效率异质型员工项目调度算法研究

柳春锋[1,2],杨善林[1]
LIU Chunfeng[1,2],YANG Shanlin[1]
1.合肥工业大学 管理学院,合肥 230009
2.浙江旅游职业学院,杭州 311231
1.School of Management,Hefei University of Technology,Hefei 230009,China
2.Tourism College of Zhejiang,Hangzhou 311231,China

LIU Chunfeng,YANG Shanlin.Study on algorithm for project scheduling with heterogeneous efficiency workforce.Computer Engineering and Applications,2011,47(13):18-21.

Abstract: Project scheduling with heterogeneous efficiency workforce to minimize project duration is considered,and the corresponding integer linear programming is presented.To solve the NP-hard problem,a priority rule-based heuristic is proposed,which selects the prior task-employee pair according to precedence constraints and priority rule to assign the task at each iteration until all tasks are scheduled.A Hybrid Simulated Annealing(HSA) can be constructed through applying the heuristic to generate initial schedule,choosing the swap neighborhood structure and insertion neighborhood structure to obtain neighbor schedule,and employing the revised forward recursion algorithm to compute objective function value.Numerical experiments are conducted to show that the proposed HSA performs well with respect to accuracy and efficiency of solution.
Key words: project scheduling;heterogeneous efficiencies;heuristic;simulated annealing;precedence constraints

摘 要: 研究了员工具有异质效率、最小化项目工期的项目调度问题,并建立了相应的整数线性规划模型。为解决此NP-hard问题,提出了基于优先规则的启发式算法,其在每次迭代中根据优先约束和优先规则选择优先任务员工对以分配任务,直至所有任务都完成调度。通过应用启发式算法生成初始调度,选用交换邻域结构和插入邻域结构产生邻域调度,并使用改进的前向递归算法求解目标函数值,构造出混合模拟退火算法。数值实验显示该算法能快速准确地进行寻优。
关键词: 项目调度;异质效率;启发式算法;模拟退火;优先约束
DOI:10.3778/j.issn.1002-8331.2011.13.006 **文章编号:**1002-8331(2011)13-0018-04 **文献标识码:**A **中图分类号:**O221;F406.2;TP301

1 前言

项目调度和员工配置问题(Project-Scheduling and Staff-Allocation Problem,PSSAP)是一个近年来备受关注的优化问题,广泛出现于现代项目管理中,如软件开发项目、建筑项目等,这些管理活动中人力资源成本占总成本比例很大。

一些研究人员采用精确算法、启发式算法和智能算法来求解不同情形的PSSAP问题,如Bellenguez-Morineau和Néron[1]提出分支定界方法来解决多技能员工调度问题,使活动按排满足技能需求和供应。Alfares和Bailey[2]考虑了建筑任务和员工调度问题以最小化项目和人员成本。他们提出了一种整数线性规划模型和启发式算法来分配任务给一周休息两天的员工。Bassett[3]提出了一种递归方法和启发式算法来分配项目和任务,使得有效利用员工的专业技能并最小化研发外包。Alba和Chicano[4]通过遗传算法来解决不同软件开发项目情形,其中员工是多技能的,任务也需要多种技能。

上述文献以及经典的资源受限项目调度问题(Resource-Constrained Project Scheduling Problem,RCPSP)通常假设相同技能的人力资源是效率同质的,但现实生产中相同技能的新老员工往往是效率异质的。本文提出一种启发式算法及在此基础上的混合模拟退火算法来求解效率异质型员工项目调度问题。

2 问题描述

给定一个项目由 n 个任务组成,由 m 个员工来完成;不仿假设每项任务需要一种技能(因为多技能任务可以分解为单技能任务),每个员工具有一种技能;有技能处理任务 j 的员工集合为 E_j,每个员工 $v \in E_j$ 处理任务 j 的效率不同,处理时间为 p_{jv};每项任务只需要一个员工处理且不可中断,每个员

基金项目: 国家自然科学基金重点项目(the Key National Natural Science Foundation of China under Grant No.70631003);国家高技术研究发展计划(863计划)重点项目(No.2008AA042901)。
作者简介: 柳春锋(1977—),男,博士研究生,副教授,研究方向:生产调度、项目管理、旅游优化与决策等;杨善林(1948—),男,教授,博士研究生导师。E-mail:lcf_spring@163.com
收稿日期:2010-12-29;**修回日期:**2011-03-17

工在同一时间最多处理一项任务；此外，任务之间具有优先约束关系，即一项任务在其紧前任务完成后才能开始。令两个处理时间为0的亚元任务 s、t 分别表示项目的开始和结束。目标是要寻找一个可行的调度能最小化项目工期（即时间表长）$C_{\max}$。此问题通常可以用三元组表示为 $R^{'}m|prec|C_{\max}$，其中，$R^{'}m$ 表示 m 个效率异质的员工，$prec$ 表示任务之间具有优先约束关系。因为根据Ullman[5]的证明，单位处理时间的同型机问题 $P|p_i=1;prec|C_{\max}$ 是NP-hard问题，那么问题 $R^{'}m|prec|C_{\max}$ 显然是NP-hard问题，本文寻求它的近优解。

问题 $R^{'}m|prec|C_{\max}$ 可表示为如下整数线性规划模型[6]：

$$\text{Min } C_{\max}=\max\{FT_j|j\in J\} \tag{1}$$

$$\text{s.t.} \sum_{v=1}^{m}\sum_{r=1}^{UB}x_{jvr}=1 \ (\forall j\in J) \tag{2}$$

$$\sum_{j=1}^{n}x_{jvr}\le 1 \ (\forall r\in R,\forall v\in M) \tag{3}$$

$$\sum_{i=1}^{n}x_{ivr}-\sum_{j=1}^{n}x_{j,v,r-1}\le 0 \ (\forall v\in M,\forall r\in\{2,3,\cdots,UB\}) \tag{4}$$

$$FT_j-FT_i+L(2-x_{jvr}-x_{i,v,r-1})\ge p_{jv}$$
$$(\forall i,j\in J,i\ne j,\forall v\in M,\forall r\in\{2,3,\cdots,UB\}) \tag{5}$$

$$FT_j\ge\sum_{r=1}^{UB}p_{jv}x_{jvr} \ (\forall j\in J,\forall v\in M) \tag{6}$$

$$FT_j-FT_i\ge\sum_{v=1}^{m}\sum_{r=1}^{UB}p_{jv}x_{jvr} \ (\forall i\in P_j) \tag{7}$$

$$x_{jvr}\begin{cases}\in\{0,1\} \ (\forall j\in J,\forall v\in E_j,\forall r\in R)\\ =0 \ (\forall j\in J,\forall v\in M,v\notin E_j,\forall r\in R)\end{cases} \tag{8}$$

$$FT_j\ge 0 \ (\forall j\in J) \tag{9}$$

模型的输入参数包括：

J：任务集合，$J=\{1,2,\cdots,n\}$。

M：员工集合，$M=\{1,2,\cdots,m\}$。

E_j：有技能处理任务 j 的员工集合，$j\in J$。

p_{jv}：员工 v 处理任务 j 需要的时间，$j\in J$，$v\in E_j$。

UB：员工处理任务的最大时间段数（假设一个时间段处理一个任务，那么 $UB=n-m+1$，即能最大程度地利用员工使得所有的员工都有任务处理）。

R：时间段集合，$R=\{1,2,\cdots,UB\}$。

P_j：任务 j 的紧前任务集合，$j\in J$。

L：一个大的正整数。

模型的决策变量包括：

x_{jvr}：如果员工 v 在第 r 个时间段处理任务 j，则 x_{jvr} 等于1，否则等于0，$j\in J$，$v\in E_j$，$r\in R$。

FT_j：任务 j 的完成时间，$j\in J$。

目标方程(1)最小化项目工期。约束式(2)保证每项任务能被某个员工在某个时间段处理。约束式(3)保证在每个时间段最多有一个任务被处理。约束式(4)保证只有当员工在某个时间段有任务处理时才能在下个时间段处理另外的任务。约束式(5)保证员工完成某项任务的时间至少要等于他完成前一时间段任务的时间加上处理该任务的时间。约束式(6)保证每项任务的完成时间不少于其实际的处理时间。约束式(7)保证了任务间的约束关系。约束式(8)和式(9)定义了决策变量的类型。

3　基于优先规则的启发式算法

基于优先规则的启发式算法（PRBHA）包括 n 次迭代。在每次迭代中，根据任务优先约束选择可行任务，然后由给定的优先规则选择优先任务员工对实现任务分配，直至所有任务都完成调度。为此作如下定义：

C：完成集，即已经调度的任务集。

D：决策集，即紧前任务都已调度的未调度任务集。

I_v：员工 v 开始空闲的时间，$v\in M$。

ft_{jv}：任务 j 由员工 v 处理的假设完成时间，$j\in J$，$v\in E_j$。

θ_j：任务 j 的紧前任务完成时间的最大值，$j\in J$。

(j^*,v^*)：优先任务员工对，$j^*\in J$，$v^*\in E_{j^*}$。

算法1给出了伪代码描述的具体过程。第1步初始化变量。第2步进行 n 次迭代，即依次分配 n 个任务。在每次迭代中，先计算决策集 D，再计算其中每个任务 j 由不同员工 $v\in E_j$ 处理的假设完成时间 ft_{jv}，然后选择假设完成时间最小的任务员工对 (j^*,v^*)，并记录任务 j^* 的实际完成时间 FT_{j^*}、更新员工 v^* 开始空闲的时间。第3步计算项目工期 $C_{\max}$。

算法1　PRBHA

$D=\{j|j\notin C,P_j\subseteq C\}$

1.初始化：$FT_s=0$，$C=\{s\}$，$I_v=0,\forall v\in M$。

2.FOR　$g=1$　TO　n

　　2.1.计算　D；

　　2.2. $\theta_j=\max\{FT_h|h\in P_j\}$，$\forall j\in D$

　　2.3. $ft_{jv}=\max\{\theta_j,I_v\}+p_{jv}$，$\forall j\in D,\forall v\in E_j$

　　2.4. (j^*,v^*)：$ft_{j^*v^*}=\min\{ft_{jv}|j\in D,v\in E_j\}$

　　2.5. $FT_{j^*}=ft_{j^*v^*}$，$I_{v^*}=FT_{j^*}$

　　2.6. $C:=C\cup\{j^*\}$

3. $C_{\max}=\max\{FT_j|j\in J\}$

4　混合模拟退火算法

4.1　基本思想

模拟退火算法（SA）是一种广泛应用于组合优化的邻域搜索技术。本文所提的混合模拟退火算法（HSA）应用基于优先规则的启发式算法（PRBHA）生成一个初始调度，并设计了邻域调度的两种生成方法（交换邻域结构和插入邻域结构），目标函数值的计算采用改进的前向递归算法（RFRA）（见算法2）。在每一阶段，如果新生成的邻域调度 x_1 的目标函数值不大于当前调度 x_c 的目标函数值，则接受 x_1 为当前调度；否则以一定的概率 p 接受 x_1 为当前调度，p 随 x_1 与 x_c 目标函数值之差的增加和温度的降低而减少。温度间断性地从高温降低到接近零度。因此，在HSA开始阶段，许多比当前差的调度都可能被接受，到最后阶段，基本上只接受比当前好的调度，并收敛于近优调度。

4.2　邻域调度的生成

给定一个调度 x，其邻域调度可由下列方式之一获得：

(1)交换邻域结构

对于调度 x，随机选择两个具有相同技能的员工 $e1$ 和 $e2$，再随机选择由 $e1$ 处理的任务 $j1$、由 $e2$ 处理的任务 $j2$，变换为由 $e1$ 处理任务 $j2$、由 $e2$ 处理任务 $j1$，其余保持不变。通

过算法RFRA计算新调度 x_1 的目标函数值 $OFV(x_1)$。

(2)插入邻域结构

对于调度 x,随机选择两个具有相同技能的员工 $e1$ 和 $e2$,再随机选择由 $e1$ 处理的任务 $j1$,变换为由 $e2$ 处理任务 $j1$,其余保持不变。通过算法RFRA计算新调度 x_1 的目标函数值 $OFV(x_1)$。

算法2 RFRA

1.初始化未分配任务集 $U=\{1,2,\cdots,n\}$,以及所有员工开始空闲的时间 $I_v=0,\forall v\in M$。

2.根据3从 U 中随机选择一项任务进行分配,直至所有的任务都被分配。

3.分配任务 j 给员工 k(根据调度可知 k 值)。

3.1.如果紧前任务 i 的完成时间未定,针对任务 i 递归地执行3。

3.2.计算任务 j 的开始 $ST_j=\max\{\theta_j,I_k\}$ 和完成时间 $FT_j=ST_j+p_{jk}$。

3.3.更新未分配任务集和员工 k 开始空闲的时间:$U:=U\backslash\{j\},I_k=FT_j$。

4. $C_{\max}=\max\{FT_j|j\in J\}$

4.3 HSA的具体实施

HSA的具体过程见算法3,实施细节描述如下:

(1)算法参数

初始温度 T、最终温度 T_f 和冷却比率 α 分别取经验值200、0.5和0.95。考虑到平均每项任务可由 m/k 个员工来处理(k 为整个项目所需要的技能数),因此特定温度下的最大迭代次数 $L_{\max}$ 应与 m/k 成正比,也与 n 成正比,所以 $L_{\max}$ 取经验值 nm/k。

(2)降温条件

当特定温度的迭代次数 $Total\geq L_{\max}$,温度以 $T:=\alpha\times T$ 下降。

(3)终止条件

令到目前为止最好的调度为 x_{best},当温度降到最终温度,或连续3个温度下 x_{best} 都没有改变,则终止算法。为此,设置特定温度下 x_{best} 被改变的计数器 $Change$,以及整个算法中 x_{best} 没有被改变的计数器 $Unchange$(即特定温度下 x_{best} 没有被改变,则 $Unchange$ 累加1)。

算法3 HSA

1.初始化

1.1. $T=200,T_f=0.5,\alpha=0.95,Total=0$,$Change=0$,$Unchange=0$

1.2.获得初始调度作为当前调度 x_c,计算目标函数值 $OFV(x_c)$。

2.WHILE $(T>T_f)$ DO

2.1.WHILE $(Total<L_{\max})$ DO

2.1.1.产生一个随机数 $r_1\sim U(0,1)$,

IF $(r_1>0.5)$

选择一个交换邻域调度 x_1,计算 $OFV(x_1)$

ELSE

选择一个插入邻域调度 x_1,计算 $OFV(x_1)$

2.1.2.IF $(OFV(x_l)\leq OFV(x_c))$

$x_c=x_1$

IF $(OFV(x_c)\leq OFV(x_{best}))$

$x_{best}=x_c,Change:=Change+1,Unchange=0$

ELSE

产生一个随机数 $r_2\sim U(0,1)$,令 $\Delta=OFV(x_1)-OFV(x_c)$

IF $(e^{-\Delta/T}>r_2)$

$x_c=x_1$

2.1.3. $Total:=Total+1$

2.2.IF $(Change=0)$

$Unchange:=Unchange+1$

2.3.IF $(Unchange=3)$

返回 $OFV(x_{best})$ 为最终解,结束。

2.4. $T:=\alpha\times T$,$Total=0$,$Change=0$

3.返回 $OFV(x_{best})$ 为最终解,结束。

5 数值实验

为了评价算法HSA的性能,以及算法PRBHA在HSA中生成初始调度的作用,令SA表示相应的随机生成初始调度的模拟退火算法。比较HSA和SA的性能,使用4个影响因子,包括任务数(n)、员工数(m)、技能数(φ)和处理时间(p_{jv})。

为检验变量 n、m 和 φ 的影响,选择4个不同的 n 值($n=20,40,60,80$),4个不同的 m 值($m=10,20,30,40$),以及4个不同的 φ 值($\varphi=15,20,25,30$)。为检验变量 p_{jv} 的影响,选择4个不同的 p_{jv} 的分布,包括 $p_{jv}\sim$DU[10,20]、$p_{jv}\sim$DU[10,30]、$p_{jv}\sim$DU[10,40] 和 $p_{jv}\sim$DU[10,50],这里 DU[a,b] 代表从 a 到 b 的均匀分布。

实验采用Hall和Posner[7]提出的产生任务优先约束关系的方法。为此先定义:有任务 i、j,Pr 为概率,D=优先约束图密度=Pr {存在 $arc(i,j)$,即 i 是 j 的前任任务$|\ \forall 1\leq i<j\leq n$},$P_{ij}=Pr$ {存在 $arc\_immediate(i,j)$,即 i 是 j 的紧前任务$|\ \forall 1\leq i<j\leq n$}。

根据Hall和Posner的证明,当 $D\in(0,1)$ 时,有

$$P_{ij}=\frac{D(1-D)^{j-i-1}}{1-D\left[1-(1-D)^{j-i-1}\right]} \tag{10}$$

因此,如果已知优先约束图密度 D,可计算存在 $arc\_immediate(i,j)$ 的概率 P_{ij}。从 $[0,1]$ 均匀分布中产生一个随机数 r_{ij},如果 $r_{ij}<P_{ij}$,则优先约束图中存在 $arc\_immediate(i,j)$。D 反映了优先约束图中任务间优先关系的强弱,D 越大,优先关系越强(即存在优先关系的概率越大),反之,优先关系越弱(即存在优先关系的概率越小)。极端情况,当 $D=1$ 时,优先关系为链式类型;当 $D=0$ 时,任务间无优先关系。实验取最一般的情况 $D=0.5$。

为了其他研究者易于比较,取一个简单易行的下界 LB:首先令每项任务 j 的最小处理时间 $p_j^{\min}$ 为能够处理该任务的员工中对其处理时间的最小值,即 $p_j^{\min}=\min\{p_{jv}|v\in E_j\}$,然后根据传统的关键路径法求出项目工期作为下界。

下面进行4组实验。第一组实验假定 $m=10$、$\varphi=5$、

p_{jv}~DU[10，20]，观察不同任务数(n)的SA和HSA的性能(见表1)；第二组实验假定$n=60$、$\varphi=5$、p_{jv}~DU[10，20]，观察不同员工数(m)的SA和HSA的性能(见表2)；第三组实验假定$n=80$、$m=40$、p_{jv}~DU[10，20]，观察不同技能数(φ)的SA和HSA的性能(见表3)；第四组实验假定$n=20$、$m=10$、$\varphi=5$，观察不同处理时间(p_{jv})分布的SA和HSA的性能(见表4)。算法在VC++ 6.0编译环境下用C++语言实现。实验环境为：Intel® Pentium® 4、CPU 2.40 GHz、内存496 MB、操作系统Microsoft Windows XP SP3。

表1~4中符号定义如下：算法SA的相对误差$SA_Gap=(C_{SA}-LB)/LB\times100$，算法HSA的相对误差$HSA_Gap=(C_{HSA}-LB)/LB\times100$，其中$C_{SA}$、$C_{HSA}$分别表示SA和HSA的解。另外，对于表格中每个相同问题情形，进行10次不同随机数据实验，*MIN*、*MAX*、*AVE*分别对应它们的最小值、最大值和平均值。*Decrease_AveGap*表示平均*HSA_Gap*比平均*SA_Gap*的降幅百分比。*SA_CPU*、*HSA_CPU*分别表示SA和HSA的平均运行时间，*Decrease_CPU*表示*HSA_CPU*比*SA_CPU*的降幅百分比。

由表1可以看出，当员工数(m)、技能数(φ)和处理时间(p_{jv})分布相同时，平均*HSA_Gap*随任务数(n)的增加呈上升趋势，这是因为规模越大，算法寻优的难度也越大，同时下界*LB*偏离最优值的距离也越远。但*Decrease_AveGap*随n的增加呈显著增加趋势，说明在HSA中算法PRBHA生成初始调度有助于全局寻优。

由表2可以看出，当任务数(n)、技能数(φ)和处理时间(p_{jv})分布相同时，平均*HSA_Gap*随员工数(m)的增加而降低，这是因为员工数增加使平均每项任务分配的可选员工增多，降低了员工等待的可能性，从而减小了项目工期。

由表3可以看出，当任务数(n)、员工数(m)和处理时间(p_{jv})分布相同时，平均*HSA_Gap*随技能数(φ)的增加而降低，这是因为本文中技能数的增加意味着平均每项任务分配的可选员工减少，增加了员工等待的可能性，从而增加了项目工期。

由表4可以看出，当任务数(n)、员工数(m)和技能数(φ)相同时，平均*HSA_Gap*随处理时间(p_{jv})分布的增加呈上升趋势，这主要因为文中下界*LB*把任务处理时间松弛为可能的处理时间的最小值，所以当处理时间分布扩大时，*LB*离最优值的距离会变远，从而*HSA_Gap*会增大。

由表1~4可以看出，平均*HSA_Gap*比平均*SA_Gap*的降幅可达11%~98%之间，且*HSA_CPU*比*SA_CPU*的降幅可达10%~51%之间，说明算法PRBHA的应用既提高了模拟退火的搜索质量又降低了搜索时间。另外，平均*HSA_Gap*的统计值(即平均值)为10.17%，大部分最小*HSA_Gap*为0(即得到最优解)，且*HSA_CPU*的统计值为0.81 s，说明针对本文的调度问题，HSA能快速搜索到高质量的解。

6　结论

考虑了员工具有异质效率、最小化项目工期的项目调度问题。对该问题建立了整数线性规划模型，提出了一种混合模拟退火算法(HSA)。该算法应用基于优先规则的启发式算法(PRBHA)生成初始调度，随机选择交换邻域结构或插入邻

表1　不同任务数(n)的SA和HSA的性能比较表(m=10，φ=5，p_{jv}~DU[10，20])

n	*SA_Gap*/(%)			*HSA_Gap*/(%)			*Decrease_AveGap*/(%)	*SA_CPU*/s	*HSA_CPU*/s	*Decrease_CPU*/(%)
	MIN	*MAX*	*AVE*	*MIN*	*MAX*	*AVE*				
20	0.67	34.58	11.36	0	39.25	10.02	11.86	0.09	0.04	49.94
40	2.00	62.10	19.48	0	59.65	16.12	17.26	0.32	0.25	22.45
60	14.81	70.90	42.80	0.88	58.99	20.17	52.86	0.99	0.57	42.05
80	35.91	110.14	72.05	15.19	64.43	34.99	51.43	1.68	1.20	28.78

表2　不同员工数(m)的SA和HSA的性能比较表(n=60，φ=5，p_{jv}~DU[10，20])

m	*SA_Gap*/(%)			*HSA_Gap*/(%)			*Decrease_AveGap*/(%)	*SA_CPU*/s	*HSA_CPU*/s	*Decrease_CPU*/(%)
	MIN	*MAX*	*AVE*	*MIN*	*MAX*	*AVE*				
10	9.14	61.45	30.87	2.81	30.49	11.56	62.57	0.89	0.59	33.62
20	19.34	51.70	34.36	0	25.45	6.09	82.29	1.56	0.97	37.71
30	16.52	43.31	26.29	0	11.02	2.30	91.24	2.59	1.57	39.22
40	22.67	46.74	30.49	0	5.43	0.64	97.91	3.94	1.94	50.81

表3　不同技能数(φ)的SA和HSA的性能比较表(n=80，m=40，p_{jv}~DU[10，20])

φ	*SA_Gap*/(%)			*HSA_Gap*/(%)			*Decrease_AveGap*/(%)	*SA_CPU*/s	*HSA_CPU*/s	*Decrease_CPU*/(%)
	MIN	*MAX*	*AVE*	*MIN*	*MAX*	*AVE*				
15	5.91	35.47	19.53	0	28.08	6.46	66.92	2.55	1.67	34.50
20	6.18	33.14	17.06	0	17.91	5.48	67.91	1.77	1.37	22.78
25	1.39	14.55	5.64	0	9.33	2.86	49.23	1.65	1.12	31.90
30	0	7.72	2.13	0	5.37	1.28	40.07	1.88	1.35	27.95

表4　不同处理时间(p_{jv})分布的SA和HSA的性能比较表(n=20，m=10，φ=5)

p_{jv}	*SA_Gap*/(%)			*HSA_Gap*/(%)			*Decrease_AveGap*/(%)	*SA_CPU*/s	*HSA_CPU*/s	*Decrease_CPU*/(%)
	MIN	*MAX*	*AVE*	*MIN*	*MAX*	*AVE*				
DU[10，20]	0.93	43.24	9.24	0	33.78	5.12	44.65	0.09	0.05	46.51
DU[10，30]	1.96	33.33	9.33	0	40.20	6.81	27.09	0.08	0.05	42.12
DU[10，40]	3.32	105.69	25.76	0	100.81	16.65	35.37	0.06	0.06	10.08
DU[10，50]	1.79	44.44	19.10	0	44.44	16.20	15.22	0.10	0.08	25.31

(下转62页)