{LB}(I)$

下界，就是至少为r：

构造实例，或者是一族实例，使得$C^A(I)/C^*(I)=r\ or \rightarrow r$

如果上界与下界相等，那么界就是紧的。

背包问题的近似算法：

贪婪算法

1.按密度非增排序

2.一个一个加，直到放不下

注意，我们可以构造实例，使得最坏情况界趋向于∞

算法改进：

将当前得到的值与物品的价值最大值比，取max

$$
C^C(I)=max\{\sum_{i=1}^{j-1}p_j,p_{max}\}
$$

证明复合算法的最坏情况界为2

(注意，因为这里是求最大值，所以上界r应该满足对任意实例有$C^*(I)\leq r*C^C(I)$),也就是被bound住了，意味着我们的最大值确实蛮大的。如果是求最小值的话，应该是$C^A(I)\leq rC^*(I),r>=1$，这也意味着我们求得的最小值确实比较小

怎么证明呢?

首先证个上界：

首先最优解肯定小于松弛规划最优解，我们要找到一个r，使得

$C^*(I)\leq rC^A(I)对所有的实例I成立$

可知最优解肯定小于松弛规划最优解，也就是说

$C^*(I)\leq \sum_{i=1}^{j-1}p_i +p_j*(C-\sum w_i)/w_j $

那么我们把这个界再扩大一下

$C^*(I)<=\sum p_i +max(p_j)$

而同时我们有$\frac{\sum p_i +max(p_j)}{max{\sum p_i,max(p_j)}}<=2$

这是显然的

因此最坏情况界上界为2

构造一个/一组实例使得它们接近2:

物品	**1**	**2**	
价值	1+eps	M	M
大小	1	M	M
背包容量	2M		

我们可以看到，最佳的当然是2M，但是我们G法得到的是M+1+eps

那么它们比在M极大的时候趋向于2

平均情况界

难近似性：

将一个问题称为难近似的，即如果$P\neq NP$，那么不存在最坏情况界为有限常数的多项式时间近似算法。

证明TSP问题不存在最坏情况界为常数的多项式时间近似算法。

（一般涉及到P不等于NP的，都是通过归约来证，就是如果存在了，那么某个NP-C问题就存在多项式时间解，也就是P=NP了）

我们假设TSP问题存在最坏情况界小于M的多项式时间算法A

任意给出NP-C的HC实例I_{HC}：图G=(V,E)$.

构造TSP的实例I_{TSP}如下：

城市数目$n=|V|$

城市距离：1（如果有边）（n*M-n+1）（如果没边）

如果G中存在H圈，TSP问题，那么$C^*(I_{TSP})=n$

如果G中不存在H圈，那么任意一个TSP环游中一定至少有一个距离为n*M-n+1的相邻城市，也就是

$C^*(I_{TSP})>=(n-1)+n*M-(n-1)=n*M$

那么我们如何构造一个iff实例呢？

用算法A求解I_{TSP}，那么$\frac{C^{A}(I_{TSP})}{C^*(I_{TSP})}<M$

因为我们求的是一个最少环游问题，也就是此时满足求解的不太大

那么此时有$C^*(I_{TSP})>C^A(I_{TSP})/M$

如果说此时$C^A(I_{TSP})>n*M$

那么因为最坏情况界的bound下，我们有$C^*>n$

此时不存在H圈，不然存在H圈，此时HC问题是多项式问题，矛盾了

度量TSP问题（城市距离满足三角不等式）

p40-46页还没有搞懂。

顶点覆盖问题：

定义：

给定图$G=(V,E)$，顶点集V的子集V'称为G的顶点覆盖，如果E的每条边至少有一个端点在V'中。我们一般称集合V覆盖了G的边。

最小权顶点覆盖问题（WVC）：

给定图$G=(V,E)$，并知道每个顶点的权，求G的总权和最小的顶点覆盖

VC是NP-H问题

证明$3SAT\leq ^p_m VC$

如何把最小权顶点覆盖问题的整数规划写出来呢？

决策变量：

$x_i=1(if v_i\in V')0(else)$

整数规划：

$$
min\ \sum_{i=1}^nw_ix_i\\

s.t.\ x_j+x_k>=1,\forall\ v_jv_k\in E\\
x_i=0或1
$$

松弛线性规划(LP)

$0<=x_i<=1$

线性规划松弛算法：

求解LP，令$V'=\{v_i|x_i^{LP}>=1/2\}$

首先证明V'是顶点覆盖

证明顶点覆盖问题的松弛算法的最坏情况界小于2

（反证法，证明小于1，矛盾）

$$
\sum_{v_i'\in V'}w_i=\sum_{v_i'\in V'}w_i*I(x_i^{LP}\geq\frac{1}{2})\leq 2*\sum
\leq 2*\sum_{i=1}^n w_ix_i^*
$$

近似方案

（定义，性质）

多项式时间近似方案（PTAS）与完全多项式时间近似方案(FPTAS)：

算法族$\{A_\epsilon\}$被称为多项式时间近似方案，如果对于任意给定eps，算法族的最坏情况界为1+eps，且时间复杂性为$f(n)=O(p(n))$

算法族被称为完全多项式时间近似方案，如果对给定的eps，算法的最坏情况界为$1+eps$，且A_{eps}的时间复杂性为$f(n,eps)=O(p(n,\frac{1}{eps}))$

也就是和实例规模以及近似比倒数的多项式

注意，时间复杂性为$O(n^{1/eps})$是一个PTAS，但是不是一个FPTAS

因为关于1/eps不是一个多项式

背包问题的PTAS

复合思想

FPTSA

微缩

（伪多项式时间算法）

背包问题的近似方案：

PTAS：

有限枚举法，误差来源于从有限到全部的过程

FPTAS呢：

设计思路是动态规划法（伪多项式时间算法）

误差来源是用缩微实例最优解为近似解

近似方案的存在性

强NP-Hard问题的近似方案

假设A是一个极大化某个问题的强NP-Hard问题（也就是不存在伪多项式时间算法），存在二元多项式q(x,y)，使得$C^*(I)<=q(size(I),max(I))$，那么A不存在FPTAS，除非P=NP

积划分问题：

给定正整数集A，是否能把它划分为两个元素乘积相等的子集

极小化划分问题：

给定正整数集合A，求一个划分，使得划分的2个子集元素乘积的最大值最小

度量TSP也是难近似的