求解阈值坏场景集鲁棒一致并行机调度问题的混合布谷鸟算法

**摘要：**本文研究了以最大完工时间作为性能指标的不确定一致并行机调度问题，其中不确定加工时间用离散场景进行表示。本文将一种阈值坏场景集鲁棒优化模型应用到不确定一致并行机调度问题中，通过优化坏场景下性能与给定基准性能间的惩罚值来提升调度解对坏场景下性能下降风险的抵御能力。为了提升对该模型的求解效率，本文提出了一种面向问题的合并场景邻域构造方式，并在布谷鸟算法中加入应用了该邻域构造方式的局部搜索用于求解。本文将该混合算法与其他三种已有的智能优化算法进行仿真对比，最终的实验结果表明本文采用的邻域构造方式和混合算法在对该鲁棒模型的求解上更具优势。

**关键词**：鲁棒一致并行机调度，离散场景，布谷鸟算法，合并场景领域

1. 引言

一致并行机调度问题(identical parallel machine scheduling, IPMS)是一类常见的生产调度问题，涉及多台可同时进行工作且加工能力相同的机器。通过确定各台机器上的加工工件及其对应的加工顺序，来使得某项性能指标达到最优。一致并行机的加工特性符合多种生产场景，广泛地存在于日常的各种生产活动中[1]-[3]。对于一致并行机调度问题的研究通常在理想的确定性环境下进行，各种生产参数如工件的加工时间等都是已知且固定不变的。然而由于存在加工熟练度，机器故障等各种不确定因素的影响，传统的确定性优化模型与实际加工模型间存在一定差距，此时求解确定性生产模型得到的方案难以达到生产预期。如何处理一致并行机调度中的不确定加工时间，建立合适的不确定模型并设计求解不确定模型的高效算法逐渐成为了研究的热门。

场景方法是一种有效的处理不确定参数的方法。同随机分析法[4]和模糊分析法[5]不同，场景方法通过某个区间的连续场景[6]或者有限个离散场景的集合[7]来描述不确定性。各场景的分布相对独立，不需要预先确定具体的概率分布模型或者模糊隶属度函数，适合在不确定因素的发生概率未知或历史信息不足的情况下描述不确定性参数。目前在各类调度问题中有着广泛的应用[8]-[10]。

Li等[11]将基于场景的不确定模型分为两类：基于场景的随机优化模型和基于场景的鲁棒优化模型，两种模型都有各自的决策偏向。基于场景的随机优化模型将整体期望性能作为优化目标[12]，考虑优化整个调度系统在所有可能场景下性能的平均水平。而基于场景的鲁棒优化模型则是以增加系统抵抗不确定性干扰的能力为优化目标。其中比较常见的有min-max模型(最坏场景模型)和min-max regret模型(最大后悔场景模型)。这两种模型通过优化所有场景下的最坏性能或者最大遗憾值来体现决策者的风险厌恶偏向，在调度类问题中应用广泛。如Kouvelis等[10]求解了一个双机器流水车间调度的最大后悔场景模型，同时采用了区间场景和离散场景表示不确定的加工时间。Drwal[13]研究了区间场景加工时间的单机调度问题，优化目标是加权滞后工件数的最大遗憾值。在这两种传统鲁棒优化模型的基础上，Wang[16]等提出了一种阈值坏场景集鲁棒优化模型。该模型不再聚焦单个最坏场景下的性能，而是以一个基准性能为阈值，筛选出性能较差的坏场景，对坏场景施加一定的惩罚，围绕惩罚值制定不同的鲁棒衡量指标进行优化。Wang等[16]将这类阈值坏场景集惩罚模型成功应用于不确定作业车间调度问题中。实验仿真结果表明求解该模型能够获得抵御坏场景下不佳性能的鲁棒解。

目前对于鲁棒一致并行机调度问题研究仍以优化最坏场景性能的传统鲁棒模型为主。如Xu等[6]优化了区间场景下一致并行机的最大完工时间的最大遗憾值，采用了松弛迭代法和模拟退火算法(simulated annealing, SA)来求解不同规模的问题。Wang等[14]研究了工件可外包的一致并行机调度问题，目标函数是外包消耗花费和非外包加工花费的生产总成本。他们通过离散场景和区间场景来描述不确定的加工时间，利用2-近似算法(2-approximation algorithm)分别对区间场景下生产总成本的最大遗憾值和离散场景下的最大生产总成本问题进行了求解。Drwal等[17]对区间加工时间下的一致并行机调度问题进行了研究，优化目标是最大完工时间的最大遗憾值。

然而传统的鲁棒优化模型存在一定的缺陷。最坏场景模型和最大后悔模型过度关注性能表现最差的单一最坏场景，忽视了其他一些场景下较坏性能造成的影响，导致最终解的保守性过高。同时单一场景下的确定一致并行机调度已经是NP-hard问题[6]，而求解这类鲁棒优化模型需要获得每个场景下的最优性能，这无疑大大提升了求解难度。考虑到以上几点，本文尝试在鲁棒一致并行机调度中求解Wang等[16]提出的阈值坏场景集优化模型，通过关注更多坏场景下的性能来降低最终鲁棒解的保守性，这也是对鲁棒一致并行机调度问题的一次全新探索与挑战。

本文求解的一致并行机调度问题的性能指标为调度解的最大完工时间makespan。对于如何求解一致并行机调度的makespan，最初有学者选择使用一些精确算法。如Dell等[20]提出了一种分支定界算法来求解最小化makespan的一致并行机调度问题，Mokotoff[21]用数学规划的形式对一致并行机调度问题进行描述，并设计了一种割平面法来进行求解。随着求解规模的增加，求解难度的增大，越来越多的研究人员开始采用高效的智能优化算法进行求解。如Kashan和Karimi[22]设计了一种离散粒子群算法(discrete particle swarm optimization, DPSO)用于求解目标函数为makespan的一致并行机调度问题，Alharkan等[23]采用变邻域搜索算法(variable neighbourhood search, VNS)对一致并行机调度问题的makespan进行求解，Xu等在求解区间场景下的一致并行机调度问题时，对于较大的问题规模选择使用登山算法(hill climbing algorithm)和模拟退火算法(simulated annealing, SA)两种局部搜索算法进行求解。

随着研究的逐渐深入，为了进一步提高求解效率，研究人员开始采用混合智能优化算法对一致并行机调度问题进行求解。Wang等[25]在优化有工艺约束的并行机问题的makespan时，采用了一种混合了禁忌搜索的遗传算法(tabu-genetic algorithm)，用TS来增加GA的局部搜索能力。Chen等[26]在和声搜索算法(harmony search, HS)的基础上加入了变邻域搜索算法，加强了HS局部搜索的能力。并根据并行机调度问题的特点设计了四种面向问题的邻域构造方式，包括基于关键机器的交换，插入，一对多交换，两两交换等操作。最后采用混合后的算法进行一致并行机调度问题的求解。Guo等[28]设计了三种面向面向问题的邻域结构，并将采用了这些邻域的变邻域搜索算法与布谷鸟算法(cuckoo search, CS)结合的混合算法用于一致并行机调度问题的求解。Lee等[27]在求解一致并行机调度问题时，在遗传算法的基础上加入了局部搜索（GA+LS），通过混合算法的形式来提升求解效率，其中每个个体在每次迭代过程中都会有一定的概率被其邻域解代替

相比单一的智能优化算法，加入了局部搜索的混合智能优化算法在对一致并行机调度的问题的求解中表现得更加出色[25]-[27]。这些混合算法往往结合了群智能并行搜索算法和局部搜索的优点，对解空间有着更强的搜索能力。在已有的群智能并行搜索算法中，布谷鸟算法是一种较为新颖的算法。其模仿布谷鸟的寄生性育雏行为，通过莱维飞行进行每一代种群的更新。大量研究表明莱维飞行比其他随机游走能对解空间进行更有效的探索，因而布谷鸟算法相比其他一些群搜索算法如PSO，GA等表现出了更强的求解能力[29]。而在有着强大全局搜索能力的布谷鸟算法的基础上加入局部搜索，可以对种群中性能表现较差的个体进行更新，将布谷鸟算法引向更好更广的搜索空间，实现不同搜索方式的优势互补。加入了局部搜索的混合布谷鸟算法已经被成功运用于一致并行机调度问题的求解中[28][29]，比未结合局部搜索的单一布谷鸟算法表现出了更优秀的实验结果。

目前已有的用于一致并行机调度的局部搜索，一般基于关键机器进行邻域解的构造[26]-[30]。这是一种面向确定性并行机调度问题特点的邻域构造方式，在一致并行机调度问题中应用广泛，但尚未有面向离散场景下不确定一致并行机调度问题特征的邻域结构出现。

根据一致并行机调度问题的研究现状，本文将采用离散场景来描述不确定的加工时间，求解一致并行机问题下的阈值坏场景集鲁棒惩罚模型。本文将基于确定性一致并行机调度的邻域构造方式，根据离散场景下阈值坏场景集鲁棒惩罚模型的特点，设计面向该鲁棒模型的的邻域构造方式。并在布谷鸟算法中加入采用该邻域构造方式的局部搜索，用该混合算法来进行离散场景下一致并行机阈值坏场景集鲁棒惩罚模型的求解。根据本文了解到的研究现状，该邻域构造方式和该鲁棒模型都是首次应用于一致并行机鲁棒调度问题。然后本文通过大量的实验仿真结果以及与其他算法的对比来展示本文所设计的邻域构造方式和混合布谷鸟算法在求解该鲁棒模型上的高效性。

本文的剩余部分安排如下：在第二节将对一致并行机的阈值坏场景集模型进行描述。第三节是对标准布谷鸟算法的介绍。第四节将详细介绍本文设计的邻域构造方式和混合布谷鸟算法。第五节是对算法的仿真和对比，最后是总结与展望。

1. 问题描述

一致并行机调度问题IPMS一般描述如下：个工件，台加工速率相同的机器，任意一个工件可在任意一台机器上进行加工，工件之间互相独立，没有优先级。一台机器同一时刻只能加工一个工件，并且一旦开始加工就无法中途停止。表示第个工件，为所有工件的集合；表示第台机器，为所有机器的集合。表示第个工件的加工时间，与机器无关，无论工件在哪一台机器上加工均保持不变。为所有调度解的集合，表示一个具体的可行调度解，如果选择在上加工，则，否则。表示机器在调度方案下的完工时间，为在上加工的所有工件的加工时间之和：



IPMS中最常见的目标函数为所有机器中最长的完工时间makespan，记作：



当所有工件的加工时间都确定时，这样的问题被称为确定性一致并行机调度问题。

本文研究用有限个离散场景表示不确定加工时间的一致并行机问题(scenario identical parallel machine scheduling, SIPMS)，其中用表示所有离散场景的集合，为场景集中的场景总数。表示一个具体的场景，为一组可能的加工时间取值。表示调度解中的机器在场景下的完工时间，表示在场景下调度解的makespan。

本文研究在坏场景集下一致并行机调度解的性能。参考文[16]，对于一个调度解，给定一个基准性能，或称为阈值，场景集中令表现性能劣于的场景即为的一个坏场景，所有坏场景的集合被称作的坏场景集，记作:



为了对抗坏场景下性能降低的风险，Wang等[16]对无法达到预期的坏场景性能施加了一定的惩罚值，定义了阈值坏场景集惩罚值，将所有坏场景下实现的性能与阈值的差值的平方项作为度量：



本文求解的阈值坏场景集惩罚模型PTM（PT model）则是将作为最终的优化目标：

(PTM):



由式和式可以发现，阈值作为衡量坏场景的标准，体现了决策者的主观偏向。其值可以凭经验或者其他方式获得，在本文中不给出具体的求解方法。根据Wang等[16]的建议，阈值的合理取值范围应不优于所有场景下的最佳期望性能。即当时，PTM模型可以表达决策者抵御坏场景下不合格性能的抗风险偏向：



确定性一致并行机问题是NP-hard问题[18]，可以看作SIPMS在单一场景下的一种特例。而在其基础上建立的涉及多个场景的PTM模型必定也为NP-hard问题[19]。为了在合理的时间内对PTM问题进行求解，本文根据SIPMS下的PTM问题提的特点采用了一种面向问题的邻域构造方式，并将其加入到布谷鸟算法和局部搜索的混合算法中。在下文中将对该邻域构造方式和算法进行详细的介绍。

1. 布谷鸟算法

在本节中将对标准的布谷鸟算法做一个简单的介绍。布谷鸟算法是近些年来较为新颖的群智能优化算法之一，由Yang和Deb[33]在2009年提出，算法设计的灵感来源是布谷鸟的寄生性产卵行为。自然界中的布谷鸟不会筑巢和育雏，它们会把蛋下在精心挑选的优质宿主的窝中让宿主替它们完成哺育工作。布谷鸟会尽力让自己蛋的花纹颜色等与宿主的蛋相似，但某些伪装的不好的蛋也会有一定的几率被发现抛弃。

布谷鸟搜寻鸟窝的方式为莱维飞行(Lévy fights)，这是一种步长满足莱维分布(Lévy distribution)的随机游走，莱维分布是一种重尾幂律分布(power-law step-length distribution with a heavy tail)，自然界中许多动物的运动行为都表现出了莱维飞行的特征。

Yang等[33]根据布谷鸟搜索宿主鸟窝进行寄生下蛋的过程设计了布谷鸟算法。并为了简化算法，制定了以下三条规则：1)每只布谷鸟每次只选择一个随机的窝下一个蛋；2)每代中产生最优秀个体的鸟巢会被保留到下一代；3)每代种群中会有占总数伪装较差的蛋被宿主以一定概率发现抛弃，为了保持蛋的数量不变，布谷鸟会选择一个新窝重新下蛋。

根据以上规则，布谷鸟算法中每个个体对应一个鸟窝或蛋，即为一个解，每代种群数量即为每代解的数量。鸟窝或蛋的优劣程度对应解的性能好坏。布谷鸟寻找鸟巢下蛋的过程就是在解空间里搜索新解的过程。蛋被抛弃寻找新窝的过程则为每代种群中性能较差的解以一定概率被新解代替的步骤。



在布谷鸟算法中，用表示当前的迭代代数，表示当前代种群，每代种群包含了个解。整个标准布谷鸟算法的流程如图1所示。主要包括了以下的几个步骤：

Step1：随机产生个个体组成初始代种群。并记录下其中性能最好的解。 Step2：对每代种群中的每个解利用式得到下一代的解，组成下一代种群：



其中为步长因子，通常取[28][29]，为满足莱维分布的随机步长：





式中的为当前代数，为给定的常数。

Step3：每代种群更新完毕后，计算更新后每一个解的性能，排在后的坏解会被随机产生的新解代替。同时根据最新种群中各解的性能更新最优解。

更新种群和替换坏解的步骤Step2和Step3会重复执行，直到当前代数达到最大迭代次数即时停止。

1. 求解SIPMS的PTM问题的混合布谷鸟算法

布谷鸟算法最初多用于求解连续性优化问题，在用于调度问题的求解时需要加入离散化的步骤来实现离散调度解和连续位置解之间的相互转换[30]。

作为一种全局搜索能力较强的智能优化算法，布谷鸟算法中用于个体位置更新的步长服从莱维分布，这使得个体容易在较大的搜索区域内进行跳跃，导致了算法的局部搜索能力较差。因而越来越多的研究人员尝试将布谷鸟算法与局部搜索混合，通过发挥各自的优势来提升求解的性能。在对一致并行机调度问题的求解过程中，Guo等[28]和Laha等[29]都采用了标准布谷鸟算法和局部搜索混合的形式，而混合后的布谷鸟算法比常规的布谷鸟算法以及其他智能优化算法表现出了更强的求解能力。

本文将先对布谷鸟算法进行离散化的处理，使其能够用于一致并行机调度问题的求解。再参照Wang等[16]的做法，根据离散场景下SIPMS的PTM问题的特点，设计了一种面向问题的邻域构造方式。本文将集中所有单一坏场景下形成的邻域解，组成合并场景邻域(united-scenario neighborhood, UN)，然后对UN进行局部搜索，并将该局部搜索方式与布谷鸟算法结合，以此来替换标准布谷鸟算法中利用随机解代替坏解的步骤。接下来将对这种混合了基于合并场景邻域局部搜索的布谷鸟算法(UN based CSA, UNCSA)的步骤进行详细的介绍，包括编码方式，初始解的构造，离散化的方法，UN的构造方式，基于UN的局部搜索过程等，最后给出UNCSA算法的总流程。

* 1. 编码和解码

在一致并行机调度问题中，主要有基于机器的编码[26](encode based on machine)和基于工件的编码[30](encode based on job)两种编码方式。考虑到本文设计的布谷鸟算法后续会加入离散化的步骤，需要解中每个维度的数都不同，因此本文采用基于工件的编码方式。对一个涉及台机器，个工件问题的一致并行机问题采用基于工件的编码，每个解由位整数组成，每个数字代表对应工件的下标。在同一台机器上加工的工件按照加工顺序组成一个子集，所有子集按机器序号排列成一个解。

以一个7工件3机器的一致并行机问题为例，各工件在两个场景下的加工时间如表1所示。

如果一个调度解为，则表示工件在上加工，

在上加工，在上加工。对应的在下的生产加工甘特图如下：



* 1. 初始解的构造

在传统的布谷鸟算法中，往往通过随机生成的方式产生初始种群[34]，而良好的初始解可以提升求解的效率。参考Laha[29]在求解确定性一致并行机问题时采用LPT规则[38]生成初始解的方式，在本文中，如果算法设定的种群大小场景个数时，选择在前个场景下利用LPT规则产生的解作为初始的种群。如果种群大小>场景个数,则前个初始解利用LPT规则生成，后个解通过对前面已产生的初始解进行扰动产生。

* 1. 连续化和离散化

布谷鸟算法最初的设计是为了求解连续优化问题，而一致并行机调度问题需要求解具体的工件排序，属于离散的优化问题。因此本文在标准的布谷鸟算法基础上加入了连续化和离散化的步骤。本文中用表示每代的调度解种群，里面的每个个体为一个离散的调度解。用表示连续位置解种群，里面的每个连续解个体与一一对应。在生成初始的调度解种群后，利用连续化步骤得到初始连续位置解种群。每次迭代中对中的个体采用莱维飞行进行更新得到，再对进行离散化得到新的调度解种群。

对于离散调度解，其对应的连续解可以通过下式计算得到：



而对连续解进行离散化得到离散解，常用的方法有最大位置规则（largest position value, LPV）[39],最小位置值规则（smallest position value，SPV）[30]等。参考文[30]，本文中采用SPV规则进行离散化操作，将连续的位置解转化为离散的调度加工序列。

以一个7工件3机器的一致并行机调度问题为例，为某代得到的连续解。将的各位数值按照升序进行排序，排序后第位的数值原本在中的位置序号即为离散解的。如表所示，连续解离散化后得到的调度解。



* 1. 基于合并场景邻域构造的局部搜索

邻域构造是局部搜索中的关键步骤，本文求解SIPMS下的PTM问题，需要考虑所有坏场景下的性能。而单一场景下的SIPMS问题可以看作是确定性一致并行机问题，因此本文在单一坏场景下参考已有的确定性一致并行机调度问题的邻域构造方式，再根据SIPMS的PTM问题的特点，设计一种面向问题的合并场景邻域构造方式用于局部搜索。

* + 1. 面向PTM问题的合并场景邻域构造

在确定性并行机调度问题中，加工时间最长的机器被称为关键机器，其余的机器被称为非关键机器[6]。确定性并行机问题通过围绕关键机器的邻域操作进行邻域的构造。其中的邻域操作可以分为交换(swap)和插入(insert)[26]，而交换又可以分为一对一交换(swap)，非对称交换(asymmetric swap)和两两交换(double swap)。插入操作是将关键机器上的一个工件移动到另一台非关键机器上加工，一对一交换是选择关键机器上的一个工件，与另一台非关键机器上的工件交换加工位置；非对称交换将关键机器上的一个工件与某台非关键机器的两个工件进行加工位置的交换；两两交换则是交换关键机器上的两个工件与某台非关键机器上的两个工件。

以上的几种方式，为面向并行机调度问题的邻域构造方式，不止用于一致并行机调度问题，而是适用于所有不同类型的并行机调度问题(一致并行机，异速并行机，无关并行机)。这几种方式在进行交换或者插入时随机选择工件进行操作。相较于其他类型的并行机调度问题，在一致并行机调度中，同一个工件在不同机器上的加工时间是固定不变的。根据这个特点，本文在原有的邻域构造方式基础上，设计了两种面向一致并行机调度问题的邻域构造方式，有针对性地选择进行插入和交换的工件。

第一种邻域构造在进行插入时，选择关键机器上加工时间最短的，插入到总加工时间最短的非关键机器上。第二种邻域构造为一对一交换。进行交换操作时，选择选择关键机器上加工时间最长的，与总加工时间最短的非关键机器上加工时间最短的进行交换。

以表1所示的下的加工数据为例，对于一个调度解，选择关键机器上加工时间最短的，插入到总加工时间最短的非关键机器上，通过插入操作得到邻域解为。进行交换操作时，选择选择关键机器上加工时间最长的，与总加工时间最短的非关键机器上加工时间最短的进行交换得到邻域解。

相比于随机选择工件，这样的操作能让加工时间的分配更加均匀，从而减少当前解的最大完工时间，产生相对优秀的邻域解。通过图3也可以看出邻域解对当前解在最大完工时间上的改善。

本文求解SIPMS下的PTM问题，需要考虑同一个解在多个坏场景下的性能。但由于不同场景下工件的加工时间不同，因此同一个解在不同场景下会呈现出不同的关键机器，也会产生完全不同的邻域解。以表1中的两个场景下的加工时间为例，当前解。如图4所示，在下，关键机器为，对应的加工时间最短的工件为，插入操作是将移动到总加工时间最短的，产生的邻域解为。而在下的关键机器为，则选择将上的插入到总加工时间最短的上加工，产生的邻域解为。

因此在求解PTM模型时需要充分考虑不同场景对邻域操作造成的影响。Wang等[16]在求解离散场景下的作业车间调度问题时，为了解决不同场景下关键路径不同导致邻域解不同的影响，提出了合并场景邻域的构造方式，将所有单一场景下的邻域解集合起来构造合并邻域：

对于一个解，其在场景下产生的邻域解用表示，合并场景邻域表示在场景集下的所有场景产生的邻域集合：



本文将参考文[16]的做法，在单场景下的一致并行机邻域解构造的基础上，围绕建立面向PTM问题的合并场景邻域。

本文用分别表示在场景下采取插入和交换操作后产生的邻域解，则。在其坏场景集下的合并坏场景邻域用表示：



同时在中，不同坏场景下生成的重复候选邻域解会被删除。

整个基于坏场景集产生合并场景邻域的伪代码如图5所示：

* + 1. 基于合并场景邻域构造的局部搜索

在UNCSA中将对每一代中性能表现较差的后的每个解构造其坏场景集下的合并场景邻域，并通过对合并场景邻域进行局部搜索得到的邻域解来代替当前的坏解。这一局部搜索的步骤记作，具体的操作步骤如图6的伪代码所示：

对于每一代种群中的每个坏解，首先求出其坏场景集和在所有坏场景下的邻域解集合，然后从中选择出性能最好的一个邻域解，如果的性能优于当前解，则用代替。否则以一定的概率采用更新。

* 1. 终止准则

UNCSA的终止条件和其他布谷鸟算法类似[26]-[30]，都以最大迭代次数作为终止条件。在下一节的实验仿真中我们将会确定的具体数值

* 1. UNCSA算法流程图

整个用于解决SIPMS下的PTM问题的UNCSA算法的流程图如下所示。首先利用4.2节提到的方法产生种群大小为的初始种群，然后将其转化为连续解种群。接着对每一代的连续解种群采用莱维飞行更新得到，利用4.3节中的SPV规则处理得到下一代的调度解种群。对新一代种群中性能表现为后的每个坏解构造其坏场景集下的合并邻域，利用4.4节中设计的局部搜索对进行更新，同时根据的更新方式对其对应的连续解进行更新。整个算法一直迭代进行直到满足终止准则。



1. 仿真与分析

本节中将对用于求解SIPMS下PTM问题的UNCSA进行仿真测试。首先将测试不同迭代次数下UNCSA的求解结果，以此确定合适的终止条件。然后通过比较UNCSA算法和其他已有的三种智能优化算法的求解结果，展示UNCSA在求解PTM模型时的优势。

除外，本文中的UNCSA算法参数参考文[28][29]设置如下：种群大小，步长因子，。每代选择的解利用局部搜索进行更新，局部搜索时坏解接受概率。

本文中的所有仿真程序均采C++语言编写实现，在Windows10操作系统环境下采用Visual Studio 2017软件运行。仿真用计算机的CPU为Intel(R) Xeon(R) Gold 5188 ,运行内存为256G。

参照文[21][28][29]中的测试算例规模，本文设置了30种不同规模的算例，并将其分为三种类型：小规模算例,中规模算例和大规模算例，每种类型各10种算例。不同规模算例的工件数和机器数如表3所示。同时参考[10]中离散场景的产生方式来对加工时间进行不确定化处理。所有算例中的工件加工时间均从中产生，其中为控制加工时间取值的参数，在中选取。每个参数组合下生成个场景的加工时间作为一个测试算例。同时对于每个参数组合，均产生5个测试算例，一共测试600个测试算例。在实际测试过程中发现，同样参数组合下的5个算例仿真结果相近，而不同时间参数的取值只是通过该变加工时间来该变最终求得的值，对算法的运行趋势和规律没有影响。因此本文接下来将在固定 的情况下，展示不同参数组合5次结果中的一个测试算例结果。

* 1. UNCSA终止准则测试

在本小节中将对UNCSA的终止准则进行测试，从不同规模类型的算例中各选取了两个规模，在固定时间参数时用UNCSA在不同的最大迭代次数下对PTM进行求解。测试的结果如表4所示，其中的为不同参数组合对应的阈值，为求解PTM模型得到的最终性能值，为整个求解过程消耗的CPU时间。



从表4中可以看出，对于UNCSA，增加最大迭代次数会提升最终结果的求解质量。在迭代次数相对较少时，UNCSA的求解结果随着代数的增加改善比较明显。而在迭代到一定代数后解的改善速度开始减缓，之后对UNCSA继续进行迭代并不能有效地提升求解结果，只会无意义地增加时间消耗。为了更清楚地观察这种趋势，本文在不同规模类型中分别挑选了三个规模的算例，绘制了与迭代次数的收敛图，如图8~图10所示。





结合收敛图和表4可以发现，采用UNCSA算法对PTM模型进行求解时，大约会在250次迭代后进入收敛，此时再牺牲CPU时间并不能明显改善解的质量。综合考虑求解质量和CPU时间的消耗，本文中的UNCSA算法的终止准则设置为最大迭代次数。

* 1. UNCSA与其他算法的比较

为了测试UNCSA算法在求解PTM模型时的有效性和优势，本文将采用UNCSA算法和其他三种求解一致并行机问题的智能优化算法一起进行求解对比：Laha等[29]设计的加入了普通局部搜索的布谷鸟算法(ICSA)，Lee等[27]设计的加入了局部搜索的遗传算法(GA+LS)，还有Xu[6]等采用的模拟退火算法(SA)。

表5展示了在时不同算法对不同规模算例的求解情况。对比了求解所用的CPU时间和最终求得的值。从表中可以看出，在求解规模较小，如时各算法的求解结果还未看出明显的差距。但整体来看，在对各种不同规模的算例求解中，UNCSA无论在求解时间还是求解结果上都领先于其他算法。相比于同样在布谷鸟算法中加入局部搜索的ICSA，UNCSA能够在更短的时间内求得更好的值。这一结果说明本文使用的面向PTM问题的合并场景邻域比普通邻域构造对解空间的挖掘更为合理，因此提升了UNCSA的局部搜索质量，使其能在每一次的迭代过程中对解空间进行更加全面深入的搜索。作为同样采用群智能优化算法混合局部搜索的GA+LS算法，虽然比起SA算法能够求得更好的，但其在四种算法中花费了最多的求解时间。而且这一时间上的差距随着问题规模的增加更加明显。这一结果反应了GS和CSA求解效率之间的差距，CSA能够更加快速高效地求解更为复杂的调度问题。而作为典型的串行邻域搜索算法，SA的每次迭代过程只涉及单一解的邻域搜索，因此相比GA+LS消耗相对较少的CPU，求解时间与ICSA相近。但其对解空间进行的探索范围有限，在所有测试算法中最后的求解结果性能最差。

为了更清楚地反应UNCSA相比其他算法的求解优势，本文绘制了四种算法求解三个规模算例时的求解过程收敛图，如图11~图13所示。从图中可以进一步看出在四种算法中，SA算法由于无法对解空间开展较为全面的搜索，最早进入了收敛，最终求解结果表现最差。而其他三种混合算法在算法运行的前期的表现相近。但由于搜索效率的差异，GA+LS对解的改善程度越来越小，与ICSA和UNCSA求解结果之间的差距也逐渐增大。对于同样采用了CSA作为全局搜索方式的ICSA和UNCSA，其收敛曲线前期的走势相近。但在算法运行的后期，相比普通的邻域搜索方式，采用合并场景邻域构造的UNCSA在每次迭代过程中可以搜寻到更多邻域解，每一次迭代过程中对邻域的搜索也更为深入，因此在ICSA进入收敛后，UNCSA依旧可以对现有的结果进行一定程度的更新。

求解结果和算法收敛曲线充分体现了UNCSA算法相较其他算法在求解PTM问题时的优势。



1. 总结和展望

本文求解了不确定一致并行机的阈值坏场景集惩罚模型PTM，其中不确定的加工时间利用离散场景表示。给定一个阈值，筛选出场景集中的所有坏场景，通过优化该阈值下所有坏场景性能造成的惩罚值来提升系统的抗风险能力。

为了更好地求解该模型，本文在标准的布谷鸟算法基础上加入了邻域搜索的部分，提出了一种改进的布谷鸟算法UNCSA。除了根据一致并行机调度问题的特点设计了单一场景下新的面向一致并行机调度的邻域构造方式外，还结合PTM模型的问题背景，采用了基于合并坏场景的邻域构造方式，将当前解在所有坏场景下产生的邻域解集合起来形成合并场景邻域进行局部搜索。本文将UNCSA算法和其余的三种智能优化算法进行了比较，仿真结果表明UNCSA算法兼顾了布谷鸟算法和合并场景邻域局部搜索的优点，在求解SIPMS的PTM模型时更具优势。

未来的研究工作可能会朝着以下的方向进行：

1. 在本文的求解过程中，PTM模型的阈值是预先确定的，并没有给出具体的求解方法。之后会先设计算法确定阈值，然后再对PTM模型进行求解。
2. 基于阈值的坏场景集模型最初在作业车间调度问题上应用，本文首次将其运用到一致并行机问题中，之后会尝试用于其他的调度问题。
3. 本文将采用合并场景邻域构造方式的局部搜索与布谷鸟算法进行了结合，未来可以尝试用更多不同的智能优化算法进行混合求解。

参考文献：

1. Rocholl J , Mnch L . Decomposition heuristics for parallel-machine multiple orders per job scheduling problems with a common due date[J]. Journal of the Operational Research Society, 2019:1-17.
2. Lee C H . A New Discrete Electromagnetism-Like Mechanism Algorithm for Identical Parallel Machine Scheduling Problem with Eligibility Constraints in Metal Nuts Manufacturing[J]. Arabian Journal for Science & Engineering, 2017, 42(8):1-12.
3. Gungor, Murat, Unal, et al. A parallel machine lot-sizing and scheduling problem with a secondary resource and cumulative demand[J]. International Journal of Production Research, 2018.
4. Al-Khamis T, Rym M’Hallah. A two-stage stochastic programming model for the parallel machine scheduling problem with machine capacity[J]. Computers & Operations Research, 2011, 38(12):1747-1759.
5. Dubois D, Prade H . Qualitative possibility theory and its applications to constraint satisfaction and decision under uncertainty[J]. International Journal of Intelligent Systems, 1999, 14(1):45-61.
6. Xu X, Cui W, Lin J, et al. Robust makespan minimisation in identical parallel machine scheduling problem with interval data[J]. International Journal of Production Research, 2013, 51(12): 3532-3548.
7. Naji W. Robust and stable optimization for parallel machine scheduling problems[D]. Grenoble Alpes, 2018.
8. Kala R, Lamboray C, Vanderpooten D. Lexicographic α-robustness: An alternative to min–max criteria[J]. European Journal of Operational Research, 2012, 220(3): 722-728.
9. Yang J , Yu G . On the Robust Single Machine Scheduling Problem[J]. Journal of Combinatorial Optimization, 2002, 6(1):17-33.
10. Kouvelis P, Daniels R L, Vairaktarakis G. Robust scheduling of a two-machine flow shop with uncertain processing times[J]. IIE Transactions, 2000, 32(5): 421-432.
11. Li Z , Ierapetritou M G . Robust Optimization for Process Scheduling Under Uncertainty[J]. Industrial & Engineering Chemistry Research, 2008, 47(12):4148-4157.
12. Sarin S C , Nagarajan B , Jain S , et al. Analytic evaluation of the expectation and variance of different performance measures of a schedule on a single machine under processing time variability[J]. Journal of Combinatorial Optimization, 2009, 17(4):400-416.
13. Drwal M , Jerzy Józefczyk. Robust min–max regret scheduling to minimize the weighted number of late jobs with interval processing times[J]. Annals of Operations Research, 2020, 284(2).
14. Wang S , Cui W . Approximation algorithms for the min-max regret identical parallel machine scheduling problem with outsourcing and uncertain processing time[J]. International Journal of Production Research, 2020(1):1-14.
15. Wang B, Xia X, Meng H, et al. Bad-scenario-set robust optimization framework with two objectives for uncertain scheduling systems[J]. IEEE/CAA Journal of Automatica Sinica, 2017, 4(1): 143-153
16. Wang B, Wang X, Lan F, et al. A hybrid local-search algorithm for robust job-shop scheduling under scenarios[J]. Applied Soft Computing, 2018, 62: 259-271.
17. Drwal M, Roman, Complexity of interval minmax regret scheduling on parallel identical machines with total completion time criterion[J]. Operations Research Letters, 2016.
18. Garey M R, Johnson D S. Computers and intractability[M]. San Francisco: freeman, 1979.
19. Lenstra J K , Kan A H G R . Computational Complexity of Discrete Optimization Problems[J]. Annals of Discrete Mathematics, 1979, 4(08):121-140.
20. Dell’Amico, Mauro, Martello S . Optimal Scheduling of Tasks on Identical Parallel Processors[J]. Orsa Journal on Computing, 1995, 7(2):191-200.
21. Mokotoff E . An exact algorithm for the identical parallel machine scheduling problem[J]. European Journal of Operational Research, 2004, 152(3):758-769.
22. Kashan A H , Karimi B . A discrete particle swarm optimization algorithm for scheduling parallel machines[J]. Computers & Industrial Engineering, 2009, 56(1):216-223.
23. Alharkan I, Bamatraf K, Noman M A, et al. An order effect of neighborhood structures in variable neighborhood search algorithm for minimizing the makespan in an identical parallel machine scheduling[J]. Mathematical Problems in Engineering, 2018.
24. Kim J G, Song S, Jeong B J. Minimising total tardiness for the identical parallel machine scheduling problem with splitting jobs and sequence-dependent setup times[J]. International Journal of Production Research, 2020, 58(6): 1628-1643
25. Wang S, Zou H, Wang S. A Tabu-GA-based parallel machine scheduling with restrained tool resources[J]. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2020: 0954405420928691.
26. Chen J, Pan Q K, Wang L, et al. A hybrid dynamic harmony search algorithm for identical parallel machines scheduling[J]. Engineering Optimization, 2012, 44(2): 209-224.
27. Lee W C , Wang J Y , Lee L Y . A hybrid genetic algorithm for an identical parallel-machine problem with maintenance activity[J]. Journal of the Operational Research Society, 2015.
28. Guo P, Cheng W, Wang Y. Parallel machine scheduling with step-deteriorating jobs and setup times by a hybrid discrete cuckoo search algorithm[J]. Engineering Optimization, 2015, 47(11): 1564-1585.
29. Laha D, Gupta J N D. An improved cuckoo search algorithm for scheduling jobs on identical parallel machines[J]. Computers & Industrial Engineering, 2018, 126: 348-360.
30. Wang H, Wang W, Sun H, et al. A new cuckoo search algorithm with hybrid strategies for flow shop scheduling problems[J]. Soft Computing, 2017, 21(15): 4297-4307.
31. Dasgupta P , Das S . A Discrete Inter-Species Cuckoo Search for flowshop scheduling problems[J]. 2015.
32. Li X, Yin M. A hybrid cuckoo search via Lévy flights for the permutation flow shop scheduling problem[J].International Journal of Production Research, 2013, 51(16): 4732-4754.
33. Yang X S , Deb S . Cuckoo Search: Recent Advances and Applications[J]. Neural Computing & Applications, 2014, 24(1):169-174.
34. Yang X S , Deb S . Engineering Optimisation by Cuckoo Search[J]. International Journal of Mathematical Modelling & Numerical Optimisation, 2010, 1(4):330-343.
35. Gandomi A H , Yang X S , Alavi A H . Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems[J]. Engineering with Computers, 2013, 29(1):17–35.
36. Civicioglu P , Besdok E . A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms[M]. Kluwer Academic Publishers, 2013, 39(3):315–346.
37. Yang X S, Deb S. Multiobjective cuckoo search for design optimization[J]. Computers & Operations Research, 2013, 40(6): 1616-1624.
38. Graham R L. Bounds on multiprocessing timing anomalies[J]. SIAM journal on Applied Mathematics, 1969, 17(2): 416-429.
39. Liang J J , Pan Q K , Tiejun C , et al. Solving the blocking flow shop scheduling problem by a dynamic multi-swarm particle swarm optimizer[J]. The International Journal of Advanced Manufacturing Technology, 2010.