Maximizing Total Profit in Two-agent Problem of Order Acceptance and Scheduling

Mohammad Reisi-Nafchi[1], Ghasem Moslehi[2], Mehdi Bijari[3]

1 Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran

2 Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran

3 Department of Industrial and Systems Engineering, Isfahan University of Technology, Isfahan, Iran

Abstract:

In competitive markets, attracting potential customers and keeping current customers is a survival condition for each company. So, paying attention to the requests of customers is important and vital. In this paper, the problem of order acceptance and scheduling has been studied, in which two types of customers or agents compete in a single machine environment. The objective is maximizing sum of the total profit of first agent's accepted orders and the total revenue of second agent. Therefore, only the first agent has penalty and its penalty function is lateness and the second agent's orders have a common due date and this agent does not accept any tardy order. To solve the problem, a mathematical programming, a heuristic algorithm and a pseudo-polynomial dynamic programming algorithm are proposed. Computational results confirm the ability of solving all problem instances up to 70 orders size optimally and also 93.12% of problem instances up to 150 orders size by dynamic programming.

Keywords: Single machine; Orders acceptance; two-agent scheduling; Mathematical programming; Dynamic programming

مدیریت تولید و عملیات، دوره۸، پیاپی(۱٤)، شماره (۱)، بهار و تابستان ۱۳۹٦
دریافت: ۱۳۹۲/۰۳/۱۱ پذیرش: ۱۳۹۵/۰۵/۰۷
صص:۱۰۰-۷۹

بیشینه‌سازی سود در مسئلهٔ دوعاملی پذیرش و زمان‌بندی یکپارچهٔ سفارش‌ها

محمد رئیسی نافچی[۱]، قاسم مصلحی[۲]*، مهدی بیجاری[۳]

۱- استادیار، دانشکدهٔ مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان
۲- استاد، دانشکدهٔ مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان
۳- استاد، دانشکدهٔ مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان

چکیده: در بازارهای رقابتی شرط بقای یک سازمان، جذب مشتریان بالقوه و حفظ مشتریان فعلی است؛ بنابراین توجه به نیازها و خواسته‌های مشتریان بسیار مهم است. در این مقاله مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها، در حالتی بررسی شده است که دو نوع مشتری یا عامل در یک محیط تک‌ماشین برای رسیدن به اهداف خود با هم رقابت می‌کنند. هدف بیشینه‌سازی مجموع سود سفارش‌های عامل اول و درآمد سفارش‌های عامل دوم است؛ بنابراین فقط عامل اول جریمه دارد و تابع آن مجموع مغایرت زمان تکمیل و موعد تحویل است. سفارش‌های عامل دوم نیز دارای یک موعد تحویل مشترک بوده و این عامل هیچ سفارش همراه به دیرکرد را نمی‌پذیرد. برای حل مسئله مدلی ریاضی، یک الگوریتم ابتکاری و یک برنامه‌ریزی پویای شبه‌چندجمله‌ای ارائه شده است. نتایج حل این الگوریتم‌ها در مسائل نمونه حاکی از توانایی حل بهینهٔ تمامی مسائل تا ابعاد ۷۰ سفارش و ٪۹۳/۱۲ از مسائل تا ابعاد ۱۵۰ سفارش توسط برنامه‌ریزی پویا است.

واژه‌های کلیدی: تک‌ماشین، پذیرش سفارش، زمان‌بندی دوعاملی، مدل ریاضی، برنامه‌ریزی پویا

نویسندهٔ مسئول: قاسم مصلحی پست الکترونیک: moslehi@cc.iut.ac.ir

۱- مقدمه

امروزه در بازارهای رقابتی، رمز بقای یک تولیدکننده در جذب، حفظ و نگهداری مشتریان نهفته است؛ اما در عمل نیازها و خواسته‌های مشتریان متفاوت است که در صورت تأمین‌نشدن این نیازها در بازار رقابتی، مشتری به رقبا مراجعه می‌کند و در عمل تولیدکننده متضرر خواهد شد. آنچه که در مطالعات انجام شده در پذیرش و زمان‌بندی سفارش‌ها مشاهده می‌شود، مغفول‌ماندن نیازهای متفاوت مشتریان است. سعی شده که در این مقاله این نقیصه با درنظرگرفتن دو دسته مشتری هریک با توابع جریمهٔ متفاوت برطرف شود؛ به‌طوری که مشتریان دستهٔ اول با افزایش دیرکرد، جریمه دریافت می کنند و با تحویل زودتر از موعد حاضر به دادن پاداش به تولیدکننده در قبال دریافت سفارش خود، زودتر از موعد نهایی هستند. از طرف دیگر مشتریان عامل دوم به هیچ عنوان حتی با دریافت جریمه نیز حاضر به تحویل گرفتن سفارش‌های همراه با دیرکرد نیستند. در دههٔ گذشته در ادبیات موضوع رقابت چند دسته کار بر یک محیط ماشینی هریک با اهداف متفاوت، تحت عنوان «زمان‌بندی چندعاملی» مطرح شده که مطالعات در این حوزه نیز رو به افزایش است. بدیهی است که استفاده از این ایده در حوزهٔ پذیرش و زمان‌بندی سفارش‌ها هم با واقعیت تطبیق بیشتری دارد و هم مورد توجه تولیدکنندگان خواهد بود. در ادامه برخی از مهم‌ترین مطالعات صورت‌گرفته در این دو زمینه به اختصار مرور می شود.

اسلاتنیک[۲] (۲۰۱۱) یک مقالهٔ مروری دربارهٔ مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها نوشته است. در این مقاله وی مطالعات صورت‌گرفته را دسته‌بندی کرده و در هر دسته آخرین دستاوردها را بیان کرده است. به‌منظور جلوگیری از طولانی‌شدن بحث، خوانندگان به این مقاله ارجاع داده می‌شوند. اسلاتنیک و مورتون[۳] (۱۹۹٦) مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها را با هدف بیشینه‌سازی سود و درنظرگرفتن مجموع مغایرت وزنی به‌عنوان تابع جریمه بررسی کردند. از آنجا که با فرض معلوم‌بودن مجموعهٔ سفارش‌های پذیرفته‌شده، تعیین توالی بهینهٔ سفارش‌ها براساس ترتیب WSPT[٤] به دست می‌آید، آنان چند خاصیت برای مسئلهٔ اصلی ارائه کرده‌اند. در این مطالعه یک الگوریتم شاخه و کران و دو الگوریتم ابتکاری برای حل مسئله توسعه داده شده و نشان داده شده که شاخه و کران، مسائل نمونه را تا ابعاد ۱۷ سفارش، به‌طور بهینه حل می‌کند. در ادامه قوش[۵](۱۹۹۷) نشان داد که مسئلهٔ فوق NP-hard بود و برای حل آن دو الگوریتم برنامه‌ریزی پویای شبه‌چندجمله‌ای و یک الگوی تقریب کاملاً چندجمله‌ای (FPTAS[٦]) با پیچیدگی $O\left(n^2/\varepsilon\right)$ توسعه دادند.

اسلاتنیک و مورتون (۲۰۰۷) مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها در حالت تک‌ماشین را با درنظرگرفتن جریمهٔ دیرکرد وزنی بررسی کردند. آنان نشان دادند که این مسئله NP-hard بود و از این رو برای حل آن یک رویهٔ شاخه و کران و چند الگوریتم ابتکاری ارائه کردند. رام و اسلاتنیک[۷](۲۰۰۹) نیز یک الگوریتم ژنتیک برای همین مسئله توسعه دادند که جواب‌های با کیفیت بیشتری نسبت به الگوریتم‌های ابتکاری ارائه‌شده در ادبیات موضوع تولید می‌کند. نوبیبون و لئوس[۸](۲۰۱۱) نیز این مسئله را در حالتی تعمیم دادند که تعدادی سفارش از قبل پذیرفته‌شده و تعدادی سفارش در انتظار تصمیم وجود دارد. پس از بررسی پیچیدگی، این نویسندگان دو مدل برنامه‌ریزی خطی عدد صحیح مختلط و دو الگوریتم شاخه و کران

با به‌کارگیری خواص ارائه‌شده توسط اسلاتنیک و مورتون (۲۰۰۷) و رام و اسلاتنیک (۲۰۰۹) ارائه کردند. نتایج محاسباتی آنان عملکرد این رویه‌ها را تحت سناریوهای مختلف مقایسه کرده است.

سیسارت و همکاران (۲۰۱۲) نیز مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها را با درنظرگرفتن ضرب‌الاجل تحویل برای سفارش‌ها و جریمهٔ دیرکرد وزنی برای تأخیر در تحویل در بازهٔ بین موعد تحویل و ضرب‌الاجل تحویل بررسی کردند. در این مطالعه برای هر سفارش زمان ورود و زمان آماده‌سازی وابسته به توالی در نظر گرفته شده است. آنان برای حل این مسئله یک الگوریتم جستجوی ممنوع (TS[9]) توسعه دادند و عملکرد آن را با دو الگوریتم ابتکاری موجود در ادبیات موضوع مقایسه کردند. چن و همکاران[10] (۲۰۱٤) بر مسئله‌ای که سیسارت و همکاران[11] (۲۰۱۲) بررسی کردند، مطالعه انجام دادند و برای حل آن یک الگوریتم ژنتیک ارائه کردند. نتایج محاسباتی آنان حاکی از کیفیت بهتر جواب‌های به‌دست‌آمده در مقایسه با روش‌هایی است که سیسارت و همکاران (۲۰۱۲) ارائه کردند.

به‌طور رسمی در سال ۲۰۰۳، بیکر و اسمیت (۲۰۰۳) مسئلهٔ زمان‌بندی چندعاملی را معرفی کردند. آنان نشان دادند که می‌توان مسئلهٔ $1\|C^1_{\max}+\theta L^2_{\max}$ را در زمان چندجمله‌ای حل کرد. همچنین نشان دادند که مسئلهٔ $1\|C^1_{\max}+\theta\sum w_jC^2_j$ را می‌توان به مسئلهٔ $1\|\sum w_jC^1_j+\theta\sum w_jC^2_j$ تبدیل کرد و ثابت کردند که مسئلهٔ $1\|\sum w_jC^1_j+\theta L^2_{\max}$ از نظر پیچیدگی NP-hard است و برای حل حالت خاص آن یعنی مسئلهٔ $1\|\sum C^1_j+\theta L^2_{\max}$ یک برنامه‌ریزی پویا ارائه کردند. یوان و همکاران (۲۰۰۵) نشان دادند که برنامه‌ریزی‌های پویایی که بیکر و اسمیت (۲۰۰۳) برای مسائل $1\|L^1_{\max}+\theta L^2_{\max}$ و $1\|\sum C^1_j+\theta L^2_{\max}$ ارائه کرده‌اند اشتباه هستند. آنان برای حل این مسائل یک الگوریتم با پیچیدگی چندجمله‌ای ارائه کردند.

اگنتیس و همکاران[12] (۲۰۰٤) به بررسی مسائل زمان‌بندی دوعاملی با توابع هدف بیشینه مقدار یک تابع منظم[13] ($f_{\max}$)، تعداد کارهای همراه با دیرکرد ($\sum U_i$) و مجموع وزنی زمان‌های تکمیل ($\sum w_iC_i$) پرداختند. آنان سناریوهای مختلفی وابسته به تابع هدف هر عامل و نیز محیط کارگاهی، در نظر گرفتند و برای هر سناریو پیچیدگی مسائل مربوط را بررسی کردند. لیونگ و همکاران (۲۰۱۰) نیز در مطالعهٔ خود به بررسی پیچیدگی مسائل زمان‌بندی دوعاملی با استفاده از ترکیبات مختلف توابع هدف $\sum C_i$، $\sum U_i$، $f_{\max}$ و مجموع دیرکرد ($\sum T_i$) در حالت کمینه‌سازی تابع هدف یک عامل و محدودکردن تابع عامل دوم پرداختند. آنان مسائل مختلفی را در حالت تک‌ماشین، ماشین‌های موازی یکسان و مجازبودن یا نبودن انقطاع برای کارهای هر عامل بررسی کردند؛ مثلاً آنان نشان دادند که مسئلهٔ $1\|\sum T^1_i:\sum U^2_i\le Q_2$ به‌طور عادی NP-hard[14] بود و برای هریک از مسائل $1\|\sum T^1_i:\sum C^2_i\le Q_2$ و $1\|\sum T^1_i:f^2_{\max}\le Q_2$ یک الگوریتم شبه‌چندجمله‌ای ارائه کردند.

یین و همکاران (۲۰۱۲) مسئلهٔ بهینه‌سازی مقید کمینه‌سازی مجموع دیرکرد کارهای عامل اول با در نظر گرفتن محدودیت بر مقدار بیشینهٔ مغایرت زمان تکمیل و موعد تحویل عامل دوم در حالت زمان‌های ورود برای هر کار، بررسی کردند. آنان یک مدل برنامه‌ریزی عدد صحیح و یک الگوریتم شاخه و کران برای حل بهینهٔ مسئله ارائه کردند و برای دستیابی به جواب های نزدیک بهینه در مسائل بزرگ، یک

الگوریتم فراابتکاری بهینه‌سازی ازدواج در زنبورهای عسل[۱۵] ارائه کردند. یین و همکاران (۲۰۱۳) مسئلهٔ کمینه‌سازی مجموع وزنی زودکرد تمامی کارها با محدودکردن بیشینه زودکرد کارهای یکی از عوامل را مورد بررسی قرار دادند. آنان نشان دادند که این مسئله به شدت NP-hard[۱۶] است. برای حل آن نیز یک مدل MIP و یک الگوریتم شاخه و کران ارائه کردند. همچنین در این مطالعه یک الگوریتم شبیه‌سازی تبرید(SA[۱۷]) برای حل مسائل در ابعاد بزرگ ارائه شده است.

وو و همکاران (۲۰۱۳) مسئلهٔ دوعاملی کمینه‌سازی مجموع زمان‌های تکمیل عامل اول و محدودکردن مجموع زمان‌های تکمیل عامل دوم با فرض درنظرگرفتن زمان ورود برای هر کار ($1|r_i|\sum C_i^1:\sum C_i^2\le Q_2$) را مطالعه کردند. آنان نشان دادند که این مسئله به شدت NP-hard بود و برای حل بهینهٔ آن یک الگوریتم شاخه و کران ارائه کردند. همچنین در این مطالعه یک الگوریتم اجتماع مورچگان[۱۸] و چهار الگوریتم ژنتیک نیز برای یافتن جواب‌های نزدیک بهینه توسعه داده شده است. ژائو و لو (۲۰۱۳) نشان دادند که مسائل زمان‌بندی دوعاملی $P_m\,||C_{\max}^1:C_{\max}^2\le Q_2$ و $P_m\,||\sum C_i^1:C_{\max}^2\le Q_2$ دارای پیچیدگی NP-hard بود و برای هریک از این مسائل یک الگوریتم برنامه‌ریزی پویای شبه‌چندجمله‌ای ارائه کردند. همچنین بر مبنای این الگوریتم‌ها برای هر مسئله یک FPTAS نیز توسعه دادند. چنگ و همکاران (۲۰۱۳) برای حل مسئلهٔ $1|r_i|\sum C_i^1:L_{\max}^2\le Q_2$ یک الگوریتم شاخه و کران و یک الگوریتم SA توسعه داده و نشان دادند که الگوریتم SA دارای میانگین درصد خطای ۰/۵٪ در کل مسائل نمونه است.

لی و وانگ (۲۰۱٤) مسئلهٔ زمان‌بندی سه‌عاملی کمینه‌سازی مجموع وزنی زمان‌های تکمیل کارهای عامل اول را مشروط به اینکه دامنهٔ عملیات عامل دوم کوچک‌تر از یک مقدار معین باشد و فعالیت نگهداری و تعمیرات عامل سوم در یک بازهٔ مشخص از زمان انجام شود، حل کرده‌اند. آنان یک الگوریتم شاخه و کران برای حل بهینه و یک الگوریتم ژنتیک برای یافتن جواب‌های نزدیک بهینه ارائه کرده‌اند. شاخه و کران ارائه‌شده در این مقاله قادر به حل بهینهٔ مسائل تا ابعاد ۲٤ کار است. ژائو و لو (۲۰۱٤) یک الگوریتم تقریب با حد بدترین خطای ۲ و پیچیدگی $O(n)$ برای مسئلهٔ $P_m\,||C_{\max}^1:C_{\max}^2\le Q_2$ ارائه کردند. همچنین آنها برای حالتی که تعداد ماشین‌های موازی برابر دو ماشین باشد یک الگوریتم چندجمله‌ای با پیچیدگی $O(n\log n)$ توسعه دادند که حد بدترین خطای آن برابر ۱/۲۸ است.

بیشتر مطالعات انجام شده در ادبیات موضوع زمان‌بندی چندعاملی در حالت دوعاملی انجام شده است که به نظر می‌رسد دلیل آن افزایش پیچیدگی مسائل با بیش از دوعامل است و معدود مطالعات بیش از دو عامل نیز در حالت‌های ساده و خاص مسئله بوده است. در این مقاله نیز همان‌طور که گفته شد مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها در حالت دوعاملی بررسی شده است؛ بنابراین نوآوری مقاله در بررسی توأم این دو موضوع و نیز تلاش بر حل بهینهٔ مسئله است. بدیهی است که ایدهٔ ترکیب دو مسئلهٔ پذیرش و زمان‌بندی سفارش‌ها و زمان‌بندی چندعاملی در راستای کاربردی‌کردن هر دو مسئله است و در آینده می‌تواند از حوزه‌های جذاب پژوهشی باشد.

در ادامهٔ مقاله، مسئلهٔ موردنظر و نمادهای موردنیاز در

بخش دوم تعریف شده و سپس در بخش سوم یک مدل ریاضی، یک الگوریتم ابتکاری و یک برنامه‌ریزی پویای شبه‌چندجمله‌ای برای حل مسئله ارائه شده است. در بخش چهارم نیز نتایج حل روش‌های توسعه‌داده‌شده بر مسائل نمونه ارائه و تحلیل شده است. در انتها در بخش پنجم، نتیجه‌گیری و پیشنهادهایی برای مطالعات آتی ذکر شده است.

۲- تعریف مسئله و نمادهای موردنیاز

در این مقاله مسئلهٔ دوعاملی پذیرش و زمان‌بندی سفارش‌ها در محیط تک‌ماشین با هدف بیشینه‌کردن مجموع سود سفارش‌های پذیرفته‌شدهٔ عامل اول و درآمد سفارش‌های پذیرفته‌شدهٔ عامل دوم در حالت مجازنبودن دیرکرد برای سفارش‌ها عامل دوم بررسی می‌شود. تابع جریمهٔ عامل اول نیز مجموع مغایرت زمان تکمیل و موعد تحویل در نظر گرفته شده که معنای آن دریافت پاداش درصورت تحویل زودتر از موعد و پرداخت جریمه درصورت تأخیر در تحویل می‌باشد. برای سفارش‌ها عامل دوم نیز موعد تحویل مشترک در نظر گرفته شده است. همچنین به‌هنگام پردازش هر سفارش پذیرفته‌شده انقطاع مجاز نیست و بیکاری عمدی ماشین نیز توجیهی ندارد. برخی نمادهای استفاده‌شده برای بیان مسئله عبارت‌اند از:

S : مجموعهٔ کل سفارش‌ها

n : تعداد کل سفارش‌ها

n_k : تعداد سفارش‌های متعلق به عامل k

$n_k > 0,\ \sum_{k=1}^{2} n_k = n$

$(k = 1,2)$

S_k : مجموعهٔ سفارش‌های متعلق به عامل k

$(k = 1,2)$ $\quad (\bigcap_{k=1}^{2} S_k = \varnothing$ و $\bigcup_{k=1}^{2} S_k = S)$

J_i^k : سفارش i متعلق به عامل k

$(k = 1,2$ و $i = 1,\ldots,n_k)$

p_i^k : مدت‌زمان پردازش J_i^k

$(k = 1,2$ و $i = 1,\ldots,n_k)$

P_{sum}^1: مجموع زمان پردازش سفارش‌های عامل اول

$(P_{sum}^1 = \sum_{i=1}^{n_1} p_i^1)$

P_{sum}^2: مجموع زمان پردازش سفارش‌های عامل دوم

$(P_{sum}^2 = \sum_{i=1}^{n_2} p_i^2)$

P_{sum}: مجموع زمان پردازش کل سفارش‌ها

$(P_{sum} = \sum_{k=1}^{2} \sum_{i=1}^{n_k} p_i^k)$

q_i^k : درآمد J_i^k $(k = 1,2$ و $i = 1,\ldots,n_k)$

C_i^k : زمان تکمیل J_i^k $(k = 1,2$ و $i = 1,\ldots,n_k)$

$C_i^k(\sigma)$: زمان تکمیل J_i^k در توالی σ

$(k = 1,2$ و $i = 1,\ldots,n_k)$

d_i^k : موعد تحویل J_i^k $(k = 1,2$ و $i = 1,\ldots,n_k)$

d_2 : موعد تحویل مشترک سفارش‌ها عامل دوم

$(\forall i : d_i^2 = d_2)$

L_i^k : مغایرت زمان تکمیل و موعد تحویل J_i^k، که برابر با $C_i^k - d_i^k$ است. $(k = 1,2$ و $i = 1,\ldots,n_k)$

U_i^k : اگر برای J_i^k مقدار L_i^k مثبت باشد، این سفارش همراه با دیرکرد است و مقدار U_i^k برابر ۱ و در غیر این صورت برابر صفر قرار می‌گیرد.

$(k = 1,2$ و $i = 1,\ldots,n_k)$

بر این اساس مسئلهٔ موردبررسی را می‌توان مطابق نمادگذاری سه‌جزئی گراهام و همکاران (۱۹۷۹) به صورت $1|OA, d_i^2 = d_2, \sum U_i^2 = 0 | \sum_{k=1}^{2} \sum_{i=1}^{n_k} q_i^k - \sum_{i=1}^{n_1} L_i^1$ نشان داد. در این نماد OA نشان‌دهندهٔ پذیرش یا رد سفارش‌ها است و موعد تحویل مشترک سفارش‌های عامل دوم برابر d_2 است.

۳- بررسی مسئله

در مسئلهٔ $1|OA,d_i^2=d_2,\sum U_i^2=0|\sum_{k=1}^{2}\sum_{i=1}^{n_k}q_i^k-\sum_{i=1}^{n_1}L_i^1$ با فرض اینکه تعداد سفارش‌های عامل اول صفر در نظر گرفته شود، این مسئلهٔ به فرم ساده‌تر $1|OA,d_i^2=d_2,\sum U_i^2=0|\sum_{i=1}^{n_2}q_i^2$ تبدیل می‌شود که در آن هر سفارش یک درآمد و یک زمان پردازش دارد و سفارش‌های پذیرفته‌شده باید قبل از موعد تحویل مشترک d_2 به اتمام برسند. همان‌طور که مشاهده می‌شود مسئلهٔ فوق، یک مسئلهٔ کوله‌پشتی است که دارای پیچیدگی به‌طور عادی NP-hard است (گری و جانسون (۱۹۷۹)؛ از این رو، می‌توان نتیجه گرفت که پیچیدگی مسئلهٔ $1|OA,d_i^2=d_2,\sum U_i^2=0|\sum_{k=1}^{2}\sum_{i=1}^{n_k}q_i^k-\sum_{i=1}^{n_1}L_i^1$ نیز حداقل به‌طور عادی NP-hard است؛ اما از آنجا که در ادامه برای این مسئله یک برنامه‌ریزی پویای شبه‌چندجمله‌ای ارائه شده است، پیچیدگی دقیق مسئله به‌طور عادی NP-hard است. در رابطه با این مسئله، دو لم زیر ارائه می‌شود:

لم ۱: با فرض معلوم‌بودن مجموعهٔ سفارش‌های پذیرفته‌شدهٔ هر عامل، یک توالی بهینه وجود دارد؛ به طوری که در آن سفارش‌های پذیرفته‌شدهٔ عامل دوم به ترتیب دلخواه پشت سر هم و قبل از d_2 زمان‌بندی شده‌اند.

اثبات: در هر توالی بهینه با شیفت‌دادن سفارش‌های پذیرفته‌شدهٔ عامل دوم به سمت موعد تحویل مشترک d_2 (سمت راست)، زمان تکمیل سفارش‌های پذیرفته‌شدهٔ عامل اول در قبل از موعد تحویل d_2 افزایش نخواهد یافت و زمان تکمیل سفارش‌های عامل اول در بعد از d_2 نیز ثابت خواهد ماند؛ بنابراین مجموع مغایرت عامل اول افزایش ندارد و می‌توان در هر توالی بهینه، سفارش‌های پذیرفته‌شدهٔ عامل دوم را به‌سمت موعد تحویل d_2 شیفت داد. همچنین به‌دلیل موعد تحویل مشترک سفارش‌ها عامل دوم، ترتیب قرارگرفتن آنها در قبل از d_2 می‌تواند به‌طور دلخواه باشد. ■

لم ۲: با فرض معلوم‌بودن مجموعهٔ سفارش‌های پذیرفته‌شدهٔ هر عامل، یک توالی بهینه وجود دارد که در آن سفارش‌های پذیرفته‌شدهٔ عامل اول به ترتیب غیرنزولی از زمان پردازش (SPT)[۹] زمان‌بندی شده‌اند.

اثبات: بدیهی است که سفارش‌های عامل اول در قبل و بعد از بلوک سفارش‌های عامل دوم به‌تنهایی از ترتیب SPT پیروی می‌کنند؛ بنابراین کافی است ثابت شود که این ترتیب به‌طور سراسری و بین کلیهٔ سفارش‌های قبل و بعد از بلوک عامل دوم نیز برقرار است؛ از این رو فرض کنید که در توالی بهینهٔ σ، ترتیب SPT بین سفارش‌های عامل اول رعایت نشده باشد. بدیهی است که توالی بهینه امکان‌پذیر است. حال فرض می‌شود که در توالی σ دو سفارش دلخواه که ترتیب SPT سفارش‌های عامل اول را بر هم زده‌اند مانند i و j ($p_i^1<p_j^1$) انتخاب شده و با جابه‌جایی آنها توالی جدیدی با نام σ' ایجاد شود به‌طوری که سفارش i قبل از سفارش j قرار گیرد. این دو توالی مطابق لم ۱ در شکل ۱ نشان داده شده‌اند. در این توالی‌ها، σ_k^1 ($k=1,2,3,4$) بلوک‌هایی از سفارش‌های عامل اول و σ^2 بلوک سفارش‌های عامل دوم است.

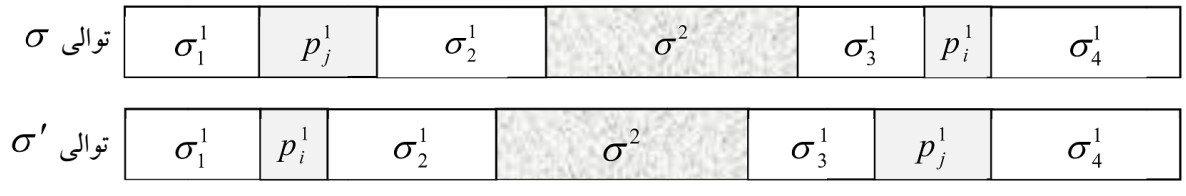
مطابق شکل ۱، زمان تکمیل سفارش‌ها درون بلوک

σ_1^1 و σ_4^1 در هر دو توالی یکسان است. برای هر سفارش h درون بلوک های σ_2^1 و σ_3^1، رابطهٔ $C_h^1(\sigma') \le C_h^1(\sigma)$ و نیز رابطه $C_i^1(\sigma)=C_j^1(\sigma')$ و $C_i^1(\sigma')<C_j^1(\sigma)$ برقرار است؛ بنابراین مجموع مغایرت سفارش های پذیرفته‌شدهٔ عامل اول در توالی σ' کوچک‌تر یا مساوی مجموع مغایرت این سفارش‌ها در توالی σ است. همچنین از آنجا که زمان تکمیل سفارش‌های درون بلوک σ^2 نیز افزایش نداشته، این توالی امکان‌پذیر است و می‌توان نتیجه گرفت که توالی σ' نیز یک توالی بهینه است. همین رویه را برای سایر سفارش‌هایی که ترتیب SPT را بر هم زده‌اند می‌توان ادامه داد تا ترتیب SPT برای سفارش‌های عامل اول برقرار شود و از آنجا که مقدار مجموع درآمد سفارش‌های پذیرفته شدهٔ هر دو توالی یکسان است، اثبات کامل است. ■

نتیجهٔ ۱: بر اساس لم‌های فوق می‌توان فرمت کلی جواب بهینه را به‌صورت یک توالی با سه بلوک در نظر گرفت که بلوک وسط از سفارش‌های پذیرفته‌شدهٔ عامل دوم و بلوک‌های اول و سوم از سفارش‌های پذیرفته‌شدهٔ عامل اول تشکیل شده‌اند که ترتیب کلی سفارش‌های عامل اول از ترتیب SPT پیروی می‌کند.

در زیر یک حد بالا برای مسئله ارائه می‌شود؛ در حالتی که تعدادی از سفارش‌های عامل دوم از قبل پذیرفته یا رد شده‌اند. در این حد بالا ترتیبی از سفارش‌های مجازی با نام V_1 بهره گرفته شده که با استفاده از سفارش‌های عامل اول ایجاد می‌شود؛ به‌طوری که زمان پردازش، موعد تحویل و درآمد سفارش‌های عامل اول به‌ترتیب به‌طور غیرنزولی، غیرصعودی و غیرصعودی به سفارش‌های مجازی تخصیص یابد. همین ترتیب برای سفارش‌های تعیین‌تکلیف‌نشدهٔ عامل دوم با نام V_2 نیز استفاده شده است. همچنین برای نشان‌دادن تعداد اعضای یک مجموعه مانند A نماد $|A|$ به کار رفته است.

لم ۳ (حد بالا): فرض می‌شود که مجموعهٔ سفارش‌های پذیرفته‌شده و ردشدهٔ عامل دوم به‌ترتیب A_2 و R_2 باشد و مقداری مانند h ($0 \le h \le |V_2|$) وجود داشته باشد؛ به‌طوری که رابطه $\sum_{i \le h, i \in V_2} p_i^2 + \sum_{i \in A_2} p_i^2 \le d_2$ و $\sum_{i \le h+1, i \in V_2} p_i^2 + \sum_{i \in A_2} p_i^2 > d_2$ برقرار باشند. همچنین می‌توان سفارش‌های درون A_2 را بلافاصله قبل از d_2 به ترتیب دلخواه قرار داد و از ابتدای ترتیب V_1 سفارش‌های مجازی را تا زمانی که قبل از $d_2 - \sum_{i \in A_2} p_i^2$ جا شده و مقدار سود آنها مثبت است، زمان‌بندی کرد.



شکل ۱. توالی‌های σ و σ' در اثبات لم ۲

سپس سفارش‌های عامل دوم را تا حد امکان به سمت چپ شیفت داد و مابقی سفارش‌های ترتیب V_1 را بعد از بلوک عامل دوم تا زمانی که سود آنها مثبت است، زمان‌بندی کرد. اگر در ترتیب حاصل، مجموع سود حاصل از سفارش‌های مجازی عامل اول به‌مقدار $\sum_{i\le h, i\in V_2} q_i^2 + \sum_{i\in A_2} q_i^2$ افزوده شود، یک حد بالا برای مسئله به دست آمده است.

اثبات: مقدار $\sum_{i\in A_2} q_i^2$ برابر با مجموع درآمد سفارش‌های پذیرفته‌شدهٔ عامل دوم است. در صورتی که قرار باشد فضای باقیمانده در قبل از d_2 تنها با سفارش‌های تعیین‌تکلیف‌نشدهٔ عامل دوم پر شود، قراردادن سفارش‌ها در این فضا، از ابتدای ترتیب V_2 به‌دلیل اینکه با افزایش مدت‌زمان پردازش، درآمد هر سفارش نیز کاهش می‌یابد، می‌تواند یک حد بالا برای این حالت ایجاد کند که مقدار آن برابر $\sum_{i\le h, i\in V_2} q_i^2 + \sum_{i\in A_2} q_i^2$ است. حال فرض می‌شود که فضای قبل از بلوک عامل دوم و نیز بعد از آن با سفارش‌های عامل اول پر باشد. بدیهی است با ورود سفارش‌های ترتیب V_1، در صورتی که سود یک سفارش مجازی مثبت نشود به‌دلیل افزایش زمان پردازش سفارش بعدی در ترتیب V_1 و کاهش موعد تحویل آن، مقدار مغایرت آن کاهش می‌یابد و چون درآمد آن نیز کم می‌شود، سود این سفارش و سفارش‌های بعدی نیز مثبت نخواهد شد. همچنین سفارش‌های پذیرفته‌شده از ترتیب V_1 نیز به‌دلیل تخصیص غیرصعودی درآمد و موعد تحویل به ترتیب غیرنزولی از زمان پردازش، در بهترین حالت ممکن از نظر مجموع سود قرار دارند؛ بنابراین با افزودن مجموع سود این سفارش ها به‌مقدار $\sum_{i\le h, i\in V_2} q_i^2 + \sum_{i\in A_2} q_i^2$ یک حد بالا برای مسئله حاصل می‌شود. ■

نتیجهٔ ۲: در لم ۳ در صورتی که کلیهٔ سفارش‌های عامل دوم و تعدادی از سفارش‌های عامل اول، تعیین‌تکلیف شده باشد، حد بالای مسئله را می‌توان با تشکیل V_1 برای سفارش‌های تعیین‌تکلیف‌نشدهٔ عامل اول و زمان‌بندی آنها مطابق نتیجهٔ ۱، در کنار سفارش‌های پذیرفته‌شدهٔ هر دو عامل و افزودن سود سفارش‌های مجازی زمان‌بندی‌شده به‌مقدار تابع هدف حاصل از سفارش‌های پذیرفته‌شدهٔ هر دو عامل به دست آورد.

۳-۱- مدل برنامه‌ریزی ریاضی M1

براساس نتیجهٔ ۱ برای مسئلهٔ مورد بررسی می‌توان یک مدل ریاضی عدد صحیح مختلط ارائه کرد. متغیرهای تصمیم زیر برای نوشتن مدل تعریف شده است:

X_{i1} : اگر J_i^1 پذیرفته شود و قبل از d_2 به اتمام برسد، مقدار ۱ و در غیر این صورت مقدار صفر می‌گیرد. $(i=1,...,n_1)$

X_{i2} : اگر J_i^1 پذیرفته شود و بعد از d_2 به اتمام برسد، مقدار ۱ و در غیر این صورت مقدار صفر می‌گیرد. $(i=1,...,n_1)$

Y_i : اگر J_i^2 پذیرفته شود مقدار ۱ و در غیر این صورت مقدار صفر می‌گیرد. $(i=1,...,n_2)$

C_{i1} : برابر زمان تکمیل سفارش پذیرفته‌شدهٔ J_i^1 است، اگر این سفارش قبل از d_2 به اتمام برسد و در صورت پذیرفته‌نشدن J_i^1 برابر صفر است. $(i=1,...,n_1)$

C_{i2} : برابر زمان تکمیل سفارش پذیرفته‌شدهٔ J_i^1 است؛ اگر این سفارش بعد از d_2 به اتمام برسد و در صورت پذیرفته‌نشدن J_i^1 برابر صفر است.

$(i = 1,...,n_1)$

با استفاده از متغیرهای فوق مدل ریاضی برنامه‌ریزی خطی مختلط عدد صحیح زیر با نام M1 برای مسئله ارائه می‌شود. مطابق لم ۲ در این مدل فرض شده که سفارش‌های عامل اول به ترتیب SPT شماره‌گذاری شده‌اند.

$$Max \quad \sum_{i=1}^{n_1}(q_i^1(X_{i1}+X_{i2})-(C_{i1}+C_{i2}) \qquad (۱)$$
$$-d_i^1(X_{i1}+X_{i2}))+\sum_{i=1}^{n_2}q_i^2Y_i$$

$$s.t.$$

$$X_{i1}+X_{i2}\le 1 \qquad i=1,2,...,n_1 \qquad (۲)$$

$$\sum_{h=1}^{i}p_h^1X_{h1}\le C_{i1}+M_1(1-X_{i1})$$
$$i=1,2,...,n_1 \qquad (۳)$$

$$\sum_{h=1}^{n_1}p_h^1X_{h1}+\sum_{h=1}^{n_2}p_h^2Y_h+\sum_{h=1}^{i}p_h^1X_{h2}\le \qquad (٤)$$
$$C_{i2}+M_2(1-X_{i2}) \qquad i=1,2,...,n_1$$

$$\sum_{i=1}^{n_1}p_i^1X_{i1}+\sum_{i=1}^{n_2}p_i^2Y_i\le d_2 \qquad (۵)$$

$$C_{i1},C_{i2}\ge 0 \qquad i=1,2,...,n_1 \qquad (٦)$$

$$X_{i1},X_{i2},Y_h\in\{0,1\}$$
$$i=1,2,...,n_1, h=1,2,...,n_2 \qquad (۷)$$

در مدل فوق رابطهٔ (۱) تابع هدف مجموع سود سفارش‌های پذیرفته‌شدهٔ عامل اول و درآمد سفارش‌های پذیرفته‌شدهٔ عامل دوم را نشان می‌دهد. محدودیت (۲) برای جلوگیری از قرارگرفتن یک سفارش از عامل اول به‌طور هم‌زمان در قبل و بعد از d_2 نوشته شده است. در محدودیت‌های (۳) و (٤) زمان تکمیل سفارش‌های عامل اول محاسبه شده که در آنها M_1 و M_2 دو عدد بزرگ به‌ترتیب با حداقل مقادیر $\min\{d_2,\sum_{i=1}^{n_1}p_i^1\}$ و $\sum_{k=1}^{2}\sum_{i=1}^{n_k}p_i^k$ هستند. محدودیت (۵) نیز برای جلوگیری از دیرکردداشتن سفارش‌های پذیرفته‌شدهٔ عامل دوم و نیز جلوگیری از بیشترشدن زمان تکمیل سفارش‌های پذیرفته‌شده از عامل اول که قبل از بلوک عامل دوم قرار دارند، از مقدار d_2 نوشته شده است. محدودیت (٦) ضامن مثبت‌بودن متغیرهای مربوط به زمان تکمیل سفارش‌های عامل اول و بالاخره رابطهٔ (۷) نشان‌دهندهٔ صفرویک‌بودن سایر متغیرهای مسئله است.

مدل فوق دارای $2n_1+n_2$ متغیر صفرویک، $2n_1$ متغیر پیوسته و $3n_1+1$ محدودیت است. بر این اساس تعداد متغیرها و محدودیت‌های این مدل تابعی خطی و درجه یک از تعداد سفارش‌های هر عامل است.

۳-۲- الگوریتم ابتکاری GAO

در این قسمت یک الگوریتم ابتکاری حریصانه با نام GAO، برای حل مسئله ارائه می‌شود. در این الگوریتم کلیهٔ سفارش‌ها به ترتیب غیرصعودی مقدار $\left(q_i^k+\left(d_i^k-p_i^k\right)\right)/p_i^k$ مرتب شده و به همین ترتیب نیز به توالی جواب افزوده می‌شوند. در صورتی که مقدار تابع هدف افزایش نیابد سفارش واردشده از توالی حذف می‌شود. درنهایت توالی جواب به‌عنوان خروجی ارائه می‌شود. گام‌های این الگوریتم به‌قرار زیر است:

گام ۱: سفارش‌ها را به ترتیب غیرصعودی از مقدار $\left(q_i^k+\left(d_i^k-p_i^k\right)\right)/p_i^k$ مرتب کنید و این ترتیب را $RSPR^{\cdot}$ بنامید. توالی جواب را σ بنامید و قرار دهید $\sigma=\phi$، $j=1$ و $obj=0$.

گام ۲: اگر $j>n$ به گام ۸ بروید و در غیر این

صورت سفارش j ام از ترتیب $RSPR$ را O_j نامیده و اگر این سفارش متعلق به عامل اول است به گام ۳ بروید و در غیر این صورت به گام ٤ بروید.

گام ۳: سفارش O_j را در توالی σ طوری قرار دهید که ترتیب SPT سفارش‌های عامل اول در این توالی حفظ شود؛ سپس به گام ۵ بروید.

گام ٤: اگر مجموع زمان پردازش سفارش‌های عامل دوم در توالی σ به علاوهٔ زمان پردازش سفارش O_j بزرگ‌تر از d_2 است به گام ۷ بروید و در غیر این صورت سفارش O_j را در توالی σ بعد از آخرین سفارش از عامل دوم قرار دهید و به گام ۵ بروید.

گام ۵: اگر زمان تکمیل آخرین سفارش از عامل دوم در توالی σ بزرگ‌تر از d_2 است؛ تا زمانی که این مقدار کوچک‌تر یا مساوی d_2 شود، آخرین سفارش از عامل اول در قبل از بلوک سفارش‌های عامل دوم را به بعد از این بلوک منتقل کنید.

گام ٦: اگر مقدار تابع هدف توالی σ کوچک‌تر یا مساوی مقدار obj است، سفارش O_j را از توالی σ حذف کنید و در غیر این صورت مقدار obj را برابر با مقدار تابع هدف توالی σ قرار دهید.

گام ۷: قرار دهید $j = j + 1$ و به گام ۲ بروید.

گام ۸: توالی σ را به‌عنوان جواب ارائه دهید.

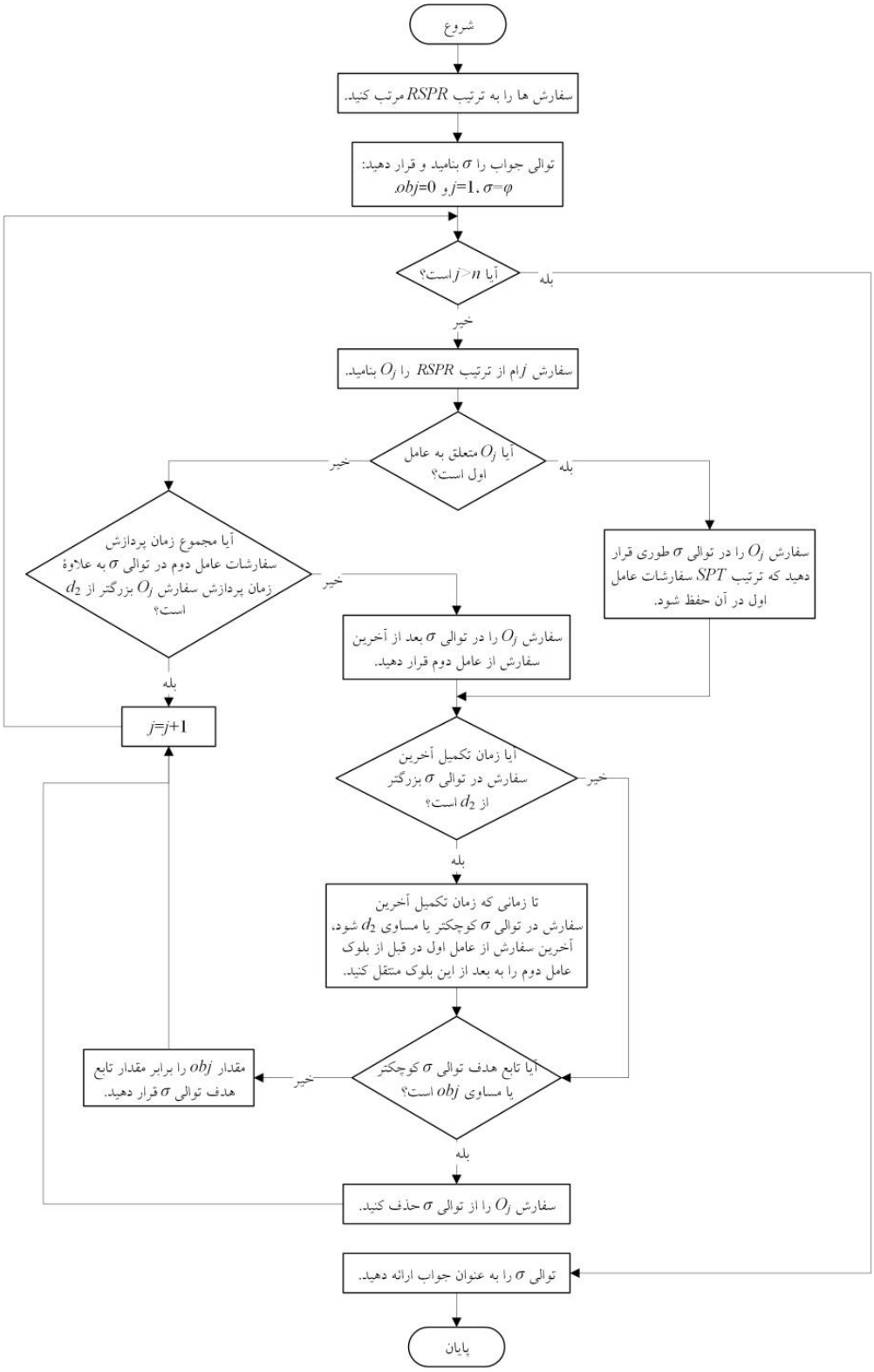
گام ۹: پایان.

در شکل ۲ نیز فلوچارت الگوریتم فوق نشان داده شده است.

پیچیدگی گام ۱ الگوریتم GAO به‌دلیل مرتب‌کردن سفارش‌های هر عامل $O(n \log n)$ است. گام‌های ۳ تا ٦ نیز با پیچیدگی $O(n)$ به تعداد n بار تکرار می‌شود؛ از این رو پیچیدگی الگوریتم GAO $O(n^2)$ است.

۳-۳- الگوریتم برنامه‌ریزی پویای DP1

در این قسمت یک الگوریتم برنامه‌ریزی پویا با نام DP1 و پیچیدگی شبه‌چندجمله‌ای برای حل مسئله توسعه داده شده است. این الگوریتم دارای دو فاز است که در فاز اول، سفارش‌های عامل دوم به ترتیب دلخواه وارد می شود و پس از تولید حالت‌های جدید با به‌کارگیری حدود بالا و پایین و یک اصل غلبه، فضای حالت شماره ۱ به‌روز می‌شود و این کار تا اتمام سفارش‌های عامل دوم ادامه می‌یابد. در فاز دوم به‌ترتیب هریک از حالت‌های درون فضای حالت شماره ۱ در فضای جدیدی به‌نام فضای حالت شماره ۲ قرار گرفته و واردکردن سفارش‌های عامل اول آغاز می‌شود. حالت‌های جدید، تولید می‌شود و پس از مقایسهٔ حدود بالا و پایین و به‌روزرسانی فضای حالت شماره ۲ این روند تا اتمام ورود سفارش‌های عامل اول ادامه می‌یابد. سپس بهترین جواب به‌دست‌آمده تاکنون از بین حالت‌های نهایی فضای حالت شماره ۲ به‌روز شده و فضای حالت شماره ۲ تهی می‌شود. حال این رویه با ورود حالت بعدی از فضای حالت شماره ۱ به فضای حالت شماره ۲ ادامه می‌یابد تا درنهایت جواب بهینه حاصل شود.



شکل ۲. فلوچارت الگوریتم GAO

با توجه به نتیجهٔ ۱، نحوهٔ نمایش حالت‌ها در فضای حالت در این الگوریتم به‌صورت $\upsilon=\left(Z,p^1_{sum\_before},p^1_{sum\_after},p^2_{sum},L^1_{sum}\right)$ در نظر گرفته شده است که در آن Z مقدار تابع هدف، $p^1_{sum\_before}$ و $p^1_{sum\_after}$ به مجموع زمان پردازش سفارش‌های پذیرفته‌شدهٔ عامل اول با زمان تکمیل به‌ترتیب قبل و بعد از d_2، p^2_{sum} مجموع زمان پردازش سفارش‌های پذیرفته‌شدهٔ عامل دوم و L^1_{sum} مقدار مجموع مغایرت سفارش‌های پذیرفته‌شدهٔ عامل اول است. هر حالت در فاز اول در صورتی در فضای حالت قرار داده می شود که هیچ‌کدام از سفارش‌های پذیرفته‌شدهٔ عامل دوم در آن دیرکرد نداشته باشند. گام‌های الگوریتم DP1 به‌شرح زیر است:

گام ۱: سفارش‌های عامل اول را به ترتیب SPT در مجموعهٔ S_1 قرار دهید.
سفارش‌های عامل دوم را به ترتیب دلخواه در مجموعهٔ S_2 قرار دهید.
حالت $(0,0,0,0,0)$ را در فضای حالت شمارهٔ ۱ قرار دهید.
بهترین تابع هدف فعلی را برابر با مقدار تابع هدف خروجی الگوریتم ابتکاری GAO در نظر بگیرید.

گام ۲: در صورتی که $S_2=\phi$ به گام ۳ بروید و در غیر این صورت سفارش ابتدای S_2 را j بنامید و آن را از S_2 حذف کنید و به گام ۲-۱ بروید.

گام ۲-۱: برای تمامی حالت‌های موجود در فضای حالت شماره ۱ مانند $\left(Z,0,0,p^2_{sum},0\right)$، مقدار $\delta=d_2-p^2_{sum}$ را محاسبه کنید. اگر $\delta\geq p^2_j$ است، حالت جدید $\left(Z+q^2_j,0,0,p^2_{sum}+p^2_j,0\right)$ را ایجاد کنید. در پایان به گام ۲-۲ بروید.

گام ۲-۲: (محاسبهٔ حد بالا) برای هریک از حالت‌های جدید ایجادشده در گام ۲-۱، حد بالا را مطابق لم ۳ محاسبه کنید و اگر این حد از بهترین تابع هدف فعلی بزرگ‌تر است آن را به فضای حالت شمارهٔ ۱ بیفزایید و در غیر این صورت این حالت را نادیده بگیرید. اگر حالت جدیدی به فضای حالت شمارهٔ ۱ اضافه شده به گام ۲-۳ و در غیر این صورت به گام ۲ بروید.

گام ۲-۳: (اصل غلبهٔ ۱) برای هر دو حالت $\upsilon=\left(Z,0,0,p^2_{sum},0\right)$ و $\upsilon'=\left(Z',0,0,p^2_{sum}{}',0\right)$ در فضای حالت شمارهٔ ۱، در صورتی که روابط $Z\geq Z'$ و $p^2_{sum}\leq p^2_{sum}{}'$ برقرار باشد، کافی است تنها حالت υ در فضای حالت شمارهٔ ۱ نگه داشته شود. این شرایط را بر فضای حالت شمارهٔ ۱ اعمال کنید و پس از به‌روزرسانی آن به گام ۲ بروید.

گام ۳: تعداد کل حالت‌های موجود در فضای حالت شمارهٔ ۱ را برابر $NS1$ در نظر بگیرید و قرار دهید: $k=1$.

گام ٤: اگر $k>NS1$ به گام ۷ بروید و در غیر این صورت حالت kام از فضای حالت شمارهٔ ۱ را در یک فضای جدید با نام فضای حالت شمارهٔ ۲ وارد کنید، قرار دهید: $S_1'=S_1$ و به گام ۵ بروید.

گام ۵: در صورتی که $S_1' = \phi$ به گام ٦ بروید؛ در غیر این صورت سفارش ابتدای S_1' را j بنامید و آن را از S_1' حذف کنید و به گام ۵-۱ بروید.

گام ۵-۱: برای هریک از حالت‌های درون فضای حالت شمارهٔ ۲ مانند $\left(Z, p^1_{sum\_before}, p^1_{sum\_after}, p^2_{sum}, L^1_{sum}\right)$، مقدار $\delta = d_2 - p^2_{sum} - p^1_{sum\_before}$ را محاسبه کنید و اگر $\delta \geq p^1_j$، حالت $\begin{pmatrix} Z + q^1_j, p^1_{sum\_before} + p^1_j, p^1_{sum\_after} \\ , p^2_{sum}, L^1_{sum} + \left(p^1_{sum\_before} + p^1_j - d^1_j\right) \end{pmatrix}$ و در غیر این صورت حالت $\begin{pmatrix} Z + q^1_j, p^1_{sum\_before}, p^1_{sum\_after} + p^1_j, \\ p^2_{sum}, L^1_{sum} + \left(p^1_{sum\_before} + p^2_{sum} + p^1_j - d^1_j\right) \end{pmatrix}$ را ایجاد کنید و در پایان به گام ۵-۲ بروید.

گام ۵-۲: **(محاسبهٔ حد بالا)** برای هریک از حالت‌های جدید ایجادشده در گام ۵-۱، حد بالا را مطابق نتیجهٔ ۲ محاسبه کنید و اگر این حد از بهترین تابع هدف فعلی بزرگ‌تر است آن را به فضای حالت شمارهٔ ۲ بیفزایید و در غیر این صورت این حالت را نادیده بگیرید. اگر حالت جدیدی به فضای حالت شمارهٔ ۲ اضافه شده به گام ۵-۳ و در غیر این صورت به گام ۵ بروید.

گام ۵-۳: **(اصل غلبهٔ ۲)** برای هر دو حالت $\upsilon = \left(Z, p^1_{sum\_before}, p^1_{sum\_after}, p^2_{sum}, L^1_{sum}\right)$ و $\upsilon' = \left(Z', p^1_{sum\_before}{}', p^1_{sum\_after}{}', p^2_{sum}, L^1_{sum}{}'\right)$ در صورتی که رابطه‌های $Z \geq Z'$، $L^1_{sum} \leq L^1_{sum}{}'$ و $p^1_{sum\_before} + p^1_{sum\_after} \leq p^1_{sum\_before}{}' + p^1_{sum\_after}{}'$ برقرار باشد کافی است تنها حالت υ در فضای حالت شمارهٔ ۲ نگه داشته شود. این شرایط را بر فضای حالت شمارهٔ ۲ اعمال کنید و پس از به‌روزرسانی آن به گام ۵ بروید.

گام ٦: در صورتی که بهترین تابع هدف حالت‌های موجود در فضای حالت شمارهٔ ۲ بزرگ‌تر از بهترین تابع هدف فعلی است آن را جایگزین بهترین تابع هدف فعلی کنید. قرار دهید: $k = k + 1$، فضای حالت شمارهٔ ۲ را تهی کنید و به گام ٤ بروید.

گام ۷: بهترین تابع هدف فعلی را به‌عنوان جواب ارائه کنید.

گام ۸: پایان.

در ادامه اصول غلبهٔ استفاده‌شده در گام‌های ۲-۳ و ۵-۳ الگوریتم DP1 در قالب دو لم اثبات می‌شود. این اصول کارآیی الگوریتم را بالا برده و منجر به دستیابی به پیچیدگی شبه‌چندجمله‌ای برای آن شده است.

لم ٤ (اصل غلبهٔ ۱): در گام ۲-۲ الگوریتم DP1، در صورتی که دو حالت $\upsilon = \left(Z, 0, 0, p^2_{sum}, 0\right)$ و $\upsilon' = \left(Z', 0, 0, p^2_{sum}{}', 0\right)$ با شرایط زیر وجود داشته باشند، می‌توان بدون ازدست‌دادن جواب بهینه حالت υ' را از فضای حالت حذف کرد.

$$Z \geq Z' \qquad (۸)$$

$$p^2_{sum} \leq p^2_{sum}{}' \qquad (۹)$$

اثبات: فرض کنید با حذف حالت υ' از فضای حالت جواب بهینه از دست می‌رود، بدین معنی که این حالت منجر به رسیدن به جواب بهینه می‌شود. از طرفی از

آنجا که در الگوریتم DP1، هر دو حالت فوق در فاز اول یعنی زمانی بررسی می‌شوند که تنها سفارش‌های عامل دوم وارد شده‌اند؛ بنابراین برای رسیدن به جواب بهینه در گام‌های بعدی الگوریتم هم می‌توان سفارش‌های باقیماندهٔ عامل دوم و نیز سفارش‌های عامل اول را وارد کرد. در این صورت فرض کنید که مجموعهٔ سفارش‌های عامل اول و دوم که باید به حالت υ' افزود تا جواب بهینه به دست آید به‌ترتیب OS^1 و OS^2 باشند. همچنین فرض می‌شود که مجموعهٔ OS^1 به دو مجموعه، شامل سفارش‌های عامل اول با زمان تکمیل قبل و بعد از d_2 به‌ترتیب با نام‌های OS^1_{before} و OS^1_{after} افزار شود. ($OS^1 = OS^1_{before} \cup OS^1_{after}$).

از آنجا که جواب بهینه (که مطابق فرض اولیه از حالت υ' به دست آمده)، یک جواب امکان‌پذیر است، داریم:

$$\sum_{i\in OS^1_{before}} p_i^1 + \sum_{i\in OS^2} p_i^1 + p_{sum}^{2\,\prime} \le d_2 \qquad (۸)$$

فرض کنید که $\sum_{i\in OS^1_{after}} C_i^{1\prime}$ و $\sum_{i\in OS^1_{before}} C_i^{1\prime}$ در توالی بهینهٔ به‌دست‌آمده از حالت υ'، برابر مجموع زمان تکمیل سفارش‌های OS^1_{before} و OS^1_{after} باشند. حال اگر سفارش‌های درون OS^1_{before} و OS^1_{after} به همین ترتیب به مجموعهٔ سفارش‌ها قبل و بعد از بلوک عامل دوم در حالت υ افزوده شوند و سفارش‌های مجموعهٔ OS^2 نیز به بلوک عامل دوم در این حالت افزوده شوند، از آنجا که رابطهٔ (۹) برقرار است، در آن صورت روابط $\sum_{i\in OS^1_{before}} C_i^1 = \sum_{i\in OS^1_{before}} C_i^{1\prime}$ و $\sum_{i\in OS^1_{after}} C_i^1 \le \sum_{i\in OS^1_{after}} C_i^{1\prime}$ نیز برقرار است؛ بنابراین داریم:

$$\left(\sum_{i\in OS^1_{before}} C_i^1 + \sum_{i\in OS^1_{after}} C_i^1\right) - \sum_{i\in OS^1} d_i^1 \le \left(\sum_{i\in OS^1_{before}} C_i^{1\prime} + \sum_{i\in OS^1_{after}} C_i^{1\prime}\right) - \sum_{i\in OS^1} d_i^1 \qquad (۹)$$

از طرفی باز به‌دلیل برقراری رابطهٔ (۹) خواهیم داشت:

$$\sum_{i\in OS^1_{before}} p_i^1 + \sum_{i\in OS^2} p_i^1 + p_{sum}^2 \le \sum_{i\in OS^1_{before}} p_i^1 + \sum_{i\in OS^2} p_i^1 + p_{sum}^{2\,\prime} \qquad (۱۰)$$

با ترکیب روابط (۸) و (۱۰) رابطهٔ زیر به دست می‌آید که درحقیقت بیانگر امکان‌پذیری جواب به‌دست‌آمده از حالت υ با افزودن OS^1_{before}، OS^1_{after} و OS^2 به آن، با ترتیب گفته‌شده در بالا، است.

$$\sum_{i\in OS^1_{before}} p_i^1 + \sum_{i\in OS^2} p_i^1 + p_{sum}^2 \le d_2 \qquad (۱۱)$$

همچنین با توجه به برقراری رابطهٔ (۸) می‌توان نوشت:

$$Z + \sum_{i\in OS^1} q_i^1 + \sum_{i\in OS^2} q_i^2 \ge Z' + \sum_{i\in OS^1} q_i^1 + \sum_{i\in OS^2} q_i^2 \qquad (۱۲)$$

با ترکیب روابط (۹) و (۱۲) این نتیجه به دست می‌آید که مقدار تابع هدف جواب به‌دست‌آمده از حالت υ کمتر از حالت υ' نیست و هر دو جواب نیز امکان‌پذیر هستند، فرض بهینه‌بودن جواب به‌دست‌آمده از حالت υ' نقض می‌شود؛ از این رو این فرض باطل است و حذف حالت υ' منجر به ازدست‌رفتن جواب بهینه نمی‌شود. ■

لم ۵ (اصل غلبهٔ ۲): در گام ۸ الگوریتم DP1، در صورتی که دو حالت $\upsilon = \left(Z, p^1_{sum\_before}, p^1_{sum\_after}, p^2_{sum}, L^1_{sum}\right)$ و $\upsilon' = \left(Z', p^{1\,\prime}_{sum\_before}, p^{1\,\prime}_{sum\_after}, p^2_{sum}, L^{1\,\prime}_{sum}\right)$ با

شـــرایط زیر وجود داشـــته باشـــند می‌توان بدون ازدست‌دادن جواب بهینه حالت υ' را از فضای حالت شمارهٔ ۲ حذف کرد.

$$Z \geq Z' \qquad (۱۵)$$

$$p^1_{sum\_before} + p^1_{sum\_after} \leq p^1_{sum\_before}{}' + p^1_{sum\_after}{}' \qquad (۱۶)$$

$$L^1_{sum} \leq L^1_{sum}{}' \qquad (۱۷)$$

اثبات: مشابه اثبات لم ٤ است. ■

چون در فاز اول سفارش‌های عامل دوم وارد شده و این سفارش‌ها نیز دارای موعد تحویل مشترک هستند گویی یک مسـئلهٔ کوله‌پشـتی طرح شـده که در آن هدف، پرکردن فضای قبل از d_2 با سفارش‌های عامل دوم است؛ به‌طوری که درآمد مربوطه حداکثر شود؛ از این رو با اعمال اصـل غلبهٔ ۱ در گام ۲-۲، پیچیدگی این قسـمت از الگوریتم با احتسـاب پیچیدگی حد بـالای ارائـه شـده، بـرابـر $O\left(n_2\left(n_1\log n_1 + n_2\log n_2\right)\min\left\{d_2, P^2_{sum}\right\}\right)$ خواهد بود. همچنین حداکثر تعداد حالت‌های درون فضـای حالت شـمارهٔ ۱ در پایان این فاز برابر $\min\left\{d_2, P^2_{sum}\right\}$ اسـت. سـپس در فاز دوم به‌ازای هر حالت موجود در فضای حالت شمارهٔ ۱ به‌نوعی یک الگوریتم برنامه‌ریزی پویا اجرا می‌شـود؛ بنابراین با به‌کارگیری اصـل غلبهٔ ۲ در گام ۵-۳ حداکثر تعداد حالت های درون فضـای حالت شـمارهٔ ۲ برابر $\left(n_1P_{sum}\right)\left(P^1_{sum}\right)$ خواهـد بود کـه در آن $\left(P^1_{sum}\right)$ و $\left(n_1P_{sum}\right)$ بـه ترتیـب دامنـهٔ تغییرات L^1_{sum} و $p^1_{sum\_before} + p^1_{sum\_after}$ ا ست؛ پس پیچیدگی اجرای فاز دوم با احتسـاب پیچیدگی حد بالا و درنتیجه پیـچیـدگـی کـل الـگـوریـتـم، بـرابـر $O\left(n_1\left(n_1\log n_1\right)\left(n_1P_{sum}\right)P^1_{sum}\min\left\{d_2, P^2_{sum}\right\}\right)$ اسـت که می‌توان آن را به‌صـورت $O\left(n_1^3\log n_1\left(P_{sum}\right)^3\right)$ نیز بیان کرد. همچنین در صـورتی که زمان پردازش همگی سفارش‌ها برابر واحد با شد، این الگوریتم در زمان چندجمله‌ای و با پیچیدگی $O\left(n_1^3n_2n\log n_1\right)$ قادر به حل مسئله است.

٤- نتایج محاسباتی

به‌منظور بررسی کارایی الگوریتم‌های ارائه‌شده در حل مسـئلـهٔ دوعـامـلـی پـذیـرش و زمـان‌بـنـدی $1|OA, d_i^2 = d_2, \sum U_i^2 = 0|\sum_{k=1}^{2}\sum_{i=1}^{n_k} q_i^k - \sum_{i=1}^{n_1} L_i^1$ سـعی شده تا کیفیت آنها در حل مسائل نمونه بررسی شود. این الگوریتم‌ها در محیط برنامه‌نویسـی Visual C# 2010 پیاده‌سـازی شـده و بر یک دسـتگاه رایانه با مشخصات Intel® Core™ i7-2600 CPU 3.4 GHz و 4 GB RAM در محیط سیستم عامل Windows 7 به اجرا گذا شته شده‌اند. همچنین مدل ریا ضی M1 با اسـتفاده از نرم‌افزار CPLEX نسـخهٔ ۱۱/۱حل شـده ا ست. محدودیت زمانی حل نیز در کلیهٔ م سائل برابر ۳٦۰۰ ثانیه در نظر گرفته شده است.

با توجه به جدیدبودن مسئلهٔ پذیرش و زمان‌بندی سفارش‌های دوعاملی که در این مقاله طرح شده، سعی شده تا با الگوگیری از نحوهٔ تولید مسائل نمونه در ادبیات موضوع پذیرش و زمان‌بندی سفارش‌ها و نیز مطالعات گذشته در زمینهٔ زمان‌بندی چندعاملی و اعمال تغییراتی براساس شرایط مسئلهٔ جدید، به طراحی مسائل نمونه به‌طور تصادفی پرداخته شود.

از آنجا که آزمایش‌های مقدماتی نشان‌دهندهٔ این است که تنها پارامترهای تعداد سـفارش هر عامل، درآمد و مدت‌زمان پردازش سـفارش‌های عامل اول و عامل دیرکرد (τ_1) عامل اول در کارایی و عملکرد الگوریتم

ارائه‌شده تأثیرگذار است، در ادامه تنها این پارامترها در آزمایش‌های محاسباتی در نظر گرفته شده‌اند؛ بنابراین زمان‌های پردازش سفارش‌های عامل اول از توزیع یکنواخت گسسته در دو بازهٔ $[1,10]$ مطابق نوبیبون و لئوس (۲۰۱۱) و $[1,100]$ براساس ادبیات موضوع زمان‌بندی چندعاملی (سلطانی و همکاران (۲۰۱۰)، چنگ و همکاران (۲۰۱۳) و لی و همکاران (۲۰۱۰)) تولید شده است. موعد تحویل هر سفارش از عامل اول مانند i نیز مطابق مطالعات قبلی (سلطانی و همکاران (۲۰۱۰)، لی و همکاران (۲۰۱۰)، نوبیبون و لئوس (۲۰۱۱) و یین و همکاران (۲۰۱۲) و از توزیع یکنواخت گسسته به‌صورت $\max\{p_i, U[P_{sum}(1-\tau_1-R/2), P_{sum}(1-\tau_1+R/2)]\}$ تولید شده که در آن τ_1 و R به‌ترتیب عامل دیرکرد و عامل پراکندگی موعد تحویل عامل اول هستند. در این مطالعه مقادیر ۰/۳ و ۰/۷ برای τ_1 در نظر گرفته شده است. به‌منظور بررسی تأثیر تعداد سفارش‌های هر عامل، در نیمی از مسائل تعداد سفارش‌های عامل اول نصف تعداد سفارش‌های عامل دوم ($2n_1 = n_2$) و در نیمی دیگر دو برابر تعداد سفارش‌های عامل دوم ($n_1 = 2n_2$) لحاظ شده است. همچنین مقدار درآمد سفارش های عامل اول از توزیع یکنواخت گسسته در دو بازهٔ $[1, 2\times p_i^1]$ و $[1, 20\times p_i^1]$ تولید شده است. مقادیر درنظرگرفته‌شده برای پارامترهای مسئله و سطوح مربوط به آنها در جدول نشان داده شده است. بدین ترتیب براساس مقادیر درنظرگرفته شده برای درآمد و زمان پردازش عامل اول، τ عامل اول و ترکیب تعداد سفارش عامل‌ها تعداد ۱٦ ($2\times2\times2\times2$) گروه مختلف با نام های G01 تا G16، شکل می‌گیرد. به‌ازای هر یک از گروه های ۱٦گانه، سه اندازهٔ مسئلهٔ ۷۰، ۱۱۰ و ۱۵۰ و به‌ازای هر اندازه در هر سری ۱۰ مسئلهٔ نمونه بررسی شده است.

جدول(۱): مشخصات پارامترهای استفاده‌شده در تولید مسائل نمونه

نام پارامتر	نماد	مقدار	سطح
اندازه	n	۷۰ ۱۱۰ ۱۵۰	۷۰ ۱۱۰ ۱۵۰
تعداد سفارش عامل اول	n_1	$2n_1=n_2$	Low
		$n_1=2n_2$	High
درآمد عامل اول	qi_1	$[1,2\times p_i^1]$	Low
		$[1,20\times p_i^1]$	High
زمان پردازش عامل اول	pi_1	[1,10]	Low
		[1,100]	High
مقدار τ عامل اول	tou_1	۰/۳	Low
		۰/۷	High

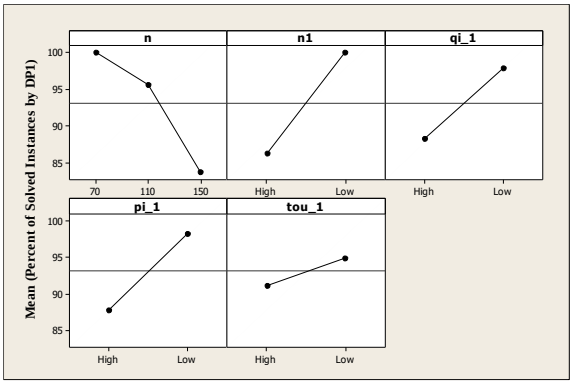
چون نتایج محاسباتی نشان داد که مدل ریاضی M1 قادر به حل کلیهٔ مسائل نمونه تنها تا ابعاد ۳۵ سفارش است و از طرفی الگوریتم DP1 مسائل را تا ابعاد ۷۰ سفارش به‌طور کامل حل کرده است؛ در ادامه تنها نتایج الگوریتم DP1 ارائه می‌شود.برای بررسی تأثیر پارامترهای جدول بر الگوریتم DP1، از آنالیز واریانس[۱] استفاده شده است؛ بنابراین با توجه به تعداد گروه‌ها و اندازه‌های مختلف، ٤۸ (۳×۱٦) تیمار[۲] در نظر گرفته شده است. گفتنی است که سه پیش‌فرض اصلی آنالیز واریانس شامل نرمال‌بودن و مستقل‌بودن باقیمانده‌ها و برابری واریانس‌ها بررسی شد و همهٔ این پیش‌فرض‌ها پذیرفته شده‌اند. نتیجهٔ این تحلیل در جدول (۲) برای متغیر پاسخ «درصد مسائل نمونهٔ بهینه حل‌شده توسط DP1 در ۳٦۰۰ ثانیه» به نمایش در آمده است. براساس جدول (۱) نیز درصد مسائل بیشتری حل شده‌اند.

جدول (۲): جدول آنالیز واریانس برای درصد مسائل نمونهٔ بهینه حل‌شده

Source	S.S.*	d.f.**	M.S.***	F	P-Value
Model	۱۵۱۵۸۳/۳۳۳	۲۰	۷۵۷۹/۱۶۷	۲۲/۳۳۹	۰/۰۰۰
n	۲۲۶۲۵/۰۰۰	۲	۱۱۳۱۲/۵۰۰	۳۳/۳۴۳	۰/۰۰۰
n_1	۲۲۶۸۷/۵۰۰	۱	۲۲۶۸۷/۵۰۰	۶۶/۸۷۰	۰/۰۰۰
q_i^1	۱۱۰۲۰/۸۳۳	۱	۱۱۰۲۰/۸۳۳	۳۲/۴۸۳	۰/۰۰۰
p_i^1	۱۳۰۲۰/۸۳۳	۱	۱۳۰۲۰/۸۳۳	۳۸/۳۷۸	۰/۰۰۰
τ_1	۱۶۸۷/۵۰۰	۱	۱۶۸۷/۵۰۰	۴/۹۷۴	۰/۰۲۶
$n * n_1$	۲۲۶۲۵/۰۰۰	۲	۱۱۳۱۲/۵۰۰	۳۳/۳۴۳	۰/۰۰۰
$n * q_i^1$	۸۰۴۱/۶۶۷	۲	۴۰۲۰/۸۳۳	۱۱/۸۵۱	۰/۰۰۰
$n * p_i^1$	۱۰۲۹۱/۶۶۷	۲	۵۱۴۵/۸۳۳	۱۵/۱۶۷	۰/۰۰۰
$n * \tau_1$	۱۶۲۵/۰۰۰	۲	۸۱۲/۵۰۰	۲/۳۹۵	۰/۰۹۲
$n_1 * q_i^1$	۱۱۰۲۰/۸۳۳	۱	۱۱۰۲۰/۸۳۳	۳۲/۴۸۳	۰/۰۰۰
$n_1 * p_i^1$	۱۳۰۲۰/۸۳۳	۱	۱۳۰۲۰/۸۳۳	۳۸/۳۷۸	۰/۰۰۰
$n_1 * \tau_1$	۱۶۸۷/۵۰۰	۱	۱۶۸۷/۵۰۰	۴/۹۷۴	۰/۰۲۶
$q_i^1 * p_i^1$	۴۶۸۷/۵۰۰	۱	۴۶۸۷/۵۰۰	۱۳/۸۱۶	۰/۰۰۰
$q_i^1 * \tau_1$	۷۵۲۰/۸۳۳	۱	۷۵۲۰/۸۳۳	۲۲/۱۶۷	۰/۰۰۰
$p_i^1 * \tau_1$	۲۰/۸۳۳	۱	۲۰/۸۳۳	۰/۰۶۱	۰/۸۰۴
Error	۱۵۵۷۲۹/۱۶۷	۴۵۹	۳۳۹/۲۷۹		
Total	۴۴۷۰۰۰۰/۰۰۰	۴۸۰			

* S.S. : Sum of Squares
** d.f. : degree of freedom
*** M.S. : Mean of Squares

پارامترهای درنظرگرفته‌شده در این آزمایش در سطح اطمینان[۴۳]۵درصد معنادار هستند. آثار متقابل پارامترها نیز به‌جز در دو مورد که مربوط به پارامتر τ_1 است، همگی در سطح اطمینان ۵درصد معنادار هستند. در شکل ۳ نیز آثار اصلی پارامترها بر متغیر پاسخ نشان داده شده است. همان‌طور که در این شکل نیز مشخص است کلیهٔ پارامترها بر متغیر پاسخ اثرگذار هستند. بر این اساس با افزایش ابعاد مسئله درصد کمتری از مسائل به‌طور بهینه توسط DP1 حل شده‌اند؛ به‌طوری که در ابعاد ۷۰، ۱۱۰ و ۱۵۰ سفارش به‌ترتیب ٪۱۰۰، ٪۹۵/۶۲ و ٪۸۳/۷۵ از مسائل نمونه به‌طور بهینه حل شده‌اند. همچنین با قرارگیری سایر پارامترها در سطح Low نیز درصد مسائل بیشتری حل شده‌اند.



شکل ۳. نمودار آثار اصلی پارامترها، متغیر پاسخ: درصد مسائل نمونهٔ بهینه حل‌شده توسط DP1

در جدول نتایج حل به‌ازای ۱۶ گروه G01 تا G16، ارائه شده است.

در این جدول برای بررسی عملکرد الگوریتم ابتکاری GAO از معیار میانگین درصد انحراف نسبی (RPD)[۴۴] استفاده شده که مطابق رابطهٔ زیر محاسبه می‌شود:

$$RPD = \frac{Z_{opt} - Z_{GAO}}{Z_{opt}} \times 100 \qquad (۱۳)$$

در رابطهٔ فوق Z_{opt} و Z_{GAO} به‌ترتیب مقدار تابع هدف حاصل از الگوریتم‌های DP1 (بهینه) و GAO است.

در جدول می‌توان مشاهده کرد که با افزایش ابعاد مسائل، مدت‌زمان حل کل الگوریتم DP1 و نیز مدت‌زمان صرف‌شده در هریک از دو فاز این الگوریتم نیز افزایش یافته است؛ به‌طوری که به‌ترتیب، میانگین مدت‌زمان حل مسائل نمونهٔ بهینه حل‌شده با ابعاد ۷۰، ۱۱۰ و ۱۵۰ سفارش برابر ۱۳/۰۴، ۲۹۸/۸۶ و ۴۱۶/۲۲ ثانیه است.

جدول(۳): نتایج حل DP1 به‌ازای گروه‌های ۱٦گانه

گروه	مقادیر پارامترها					تعداد نمونه بهینه از ۱۰ نمونه	میانگین RPD برای الگوریتم GAO	میانگین مدت زمان حل (ثانیه)			فاز اول: میانگین درصد حالات				فاز دوم: میانگین درصد حالات			
	n_i	q^i_1	p^i_1	τ_1	n			کل	فاز اول	فاز دوم	بررسی شده	حذف شده به دلیل		منتقل شده به فاز دوم	بررسی شده	حذف شده به دلیل		باقی مانده
												حد بالا	اصل غلبهٔ ۱			حد بالا	اصل غلبهٔ ۲	
G01	$2n_1=n_2$	$[1,2\times p^i_1]$	[۱,۱۰]	۰/۳	۷۰	۱۰	۸/۲۸	۰/۰۸	۰/۰۲	۰/۰٦	۲٦/٦۰	۸/۳۸	۸۷/٤۳	٤/۲۰	۷۳/٤۰	۹۰/٦۲	۱/۳۷	۸/۰۱
					۱۱۰	۱۰	۷/۲۰	۰/۳٦	۰/۰۷	۰/۲۹	۲۵/۳۸	٦/٤٦	۹۰/۸۲	۲/۷۲	۷٤/٦۲	۹۳/٤۰	۱/۱۵	۵/٤۵
					۱۵۰	۱۰	٦/۵۹	۱/۰۹	۰/۲٤	۰/۸۵	۱۹/۱۱	٤/۹۰	۹۳/۱۷	۱/۹۳	۸۰/۸۹	۹۳/۱۹	۲/۲۹	٤/۵۱
G02			[۱,۱۰]	۰/۷	۷۰	۱۰	۲٤/۱۵	۰/۲۵	۰/۰۲	۰/۲۳	۵/۸۰	۲/۹۱	۹۲/۳٦	٤/۷٤	۹٤/۲۰	۸۷/۷۹	۳/۷۱	۸/۵۰
					۱۱۰	۱۰	۲٤/۹۹	۱/۷٦	۰/۱۰	۱/٦٦	۲/٦٦	۱/۷۵	۹۵/۱۲	۳/۱۲	۹۷/۳٤	۹۰/۱۷	٤/٦۰	۵/۲۳
					۱۵۰	۱۰	۲۵/۲۱	۷/۹۲	۰/۲۹	۷/٦٤	۱/٦۱	۱/٤۲	۹٦/۳۰	۲/۲۸	۹۸/۳۹	۸۹/۳۰	٦/٤٤	٤/۲٦
G03			[۱,۱۰۰]	۰/۳	۷۰	۱۰	۱۲/۸٦	۰/۲۸	۰/۰۳	۰/۲۵	۵/۷۹	۰/۰۰	۹٤/۰۰	٦/۰۰	۹٤/۲۱	۸۱/۹۸	٦/۹٦	۱۱/۰٦
					۱۱۰	۱۰	۱۰/۲٤	۳/۵۱	۰/۱۱	۳/٤۰	۲/٦۰	۰/۰۰	۹٦/۰۷	۳/۹۳	۹۷/٤۰	۸۰/۳۹	۱۱/۳۰	۸/۳۱
					۱۵۰	۱۰	۱۱/۲۲	۱۸/٤۲	۰/۳۹	۱۸/۰۳	۱/٦۳	۰/۰۰	۹۷/۲۰	۲/۸۰	۹۸/۳۷	۷۹/٤٤	۱٤/٤۹	٦/۰۷
G04			[۱,۱۰۰]	۰/۷	۷۰	۱۰	۱۱/٤٤	۰/٦۷	۰/۰۳	۰/٦٤	۱/۹۸	۰/۰۰	۹٤/۰۳	۵/۹۷	۹۸/۰۲	۸۹/۱۵	۳/۳۹	۷/٤۷
					۱۱۰	۱۰	۱۰/۷۲	۱۸/۲۲	۰/۱۱	۱۸/۱۱	۰/۳٤	۰/۰۰	۹٦/۰۷	۳/۹۳	۹۹/٦٦	۸٦/۷۹	۸/٤۷	٤/۷۵
					۱۵۰	۱۰	۱۲/۵٤	۱۳۲/۸۳	۰/۳۸	۱۳۲/٤٤	۰/۱٤	۰/۰۰	۹۷/۲۱	۲/۷۹	۹۹/۸٦	۸۸/۰۱	۸/٤۸	۳/۵۱
G05		$[1,20\times p^i_1]$	[۱,۱۰]	۰/۳	۷۰	۱۰	۲/۱٦	۰/۰۷	۰/۰۲	۰/۰۵	۲۵/۱۱	۱۰/۹۲	۸۵/۲۰	۳/۸۸	۷٤/۸۹	۹۲/۳۲	۰/۰۰	۷/٦۸
					۱۱۰	۱۰	۳/٦۰	۰/۳۷	۰/۰۸	۰/۲۹	۲۱/٤٤	۱۰/٤۱	۸۷/۰٤	۲/۵۵	۷۸/۵٦	۹۳/۸۸	۰/۷٤	۵/۳۸
					۱۵۰	۱۰	٤/۱۸	۱/۰۸	۰/۲۷	۰/۸۱	۲۳/۹٦	۵/۹۱	۹۲/۲۵	۱/۸۵	۷٦/۰٤	۹۵/۲٤	۰/٦۱	٤/۱۵
G06			[۱,۱۰]	۰/۷	۷۰	۱۰	۱۵/٤۹	۰/۲۸	۰/۰۲	۰/۲٦	٦/۰۰	۲/۹۱	۹۲/۳٦	٤/۷۳	۹٤/۰۰	۸۷/۸۷	۳/۵۸	۸/۵۵
					۱۱۰	۱۰	۲۳/۰۸	۵/۱۲	۰/۱۰	۵/۰۲	۱/٦۷	۱/۷۷	۹۵/۰۹	۳/۱۳	۹۸/۳۳	۸۵/٦۳	۸/۵۹	۵/۷۸
					۱۵۰	۱۰	۲۱/٦۲	۳۸/٦۲	۰/۳۳	۳۸/۲۹	۰/۸۷	۱/٤۱	۹٦/۳۱	۲/۲۸	۹۹/۱۳	۸٦/۸۲	۸/۷۹	٤/۳۸
G07		$[1,20\times p^i_1]$	[۱,۱۰۰]	۰/۳	۷۰	۱۰	۷/۸۲	۰/۲٤	۰/۰۳	۰/۲۱	۸/۰٦	۰/۰۰	۹٤/۱۲	۵/۸۸	۹۱/۹٤	۸۸/۹۱	۲/۷۳	۸/۳۵
					۱۱۰	۱۰	۹/۱۱	۲/۲۷	۰/۱۲	۲/۱۵	۳/۱۵	۰/۰۰	۹٦/۱۳	۳/۸۷	۹٦/۸۵	۸۵/۸۰	۷/۳٦	٦/۸٤
					۱۵۰	۱۰	۹/۵۳	۲٦/۹۳	۰/۳۷	۲٦/۵٦	۰/۸۰	۰/۰۰	۹۷/۱۷	۲/۸۳	۹۹/۲۰	۸٦/۳۲	۷/۹۵	۵/۷٤
G08			[۱,۱۰۰]	۰/۷	۷۰	۱۰	۱٤/۹۵	۱/٦٤	۰/۰۳	۱/٦۱	۱/۰۳	۰/۰۰	۹٤/۰٦	۵/۹٤	۹۸/۹۷	۸۵/۰۹	٦/۲۷	۸/٦٤
					۱۱۰	۱۰	۱٦/٤٤	۷٦/۰۹	۰/۱۲	۷۵/۹۷	۰/۱۸	۰/۰۰	۹٦/۰۹	۳/۹۱	۹۹/۸۲	۸۳/۰۰	۱۱/۳٦	۵/٦٤
					۱۵۰	۱۰	۱٤/۲۱	٦۷۸/۸۱	۰/۳۸	٦۷۸/٤٤	۰/۰۵	۰/۰۰	۹۷/۱۷	۲/۸۳	۹۹/۹۵	۷۸/۵۳	۱۷/۱۱	٤/۳٦
G09	$n_1=2n_2$	$[1,2\times p^i_1]$	[۱,۱۰]	۰/۳	۷۰	۱۰	۲/۵۹	۰/٤۳	۰/۰۱	۰/٤۲	۰/۹۵	۱۲/۰۷	۷۸/۱۸	۹/۷۵	۹۹/۰۵	۸٤/۸٤	۸/۲۵	٦/۹۱
					۱۱۰	۱۰	۲/۷۱	۳/٤٤	۰/۰۵	۳/۳۹	۰/۵۷	۱۰/۱۵	۸٤/۲۱	۵/٦٤	۹۹/٤۳	۸۲/٤٦	۱۲/۷۱	٤/۸۳
					۱۵۰	۱۰	۲/٦۵	۲۳/۵۸	۰/۱٦	۲۳/٤۲	۰/۳۵	٦/٦۹	۸۹/۲۰	٤/۱۱	۹۹/٦۵	۸۳/٦۹	۱۲/۵۳	۳/۷۸
G10			[۱,۱۰]	۰/۷	۷۰	۱۰	٤/۳٤	۱/۵٦	۰/۰۱	۱/۵۵	۰/۱۸	۱/۹۹	۸۷/۰۷	۱۰/۹۵	۹۹/۸۲	۸۸/۷۰	۷/٤۳	۳/۸۸
					۱۱۰	۱۰	٤/۷۵	۲۹/۸۵	۰/۰٦	۲۹/۷۹	۰/۰٦	۱/۱۵	۹۱/۹۰	٦/۹۵	۹۹/۹٤	۸۸/۱۰	۹/۳۹	۲/۵۱
					۱۵۰	۱۰	٤/۳۲	۱۵٤/۵۵	۰/۲۰	۱۵٤/۳۵	۰/۰۳	۰/۹٤	۹۳/۹۳	۵/۱٤	۹۹/۹۷	۸۸/۳۷	۹/۸۳	۱/۸۰
G11		$[1,20\times p^i_1]$	[۱,۱۰۰]	۰/۳	۷۰	۱۰	۳/۲۷	۱٦/۸٤	۰/۰۲	۱٦/۸۲	۰/۰۸	۰/۰۰	۸۷/۷۲	۱۲/۲۸	۹۹/۹۲	۷۵/۹۱	۱٦/۷۲	۷/۳۷
					۱۱۰	۱۰	۲/۹۹	۱۱۹/۳۷	۰/۰۸	۱۱۹/۲۸	۰/۰۳	۰/۰۰	۹۲/۲۰	۷/۸۰	۹۹/۹۷	۷۸/٤۷	۱٦/۵٦	٤/۹۷
					۱۵۰	۵	۱/۰٤	۳۹۰/۱۲	۰/۲۸	۳۸۹/۸۳	۰/۰۱	۰/۰۰	۹٤/۳۱	۵/٦۹	۹۹/۹۹	۹۳/۳۳	٤/۲٦	۲/٤۱
G12			[۱,۱۰۰]	۰/۷	۷۰	۱۰	۳/۷٤	۱۲/۸۵	۰/۰۱	۱۲/۸٤	۰/۰۳	۰/۰۰	۸۷/۰۲	۱۲/۹۸	۹۹/۹۷	۸۷/۱۳	۸/۹۲	۳/۹٤
					۱۱۰	۱۰	٤/۵۵	۲۵۳/٦۲	۰/۰۹	۲۵۳/۵۳	۰/۰۱	۰/۰۰	۹۲/٤۱	۷/۵۹	۹۹/۹۹	۸۷/۰٤	۱۰/۵۷	۲/۳۹
					۱۵۰	۱۰	٤/٤۷	۱۸۸٤/٤۷	۰/۲۷	۱۸۸٤/۲۱	۰/۰۰	۰/۰۰	۹٤/۳۳	۵/٦۷	۱۰۰/۰۰	۸۷/۷٤	۱۰/۵٦	۱/۷۰
G13		$[1,20\times p^i_1]$	[۱,۱۰]	۰/۳	۷۰	۱۰	۱/۵۸	۰/٤۸	۰/۰۱	۰/٤۷	۱/۵٦	۱۵/٤۲	۷٤/۹۱	۹/٦۷	۹۸/٤٤	۸۹/۷۲	٤/۰۷	٦/۲۱
					۱۱۰	۱۰	۱/۸۳	۱۱/۳۸	۰/۰٦	۱۱/۳۲	۰/٦٦	۱۱/۵۹	۸۲/۹۲	۵/۵۰	۹۹/۳٤	۸۸/٤۱	۷/۰٤	٤/۵۵
					۱۵۰	۱۰	۲/٦۲	۱۱۲/٤۰	۰/۲۱	۱۱۲/۱۹	۰/۲۲	۵/۱۰	۹۰/۵۸	٤/۳۲	۹۹/۷۸	۸٤/٤۳	۱۱/۸۹	۳/٦۸
G14			[۱,۱۰]	۰/۷	۷۰	۱۰	٤/۲۹	۱۳/۱٤	۰/۰۱	۱۳/۱۳	۰/۰٦	۱/۵٤	۸۷/۱٤	۱۱/۳۱	۹۹/۹٤	۸۰/٤۵	۱٤/۹۲	٤/٦۳
					۱۱۰	۱۰	٤/۲۸	۵٦٦/۰٤	۰/۰۹	۵٦۵/۹۵	۰/۰۲	۰/۷۹	۹۲/۳٦	٦/۸۵	۹۹/۹۸	۷٦/۳۸	۲۰/۵۷	۳/۰٦
					۱۵۰	٦	۷/۰۵	۱۹٦۲/۱۵	۰/۲۷	۱۹٦۱/۸۸	۰/۰۱	۰/٦٤	۹٤/۳۰	۵/۰۷	۹۹/۹۹	۸۰/۸۳	۱۷/۰۱	۲/۱۵
G15		$[1,20\times p^i_1]$	[۱,۱۰۰]	۰/۳	۷۰	۱۰	۲/۲۷	۳/۷۷	۰/۰۱	۳/۷٦	۰/۱۱	۰/۰۰	۸۷/۷۲	۱۲/۲۸	۹۹/۸۹	۸٦/۳۱	۷/٤٦	٦/۲۳
					۱۱۰	۱۰	۳/۰٦	۸۳٤/۲۵	۰/۰۷	۸۳٤/۱۸	۰/۰۱	۰/۰۰	۹۲/۲۸	۷/۷۲	۹۹/۹۹	۸۰/۷۵	۱٤/٤۹	٤/۷٦
					۱۵۰	۳	۱/٤۱	۸۱۰/۳۹	۰/۲۹	۸۱۰/۱۱	۰/۰۱	۰/۰۰	۹٤/۵٦	۵/٤٤	۹۹/۹۹	۹۲/۸۷	٤/٦۹	۲/٤٤
G16			[۱,۱۰۰]	۰/۷	۷۰	۱۰	۵/۱٤	۱۵٦/۰۸	۰/۰۱	۱۵٦/۰۷	۰/۰۱	۰/۰۰	۸۷/۵۲	۱۲/٤۸	۹۹/۹۹	۷۱/٤۱	۲۳/۷٤	٤/۸۵
					۱۱۰	۳	۳/٦٦	۲۸۵٦/۱۸	۰/۰۸	۲۸۵٦/۱۰	۰/۰۰	۰/۰۰	۹۱/۹٤	۸/۰٦	۱۰۰/۰۰	۷۳/۲۷	۲۳/۷۵	۲/۹۸
					۱۵۰	۰	-	-	-	-	-	-	-	-	-	-	-	-
میانگین							۸/۲۲	۲۳۹/۰۲	۰/۱۳	۲۳۸/۸۹	٤/۰٦	۲/۷۱	۹۱/۵۸	۵/۷۷	۹۵/۹٤	۸۵/۷۵	۹/۰٤	۵/۲۱

همچنین براساس این جدول سخت‌ترین مسائل مربوط به گروه G16 است که در آن تمامی پارامترها در بیشترین مقدار خود قرار دارند؛ به‌طوری که در این گروه هیچ‌یک از ۱۰ مسئلهٔ نمونهٔ تولیدشده در ابعاد ۱۵۰ سفارش توسط DP1 حل نشده‌اند. بررسی نتایج نشان داد که دلیل این امر را می‌توان افزایش تعداد حالت‌های فضای حالت شمارهٔ ۲ در فاز دوم ذکر کرد که علی‌رغم عملکرد خوب اصل غلبهٔ ۲ و حد بالا در این فاز، تعداد حالت‌ها به‌طور چشم‌گیری افزایش داشته و بنابراین الگوریتم نتوانسته در مدت‌زمان تعیین‌شده همهٔ آنها را بررسی کرده و به جواب دست پیدا کند.

نکتهٔ دیگر در جدول، مدت‌زمان بسیار کم صرف شده در فاز اول الگوریتم است. همان‌طور که ذکر شد با توجه به اینکه در فاز اول به‌نوعی یک مسئلهٔ کوله‌پشتی حل می‌شود و از آنجا که این مسئله به‌دلیل استفاده از اصل غلبهٔ ارائه‌شده، در ابعاد موردبررسی بسیار سریع حل می‌شود، این امر دور از انتظار نیست. از طرف دیگر چون به‌ازای هر حالت نهایی در فضای حالت شمارهٔ ۱ یک الگوریتم برنامه‌ریزی پویا انجام می‌شود. بدیهی است که فاز دوم الگوریتم زمان‌بر باشد. براساس درصد حالات بررسی‌شدهٔ هر فاز نیز می‌توان این امر را تأیید کرد.

با توجه به جدول می‌توان عملکرد اصول غلبه و حد بالا را نیز بررسی کرد. در فاز اول الگوریتم، اصل غلبهٔ ۱ به‌طور میانگین ٪۹۱/۵۸ از حالت‌های تولیدشده را حذف کرده است؛ اما بر خلاف فاز اول، در فاز دوم حد بالا عملکرد قابل‌ملاحظه‌ای داشته و در فاز دوم به‌طور میانگین ٪۸۵/۷۵ از حالت‌های این فاز به‌دلیل حد بالا حذف شده‌اند و به‌طور طبیعی درصد بسیار کمتری توسط اصل غلبهٔ ۲ می‌تواند حذف شود که این مقدار برابر ٪۹/۰۴ است. به هر حال، مجموع حد بالا و اصول غلبه در فاز اول و دوم به‌ترتیب ٪۹۴/۲۹ و ٪۹۴/۷۹ از حالت‌های تولیدشده را حذف کرده‌اند.

همان‌طور که در گام ۱ الگوریتم برنامه‌ریزی پویا آمده است، جواب الگوریتم ابتکاری GAO به‌عنوان یک جواب اولیه (حد پایین مسئله) برای استفاده در الگوریتم برنامه‌ریزی پویا استفاده شده است؛ بنابراین از آنجا که هدف اصلی از توسعهٔ این الگوریتم استفاده از جواب آن به‌عنوان در الگوریتم برنامه‌ریزی پویا بوده است، ادعای چندانی مبنی بر کیفیت جواب آن وجود ندارد. با این حال با میانگین درصد انحراف نسبی این الگوریتم در تمام گروه‌هایی که جواب بهینه از الگوریتم DP1 به دست آمده است، برابر ٪۸/۲۲ است که عملکرد قابل‌قبولی به نظر می‌رسد.

۵- نتیجه‌گیری

در این مقاله یک مسئلهٔ کاربردی در حوزهٔ تحقیق در عملیات با عنوان پذیرش و زمان‌بندی سفارش‌ها بررسی شد. این مسئله در حالت وجود دو عامل رقابتی یا دو نوع مشتری مطالعه شد و یک مسئلهٔ جدید با هدف بیشینه‌سازی مجموع سود سفارش‌های عامل اول و مجموع درآمد سفارش‌های عامل دوم با درنظرگرفتن مجموع مغایرت به‌عنوان تابع جریمهٔ عامل اول بررسی شد. در این مسئله فرض شد که عامل دوم هیچ سفارش همراه با دیرکردی را نمی‌پذیرد و نیز سفارش‌های این عامل دارای موعد تحویل مشترک هستند. پس از اینکه نشان داده شد این مسئله به‌طور عادی NP-hard است؛ برای حل آن یک مدل ریاضی، یک الگوریتم ابتکاری و یک برنامه‌ریزی پویای

شبه‌چندجمله‌ای ارائه شد. برای بررسی عملکرد الگوریتم‌ها نیز تعدادی مسئلهٔ نمونه تولید شد و تأثیر پارامترهای لحاظ‌شده در تولید این مسائل با استفاده از آنالیز واریانس تحلیل شد. نتایج حل حاکی از قابلیت حل تمامی مسائل تا ابعاد ۷۰ سفارش در مدت‌زمان اندک، توسط الگوریتم برنامه‌ریزی پویا است. همچنین در کل ٪۹۳/۱۲ از مسائل تا ابعاد ۱۵۰ سفارش توسط این الگوریتم در مدت‌زمان ۳٦۰۰ ثانیه به‌طور بهینه حل شده است. الگوریتم ابتکاری نیز با میانگین درصد انحراف نسبی ٪۸/۲۲ عملکرد قابل‌قبولی از خود نشان داده است.

برای مطالعات آتی با توجه به جدیدبودن مسئلهٔ موردبررسی می‌توان فرضیات متنوعی را در نظر گرفت و مسئله را حل کرد؛ مثلاً درنظرگرفتن تابع جریمهٔ دیرکرد وزنی و یا محیط چندماشین می‌تواند جالب‌توجه باشد. همچنین می‌توان روش‌های حل دیگری نظیر شاخه و کران را بررسی کرد و با برنامه‌ریزی پویای ارائه‌شده مقایسه کرد. جدای از این موارد کلی می‌توان وزن‌دارکردن سفارش‌ها را نیز در مسئله افزود و یا حالت‌های مختلف دو عامل را نیز بررسی کرد که از آن جمله می‌توان درنظرگرفتن جریمهٔ هر دو عامل در تابع هدف را ذکر کرد. همگی این موارد به کاربردی‌ترشدن مسئله کمک می‌کنند؛ ولی حل آن را پیچیده می‌کنند.

منابع

Agnetis, A., Mirchandani, P. B., Pacciarelli, D., & Pacifici, A. (2004). "Scheduling Problems with Two Competing Agents". *Operations Research, 52*(2), 229-242.

Baker, K. R., & Smith, J. C. (2003). "A multiple criterion model for machine scheduling". *Journal of Scheduling*, 6(1): 7-16.

Cesaret, B., Oğuz, C., & Salman, F. S. (2012). "A tabu search algorithm for order acceptance and scheduling". *Computers and Operations Research*, 39, 1197-1205.

Chen, C., Yang, Z., Tan, Y., & He, R. (2014). "Diversity Controlling Genetic Algorithm for Order Acceptance and Scheduling Problem". *Mathematical Problems in Engineering*, 2014, 1-11.

Cheng, T. C. E., Chung, Y. H., Liao, S. C., & Lee, W. C. (2013). "Two-agent singe-machine scheduling with release times to minimize the total weighted completion time". *computers and Operations Research*, 40, 353–361.

Garey, M. R., & Johnson, D. S. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness (First Edition ed.): W. H. Freeman.

Ghosh, J. B. (1997). "Job selection in a heavily loaded shop". *Computers and Operations Research*, 24(2): 141-145.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1979). "Optimization and approximation in deterministic machine scheduling: A survey". *Annals of Discrete Mathematics*, 5, 287-326.

Lee, W. C., Chen, S. K., & Wu, C. C. (2010). "Branch-and-bound and simulated annealing algorithms for a two-agent scheduling problem". *Expert Systems with Applications*, 37(9), 6594–6601.

Lee, W. C., & Wang, J. Y. (2014). "A scheduling problem with three competing agents". *Computers & Operations Research*, 51(0), 208-217.

Leung, J. Y. T., Pinedo, M., & Wan, G. (2010). "Competitive Two-Agent Scheduling and Its Applications". *Operations Research*, 58(2), 458-469.

Nobibon, F. T., & Leus, R. (2011). "Exact algorithms for a generalization of the order acceptance and scheduling problem in a single-machine environment". *Computers and Operations Research*, 38(1), 367-378.

Rom, W. O., & Slotnick, S. A. (2009). "Order acceptance using genetic algorithms*". computers and Operations Research*, 36, 1758–1767.

Slotnick, S. A. (2011). "Order acceptance and scheduling: A taxonomy and review". *European Journal of Operational Research*, 212, 1-11.

Slotnick, S. A., & Morton, T. E. (1996). "Selecting jobs for a heavily loaded shop with lateness penalties". *Computers and Operations Research*, 23(2), 131-140.

Slotnick, S. A., & Morton, T. E. (2007). "Order acceptance with weighted tardiness". *Computers and Operations Research*, 34, 3029–3042.

Soltani, R., Jolai, F., & Zandieh, M. (2010). "Two robust meta-heuristics for scheduling multiple job classes on a single machine with multiple criteria". *Expert Systems with Applications*, 37, 5951-5959.

Wu, C. C., Wu, W. H., Chen, J. C., Yin, Y., & Wu, W. H. (2013). "A study of the single-machine two-agent scheduling problem with release times". *Applied Soft Computing*, 13, 998–1006.

Yin, Y., Wu, C. C., Wu, W. H., Hsu, C. J., & Wu, W.

H. (2013). "A branch-and-bound procedure for a single-machine earliness scheduling problem with two agents". *Applied Soft Computing*, 13, 1042–1054.

Yin, Y., Wu, W. H., Cheng, S. R., & Wu, C. C. (2012). "An investigation on a two-agent single-machine scheduling problem with unequal release dates". *Computers and Operations Research*, 39, 3062–3073.

Yuan, J. J., Shang, W. P., & Feng, Q. (2005). "A note on the scheduling with two families of jobs". *Journal of Scheduling*, 8, 537-542.

Zhao, K., & Lu, X. (2013). "Approximation schemes for two-agent scheduling on parallel machines". *Theoretical Computer Science*, 468, 114–121.

Zhao, K., & Lu, X. (2014). "Two approximation algorithms for two-agent scheduling on parallel machines to minimize makespan". *Journal of Combinatorial Optimization*, 1-19.

پی نوشت:

[1] multi agent scheduling
[2] Slotnick
[3] Slotnick and Morton
[4] Weighted Shortest Processing Time
[5] Ghosh
[6] Fully Polynomial Time Approximation Scheme
[7] Rom and Slotnick
[8] Nobibon and Leus
[9] Tabu Search
[10] Chen etal.
[11] Cesaret et al.
[12] Agnetis et al.
[13] regular function
[14] ordinary NP-hard
[15] marriage in honey-bees optimization algorithm
[16] strongly NP-hard
[17] Simulated Annealing
[18] ant colony
[19] Shortest Processing Time
[20] Revenue and Slack time to Processing time Ratio
[21] ANalysis Of VAriance
[22] treatment
[23] confidence level
[24] Relative Percentage Deviation