运筹学与控制论

DOI:10.11721/cqnuj20260408

边长度有不确定性的树图上单机排序与选址集成问题*

陈贝宁，耿志超
(郑州大学 数学与统计学学院，郑州 450001)

摘要:研究树图上单机排序与选址集成问题,其中工件的加工时间、运输起始时间以及运输速度是给定的;树图上的顶点和边也是给定的,但边长度具有不确定性,且它的值依赖于一组离散场景;机器的放置位置和工件的加工顺序需由决策确定;工件在加工之前需先被运输至机器位置。关注排序指标即工件的最大完工时间(或时间表长),采用鲁棒优化方法对最大绝对鲁棒问题和最大遗憾问题,分别给出了多项式时间算法;对最大相对遗憾问题,设计了完全多项式时间近似方案。最后,通过实例说明了算法的执行过程和有效性。

关键词:排序与选址集成问题;多场景;鲁棒优化;最大完工时间;树图;不确定的边长度
中图分类号:O224　　**文献标志码**:A　　**文章编号**:1672-6693(2026)04-0039-09

作为运筹学的 2 个重要分支,排序论和关于设施选址的理论已经以相对独立的方式被广泛地研究。然而,在许多实际应用中又同时涉及排序和选址事项,这类问题常被称作“排序与选址集成(scheduling-location,SchLoc)问题”。例如,由 Elvikis 等人[1]描述的“用于货物装卸运输的集装箱港口设施、生产规划中可移动的机器设备”。对 SchLoc 问题设计求解算法时,相较于独立或依次处理排序与选址事项,采用统筹考虑二者的综合方式是更好选择。在典型的 SchLoc 问题中,需确定每台机器的放置位置(可以在某个节点或某条边上),分配工件至机器上,并为分配在同一台机器上的工件确定加工顺序。特别地,要求每个工件(该工件的初始位置在某个节点上)在加工之前先沿图上的某条路径运输至它被分配的机器的放置位置。另外,由于受到如机器故障、劳动力短缺以及市场波动等多种因素的影响,许多优化问题中的参数往往呈现不确定性。用“多场景(multi-scenario)”来建模这种不确定性是常用方法之一[2]。鲁棒优化是处理与多场景相关的优化问题的有效方法之一,目标是寻找在各种场景下均可接收的可行解。

Hamacher 等人[3]最早研究了单机上最小化最大完工时间的 SchLoc 问题,分别对树图和一般图上的情形给出了多项式时间算法。Elvikis 等人[1]将此研究扩展至平图上(planar graph)的情形,允许机器放置在任意位置(不局限在节点或边上),对多个特殊情形分别给出了多项式时间算法。Kalsch 等人[4]也研究了平图上的 SchLoc 问题,对 2 个排序指标最大完工时间和完工时间之和分别给出了分支定界算法,并在不同范数(如欧几里得范数、直线型范数)意义下通过数据实例验证了算法的有效性。Krumke 等人[5]研究了树图上边长度有不确定性的鲁棒 SchLoc 问题(简记为 R-SchLoc 问题),将不确定的边长度建模为区间型场景集 $\left\{\xi \in \mathbf{R}^E : \xi_e \in [\underline{l}_e, \overline{l}_e], \forall e \in E, \sum_{e \in E} (\xi_e - \underline{l}_e) \leqslant \Gamma\right\}$,对最大绝对鲁棒问题给出了多项式时间算法。注意,当所有工件的加工时间和运输起始时间均为 0 时,一般图上边长度有不确定性的 R-SchLoc 问题退化为鲁棒 1-中心问题;当机器位置固定时,它又等价于工件的到达时间有不确定性的鲁棒排序问题。Elvikis 等人[1]证明了一般图上边长度有不确定性的最大遗憾鲁棒 1-中心问题是强-NP 难的。这意味着对应的一般图上最大遗憾 R-SchLoc 问题也是强-NP 难的。Averbakh 等人[7]对树图上边长度有不确定性的最大遗憾鲁棒 1-中心问题给出了多项式时间算法。Bachtler 等人[8]研究了单机上工件的到达时间有不确定性的最小化最大完工时间的鲁棒排序问题,将工件的到达时间的不确定性建模为 2 个场景集:

* **收稿日期**:2025-12-31　**修回日期**:2026-04-05　**网络出版时间**:2026-07-24T16:27
资助项目:国家自然科学基金面上项目(No. 12271491);河南省自然科学基金面上项目(No. 252300423006)
第一作者简介:陈贝宁,男,研究方向为组合优化、排序与调度,E-mail:bnchenmath@163.com;**通信作者**:耿志超,男,讲师,博士,E-mail:zcgeng@zzu.edu.cn
网络出版地址:https://link.cnki.net/urlid/50.1165.n.20260724.1323.002

$$\left\{\xi: r_j^{(\xi)} \in [\underline{r}_j, \overline{r}_j], \forall j, \sum_{J_j \in J} (r_j^{(\xi)} - \underline{r}_j) \leqslant \Gamma_1\right\},$$

$$\{\xi: r_j^{(\xi)} \in [\underline{r}_j, \overline{r}_j], \forall j, |J_j \in J: r_j^{(\xi)} \neq \underline{r}_j| \leqslant \Gamma_2\},$$

分别对最大绝对鲁棒问题和最大遗憾问题给出了多项式时间算法。Geng 等人[9]研究了与文献[8]相似的问题，不同之处在于前者的研究中，工件到达时间的不确定性被建模为离散场景集$\{\xi_1, \xi_2, \cdots, \xi_m\}$，分别对最大绝对鲁棒问题、最大遗憾问题以及最大相对遗憾问题给出了多项式时间算法。另外一些关于鲁棒排序问题的较新研究工作可参考 Wu 等人[10-11]、Bougeret 等人[12]的研究成果。

不同于 Krumke 等人[5]考虑的区间型场景集，本文考虑离散型场景集$\{\xi_1, \xi_2, \cdots, \xi_m\}$[2]。注意，正如 Bachtler 等人[8]和 Geng 等人[9]描述的那样，区间型场景集下的(优化)问题实例可以转化为离散型场景集下的问题实例，尽管此转换用时可能不是多项式时间；反之不然。从这个角度看，本文研究的最大绝对鲁棒问题是 Krumke 等人[5]研究的问题的扩展。此外，本文也研究了最大遗憾问题和最大相对遗憾问题。具体地，对最大绝对鲁棒问题和最大遗憾问题分别给出了多项式时间算法；对最大相对遗憾问题，设计了完全多项式时间近似方案。

1 问题描述

本文研究的 R-SchLoc 问题可描述如下：给定 n 个工件 $J_1, J_2, \cdots, J_n$，这些工件的集合记为 J，即 $J=\{J_1, J_2, \cdots, J_n\}$，再给定树图 T 且 $T=(V,E)$。这些工件需安排在单机上不中断地完成加工，T 上有 n 个节点和 $n-1$ 条边，令 $|V|=n$，$|E|=n-1$。工件 J_j 的加工时间为 p_j 且 $p_j>0$，初始位置在节点 v_j。T 中的各边的长度依赖于 m 个场景 $\xi_1, \xi_2, \cdots, \xi_m$，记 $S=\{\xi_1, \xi_2, \cdots, \xi_m\}$。边 $e\in E$ 在场景 ξ_k 下的长度为 $l_e^{(k)}$ 且 $l_e^{(k)}>0$，场景 ξ_k 也可看作 $n-1$ 维向量 $(l_{e_1}^{(k)}, l_{e_2}^{(k)}, \cdots, l_{e_{n-1}}^{(k)})$。在场景 ξ_k 下节点 v_i 与 v_j 之间的距离 $d^{(k)}(v_i, v_j)$ 是 T 中连接这 2 个节点的唯一路径上所有边的长度之和。令 e 为 T 中连接任意 2 个节点 v_i 和 v_j $(i<j)$ 的边。引入参数 x $(x\in[0,1])$ 来定义 e 上的一点，记作 (e,x)，满足在场景 ξ_k 下从点 (e,x) 到节点 v_i 与 v_j 的距离分别为 $d^{(k)}(v_i,(e,x))=xl_e^{(k)}$ 和 $d^{(k)}(v_j,(e,x))=(1-x)l_e^{(k)}$。当 $x=0$ 时，(e,x) 即为节点 v_i；当 $x=1$ 时，(e,x) 即为节点 v_j。为方便起见，自此之后，在不产生异义的前提下 T 也表示树图中的所有点(包括节点和边上的点)，$x\in T$ 也代表 T 中的任意一点。

对于连接节点 v_a 和 v_b $(a<b)$ 的边 e，定义在场景 ξ_k 下从点 $x\in e$ 到节点 $v_j\in V\backslash\{v_a, v_b\}$ 的距离 $d^{(k)}(v_j,x)$ 为：

$$d^{(k)}(v_j,x)=\begin{cases} d^{(k)}(v_j,v_a)+xl_e^{(k)}, v_j\in T_a \\ d^{(k)}(v_j,v_b)+(1-x)l_e^{(k)}, v_j\in T_b \end{cases}。\tag{1}$$

其中：T_a 和 T_b 分别表示从树图 T 中删去边 e 得到的以 v_a 和 v_b 为根节点的子树。机器可以放置在树图中的任意节点或边上，具体位置由决策确定。每个工件在加工之前需先沿图中的路径运输至机器位置。假设工件 J_j 的运输起始时间为 s_j $(s_j\geqslant 0)$，运输速度为 τ_j $(\tau_j>0)$。相对于(机器)位置 $x\in T$，在场景 ξ_k 下工件 J_j 的到达时间为：

$$r_j^{(k)}(x)=s_j+\frac{d^{(k)}(v_j,x)}{\tau_j}。\tag{2}$$

假设所有参数 $p_j, s_j, \tau_j, l_e^{(k)}$ 均为实数。令 π 为一个工件排序(或置换)，Π 为所有排序的集合。用 $C_j^{(k)}(\pi,x)$ 表示工件 J_j 在 π 中在场景 ξ_k 下相对于位置 x 的完工时间。若工件 J_j 在 π 中排在第 i 个位置，即 $\pi(i)=j$，则 $C_j^{(k)}(\pi,x)$ 可按下面方式计算：

$$C_{\pi(1)}^{(k)}(\pi,x)=r_{\pi(1)}^{(k)}(x)+p_{\pi(1)}, C_{\pi(i)}^{(k)}(\pi,x)=\max\{C_{\pi(i-1)}^{(k)}(\pi,x), r_{\pi(i)}^{(k)}(x)\}+p_{\pi(i)}, \forall i=2,3,\cdots,n。$$

π 在场景 ξ_k 下相对于位置 x 的时间表长 $C_{\max}^{(k)}(\pi,x)$ 定义为工件的最大完工时间，即 $C_{\max}^{(k)}(\pi,x)=C_{\pi(n)}^{(k)}(\pi,x)$。

假设在每种场景下各条边的长度是已知的(或者说各种场景下的边长度以及场景个数 m 都是输入实例的一部分)，但哪种场景实际出现是未知的。为了防止最坏情况发生，本文采用鲁棒优化方法求解 3 个具体问题：最大绝对鲁棒问题、最大遗憾问题和最大相对遗憾问题，目标函数分别为：

$$\min_{x\in T,\pi\in\Pi}\{\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)\}\}, \min_{x\in T,\pi\in\Pi}\{\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)-C^{(k)}\}\}, \min_{x\in T,\pi\in\Pi}\left\{\max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi,x)-C^{(k)}}{C^{(k)}}\right\}\right\}。$$

其中:$C^{(k)}$ 表示在场景 ξ_k 下所有排序的时间表长的最小值,即 $C^{(k)}=\min\limits_{x\in T,\pi\in\Pi}C_{\max}^{(k)}(\pi,x)$,它仅依赖于场景 ξ_k。

2　算法与分析

本节不再逐个求解最大绝对鲁棒问题(P1)、最大遗憾问题(P2)和最大相对遗憾问题(P3),而是考虑下面更具一般性的问题。

树图上附加场景偏差的 R-SchLoc 问题(简记为 RSAB)

输入:工件 $J=\{J_1,J_2,\cdots,J_n\}$,加工时间 $p_1,p_2,\cdots,p_n$,运输起始时间 $s_1,s_2,\cdots,s_n$ 及运输速度 $\tau_1,\tau_2,\cdots,\tau_n$;树图 $T=(V,E)$;场景 $S=\{\xi_1,\xi_2,\cdots,\xi_m\}$ 和场景偏差 $D^{(1)},D^{(2)},\cdots,D^{(m)}$。

输出:机器位置 $x\in T$ 和工件排序 π,使得 $\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}$ 尽可能小。

注意,RSAB 问题中的常数 $D^{(1)},D^{(2)},\cdots,D^{(m)}$ 可能依赖于对应的场景,但与机器位置和工件排序无关。接下来,本文对 RSAB 问题给出多项式时间算法,该算法的设计思想借鉴文献[5],并能直接用于求解问题 P1 和 P2,也能通过重复调用近似求解问题 P3。在后续讨论中,除非特别说明,e 总是表示树图上任意给定的边,端点为节点 v_a 和 $v_b(a<b)$,x(对应机器位置)总是表示 e 上任意一点(包括 v_a 和 v_b)。为方便起见,若 $\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}$ 在所有排序中是最小的,则称排序 π 为最优排序(相对于位置 x,下同),也称 $\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}$ 为最优值;若 J_j 是 π 中最后 1 个恰好在它到达时间 $r_j^{(k)}(x)$ 开始加工的工件(这意味着在场景 ξ_k 下 π 的时间表长 $C_{\max}^{(k)}(\pi,x)$ 等于 $r_j^{(k)}(x)$ 与 π 中 J_j 及后面的所有工件的总加工时间之和),则称 J_j 为 π 在场景 ξ_k 下相对于位置 x 的关键工件;若场景 $\xi_u\in S$ 满足 $C_{\max}^{(u)}(\pi,x)+D^{(u)}=\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}$,则称 ξ_u 是 π 的一个相对于位置 x 的最坏场景。另外,对任意的 $x\in T,j\in\{1,2,\cdots,n\}$,用 $\xi_{k(j,x)}$ 表示工件 J_j 相对于位置 x 的关键场景。在此场景下,$r_j^{(k(j,x))}(x)+D^{(k(j,x))}$ 的值是在各种场景下最大的,即:

$$r_j^{(k(j,x))}(x)+D^{(k(j,x))}=\max_{\xi_k\in S}\{r_j^{(k)}(x)+D^{(k)}\}。\tag{3}$$

引理 1　假设 π^* 为按数值 $r_j^{(k(1,x))}(x)+D^{(k(1,x))},r_j^{(k(2,x))}(x)+D^{(k(2,x))},\cdots,r_j^{(k(n,x))}(x)+D^{(k(n,x))}$ 的非减顺序排列工件得到的排序,则 π^* 是一个相对于位置 x 的最优排序。

证明　令 π 是一个任意排序。假设 $\xi_u\in S$ 是 π^* 的一个最坏场景,J_{j^*} 是 π^* 在场景 ξ_u 下(相对于位置 x,下同)的关键工件,且 J_{j^*} 在 π^* 中排在第 v 个位置。由上述假设可知:

$$C_{\max}^{(u)}(\pi^*,x)+D^{(u)}=\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\},\tag{4}$$

$$C_{\max}^{(u)}(\pi^*,x)=r_{j^*}^{(u)}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*}(j)。\tag{5}$$

比较排序 π 与 π^*,可能出现以下 2 种情形。

情形 1,π^* 中排在 J_{j^*} 后面的所有工件在 π 中依旧排在 J_{j^*} 的后面。则在场景 ξ_u 下 π 的时间表长满足:

$$C_{\max}^{(u)}(\pi,x)\geqslant r_{j^*}^{(u)}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*}(j)。\tag{6}$$

由式(4)~(6)可推出:

$$\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}\geqslant C_{\max}^{(u)}(\pi,x)+D^{(u)}\geqslant C_{\max}^{(u)}(\pi^*,x)+D^{(u)}=\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}。$$

再由 π 的任意性可知,π^* 是一个最优排序。

情形 2,存在工件在 π^* 中排在 J_{j^*} 后面,而在 π 中排在 J_{j^*} 的前面。假设 J_l 是这些工件中在 π 中排得最靠前的工件,且 J_l 在 π 中排在第 v' 个位置。这说明 π^* 中 J_{j^*} 及后面的工件在 π 中排在 J_l 的位置或后面。因而,在 J_l 的关键场景 $\xi_{k(l,x)}$ 下 π 的时间表长满足:

$$C_{\max}^{(k(l,x))}(\pi,x)\geqslant r_l^{(k(l,x))}(x)+\sum_{v'\leqslant j\leqslant n}p_{\pi(j)}\geqslant r_l^{(k(l,x))}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*(j)}。$$

进而得:

$$C_{\max}^{(k(l,x))}(\pi,x)+D^{(k(l,x))}\geqslant r_l^{(k(l,x))}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*(j)}+D^{(k(l,x))}。\tag{7}$$

由于在 π^* 中工件按 $r_j^{(k(1,x))}(x)+D^{(k(1,x))},r_j^{(k(2,x))}(x)+D^{(k(2,x))},\cdots,r_j^{(k(n,x))}(x)+D^{(k(n,x))}$ 的非减顺序排序且 J_l

在 J_{j^*} 前面,故:

$$r_l^{(k(l,x))}(x)+D^{(k(l,x))}\geqslant r_{j^*}^{(k(j^*,x))}(x)+D^{(k(j^*,x))}。\tag{8}$$

又由工件 J_{j^*} 的关键场景 $\xi_{k(j^*,x)}$ 的定义可知:

$$r_{j^*}^{(k(j^*,x))}(x)+D^{(k(j^*,x))}=\max_{\xi_k\in S}\{r_{j^*}^{(k)}(x)+D^{(k)}\}\geqslant r_{j^*}^{(u)}(x)+D^{(u)}。\tag{9}$$

综合式(4)、(5)、(7)~(9)得:

$$\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi,x)+D^{(k)}\}\geqslant C_{\max}^{(k(l,x))}(\pi,x)+D^{(k(l,x))}\geqslant r_l^{(k(l,x))}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*(j)}+D^{(k(l,x))}\geqslant$$
$$r_{j^*}^{(k(j^*,x))}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*(j)}+D^{(k(j^*,x))}\geqslant r_{j^*}^{(u)}(x)+\sum_{v\leqslant j\leqslant n}p_{\pi^*(j)}+D^{(u)}=$$
$$C_{\max}^{(u)}(\pi^*,x)+D^{(u)}=\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}。$$

再由 π 的任意性可知,π^* 是一个最优排序。 证毕

记 $R_j(x)=r_j^{(k(j,x))}(x)+D^{(k(j,x))}$,$\forall j\in\{1,2,\cdots,n\}$。用 $\pi^*(x)$ 表示按 $R_1(x),R_2(x),\cdots,R_n(x)$ 的值非减的顺序排列工件得到的排序。根据引理1,$\pi^*(x)$ 是一个相对于位置 x 的最优排序。由式(1)~(3)可观察出,$R_j(x)$ 有如下性质。

引理2 $\forall j\in\{1,2,\cdots,n\}$,$k\in\{1,2,\cdots,m\}$,$r_j^{(k)}(x)+D^{(k)}$ 是关于 x 的线性函数,斜率为 $\dfrac{l_e^{(k)}}{\tau_j}$ 或 $-\dfrac{l_e^{(k)}}{\tau_j}$;$R_j(x)$ 是关于 x 的分段线性函数。

由式(3)可知,$\forall j\in\{1,2,\cdots,n\}$,可以通过求解至多 $m(m-1)/2$ 个方程

$$r_j^{(k)}(x)+D^{(k)}=r_j^{(k')}(x)+D^{(k')}(\forall k,k'\in\{1,2,\cdots,m\})$$

来确定 $R_j(x)$ 的线性分割点。事实上,按这种方式得到的点集(由所有可能的方程的解构成的集合)包含了 $R_j(x)$ 的所有线性分割点,也可能有一些点并不是 $R_j(x)$ 的线性分割点。但是,这对后续讨论没有影响。为方便起见,按上述方式得到的所有点均称作边 e 的一级分割点,也将点 $x=0$ 和点 $x=1$ 视为 e 的一级分割点。令 α 和 $\alpha'(\alpha<\alpha')$ 为 e 的任意2个相邻的一级分割点。由引理2可知,$R_j(x)$ 也有下面的性质。

引理3 $\forall j\in\{1,2,\cdots,n\}$,当 $x\in[\alpha,\alpha']$ 时,$R_j(x)$ 是关于 x 的线性函数。

尽管引理3说明了当 $x\in[\alpha,\alpha']$ 时,$R_j(x)$ 是关于 x 的线性函数,但是,$\forall j,j'\in\{1,2,\cdots,n\}$,$R_j(x)$ 与 $R_{j'}(x)$ 之间的大小关系,即 $R_j(x)>R_{j'}(x)$ 或 $R_j(x)\leqslant R_{j'}(x)R_j(x)\leqslant R_{j'}(x)$,可能随位置 x 的变化而有差异。为此,$\forall j,j'\in\{1,2,\cdots,n\}$,需计算 $R_j(x)$ 与 $R_{j'}(x)$ 在 $x\in[\alpha,\alpha']$ 上的可能交点。为方便起见,称这些所有可能交点为边 e 的二级分割点,也将 α 和 α' 视为 e 的二级分割点。令 β 和 $\beta'(\beta<\beta')$ 视为 e 在区间 $[\alpha,\alpha']$ 内任意2个相邻的二级分割点。由引理1~引理3可知,下面的引理也成立。

引理4 $\forall j,j'\in\{1,2,\cdots,n\}$,当 $x\in[\beta,\beta']$ 时,$R_j(x)>R_{j'}(x)$ 或 $R_j(x)<R_{j'}(x)$ 恒成立。因而,相对于任意位置 $x\in[\beta,\beta']$ 的最优排序 $\pi^*(x)$ 是相同的。

在接下来的讨论中,将关于位置 $x\in[\beta,\beta']$ 的最优排序 $\pi^*(x)$ 简记为 π^*。下一步,关注 π^* 在最坏场景下的目标值 $\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}$。

引理5 $\max\limits_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}=\max\limits_{1\leqslant j\leqslant n}\{R_j(x)+P(j,\pi^*)\}$。其中,$P(j,\pi^*)$ 表示 π^* 中 J_j 及后面的所有工件的总加工时间。

证明 假设 $\xi_u\in S$ 是 π^* 相对于位置 $x\in[\beta,\beta']$ 的一个最坏场景,J_{j^*} 是 π^* 在场景 ξ_u 下相对于位置 x 的关键工件。则有:

$$C_{\max}^{(u)}(\pi^*,x)+D^{(u)}=\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\},\tag{10}$$

$$C_{\max}^{(u)}(\pi^*,x)=r_{j^*}^{(u)}(x)+P(j^*,\pi^*)。\tag{11}$$

断言:工件 J_{j^*} 在场景 ξ_u 下相对于位置 x 的到达时间 $r_{j^*}^{(u)}(x)$ 满足 $r_{j^*}^{(u)}(x)+D^{(u)}=\max\limits_{\xi_k\in S}\{r_{j^*}^{(k)}(x)+D^{(k)}\}$。

假设 $r_{j^*}^{(u)}(x)+D^{(u)}<\max\limits_{\xi_k\in S}\{r_{j^*}^{(k)}(x)+D^{(k)}\}$,则存在某个场景 $\xi_{u'}\in S$ 使得在此场景下 J_{j^*} 相对于位置 x 的到达时间 $r_{j^*}^{(u')}(x)$ 满足 $r_{j^*}^{(u')}(x)+D^{(u')}>r_{j^*}^{(u)}(x)+D^{(u)}$。由式(10)和式(11)可知,$\pi^*$ 在场景 $\xi_{u'}$ 下相对于位置 x 的

目标值 $C_{\max}^{(u')}(\pi^*,x)+D^{(u')}$ 满足:

$$C_{\max}^{(u')}(\pi^*,x)+D^{(u')}\geqslant r_{j^*}^{(u')}(x)+P(j^*,\pi^*)+D^{(u')}>r_{j^*}^{(u)}(x)+P(j^*,\pi^*)+D^{(u)}=C_{\max}^{(u)}(\pi^*,x)+D^{(u)}。$$

这与假设 ξ_u 是 π^* 相对于位置 $x\in(\beta,\beta')$ 的一个最坏场景相矛盾。

由断言、式(3)和记号 $R_{j^*}(x)$ 的含义可知:

$$r_{j^*}^{(u)}(x)+D^{(u)}=r_{j^*}^{(k(j^*,x))}(x)+D^{(k(j^*,x))}。\tag{12}$$

综合式(10)~(12)可得:

$$\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}=r_{j^*}^{(u)}(x)+D^{(u)}+P(j^*,\pi^*)=R_{j^*}(x)+P(j^*,\pi^*)。\tag{13}$$

注意,在关键场景 $\xi_{k(j,x)}$ 下相对于位置 x,每个工件 J_j 的到达时间 $r_j^{(k(j,x))}(x)$ 和排序 π^* 的时间表长 $C_{\max}^{(k(j,x))}(\pi^*,x)$ 满足:

$$C_{\max}^{(k(j,x))}(\pi^*,x)\geqslant r_j^{(k(j,x))}(x)+P(j,\pi^*)。$$

进而有:

$$\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}\geqslant r_j^{(k(j,x))}(x)+P(j,\pi^*)+D^{(k(j,x))}=R_j(x)+P(j,\pi^*)。\tag{14}$$

其中:最后的等式成立是因为 $R_j(x)=r_j^{(k(j,x))}(x)+D^{(k(j,x))}$。再由式(13)和式(14)可知:

$$\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x)+D^{(k)}\}=\max_{1\leqslant j\leqslant n}\{R_j(x)+P(j,\pi^*)\}。$$

证毕

记 $F(x)=\max\limits_{1\leqslant j\leqslant n}\{R_j(x)+P(j,\pi^*)\}$。根据引理 5,$F(x)$ 是排序 π^* 在最坏场景下相对于位置 $x\in(\beta,\beta')$ 的目标值。又由引理 3 可知 $R_j(x)$ 是关于 $x\in(\beta,\beta')$ 的线性函数,且对于最优排序 π^* 和任意给定的 $j\in\{1,2,\cdots,n\}$,$P(j,\pi^*)$ 是常数。因而,$F(x)$ 有如下性质。

引理 6　当 $x\in[\beta,\beta']$ 时,$F(x)$ 是关于 x 的分段线性函数。

进一步,可通过求解至多 $n(n-1)/2$ 个方程 $R_j(x)+P(j,\pi^*)=R_{j'}(x)+P(j',\pi^*)(\forall j,j'\in\{1,2,\cdots,n\})$ 确定 $F(x)$ 在区间 $[\beta,\beta']$ 上的线性分割点。事实上,按这种方式得到的点集包含了 $F(x)$ 在区间 $[\beta,\beta']$ 上所有线性分割点,也可能有一些点不是 $F(x)$ 的线性分割点。但是,这依旧不影响后续讨论。为方便起见,称这些得到的所有点均为边 e 的三级分割点,也将 β 和 β' 视为 e 的三级分割点。令 γ 和 $\gamma'(\gamma<\gamma')$ 为 e 上任意 2 个相邻的三级分割点。根据引理 6,$F(x)$ 也有下面的性质。

引理 7　当 $x\in[\gamma,\gamma']$ 时,$F(x)$ 是关于 x 的线性函数。因而,$F(x)$ 在区间 $[\gamma,\gamma']$ 上的最小值可以由 $\min\{F(\gamma),F(\gamma')\}$ 给定。

基于上面的讨论,可对 RSAB 问题给出如下算法。

算法 1　第 1 步,计算树图 T 上任意一对节点在各个场景下的最短路径长度。

第 2 步,对每条边 $e\in E$,执行以下操作:

第 2.1 步,通过求解方程 $r_j^{(k)}(x)+D^{(k)}=r_j^{(k')}(x)+D^{(k')}$,$\forall k,k'\in\{1,2,\cdots,m\}$,$j\in\{1,2,\cdots,n\}$,确定边 e 的一级分割点。

第 2.2 步,对边 e 上的每对相邻的一级分割点 α 和 $\alpha'(\alpha<\alpha')$,通过求解方程 $R_j(x)=R_{j'}(x)$,$x\in[\alpha,\alpha']$,$\forall j,j'\in\{1,2,\cdots,n\}$,确定 e 的二级分割点。

第 2.3 步,对边 e 上的每对相邻的二级分割点 β 和 $\beta'(\beta<\beta')$,执行以下操作:

1) 在区间 (β,β') 中任取一点 x,按 $R_1(x),R_2(x),\cdots,R_n(x)$ 的值非减的顺序排列工件,确定相对于区间 $[\beta,\beta']$ 上任意位置的相同的最优排序 π^*;

2) 通过求解方程 $R_j(x)+P(j,\pi^*)=R_{j'}(x)+P(j',\pi^*)$,$x\in[\beta,\beta']$,$\forall j,j'\in\{1,2,\cdots,n\}$,确定 e 的三级分割点。

第 2.4 步,对边 e 上的每对相邻的三级分割点 γ 和 $\gamma'(\gamma<\gamma')$,计算 $F(x)$ 在区间 $[\gamma,\gamma']$ 上的最小值 $\min\{F(\gamma),F(\gamma')\}$,并记录对应的机器位置。

第 3 步,比较所有边上任意相邻的三级分割点 γ 和 γ' 的由上一步得到的 $\min\{F(\gamma),F(\gamma')\}$,确定最小值,并输出最优的机器位置、最优排序和最优目标值。

用下面的实例解释算法 1 的执行过程:有 6 个工件 J_1,J_2,J_3,J_4,J_5,J_6;2 个场景 ξ_1,ξ_2;场景偏差 $D^{(1)}=2.8,D^{(2)}=3.5$;树图 T(如图 1)有 6 个节点 v_1,v_2,v_3,v_4,v_5,v_6 和 5 条边 e_1,e_2,e_3,e_4,e_5;每个工件 J_j 的初始

位置在节点 v_j。其他参数值见表1。

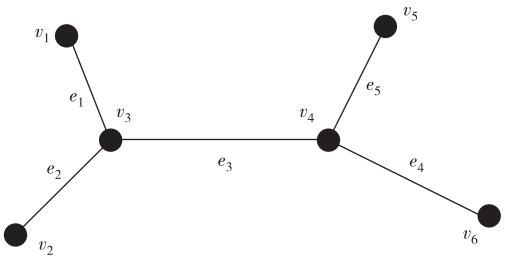

图1 有6个节点和5条边的树图

Fig. 1 A tree graph with six nodes and five edges

表1 其他参数值

Tab. 1 Other parameter values

参数类型	参数名称	参数值
工件	加工时间	$p_1=2,p_2=4,p_3=3,p_4=5,p_5=4,p_6=1$
	运输起始时间	$s_1=1,s_2=0,s_3=2,s_4=2,s_5=1,s_6=0$
	运输速度	$\tau_1=1,\tau_2=2,\tau_3=1,\tau_4=2,\tau_5=4,\tau_6=2$
边长度	场景ξ_1	$l_{e_1}^{(1)}=4,l_{e_2}^{(1)}=7,l_{e_3}^{(1)}=4,l_{e_4}^{(1)}=6,l_{e_5}^{(1)}=6$
	场景ξ_2	$l_{e_1}^{(2)}=2,l_{e_2}^{(2)}=5,l_{e_3}^{(2)}=8,l_{e_4}^{(1)}=3,l_{e_5}^{(2)}=1$
任意2个节点间的距离	场景ξ_1	$d^{(1)}(v_1,v_2)=11,d^{(1)}(v_1,v_3)=4,d^{(1)}(v_1,v_4)=8,d^{(1)}(v_1,v_5)=13,$ $d^{(1)}(v_1,v_6)=14,d^{(1)}(v_2,v_3)=7,d^{(1)}(v_2,v_4)=11,d^{(1)}(v_2,v_5)=16,$ $d^{(1)}(v_2,v_6)=17,d^{(1)}(v_3,v_4)=4,d^{(1)}(v_3,v_5)=9,d^{(1)}(v_3,v_6)=10,$ $d^{(1)}(v_4,v_5)=6,d^{(1)}(v_4,v_6)=6,d^{(1)}(v_5,v_6)=11$。
	场景ξ_2	$d^{(2)}(v_1,v_2)=7,d^{(2)}(v_1,v_3)=2,d^{(2)}(v_1,v_4)=10,d^{(2)}(v_1,v_5)=11,$ $d^{(2)}(v_1,v_6)=13,d^{(2)}(v_2,v_3)=5,d^{(2)}(v_2,v_4)=13,d^{(2)}(v_2,v_5)=14,$ $d^{(2)}(v_2,v_6)=16,d^{(2)}(v_3,v_4)=8,d^{(2)}(v_3,v_5)=9,d^{(2)}(v_3,v_6)=11,$ $d^{(2)}(v_4,v_5)=1,d^{(2)}(v_4,v_6)=3,d^{(2)}(v_5,v_6)=4$。

为减少冗余,下面只列出算法1在执行时对边 e_3 的迭代过程。

1) 一级分割

一级分割点:$\alpha_1=0,\alpha_2=0.35,\alpha_3=0.65,\alpha_4=1$。

函数 $R_j(x)$:$R_1(x)=2x+4.5(0\leqslant x<0.35)$ 和 $R_1(x)=4x+3.8(0.35\leqslant x\leqslant 1)$;

$R_2(x)=-2x+8.3(0\leqslant x\leqslant 1)$;$R_4(x)=-x+10.5(0\leqslant x\leqslant 1)$;

$R_3(x)=-4x+8.8(0\leqslant x\leqslant 0.65)$和$R_3(x)=-2x+7.5(0.65\leqslant x\leqslant 1)$;

$R_5(x)=-0.5x+7.25(0\leqslant x\leqslant 1)$;$R_6(x)=-x+10(0\leqslant x\leqslant 1)$。

2) 二级分割

分割区间:[0,0.35],[0.35,0.65],[0.65,1]。

二级分割点:$\beta_1=0,\beta_2=0.25,\beta_3=0.95,\beta_4=0.442\ 857,\beta_5=0.625,\beta_6=0.65,\beta_7=0.7,\beta_8=0.75,\beta_9=0.766\ 667,\beta_{10}=1$。

3) 三级分割

分割区间:[0,0.25],[0.25,0.3],[0.3,0.35],[0.35,0.442 857],[0.442 857,0.625],[0.625,0.65],[0.65,0.7],[0.7,0.75],[0.75,0.766 667],[0.766 667,0.84],[0.84,1]。

三级分割点:$\gamma_1=0,\gamma_2=0.25,\gamma_3=0.3,\gamma_4=0.35,\gamma_5=0.442\ 857,\gamma_6=0.625,\gamma_7=0.65,\gamma_8=0.7,\gamma_9=0.75,\gamma_{10}=0.766\ 667,\gamma_{11}=0.84,\gamma_{12}=1$。

每个子区间上的最优排序、最优机器位置和最优值:

在[0,0.25]上:$\pi^*=(J_1,J_5,J_2,J_3,J_6,J_4)$,在 $x=0.25$ 且 $F_{\min}=24.125$;

在[0.25,0.3]上:$\pi^*=(J_1,J_5,J_3,J_2,J_6,J_4)$,在 $x=0.3$ 且 $F_{\min}=24.1$;

在[0.3,0.35]上:$\pi^*=(J_1,J_5,J_3,J_2,J_6,J_4)$,在 $x=0.3$ 且 $F_{\min}=24.1$;

在[0.35,0.442 857]上:$\pi^*=(J_1,J_5,J_3,J_2,J_6,J_4)$,在 $x=0.35$ 且 $F_{\min}=24.2$;

在[0.442 857,0.625]上:$\pi^*=(J_1,J_3,J_5,J_2,J_6,J_4)$,在 $x=0.442\ 857$ 且 $F_{\min}=24.571\ 429$;

在[0.625,0.65]上:$\pi^*=(J_3,J_1,J_5,J_2,J_2,J_6)$,在 $x=0.65$ 且 $F_{\min}=25.2$;

在[0.65,0.7]上:$\pi^*=(J_3,J_1,J_5,J_2,J_6,J_4)$,在 $x=0.7$ 且 $F_{\min}=25.1$;

在[0.7,0.75]上:$\pi^*=(J_3,J_1,J_2,J_5,J_6,J_4)$,在 $x=0.75$ 且 $F_{\min}=25$;

在[0.75,0.766 667]上:$\pi^*=(J_3,J_2,J_1,J_5,J_6,J_4)$, 在 $x=0.766\ 667$ 且 $F_{\min}=24.966\ 667$;

在[0.766 667,0.84]上:$\pi^*=(J_3,J_2,J_5,J_1,J_6,J_4)$,在 $x=0.84$ 且 $F_{\min}=24.82$;

在[0.84,1]上:$\pi^*=(J_3,J_2,J_5,J_1,J_6,J_4)$,在 $x=1$,且 $F_{\min}=24.55$。

对于 e_3,得最优排序:$J_6\prec J_5\prec J_4\prec J_2\prec J_3\prec J_1$,最优机器位置:$x=0.3$,最优目标值:24.1。

定理 1　算法 1 能够在 $O(m^2n^6)$时间内求解 RSAB 问题。

证明　算法 1 的正确性由之前的讨论和执行过程来保证,这里只分析时间复杂性。

第 1 步需要 $O(mn^2)$时间。这是因为在每种场景下计算树图上所有节点对之间的最短路径需 $O(n^2)$时间[13],且共有 m 种场景。

在第 2.1 步中,需解 $nm(m-1)/2$ 个方程,每个方程可在常数时间内求解,在每条边上至多可找到 $nm(m-1)/2$ 个一级分割点,共需 $O(nm^2)$时间。

在第 2.2 步中,每对相邻的一级分割点 α 和 α'之间至多有 $n(n-1)/2$ 个二级分割点,寻找任意 2 个一级分割点之间的二级分割点需 $O(n^2)$时间。故确定每条边上所有的二级分割点共需 $O(m^2n^3)$时间,且在每条边至多找到 $O(m^2n^3)$个二级分割点。

在第 2.3 步中,对每对相邻的二级分割点 β 和 β',确定最优排序 π^*需 $O(n\log n)$时间,寻找二者之间的三级分割点(至多有 $n(n-1)/2$ 个)需花费 $O(n^2)$时间。故共需 $O(m^2n^5)$时间且在每条边至多可找到 $O(m^2n^5)$个三级分割点。

第 2.4 步需 $O(m^2n^5)$时间。因为需对每条边进行迭代,所以第 2 步总共需 $O(m^3n^6)$时间。

第 3 步需 $O(m^2n^6)$时间。

综上所述,算法 1 的时间复杂性是 $O(m^2n^6)$。　　证毕

注意到,当 $D^{(k)}=0,\forall k\in\{1,2,\cdots,m\}$时,问题 RSAB 即为本文研究的最大绝对鲁棒问题(P1)。因而,下面的推论成立。

推论 1　问题 P1 可在 $O(m^2n^6)$时间内求解。

又注意到,当 $D^{(k)}=-C^{(k)},\forall k\in\{1,2,\cdots,m\}$,问题 RSAB 即为本文研究的最大遗憾问题(P2)。此时,需提前计算在各个场景 $\xi_1,\xi_2,\cdots,\xi_m$ 下的最优值 $C^{(1)},C^{(2)},\cdots,C^{(m)}$。而当场景固定时,问题 RSAB 退化为树图上最小化最大完工时间的 SchLoc 问题,利用 Hamacher 等人[3]给出的算法可在 $O(n^3)$时间内求解。因而,下面的推论也成立。

推论 2　问题 P2 可在 $O(m^2n^6)$时间内求解。

对于最大相对遗憾问题(P3),可以由下面的引理容易观察出来。

引理 8　对于问题 P3 的任意排序 π 和任意位置 $x\in T$,有:

$$\max_{\xi_k\in S}\{(C^{(k)}_{\max}(\pi,x)-C^{(k)})/C^{(k)}\}\leqslant\lambda,$$

当且仅当 $\max_{\xi_k\in S}\{C^{(k)}_{\max}(\pi,x)-(1+\lambda)C^{(k)}\}\leqslant 0$。其中:$\lambda$ 是给定常数。

注意,对任意的 $\lambda>0$,若令 $D^{(k)}=-(1+\lambda)C^{(k)},\forall k\in\{1,2,\cdots,m\}$,并对问题 RSAB 运行算法 1,则由定理 1 可知,得到的解能够最小化 $\max_{\xi_k\in S}\{C^{(k)}_{\max}(\pi,x)-(1+\lambda)C^{(k)}\}$。从而,结合引理 8,可对问题 P3 设计多项式时间近似方案。主要思想是采用二元搜索策略,逐渐减小可行的 λ 取值,并重复调用算法 1。

为此,首先需建立关于问题 P3 的最优值(即最大相对遗憾的最小值)的上界。明显地,在任意场景 ξ_k 下的最大完工时间的最优值 $C^{(k)}$满足:

$$C^{(k)} \geqslant \min_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x) + \sum_{1\leqslant j\leqslant n} p_j,$$

对任意排序 π 和任意位置 $x\in T$,在场景 ξ_k 下 π 的时间表满足:

$$C_{\max}^{(k)}(\pi,x) \leqslant \max_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x) + \sum_{1\leqslant j\leqslant n} p_j。$$

注意,对于给定的树图和场景 ξ_k,上面2个式子中的 $\min_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x)$ 和 $\max_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x)$(分别简记为 Δ 和 $\Delta^{(k)}$)均为常数,并且实际上前者等于 $\min_{1\leqslant j\leqslant n, x\in T} s_j$,后者等于 $\max_{1\leqslant i,j\leqslant n}\{s_j + d^{(k)}(v_i,v_j)/\tau_j\}$。因而,有:

$$0 \leqslant \frac{C_{\max}^{(k)}(\pi,x) - C^{(k)}}{C^{(k)}} \leqslant \frac{\max_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x) - \min_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x)}{\min_{1\leqslant j\leqslant n, x\in T} r_j^{(k)}(x) + \sum_{1\leqslant j\leqslant n} p_j} = \frac{\Delta^{(k)} - \Delta}{\Delta + \sum_{1\leqslant j\leqslant n} p_j}。$$

进而可推出:

$$0 \leqslant \min_{x\in T,\pi\in\Pi}\left\{\max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi,x) - C^{(k)}}{C^{(k)}}\right\}\right\} \leqslant \max_{\xi_k\in S}\left\{\frac{\Delta^{(k)} - \Delta}{\Delta + \sum_{1\leqslant j\leqslant n} p_j}\right\}。$$

令 ε 为任意给定的常数且 $0<\varepsilon<1$。对问题 P3,可设计如下的多项式时间近似方案。

算法 2 P3 的多项式时间近似方案。

$L := 0$;

$R := \max_{\xi_k\in S}\left\{\frac{\Delta^{(k)} - \Delta}{\Delta + \sum_{1\leqslant j\leqslant n} p_j}\right\}$;

While $R > (1+\varepsilon)L$, do

  $\lambda := (L+R)/2$;

  For $k = 1,2,\cdots,m$, do

    $D^{(k)} = -(1+\lambda)C^{(k)}$;

    运行算法 1 于对应的问题 RASB,得到最优解(π^*, x^*);

    If $\max_{\xi_k\in S}\{C_{\max}^{(k)}(\pi^*,x^*) - (1+\lambda)C^{(k)}\} \leqslant 0$, then

      $R := \lambda$;

    Else

      $L := \lambda$;

Return 最优解(π^*, x^*)

定理 2 算法 2 是求解问题 P3 的多项式时间近似方案,所需时间为 $O(m^2n^6\log(1/\varepsilon))$。

证明 假设算法 2 终止时参数 L 和 R 的值分别为 L^* 和 R^*。由算法 2 的执行过程可知:

$$L^* \leqslant \max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi^*,x^*) - C^{(k)}}{C^{(k)}}\right\} \leqslant R^* \text{ 和 } R^* \leqslant (1+\varepsilon)L^*。$$

注意到,在算法 2 的整个搜索过程中,L 始终是问题 P3 的最优目标值的下界,即:

$$\min_{x\in T,\pi\in\Pi}\left\{\max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi,x) - C^{(k)}}{C^{(k)}}\right\}\right\} \geqslant L^*。$$

因而,可推出:

$$\max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi^*,x^*) - C^{(k)}}{C^{(k)}}\right\} \leqslant R^* \leqslant (1+\varepsilon)L^* \leqslant (1+\varepsilon)\min_{x\in T,\pi\in\Pi}\left\{\max_{\xi_k\in S}\left\{\frac{C_{\max}^{(k)}(\pi,x) - C^{(k)}}{C^{(k)}}\right\}\right\}。$$

这说明(π^*, x^*)是问题 P3 的$(1+\varepsilon)$-近似解。另外,算法 2 至多进行 $O(\log(1/\varepsilon))$次迭代,而每次迭代需调用 1 次算法 1。结合定理 1 可知,算法 2 的时间复杂性是 $O(m^2n^6\log(1/\varepsilon))$。 证毕

3 结论

本文研究了树图上边长度有不确定性的 SchLoc 问题,分别对 P1 和 P2 给出了多项式时间算法,对 P3 设计了多项式时间近似方案,并通过实例说明了所设计算法的执行过程和有效性。在未来研究中,拟考虑以下 2 个问题:

1) 问题 P3 是否是多项式时间可解的?

2) 问题 P1 和问题 P2 的随机形式(即最小化各种场景下最大完工时间和最大遗憾的期望值)是否也是多项式时间可解的?

参考文献:

[1] Elvikis D, Hamacher H W, Kalsch M T. Simultaneous scheduling and location (ScheLoc): the planar ScheLoc makespan problem[J]. Journal of Scheduling, 2009, 12 (4): 361-374.

[2] Shabtay D, Gilenson M. A state-of-the-art survey on multi-scenario scheduling[J]. European Journal of Operational Research, 2023, 310 (1): 3-23.

[3] Hamacher H W, Hennes H. Integrated scheduling and location models: single machine makespan problems: 82 [R]. Kaiserslautern: University of Kaiserslautern, Department of Mathematics, 2002.

[4] Kalsch M T, Drezner Z. Solving scheduling and location problems in the plane simultaneously[J]. Computers & Operations Research, 2010, 37 (2): 256-264.

[5] Krumke S O, Le H M. Robust absolute single machine makespan scheduling-location problem on trees[J]. Operations Research Letters, 2020, 48 (1): 29-32.

[6] Averbakh I. Complexity of robust single facility location problems on networks with uncertain edge lengths[J]. Discrete Applied Mathematics, 2003, 127 (3): 505-522.

[7] Averbakh I, Berman O. Algorithms for the robust 1-center problem on a tree[J]. European Journal of Operational Research, 2000, 123 (2): 292-302.

[8] Bachtler O, Krumke S O, Le H M. Robust single machine makespan scheduling with release date uncertainty[J]. Operations Research Letters, 2020, 48 (6): 816-819.

[9] Geng Z C, Gao Y. Single-machine makespan scheduling with scenario-dependent release dates[J]. Operations Research Letters, 2025, 63: 107353.

[10] Wu W, Tang L, Pizzuti A. Robust scheduling for minimizing maximum lateness on a serial-batch processing machine[J]. Information Processing Letters, 2024, 186: 106473.

[11] Wu W, Hayashi T, Haruyasu K, et al. Exact algorithms based on a constrained shortest path model for robust serial-batch and parallel-batch scheduling problems[J]. European Journal of Operational Research, 2023, 307 (1): 82-102.

[12] Bougeret M, Pessoa A, Poss M. Single machine robust scheduling with budgeted uncertainty[J]. Operations Research Letters, 2023, 51 (2): 137-141.

[13] Diestel R. Graph theory[M]. 6th ed. Heidelberg: Springer-Verlag, 2025.

Operations Research and Cybernetics

Single Machine Scheduling and Location Problem on a Tree Graph with Uncertain Edge Lengths

CHEN Beining, GENG Zhichao

(School of Mathematics and Statistics, Zhengzhou University, Zhengzhou 450001, China)

Abstract: This paper studies the integrated problem of single machine scheduling and location on a tree graph, where the processing times of jobs, transportation starting times, and the transportation speeds of jobs are given; the vertices and edges of the tree graph are also given, but the edge lengths are uncertain and their values depend on a finite set of discrete scenarios. The placement of the machine and the processing sequence of jobs need to be determined by decision-making; jobs need to be transported to the machine location before processing. The scheduling criterion of interest is the maximum completion time (or makespan) of jobs. Using robust optimization methods, polynomial-time algorithms are provided for the maximum absolute robust problem and the maximum regret problem respectively; for the maximum relative regret problem, a fully polynomial-time approximation scheme is designed. Finally, numericals examples are given to illustrate the execution process and effectiveness of the proposed algorithms.

Keywords: scheduling and location problem; multiple scenarios; robust optimization; maximum completion time; tree graph; uncertain edge lengths

(责任编辑 黄 颖)