基于时间轴的软件多项目任务调度遗传算法

肖 菁 吴 洲
(华南师范大学计算机学院 广州 510631)

摘 要 合理的调度可以在很大程度上提高人力资源在软件项目开发中的利用率。在研究了现有的任务调度算法的基础上,考虑了软件开发任务的可拆分特性,结合员工的技能水平与项目经验对任务调度的影响,定义了基于时间轴的多项目并行调度模型。该模型将任务按时间单元进行拆分,并且提出员工学习模式,用以动态改变员工的技能水平和项目经验,从而提高员工分配的灵活性,使得满足约束下的成本开销最小化,其中包含员工薪水和超时任务的开销。模型分解后,利用遗传算法求解,由于模型中约束条件众多,因此遗传算法中增加了各种启发式来提高算法性能。通过仿真实例验证了模型和算法的有效性。

关键词 软件管理,多项目软件管理,调度算法,遗传算法
中图法分类号 TP39 **文献标识码** A

Software Multi-project Scheduling Genetic Algorithms Based on a Time-line Model

XIAO Jing WU Zhou
(School of Computer Science, South China Normal University, Guangzhou 510631, China)

Abstract Reasonable scheduling can greatly improve the utilization of the human resources in the process of software project development. Based on the research of current task scheduling algorithms, taking the separable of software development tasks, employee's skills and project experience into consideration, the multi-project concurrent scheduling model based on time-line which splits the tasks by time unit was defined to minimize the cost that includes the employees' salaries and the penalties of overtime. At the same time, in order to improve the flexibility of assignment of employees, the model also proposes that the employees' skills and experience can be improved by training and working on some tasks. Since the model contains many conditions, genetic algorithm with some heuristics is used to implement the model. The effectiveness of this model and algorithm is verified by the simulation results.

Keywords Software management, Multi-project software management, Scheduling algorithm, Genetic algorithm

1 引言

随着现代企业管理制度改革的不断深入,企业运营、开发、设计工作逐渐以项目的形式组织和开展。而在项目管理领域中,项目资源调度的优化成为当前的重要研究前沿,求解这类问题有传统的启发式算法,例如:Min-Min[1,2]算法、Max-Min[1,2]算法、模拟退火算法等,而 Housesh[3]率先将遗传算法引入任务调度问题研究,为遗传算法解决任务调度问题提供了基本框架。当前的研究模型主要以资源受限的多项目调度问题为主,S. Kumanan 和 G. Jegan Jose[4]提出了资源受限的多项目多模式调度模型,即各种项目的各个活动存在不同的作业模式,并且在资源配置时用启发式规则进行优化,结合遗传算法来达到最小化项目工期的目标。与之类似,Po-Han Chend[5]提出了遗传算法与模拟退火结合的混合算法来求解多资源受限的多项目调度问题。在该模型中,Po-Han Chend 用基于 NTF(next time frame)的资源分配模式以及基于任务优先级的资源分配策略求解项目完成工期的最小值;Tyson[6]则着重研究了优先级启发式在多项目任务调度中发挥的作用。软件开发项目的任务调度问题作为调度问题的一类,与一般的项目调度问题所不同的是:人力资源是项目中最主要的资源;任务可以由单人执行,也可以由多人执行;任务可以拆分成若干次执行,中途暂停对任务影响很小。针对软件项目开发的特点,Antoniol 等人[7]提出用遗传算法进行软件项目的计划并考虑了时间和成本因素;Luo[8]在单项目的任务调度模型中将任务进行分解,运用遗传算法求解项目的最小工期,并且在变异时加入混沌系统的漂移特性以改善种群的多样性,使得算法性能得到改善,但在该模型中只是单项目,应用规模较小,也没有考虑各人的各项技能水平的高低对调度结果的影响;而 Deng 和 Guo[9]在多项目的人力资源的分配模型中,考虑了每个人具备不同的技能,且技能水平有高低,用病毒感染遗传算法(即在初始化种群后用病毒染色体感染所有染色体,使其适应度得到整体性提高)求解多项目的最小工期,并且使得完成任务的质量尽可能高(即分派给该任务的员工所应具备的能力系数较高)。同样地,Fu 和 Zhou[10]提

到稿日期:2012-02-18 返修日期:2012-05-25 本文受高等学校博士学科专项科研基金(20090171120003)资助。
肖 菁(1975-),女,博士,副教授,主要研究方向为计算智能、运筹学和数据挖掘等,E-mail:xiaojing@scnu.edu.cn;吴 洲(1989-),男,硕士生,主要研究方向为进化算法及其应用。

出与 Deng 类似的模型，但其侧重目标在求解完成项目所需要支付的总成本，且运用免疫遗传算法和列生成相结合的方法求解问题的最优解。这些模型都假设员工的各种技能是固定的，而现实中员工的技能会随工作时间变化。Wu 和 Sun[11] 考虑到人员的学习能力，即从事同一工作的时间越长，平均工作效率越高，使用了遗传算法来解决单项目的调度问题。

以上的各种模型以及当前研究的其他类似模型都不够完善，都只是偏重于对某一影响因素的研究，而 Carl[12] 对他早期的模型[13]进行了扩充和完善，提出了一个较为完善且贴近现实的单项目调度模型，其主要特点如下：

(1)利用软件开发任务的可拆分性，在时间上将任务进行拆分；

(2)考虑了员工技能水平高低与项目经验对任务调度的影响；

(3)考虑了员工具有学习技能，员工的技能和项目经验会随时间动态变化。

该模型只可用于求解单项目调度问题，求解问题的规模比较小；另外，在现实的软件开发过程中，一个员工可能不止同时参加一个项目，一个软件开发公司也往往会在一个时间段同时开展几个不同的项目。

为此，文本在 Carl 的模型上进行扩展，将该模型扩展为解决多项目并行调度问题的模型，并且考虑了各个项目之间存在的优先级别对调度过程的影响，最后用仿真实验证明了模型和算法的有效性。

2 问题表示

在传统的任务调度模型中，任务是不可分离的，即从任务的开始到结束，整个持续时间段内给任务分配的员工是固定不变的。本问题模型中，假设任务是可分离的，即在时间上将每个任务的持续时间分成若干时间单元，在每个时间单元对任务分配员工，选取时间单元参数 ϕ，确定每个时间单元的具体长度。这里具体指将 1 个月分割成若干个时间单元，如以 1 天为 1 个时间单元，则 $\phi=30$；若以 1 周为 1 个时间单元，则 $\phi=4$；假设完成所有项目的时间上限为 T_{limit}。

2.1 项目模型

基于时间轴多项目的任务调度问题可以用图 1 描述。该问题由一个项目集合 A 构成，项目集 $A=\{P_1,P_2,\cdots,P_i,\cdots,P_N\}$ 由 N 个项目组成，每个项目包括若干个任务，即 $P_i=\{t_{i0},t_{i1},\cdots,t_{iT_i},t_{iM}\}$，$T_i$ 表示项目 i 的任务总数，t_{i0} 和 t_{iM} 是虚拟任务，代表项目 P_i 的开始任务和结束任务。$Effort_{i,j}$ 代表项目 i 中任务 j 所需的工作量，是一个百分比，$0<Effort_{i,j}\leqslant 100$。集合 $SK_{i,j}$ 代表完成项目 i 中任务 j 所需要具备的技能，$SK_{i,j}\in\{s_1,s_2,\cdots,s_m\}$，$m$ 代表完成所有工程所需的全部技能种类总数。$F_{i,j}$，$S_{i,j}$ 分别代表项目 i 中任务 j 的完工时间和开始时间。$SD_{i,j}$ 和 $HD_{i,j}$ 分别代表要求其完成的软最迟时间和硬最迟时间，$SD_{i,j}$ 和 $HD_{i,j}$ 都为 0 代表该任务无软硬时间限制，否则若任务完工时间超过 $SD_{i,j}$ 但不超过 $HD_{i,j}$，则需要计算超时开销。$Pa_{i,j}$ 用于计算超时开销，如果出现有任务超过 $HD_{i,j}$，则该解非法。用 $Pe_{i,j}$ 表示完成项目 i 中任务 j 所需要的超时开销。$MH_{i,j}$ 代表能同时分配给项目 i 中任务 j 的员工总数上限。集合 $Pro_{i,j}$ 代表需要在项目 i 中任务 j 开始前完成的任务集合，即其前驱任务集。

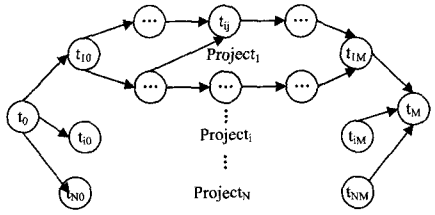

图 1 多项目调度示意图

本模型是多项目调度，各项目之间存在优先权的问题，用 Pri_j 表示项目 j 的优先权等级，该值与项目需要的总工作量相关。同一个项目内的任务之间的优先权用有向无环图表示，结点代表任务，边代表任务的优先顺序。

2.2 员工模型

表 1 符号意义

符号	表示意义
T_{limit}	完成所有项目的时间单元上限(共 N 个项目)
$Effort_{i,j}$	项目 i 中任务 j 所需的工作量
$SK_{i,j}$	完成项目 i 中任务 j 所需要具备的技能集合
m	完成所有工程所需的全部技能种类总数
$F_{i,j}$	项目 i 中任务 j 的完工时间
$S_{i,j}$	项目 i 中任务 j 的开始时间
$SD_{i,j}$	项目 i 中任务 j 完成的软最迟时间
$HD_{i,j}$	项目 i 中任务 j 完成的硬最迟时间
$Pe_{i,j}$	完成项目 i 中任务 j 所需要的超时开销
$Pa_{i,j}$	用于计算超时开销
$MH_{i,j}$	能同时分配给项目 i 中任务 j 的员工总数上限
$Pro_{i,j}$	需要在项目 i 中任务 j 开始前完成的任务集合，即其前驱任务集
T_i	项目 i 的任务总数
Pri_j	项目 j 的优先权等级
E_N	员工总数
Man_e	员工类型标识
$ES_{t,e}$	t 时刻员工 e 所拥有的技能集
u_e	员工 e 的学习率
$Train_{(e,t,s)}$	在时间单元 t 结束后对员工 e 技能 s 的总培训量
$LS_{t,e,s}$	员工 e 在时间单元 t 结束后技能 s 的熟悉度
$LS_{0,e,s}$	员工的 e 的技能 s 的初始熟悉度，(上限为 MS)
$EE_{t,e}$	时间单元 t 结束后员工 e 的项目经验值
$EE_{0,e}$	员工 e 的项目初始经验值(上限为 ME)
MW_e	员工 e 在每个时间单元内的总工作量(包括培训)的上限
$w_{t,e,i,j}$	在时间单元 t 时员工 e 对项目 i 中的任务 j 所分配的工作量
$H_{t,e,i,j}$	标识在时间单元 t 员工 e 对项目 i 中的任务 j 是否分配了工作
$Train_{t,e,s}$	t 时间单元员工 e 对技能 s 进行培训量
$Se_{,100}$	员工 e 每个月的基本工资
$Se_{,o}$	用于员工 e 加班后的补贴计算
$C_{t,e}$	员工 e 在时间单元 t 的薪酬
$ET_{t,st}$	员工 e 允许参加项目的最早时间
$ET_{t,en}$	员工 e 允许参加项目的最迟时间

本模型中考虑了不同员工对同一项技能的熟悉程度是不同的，技能的熟悉度也考虑在任务调度的影响因素之内。假设员工总数为 E_N，员工分两类，一类作为管理人员，不对其进行技能的培训；另一类则是普通员工，可对其进行各项技能的培训。其用 Man_e 标识，$Man_e=1$ 代表员工 e 为管理人员，否则为 0。在传统的模型中，往往都假设员工的技是从所有项目开始到结束都保持固定不变。在本模型中，员工的各项技能都是动态变化的，集合 $ES_{t,e}$ 代表在 t 时刻员工 e 所拥有的技能，$ES_{t,e}\in\{s_1,s_2,\cdots,s_m\}$，员工可以通过培训提升自己的技能，技能的变化与培训量和员工的学习率相关。用 u_e 代表员工 e 的学习率；用 $Train(e,t,s)$ 定义在时间单元 t 结束后对员工 e 技能 s 的总培训量；用 $LS_{t,e,s}$ 代表员工 e 在时间单元 t

结束后技能 s 的熟悉度，$LS_{0,e,s}$ 则代表员工 e 的技能 s 的初始熟悉度，所有技能熟悉度上限为 MS。员工对项目的经验值也作为考虑任务分配的影响因子之一，因此用 $EE_{0,e}$ 代表员工 e 的项目初始经验值，$EE_{t,e}$ 表示时间单元 t 结束后员工 e 的项目经验值，上限为 ME。每个员工在每个时间单元内的总工作量(加上培训)都有上限，用 MW_e 表示员工 e 的上限，也是一个百分比，在每个时间单元内计算员工的薪水要考虑不加班和加班两种不同的情况。用 $S_{e,100}$ 表示员工 e 每个月的基本工资，用 $S_{e,o}$ 表示员工 e 加班后的补贴计算，用 $w_{t,e,i,j}$ 表示在时间单元 t 员工 e 对项目 i 中的任务 j 所分配的工作量，也同样是一个百分比，用 $H_{t,e,i,j}$ 作为标志，如果 $w_{t,e,i,j}=0$ 则 $H_{t,e,i,j}=0$，否则 $H_{t,e,i,j}=1$。用 $Train_{t,e,s}$ 表示在 t 时间单元员工 e 对技能 s 所进行的培训量。用 $C_{t,e}$ 表示员工 e 在时间单元 t 的薪酬。每个员工参加任务分配有具体的时间段的限制，用 $ET_{t,s}$ 和 $ET_{t,en}$ 分别代表员工 e 允许参加项目的最早时间和最迟时间。为方便理解以下公式的定义，表 1 列出上述用于定义公式的符号意义。

2.3 目标函数

$$\min\left(\sum_{t=1}^{T_{limit}}\sum_{e=1}^{E_N}c_{t,e}+\sum_{i=1}^{N}\sum_{j=1}^{T_i}Pe_{i,j}\right) \tag{1}$$

式(1)意味着求使得总开销最小的可行任务分配。总开销包括从开始时间单元到结束的时间单元需要支付的员工薪水和各个项目中完工时间超过软最迟时间的任务所需要的超时开销。

2.4 约束

$$F_{i,j}\leqslant HD_{i,j},i=1,2,\cdots,N,j=1,2,\cdots,T_i,HD_{i,j}\leqslant T_{limit} \tag{2}$$

$$S_{i,j}>F_{i,k},i=1,2,\cdots,N,j=1,2,\cdots,T_i,\forall k\in Pro_{i,j} \tag{3}$$

$$\left(\sum_{i=1}^{N}\sum_{j=1}^{T_i}w_{t,e,i,j}+\sum_{s=1}^{m}Train_{t,e,s}\right)\leqslant MW_e$$
$$t=1,2,\cdots,T_{limit},e=1,2,\cdots,E_N \tag{4}$$

$$\sum_{e=1}^{E_N}H_{t,e,i,j}\leqslant MH_{i,j},t=1,2,\cdots,T_{limit},i=1,2,\cdots,N,j=1,2,\cdots,T_i \tag{5}$$

$$ET_{t,s}\leqslant\left\lceil\frac{t}{\phi}\right\rceil\leqslant ET_{t,en},t=1,2,\cdots,T_{limit},e=1,2,\cdots,E_N \tag{6}$$

$$H_{t,e,i,j}=\{0\mid ES_{t,e}\notin SK_{i,j},ES_{t,e}\notin\Phi\},t=1,2,\cdots,T_{limit},$$
$$i=1,2,\cdots,N,j=1,2,\cdots,T_i,e=1,2,\cdots,E_N \tag{7}$$

$$LS_{t,e,s}=\min\{LS_{0,e,s}+u_e\times Train(e,t,s),MS\},$$
$$Train(e,t,s)=\frac{1}{\varphi}\times\sum_{i=0}^{t}Train_{i,e,s},t=1,2,\cdots,T_{limit}$$
$$e=1,2,\cdots,E_N,s=1,2,\cdots,m \tag{8}$$

$$EE_{t,e}=\min\left\{\left(EE_{t-1,e}+\frac{u_e}{\phi}\times\sum_{i=1}^{N}\sum_{j=1}^{T_i}w_{t,e,i,j}\right),ME\right\},t=1,2,$$
$$\cdots,T_{limit},e=1,2,\cdots,E_N \tag{9}$$

式(2)表示所有项目的每个任务必须在硬最迟时间内完成；式(3)表示所有项目的每个任务必须在其前驱任务完成后才能开始，即必须满足任务之间的优先顺序；式(4)要求每个员工在任意时间单元内所分配的工作量(包括培训)不能超过其工作上限；式(5)要求同时对一个任务分配的员工数不能超过其最大值；式(6)规定了各个员工进入和退出项目的时间；式(7)要求给各个任务分配的员工需要满足该任务的各项技能要求；式(8)给出了员工技能熟悉度的变化算法，也规定了在任意时刻员工的技能熟悉度都不能超过上限；式(9)给出了员工项目经验值的变化算法，同时规定了最大值。

3 算法求解

本文采用遗传算法对问题模型进行求解，求解过程建立在 C++遗传算法库(GAlib)[14]上，染色体编码采用了 GAlib 中的 3D array，一维代表时间，一维代表员工，另一维代表某一项目中的某一个任务(同时包括技能培训)，杂交和变异都采用 GAlib 提供的基于 3D array 的接口函数。表 2 代表在 t 时间单元时员工对任务的工作量分配情况，求解结果就是得出从开始时间单元到结束时间单元中每个时间单元的人员分配情况。

表 2 t 时间单元时人员分配示例

	e_1	…	e_i	…	e_{E_N}
$pro_1,Task_1$			…		
…					
$pro_i,Task_j$			$w_{t,e,i,j}$		
…					
$pro_N,Task_{T_N}$			…		
$skill_1$					
…					
$skill_s$			$Train_{t,e,s}$		
…			…		
$skill_m$					

3.1 启发式

鉴于遗传算法的初始种群都是随机产生的，为使产生的解能满足问题模型中的各种约束条件，需要加入一些启发式对种群的染色体进行优化，这也能加快求解的速度，这些启发式在整个工程的每个时间单元都必须应用。

(1)撤销已经完成的任务：如果某些任务在某一时间单元的上个时间单元已经完成，则将这个时间单元内分配给这些任务的所有员工撤销，即不再给这些任务分配员工。

(2)撤销不合时的任务：如果某些任务在某一时间单元尚不能开始，即其前驱任务尚未全部完成，则将这个时间单元内分配给这些任务的所有员工撤销。

(3)剔除不合适的员工：如果某些员工没有达到某些任务要求的全部技能，则将分配给这些任务的不满足技能要求的员工撤销。

(4)剔除不符合工作时间的员工：对于在下一时间单元不在其可工作时间内的员工，取消对这些员工分配的所有任务。

(5)剔除对管理人员的技能培训。

(6)限制员工的训练：如果员工的某项技能的熟悉度已经达到一个规定值，如 E_{xess}，则撤销该员工对此项技能的培训。

(7)调整员工的工作量：在多项目的任务调度中，任务数比较多，同一个时间单元对员工分配的工作量可能会超过员工的工作量上限，为每个员工创建一个链表，用于记录该员工为哪些任务分配了任务。链表中元素按项目的优先级别排序，优先级别高的排在链首，同一项目之间的任务按员工为该任务分配的工作量排序，工作量多的排在前面。整个链表的所有任务工作量的总和等于员工在这个时间单元的工作量，当超过该员工允许的最大工作量时，剔除在链表末尾的元素，即先消除项目优先级最低，且为该任务分配工作量最少的任务，重复这个操作，直到员工在这个时间单元的工作量达到上限。

(8)调整员工分配：当为某一任务分配的员工超过该任务允许的人数上限时，计算各个员工对该任务的适合度(具体计算过程在下文式(15)给出)，将符合度最低的员工从该分配中删除，重复直到达到任务允许分配的人数上限。

3.2 fitness 函数

个体的 fitness 函数由下式确定：

$$Fit(x)=ConstMax-(\sum_{t=1}^{T_{limit}}\sum_{e=1}^{E_N}c_{t,e}+\sum_{i=1}^{N}\sum_{j=1}^{T_i}Pe_{i,j}) \tag{10}$$

这里设 $ConstMax$ 为一足够大的正数，因此优化目标即为使个体的 fitness 值最高。

计算 $C_{t,e}$，设

$$Q=\sum_{i=1}^{N}\sum_{j=1}^{T_i}w_{t,e,i,j}+\sum_{s=1}^{m}Train_{t,e,s}$$

则

$$C_{t,e}=\frac{1}{\phi}\times\begin{cases}S_{e,100}\times Q\%, & 0\leqslant Q\leqslant 100\\ S_{e,100}\times 100\%+S_{e,o}\times(Q-100)\%, & 100\leqslant Q\leqslant MW_e\\ \infty, & Q>MW_e\end{cases} \tag{11}$$

计算 $Pe_{i,j}$，即

$$Pe_{i,j}=\begin{cases}0, & F_{i,j}\leqslant SD_{i,j}\\ Pa_{i,j}\times(F_{i,j}-SD_{i,j}), & SD_{i,j}<F_{i,j}\leqslant HD_{i,j}\\ \infty, & F_{i,j}>HD_{i,j}\end{cases} \tag{12}$$

另外，模型考虑到项目之间的优先级别问题。本文简单假设项目之间的优先等级通过各项目的所有任务需要的总工作量的大小来确定，工作量大的优先级别高，员工培训默认为优先级别最低(在实际情况中往往还要考虑项目之间的缓急程度等其他因素，这里不做详细讨论)，因此定义

$$Pri_i=\sum_{j=1}^{T_i}Effort_{i,j},i=1,2,\cdots,N \tag{13}$$

按式(13)对所有项目的优先权进行计算，然后对其排序，得到项目之间的优先等级。

根据任务的剩余工作量来判断一个任务是否完成。项目 i 中任务 j 的工作量的初始化值在前文已经定义为 $Effort_{i,j}$，完成该任务需要具备的技能为集合 $Sk_{i,j}$，用链表 LE 存储分配到该任务的员工。定义 $ReEff_{t,i,j}$ 为在 t 时间单元结束后项目 i 中任务 j 的剩余工作量，$ReEff_{t,i,j}\leqslant 0$ 标志该任务已经完成。通过下列式(14)－式(18)求解出 $ReEff_{t,i,j}$：

$$TaskProf_{e,i,j}=\prod_{s\in S_{i,j}}\frac{LS_{t-1,e,s}}{m} \tag{14}$$

有了员工对任务的熟悉度 $TaskProf_{e,i,j}$ 和经验值，就可以定义员工对任务的适合度。对任意员工 $e\in LE$，用 $TaskFit_{e,i,j}$ 定义员工 e(假设在 t 时间单元)对项目 i 中任务 j 的适合度：

$$TaskFit_{e,i,j}=\frac{TaskProf_{e,i,j}+EE_{t-1,e}/m}{2} \tag{15}$$

定义

$$F=\frac{\sum_{e\in E}(TaskFit_{e,i,j}\times w_{t,e,i,j})}{\sum_{e\in E}w_{t,e,i,j}} \tag{16}$$

将 F 转换为代价值 $V=8-\mathrm{round}(F*7+0.5)$ (17)

$$ReEff_{t,i,j}=ReEff_{t-1,i,j}-\frac{\sum_{e\in E}w_{t,e,i,j}/100}{V\times\phi} \tag{18}$$

个体适度值求解步骤如下：

步骤 1　初始化员工的各项技能初始值和项目经验初始值；

步骤 2　初始化时间单元 $t=1$；

步骤 3　如果所有项目都完工，则转步骤 13，否则转步骤 4；

步骤 4　如果 $t>T_{limit}$，则该个体为非法解，结束，否则转步骤 5；

步骤 5　应用上述的所有启发式；

步骤 6　计算薪酬 $\sum_{e=0}^{E_N}C_{t,e}$(式(11))；

步骤 7　调整所有项目各个任务的剩余工作量(式(18))；

步骤 8　如果有任务已经完成，转步骤 9，否则转步骤 11；

步骤 9　计算已完成的任务的所需的超时开销 $Pe_{i,j}$(式(12))；

步骤 10　如果存在 $F_{i,j}>HD_{i,j}$，则个体为非法解，结束，否则转步骤 11；

步骤 11　调整员工各项技能熟悉度(式(8))和项目经验值(式(9))；

步骤 12　$t=t+1$，转步骤 3；

步骤 13　计算 fitness 值(式(10))，结束。

4　仿真实验与结果分析

4.1　仿真实验

为了验证模型和算法的有效性，本文使用实验数据对软件项目开发多项目的任务调度问题进行仿真计算。本次实验选取 5 个项目，表 3 至表 7 给出各个项目的每个任务的具体信息，$Effort$ 列代表每个任务需要的工作量(百分比)，SD 和 HD 分别代表任务的软最迟时间和硬最迟时间，Pa 用于计算任务的延时开销，MH 为可同时分配的员工总数上限，S 为完成该任务需要的各项技能编号，设定总共有 5 项目技能。根据实验数据，项目 1 所需的总 $Effort$ 为 7.4，项目 2 的总 $Effort$ 为 3.25，项目 3 的总 $Effort$ 为 6.05，项目 4 的总 $Effort$ 为 3.8，项目 5 的总 $Effort$ 为 4.95，按各项目的总 $Effort$ 排列，其优先级为 $Project_1>Project_3>Project_5>Project_4>Project_2$，图 2 至图 6 给出了每个项目内部任务之间的优先级别。选取时间单元为一周，即令 $\phi=4$，要求所有项目在 60 个时间单元内完成，即 $T_{limit}=60$。令 $m=5$，即完成所有任务要求的技能总数为 5，员工所能拥有的各项技能的熟悉度上限 $MS=5$，员工的项目经验值上限 $ME=5$。选派 10 个员工完成项目，表 8 给出员工的各项具体信息，表中 Man 列标识该员工是否为管理人员，为 1 则表示是管理人员，否则为 0，S_{100} 代表员工每个月的基本工资，S_o 用于计算员工加班的工资，MW 表示员工每个时间单元能分配的最大工作量，EE_o 为员工的初始项目经验值，u 为学习率，ET_s 和 ET_e 分别为员工允许参与项目的开始时间和结束时间。表 9 给出员工各项技能的初始熟悉度。为方便计算，令 $w_{t,e,i,j}$ 和 $Train_{t,e,s}$ 的取值在 $\{0,25,50,75,100\}$ 之间。

表 3　项目 1 详细信息

TaskID	Effort	SD	HD	Pa	MH	S
1	25	2	4	1000	2	1 3
2	50	8	12	1000	2	2 3 5
3	80	12	15	2000	3	1 2 4 5
4	25	0	0	0	3	1 2
5	60	0	0	0	3	1 3
6	50	0	0	0	3	3 4
7	30	0	0	0	2	2 4 5
8	40	24	32	2000	2	2 3
9	25	0	0	0	2	2 4
10	50	0	0	0	2	4 5
11	50	0	0	0	2	1 3 5
12	25	0	0	0	2	2 4
13	80	0	0	0	3	2 3 4
14	50	40	52	3000	2	1 2 3 5
15	100	52	60	5000	3	2 3

表 4　项目 2 详细信息

TaskID	Effort	SD	HD	Pa	MH	S
1	25	10	20	1000	2	1
2	60	0	0	0	3	2 3
3	25	20	30	1000	2	2 4
4	80	0	0	0	3	1 2 5
5	60	50	54	2000	2	5
6	50	0	0	0	3	2 4
7	25	55	60	3000	2	2

表 5　项目 3 详细信息

TaskID	Effort	SD	HD	Pa	MH	S
1	100	8	15	1000	2	1 3
2	80	14	26	1500	2	1 2
3	70	0	0	0	3	3 5
4	40	0	0	0	3	3 4
5	60	0	0	0	2	2 3
6	75	30	35	1000	2	2 3 4
7	50	30	35	2000	2	2 5
8	50	0	0	0	3	1 4
9	80	0	0	0	3	2 4

表 6　项目 4 详细信息

TaskID	Effort	SD	HD	Pa	MH	S
1	80	10	20	1000	2	1 2
2	75	30	45	2000	3	1 3
3	75	0	0	0	2	2 4
4	50	0	0	0	3	1 3 5
5	60	45	55	2000	2	2 3 5
6	40	0	0	0	2	1 2 3

表 7　项目 5 详细信息

TaskID	Effort	SD	HD	Pa	MH	S
1	80	0	0	0	2	2 4
2	100	20	30	2000	3	3 5
3	60	0	0	0	2	1 2 4
4	40	40	44	1500	2	2 5
5	75	0	0	0	3	1 4
6	50	0	0	0	2	2 3 5
7	60	45	55	1000	2	2 3
8	30	0	0	0	2	1 2 5

表 8　员工详细信息

EID	Man	S_{100}	S_0	MW	EE_0	u	ET_s	ET_e
1	0	6000	100	150	4.8	1.5	1	30
2	0	5300	0	100	4.7	1.3	1	30
3	1	4800	0	100	4.6	1.3	1	30
4	0	5000	0	75	4.7	0.9	1	30
5	0	5000	0	50	4.6	1.1	1	30
6	0	5800	100	125	4.8	1.4	1	30
7	0	5800	0	100	4.8	1.5	1	30
8	0	5000	0	100	4.5	1.0	1	30
9	1	4600	0	75	4.3	1.2	1	30
10	0	5300	0	50	4.6	1.3	1	30

表 9　员工技能初始值

EID	LS_1	LS_2	LS_3	LS_4	LS_5
1	4.5	5	0	5	4.8
2	0	0	3.5	0	4.8
3	4.3	4	3.5	0	4.8
4	0	4.7	0	0	0
5	4.5	4	3.8	0	5
6	0	4.5	4.3	4	0
7	4.5	4.8	5	0	0
8	0	0	0	4.5	4.6
9	0	0	3.9	4.8	0
10	4.7	4	0	3	4

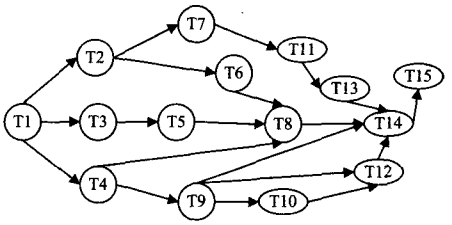

图 2　项目 1TPG

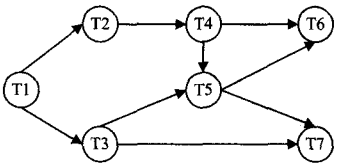

图 3　项目 2TPG

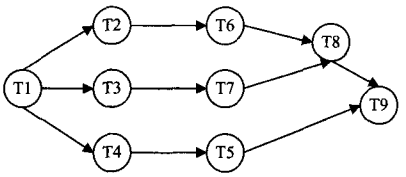

图 4　项目 3TPG

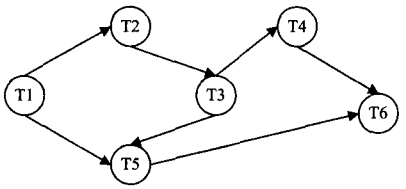

图 5　项目 4TPG

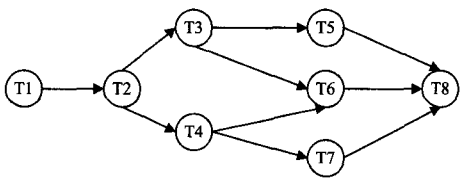

图 6　项目 5TPG

根据以上的数据，选取种群规模为 1000，遗传代数为 1000，变异率为 0.001，应用本文的模型和算法得到最优个体的总开销为 57254900，对应的调度甘特图如图 7 所示，图中横坐标代表时间，单位为周，纵坐标表明具体的项目号和任务号。从甘特图中可以看出，项目 1 的最后一个任务完工时间为第 33 时间单元，项目 2 的最后一个任务的完工时间为第 49 时间单元，项目 3 最后一个任务的完工时间为第 34 时间单元，项目 4 最后一个任务的完工时间为第 43 时间单元，项目 5 最后一个任务的完工时间为第 41 时间单元，各个项目的完工时间按从早到晚排序为 $Project_1 < Project_3 < Project_5 <$

$Project_4 < Project_2$。根据仿真实验的数据和计算公式可以得出，5个项目的优先级排序也如完工时间的排序，即项目1的工作量最多，优先级别最高，最早完工；同时，从甘特图中也可以看出各个项目内部的任务调度也满足任务的优先权约束。遗传算法的进化过程如图8所示，图中横坐标代表遗传代数，纵坐标为目标函数值，目标函数值随遗传代数增加逐渐优化。

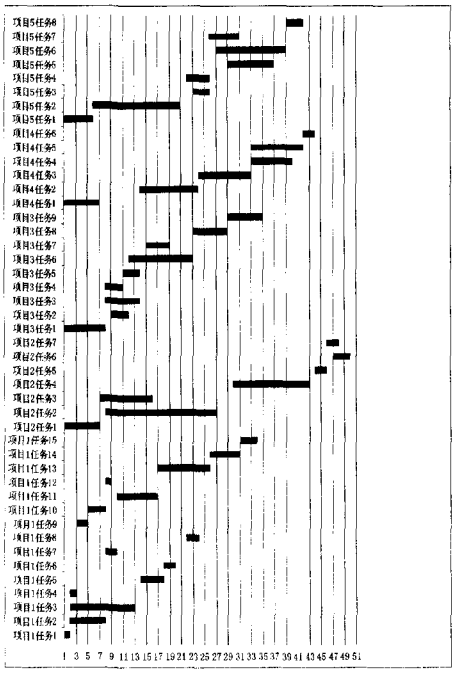

图7　最优解对应甘特图

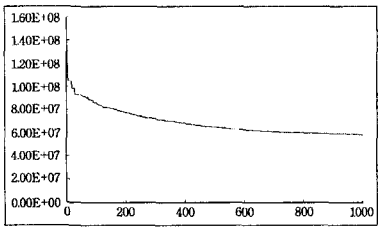

图8　进化过程

4.2　算法比较

为了验证遗传算法的性能，本文对上述的模型采用爬山算法和模拟退火算法进行求解，对3种算法分别运行了10次，取得每次实验结果，即优化的目标函数值(完成任务的总开销)。比较结果如表10所列，*Best* 值表示实验过程中能达到的最优值(最低开销)，*Worst* 为最坏情况下的目标函数值(最高开销)，*Average* 为10次实验结果的目标函数平均值(平均开销)。实验结果表明，GA在最优情况、最坏情况以及平均情况下得到的目标函数值都比爬山算法和模拟退火更优。

表10　算法比较结果

Algorithm	Best	Average	Worst
GA	57254900	62100590	65815600
Hill-climbing	74417000	81080890	87196200
SA	63772600	68882500	74488400

结束语　本文针对软件开发并行多项目调度问题的特点，考虑了多项目并行处理的优先级、项目任务对员工技能、水平的不同要求，并且从时间上对任务进行拆分细化，从而使员工的分配更加灵活。与此同时，赋予员工学习的能力，使员工的技能和经验水平动态变化，更加贴近现实；建立了相应的模型，目标为总开销最小化，运用遗传算法进行对其求解，另外在求解过程加入相应的启发式，从而使解能满足模型的约束条件并且加速了问题的求解。仿真实验的结果和算法比较验证了模型的有效性和算法的性能。

本文的模型仍有待进一步完善。该模型中项目的工期是作为约束存在的，并没有把工期考虑到目标函数中，目标函数只是单一考虑成本开销的问题，只是做到多项目单目标优化，而如何把工期和成本开销同时作为求解目标，在工期和成本之间加以共同优化，以实现二者的平衡，达到多目标优化，在实际中也是非常有意义的。

参 考 文 献

[1] Braun T, Siegel H, Netal B. A comparison study of static mapping heuristics for a class of meta-tasks on heterogeneous computing systems [C] // 8th IEEE Heterogeneous Computing Workshop. 1999:15-29

[2] Moreno R. Job Scheduling and resource management techniques in dynamic grid environment [C] // 1st European Across Grids Conference. 2003

[3] Housesh, Ansarin, Renh. A genetic algorithm for multi-processor scheduling[J]. IEEE Transactionon Parallel and Distributed Systems, 1994, 5(2):113-120

[4] Kumanan S, Jose G J, Raja K. Multi-project scheduling using an heuristic and a genetic algorithm [J]. Int J Adv Manuf Technol, 2006, 31:360-366

[5] Chen Po-han, Shahandashti S M. Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints [J]. Automation in Construction, 2009, 18:434-443

[6] Browning T R, Yassine A A. Resource-constrained multi-project scheduling: Priority rule performance revisited [J]. Production Economics, 2010, 126:212-228

[7] Antoniol G, Di P M, Hanman M. Search based techniques for optimizing software project resource allocation [C] // Kalyanmoy Deb, ed. Proc of the Genetic and Evolutionary Computation Conf. Seattle: Springer-Verlag, 2004:1425-1426

[8] 雒兴刚，汪定伟，唐加福. 软件开发项目中任务调度的混沌遗传算法[J]. 小型微型计算机系统，2006，27(10)：1923-1926

[9] 邓晓刚，郭钢. 在多项目管理人力资源配置中的病毒遗传算法[J]. 计算机工程与应用，2009，45(33)：19-21

[10] 付芳，周泓. 基于免疫遗传算法和列生成的多项目人力资源调度研究[J]. 中国管理科学，2010，18(2)：120-126

[11] Wu W, Sun S. A project scheduling and staff assignment model considering learning effect [J]. International Journal Advanced Manufacture Technology, 2006, 28:1190-1195

[12] Chang C K, Jiang H-Y, Di Yu. Time-line based model for software project scheduling with genetic algorithms [J]. Information and Software Technology, 2008, 50:1142-1154

[13] Chang C K, Christensen M J, Zhang Tao. Genetic Algorithms for Project Management [J]. Annals of Software Engineering, 2001, 11:107-139

[14] Wall M. Lib: A C++ Library of Algorithm Components[OL]. http://lancet.mit.edu/ga