

文章编号:1006-2467(2020)06-0624-12

DOI: 10.16183/j.cnki.jsjtu.2018.134

考虑人力资源排班的资源受限项目 调度问题建模与优化

朱宏伟, 陆志强

(同济大学 机械与能源工程学院, 上海 201804)

摘要: 针对实际生产系统中人力资源以排班的形式进行生产活动的情况,提出考虑人力资源排班的资源受限项目调度问题,以最小化项目工期为目标建立了问题的数学模型.由于串行调度在传统任务列表编码对应的解空间下难以获得较优解,本文借鉴车间调度中析取弧的概念,提出了一种改进任务列表编码方式,通过在任务之间添加析取弧的方式扩大算法的搜索范围.此外,为提升遗传算法的局部搜索能力,在改进任务列表编码基础上设计分支定界搜索框架,对遗传算法得到的染色体进行分段深度搜索,并设计支配规则降低算法计算时间.结果表明:内嵌分支定界搜索框架的遗传算法能够提高求解质量,而设计的支配规则能有效降低算法的运算时间.

关键词: 资源受限项目调度; 人力资源排班; 改进编码方式; 分支定界搜索框架

中图分类号: TP 29

文献标志码: A

Modeling and Optimization of Resource Constrained Project Scheduling Problem Considering Employee-Timetabling

ZHU Hongwei, LU Zhiqiang

(School of Mechanical and Energy Engineering, Tongji University, Shanghai 201804, China)

Abstract: Aimed at the practical situation where human resources conduct production activities in the form of shifts in production systems, this paper addresses the resource constrained project scheduling problem considering employee-timetabling and establishes a mathematical model with the objective of minimizing project makespan. Since the serial schedule generation scheme has difficulty in generating a good solution under the solution space delivered by traditional activity list, an improved activity list coding method based on the concept of disjunctive arc in job shop scheduling problem is designed to expand the search extent. Moreover, to improve the local search capability of the genetic algorithm, a branch-and-bound-based search framework based on the improved activity list coding method is designed to sectionally and deeply search the chromosome obtained by the genetic algorithm, and dominant rules are designed to reduce the computational time. The results show that the genetic algorithm with the branch-and-bound-based search framework could improve the solution quality, and the dominant rules could reduce the computing time efficiently and effectively.

Key words: resource-constrained project scheduling; employee-timetabling; improved coding method; branch-and-bound-based search framework

收稿日期: 2018-05-02

基金项目: 国家自然科学基金资助项目(61473211, 71171130)

作者简介: 朱宏伟(1993-), 男, 浙江省温州市人, 博士生, 主要研究资源受限项目调度问题.

通信作者: 陆志强, 男, 教授, 博士生导师, 电话(Tel.): 021-69589485; E-mail: zhiqianglu@tongji.edu.cn.

资源受限项目调度(RCPSP)作为一类重要的调度模型,广泛存在于制造、建筑等领域,其最大的特点是考虑资源为可更新资源,即资源在任务结束后立即释放,同时项目进程中资源总量不发生改变.然而,在许多实际装配生产系统中,例如飞机移动装配中,人力资源以排班的形式进行装配生产活动,其可用量会在生产过程中会受到排班的影响,因此事先制定的调度计划与实际的生产进度存在很大差距.此时,调度的关键在于如何将人力资源进行合理的排班,以确定每个班次下的人力资源数量,并在排班决策下为任务安排合理的开始时间,以获得最小的装配时间.此问题称为考虑人力资源排班的资源受限项目调度问题(RCPSP-ET),其本质上是资源受限项目调度与人力资源排班(ETP)的整合问题.由于 RCPSP-ET 问题与传统 RCPSP 问题的决策范围不同,现有的模型与算法无法直接应用,所以对 RCPSP-ET 问题的建模以及算法研究具有重要的实际与理论意义.

RCPSP 作为车间调度问题的一般化,已经被证明是非确定性多项式困难问题^[1],RCPSP-ET 作为其扩展问题,因此也是非确定性多项式困难问题.针对 RCPSP,Brucker 等^[2-3]采用分支定界算法进行求解.虽然分支定界等精确算法能求得问题的精确解,但是其求解规模有限,因此许多文献关注于启发式和元启发式算法的研究,如遗传算法(GA)^[4],粒子群算法(PSO)^[5-6]等.除此之外,Wang 等^[7-8]在求解 RCPSP 及其扩展问题时采用混合算法,通过将多种算法进行混合,或在原有算法框架上设计改进机制,来提升算法性能.

人力资源排班是为人员安排合适的工作时间窗以满足组织内部和外部需求的过程. Bergh 等^[9]根据问题决策的不同,认为研究可以分为 5 类:在满足一定条件下决策人员所分配的任务^[10];确定人员的工作班次^[11];决策人员之间的成组^[12];时间约束下确定人员的工作时间^[13]以及其他一些决策.然而, Bergh 等指出,当前研究人力资源排班的文献缺少人力资源排班与其他调度问题(如机器调度、生产调度)的结合.在特定的调度背景下,需将人力资源排班与其他调度问题统筹考虑,才能真正地反映系统的调度需求.

RCPSP-ET 本质上是人力资源排班与 RCPSP 的整合问题,目前有极少文献对该问题进行研究. Drezet 等^[14]将软件开发过程抽象为资源受限项目调度问题,在给定人员排班的情况下考虑劳动法规等因素对任务安排的影响.由于 RCPSP 与生产调

度又有一定的相似性,所以人力资源排班与生产调度结合的整合问题对于 RCPSP-ET 的研究具有参考意义. Artigues 等^[15]考虑工人排班与车间调度的整合问题,设计基于整数规划和约束规划的混合分支定界算法进行求解. Guyon 等^[16]在探究求解工人排班与生产排班整合问题的精确算法时,为了减小决策变量搜索空间,运用割生成求解该整合问题,将 Benders 分解以及 CPLEX 对比,验证了算法的效率. Ahmadi-Javid 等^[17]进一步提升整合问题的维度,提出结合加工车间调度、运输工具调度以及人力资源排班的整合问题.

综上所述,目前极少学者考虑将人力资源排班与 RCPSP 结合的整合问题,然而 RCPSP 作为一类重要的生产过程抽象,研究 RCPSP-ET 问题具有重要意义.除此之外,目前解决这些问题的算法主要是基于问题模型研究而设计的精确算法,缺少对混合算法等高效算法的研究.因此,本文考虑人力资源排班中存在的约束,以最小化完工时间为目标,建立了 RCPSP-ET 的整数规划模型.同时,本文设计了一种改进任务列表编码方式,并借鉴混合算法的思路,提出了一种基于分支定界搜索框架的遗传算法(BBGA),并在此基础上提出支配规则,从而降低算法运算时间,提升算法的求解效率.

1 考虑人力资源排班的资源受限项目调度问题

1.1 问题描述

某一装配工位由 N 项任务组成,记任务集合为 $J = \{1, 2, \dots, N\}$, $j \in J$ 为任务编号;第 j 项任务的紧前任务集合为 P_j ; $i \in P_j$ 为第 j 项任务的第 i 项紧前任务;任务 j 的执行时间为 t_j ,任务一旦开始便无法中断.目标为最小化装配总工期 T .

任务的执行需要满足其人力资源种类和数量的需求.所有任务共享人力资源,记人力资源集合、人力资源种类集合分别为 $E = \{1, 2, \dots, B\}$, $K = \{1, 2, \dots, R\}$, $e \in E$ 为人力资源编号, $k \in K$ 为人力资源种类编号, E_k 表示第 k 类人力资源集合, r_{jk} 表示执行任务 j 所需第 k 类人力资源的数量.人力资源无法抢占,即任务一旦开始,其执行过程不能更换执行的人力资源.

装配过程人力资源以 n 班制进行装配生产活动(如两班制,三班制分别对应 $n = 2, n = 3$).记班次集合 $S = \{1, 2, \dots, H\}$, $s \in S$ 为班次编号;对时间进行离散化处理,记离散时间集合为 $D = \{1, 2, \dots, V\}$, $d \in D$ 为离散时间节点, D_s 表示第 s 个班次所包

含的离散时间集合. 人力资源受到劳动法规的约束, 在连续 n 个班次中最多能被安排 1 个班次. 由于不允许人力资源抢占, 而某一人力资源又无法安排相邻的班次, 所以任务的执行期不允许跨班次.

1.2 数学模型

决策变量:

$$x_{jd} = \begin{cases} 0, & \text{任务 } j \text{ 不在时刻 } d \text{ 执行} \\ 1, & \text{任务 } j \text{ 在时刻 } d \text{ 执行} \end{cases}$$
$$y_{es} = \begin{cases} 0, & \text{人力资源 } e \text{ 不在班次 } s \text{ 执行} \\ 1, & \text{人力资源 } e \text{ 在班次 } s \text{ 执行} \end{cases}$$
$$z_{js} = \begin{cases} 0, & \text{任务 } j \text{ 不在班次 } s \text{ 执行} \\ 1, & \text{任务 } j \text{ 在班次 } s \text{ 执行} \end{cases}$$

模型:

$$\min T$$

(1)

s. t.

$$\sum_{d \in D} x_{jd} = t_j, \quad \forall j \in J$$

(2)

$$t_i x_{jd} \leq \sum_{p=1}^{d-1} x_{ip}, \quad \forall i \in P_j, \forall j \in J, \forall d \in D$$

(3)

$$t_i x_{jd} - t_i x_{j(d+1)} + \sum_{p=d+2}^V x_{jp} \leq t_j, \quad \forall i \in P_j, \forall j \in J, \forall d \in D$$

(4)

$$dx_{jd} \leq T, \quad \forall j \in J, \forall d \in D$$

(5)

$$\sum_{j \in J} x_{jd} r_{jk} \leq \sum_{e \in E_k} y_{es},$$

(6)

$$\forall k \in K, \forall d \in D, s = [(d-1)/n] + 1$$
$$y_{es} + \dots + y_{e(s+n-2)} + y_{e(s+n-1)} \leq 1, \quad \forall e \in E, \forall s \in S$$

(7)

$$\sum_{s \in S} z_{js} = 1, \quad \forall j \in J$$

(8)

$$Mz_{js} \geq \sum_{d \in D_s} x_{jd}, \quad \forall j \in J, \forall s \in S$$

(9)

其中: 式(1)表示调度以最小化装配工期为目标函数; 式(2)表示为任意任务安排的执行时间必须等于

其规定的执行时间; 式(3)表示任意任务一旦开始则不能中断; 式(4)为优先关系约束, 表示任意任务只有在其所有的紧前任务执行结束后才能开始执行; 式(5)表示任务的执行时间与装配总工期的关系; 式(6)表示某类人力资源在某一时刻中的供应量必须满足该时刻所有任务对该类人力资源的需求量; 式(7)为劳动力约束, 表示任意人力资源在连续 n 个班次中最多能被安排 1 个班次; 式(8)表示任务不允许跨班次执行; 式(9)定义决策变量 z_{js} 与决策变量 x_{jd} 之间的关系, 其中 M 为无穷大的数.

2 算法设计

目前解决 RCPSP 及其扩展问题的算法大多数是基于任务列表编码的编码形式. 所有任务在满足优先关系的前提下随机(或按规则)构成任务列表, 每一条任务列表代表一个解空间, 任务在任务列表中的顺序代表其执行顺序. 在已知任务列表的情况下, 一般采用串行调度进行解码, 即在满足资源约束的情况下尽早安排任务开始时间.

然而, 在求解 RCPSP-ET 问题时, 由于串行调度未考虑任务开始时间对人力资源利用情况的影响, 所以在任务列表编码方式对应的解空间下串行调度解码很难获得较优解. 图 1 所示为一个仅考虑单种人力资源的示例, 图中横坐标 t 为时间, 纵坐标 R 为人力资源编号. 根据图中的项目网络生成调度计划时, 因为班次 S_1 资源量充足, 任务 2、3 和 4 通过串行调度均安排在 $d = 0$ 时刻开始执行, 此时班次 S_1 的资源使用量为 5, 班次 S_2 的剩余资源量为 1. 由于不允许任务跨班次执行, 同时班次 S_2 的剩余资源量不足以执行任务 5, 所以任务 5 只能在之后的班次开始执行. 容易发现, 无论如何对换任务列表中任务 2、3、4 的顺序, 班次 S_1 的资源占用量始终为 5, 这几种情形下任务 5 都无法在班次 S_2 中执行.

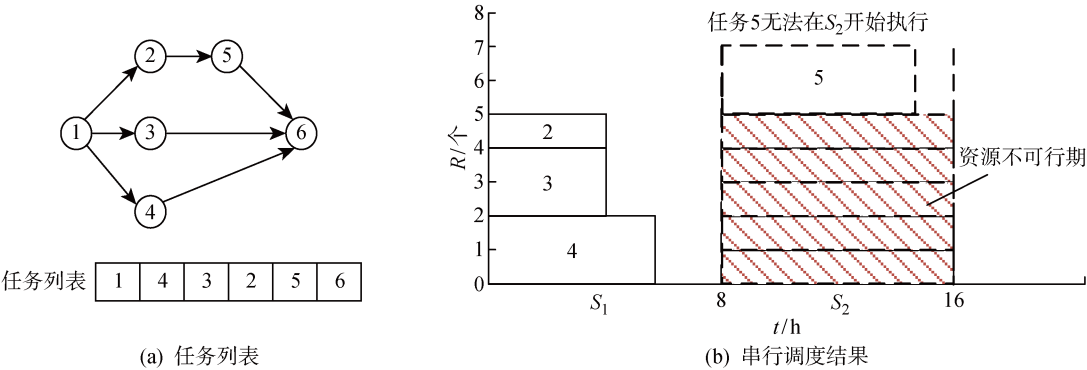


图 1 任务列表与调度结果示例

Fig. 1 Example of activity list and scheduling result

针对上述任务列表编码不足的缺点,本文设计了一种改进任务列表编码方式,并在此基础上设计了 BBGA. 改进任务列表编码是在不改变原优先关系的基础上,通过添加析取弧的方式来描述部分待确定的优先关系,从而提升串行调度解码的邻域搜索能力. 同时,针对遗传算法深度搜索能力弱的缺点, BBGA 在遗传算法所得最优染色体的基础上,对其进行分段深度搜索,进一步提高算法的求解质量.

2.1 改进任务列表编码方式

改进任务列表编码方式借鉴车间调度中析取弧的思想,通过添加析取弧的方式,在原先无直接或间接优先关系的任务之间增加额外的优先关系,使得解码过程中部分并行执行的任务转换为先后执行,达到降低当前班次人力资源使用量和提升资源利用率的目的. 改进任务列表将获得新的优先关系的任务组成一层,每一层对应染色体上的一个基因. 编码过程要求:① 后一层中不得包含前一层任务的直接或间接紧前任务;② 同一层中的任务在添加析取弧之前不能存在优先关系;③ 同一层中析取弧方向由

最左边的任务依次指向最右边的任务;④ 同一层中所有任务的工期和不能大于一个班次的工期之和. 同一层中后序任务以前序任务的结束时间作为实际最早可开始时间. 只有前一层中的所有任务都被执行完成后,后一层的任务才可以安排开始.

改进任务列表编码过程如图 2 所示. 以图 1 中的项目网络为例,改进任务列表第 1 层和第 2 层分别选择任务 1 和 4. 进行第 3 层编码时,可选择任务为任务 2 和 3. 根据改进任务列表的性质,任务 2 和 3 工期之和为 8,与班次时间跨度相等,因此第 3 层可选择编码为 {2}、{3}、{2,3} 和 {3,2}. 此时,随机选择 {3,2} 作为第 3 层编码. 按照上述方法继续分层,得到的改进任务列表如图 2(a) 所示. 根据染色体的编码,为项目网络添加 3→2 的析取弧. 图 2(b) 给出了通过串行调度对改进任务列表进行解码所得的调度计划. 相比图 1 中的调度计划,图 2(b) 中的调度计划能够通过延迟任务 2 的开始时间来降低班次 S_1 中的人力资源使用量,使得班次 S_2 有足够的资源来执行任务 5. 此时任务 5 的开始时间为 8,项目工期 $T=15$.

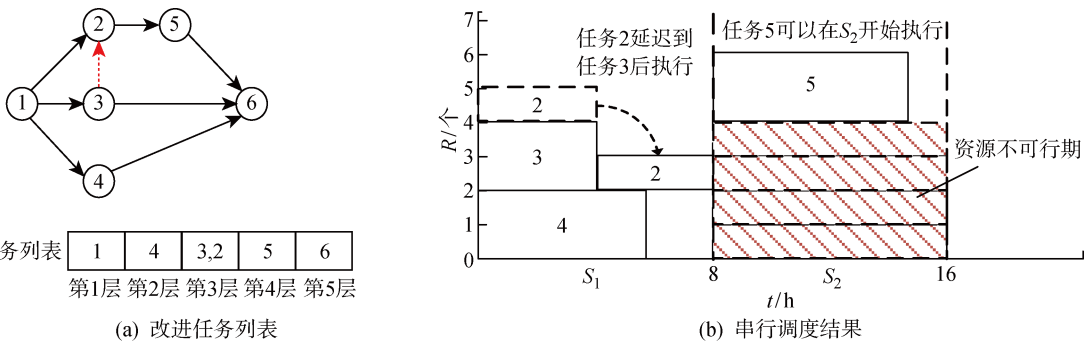


图 2 改进任务列表与调度结果示例

Fig. 2 Example of improved activity list and scheduling result

2.2 BBGA

BBGA 结合 GA 的广度搜索优势与分支定界算法的深度搜索优势,以改进任务列表编码方式作为 GA 染色体的表达方式,通过 GA 获得初始染色体. 在此基础上,分支定界搜索框架将该染色体作为搜索树中的初始分支对该染色体进行分段局部回溯,并将每一阶段搜索得到的最优染色体作为下一阶段的初始染色体,通过这种邻域搜索机制进一步优化调度结果.

2.2.1 遗传算法 标准 GA 分为 5 个步骤:① 初始化种群;② 交叉;③ 变异;④ 选择;⑤ 判断是否满足终止条件,不满足返回②,满足则终止算法. 为保证进行交叉变异操作之后所得的改进任务列表仍满足编码要求,本文对传统的交叉变异算子进行改

进,交叉操作与变异操作如图 3 所示.

交叉算子采用单点交叉,在染色体层与层之间随机生成交叉点位置. 子代染色体保留父代交叉点前的编码顺序,剩余编码顺序根据母代剔除已选任务所保留的编码顺序决定.

变异算子采用对换变异,对换层允许进行变异操作的条件是两个对换层中间所有任务均不存在优先关系.

2.2.2 启发式进度生成机制 根据改进任务列表编码方式的特点,本文设计新的启发式进度生成方法对相应的染色体(编码)进行解码. 具体决策方法如下:按照层级从前往后选择任务,对于每层编码中的任务,按照顺序选择,第一个任务按尽早开始原则决策资源的选择,后序任务将前序任务的结束时间

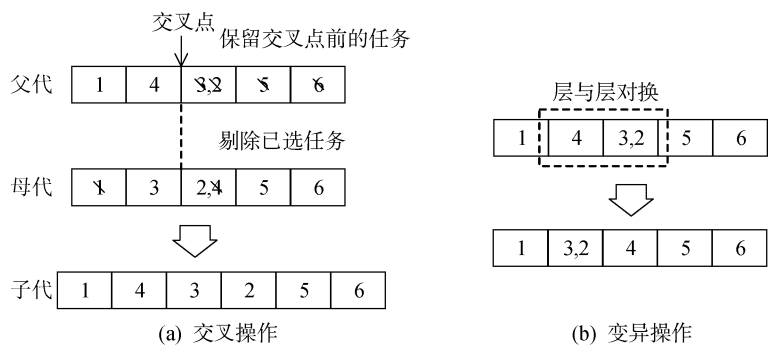


图 3 交叉操作与变异操作

Fig. 3 Crossover operation and mutation operation

作为最早可开始时间进行安排. 按此方法逐步确定每层中任务的开始时间, 获得完整的调度计划. 具体的算法步骤如下:

- 步骤 1 令染色体层级编号 $g=1$.
- 步骤 2 读取第 g 层编码中的任务集合 A_g .
- 步骤 3 按照顺序从小到大选择序号最小的任务 $j_{\min} \in A_g$, 按尽早开始原则为任务 j_{\min} 分配资源, 计算开始时间, 更新分配后的资源占有量, 更新 $A_g = A_g - j_{\min}$.
- 步骤 4 如果 $A_g \neq \emptyset$, 转步骤 5; 否则, 转步骤 7.
- 步骤 5 将已安排的前序任务 j 的结束时间 d_j^{FT} 作为后序任务 j' 的最早开始时间 $d_{j'}^{\text{EST}}$, 为后序任务 j' 分配资源, 更新分配后的资源占有量, 更新 $A_g = A_g - j'$.
- 步骤 6 如果 $A_g \neq \emptyset$, 转步骤 5; 否则, 转步骤 7.
- 步骤 7 如果调度计划未完成, $g=g+1$, 转步骤 2; 否则结束算法.

2.2.3 分支定界搜索框架 分支定界搜索框架将改进任务列表编码的层级关系转换为搜索树中的树节点关系, 通过事先设定好的步长进行多阶段的局部分支(分支过程以深度优先)与回溯. 其中, 分支定界搜索框架以遗传算法所求得的最佳染色体作为第一阶段的初始解, 之后保留每一阶段的最佳染色体作为下一阶段的初始解.

(1) 算法步骤. 分支定界搜索框架的具体步骤如下所示:

- 步骤 1 初始化步长 Q , 分支起始层 $n=2$, 阶段 $s=1$, 初始染色体设为遗传算法中适应值最小的染色体. L_{best} 为该染色体任务列表, M 为该染色体的适应值, W 为染色体总层数, F 为算法终止判断符号.
- 步骤 2 如果 $Q+n < W$, 则以第 n 层到第 $Q+n-1$ 层中的任务组成回溯任务集合 J_s , 保留其余层

的任务序列, 令 F 为假; 否则, 以第 n 层到第 $W-1$ 层中的任务组成回溯任务集合 J_s , 保留其余层的任务序列, 令 F 为真.

- 步骤 3 令 $g=n-1$.
- 步骤 4 确定节点 g 的候选任务组合集合 C_g , 从 C_g 中随机选择任务集合 A_g 生成新的节点 $g+1$, 更新 $C_g = C_g - A_g$. 通过支配规则判断能否截除, 如果截除则转步骤 6.
- 步骤 5 如果存在未分支的任务 $j \notin A_g$, 则表明节点 g 可以继续分支, 令 $g=g+1$, 转步骤 4. 否则, 将得到的分支与步骤 2 所保留其余层的任务序列合并, 组成新的染色体. 调用启发式进度生成机制对该染色体进行解码. 如果目标函数值优于 M , 则更新 M 并记录最优任务列表 L_{best} , 转步骤 6.
- 步骤 6 回溯搜索树, 如果找到某节点 g 的 $C_g \neq \emptyset$, 则转至步骤 4; 否则, 结束分支, 转步骤 7.
- 步骤 7 若 F 为假, 则更新 $s=s+1$, $n=Q+n$, 转至步骤 2; 否则满足终止条件, 转步骤 8.
- 步骤 8 若 F 为真, 则满足终止条件, 结束算法.

图 4 所示为分支定界搜索框架的示例, 初始染色体编码为 $\{1\}$ 、 $\{4\}$ 、 $\{3, 2\}$ 、 $\{5\}$ 、 $\{6, 7\}$ 、 $\{8\}$, 设定步长 $Q=2$, 分支起始层 $n=2$. 在阶段 $s=1$ 时, 回溯第 2 和第 3 层任务, 回溯任务集合 $J_1 = \{2, 3, 4\}$, 通过回溯分支得到阶段 $s=1$ 的最优染色体为 $\{1\}$ 、 $\{4\}$ 、 $\{2, 3\}$ 、 $\{5\}$ 、 $\{6, 7\}$ 、 $\{8\}$. 在阶段 $s=2$ 时, 将阶段 $s=1$ 的最优染色体作为初始染色体, 回溯第 4 和第 5 层任务, 回溯任务集合 $J_2 = \{5, 6, 7\}$, 通过回溯分支得到阶段 $s=2$ 处的最优染色体为 $\{1\}$ 、 $\{4\}$ 、 $\{2, 3\}$ 、 $\{5\}$ 、 $\{6\}$ 、 $\{7\}$ 、 $\{8\}$. 由于此时 $Q+n=6$, 满足终止条件, 因此将分支定界算法得到的最优染色体输出.

(2) 支配规则. 为了提升分支定界算法的效率, 本文提出 2 种支配规则, 用于分支过程中剪除被支配的分支. 为了方便支配规则的描述, 事先给出以下

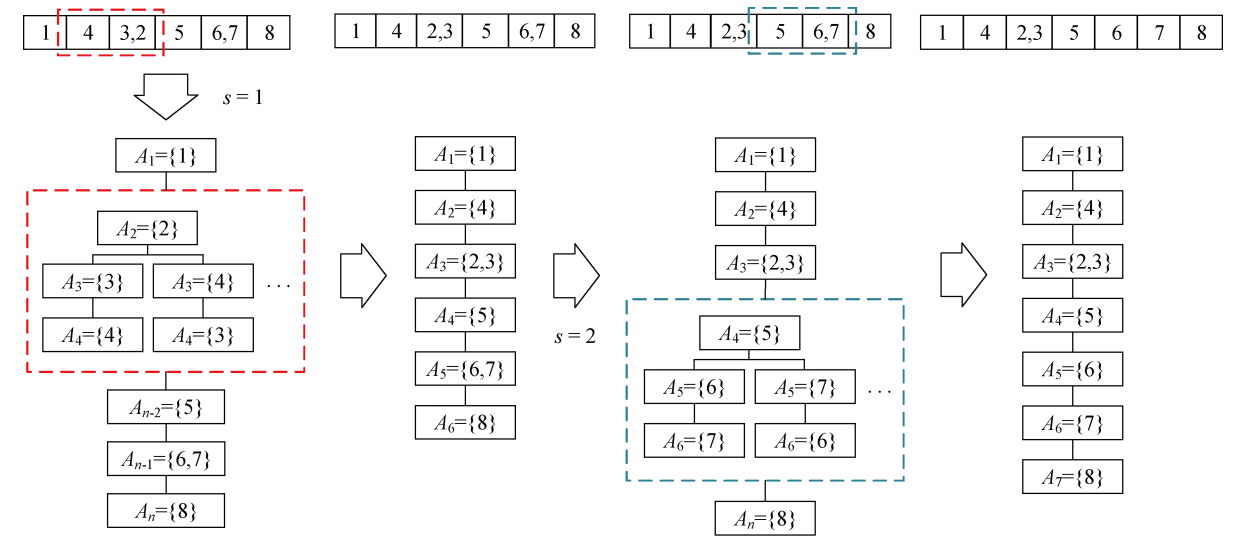


图 4 分支定界搜索框架示例

Fig. 4 Example of branch-and-bound-search framework

定义:

定义 1 记节点 q 为当前节点,如果节点 q 由节点 q' 分支得到,则称节点 q' 为节点 q 的直接根节点.

根据定义 1 的描述,记直接根节点调度的任务集合为 $A_{q'}$,当前节点调度的任务集合为 A_q ,所有任务 $i \in A_{q'}$ 的工期之和为 $d^{q'}$,所有任务 $j \in A_q$ 的工期之和为 d^q ,班次跨度为 d^s .

定义 2 若当前节点 q 中的所有任务 $j \in A_q$ 均可以合并入其直接根节点 q' ,且合并后的任务列表与合并前的任务列表解码所得的调度计划相同,则称节点 q 为重复节点.

依据上述定义,给出 2 种支配规则的描述:
规则 1(重复节点规则) 当前节点 q 与其直接根节点 q' 满足以下 3 个条件时,节点 q 为重复节点,不再对节点 q 进行分支.

条件 1 对于任意节点 q 与其直接根节点 q' ,如满足条件 $d^q + d^{q'} > d^s$,则当前节点为不重复情况.

证明 如果满足 $d^q + d^{q'} > d^s$,则表明两节点中所有任务的工期和大于班次时间,当前节点中的任

务无法全部合并入其直接根节点,因此当前节点不为重复情况.

条件 2 如果任务 $j \in A_q$ 与任意任务 $i \in A_{q'}$ 存在时序约束关系,则当前节点不为重复情况.

证明 如果两节点中的任务间存在时序约束关系,则当前节点调度的任务无法全部合并入其根节点,因此当前节点不为重复情况.

条件 3 如果不满足条件 1 和条件 2,计算任务 A_q 中绝对顺序最大的任务 i_{\max} 的最早结束时间 $d_{i,\max}^{\text{EST}}$.如果存在任务 $j \in A_q$,满足 $d_{i,\max}^{\text{EST}} + d_j \leq nd^s$ 且 $d_j^{\text{EST}} \leq d_{i,\max}^{\text{EST}}$,则当前节点不为重复情况.

证明 如果不能满足条件 1 和条件 2,则表明 $d^q + d^{q'} < d^s$ 且当前节点的所有任务 $j \in A_q$ 与任务 $i \in A_{q'}$ 不存在时序约束关系.图 5(a)为任务 $j \in A_q$ 满足 $d_{i,\max}^{\text{EST}} + d_j \leq nd^s$ 且 $d_j^{\text{EST}} \leq d_{i,\max}^{\text{EST}}$ 时不合并的调度情况.如图 5(b)所示,如果将当前节点任务 j 并入根节点,则任务 j 会在任务 i 结束后执行,所以合并前和合并后的调度计划必不相同,因此不为重复情况.

规则 2(高界规则) 如果当前节点的任务的完

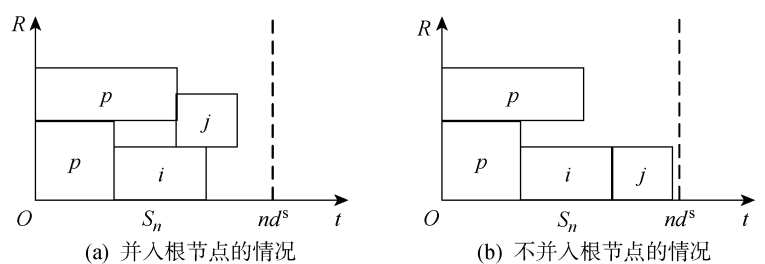


图 5 条件 3 示例

Fig. 5 Example of Condition 3

成时间超过目前所得最优解,则该节点不再分支.

记目前所得的项目最短工期为 M^{UB} . 如果任务 $j \in A_g$ 的完成时间 $F_j > M^{\text{UB}}$, 则节点 g 不再分支. 新的完整调度计划完成时, 如果满足 $F_j < M^{\text{UB}}$, 则更新 $M^{\text{UB}} = F_j$.

3 数据实验

本文算法运用 C# (Visual Studio 2015)编程实现,测试实验在 Internet Core i7 处理器,3.4 GHz 主频,8 GB 内存的测试平台上进行. 本文的测试算例选取自 PSPLIB 算例库,由于人力资源排班引入的资源不可行期会造成项目工期较长,所以需对算例进行改造. 改造参数包括对班次 n 、资源系数 F 、

班次长度 S 和分支定界步长 Q 进行设定,其中资源系数 F 指实际使用资源量与算例库原始资源量的比值,即 $F = R/R_0$. 对于小规模算例规模(J10,J12,J14,J16,J18),本文设定参数 n 、 F 、 S 和 Q 分别为 3、1、8、4. 对于中大规模算例(J30,J60,J90),本文设定参数 n 、 F 、 S 和 Q 分别为 3、2、11、5.

3.1 与 CPLEX 的对比

为了验证算法的有效性,本文将 BBGA 与 CPLEX 和 GA 进行对比,如表 1 所示. 每个规模均取 30 个算例,每 5 个算例组成一组进行对照试验,设定 CPLEX 的运行阈值为 7 200 s. G_{value} 表示计算结果; G_{time} 表示算法的运算时间; G_{gap} 表示不同算法所得解与本组最优解间的差值.

表 1 小规模算例数值实验结果
Tab. 1 Numerical results of small scale case

算例	组别	$G_{\text{value}}/\text{h}$			G_{time}/s			$G_{\text{gap}}/\%$	
		CPLEX	GA	BBGA	CPLEX	GA	BBGA	GA	BBGA
J10	1	25.9/25.9	26.9	26.5	31.9	4.8	5.4	4.0	2.4
	2	69.5/69.5	69.5	69.5	678.4	5.6	6.9	0.0	0.0
	3	82.7/79.9	82.5	82.4	2 124.2	8.8	14.7	−0.2	−0.3
	均值	59.4/58.4	59.6	59.5	944.9	6.4	9.0	0.3	0.2
J12	1	66.0/60.7	61.7	61.6	1 466.6	10.6	13.5	−6.6	−6.7
	2	41.4/41.4	42.5	42.1	186.1	10.4	15.2	2.7	1.6
	3	38.6/38.6	39.0	38.9	250.7	10.6	15.3	1.1	0.8
	均值	48.7/46.9	47.7	47.5	634.5	10.5	14.7	−2.1	−2.5
J14	1	62.6/53.7	57.4	56.7	3 259.8	10.4	81.0	−8.3	−9.5
	2	48.9/48.9	50.3	49.5	619.5	11.9	22.7	2.9	1.2
	3	109.9/90.4	103.8	103.3	3 422.8	11.3	42.2	−5.6	−6.0
	均值	73.8/64.3	70.5	69.8	2 434.0	11.2	48.6	−4.5	−5.4
J16	1	57.7/56.7	60.2	59.5	1 035.4	12.9	22.3	4.3	3.2
	2	67.1/52.0	62.2	61.9	2 839.2	13.6	55.0	−7.3	−7.8
	3	61.6/58.4	62.4	61.5	1 502.3	13.1	19.7	1.3	−0.1
	均值	62.1/55.7	61.6	61.0	1 792.3	13.2	32.3	−0.8	−1.8
J18	1	86.5/64.7	76.7	75.3	5 972.9	10.7	76.2	−11.4	−13.0
	2	59.9/53.8	52.2	51.0	2 554.7	11.8	104.1	−12.9	−14.9
	3	49.2/46.0	49.3	48.5	1 314.2	12.0	67.8	0.3	−1.5
	均值	63.6/53.3	59.4	58.3	3 280.6	11.5	82.7	−6.6	−8.3

从表 1 看出,在小规模算例实验中,BBGA 在可接受时间内能够求得较优的结果. 对于大多数 J10 和 J12 规模的算例,CPLEX 可求得问题的最优解,而 GA 与 CPLEX 之间的 G_{gap} 值小于 4.0%,BBGA 与 CPLEX 之间的 G_{gap} 值小于 2.4%. 并且随着算例

规模以及复杂度的增加(J14~J18),CPLEX 在许多情况下无法求得最优解,而 GA 和 BBGA 仍可求得较优解. 虽然 BBGA 计算时间有所提升,但是相比较 GA,其求解质量能够提升 2%左右,表现出一定的优势.

3.2 与现有文献的对比

由于 CPLEX 无法求解中大规模问题,同时关于人力资源调度与 RCPSP 结合问题的算法研究又十分有限,所以对于大规模算例,本文借鉴文献[18]中基于插入操作的领域搜索思想,设计了基于局部

搜索(LS)的改进启发式算法作为对比来验证本文算法的有效性.其中启发式规则选取最早开始时间(EST)规则和最短执行时间(SPT)规则.表 2 给出不同算法对于大规模问题的实验结果对比,其中每组包括 10 个算例.

表 2 中大规模算例数值实验结果
Tab. 2 Numerical results of medium and large scale cases

算例	组别	启发式+LS				GA			BBGA		
		EST+LS	$G_{\text{gap}}/\%$	SPT+LS	$G_{\text{gap}}/\%$	$G_{\text{value}}/\text{h}$	G_{time}/s	$G_{\text{gap}}/\%$	$G_{\text{value}}/\text{h}$	G_{time}/s	$G_{\text{gap}}/\%$
J30	1	96.1	7.5	95.8	7.1	92.2	14.9	3.0	89.4	61.5	0.0
	2	86.9	5.2	87.7	6.2	84.4	14.5	2.1	82.6	55.9	0.0
	3	74.7	7.3	73.7	5.9	70.8	15.0	1.7	69.6	23.6	0.0
	4	65.2	5.2	66.4	7.1	63.7	16.5	2.8	62.0	24.5	0.0
	5	80.8	5.0	81.2	5.4	78.9	16.1	2.3	77.0	28.4	0.0
	均值	80.7	6.0	81.0	6.4	78.0	15.4	2.4	76.1	38.9	0.0
J60	1	103.9	6.2	103.5	5.8	100.0	47.4	2.2	97.8	600.7	0.0
	2	81.4	7.1	81.8	7.6	77.7	71.6	2.1	76.1	196.9	0.0
	3	95.9	7.5	94.0	5.4	92.3	46.7	3.5	89.1	201.6	0.0
	4	72.0	6.9	72.1	7.0	68.7	44.7	2.0	67.3	141.5	0.0
	5	65.5	6.0	64.3	4.0	62.1	44.4	1.3	61.9	600.3	0.0
	均值	83.7	6.7	83.1	5.9	80.2	50.9	2.2	78.4	348.2	0.0
J90	1	136.7	4.7	139.9	7.2	134.0	82.9	2.5	130.6	675.5	0.0
	2	106.4	4.9	105.7	4.3	103.5	84.9	2.1	101.4	2 134.5	0.0
	3	135.3	4.1	136.1	4.7	131.7	104.9	1.2	130.0	1 565.1	0.0
	4	128.5	6.2	129.4	7.0	124.8	95.5	3.0	121.1	1 100.8	0.0
	5	137.5	7.2	134.7	5.1	130.2	89.5	1.5	128.1	1 774.0	0.0
	均值	128.9	5.4	129.2	5.6	124.8	91.5	2.1	122.2	1 450.0	0.0

由表 2 可知,在求解大规模算例时,分支定界搜索框架对于遗传算法能够取得 2% 以上优化效果,同时,基于局部搜索的改进启发式算法与 BBGA 的偏差约为 6%,证明了 BBGA 的优越性.但是,随着算例规模的增加,BBGA 计算时间增长较快,并且每个算例的计算时间差异较大,其原因是回溯集合的数目随着算例规模的增加而增长,同时不同算例下回溯集合的任务数目差异较大.

3.3 支配规则效率分析

支配规则作为 BBGA 的重要组成部分,需要验证其对算法运算效率的影响.表 3 给出 J12 中 $Q=12$ 时,本文算法分别同时使用规则 1 和规则 2(Rule 1&2)、仅使用规则 1(Rule 1)、仅使用规则 2(Rule 2)和不使用支配规则(None)时的运行时间以及搜

索树的分支数目.其中设定参数 n 、 F 、 S 和 Q 分别为 3、1、8、4,每组实验包含 3 个算例, G_{num} 栏表示在某一规则下搜索树的分支数目.图 6 显示了不同支配规则下算法的运行时间相对不使用支配规则时算法计算时间所占的百分比.

由表 3 可知,规则 1 和规则 2 均能够通过有效减少搜索树分支数目来降低算法的运算时间,且当规则 1 和规则 2 同时使用时算法的运算时间最短.由图 6 可知,使用单一规则时,算法计算时间能够缩减到无分支规则时计算时间的 60% 以下(最低为 14%);当同时使用两种规则时,算法计算时间能够缩减到无分支规则时计算时间的 30% 以下(最低为 6%),验证了本文设计的支配规则能够有效地降低算法的计算时间,提高算法的计算效率.

表 3 不同支配规则效率分析结果
Tab. 3 Results of efficiency analysis of different dominating rules

算例	组别	Rule 1&2		Rule 1		Rule 2		None	
		G_{time}/s	G_{num}	G_{time}/s	G_{num}	G_{time}/s	G_{num}	G_{time}/s	G_{num}
J12	1	69.8	91 166	130.5	101 651	125.4	165 768	258.7	183 058
	2	100.5	148 495	172.1	170 071	338.2	461 511	673.3	546 857
	3	21.3	39 459	113.6	117 319	53.9	96 019	374.5	298 307
	4	39.4	60 467	106.3	114 469	126.3	182 287	355.5	307 382
	5	136.4	225 213	344.1	350 513	318.0	466 019	880.3	731 568
	6	65.7	96 214	111.4	102 103	155.0	211 986	277.5	229 213
	7	40.7	69 500	151.4	157 978	100.3	161 620	485.9	400 475
	8	96.1	154 393	224.5	207 857	153.8	230 699	475.6	323 940
	9	20.4	31 203	37.3	35 982	58.6	82 072	131.4	96 294
	10	242	363 001	420.6	380 630	487.1	680 001	993.5	708 440
	均值	83.2	127 911	181.2	173 857	191.7	273 798	490.6	382 553

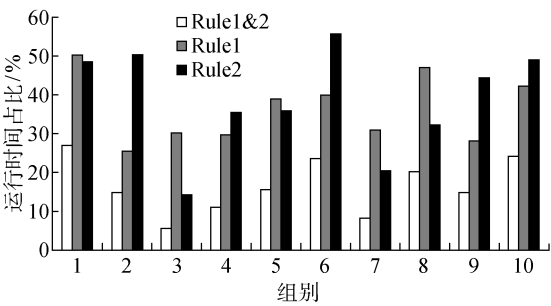


图 6 不同支配规则下计算时间相对无支配规则下计算时间的百分比
Fig. 6 Comparison of percentage of running time under different dominating rules with that under no dominating rule

3.4 集成决策有效性分析

为了验证 RCPSP-ET 集成决策对实际装配生产中装配计划制定以及人力资源排班的适应性与有效性,本文将 RCPSP-ET 集成决策与传统装配生产过程中“人员-任务”顺序决策进行比较。“人员-任务”顺序决策分为两个阶段:人员安排阶段与任务安排阶段.在人员安排阶段,各班次的人员类型及其数量根据企业预定的规则进行分配,此处假设各班次人员数量与种类保持一致.在任务安排阶段,根据上一阶段制定的排班情况,在满足各约束的条件下为各装配任务安排合理的开始时间以及执行人员.为了保证一定的求解质量,此处采用基于改进任务列表编码的遗传算法(GA-SD)生成装配计划,其中解码过程与 2.2.2 节近似,仅去除后续的人员排班过

程.表 4 给出“人员-任务”顺序决策与 RCPSP-ET 集成决策在中大规模问题下的实验结果对比, G'_{gap} 栏表示不同算法所得解与 GA-SD 之间的差值.

由表 4 可知,基于 RCPSP-ET 集成决策的 GA 以及 BBGA 较基于“人员-任务”顺序决策的 GA-SD 分别能够取得均值约为 5% 和 7% 的优化效果.同时,相比 GA,GA-SD 计算时间没有明显的缩短.因此,RCPSP-ET 集成决策在决策方式上优于传统的“人员-任务”顺序决策,更加适用于实际装配生产中装配计划制定以及人力资源排班.

3.5 实例分析

为了更直观地阐明本文方法的有效性,本节以某大型工业品装配过程中某一部分任务为例进行分析.该部分共计 16 项装配任务,装配任务优先关系如图 7 所示.其中,企业根据任务之间的顺序关系为各任务安排编号,括号内的数字分别表示任务所需操作时间和任务执行时所需的装配人员数量,例如“H03A(2,4)”表示任务 H03A 的操作时间为 2,所需装配人员数量为 4.任务 H00 和任务 H05 分别表示虚拟开始任务和虚拟结束任务,其任务所需操作时间和装配人员需求量均为 0.装配人员以两班制的形式执行装配任务,其总数为 11 人,每一班次的长度为 8 h.假定开始阶段有 4 名装配人员处于休息状态.

根据上述装配任务优先关系图以及问题描述,采用本文 BBGA 和传统任务列表编码的遗传算法(GA-AL)生成不同的任务调度计划,结果分别如

表 4 中大规模算例数值实验结果

Tab. 4 Numerical results of medium and large scale cases

算例	组别	GA-SD			GA		BBGA	
		$G_{\text{value}}/\text{h}$	G_{time}/s	$G'_{\text{gap}}/\%$	$G_{\text{value}}/\text{h}$	$G_{\text{gap}}/\%$	$G_{\text{value}}/\text{h}$	$G'_{\text{gap}}/\%$
J30	1	96.1	13.9	0.0	92.2	−4.1	89.4	−7.0
	2	89.0	13.1	0.0	84.4	−5.2	82.6	−7.2
	3	75.3	14.5	0.0	70.8	−6.0	69.6	−7.6
	4	66.2	15.7	0.0	63.7	−3.8	62.0	−6.3
	5	85.6	15.4	0.0	78.9	−7.8	77.0	−10.1
	均值	82.4	14.5	0.0	78.0	−5.4	76.1	−7.6
J60	1	103.9	93.4	0.0	100.0	−3.8	97.8	−5.9
	2	81.4	72.6	0.0	77.7	−4.6	76.1	−6.5
	3	97.8	82.9	0.0	92.3	−5.6	89.1	−8.9
	4	71.8	63.1	0.0	68.7	−4.3	67.3	−6.3
	5	65.7	57.9	0.0	62.1	−5.5	61.9	−5.8
	均值	84.1	74.0	0.0	80.2	−4.7	78.4	−6.7
J90	1	143.9	77.1	0.0	134.0	−6.9	130.6	−9.2
	2	107.3	79.0	0.0	103.5	−3.5	101.4	−5.5
	3	141.7	98.0	0.0	131.7	−7.1	130.0	−8.3
	4	133.7	88.7	0.0	124.8	−6.7	121.1	−9.4
	5	135.6	84.6	0.0	130.2	−4.0	128.1	−5.5
	均值	132.4	85.4	0.0	124.8	−5.6	122.2	−7.6

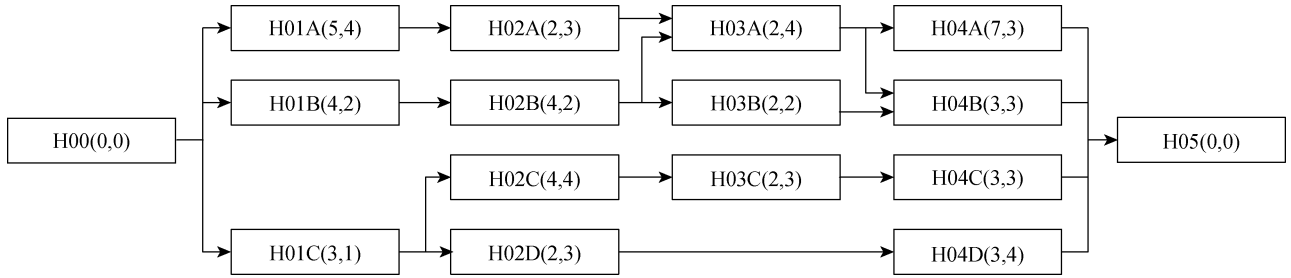


图 7 某大型工业品部分装配任务优先关系图

Fig. 7 Priority for partial assembly activities of a large industrial product

图 8 和 9 所示。图中阴影表示装配人员受劳动法规约束而产生的人员不可行期，即装配人员处于休息状态。

可以看出，该实例下 BBGA 所求得的装配任务总工期 $T=27$ ，而 GA-AL 求得的装配任务总工期 $T=29$ 。通过进一步分析，发现图 8 和 9 中于班次 S_1 内执行的任务相同，其中图 9 中班次 S_1 的人员使用量为 7，而图 8 中班次 S_1 的人员使用量为 6。该情况的出现是因为 GA-AL 采用的传统任务列表编码方

式仅考虑了原始的任务优先关系，所以采用串行调度解码时任务 H01C 只能在 $T=0$ 时刻开始。而 BBGA 采用的改进任务列表编码能够在任务 H01A 和任务 H01C 之间添加析取弧，因此任务 H01C 能够延迟到 H01A 后开始。班次 S_1 内人员使用量的降低使得班次 S_2 中处于不可行期的人员减少，从而为班次 S_2 保留了更多的人员可用量，使得任务 H03C 和任务 H03B 能够提前至班次 S_2 中同时执行。该结果直观地验证了本文 BBGA 的有效性。

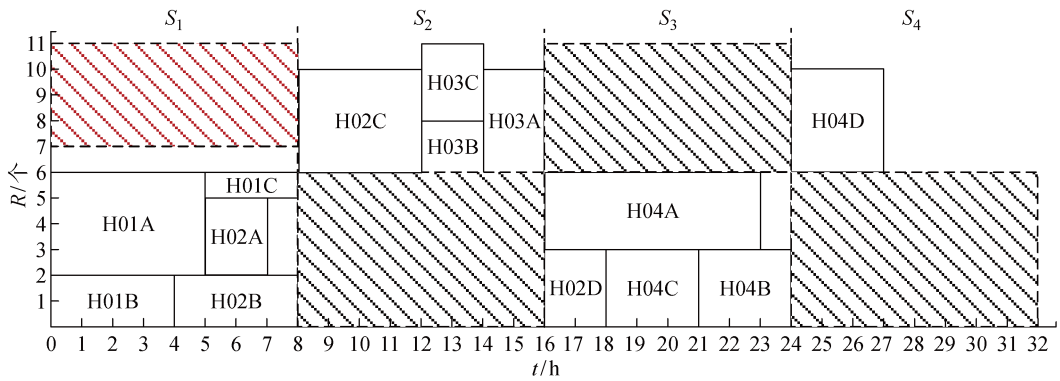


图 8 BBGA 求得的调度计划
Fig. 8 Schedule obtained by BBGA

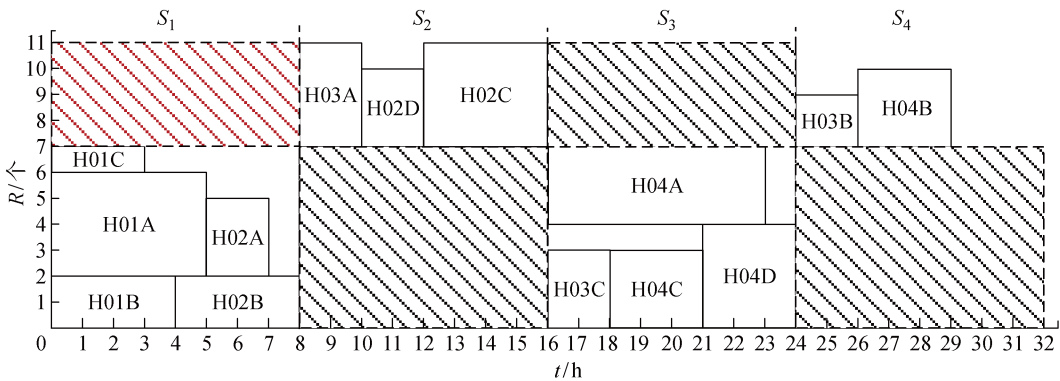


图 9 GA-AL 求得的调度计划
Fig. 9 Schedule obtained by GA-AL

4 结语

本文以考虑人力资源排班的资源受限项目调度问题为研究对象,建立离散时间下考虑劳动力约束的数学模型.针对问题的特点,提出了改进任务列表编码方式.为了提升算法的深度搜索能力,根据编码特点设计分支定界搜索框架,对遗传算法得到的染色体进行分段深度搜索.在支配规则中,通过分析判断重复节点剪除被支配的分支,避免计算不必要的分支,提高算法的计算时间.数据实验表明,对于小规模算例,本文算法与 CPLEX 最优解的差值保持在 3.2% 以下,部分算例相比遗传算法能提高 2% 优化效果;对于大规模算例,分支定界算法能够在遗传算法所得解的基础上取得 2% 以上优化效果;在计算时间方面,本文设计的支配规则能够将算法计算时间降低到未使用支配规则的 30% 以下;与此同时,相比较传统的“人员-任务”顺序决策,RCPSP-ET 集成决策能够更加有效地优化决策过程,获得更好的人员排班计划和任务执行计划.未来可以在考虑人力资源排班的资源受限项目问题中进一步研究人

力资源多技能对排班和项目工期的影响.

参考文献:

- [1] BLAZEWICZ J, LENSTRA J, KAN A. Scheduling subject to resource constraints: Classification and complexity[J]. **Discrete Applied Mathematics**, 1983, 5(1): 11-24.
- [2] BRUCKER P, KNUST S, SCHOO A, *et al.* A branch and bound algorithm for the resource-constrained project scheduling problem[J]. **Mathematical Methods of Operations Research**, 2000, 52(3): 413-439.
- [3] REYCK B, HERROELEN W. A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations[J]. **European Journal of Operational Research**, 1998, 111(1): 152-174.
- [4] ZAMANI R. A competitive magnet-based genetic algorithm for solving the resource-constrained project scheduling problem[J]. **European Journal of Operational Research**, 2013, 229(2): 552-559.
- [5] CHEN R. Particle swarm optimization with justifica-

- tion and designed mechanisms for resource-constrained project scheduling problem[J]. **Expert Systems with Applications**, 2011, 38(6): 7102-7111.
- [6] 何杰光, 陈新度, 陈新, 等. 求解资源受限项目调度的双种群准粒子群算法[J]. **计算机集成制造系统**, 2015, 21(9): 2446-2457.
- HE Jieguang, CHEN Xindu, CHEN Xin, *et al.* Double-population quasi particle swarm optimization for solving resource-constrained scheduling problem[J]. **Computer Integrated Manufacturing Systems**, 2015, 21(9): 2446-2457.
- [7] WANG L, FANG C. A hybrid estimation of distribution algorithm for solving the resource-constrained project scheduling problem[J]. **Expert Systems with Applications**, 2012, 39(3): 2451-2460.
- [8] HE J, CHEN X D, CHEN X. A filter-and-fan approach with adaptive neighborhood switching for resource-constrained project scheduling[J]. **Computers and Operations Research**, 2016, 71(1): 71-81.
- [9] BERGH J, BELIËN J, BRUECKER P, *et al.* Personnel scheduling: A literature review[J]. **European Journal of Operational Research**, 2013, 226(3): 367-385.
- [10] HANAFI R, KOZAN E. A hybrid constructive heuristic and simulated annealing for railway crew scheduling[J]. **Computers and Industrial Engineering**, 2014, 70(1): 11-19.
- [11] AL-YAKOOB S, SHERALI H. Mixed-integer programming models for an employee scheduling problem with multiple shifts and work locations[J]. **Annals of Operations Research**, 2007, 155(1): 119-142.
- [12] BILGIN B, CAUSMAECKER P, ROSSIE B, *et al.* Local search neighbourhoods for dealing with a novel nurse rostering model[J]. **Annals of Operations Research**, 2012, 194(1): 33-57.
- [13] MATTAA R, PETERS E. Developing work schedules for an inter-city transit system with multiple driver types and fleet types[J]. **European Journal of Operational Research**, 2009, 192(3): 852-865.
- [14] DREZET L, BILLAUT J. A project scheduling problem with labour constraints and time-dependent activities requirements[J]. **International Journal of Production Economics**, 2008, 112(1): 217-225.
- [15] ARTIGUES C, GENDREAU M, ROUSSEAU L, *et al.* Solving an integrated employee timetabling and job-shop scheduling problem via hybrid branch-and-bound[J]. **Computers and Operations Research**, 2009, 36(8): 2330-2340.
- [16] GUYON O, LEMAIRE P, PINSON E, *et al.* Cut generation for an integrated employee timetabling and production scheduling problem[J]. **European Journal of Operational Research**, 2010, 201(2): 557-567.
- [17] AHMADI-JAVID A, HOOSHANGI-TABRIZI P. Integrating employee timetabling with scheduling of machines and transporters in a job shop environment: A mathematical formulation and an anarchic society optimization algorithm[J]. **Computers and Operations Research**, 2017, 84(1): 73-91.
- [18] 潘全科, 高亮, 李新宇. 流水车间调度及其优化算法[M]. 武汉: 华中科技大学出版社, 2013.
- PAN Quanke, GAO Liang, LI Xinyu. Flow shop scheduling and optimization algorithm[M]. Wuhan: Huazhong University of Science and Technology Press, 2013.

(本文编辑:陈晓燕)