**考虑资源技能水平可变的多技能资源受限项目调度问题研究**

摘要：在本文中，针对多技能资源受限项目调度问题，提出了考虑资源学习技能水平可变的模型，研究目标是最小化项目工期和成本。并建立了混合整数非线性双目标规划模型。当考虑资源技能水平可变时随着调度中任务的执行资源技能水平提高，这将使得在后续的任务安排中任务资源分配具有更多选择，这使得可以有更多的调度方案选择。本文使用学者Myszkowski提出的iMose案例进行数值测试，使用NSGA\_II进行求解。当考虑资源技能水平获相比原始的能够获得更前沿的帕累托解，这证明当考虑资源学习时，能够更好的利用资源获得更小的项目工期和成本的调度方案。同时本文提出一种混合变邻搜索算法用于求解，设计了五种邻解搜索算子，能够对潜在的解进行更广的搜索，通过系统的改变邻解搜索算子能够有效的逃离局部最优。同时提出一个问题具体特点的资源搜索算子能够获得更好的资源分配。通过数值测以NSGA\_II作对比，证明了提出的算法的有效性。

关键字:资源学习|技能可变、多技能、RCPSP、混合遗传变邻搜索算法、遗传算法二代

**1简介**

资源受限项目调度问题（RCPSP）是研究最为广泛的的调度问题之一，其目标是寻找到一种资源-任务分配方案使得项目计划耗费成本最少和完工时间最短。RCPSP问题作为组合优化的NP难问题，鼓励研究学者找到足够好的方法，以便能够在有限的多项式计算时间内产生近似（次）最优解[1]。

多技能资源受限项目调度问题（MSRCPSP）是对RCPSP问题添加技能约束的一个扩展[2]。在MSRCPSP问题中，每种资源都有几种一定水平的技能，每个任务都需要一种一定水平的技能去执行。这种类型的问题有非常多的应用场景，典型应用于人力资源调度中，例如，软件工程项目[3]中需要具有不同技能的员工进行开发，每个员工有不同类型和不同熟练程度的技能，如数据库设计，编程，测试等等。项目有许多个任务组成，每个任务需要匹配其技能水平要求的员工去执行。为了能够尽快的完成项目，项目管理者需要将每个任务分配给执行者，并统筹各个任务的执行顺序。

针对MSRCPSP，在以往的文献中进行了广泛的研究[4]。Shahnazari-Shahrezaei[5]根据Shabdiz丙烯工厂的规则和规定制定多技能人力调度方案，将雇主目标和雇员偏好同时考虑在内，在工厂生产线上考虑两种专业化工作的技能类型，同时每种专业化技能分为初级、标准、高级三种水平。高级和初级水平分别是最高和最低技能水平。Valls[7]针对公司服务中心管理中的多技能劳工项目调度问题进行了研究。劳工的技能水平显示他们执行多种类型任务的效率和速度。研究获得满足最大工期和工人约束时间表同时将任务的紧急程度和工人的任务量均衡分配纳入考虑的调度方案。MSRCPSP同样属于NP难问题，准确式算法仅仅能解决小规模问题，对于大规模问题，启发式和元启发式算法被广泛的应用于在可接受的计算时间内获得近最优解。Skowronski[6]提出了两种禁忌算法用于求解问题。Santos 和 Tereso[8]同时优化完工时间和考虑资源成本的总成本，对于迟到进行处罚，对于提前进行奖励，并提出了广度优先搜索算法和一束搜索算法。Li and Womer[9]提出了一种混合Benders分解算法，这种算法将混合整数线性规划和约束规划进行优势互补，减少总的人力成本。Myszkowski等人[10]针对MSRCPSP问题提出了一些同时考虑完工时间和资源总成本的启发式方法。后来，Myszkowski等人[11]组合启发式方法和蚁群优化算法（ACO），提出了一种混合蚁群算法。Zheng[12]提出一种教学算法，采用教学阶段、学生相互学习及增强阶段三阶段搜索问题的解。

在现代公司中人力资源是一个重要的商业灵活性储备[13 Attia]，作为针对竞争对手的一个重要依据。同时人员的给定技能的能力随着经验的积累不断发展的参数。当员工被持续的分配到各种工作岗位，他们中的每一个人都会持续的练习相关的技能。通过考虑员工经验和学习，管理人员可以利用经验带来的能力提升，提高项目进展及效率带进一步增长[21]。员工技能的重要性驱动着研究者考虑员工技能的提高变化。在MSRCPSP问题的基础上考虑资源技能水平可变，将此作为对MSRCPSP问题的扩展。技能水平提高会带来两方面的影响：（1）资源技能水平的高低会影响任务的实际执行时间，因此任务的工期不再固定，当分配给任务的资源技能水平高时，任务工期短，反之，任务工期长。（2）当资源的技能水平从某一等级升至更高的一个等级时，该资源可以分配给原先不能执行的任务，对于同一任务，其执行资源选择增多，因此技能水平变化会影响资源-任务的分配结果。同时针对多技能资源受限项目调度过程中的技能成长相关的文献研究很少[15]，因此有必要针对此作深入的研究。

经验对于服务或者生产时间的影响在文献中通常被称为“学习”。目前有大量的文献提出了描述随经验增长服务和生产时间缩短的数学表达式，我们通常将这种表达式称为学习曲线。学习曲线的各种研究及应用可以在下列文献[22][23]看到。关于学习曲线的应用的一个总结可以在文献[24]。学习曲线由Wright [25]在基于观察飞机装配成本随重复性的操作执行而递减提出。本文使用由De Jong[7]基于最广泛使用的Wright[25]提出的Power 模型[22]的改进模型即De Jong模型。De Jong模型相较于原始的Power 模型解释了经验观察的现象，限制了生产或者服务时间可以减少的程度，特别是无论累积的经验多少，时间都不会变为零[22]，同时服务时间随着经验的积累而衰减的速度在减缓。考虑适用于人力资源调度，因此基于De Jong模型我们提出了资源技能水平变化模型，用以描述资源技能水平与经验的关系。

工期和成本是被以往的项目调度和员工分配中采用的最多的目标。对于多技能项目调度问题，很多研究[13 Heimerl][4]将成本作为主要的调度目标，其它的一些将项目工期作为首要的调度目标[4] [12][16]，还有的在调度建立过程中同时将工期和成本作为调度目标[11][17][18]。同时考虑上面两种目标是现实中项目调度的实际需求，本文考虑MSRCPSP在调度过程中员工资源技能的提高，同时将调度的工期和成本作为调度目标获取调度方案。双目标MSRCPSP技能可变拥有以下三种特性：多技能特性使得搜索空间大于单技能分配调度问题。技能成长特点使得技能效率值计算更复杂，调度条件和约束是动态变化的。双目标特点使得问题变化为双目标优化调度问题。这些特点都增加了调度的困难。

由于双目标优化问题带有冲突的目标，通常没有最优解，通过求解获得一系列非支配解通常是一个有效的方法。计算求解最优帕累托前沿非支配解通常是困难的，因此，通常使用元启发式算法求解目标值空间中的紧帕累托前沿解。本文，我们首先采用非支配排序遗传算法II[19]（NSGA\_II）用于求解非线性双目标组合优化问题，并对比在考虑技能可变及技能不变下MSRCPSP求解结果的对比，由于提出的问题模型的特点，针对技能可变下的MSRCPSP的求解我们修改了算法的设计，必须考虑由于技能动态提高下约束改变下的资源分配。变领域搜索算法是基于邻域的系统变化，在下降阶段找到局部最小值，并在扰动阶段，逃离相应的山谷[20]。基于变邻下降搜索的局部寻优能力及逃离局部优的搜索特点。其次，本文提出一种混合遗传变邻算法GAVN，组合遗传算法和变邻下降搜索算法，在考虑技能可变的MRCPSP下将我们提出的算法与经典的NSGA\_II进行对比，验证我们提出的算法的性能。

本文剩余章节组织如下：第二节主要描述MSRCPSP问题，第三节主要介绍如何建立资源技能变化模型，以及在模型中如何描述资源技能水平随经验的变化，第四节描述求解资源技能可变的MSRCPSP问题的方法。包括经典的NSGA\_II和本文提出的混合遗传变邻GAVN方法，第五节主要介绍计算实验，考虑技能学习与原有技能不变下调度结果的对比以验证提出的技能可变模型及将NSGA\_II与GAVN算法在资源技能可变下进行对比验证提出的算法，并在此基础上得出相关的结论。第六节展示本文的一些结论和未来的一些工作。

**2 问题描述**

在本节中我们首先描述了经典的MSRCPSP问题，在此基础上进行扩展，考虑资源技能水平会随执行时间增长而提高，并描述了资源技能水平在每轮任务执行中的该如何进行更新。

**2.1 MSRCPSP**

对于MSRCPSP问题的描述如下，一个项目由J个任务组成，表示任务j(j=1,2,…,J),所需要的执行工期为,并且任务执行时不可中断。在任务之间存在紧前关系，任务的紧前任务集为，每个任务必须要在其所有的紧前任务执行完成之后才能开始执行。每个任务都需要被一个达到一定标准水平的技能执行，执行任务(*j*∈*J)*需要的技能种类为，技能水平需要达到。项目中包含K种类型的资源和N种技能,每种资源都具有一些技能，这些技能都会用技能水平表示其熟练程度。对于资源，拥有*h*种技能，组成技能集和对应水平集，当且仅当资源r被指定需要的技能，其熟练程度水平不低于标准水平时，它才能执行相应的任务。无论何种技能，当任一资源分配给任一任务时其薪资都是一个固定的值。MSRCPSP的目标制定一个满足紧前关系和技能水平要求等约束条件的调度方案，使得项目的完工时间和总成本最小。

为了进一步说明资源能力的含义，我们用表格1表示一个包含4种资源和3种技能的项目如表格所示，任务需要技能水平为2的技能。任务具有技能和，其各自的技能水平分别为3和2。由于不包含技能，所以不能被分配给。此外，也不能分配给任务，因为所具有的技能的技能水平低于2。

表1 任务资源能力匹配关系

√ Capable

X Incapable

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
| () |  |
| () |  |
| () |  |
| () |  |

|  |  |  |  |
| --- | --- | --- | --- |
| √ | x | √ | √ |
| x | √ | x | x |
| x | x | x | √ |
| √ | √ | √ | x |

**2.2资源技能水平可变的扩展**

在资源技能水平可变的扩展中，任务的工期不再是一个固定的值，而是一个与执行任务的资源相关的值，的技能r水平值越大，工期越小。我们根据De Jong学习能力模型建立了资源技能水平随执行时间变化的模型——资源技能水平变化模型，详细的论述在第三节展现。在每次任务执行结束时都需要根据技能水平变化模型更新资源的对应的技能水平值，每种资源都会有各自的学习能力水平，所以不同资源技能水平增长的速度会有所不同。同时，在此后的资源-任务分配和任务工期计算中需要使用资源最新的技能水平值。

**3 资源技能水平变化模型**

本节中，我们首先介绍了De Jong学习能力模型，De Jong学习能力模型描述当重复进行周期性的工作，或者生产相同产品时，每个周期所需的时间在很长一个时间段内会逐渐下降。资源在使用同一技能进行重复工作时，每个周期所需的时间与其技能水平值相关，技能水平值越大，每个周期所需时间越短，因此可以将该模型转变为资源技能水平值与时间的关系模型，进而提出了我们的资源技能水平变化模型。

**3.1 De Jong学习能力模型**

近年来，下列有关每个连续工作周期时长随执行轮次增加逐渐下降的代数表达式——De Jong学习能力模型已经多次被采用。

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

此处，代表资源在批量生产工作中，第一个连续工作周期所需的时间；代表第s个连续工作周期所需的时间；代表不可变因子，，在第无穷个连续工作周期；代表下降指数。

**3.2资源技能水平变化模型**

假设在第个连续工作周期，资源的技能r水平值与工作周期时长成反比，则有:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

由De Jong学习能力模型变形可以得到：

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

结合上两式即得到资源技能水平随工作轮次增加逐渐增加的表达式：

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

此处，代表资源的技能在第一个连续工作周期开始时的技能水平；代表资源的技能在第个连续工作周期开始时的技能水平；代表第个资源的学习能力值。代表资源的技能的执行轮次。

假设在第一个连续工作周期开始时的技能水平为1，即最低技能，则有：

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

对于在项目调度开始时拥有技能水平的资源对应的连续工作周期轮次为：

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

由上进而建立资源技能水平变化模型：

|  |  |  |
| --- | --- | --- |
|  |  | (7) |
|  |  | (8) |
|  |  | (9) |
|  |  | (10) |

:资源在任务开始时刻时，技能r的技能水平

：资源在任务开始时刻t时，技能r的历史执行时长轮次

：任务的标准工期

：任务在时刻被资源执行的时长换算为轮次数

：资源在任务结束时的技能水平

：技能r一个连续工作周期即一个轮次所需的时间

在任务开始时刻，利用公式(3)计算资源的技能的历史执行轮次，代表任务开始之前，资源的技能总共执行过的轮次；并利用公式(4)将标准工期转化为执行轮次；在任务结束时刻，利用公式(5)更新资源的技能水平为；利用公式(6)计算任务实际的执行工期。

假设在第一个连续工作周期的执行时间=100，不可变因子=0.25，下降指数m=0.3，在此参数情况下对应的执行时间随执行轮次s变化及随轮次s增长技能水平成长图分别如下图所示。(刻度问题)



**3.4考虑资源技能可变的调度模型**

结合技能可变模型，本文将考虑技能可变的*MSRCPSP*建立成一个混合整数规划模型，具体的双目标优化模型如下：

|  |  |  |
| --- | --- | --- |
|  |  | (11) |
|  |  | (12) |

Subject to:

|  |  |  |
| --- | --- | --- |
|  |  | (13) |
|  |  | (14) |
|  |  | (15) |
|  |  | (16) |
|  |  | (17) |
|  |  | (18) |
|  |  | (19) |
|  |  | (20) |
|  |  | (21) |
|  |  | (22) |
|  |  | (23) |
|  |  | (24) |
|  |  | (25) |

公式(11)和式(12)是待求解的目标函数，分别表示项目工期及项目成本；式(13)表示任务开始时间与结束时间之间的关系；式(14)保证所有任务需要满足紧前约束关系；式(15)确保每个任务的技能要求得到满足；式(16)(17)在任务的开始执行时刻计算分配的资源的对应技能历史执行轮次和执行任务后的执行轮次增量；式(18)在任务完成时刻更新执行资源对应技能的水平值；式(19)计算任务的实际执行时间；式(20)保证了任何资源在同一时刻只能执行一个任务；式(21)表示所有任务都将被某种资源执行；式(22)保证每个资源的单位薪资为正数以及最少拥有一种技能；式(23)确保每个任务的开始时间以及工期都是正数。

**4 算法求解**

对于MSRCPSP这类多目标优化问题，本文首先选用经典的非支配排序遗传算法II（NSGA\_II算法）进行求解， NSGA\_II算法被广泛的用于求解多目标优化问题，具有低计算复杂度（代表目标个数，代表种群大小），而且能够在各个解之间的传递信息，这些特性使得该算法拥有更高的计算效率[19]。

首先使用NSGA\_II算法分别求解资源可学习技能可变的情况下与资源技能不可变的情况。同时针对问题的特点本文将遗传算法与变邻搜索算法进行融合，提出了混合遗传变邻算法。正如之前所述，由于获取所有的非支配解的计算和方法上的困难，我们使用提出的算法获得帕累托前沿解的近似解集。

**4.1编码和解码方案**

每一个解都是MSRCPSP问题的一个个体，因为MSRCPSP问题由两个子问题组成，资源分配和确定任务执行顺序，所以每个个体分别用一个二元列表进行表示，任务列表TL见公式和资源列表RL见公式（26）[11]。

|  |  |  |
| --- | --- | --- |
|  |  | (26) |

其中TL决定任务的执行顺序，RL决定如何分配资源给指定任务；具体的讲表示被安排在第i个执行的任务，表示分配给任务的资源。

解码的过程实际上是一个已经被分配到指定资源的任务，根据其在TL中的优先顺序和RL中资源的分配关系，来确定该任务的开始执行时间，资源技能可变的MSRCPSP问题的解码要考虑以下四个因素：

1. 每个任务开始时间的设定：满足紧前关系的前提下晚于所需资源的释放时间，即所有紧前任务的完成时间与所需资源的释放时间取最大值
2. 工期可变因素：任务的工期是任务的标准工期与任务开始时分配给该任务的资源技能水平的比值
3. 技能水平更新：资源在对应任务结束时更新技能水平
4. 前向调度：资源在两个任务执行之间可能会存在空闲时间，下一个任务可以在满足紧前关系和工期短于空闲时长的前提下插入空闲时间执行；并且此时的资源技能水平是空闲时间开始之前的技能水平,执行完该任务后，资源技能水平不更新。

我们利用一个包含4个任务和4种资源的例子解释整个解码过程,假设四种资源的学习能力因子分别为m1=0.515,m2=0.321,m3=0.152,m4=0.515.

表2 任务资源能力匹配关系

√ Capable

X Incapable

|  |  |  |  |
| --- | --- | --- | --- |
| T1 | T2 | T3 | T4 |
|  |  |  |  |

|  |  |
| --- | --- |
| R1(m1=0.515) |  |
| R2(m2=0.321) |  |
| R3(m3=0.152) |  |
| R4(m4=0.515) |  |

|  |  |  |  |
| --- | --- | --- | --- |
| √ | x | √ | √ |
| x | √ | x | x |
| x | x | x | √ |
| √ | √ | √ | x |

假设现在的任务执行列表TL={1，2，3，4},资源分配列表RL = {1,2,1,1}:

图1 任务及资源分配关系图

如图1所示，按照任务执行列表顺序，首先执行T1，根据公式(4)计算出=15,根据公式(1),(2),(3)更新资源R1的技能k2的技能水平;接着执行T2，计算出,2.03;执行T3，计算出;执行T4,计算出，并且此时由于T2与T3之间存在紧前关系，R2资源在t=15处有一个时长为15的空闲时间，此时将T4安排在空闲处执行，为避免技能水平更新对后续任务执行造成影响，R1在执行T4时的技能水平为t=15时的技能水平，并且T4执行之后不更新R1的技能水平值。最终的可行调度方案如图2所示。

图2 可行调度方案

**4.2种群初始化**

在接下去的算法求解中，我们需要构造一个规模为的种群，种群由代表MSRCPSP问题解的个体构成，而每个个体都由一个二元列表表示。因此，种群初始化的过程就是生成个二元列表。

为了尽可能搜索到解空间的所有解，本文采用了随机生成的搜索策略。具体地说，就是通过从可用任务集合中随机选取一个任务，直至所有任务都被选用的方式来生成任务列表。对于可用集合中的所有任务，其紧前任务都必须已经被选用过。遍历任务列表，随机选取符合要求的资源生成资源列表。这样生成的任务列表和资源列表不会违背约束条件，也能保证个体的合理性。

考虑资源技能可变的MSRCPSP问题中，资源在每次任务执行结束时，所使用的技能水平值都会发生变化。这一变化会影响资源的分配结果，当资源的技能从原来的等级跃迁至更高一级时，该资源会成为更多任务的可用资源。此外技能的变化也会影响任务的实际执行工期。因此，本文利用资源技能使用时间表来记录各个时刻资源的占用情况和技能水平值。可在进行前向调度时遍历技能使用时间表，判断是否可进行前向调度。资源技能使用时间表的一个案例可见表3。

表3 资源技能使用时间表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q3([start,end]:level) | [0,0]:3.0 | [82,100]:3.013 |  |  |
| Q14([start,end]:level) | [0,0]:1.0 | [1,22]:1.588 | [42,63]:1.872 | [64,81]:2.025 |
| Q13([start,end]:level) | [0,0]:2.0 |  |  |  |
| Q8([start,end]:level) | [0.0]:2.0 |  |  |  |

其中start, end, level分别代表该阶段开始，结束时间和技能水平值。假设现在有一个任务j，需要技能类型Q14,要求技能水平为1，标准工期为20；则在遍历资源R的Q14技能使用时间表时发现可以在时刻22-42之间执行该任务，当在22时刻执行时满足紧前关系就可以将该任务安排在22时刻执行。表中Q14技能在81时刻，技能水平从1.872上升到2.025，这种变化称为技能跃迁。

确定种群初始化策略后，可以通过生成的个体求解目标函数值，求解的过程展示在图3中。

**确定任务执行顺序**

**更新资源技能水平**

N

Y

**计算任务开始和结束时间**

**前序可调度**

**执行任务，选择可用资源**

**存在技能跃迁**

**更新任务可用资源列表**

**计算目标函数值**

N

Y

图3 目标函数值求解算法流程图

* 1. **NSGA\_II算法**

**4.3.1遗传算子**

**1)交叉算子**

本文使用的遗传算法采用单点交叉算子，使用P代表种群数量。执行交叉操作时，种群中的两个个体被随机的选择作为父亲，同时生成一个随机的交叉位置，将两个个体的染色体中的任务序列分成两个部分，通过交换两个个体的相同部分，生成新的个体的任务序列。两个父亲个体交叉时需要满足交叉概率的约束。由于在考虑技能可变的情况下资源技能状态是与前面执行的任务相关，故当更改了任务序列后不能对资源序列进行交叉。

**2)变异算子**

本文算法采用单点变异算子。对于每个个体，当满足变异概率时即对其执行变异操作。对于一个个体的染色体中的任务序列，我们随机的生成一个变异位置，当满足变异条件时，对于指定位置的任务如果与下一个执行的任务不违反紧前关系约束，则交换相邻两个位置的任务。同样，由于个体的任务序列的更改，先前的任务执行下的资源的技能状态不再同新任务序列的资源状态相同，故对于新生成的任务序列，在计算个体目标函数值时重新进行资源的分配。

4.3.2**非劣解选择**

对于多目标优化问题，采用的常为非支配排序和拥挤度距离的方法用来选择下一代种群，具体可见[19]。当新生成的种群和原来的种群混合时，对混合种群进行非支配排序，分别得到种群个体的非支配排序等级，拥有相同非支配等级的个体位于解空间的同一帕累托前沿，彼此之间相互非支配，然后对同一等级即同一帕累托前沿的解计算个体的拥挤度。我们选择出拥有更小的非支配排序等级的个体进入下一代，如果两个个体排序等级相同，则我们选择拥有更大拥挤度距离的个体。

**4.3.3 算法流程**

首先，随机生成一个父种群。根据非支配关系对种群进行排序，每一个解都有各自的非支配等级。对种群交叉，变异产生新的后代种群。通过对比后代种群和当前种群混合进行非劣解选择找到最优解集，产生新的种群，进行迭代直到满足迭代终止条件。

**4.4混合遗传变邻搜索算法**

**4.4.1变邻搜索算法**

变邻搜索算法（VNS）是由Mladenovic and Hansen [26]提出的用于解决组合和全局优化问题的一种元启发式算法。VNS的基本思想是在下降阶段搜索局部最优解的过程中系统的改变邻解结构同时在扰动阶段离开对应的山谷[27]以获得全局最优解。

变邻搜索算法的提出基于以下的观察[20]：

(1)相较于一个邻域结构的局部最优值不一定是另一个邻域结构的局部最优值。

(2)全局最优值是相对于所有领域结构的局部最优。

(3)经验数据表明，对于许多问题，所有或者大多数的局部最优都相互接近。

第一个属性通过使用更多的复杂的移动来搜索所有使用的邻域结构的局部最优值。第二个属性建议如果寻找到的局部最优解质量差，则使用多个邻域结构搜索，第三个属性建议增加对现有解的邻近搜索。

随着变邻搜索算法的发展，被大量的用于解决很多NP-hard问题，并成为新的当前最优的新的启发式算法[20]。因此本文拟应用变邻搜索算法于求解多技能资源受限项目调度问题以求获取更优质量的解。

本文设计的双目标变邻搜索算法工作如下。多个设计的邻解结构首先列成一个表，紧接着按照预先定义的顺序一个接一个的执行搜索。使用作为一系列的邻解结构的算子及执行搜索的循序。从一个给定的当前解X开始，取出当前解的任务序列T，双目标变邻搜索程序迭代的执行算子定义的邻结构，，按照算子预先建立的顺序生成新的任务序列，根据定义的资源搜索策略为新获取的任务序列中的每一个任务分配资源，生成资源序列R，新的任务序列和资源序列组成新的解X’，根据问题模型求解出新解的成本和工期值，当前搜索邻解搜索算子搜索一次，由于是双目标解质量的比较，所以如果获得的新解X’支配原来的解X，则将新解替换为当前搜索解，使用当前邻解搜索算子结构进行下一次搜索，否则，执行邻解算子更新，在新的邻解结构下继续搜索，直到最后的搜索算子搜索完毕，其算法程序见图4。

**Algorithm:** Sequential B-VNS

Function B-VNS (X,)

Set k1

Repeat Exploration of neighborhood

Task sequence Tsolution X

random one of

X’resource search(,)+

Compue each objective of X

IF dominate X

X

Else

kk+1//change neighborhood structure

Util k=

END

图4 双目标变邻搜索算法

**4.4.2邻解结构**

邻解结构定义当前解可以进行的修改类型，因此，不同的邻域结构提供了探索解空间的不同方式。合适的邻解结构可以进行对解空间更好的探索和开发。组成解的两个序列属性被纳入考虑用来定义五种邻解结构，因此一个更大的解空间可以被搜索，可以寻找质量更好的解的机会增强。一个解的两个属性分别是任务序列和分配的资源序列，通过对这两个属性的更改来生成下一个解。定义的邻解结构及对应的移动在下面陈述。

1. 第一个可以随机邻交换

给定一个解X，对于组成解的任务序列T，从序列中随机的选择一个位置a，然后将该位置对应的任务同它的相邻位置的任务进行交换，如果选中的位置同其相邻的位置具有紧前紧后关系，则重新进行选择直到两个位置没有紧前关系可以进行交换为止。

（2）概率相邻交换

对于当前解的任务序列T，对序列{}从头往后遍历，如果对于位置a，如果满足给定的交换概率且该位置的任务与相邻位置的任务不存在紧前关系，则进行交换，否则，进行下一个位置的判断。

（3）第一个可以随机前插

对于当前解的任务序列T，从序列中随机的选择两个位置a和b，a<b，位于之前，判断任务序列{，，…，}与是否存在紧前关系，如果该序列中每一个位置对应的任务不是的紧前任务，则将任务插入到任务之前，否则，重新进行位置和的选择。

（4）第一个可以随机后插

对于当前解的任务序列T，从序列中随机的选择两个位置和，a<b，位于之前，判断任务序列{，…，，}与是否存在紧后关系，如果该序列中每一个位置对应的任务不是位置对应的任务的紧后任务，则将任务插入到任务之后，否则，重新进行位置和的选择。

（5）第一个可以随机交换

对于当前解的任务序列T，从序列中随机的选择两个位置和，a<b，位于之前，判断任务序列{，，…，}与是否存在紧前关系，任务序列{，…，，}与是否存在紧后关系，如果序列中每一个位置对应的任务不是的紧前任务，且如果序列中每一个位置对应的任务不是的紧后任务，则将任务与任务交换，否则，重新进行位置和的选择。

**4.4.3资源搜索策略**

本文研究在技能可变下的多技能资源受限项目调度问题，考虑工期和成本最小化作为优化目标，因此，本文设计资源搜索策略以围绕当前两个目标而进行搜索以快速寻找更优质量的解。在资源搜索前当前解已经有一个任务序列， 需要为每一个任务从候选资源集中选择一个合适的资源以执行，资源搜索时使用一个平衡策略因子，主要用于在极端解搜索和双目标平衡搜索间做平衡，当进行搜索时生成的随机数小于给定的平衡因子时即进行双目标平衡搜索，即在为当前解个体任务搜索可执行资源时同时考虑项目成本和项目工期，而当随机因子大于给定的平衡策略因子时则对当期解执行极端搜索，最早完成时间资源策略作工期极端最小化搜索，最低成本执行资源策略作成本极端最小化搜索。通过平衡因子的引入使最终求得的帕累托前沿能够覆盖更广的解空间，具有更强的多样性。由于资源在执行任务时拥有的任务执行的技能得到学习，技能等级会变化，故资源执行完成后对技能进行更新，同时更新后续任务的可执行候选资源集，资源搜索算法的伪代码见图5。

Procedure resource search(T, )

p=rand(0,1)

If(p<)

For i=1 to n

Select resource r can process with the earlist completion time or least cost

Update resource r skill

End For

Else

p1=rand(0,1)

If(p1<0.5)

For i=1 to n

Select resource r can process with the earlist completion time

Update resource r skill

End For

Else

For i=1 to n

Select resource r can process with least cost

Update resource r skill

End For

End

图5 资源搜索策略的伪代码

**4.4.4非劣解集及选择策略**

在所有解中位于帕累托分级的第一帕累托前沿的解不被其它的个体支配，是问题模型的非劣解。非支配排序NS技术广泛应用于多目标优化问题的选择过程。NS能有效的保证解的质量及解的均匀性分布，能够覆盖整个帕累托前沿解。在提出的算法中使用非支配排序[19]、支配点集及超体积[29]用于选择下一代种群。

非支配排序将种群分级为n个集合。每个子集代表一个帕累托前沿集合同时带有一个分层等级系数，在每个子集集合中的解相互非支配。

支配点数量是支配解X的解集合P(t)中的解个数，定义如下：

|  |  |  |
| --- | --- | --- |
|  |  | (27) |

表示解Y支配解X，即解Y的两个目标值均优于解X。

超体积测量是用于比较多目标优化算法的结果最常用的度量之一[28]。根据两个目标，取得问题的非支配排序的帕累托第一前沿解，使用第一个目标值对这些解进行升序排序，由于这些点的都相互非支配，额外的我们获得根据第二个目标值排序的序列。

给定排序好的前沿解，根据Beume N[29]提出的计算超体积公式，解超体积计算如下()：

|  |  |  |
| --- | --- | --- |
|  |  | (28) |

从上一代种群中解中我们选择具有更小的排序等级的个体，如果两个个体拥有相同的排序，我们选择具有较小支配解数量的个体，如果两个个体的支配解数量相同，我们选择具有更高的超体积的个体。

**4.4.5混合遗传变邻框架**

在本部分呈现提出的遗传算法混合变邻搜索。混合遗传算法和变邻搜索算法能够有效的提高算法局部最优的收敛速度，另一方面，在每一次迭代中会增加计算时间。同时在每一个个体的生成过程中，资源搜索策略由于是动态的更新资源技能及任务的候选集资源和资源搜索的指导性会额外增加计算时间。混合遗传变邻搜索算法GAVN的框架见图6。

Algorithm 混合遗传变邻算法

初始化参数种群数量n 交叉概率proc 平衡因子prob 邻解算子数k 迭代次数g

根据串行调度生成策略生成任务序列和根据随机规则分配资源生成个体 建立初始种群P

Repeat

计算生成的个体适应值 对种群进行非支配排序

种群P根据交叉概率prob交叉生成种群P’

对种群P’执行变邻搜索B-VND生成种群P’’

混合种群P和P’’执行非支配排序及计算个体支配点个数及超体积

非劣解选择下一代种群

Util g=gmax

End

图6 混合遗传变邻搜索算法

**5计算实验**

为了验证提出的技能变化模型在实际应用中的效果同时验证本文提出的混遗传变邻搜索算法的作用，本文使用Myszkowski[30]等人提出的智能多目标项目调度环境（iMOPSE）基准案例库进行数值试验，可以登录<http://imopse.ii.pwr.wroc.pl/> 使用案例库。案例库由36个项目案例组成，其中包括18组100个任务的案例和18组200个任务的案例。案例的计算复杂度由任务数量，资源数量，技能种类数量，紧前关系数量控制，所有案例根据这些数量进行命名。在考虑资源技能水平可变下需对案例进行一些初始化设置，首先将案例库中的技能水平赋值，设置为最低水平为1;由于学习能力差异，将学习能力分为高中低三档，因而对指数分别赋值0.515，0.321，0.152，利用round-robin原则给资源分配不同的学习能力，假如现有6种资源，则；工期标准化，将所有工期换算成用最低技能水平操作时间表示的标准工期，如案例库中任务需要技能，所需工期为，则标准工期;设置初始值[7]。

NSGA\_II和GAVN使用java编程，所有的测试运行处理器为Intel(R) Core(TM) i5 [CPU@3.30GHz](mailto:CPU@3.30GHz)/ 8GB RAM。NSGA\_II在技能可变和技能不可变下及 GAVN在技能可变下分别运行10次。

**5.1性能评价指标**

为了评价多技能资源受限项目调度在考虑资源技能可变对实际调度的影响，同时考虑本文提出的混合遗传变邻算法的效果，总共有三种情形，分别是NSGAII求解MSRCPSP、NSGAII及GAVN求解技能可变的MSRCPSP。为了评价不同情形下获得的帕累托前沿解集的质量，引入超体积（非支配空间尺寸）H和非支配解覆盖率指标[31]。

超体积这个度量指标是由Zitzler[31]设计用来量化MOEAs的收敛行为。超体积用于计算目标空间内一系列非支配帕累托前沿最优解覆盖的解空间面积/体积。在本文需将二维目标空间中获得的非支配解目标函数值进行反转归一后再进行超体积求解。

目标函数值反转及归一公式如下：

|  |  |  |
| --- | --- | --- |
|  |  | (29) |

式中为对应目标函数值的最大值。在本文中的目标函数空间由项目成本和项目工期组成，项目工期最大值为所有任务的工期的总和。该情况发生于所有的任务一个接一个串行执行。项目成本最大值为所有的任务都分配最昂贵的资源。具有小项目工期和更低成本的帕累托前沿解拥有更大的在反转函数值空间的超体积。

覆盖率C指标用来测量两个解集之间的支配关系。对于两个算法A和B获得的两个解集和，计算如下：

|  |  |  |
| --- | --- | --- |
|  |  | (30) |

表示中的解被支配的百分率。明显的一个更大的或者一个更小的表示算法A的性能比B好。

**5.2参数设置**

在遗传算法二代中，当用于求解原始的多技能资源受限项目调度（不考虑资源技能水平可变）时，有四个参数：初始种群大小N、交叉概率、任务序列变异概率、资源分配序列变异概率。当考虑资源技能水平可变时有三个参数需要考虑：初始种群大小N、交叉概率、任务序列变异概率。本文提出的混合遗传变邻算法有三个参数需要考虑：初始种群大小N、交叉概率、资源搜索平衡因子。为了研究这些参数对NSGA\_II和GAVN算法性能的影响，本文采用正交试验设计[32]方法。每个参数的水平设置及分析见表4。本文采用正交表。对于每种参数组合，NSGA\_II（考虑技能不可变和技能可变）和GAVN（技能可变）运行独立10次的，获得对应的帕累托前沿解集。使用上文提到的指标超体积的10次运行的平均值用来衡量获得的帕累托解集的质量。根据正交试验结果获得参数的最优水平值，每个因子水平的趋势见图7。

表4 三种情形下的参数设置

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 多目标算法 |  | 参数水平 | | | Delta | Rank |
| 算法参数 | 1 | 2 | 3 |
| *NSGA-II(不考虑技能水平可变)* | 初始种群大小：N | 100 | 200 | 300 | 0.00637 | 2 |
| 交叉概率： | 0.7 | 0.8 | 0.9 | 0.00087 | 4 |
| 任务序列变异概率： | 0.1 | 0.15 | 0.2 | 0.00130 | 3 |
| 资源分配序列变异概率： | 0.1 | 0.2 | 0.3 | 0.03380 | 1 |
| *NSGA-II(不考虑技能水平可变)* | 初始种群大小：N | 100 | 200 | 300 | 0.00387 | 1 |
| 交叉概率: | 0.7 | 0.8 | 0.9 | 0.00133 | 2 |
| 任务序列变异概率 | 0.1 | 0.15 | 0.2 | 0.00083 | 3 |
| *GAVN(考虑技能水平可变)* | 初始种群大小：N | 100 | 200 | 300 | 0.00173 | 1 |
| 交叉概率： | 0.7 | 0.8 | 0.9 | 0.00057 | 3 |
| 资源搜索平衡因子 | 0.6 | 0.7 | 0.8 | 0.00110 | 2 |



a) NSGA\_II不考虑资源技能水平变化



b)NSGA\_II考虑资源技能水平变化



c)GAVN考虑资源技能水平变化

图7 因子水平趋势

从表4和图7可以看出在考虑资源技能水平变化时 ，NSGA\_II算法中种群数目是最重要的参数，紧跟着是交叉概率，任务序列变异概率是影响最不显著的参数。GAVN算法中 最重要的参数是种群数目，紧接着是资源搜索平衡因子，影响最不显著的是交叉概率。在两种算法中影响最显著的都是种群数量，因为一个更大的种群可以有更有效的探索。在GAVN中，因为本文研究目标是项目工期和成本，资源搜索平衡因子直接影响调度目标，影响帕累托前沿的边缘伸展，所以其影响仅次于种群数目。

根据上述分析，建议的参数组合为NSGA\_II（不考虑资源技能水平变化）为N=300,=0.9,,=0.1。NSGA\_II（考虑资源技能水平变化）时N=300,=0.8,.GAVN(考虑资源技能水平变化)N=300,=0.8,=0.6。

**5.3考虑资源技能水平可变的影响**

为了验证本文提出的资源技能变化模型对实际调度中的作用，分别使用遗传算法二代NSGA\_II用于求解考虑资源技能水平可变和资源技能水平不可变情形下的多技能资源受限项目调度问题，并进行对比。设置算法迭代次数为300，对每个案例分别运行10次，计算10次的平均超体积和最大超体积及最大超体积下考虑资源技能可变和不可变求解出的帕累托前沿解集覆盖率C，两种情形下的求解结果列在表5中。

表5 NSGA\_II求解考虑资源技能水平可变和不可变情形对比

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 考虑资源技能水平可变(1) | | 不考虑资源技能水平可变(2) | |  |  |
| 案例 |  |  |  |  |  |  |
| 100\_5\_20\_9\_D3 | 0.4437 | 0.4476 | 0.3223 | 0.3244 | 1.00 | 0.00 |
| 100\_5\_22\_15 | 0.3657 | 0.3693 | 0.2242 | 0.2250 | 1.00 | 0.00 |
| 100\_5\_46\_15 | 0.2676 | 0.2786 | 0.1207 | 0.1210 | 1.00 | 0.00 |
| 100\_5\_48\_9 | 0.2691 | 0.2746 | 0.1281 | 0.1299 | 1.00 | 0.00 |
| 100\_5\_64\_9 | 0.4771 | 0.4838 | 0.3957 | 0.3972 | 1.00 | 0.00 |
| 100\_5\_64\_15 | 0.3497 | 0.3565 | 0.2247 | 0.2254 | 1.00 | 0.00 |
| 100\_10\_26\_15 | 0.5051 | 0.5220 | 0.4605 | 0.4694 | 1.00 | 0.00 |
| 100\_10\_27\_9\_D2 | 0.4729 | 0.4857 | 0.4542 | 0.4601 | 1.00 | 0.00 |
| 100\_10\_47\_9 | 0.5034 | 0.5128 | 0.3746 | 0.3866 | 1.00 | 0.00 |
| 100\_10\_48\_15 | 0.5296 | 0.5386 | 0.4484 | 0.4537 | 1.00 | 0.00 |
| 100\_10\_64\_9 | 0.5717 | 0.5837 | 0.5291 | 0.5374 | 1.00 | 0.00 |
| 100\_10\_65\_15 | 0.4551 | 0.4672 | 0.3548 | 0.3615 | 0.96 | 0.00 |
| 100\_20\_22\_15 | 0.6187 | 0.6259 | 0.6201 | 0.6300 | 1.00 | 0.00 |
| 100\_20\_23\_9\_D1 | 0.5501 | 0.5560 | 0.5408 | 0.5441 | 1.00 | 0.00 |
| 100\_20\_46\_15 | 0.5467 | 0.5567 | 0.5402 | 0.5465 | 1.00 | 0.00 |
| 100\_20\_47\_9 | 0.5899 | 0.5986 | 0.6164 | 0.6233 | 1.00 | 0.00 |
| 100\_20\_65\_9 | 0.5660 | 0.5736 | 0.5764 | 0.5878 | 1.00 | 0.00 |
| 100\_20\_65\_15 | 0.5858 | 0.5946 | 0.6012 | 0.6067 | 0.75 | 0.00 |
| 200\_10\_50\_9 | 0.5644 | 0.5718 | 0.4974 | 0.5027 | 0.86 | 0.00 |
| 200\_10\_50\_15 | 0.6220 | 0.6266 | 0.5845 | 0.5924 | 1.00 | 0.00 |
| 200\_10\_84\_9 | 0.6026 | 0.6059 | 0.5410 | 0.5463 | 0.98 | 0.00 |
| 200\_10\_85\_15 | 0.4505 | 0.4593 | 0.3624 | 0.3730 | 0.73 | 0.00 |
| 200\_10\_128\_15 | 0.4416 | 0.4503 | 0.3904 | 0.3985 | 1.00 | 0.00 |
| 200\_10\_135\_9\_D6 | 0.4140 | 0.4219 | 0.3237 | 0.3305 | 1.00 | 0.00 |
| 200\_20\_54\_15 | 0.5101 | 0.5248 | 0.4559 | 0.4649 | 0.89 | 0.00 |
| 200\_20\_55\_9 | 0.6492 | 0.6536 | 0.6193 | 0.6337 | 0.65 | 0.00 |
| 200\_20\_97\_9 | 0.5413 | 0.5454 | 0.5490 | 0.5561 | 0.87 | 0.00 |
| 200\_20\_97\_15 | 0.5183 | 0.5249 | 0.5021 | 0.5105 | 1.00 | 0.00 |
| 200\_20\_145\_15 | 0.5172 | 0.5248 | 0.4492 | 0.4574 | 1.00 | 0.00 |
| 200\_20\_150\_9\_D5 | 0.4893 | 0.4947 | 0.5125 | 0.5175 | 0.00 | 0.46 |
| 200\_40\_45\_9 | 0.6156 | 0.6218 | 0.6222 | 0.6300 | 0.84 | 0.00 |
| 200\_40\_45\_15 | 0.5995 | 0.6056 | 0.5879 | 0.5961 | 0.93 | 0.00 |
| 200\_40\_90\_9 | 0.5887 | 0.5934 | 0.5992 | 0.6080 | 0.91 | 0.00 |
| 200\_40\_91\_15 | 0.6017 | 0.6099 | 0.6019 | 0.6101 | 0.97 | 0.00 |
| 200\_40\_130\_9\_D4 | 0.4468 | 0.4546 | 0.5063 | 0.5131 | 0.00 | 1.00 |
| 200\_40\_133\_15 | 0.5601 | 0.5721 | 0.5756 | 0.5838 | 0.87 | 0.00 |

从表5可以得知，在所有36个案例中26/36个案例中在考虑资源技能水平可变的情况下平均超体积和最大超体积均大于不考虑资源技能水平可变的情形，同时在最大超体积下获得的帕累托前沿解的覆盖率，在36个案例中，有34/36个案例考虑资源技能水平可变对不考虑资源技能水平可变覆盖率均大于不考虑资源技能水平可变对考虑资源技能水平可变覆盖率，这说明在94%的案例中不考虑资源技能水平可变获得的帕累托前沿解被考虑资源技能水平可变获得的帕累托前沿解支配，考虑资源技能水平可变时能获得更前沿的帕累托解，即能获得更低成本和工期的调度方案。

两种不同的情形下在典型的案例下100\_20\_46\_15(100个任务)和200-40\_91\_15(200个任务)获得的帕累托前沿解展示在图8中。



图8 NSGA获得的帕累托前沿解对比

从上述结果我们可以知道，当考虑资源技能可学习可变时，获得的帕累托解在更前沿，支配没有考虑资源技能学习的情况，这主要是当考虑资源技能可学习时，资源的技能随着执行任务次数的增加而提高，在接下来的任务安排中任务的候选资源数量变多，分配资源具有更多的选择，建立调度的方案增多。

**5.4算法比较**

因为没有其它的算法对多技能资源受限项目调度环境下考虑资源技能学习研究的工作，快速非支配排序遗传算法二代被广泛应用于求解多目标优化问题，同时在求解实际工程问题时被证明是有效的[17]。因此，采用NSGA\_II作为本文提出的混合遗传变邻算法GAVN的对比。为了公平的比较，NSGA\_II和GAVN都采用同样的初始化操作机制，计算在相同的运算时间内算法获得的帕累托前沿解的质量，。

设置两种算法的运行时间为30s,每个案例运行10次，计算10次运算获得的平均超体积和最大超体积及在最大超体积下NSGA\_II和GAVN算法获得的帕累托前沿解集的覆盖率。相关的结果列在表6中。

表6 NSGA\_II和GAVN算法帕累托前沿解结果

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | GAVN | | NSGA | |  |  |
| 案例 |  |  |  |  | C(GAVN,NSGA) | C(NSGA,GAVN) |
| 100\_5\_20\_9\_D3 | 0.5304 | 0.5338 | 0.4468 | 0.4524 | 1 | 0 |
| 100\_5\_22\_15 | 0.4208 | 0.4233 | 0.3669 | 0.3715 | 1 | 0 |
| 100\_5\_46\_15 | 0.3264 | 0.3283 | 0.2677 | 0.2699 | 1 | 0 |
| 100\_5\_48\_9 | 0.3380 | 0.3393 | 0.2700 | 0.2742 | 1 | 0 |
| 100\_5\_64\_9 | 0.5907 | 0.5919 | 0.4799 | 0.4856 | 1 | 0 |
| 100\_5\_64\_15 | 0.4435 | 0.4449 | 0.3544 | 0.3608 | 1 | 0 |
| 100\_10\_26\_15 | 0.6465 | 0.6474 | 0.4998 | 0.5073 | 1 | 0 |
| 100\_10\_27\_9\_D2 | 0.5910 | 0.5933 | 0.4739 | 0.4825 | 1 | 0 |
| 100\_10\_47\_9 | 0.6335 | 0.6343 | 0.5053 | 0.5085 | 1 | 0 |
| 100\_10\_48\_15 | 0.6450 | 0.6463 | 0.5291 | 0.5358 | 1 | 0 |
| 100\_10\_64\_9 | 0.7087 | 0.7096 | 0.5704 | 0.5743 | 1 | 0 |
| 100\_10\_65\_15 | 0.5656 | 0.5665 | 0.4603 | 0.4657 | 1 | 0 |
| 100\_20\_22\_15 | 0.7530 | 0.7538 | 0.6207 | 0.6263 | 1 | 0 |
| 100\_20\_23\_9\_D1 | 0.7319 | 0.7324 | 0.5545 | 0.5637 | 1 | 0 |
| 100\_20\_46\_15 | 0.7178 | 0.7200 | 0.5463 | 0.5584 | 1 | 0 |
| 100\_20\_47\_9 | 0.7748 | 0.7761 | 0.5895 | 0.5989 | 1 | 0 |
| 100\_20\_65\_9 | 0.7310 | 0.7346 | 0.5611 | 0.5705 | 1 | 0 |
| 100\_20\_65\_15 | 0.7167 | 0.7193 | 0.5874 | 0.6015 | 1 | 0 |
| 200\_10\_50\_9 | 0.7697 | 0.7707 | 0.5660 | 0.5733 | 1 | 0 |
| 200\_10\_50\_15 | 0.7760 | 0.7769 | 0.6205 | 0.6310 | 1 | 0 |
| 200\_10\_84\_9 | 0.7598 | 0.7608 | 0.6004 | 0.6042 | 1 | 0 |
| 200\_10\_85\_15 | 0.6030 | 0.6038 | 0.4476 | 0.4541 | 1 | 0 |
| 200\_10\_128\_15 | 0.5501 | 0.5511 | 0.4409 | 0.4522 | 1 | 0 |
| 200\_10\_135\_9\_D6 | 0.5621 | 0.5632 | 0.4130 | 0.4233 | 1 | 0 |
| 200\_20\_54\_15 | 0.6724 | 0.6736 | 0.5078 | 0.5129 | 1 | 0 |
| 200\_20\_55\_9 | 0.8584 | 0.8596 | 0.6480 | 0.6577 | 1 | 0 |
| 200\_20\_97\_9 | 0.8022 | 0.8025 | 0.5399 | 0.5539 | 1 | 0 |
| 200\_20\_97\_15 | 0.7010 | 0.7018 | 0.5137 | 0.5268 | 1 | 0 |
| 200\_20\_145\_15 | 0.7146 | 0.7156 | 0.5112 | 0.5146 | 1 | 0 |
| 200\_20\_150\_9\_D5 | 0.6410 | 0.6419 | 0.4860 | 0.4920 | 1 | 0 |
| 200\_40\_45\_9 | 0.8602 | 0.8612 | 0.6142 | 0.6195 | 1 | 0 |
| 200\_40\_45\_15 | 0.8382 | 0.8387 | 0.5974 | 0.6019 | 1 | 0 |
| 200\_40\_90\_9 | 0.8627 | 0.8631 | 0.5872 | 0.5928 | 1 | 0 |
| 200\_40\_91\_15 | 0.8232 | 0.8236 | 0.5991 | 0.6038 | 1 | 0 |
| 200\_40\_130\_9\_D4 | 0.6864 | 0.6871 | 0.4447 | 0.4611 | 1 | 0 |
| 200\_40\_133\_15 | 0.8187 | 0.8194 | 0.5594 | 0.5699 | 1 | 0 |

从表6中可以得知GAVN相比NSGA在所有的案例中获得了更大的和值，同时对所有的案例中C(GAVN,NSGA)的值为1,C(NSGA,GAVN)的值为0，这说明，NSGA获得的帕累托前沿解被GAVN获得的帕累托前沿支配，GAVN获得了更好的帕累托前沿解。同时，两种算法在典型的案例100\_20\_46\_15(100个任务)和200-40\_91\_15(200个任务)获得的帕累托前沿解展示在图9中。



图9 获得的帕累托前沿解对比

从上述结果我们可以知道相比NSGA，GAVN获得了更前沿的帕累托解。GAVN的优势主要归功于以下几个方面：1）解的搜索具有更多的邻结构，在解空间具有更大的探索性2）邻结构系统的切换能够使得当前解逃离局部最优3）资源搜索策略是针对问题的具体的算子，能够获得更好的解。

**6结论**

在本文，针对多技能资源受限项目调度，我们提出了一个技能学习变化模型。我们的目标是最小化项目的成本和工期。针对多技能资源受限项目调度的技能可变，我们建立了一个混合整数非线性双目标规划模型，首先采用NSGA\_II用于求解模型，在调度过程中，资源技能水平随着任务的执行而提高。我们采用了*Myszkowski*等[30]提出的*iMOPSE*标准案例库验证我们提出的模型。使用NSGA\_II求解获得帕累托前沿非支配解集，同不考虑资源技能可变的情形下获得的帕累托前沿非支配解作了对比，证明了当考虑资源技能学习的情况下，能够获得更前沿的帕累托解，即建立的调度方案能够获得更少的工期和更低的成本，这说明在实际中考虑资源技能学习可变能够充分利用资源获得更好的调度方案。同时本文提出了一个混合遗传变邻搜索算法GAVN，采用群体策略，基于任务序列设计了五种邻解搜索算子，能够有效的进行解空间的开发，同时系统的进行邻结构的转换能够有效的逃离局部最优，基于问题具体的特点设计了资源搜索算子，使得能够寻找到高质量的资源分配序列，进行的数值测试证明了提出的GAVN在解决多技能资源受限项目调度资源技能可变时的有效性。

未来的工作可能针对实际应用中资源情况的复杂性，对资源技能学习变化模型作进一步的改进，同时本文考虑的多技能资源受限项目调度问题是属于组合优化问题，针对对本文提出的GAVN应用于其他相关领域的实际问题也是未来的关注点。

[1] Blazewicz J, Lenstra JK, Rinnooy Kan AHG (1983) Scheduling subject to resource constraints: classification and complexity[J]. Discret Appl Math 5:11–2

[2] Skowronski M E, Myszkowski P B, Adamski M, Kwiatek P. Tabu Search approach for Multi - Skill Resource - Constrained Project Scheduling Problem[Z]. 2013.

[3] Al-Anzi F S, Al-Zame K, Allahverdi A. Weighted Multi-Skill resources project scheduling[J]. Journal of Software Engineering and Applications. 2010, 03(12): 1125-1130.

[4] Myszkowski P B, Olech Ł P, Laszczyk M, Skowroński M E. Hybrid differential evolution and greedy algorithm (DEGR) for solving Multi-Skill Resource-Constrained project scheduling problem[J]. Applied Soft Computing. 2018, 62: 1-14.

[5]Shahnazari-Shahrezaei P, Tavakkoli-Moghaddam R, Kazemipoor H. Solving a new fuzzy multi-objective model for a multi-skilled manpower scheduling problem by particle swarm optimization and elite tabu search[J]. The International Journal of Advanced Manufacturing Technology. 2013, 64(9-12): 1517-1540.

[6] Skowronski M E, Myszkowski P B, Adamski M, Kwiatek P. Tabu Search approach for Multi - Skill Resource - Constrained Project Scheduling Problem[Z]. 2013.

[7] DE Jong J R. The effects of increasing skill on cycle time and its consequences for time standards[J]. 1957, 1(1): 51-60.

【7if wrong】Valls V, Pérez Á, Quintanilla S. Skilled workforce scheduling in Service Centres[J]. European Journal of Operational Research. 2009, 193(3): 791-804.

[8] M.A. Santos, and A.P. Tereso. On the Multi-mode, Multi-skill Resource Constrained Project Scheduling Problem-A Software Application[J]. Soft Computing in Industrial Applications. Springer Berlin Heidelberg, 239-248,2011

[9] H. Li, K. Womer, Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm. Journal of Scheduling, 12: 281-298, 2009.

[10] P.B. Myszkowski, M.E. Skowronski, and L.P. Olech, Novel heuristic solutions for Multi-Skill Resource-Constrained Project Scheduling Problem[C]// in Proceedings of the 2013 Federated Conference on Computer Science and Information Systems, 2013: 159-166.

[11] Myszkowski P B, Skowroński M E, Olech Ł P, Oślizło K. Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem[J]. Soft Computing. 2015, 19(12): 3599-3619.

[12] Zheng H, Wang L, Zheng X. Teaching–learning-based optimization algorithm for multi-skill resource constrained project scheduling problem[J]. Soft Computing. 2017, 21(6): 1537-1548.

[13] Attia E, Duquenne P, Le-Lann J. Considering skills evolutions in multi-skilled workforce allocation with flexible working hours[J]. 2014, 52(15): 4548-4573.

[14]Heimerl C, Kolisch R. Scheduling and staffing multiple projects with a multi-skilled workforce[J]. OR Spectrum. 2010, 32(2): 343-368.

[15] Chen R, Liang C, Gu D, Leung J Y. A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution[J]. 2017, 55(21): 6207-6234

[16] Myszkowski P B, Laszczyk M, Kalinowski D. Co-evolutionary algorithm solving multi – skill resource – constrained project scheduling problem[Z]. 2017: 11

[17] Wang L, Zheng X. A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem[J]. Swarm and Evolutionary Computation. 2018, 38: 54-63.

[18] Maghsoudlou H, Afshar-Nadjafi B, Niaki S T A. A multi-objective invasive weeds optimization algorithm for solving multi-skill multi-mode resource constrained project scheduling problem[J]. Computers & Chemical Engineering. 2016, 88: 157-169.

[19]Kalyanmoy Deb, Associate Member, IEEE, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm:NSGA-II[J], IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 6, NO. 2, APRIL 2002

[20]Hansen P, Mladenović N, Todosijević R, Hanafi S. Variable neighborhood search: Basics and variants[J]. EURO Journal On Computational Optimization. 2017, 5(3): 423-454.

[21] Chen X, Thomas B W, Hewitt M. The technician routing problem with experience-based service times[J]. Omega. 2016, 61: 49-61.

[22] Xi Chen, Barrett W. Thomas, Mike Hewitt, The technician routing problem with experience-based service times[J], Omega 61 (2016) 49–61

[23] Attia E, Duquenne P, Le-Lann J. Considering skills evolutions in multi-skilled workforce allocation with flexible working hours[J]. 2014, 52(15): 4548-4573.

[24] Anzanello M J, Fogliatto F S. Learning curve models and applications: Literature review and research directions[J]. International Journal of Industrial Ergonomics. 2011, 41(5): 573-583.

[25] Factors affecting the cost of airplanes[J].

[26] Mladenovi´ c N, Hansen P (1997) Variable neighborhood search. Comput Oper Res 24(11):1097–1100

[27] Hansen P., Mladenović N., Brimberg J., Pérez J.A.M. (2010) Variable Neighborhood Search. In: Gendreau M., Potvin JY. (eds) Handbook of Metaheuristics. International Series in Operations Research & Management Science, vol 146. Springer, Boston, MA

[28] Emmerich M., Beume N., Naujoks B. (2005) An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In: Coello Coello C.A., Hernández Aguirre A., Zitzler E. (eds) Evolutionary Multi-Criterion Optimization. EMO 2005. Lecture Notes in Computer Science, vol 3410. Springer, Berlin, Heidelberg

[29] Beume N, Naujoks B, Emmerich M. SMS-EMOA: Multiobjective selection based on dominated hypervolume[J]. European Journal of Operational Research. 2007, 181(3): 1653-1669

[30] Myszkowski P B, Laszczyk M, Nikulin I, Skowroński M. IMOPSE: A library for bicriteria optimization in Multi-Skill Resource-Constrained Project Scheduling Problem[J]. Soft Computing. 2018.

[31]E. Zitzler, Evolutionary Algorithms for Multiobjective Optimization: Methods

and Applications 63 (1999)

[32] Montgomery DC .Design and analysis of experiments[M].2013