**带有柔性项目结构的随机资源均衡问题**

# SRLP-PS

## 问题描述

本文研究的SRLP-PS可以描述如下:

一个项目可以用一个节点式有向网络表示，其中为节点集合，表示项目中的活动，。活动0和是虚活动，分别代表项目的开始和结束。为有向弧的集合，表示活动之间具有完成-开始的零延迟优先关系，即每个活动必须在其所有紧前活动完成之后，才能开始执行。每个非虚活动的工期是一个随机变量，并且遵循一个已知的概率分布。虚活动的工期为0。因为活动的工期是随机变量，所以活动的开始时间也是随机变量，虚终止活动的开始时间对应项目的完成时间。项目的完成时间不能超过给定的项目截止日期。项目执行时需要种可更新资源，每个时段资源的供应量用表示。非虚活动执行时，在单位时段对第种可更新资源的需求量为。虚活动在执行时不消耗任何资源。资源在时段的使用量用表示，即在时段所有正在执行的活动对资源的总需求量，即，其中表示在时段正在执行的活动集合，即。每个活动一次性执行，不允许中断。

为了对柔性项目结构进行建模，将活动集合进一步分为三个互斥的子集 (Kellenbrink & Helber, 2015)：必须执行活动集、可选活动集和依赖活动集，,。必须执行活动集中的活动是总是需要执行的活动。除了必须执行活动，其他活动的执行与否是不固定的。因此，一个柔性项目中只需要执行所有活动的子集，且存在多种项目结构，不同的项目结构由不同的活动集合和相应的优先关系构成。

可选活动集可能包含多个互斥的子集，即，。在项目执行时，并非每个可选活动子集都要执行。每个子集都对应了一个选择。每个选择可以被一个活动触发，如果选择被触发，那么一个可选活动将执行，中的其他活动都不执行。活动可以是必须执行活动、可选活动或依赖活动。需要注意的是， 在Kellenbrink & Helber研究的柔性项目结构中只能是必须执行活动或可选活动。选择是拓扑排序的，被选中的活动执行可能会触发下一个选择(。

依赖活动集中活动的执行与否同可选活动有关。一个可选活动的执行可能会导致多个依赖活动执行，即如果可选活动执行，那么所有依赖的活动（这些活动的集合记为）必须都执行。此外，可选活动的依赖活动只能是它后面的活动，即可选活动的编号小于其依赖活动集中活动的编号。需要注意的是，依赖活动和可选活动之间可能不存在优先关系。

考虑到柔性项目结构和不确定的活动工期，SRLP-PS的目标是在满足优先关系、资源可用量和项目截止日期的约束下，确定需要执行的活动并调度它们，尽可能的最小化资源使用量的期望波动。

## 随机规划模型

首先引入0-1决策变量表示活动是否执行，如果活动执行，则为1，否则为0。决策变量表示活动的开始时间。接下来，建立SRLP-PS的随机规划模型M0：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (M0) | Minimize | |  | (1) |
|  | Subject to: | |  |  |
|  |  | |  | (2) |
|  |  |  | | (3) |
|  |  |  | | (4) |
|  |  | ; ; | | (5) |
|  |  |  | | (6) |
|  |  |  | | (7) |
|  |  | ; | | (8) |
|  |  | ; | | (9) |
|  |  | ; | | (10) |
|  |  |  | | (11) |

目标函数(1)衡量了任意两个相邻时间段内资源使用量的变动情况，是资源的权重，表示资源的单位惩罚成本。式(2)确保了虚开始活动0的开始时间为0。式(3)保证了所有必须执行活动都执行。式(4)确保了如果选择被执行的活动触发，那么在选择的可选活动集中必须有一个活动执行。式(5)表示可选活动执行则其相应的依赖活动也都要执行。在不确定的环境中，包含随机变量的约束可能不完全满足，采用机会约束更符合实际情况。式(6)-(8)都是机会约束，其中表示事件发生的概率，、和为置信水平，它们反映了决策者对风险的偏好。具体而言，式(6)表示活动之间的优先关系，它要求活动的完成时间早于其紧后活动的开始时间的概率大于等于，其中是一个任意大的正常数，只有活动和都执行时，该约束条件才成立，即活动和之间的优先关系才有效。式(7)要求项目按时完成的概率不应小于。式(8)确保在整个项目执行中资源的使用量满足其可用量的概率不小于。式(9)用于计算整个项目的资源使用量。式(10)和(11)给出了决策变量的取值范围。

## 模型线性化

本节将随机规划模型M0线性化为等价的整数规划模型，以便用商业求解器进行求解。用确定的情景约束代替随机机会约束，具体而言，用一组有限的情景集合表示不确定的活动工期，项目中所有活动的随机工期的一次采样（实现）构成一个情景，用来表示，情景出现的概率为。

由于M0的目标函数是非线性的，因此本文进一步将目标函数线性化。引入辅助变量表示在情景中，两个相邻时段的资源使用量与之间的偏差，可表示为：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | (12) |
|  |  |  | (13) |

则可将目标函数(1)线性化为

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | (14) |

其中，)为项目经理可接受的项目完成时间上限，即在某些情景下，项目的完成时间可以超过项目的截止日期。

在个情景下，如果每个情景中项目的执行活动都是相同的，且调度这些活动能使期望目标函数值最小，则意味着在随机环境中执行并调度这些活动可以使项目的资源使用量尽可能平稳，这也就说明活动是否执行不依赖情景。由于活动在不同情景下的工期不同，活动的开始时间也会不同。因此，本文引入二元变量表示在情景中，活动在时刻是否开始。如果是，则，否则为0。对于任意情景，约束条件(2)可以替代为：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | (15) |

需要注意的是，变量与相关联，如果活动不执行（），则在任意情景任意时刻下，。本文用式(16)连接变量和，以保证在所有情景下，执行的活动都是一样的：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | ; | (16) |

其中和分别表示在情景中，活动的最早开始时间和最晚开始时间。和分别表示在情景下活动的最早开始时间和最晚开始时间，其中最早开始时间是在只考虑必须执行的活动下利用正向计算得到的。对于最晚开始时间，以项目最晚完成时间为基础，在所有活动都执行的前提下利用逆向计算得到。

接下来，用确定性情景约束代替随机机会约束。首先，定义一个很大的正数，二元变量、和分别表示在情景中，执行活动的开始时间是否晚于其紧前活动的开始时间、项目是否在截止日期内完成和项目的资源使用量是否超过资源的可用量。通过定义以上变量，可将机会约束(6)-(8)改写为：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | |  | (17) |
|  |  |  | | (18) |
|  |  |  | | (19) |
|  |  |  | | (20) |
|  |  | ; | | (21) |
|  |  |  | | (22) |

式(20)保证了执行活动的开始时间满足优先关系（），或不满足（）。式(21)确保了在个情景下满足优先关系的约束符合期望概率。式(22)保证项目在截止日期内完成（），或超过截止日期（）。式(23)确保在所有情景下项目在截止日期内完成的概率大于等于。式(24)表示资源的使用量不超过给定量（），或超过（）。式(25)确保在所有情景下满足资源可用量的概率不小于。

式(9)可以改写为：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | ; | (23) |

根据上述计算公式，可建立整数规划模型M1：

|  |  |  |  |
| --- | --- | --- | --- |
| (M1) | Minimize = (14) |  |  |
|  | Subject to: |  |  |
|  | (3)-(5), (15)-(23) |  |  |
|  | = {0, 1} |  | (24) |
|  | , , = {0, 1} |  | (25) |

# DE

差分进化算法（differential evolution algorithm，DE）是一种基于种群的求解连续优化问题的随机搜索技术，在科学和工程领域得到了广泛的应用。由于其简单有效的特点，在项目调度领域，已有文献将DE应用于求解资源均衡问题【】。本文提出了一个改进的差分进化算法，设计了专门的编码、解码、初始化和局部搜索等操作。除此，为了增强算法搜索能力，本文采用了两种变异策略，且相关参数会随着算法的迭代而动态变化【更新】。一个局部搜索算法被用来进一步提高算法的求解质量。

本文的改进差分进化算法流程如算法1所示。首先，将进度计划编码为个体。然后，生成包含POP个个体的初始种群。由于DE求解的是连续优化问题，而SRLP-PS的解是离散的，因此，本文将初始种群中的离散个体转化为连续个体（向量），再对向量执行变异和交叉操作生成试验向量。接下来，将试验向量转化为新个体，然后对新个体进行解码，计算每个个体对应的适应值，评估个体的优劣。如果新个体的适应值小于原个体的适应值，则用新个体替换原个体。为进一步改进新个体的质量，本文选择B个最好的个体进行局部搜索。最后，判断算法是否达到循环的终止条件，如果没有，则重复上述DE操作，直到满足终止条件。

|  |  |
| --- | --- |
| **算法1：DE流程** | |
| 1 | 输入：项目网络结构，活动工期数据 |
| 2 | 输出：调度策略 |
| 3 | 初始化参数，，； |
| 4 | ; // 表示当前迭代次数 |
| 5 | 生成初始种群 |
| 6 | /\*DE迭代\*/ |
| 7 | While time< end\_time |
| 8 | ; |
| 9 | 更新参数, , , |
| 10 | /\*变异\*/ |
| 12 | If |
| 13 | 变异策略1 |
| 14 | Else |
| 15 | 变异策略2 |
| 16 | End |
| 17 | 交叉操作 |
| 18 | 生成新个体 |
| 19 | 评价个体 |
| 20 | 选择 |
| 21 | 局部搜索 |
| 22 | End whlie |

## 编码和解码

SRLP-PS的解（调度策略）可以编码为一个个体，一个个体由三部分构成：（1）执行列表向量，用于表示活动是否执行，其中的每个元素，如果活动执行，则，否则。（2）活动列表，它是所有活动的一个排列，且活动之间满足优先关系，其中虚开始（虚终止）活动始终在的第一个（最后一个）位置。（3）解码列表，用于表示安排活动开始时间的两种方式，其中的每个元素。如果，则在满足优先关系和资源约束的前提下尽早开始活动；如果，考虑其他可能的开始时间（已经调度活动的完成时间），推迟活动。

为了将个体转化为进度计划，本文设计了一种新的进度生成机制对个体进行解码（算法1）。在算法1中，根据中活动的顺序依次调度活动，对于不执行的活动，本文将其开始时间设置为很大的正数INF，这就意味着该活动不会执行。对于执行活动，根据编码列表为其安排开始时间。具体而言，对于执行活动，如果，则在满足优先关系和资源约束的前提下，尽早开始调度活动（第6~19行），以尽可能的满足截止日期约束。是活动所有紧前活动的集合。是一个二元变量，用于指示在时段资源的使用量是否超过了资源的可用量。为正在调度的活动集合，如果一个活动被安排了开始时间则添加到。如果，首先考察已经调度的活动的完成时间，在满足优先关系的前提下，选择最小的完成时间（第26行），然后进一步该时间考察是否满足资源约束（第11~20行），最终得到满足资源约束的开始时间。与第一种分配方式不同，第二种考虑的是在满足优先关系和资源约束的条件下推迟活动。对于资源均衡问题来说，适当的推迟活动，可能更利于资源均衡，避免资源使用的高峰。为保证项目尽早完成，本文选择了最小的完成时间。接下来，需要更新正在调度的活动集合，具体而言，将中完成时间小于或等于的活动移除，再将活动添加到正在执行的活动集。在安排了活动的开始时间后，需要更新资源的使用量（第38~40）。当所有的执行活动都安排开始时间后，解码结束。【或者是遍历，判断是否满足资源约束？但是这样可能会超过截止日期】

调度策略：项目进入执行阶段后，针对活动工期的不确定性，利用调度策略确定每一个决策时间点上可以执行的活动，使得资源使用量随时间的变化趋于稳定

在截止日期充裕的时候，考虑将活动右移，以均衡资源

|  |  |
| --- | --- |
| **算法2：解码** | |
| 1 | 输入：，活动的实现工期 |
| 2 | 输出：实现的进度计划 |
| 3 |  |
| 4 | For each and , ; |
| 5 | For to |
| 6 | ; |
| 7 | If |
| 8 | ; |
| 9 | If |
| 10 |  |
| 11 | ; |
| 12 | While |
| 13 | ; |
| 14 | For , |
| 15 | If |
| 16 | ; |
| 17 | break; |
| 18 | End if |
| 19 | End for |
| 20 | If , then ; |
| 21 | End while |
| 22 | , ; |
| 24 | Else |
| 25 | /\*考虑已经调度的活动的完成时间\*/ |
| 26 | ; |
| 28 | 重复Line11~Line20，得到满足资源约束的; |
| 32 | /\*更新正在执行的活动集合\*/ |
| 33 |  |
| 34 | ;; |
| 35 | End if |
| 36 | /\*更新资源使用量\*/ |
| 37 | For , |
| 38 | ; |
| 39 | End for |
| 40 | Else |
| 41 | ; |
| 42 | End if |
| 43 | End for |

## 初始种群

（1）生成：随机生成，分为两个步骤：

**Step 1**: 令必须执行活动集中的活动都执行，即。

**Step 2**: 判断选择是否被触发。根据选择的拓扑排序依次判断每个选择是否被触发，由此决定可选活动和依赖活动的状态。如果一个选择被触发，即，那么随机选中一个可选活动执行，即，中的其他活动都不执行。如果执行的可选活动有依赖活动集，则其所有依赖活动都要执行，即。当一个选择被考虑后，进一步判断下一个选择是否被触发，进而确定有关的可选活动和依赖活动的执行状态，直到所有的选择都被考虑。

（2）生成：根据生成的，进一步生成，以确保生成的中所有执行活动的排列满足优先关系。具体来说，首先，根据，将未执行的活动随机放置在上。然后，依次对上未放置活动的每一个位置，确定一组能够放在该位置的合格活动，将这组活动记为集合。中的每个活动都是执行活动，并且其所有紧前活动都已经放置在中。接下来，需要从中选择一个活动放置上，本文使用最小优先值规则，优先值越小的活动，越优先调度，即该活动在中的位置更靠前。本文利用关键路径法，根据活动的平均工期，计算每个执行活动的最晚开始时间。对于中的活动，其优先值为。

（3）生成：对于每一个活动，生成一个随机数，如果，则对应的；否则。

## 变异

DE处理的是基于连续的解，而RLP-PS是解空间是离散的。为了将DE应用于SRLP-PS，需要在个体和DE个体（向量）之间找到合适的映射，将离散解转化为连续解，具体而言：

对于和，它们都是二元向量，因此本文在每一个元素的基础上加上一个随机数。

对于，本文将中活动的调度优先顺序转化为活动的优先值，即中的第一个活动的优先值为，第二个活动的优先值为，…，最后一个活动的优先值为，这些优先值构成一个优先值向量。

在将离散个体转化为连续个体之后，就可以对个体执行DE操作。在DE进化的不同阶段，可能需要不同的策略和不同的参数设置，以使算法达到最佳性能。除此，对于复杂的优化问题来说，没有一种变异策略被证明对所有问题都是最好的，使用不同的突变策略可能会比单一的突变策略更好。因此本文使用了两种变异策略，以实现局部搜索和全局搜索的权衡。在每一次迭代中，DE使用变异策略对当前种群中的每个连续个体（目标向量）产生一个变异向量。

第一种策略是DE/rand/1的扩展，如式(48)所示：

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

其中索引，和是从中随机生成的三个互斥整数。虽然随机选择三个向量进行变异具有较强的探索能力，有利于保持种群的多样性，但算法的收敛速度较慢。为了既保持探索性，又加快收敛速度，Kaelo and Ali (2006)提出了该策略的改进，即为三个向量中最好的，和是剩下的两个个体。是第次迭代的缩放因子。

第二种策略是DE/target-to-best/1的变体，如式(49)所示，该策略有利于提高算法的收敛速度。通常在较好的个体向量上更容易、更快地找到全局最优解。

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

针对每个突变点在的方向上探索每个周围区域。

对于这两种变异策略，本文使用一个决策参数来指导策略的选择。的取值随着迭代次数的增加而增加，即，的取值范围为[0.1,1]。在算法的早期阶段，主要利用第一种策略进行广泛的搜索。算法后期主要是对某些潜在的最优解区域进行深入搜索。

分别对、和对应的向量执行变异操作。

控制参数动态变化，在开始时选择较高的值来增加全局搜索能力，然后逐渐减小到最小值，保证局部搜索能力（强化）。

## 交叉

在执行完变异操作之后，对于每一对目标向量以及其对应的变异向量执行交叉操作生成一个试验向量。本文的DE采用二项式交叉操作，如式(53)所示。

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

其中。对于活动生成一个随机数。表示第次迭代的交叉概率，它控制从突变向量复制的参数值比例，。对于来说，的取值范围为[0.5, 0.9]，而对应的的取值范围为[0.3, 0.9]。CR越大，局部搜索和收敛速度越来越好，CR值越小，在保持多样性和增强全局搜索能力方面表现越好。

## 生成新个体

由于试验向量无法直接进行解码，因此需要将试验向量转化为新的个体（离散）。对于和对应的试验向量，向量中的每个元素代表活动的优先值。新个体的和的生成与初始种群的生成类似。对于的生成，唯一区别主要在Step2。具体而言，如果选择被触发，根据对应的试验向量，从中选择优先值最大的活动执行，而初始的是随机选择的。对于的生成，每一个未放置活动的位置，选择可选活动集中优先值最大的活动放置。对于，如果试验向量中元素的取值大于等于1，则将其值置为1；否则为0。

## 评估个体

给定一组情景，本文使用算法5对个体进行评估。对于每一个个体，根据算法2获得每个情景对应的进度计划，再计算适应值。如果生成的进度计划超过了规定的截止日期，则需要施加惩罚。本文的目标函数是最小化，因此施加的惩罚为正数。具体而言，如果情景对应的进度计划超过了规定的截止日期，则惩罚的计算公式如下：

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

施加惩罚的目的是尽可能得到满足项目截止日期的调度策略。本文将模型M1的目标函数与因为超过截止日期而产生的惩罚的和作为DE的评价函数。当考虑完所有情景后，计算出期望目标函数值和按时完成的概率。

|  |  |
| --- | --- |
| **算法5：评估个体** | |
| 1 | 输入： , , |
| 2 | 输出：目标函数值，按时完工的概率 |
| 3 | ,, , ; |
| 4 | For to |
| 5 | ; |
| 6 | 使用算法2解码，获得进度计划; |
| 7 | 根据计算资源占用量; |
| 8 | If |
| 9 | ; |
| 10 | ; |
| 11 | Else |
| 12 | ; |
| 13 | End if |
| 14 | End for |
| 15 | ; |
| 16 | ; |

将试验向量对应的新个体与目标向量对应的当前个体进行比较，如果前者比后者的适应值更小，则它将取代后者，进入下一代种群。

需要注意的是，对于最后获得的最优调度策略，本文将重新计算目标函数值，对于超过截止日期的情景不施加惩罚。

## 局部搜索

为进一步提高解的质量，本文从新生成的种群中选择个最优的个体进行局部搜索。由于和是相对应的，变化可能会导致不可行。为避免产生不可行的，本文在不改变的条件下，只对和进行局部搜索，即在固定项目结构的基础上，搜索更好的活动调度顺序和活动的解码方式。具体而言，对于一个个体的，为每个执行活动生成一个随机数，如果执行活动的（为预先规定的概率），则进一步考察下一个执行活动，如果不是的紧前活动，则交换和的位置，构成新的。接下来，对该个体的中的每个元素生成一个随机数，如果，则改变当前元素的值。具体而言，如果当前元素为1，则将其置为0；否则，置为1；如果，则不对元素进行操作。如果局部搜索得到的新个体的适应值比原来的个体小，则用新个体替换原个体。

# 计算实验

## 实验设置

精确算法一般只适用于小规模数据集，因此本文首先用项目调度实例生成器ProGen按照表1的参数设置，基于全因子试验设计生成两个具有不同活动数量的规模较小的数据集（J5和 J10），每一种参数组合下生成5个实例，一共生成个实例。除此，本文还选取PSPLIB的J30数据集，并从每组中选择第六个实例作为基准数据集，一共获得48个实例。由于上述实例都是固定的项目结构，因此，本文按照

表 2的参数设置进一步扩展为具有柔性项目结构的实例，一共获得个实例。

表 1 J5数据集参数设置

|  |  |
| --- | --- |
| 参数 | 取值 |
|  | 7; 12 |
|  | 1.5 |
|  | 0.3; 0.7 |
|  | 0.2; 0.8 |
|  | 4 |

表 2 柔性项目结构参数设置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据集 |  |  |  |  |
| J5 | 1 | 2 | 1 | 1 |
| J10 | 1 | 2 | 1 | 3 |
| J30 | 2; 4 | 2; 3 | 1; 2 | 1; 3 |

本文进一步对基准数据集中的每个项目实例设置两种不同的截止日期：较短截止日期() 和较长截止日期()，其中是对应的平均项目的关键路径长度，即在平均项目中所有活动的工期是随机变量的均值。这样最终得到个实例。

由于上述实例的活动工期都是确定的，因此，本文将这些实例扩展为具有随机活动工期的项目实例。具体而言，本文以数据集中确定的活动工期作为随机活动工期的均值，对于每个实例的每一种情景，从均匀分布U 中随机抽取1个整数，作为该情景下非虚活动的工期。对于每个实例，资源波动的惩罚成本设置为1，表示不同的资源类型同等重要。设置为0.9，这就意味着在项目经理可以接受的范围内某些情景下项目的完成时间可以超过项目的截止日期。和均设置为1，即在所有情景下资源限制和优先关系都满足。【在本文中的取值为DE获得的最优个体仿真过程中最大的截止日期】

## 性能指标

本文将CPLEX求出最优解的实例记为数据集，未求出最优解的实例记为数据集，并使用以下指标评估DE的性能：

（1）平均响应变量（Average response variable，ARD）：对于J5和J10数据集，本文将CPLEX求出最优解的实例记为数据集，未求出最优解的实例记为数据集，的计算公式如下：

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

其中，表示分别表示数据集和中实例的个数。表示实例使用DE计算得到的目标函数值，表示利用CPLEX求出的实例的目标函数值。

【改指标，要么都越小越好，或者越大越好】

（2）平均相对改进（Average relative improvement，ARI）：对于J30数据集，由于CPLEX很难求出可行解，因此本文将随机搜索算法作为上界，衡量DE的性能。给定算法，ARI的就算公式如下：

|  |  |
| --- | --- |
|  | (31) |

其中（）分别表示实例使用RSA(ALG)求解的目标函数值。为J30数据集中实例的个数。

## 收敛性分析

在历史数据的基础上利用概率分布函数对不确定性进行校正，然后生成一组有限的情景集。许多文献使用仿真输出结果的标准差来研究仿真运行时的收敛性

实验数据集，选取J5数据集，截止日期为1.5和1.8，一共40个实例。截止日期设置太大，所以基本情景数得到的结果都一样。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 情景数量 | Max | Min | Gap | Avg | Std | CV | CPU(s) |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

## 参数设置

DE的性能对变异策略和相关的控制参数的选择是敏感的，为了获得最佳的性能，需要耗时的参数调优。DE算法需要多样性，但也需要加强局部搜索的能力。为获得最佳性能，需要进行耗时的参数调优。为减少寻找最优参数的时间，本文采取最普遍的策略，即从参数的可行值范围内开始，随着算法的不断迭代改变算法的参数取值，以实现全局搜索和局部搜索之间的平衡。

DE涉及三个关键控制参数：种群大小、缩放因子和交叉概率，这三个参数显著影响DE的优化性能。因此为了成功地解决特定的优化问题，需要寻找合适的策略以及相关的参数设置。

是第次迭代的缩放因子，每一次迭代中更新，表示的最小（大）取值，的取值范围为[0.3, 1.5]，表示DE最大的迭代次数，表示第次迭代的目标向量。最好的作为基向量（目标向量），剩下两个用来产生差分向量，即此时。第一种策略是保持算法的探索能力。

F决定了差分变异产生差异向量的比例，它对收敛速度和种群多样性有重大影响。

确定参数的值并不是一项简单的任务。DE的控制参数主要有种群大小（）、缩放因子和交叉概率。除此，还有三个关键参数：策略选择参数、精英个体数（）和局部搜索的概率（）。在本文中，参数、和随着算法迭代次数的变化而变化。和的更新公式是一样的。， ，其中表示在第次迭代的参数值，表示DE最大的迭代次数。表示的最小（大）取值，具体而言，的取值范围为[0.3, 1.5]，的取值范围为[0.3, 0.9]。的更新公式为，，的取值范围为[0.1,1]。

对于剩下的参数，本文采用正交试验确定。每个参数都取4个水平，如表 3所示。

利用CPLEX 12.9在2400秒的时间限制下求解基准数据集中每个实例的模型M1，，将这些求出最优解的实例记为并作为试验设计的数据集。

不是一个多目标问题，因此只考虑目标函数值。因为项目的完成时间对于均衡程度是有影响的，完成时间越长，越易于进行资源均衡。

数据集：J5和J10，衡量指标，与CPLEX比？

表 3 参数水平

|  |  |  |  |
| --- | --- | --- | --- |
| 水平 |  |  |  |
| 1 | 50 | 0.1\*( | 0.05 |
| 2 | 100 | 0.2\*( | 0.1 |
| 3 | 150 | 0.3\*( | 0.2 |

| 实验序号 | 参数 | | | ARD |
| --- | --- | --- | --- | --- |
|  |  |  |
| 1 | 1 | 1 | 1 |  |
| 2 | 1 | 2 | 2 |  |
| 3 | 1 | 3 | 3 |  |
| 4 | 2 | 1 | 2 |  |
| 5 | 2 | 2 | 3 |  |
| 6 | 2 | 3 | 1 |  |
| 7 | 3 | 1 | 3 |  |
| 8 | 3 | 2 | 2 |  |
| 9 | 3 | 3 | 1 |  |

表 4 各参数的ARD和重要程度（Pr=100%）

| 水平 |  |  |  |
| --- | --- | --- | --- |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 极差 |  |  |  |
| 排名 |  |  |  |

## 实验结果

【研究不同的停止标准对算法的性能的影响，时间，进度计划？】

只测试了J5的10个实例

|  |  |  |  |
| --- | --- | --- | --- |
| Pr | Avg\_value | Avg\_cpu | 最优解数量 |
| 0.9 | 31.322 | 1444.861 | 2 |
| 0.95 | 31.338 | 1457.577 | 2 |
| 1 | 30.89111 | 825.446 | 5 |

将实验结果分别统计按照DE求出的实例的Pr为100%和Pr<100%

表 6 在上的实验结果（不考虑Pr）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | | DE-10s | | GA-10s | |  |  |
|  |  | value |  | CPU | value |  | value |  | ARD(DE) | ARD(GA) |
| 5 |  | 36.10 | 100.00% | 65.35 | 35.17 | 97.25% | 40.38 | 98.92% | -2.95% | 13.35% |
|  | 36.55 | 100.00% | 215.78 | 35.27 | 99.45% | 40.10 | 98.55% | -2.59% | 12.53% |
| 10 |  | 72.55 | 100.00% | 870.89 | 71.82 | 97.50% | 74.00 | 97.00% | -1.22% | 1.92% |

表 7 CPLEX和DE项目完成的概率都为（Pr=100%）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | DE-10s | |  |
|  |  | value | CPU | value |  | ARD(DE) |
| 5 |  | 37.75 | 24.91 | 38.25 | 100% | 1.32% |
|  | 32 | 250.38 | 32 | 100% | 0% |
| 10 |  | 67 | 926.96 | 69.88 | 100% | 4.5% |

表 8 DE项目完成的概率不为100%

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | | DE-10s | |  |
|  |  | value |  | CPU | value |  | ARD(DE) |
| 5 |  | 32.79 | 100% | 146.23 | 29 | 91.75% | -11.49% |
|  | 57 | 100% | 60.09 | 50 | 97% | -14.22% |
| 10 |  | 78.1 | 100% | 814.82 | 73.76 | 95% | -6.95% |

表 10 在上的实验结果

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | | DE-10s | | GA-10s | |  |  |
|  |  | value |  | CPU | value |  | value |  | ARD(DE) | ARD(GA) |
| 5 |  | 58.02 | 100.00% | 1211.95 | 57.00 | 99.75% | 64.00 | 100.00% | -1.34% | 11.90% |
|  | 56.45 | 100.00% | 1360.80 | 55.56 | 100.00% | 61.28 | 100.00% | -1.25% | 10.03% |
| 10 |  | 101.10 | 100.00% | 1802.42 | 68.29 | 98.80% | 74.36 | 100.00% | -26.08% | -20.72% |
|  | 110.78 | 100.00% | 1801.61 | 70.87 | 99.95% | 75.75 | 100.00% | -28.73% | -24.54% |

表 11 CPLEX和DE项目完成的概率都为100%

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | | DE-10s | |  |
|  |  | value |  | CPU | value |  | ARD(DE) |
| 5 |  | 59.27 | 100.00% | 1215.69 | 58 | 100% | -1.63% |
|  | 56.45 | 100.00% | 1360.80 | 55.56 | 100% | -1.25% |
| 10 |  | 81.95 | 100.00% | 1802.69 | 60.54 | 100% | -25.25% |
|  | 113.01 | 100.00% | 1801.55 | 71.54 | 100% | -29.44% |

表 12 DE项目完成的概率不为100%

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | CPLEX | | | DE-10s | |  |
|  |  | value |  | CPU | value |  | ARD(EDA) |
| 5 |  | 54.24 | 100.00% | 1200 | 54 | 99% | -0.44% |
| 10 |  | 148.89 | 100.00% | 1801.35 | 99.25 | 94% | -29.41% |
|  | 68.44 | 100.00% | 1802.85 | 58 | 99.5% | -15.25% |

表 13 在在上的实验结果

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | DE-30s | | GA-30s | | RSA-30s | |  |  |
|  |  | value |  | value |  | value |  | ARI(DE) | ARI(GA) |
| 30 |  | 251.29 | 97.25% | 276.54 | 96.56% | 294.96 | 94.44% | 14.70% | 6.20% |
|  | 244.43 | 99.41% | 270.79 | 99.50% | 283.85 | 99.59% | 14.06% | 5.19% |

## 灵敏度分析

分析项目结构参数对DE的影响

置信度对于GA的影响？

Fernandez, A. A., Armacost, R. L., & Pet-Edwards, J. J. (1998). Understanding simulation solutions to resource constrained project scheduling problems with stochastic task durations. *Engineering Management Journal, 10*(4), 5-13.

Kaelo, P., & Ali, M. (2006). A numerical study of some modified differential evolution algorithms. *European Journal of Operational Research, 169*(3), 1176-1184.

Kellenbrink, C., & Helber, S. (2015). Scheduling resource-constrained projects with a flexible project structure. *European Journal of Operational Research, 246*(2), 379-391.

Stork, F. (2000). Branch-and-bound algorithms for stochastic resource-constrained project scheduling. *Technical rep*, 702-2000.