![摘要](https://imgkr2.cn-bj.ufileos.com/77f9280f-26d9-4818-8afc-f68c3db9d7c3.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=52Bt4%252F54ocTI02FqhxBkez98T68%253D&Expires=1599533430)

摘要:对车间作业调度问题特征与最优完工时间之间的关系进行了统计研究。为此，手动构造了JSSP的\*\*380个\*\*新颖的特征，每一个都代表一个特定的问题特征，然后我们借助机器学习中常用的统计分析度量，如皮尔逊相关系数，来建立这些特征与最优makespan的相关性，并作为验证这些特征可以捕获大部分相关性的方法。我们进一步使用它们来构造机器学习模型,试图在\*\*不实际求解\*\*给定实例的情况下预测最佳完工时间，这种预测是通过将案例大致分类成分别低于和高于平均水平的两类来实现的。对15000个随机生成的问题实例的交叉验证和准确度大概在80%的测试精度进行了报告和讨论。我们认为,给定获得的相关信息,人类专家可以了解JSSP结构,从而可以设计更好的案例、启发式或超启发式、标准案例，并在调度过程的各个阶段做出更好的决策和执行更好的信息权衡。为了支持这个想法，我们还演示了所获得的见解在实际应用程序是多么有用。

# 1 介绍

作者先是指出JSSP是NP-完全问题，并给出文章的研究对象是\*\*标准的JSSP\*\*，即不考虑顺序依赖准备时间、交货期、工序中断，同时要满足一台机床同一时间只能加工一道工序，一个工件同一时间只能在一台机床上加工。接着对现有的调度方法进行了评述，启发式或分派规则在求解实际规模案例时具有一定的优势，但是尽管如此，启发式很难预测\*\*何时、针对何类型\*\*的案例才能得到合理的结果，也就是说启发式的自适应能力较弱，另一方面，\*\*人类专家\*\*对启发式的干预和调整往往能提升其性能；机器学习调度试图将智能集成于调度中，但是这种系统过分依赖于人类专家的智慧，在大多数动态环境下失效，机器学习在调度研究方面分为两类：\*\*自适应启发式和自适应超启发式\*\*，前者将机器学习模型作为一种更具自适应的启发式，后者则可以帮助设计启发式或选择分派规则；尽管存在更智能的调度算法，但是许多调度员仍然凭借自己的专业知识进行决策，这些专业知识仅仅是来源于经验，却\*\*没有理论基础\*\*来帮助理解调度问题结构，有必要对特征和目标的相关性进行研究；除了调度求解以外，还应关注车间设计层，即定义调度案例，案例的设计其实也是遵循一定规律的，但是当前研究只关注案例的求解，因此车间设计人员只能相信自己的洞察力或重复执行近似算法，车间设计和车间调度是交织在一起的，因此应该一并进行研究。比如，如果一个工件的总加工时间比其他工件大很多，或者说工件的总加工时间方差很大，那么最优调度很可能无法充分利用资源，因为一个工件主导了makespan，那么在车间设计时就可以避免这样的配置。

这篇文章的贡献在于开发特征集并利用机器学习方法以实现更优的设计决策。

# 2 方法

## 2.1 问题表达

n:工件数量

m：机床数量

P：加工时间矩阵

M：机床分派矩阵

$p\_{jk}$:工件j的第k道工序的加工时间

$m\_{jk}$:工件j的第k道工序的加工机床

S：调度解

$s\_{it}$:机床i加工工件j=$s\_{it}$

有了JSSP案例x的数据后，可以将其表达为一个数值对($\phi,L$)，其中$\phi$为包含F个特征的集合，L为标签。监督机器学习问题就定义为\*\*使用训练案例的($\phi,L$)来构造一个模型以仅知道测试案例y的特征值的情况下预测其L值\*\*。如果L值是离散的，那么该问题就是分类问题，否则就是回归问题。度量特征与标签相关性的方法有两种：\*\*过滤器\*\*度量统计相关性而与具体的机器学习算法无关，\*\*包装法\*\*使用特定的机器学习算法度量预测能力。文中这两种方法都使用到了，其中后者采用了SVM作为机器学习算法。

## 2.2 特征设计和评估

首先定义一组初步特征，然后使用排序方法(上面提到的过滤或包装方法)来衡量这些特征的预测能力。这种方法的问题在于，有一些特征仅当和其他特征组合考虑时才会有较高的预测能力，所以在没有测试其所有可能的组合时不能丢弃这些特征，但是遍历所有这些组合也是不现实的，文中只考虑单个特征和选择的特征组。

当考虑单个特征时，使用了\*\*皮尔逊相关系数\*\*（ Pearson Correlation Coefficient，PCC）,\*\*信噪比\*\*（ Signal-to-Noise Ratio，SNR）和\*\*T检验\*\*（T-test）作为过滤法来评估特征与标签的相关性，其中PCC度量了两个连续变量间的线性相关的强度，取值范围为[-1,1]，PPC为0意味着两个变量间没有线性相关性，如果为1或-1则意味着完全相关或负相关。如果X和Y为从两个变量中采集的样本集合，那么这两个变量的PCC可以按公式（1）计算得到。对于调度问题，X就是单个特征，Y就是标签L。

$$\operatorname{PCC}(X, Y)=\frac{\sum\_{i}\left(X\_{i}-\bar{X}\right)\left(Y\_{i}-\bar{Y}\right)}{\sqrt{\sum\_{i}\left(X\_{i}-\bar{X}\right)^{2}} \sqrt{\left(\sum\_{i}\left(Y\_{i}-\bar{Y}\right)^{2}\right)}}\tag{1}$$

当标签为二进制时，即只能取值0或1时，SNR和T-tets可以按公式（2）和（3）计算得到，其中+和—分别对应于正标签和负标签案例所关联的变量，所以$\mu\_{+}$,$\sigma\_{+}$和$n\_{+}$为正标签案例的均值、标准差和案例个数。

$$\operatorname{SNR}(X, Y)=\frac{\left|\mu\_{+}-\mu\_{-}\right|}{\sigma\_{+}+\sigma\_{-}}\tag{2}$$

$$T(X, Y)=\frac{\mu\_{+}-\mu\_{-}}{\sqrt{\frac{\sigma\_{+}}{n\_{+}}+\frac{\sigma\_{-}}{n\_{-}}}}\tag{3}$$

这些过滤器方法可以很好的度量相关性，但是一次只能度量一个变量，包装法可以克服这种缺点，只不过需要更多的计算量。文中训练一个SVM作为包装算法，当仅考虑单个特征时使用\*\*二次核\*\*，当考虑特征组合时使用\*\*高斯核\*\*。

## 2.3 标签

标签是JSSP案例的一个特殊的目标特征。最优makespan $C\_{min}$是最重要的特征之一。考虑表1中的示例JSSP，其最优解如图1所示。如果直接使用$C\_{min}$作为标签，那么对于不同的问题将具有不同的$C\_{min}$，且总加工时间越高，$C\_{min}$也可能更高，案例间总加工时间一样时，工件与机床比n/m更高时，案例也会有更高的$C\_{min}$。因此使用公式（4）中的调度效率作为标签更合适，该度量指标成功地考虑了总加工时间和机器数量对JSSP实例的相对最优最大完工时间的影响，同时保持了最大完工时间的直接影响。

![表1 示例JSSP案例和最优解](https://imgkr2.cn-bj.ufileos.com/bd0ea4fb-fa16-46e9-9aa5-79de56a06c64.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=30Zyos81nlqNSBy7r75O%252FwCMnX0%253D&Expires=1599533881)

![图1 最优调度](https://imgkr2.cn-bj.ufileos.com/417c1496-d0d4-46a6-9289-b690b7b4253b.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=fYrqylqtrL3hY1rabHxojxk6rk8%253D&Expires=1599533909)

$$C^{\prime}=1+\frac{\sum\_{i} l\_{i}}{\sum\_{j, k} p\_{j k}}\tag{4}$$

其中，$\sum\_{j, k} p\_{j k}$是所有工件的总加工时间，$l\_{i}$为机床$i$的总空闲时间。注意，对于给定的调度，如果没有空闲时间，那么$C^{\prime}=1$，且随着空闲时间的增大$C^{\prime}$也会增大。现在考虑公式（5）：

$$\sum\_{j, k} p\_{j k}+\sum\_{i} l\_{i}=C . m\tag{5}$$

该式可以这么理解：如图1所示，$\sum\_{i} l\_{i}$是所有白色空间，即总机床空闲时间，那么再加上所有加工时间就得到了高为$m$宽为$C$的矩形面积$C.m$。两边同时除以$\sum\_{j, k} p\_{j k}$就得到：

$$1+\frac{\sum\_{i} l\_{i}}{\sum\_{j, k} p\_{j k}}=\frac{C . m}{\sum\_{j, k} p\_{j k}}\tag{6}$$

可以看到，等式的左边就是$C^{\prime}$，于是有：

$$C^{\prime}=\frac{C \cdot m}{\sum\_{j, k} p\_{j k}}\tag{7}$$

该式表明$C^{\prime}$实际上是$C$关于总加工时间和机床数量的归一化，这一归一化makespan度量是比较不同尺寸问题实例的理想方法，从而使其成为本文的理想标签。为了使得标签可以二分类，将感兴趣的问题的平均值作为参考，标记最优$C^{\prime}$大于平均值的案例为正，否则标记为负。由于$C^{\prime}$在不同规模的案例间是可比较的，就可以得到一个完全异构分类，此外，对于给定的案例，已知$C^{\prime}\_{\min}$就等于已知最优解的makespan $C\_{\min}$,这也是有价值的，因为如果我们能设计一个机器学习模型来预测$C^{\prime}\_{\min}$，\*\*就能够在不用实际计算最优调度的情况下预测最优完工时间。从而帮助设计更好的启发式或更好地洞察问题特征与最优最大完工时间的相关性\*\*。

# 3 JSSP特征

特征的选择对于监督学习至关重要，本文中提出的特性涵盖了JSSP实例的两个方面，一个是一般问题配置(直接取自P和M矩阵)，另一个是时间方面(取自应用于问题的调度规则或启发式的输出)。

## 3.1 配置特征

配置特征直接取自P和M矩阵，当推导这组特征时，矩阵中的这些值与加工时间和机器无关，它们只是数字。这组特征约占本文提出的特征的75%(总共380个特征中的288个)，提取其特征的计算成本很低。

### 3.1.1 整体问题案例

包括\*\*机床数量m\*\*，\*\*工件数量n\*\*和\*\*总加工时间\*\*$\sum\_{j, k} p\_{j k}$。

### 3.1.2 单道工序

通过观察加工时间矩阵P，可以根据\*\*单道工序的加工时间\*\*($\mathrm{OPT}\_{\mathrm{jk}} \equiv \ p\_{j k}$)的均值、中值、标准差、最小值和最大值，得到\*\*5个\*\*特征。

### 3.1.3 工件

每个工件$j$对应于$P$中包含$m$个元素的一行，将这些元素相加就得到了工件加工时间$\mathrm{JPT}\_{\mathrm{j}} \equiv \sum\_{k} p\_{j k}$，同工序特征一样，将所有工件$JPT$的均值、中值、标准值、最小值和最大值作为5个特征。由于在分类时可能会比较不同规模的案例，所以应该注意到这5个值严重依赖于平均$JPT$，平均$JPT$越高，这些值也会越高。为了克服这种明显的相关性，对这5个值\*\*关于平均$JPT$进行归一化\*\*，丢弃第一个归一化值，因为该值总是为1，然后将其余4个作为新的特征值。同时使用平均$OPT$进行归一化，又得到5个新特征值，那么就得到\*\*14个\*\*特征值。\*\*如果不同问题间机床数量相同，那么就这两种归一化是一样的，只需保留其中一个。\*\*

### 3.1.4 机床

每台机床$i$加工$n$个工件对应的$n$道工序，如果将这些工序的加工时间求和，就可以得到机床$i$的机床加工时间$\left(\mathrm{MPT}\_{\mathrm{i}} \equiv \sum\_{j, k \mid m\_{j k}=i} p\_{j k}\right)$，得到$m$个MPT值后，按照工件特征的提取方式，就可以得到机床的14个特征值。

### 3.1.5 工序位置

在矩阵P和M中，每一列$k$对应一个工序位置或工序槽（operation slot），如果将列k中的元素相加，就得到工序槽加工时间$\left(\operatorname{OSPT}\_{\mathrm{k}} \equiv \sum\_{j} p\_{j k}\right)$。不同于JPT和MPT，只考虑OSPT的标准差及其关于平均OSPT和平均OPT的归一化值，从而得到\*\*3个\*\*特征值（\*注意，平均OSPT等于平均MPT，因为所考虑的JSSP中，工序数始终等于机床数\*）。由于OSPT表示的每个工序槽上的加工量，因为其位置值对$C^{\prime}$有重要影响，如是否第一道具有很高的加工需求而第二道工序却没有，同样靠近中间或后面的工序槽也有特殊的影响。为了捕获这些位置影响，定义了\*\*11个\*\*位置特征，首先将第一个值OSPT1作为单独的特征放在一边，然后将剩余值分成10等份，对重叠部分进行平均，以便将OSPT值均匀地传播到位置区域。

![图2 位置特征构造](https://imgkr2.cn-bj.ufileos.com/1731b6e3-6c38-4192-8cfb-5623418cb8cd.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=dwhK9ZHvRWOrm1wIjuOHQQzV3U4%253D&Expires=1599570148)

在图2中，第一个工序槽有其自己的分区，其余的被分成10个区域。区域2的值通过第二个工序槽和第三个工序槽的加权平均计算得到，为了简单起见，也可以直接取重叠槽的平均值。重要的是，这些区域，每个对应一个特征，保存了有关OSPTs的位置值的信息，因此，如果OSPTs从左到右(随着时间的推移)存在特殊的模式，这些特征将有望能够捕捉至少一部分。显然，当机器数量远高于10台时，我们可以考虑使用更多的区域来获得足够的位置分辨率，但是对于m小于30的情况，当前数量应该可以。

这种位置划分将给我们总共11个新特性，在分别关于OSPT和OPT标准化后我们将有\*\*33个\*\*关于OSPT的位置特性。

现在考虑机床分配矩阵M中的列，该矩阵配置的一个重要方面是不同作业的机器配置的安排。如表1中给定的示例问题，工件0，1和2均同时需要机床2，这种冲突意味着工作的等待时间会增加，从而导致较大的完工时间，同时注意到，第一个操作槽上的任何作业都不需要机器0,1和3，这将直接影响机床的空闲时间，从而增加$C^{\prime}\_{\min}$。当我们移到下面的操作槽时，这些影响将不那么明显，但显然存在通过新特征可以捕获的模式。文中定义了两组特征来捕获这些影响，一组是\*\*工序槽缺失机床\*\*（OSMM），另一组是\*\*工序槽重复机床\*\*（OSRM）。对于OSMM，只需简单地统计工序槽中没出现的机床数量，从而得到m个值，如表2。然后计算这些值的均值、中值、标准差、最小值和最大值，再将这些值关于机床数量进行归一化，归一化的目的是机床数量会直接影响这些值，那么就得到\*\*9个\*\*特征值。按照同样的方式就可以得到OSRM的\*\*9个\*\*特征，只不过这次计算的是工序槽中重复机床的数量。\*\*注意到对于该示例问题，这两组特征值是相同的，这是因为机床数量等于工件数量，那么每重复一台机床就代表缺失一台机床，反之亦然。\*\*

![表2 示例问题中从M提出的工序槽特征](https://imgkr2.cn-bj.ufileos.com/922463ba-5ae8-407e-b36d-1f5a99a2b212.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=FYS6%252FwkqDWhlqcLDQrMVZDnXSkQ%253D&Expires=1599570204)

尽管OSRM捕获了机器分配中的冲突，但它仍然不能恰当地捕获强度，因为我们只是简单地将重复相加，而不考虑冲突存在时实际发生了什么。假设在第一个工序槽，机床0重复1次，机床1重复1次，对于这些机器中的每一台，其中一个作业必须等待一次才能有机会被处理。现在考虑另一个案例，其中机床0重复2次，OSRM同之前的案例一样还是2，但实际情况是，其中一个作业必须等待一次，另一个作业必须等待两次，所以实际上可能存在3个等待期。出于这个原因，需要定义另外一组特征，即放大的工序槽重复机床（OSRMA），每次重复计算的值随着重复次数的增加而线性增加。对于机床重复1次，OSRMA为1，重复2次就为3，重复3次就为6，如果重复r次就为$r(r+1)/2$。按照计算OSMM和OSRM同样的方式得到OSRMA的\*\*9个\*\*特征值。

以上这三类参数的位置值同样很重要，例如，调度开始或结束位置上的机床分配冲突会有不同影响，为了捕获这种潜在的影响，按照OSPT位置值得计算方式，定义OSMM、OSRM和OSRMA的位置值，只不过这里仅包含了分别关于平均OSPT和平均OPT的归一化值，即每个参数有22个特征值，总共有66个。

### 3.1.6 工序槽组合加工时间

这类特征包含两类：工序槽重复机床组合加工时间（OSCOMB）和扩大工序槽重复机床组合加工时间（OSCOMBA）。OSCOMB的思想是尽管OSRM捕获了机床分配的冲突，但是这些冲突高度取决于加工时间的，不难看出，如果加工时间增大，等待时间和对$C^{\prime}\_{\min}$的影响也会增大。\*\*OSCOMB通过将OSRM值与相应工序加工时间的平均值相乘来捕获这种效果\*\*。例如考虑示例问题中的第一个工序槽，机床2出现3次（重复2次），对应的工序加工时间是96，37和21，对于第一个工序槽，OSCOMB计算为（3-1）\*mean(96,37,21)=102.67，再加上机床4的重复值，就得到了第一个工序槽的总OSCOMB值为155.17。类似于OSRM，为OSCOMB计算9个非位置特征和22个位置特征，只是这次又分别增加了关于机床数量和平均OPT乘积的归一化的5个非位置特征值和11个位置特征值。最后考虑OSCOMB的放大版，基于OSRMA获得\*\*47个\*\*特征值。

### 3.1.7 机床负荷

机床$i$的机床负荷定义为矩阵M中的每一个工序槽中机床$i$出现的次数所构成的数组，如示例中机床0的负荷为（0，1，1，1，2），因为在第一个工序槽中没有使用机床0，但在最后一个工序槽中使用机床0两次，表3中展示了所有机床的负荷。

![表3 机床负荷和特征值](https://imgkr2.cn-bj.ufileos.com/8bb86c0e-309b-4ecb-90bd-c1b9e0d7d735.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=3MPmm26dsefupuE7jeJPrr3k8jg%253D&Expires=1599570229)

现在基于机床负荷定义两组特征。一组是机床负荷均匀性（MLDU），另一组是机床负荷空洞（MLDV）。MLDU是通过将每台机器负载的净变化相加来计算的，如考虑表3中的机床M0，从槽0道槽1机床负荷的净变化是1，接下来的两个槽变化是0，最后一个槽变化是1，从而机床M0的MLDU=2。MLDV就是简单地统计每台机床的机床负荷中‘0’出现的次数。同时考虑MLDV的放大版MLDVA，渐增地统计0连续出现的值，这里不再使用重复值，而是考虑连续出现。例如，机床0出现了一次‘0’，所以其MLDVA是1。机床1有一个单独出现，记为1，两个连续出现，记为1+2=3，机床1的MLDVA总共为4。机床有3个连续出现，那么记为1+2+3=6。MLDVA值强调0值得连续出现，因为我们认为，当一台机器上的负荷具有较低的均匀性时，该机器更有可能处于空闲状态。

获得MLDU、MLDV和MLDVA之后，就可以计算它们的均值、中值、标准差、最小值和最大值，然后关于均值和机床数量分别进行归一化，得到9特征值，那么总共就是\*\*27个\*\*特征值。此外还额外增加了\*\*2个\*\*特征，以更多强调机床负荷中‘0’的连续出现，一个特征是根据统计连续出现长于机床数量30%的数量来计算的，另一个则是机床数量的40%，同样也是随着序列的增大逐渐递增计数。例如，示例中机床数量为5，机床0没有连续出现，因此两个特征值均为0，机床1和2均有一个长度为2的0序列，由于2≥0.3\*5，两台机床的第一个特征值都为1，由于2≥0.4\*5，两台机床的第二个特征值也都为1。机床3有一个长度为3的0序列，这两个特征值为2。所以对于示例问题，第一个特征和第二个特征的值均为1+1+2=4。

## 3.2 时间特征

配置特征忽略了P和M矩阵中的值的真正含义，但这里我们使用的事实是，P包含时间值，而M包含按时间运行的机器。实现这一点的一种方法是模拟向机器分配操作以及如何解决冲突。冲突解决是使用简单的调度规则完成的，最终结果是一个实际的调度，可能是最优的，也可能不是。我们可以把最终的调度特征，比如它的最大完工时间，作为一组特征，并遵循仿真并使用随时间变化的特性的度量作为另一组特征。

### 3.2.1 SPT

将按照SPT得到的makespan和$C^{\prime}$作为\*\*2个\*\*特征，然后统计解决冲突的次数，并作为\*\*另一个\*\*特征（冲突个数-1），再查看所有工件和机床的完工时间，计算均值、中值、标准差、最小值和最大值，将这\*\*10个\*\*值作为特征，其中均值不做归一化处理，其他值除以均值以归一化。

SPT的最后一组特征称为放大调度工序槽重复工件（SOSRJA），查看获得的调度的每一个工序槽，统计重复工件的个数，当重复次数超过两次时，线性增加该值。根据这n个值计算均值、中值、标准差、最小值和最大值，然后关于工件数量进行归一化得到\*\*5个\*\*特征值。

### 3.2.2 LPT

类似于SPT，只不过在解决冲突时选择加工时间最长的工序。同样得到\*\*18个\*\*特征。

### 3.2.3 最长剩余工时MWRM

选择剩余工时最长的工件来解决冲突，同样得到\*\*18个\*\*特征。

### 3.2.4 最短剩余工时LWRM

同样得到\*\*18个\*\*特征。

### 3.2.5 先入先出+最长剩余工时FIFO+MWRM

就绪最早的工件优先级最高，如果存在多个这样的工件，那么按照MWRM消解冲突。同样得到\*\*18个\*\*特征。

在计算了所有单个启发式特征后，选择最小的$C^{\prime}\_{\min}$作为\*\*一个\*\*特征，该特征表达了所能获得的真实$C^{\prime}\_{\min}$的最优估计，然后对其关于最大JPT和最大MPT的较大值进行归一化，得到\*\*另一个\*\*特征值。

# 4 结果和讨论

总共生成了15000个JSSP实例，并将它们等分为三个大小不同的组：第一组实例包含8个作业和8台机器，称为8J8M组，第二组实例包含10个作业和10台机器，称为10J10M组，第三组实例包含11个作业和11台机器，称为11J11M组。加工时间和机床分配均是按照\*\*均匀随机分布\*\*生成的，且加工时间范围为[1,100]。所有案例通过IBM ILOG OPL计算得到最优解，记录makespan，并据此计算$C^{\prime}$作为标签。为了评估问题大小对特征性能的影响，我们对\*\*两批\*\*不同的实例进行了实验，一个批只包含10J10M实例，另一个批包含随机混合的15,000个实例。每个批进一步划分为交叉验证和测试实例，每个批的每组中留出1000个实例用于最终测试，其余的用于交叉验证。

![案例设置](https://imgkr2.cn-bj.ufileos.com/ab8fad3e-2b6f-4c17-b23f-c55db19548dc.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=5B6R%252FQtBF6QUxL2jvexvbtg1O%252Bw%253D&Expires=1599562219)

首先进行单特征的相关性分析，使用PCC、SNR和T-test获取各个特征的平均排名，然后使用SVM通过5折交叉验证分类来训练和评估每个特征的性能。表6和7展示了两个批次的前十名和后十名总体特征，以及前10名配置特征。通过观察结果，可以得出以下结论：

\* 两个批中排名第一的特征是启发式$C^{\prime}\_{\min}$的最小值。这并不奇怪，因为这个特性表示我们使用五种启发式方法所能得到的最接近标签的值。图3显示了在10J10M中，这个特征和最优配置特征与真实$C^{\prime}$之间的相关性。注意，两者之间都存在明显的线性关系，但是总体特征第一名更强烈。

![图3 相关性](https://imgkr2.cn-bj.ufileos.com/5e6ccf2e-f49f-418a-9a6b-0140ca911feb.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=CHeb8cyL65jWc%252FMPKnJZaHtftrE%253D&Expires=1599563555)

\* 在两个批的前十名中存在一个有意思的特征，MWRM启发式冲突解决的归一化，原因并不明显，但这表明试错是特性工程的重要组成部分。有些特性似乎没有什么用，或者它们的影响不明显，但可能会变得很重要，反之亦然。关于冲突解决特征的另一个有趣之处是，只有与MWRM相关联的特征才这么好。例如，FIFO\_MWRM的冲突解决特征在两个批次中排名110和74，远远不够好。

![](https://imgkr2.cn-bj.ufileos.com/64f1c6fd-d7bc-4472-b66d-b0a824724fa1.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=LT8nx3UerhLK1TS0ZTnPPJ%252FTrcw%253D&Expires=1599568308)

\* 由于n和m在第一批中是不变的，因此它们对应的特征在这批列表的底部(这意味着它们完全独立于标签)，然而，这两个特征在混合批中的排名分别为94和95，这突出了问题大小对特征性能的影响。

![](https://imgkr2.cn-bj.ufileos.com/8586c5d8-4e2a-4eb5-bbb9-949eba923b4b.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=x6vdcZubzTfoPnIxcC9aECrH3i8%253D&Expires=1599568714)

\* OSCOMBA归一化组中的一些特征的性能优于所有其他配置特征，如果我们注意到其他一些“显而易见”的配置特征的排名稍低，这就特别有趣了。例如，很明显，作业处理时间的标准偏差越高，$C^{\prime}\_{\min}$越有可能高，然而，相应的特征与Cmin的相关性并不像OSCOMBA的归一化均值那样强。

\* 没有一个特征组可以说是主导了排名。我们可以在混合批排序的后面看到OSCOMB组的特征，甚至在10J10M批排序的底部看到启发式相关的特征，这进一步强调了在确定某些特征影响的显著性时反复试验的重要性，尽管如此，一些特性组的平均表现还是比其他的要好。makespan和$C^{\prime}$组在两个批中都占据前五名，FIFO\_MWRM是性能最好的启发式，配置特征中，JPT、MPT、OSCOMBA和MLDVA是最好的，而MLDU在两个批中都是最差的。

![](https://imgkr2.cn-bj.ufileos.com/bb2a30a0-4266-4ba8-ace9-213c2ef9852a.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=lx%252FH1A2Q0YDZJIeP1NP%252FOuDClR0%253D&Expires=1599568746)

\* 位置特征的性能是喜忧参半的，且靠近两端位置的特征(如第一个和第11个位置)比靠近中间位置的特征(如第5和第6个位置)表现得好得多。

在个体相关分析后，接下来的实验是使用SVM对10J10M批次的一组特征进行分类。我们使用以下简单的算法进行分类F次，每次使用不同的特征集。对于两个不同大小的类，得到的分类精度如图4所示。在下面的算法中，$\phi$为当前要使用的特征集，R为所有特征的排序列表(R(1)为排名最高的特征，R[1..X]表示最靠前的X个特征)。我们使用了一个自动调整参数的高斯核支持向量机。

![](https://imgkr2.cn-bj.ufileos.com/be378078-1e76-4d3d-b857-9aa07b41a585.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=h1qkwRazotVIw2sybwakLZRva9I%253D&Expires=1599568824)

![图4 分类精度](https://imgkr2.cn-bj.ufileos.com/26730c10-0e0d-419d-a95c-4b5bbcef7d81.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=SVkAn9sdHM6O28qtlBJkSm9eRrU%253D&Expires=1599568893)

从图4的结果图中可以看出，当前100-150个特征被包括进来时，准确率达到了最高水平，之后，其余的特征实际上对性能是有害的。这可能意味着它们没有那么有用，尽管在采用更高级的特征选择方法(如贪婪向前/向后选择)之前，我们不能合理地断言这一点。

最后一个实验是评估类别在分类中的表现，为此，我们选择了三组不同的特征，并使用带有自动调优高斯核的SVM对每批的交叉验证和测试实例进行分类，一组由所有的特征组成，下一组由配置特征组成，最后一组又时间特征组成。

![表9 分类性能](https://imgkr2.cn-bj.ufileos.com/74f722ec-4666-495a-b66d-b625b686f0f6.png?UCloudPublicKey=TOKEN\_8d8b72be-579a-4e83-bfd0-5f6ce1546f13&Signature=GaSIk4bNsvfLQZRe7VpBXKVbNfM%253D&Expires=1599569390)

表9显示了每个集合的分类精度。注意，当应用于10J10M或混合批次时，所有集合似乎都有类似的性能。这里重要的一点是，配置特征可以达到70%的准确率，而添加时间特征会使准确率提高约10%。时间特征单独使用时也可以执行得很好，因此这可能意味着配置特征对它们来说是多余的。在实用性方面，获得配置特征可能比获得时间特征更容易。