Original Article

مسئله‌ی بهینه‌سازی مقید پذیرش و زمان‌بندی سفارشات دو عاملی با هدف بیشینه‌سازی مجموع سود

محمد رئیسی‌نافچی (استادیار)
قاسم مصلحی * (استاد)
دانشکده‌ی مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان

مهندسی صنایع و مدیریت شریف، (زمستان ۱۳۹۳)
دوره‌ی ۱ - ۳۰، شماره‌ی ۲، ص. ۸۷-۹۹

در این مقاله مسئله‌ی پذیرش و زمان‌بندی سفارشات با مسئله‌ی زمان‌بندی دوعاملی ترکیب شده و یک مسئله‌ی کاربردی‌تر براساس نیازهای متفاوت مشتریان مورد بررسی قرار گرفته است. لذا فرض شده که دو دسته مشتری (عامل) وجود دارد و هدف بیشینه‌سازی مجموع سود سفارشات پذیرفته شده‌ی عامل اول به علاوه‌ی مجموع درآمد سفارشات پذیرفته شده‌ی عامل دوم است؛ به‌طوری که هیچ سفارشی از عامل دوم دیرکرد نداشته باشد. همچنین با این فرض که سفارش‌های عامل اول همگی دارای زمان پردازش یکسان هستند، نشان داده شده که این مسئله NP-hard است. در حالت معلوم بودن سفارشات پذیرفته شده، یک الگوریتم چندجمله‌یی برای تعیین توالی بهینه‌ی سفارشات، ارائه شده و برای حل مسئله‌ی اصلی نیز یک الگوریتم ابتکاری و یک برنامه‌ریزی پویای شبه‌چندجمله‌یی توسعه داده شده است. نتایج نشان می‌دهد که ۹۳٫۶۵٪ از مسائل تا ابعاد ۱۵۰ سفارش به‌صورت بهینه حل شده است.

واژگان کلیدی: پذیرش سفارش، زمان‌بندی، مغایرت زمان تکمیل و موعد تحویل، تعداد سفارش‌های دیرکرددار، آنالیز واریانس.

reisi.m@cc.iut.ac.ir
moslehi@cc.iut.ac.ir

۱. مقدمه

در طول دو دهه‌ی گذشته تولیدکنندگان و نیز محققین اهمیت پذیرش سفارشات را درک کرده‌اند. تصمیم‌گیری در مورد رد یا پذیرش یک سفارش ممکن است دلایل مختلفی داشته باشد. برای مثال رویکرد استراتژیک شرکت در تمرکز بر بخشی از بازار یا ظرفیت منابع موجود می‌تواند دلیل رد برخی از سفارشات مشتریان باشد. در این میان تصمیم درمورد رد یا پذیرش سفارشات می‌تواند مبتنی بر موازنه بین هزینه‌های تولید و رد آن سفارش و درآمدهای ناشی از پذیرش آن باشد. بر این اساس، در ادبیات موضوع مطالعات متنوعی با در نظرگرفتن فرضیات مختلف روی این مسئله صورت گرفته است. در یک مقاله مروری، آخرین مطالعات انجام شده در زمینه‌ی پذیرش و زمان‌بندی سفارشات (OAS)[1]، دسته‌بندی شده[1] که برای مطالعه بیشتر می‌توان به آن رجوع کرد.

با مرور ادبیات موضوع در این حیطه، به نظر می‌رسد که تاکنون فرض بر این بوده که تولیدکننده فقط برمبنای یک نوع قرارداد با مشتریان خود برخورد می‌کند. این فرض در شرایط بازار رقابتی امروز چندان کاربردی نیست. در این نوشتار سعی شده تا مسئله‌ی فوق با در نظرگرفتن تنوع مشتریان تعمیم داده شود. اخیراً این نوع نگاه در ادبیات موضوع زمان‌بندی تحت عنوان «زمان‌بندی چندعاملی (MAS)[2]» یا «زمان‌بندی با عامل‌های رقابتی[3]» مطرح شده و مورد توجه قرارگرفته است. لذا نوآوری این مقاله، پرداختن به این دو مسئله به‌طور همزمان است در حالی که در مطالعات قبلی این امر مشاهده نشده است. در ادامه، برخی از مطالعات صورت گرفته در حوزه‌ی پذیرش و زمان‌بندی سفارشات و مسئله‌ی زمان‌بندی چندعاملی به‌طور مختصر مرور شده است.

اسلاتنیک و مورتون[2] مسئله‌ی پذیرش و زمان‌بندی سفارشات (OAS) را با هدف بیشینه‌سازی سود که برابر است با مجموع درآمد منهای جریمه‌های وزن‌دار مغایرت زمان تکمیل با موعد تحویل، مورد مطالعه قرار دادند. آنان شرایطی ارائه کردند که براساس آن، زیرمجموعه‌یی از سفارش‌ها ــ که قطعاً در جواب بهینه حضور دارند ــ و زیرمجموعه‌یی از سفارش‌ها ــ که باید رد شوند ــ تشخیص داده شوند. این خاصیت فضای جست‌وجو را کاهش می‌دهد. همچنین در این مطالعه یک الگوریتم شاخه و کران، یک الگوریتم جست‌وجوی شعاعی[4] و یک الگوریتم ابتکاری حریصانه[5] ارائه شده و کارآیی آنها بر تعدادی مسئله‌ی نمونه بررسی شده است. در مطالعات بعدی[3] نشان داده شد که این مسئله NP-hard است و برای حل آن دو الگوریتم برنامه‌ریزی پویای شبه چندجمله‌یی با ساختاری ساده با هدف توسعه‌ی یک الگوی تقریب کاملاً چندجمله‌یی (FPTAS)[6] ارائه شد.

---

* نویسنده مسئول

تاریخ: دریافت ۱۳۹۲/۴/۸، اصلاحیه ۱۳۹۲/۷/۹، پذیرش ۱۳۹۲/۷/۲۷.

در ادامه‌ی بررسی‌ها[۴] مسئله‌ی پذیرش و زمان‌بندی سفارش‌ها در حالت تک‌ماشین، با در نظر گرفتن جریمه‌ی دیرکرد وزنی بررسی شد که طی آن، یک الگوریتم شاخه و کران و چند الگوریتم ابتکاری برای حل مسئله ارائه شد. دیگر محققین نیز یک الگوریتم ژنتیک برای مسئله‌ی فوق توسعه دادند[۵] که جواب‌های با کیفیت بیشتری نسبت به الگوریتم‌های ابتکاری پیشین تولید می‌کند. در مطالعات بعدی همین مسئله در حالتی بررسی شد که تعدادی سفارش پذیرفته شده از قبل و تعدادی سفارش در انتظار تصمیم وجود دارد.[۶] در آن بررسی، دو مدل برنامه‌ریزی خطی عدد صحیح مختلط (MILP)[۷] و دو الگوریتم شاخه و کران ارائه شد.

بررسی مسئله‌ی OAS با هدف بیشینه‌سازی سود همراه با در نظر گرفتن هزینه‌های دیرکرد، کاهش زمان پردازش و توسعه‌ی افق زمان‌بندی مورد بررسی قرار گرفت.[۷] محققین در آن مطالعه یک مدل MILP برای حل مسائل کوچک و تعدادی الگوریتم ابتکاری برای مسائل بزرگ ارائه کردند.

در مطالعه‌یی دیگر[۸] فرض مجاز بودن اضافه‌کاری و مجاز نبودن تأخیر در محیط کارگاهی[۸] به‌منظور بیشینه‌سازی سود حاصل از سفارش‌های پذیرفته شده در مسئله‌ی OAS مورد بررسی قرارگرفت. در مطالعه‌ی مذکور افق برنامه‌ریزی به تعدادی دوره زمانی تقسیم، و ورود سفارش‌ها در هر دوره ایستا فرض شد. تصمیم درمورد رد یا پذیرش هر سفارش نیز در ابتدای هر دوره صورت می‌گیرد. برای این مسئله یک مدل MILP توسعه داده شد که با استفاده از نرم‌افزارهای عمومی حل مدل، قادر به حل مسائل با ابعاد کوچک است؛ برای مسائل بزرگ‌تر نیز یک الگوریتم شاخه و قیمت[۹] ارائه کردند. مسئله‌ی OAS در محیط فلوشاپ دوماشینه[۱۰] در دو حالت بررسی شد.[۹] در حالت نخست تابع هدف کمینه‌سازی مجموع دامنه‌ی عملیات و مجموع هزینه‌های رد سفارش بررسی شد و نشان داده شد که این مسئله NP-hard است؛ و برای حل آن نیز دو الگوریتم تقریب متفاوت و یک الگوریتم شبه‌چندجمله‌یی توسعه داده شد. در حالت دوم یافتن کلیه‌ی نقاط پارتوی مسئله با اهداف دامنه‌ی عملیات و مجموع هزینه‌های رد سفارش تعقیب شده و پس از اثبات NP-hard بودن آن نشان دادند که مسئله در این حالت نیز در زمان شبه‌چندجمله‌یی قابل حل است.

برخی مطالعات صورت گرفته در مسئله‌ی زمان‌بندی چندعاملی نیز در ادامه مرور شده است. به‌منظور نشان دادن مسائل بررسی شده در ادبیات موضوع، از نمادگذاری سه‌جزئی[۱۰] به‌شکل $\alpha|\beta|\gamma$ استفاده شده است. در این نمادگذاری اجزاء α، β و γ به‌ترتیب نشانگر محیط ماشینی، ویژگی‌های مسئله و تابع هدف آن هستند. برخی نمادهای مورد استفاده برای نمایش مسائل مطرح در ادبیات موضوع نیز بدین صورت است که w_i^k و C_i^k به‌ترتیب وزن و زمان تکمیل کار iام از عامل kام بوده و $C_{\max}^k$ و $L_{\max}^k$ نیز به‌ترتیب بیشینه‌ی زمان تکمیل و بیشینه‌ی مغایرت زمان تکمیل و موعد تحویل کارهای عامل kام هستند. همچنین مقدار U_i^k در صورت دیرکرد کار iام از عامل kام برابر ۱ و در غیر این صورت برابر صفر است. کران بالای مقدار تابع هدف عامل kام نیز با Q_k نشان داده شده است.

اگنتیس و همکاران[۱۱]، مسائل زمان‌بندی دوعاملی با توابع هدف بیشینه‌ی مقدار یک تابع منظم[۱۱]، تعداد کارهای دیرکرددار و مجموع وزنی زمان‌های تکمیل را مورد توجه قرار دادند. آنان سناریوهای مختلفی برای تابع هدف هر عامل و نیز محیط کارگاهی در نظر گرفتند و برای هر سناریو پیچیدگی مسائل مربوطه را مورد بررسی قرار دادند.

در بررسی مسئله‌ی امکان‌پذیری $1||\sum_{1\le i\le n_k} w_i^k U_i^k \le Q_k (k = 1, \dots, K)$، که در آن n_k تعداد کارهای عامل kام، نشان داده شده که این مسئله در حالت کلی به‌شدت NP-complete است.[۱۲] در این مورد، چنانچه تعداد عامل‌ها ثابت باشد در حالت عدد صحیح بودن وزن‌کارها می‌توان آن را در زمان شبه چندجمله‌یی حل کرد و در حالتی که تمامی وزن‌ها مساوی باشد می‌توان مسئله را در زمان چندجمله‌یی حل کرد.

در مطالعه‌ی لیو و تانگ[۱۳] مسئله‌ی زمان‌بندی تک‌ماشین دوعاملی با در نظر گرفتن کارهای زوال‌دار[۱۲] مورد بررسی قرارگرفت. در مطالعه‌ی مذکور با در نظرگرفتن چهار تابع هدف دامنه‌ی عملیات[۱۳]، بیشینه‌ی مغایرت، بیشینه‌ی هزینه و مجموع زمان‌های تکمیل، ویژگی‌های مسائل زمان‌بندی دوعاملی متشکل از ترکیبات مختلف این توابع هدف مورد بررسی قرار گرفت. در مسائل بررسی شده هدف کمینه‌سازی تابع هدف یک عامل، و قراردادن یک کران بالا برای تابع هدف عامل دیگر است. بر این اساس، الگوریتم‌های بهینه‌ی چندجمله‌یی برای دو مسئله‌ی متفاوت ارائه شده است.

اگنتیس و همکاران[۱۴] برای حل سه‌مسئله‌ی $1||\sum w_i^1 C_i^1 : \sum w_i^2 C_i^2 \le Q_2$، $1||\sum w_i^1 C_i^1 : L_{\max}^2 \le Q_2$ و $1||\sum w_i^1 C_i^1 : C_{\max}^2 \le Q_2$ از روش شاخه و کران استفاده کردند و در آن از یک حد پایین برمبنای آزادسازی لاگرانژ بهره بردند. همچنین مسئله‌ی زمان‌بندی دوعاملی $1||\sum w_i^1 C_i^1 + \theta L_{\max}^2$، که تابع هدف آن ترکیب وزنی توابع هدف عامل‌هاست، مورد مطالعه قرارگرفت.[۱۵] از آنجا که این مسئله در دسته مسائل NP-hard قرار دارد برای حل آن از دو روش فراابتکاری الگوریتم ژنتیک و الگوریتم ترکیبی کانگورو و شبیه‌سازی تبرید[۱۴] استفاده شده است.

در مطالعه پیرامون مسئله‌ی زمان‌بندی دوعاملی در محیط فلوشاپ دوماشینه با نماد $F_2||\sum T_i^1 : \sum U_i^2 \le 0$، T_i^1 معرف مقدار زودکرد کار iام از عامل اول است.[۱۶] برای این مسئله یک الگوریتم شاخه و کران و یک الگوریتم شبیه‌سازی تبرید ارائه شده است. همچنین بررسی مسئله‌ی $F_2||\sum C_i^1 : \sum U_i^2 \le 0$ نشان داد که این مسئله‌ی به‌شدت NP-hard است[۱۷] لذا برای حل آن یک الگوریتم شاخه و کران و یک الگوریتم شبیه‌سازی تبرید ارائه شد.

محققین مسئله‌ی زمان‌بندی دوعاملی را با در نظر گرفتن توابع هدف زودکرددار مورد بررسی قرار دادند.[۱۸] دو مسئله‌ی مورد بررسی آنان $1||E_{\max}^1 : E_{\max}^2 \le Q_2$ و $1||\sum w_i^1 E_i^1 : E_{\max}^2 \le Q_2$ بوده که در آنها E_i^k و $E_{\max}^k$ به‌ترتیب مقدار زودکرد کار iام از عامل kام، و بیشینه زودکرد کارهای عامل kام هستند. در مطالعه‌ی یادشده برای مسئله‌ی $1||E_{\max}^1 : E_{\max}^2 \le Q_2$ یک الگوریتم بهینه‌ی چندجمله‌یی ارائه، و نشان داده شد که مسئله‌ی $1||\sum w_i^1 E_i^1 : E_{\max}^2 \le Q_2$ با فرض موعد تحویل مشترک برای تمامی کارها به طور عادی NP-hardبوده و بدون در نظر گرفتن این فرض به‌شدت NP-hard است. همچنین یک الگوریتم ابتکاری کارا برای این مسئله توسعه داده شده است.

در تحقیقی دیگر،[۱۹] مسئله‌ی زمان‌بندی تک‌ماشین دسته‌یی[۱۵] با توابع هدف $\sum C_i^1 : \sum C_i^2 \le Q_2$ ــ در حالتی که تمامی کارها مشابه‌اند و زمان آماده‌سازی به دسته وابسته است ــ مورد تحقیق قرارگرفته است. در آن بررسی فرض بر آن بوده که کارهای متعلق به عامل دوم باید تماماً پشت سر هم انجام شود. برای این مسئله در حالتی که الزامی برای عدد صحیح بودن اندازه هر دسته وجود نداشته باشد، یک الگوریتم بهینه با پیچیدگی $O(n^{3/2})$ ارائه شد و سپس یک روش گردکردن[۱۶] با پیچیدگی $O(n^{1/2})$ برای به دست آوردن یک جواب با اندازه دسته‌های عدد صحیح پیشنهاد شد.

چنان که در ابتدای این بخش نیز اشاره شد از آنجا که در ادبیات موضوع، دو مسئله‌ی OAS و MAS به‌طور همزمان بررسی نشده است، نوآوری ارائه شده در این مقاله عبارت است از در نظر گرفتن عامل‌های مختلف ــ همان مشتریان با

توابع جریمه‌ی مختلف ــ در مسئله‌ی OAS و کاربردی‌تر کردن آن به‌منظور پوشش شرایط با واقعیت بیشتر در محیط‌های تولیدی.

در ادامه، تعریف مسئله و نمادهای مورد نیاز در بخش دوم ارائه شده است. سپس در بخش سوم، با بررسی مسئله یک الگوریتم ابتکاری چندجمله‌یی و یک الگوریتم برنامه‌ریزی پویای شبه چندجمله‌یی توسعه داده شده است. در بخش چهارم نیز نتایج حل الگوریتم‌های توسعه یافته روی مسائل نمونه ارائه و مورد تحلیل قرارگرفته است. نتیجه‌گیری و پیشنهاداتی برای مطالعات آتی نیز در بخش آخر ارائه شده است.

۲. تعریف مسئله و نمادهای مورد نیاز

در این نوشتار فرض شده که دو عامل هریک با تعدادی سفارش در یک محیط تک‌ماشینه با هم رقابت می‌کنند. برای عامل اول هم دیرکرد و هم زودکرد سفارشات اهمیت دارد، به‌گونه‌یی که برای هر واحد دیرکرد سفارش، تولیدکننده مجبور به پرداخت یک واحد جریمه بوده و به‌ازای هر واحد زودکرد نیز یک واحد پاداش از طرف مشتری به تولیدکننده پرداخت می‌شود. این در حالی است که عامل دوم، دیرکرد سفارش‌های خود را به هیچ عنوان نپذیرفته، یا به عبارتی برای سفارشات خود ضرب‌الاجل تعیین کرده است. همچنین فرض شده که سفارشات عامل اول از نظر زمان پردازش مشابه‌اند و مذاکره بین عامل‌ها نیز وجود ندارد. برخی نمادهای مورد استفاده در بیان مسئله عبارت‌اند از:

S: مجموعه‌ی کل سفارش‌ها؛

S_A: مجموعه‌ی سفارش‌های پذیرفته‌شده $(S_A \subseteq S)$؛

n: تعداد کل سفارش‌ها؛

n_k: تعداد سفارش‌های متعلق به عامل k $(\sum_{k=1}^{2} n_k = n)$؛

S_k: مجموعه‌ی سفارش‌های متعلق به عامل k $(\bigcup_{k=1}^{2} S_k = S)$ و $(\bigcap_{k=1}^{2} S_k = \phi)$؛

J_i^k: سفارش i متعلق به عامل k $(k = 1, 2,\ i = 1, ..., n_k)$؛

p_i^k: مدت زمان پردازش J_i^k $(k = 1, 2,\ i = 1, ..., n_k)$؛

p^1: مدت زمان پردازش هر سفارش عامل اول $(\forall\, i : p_i^1 = p^1)$ ؛

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

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

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

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

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

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

U_i^k: اگر برای J_i^k مقدار L_i^k مثبت باشد، این سفارش دیرکرددار بوده و مقدار U_i^k برابر ۱ و در غیر این صورت برابر صفر خواهد بود $(k = 1, 2,\ i = 1, ..., n_k)$؛

$U_i^k(C)$: در صورتی که زمان تکمیل J_i^k برابر C بوده و این سفارش دیرکرددار باشد مقدار ۱ و در غیر این صورت مقدار صفر می‌گیرد $(k = 1, 2,\ i = 1, ..., n_k)$.

مسئله‌ی فوق را می‌توان براساس نمادگذاری انجام‌شده[۱۰] به‌صورت:

$$1|OA, p_i^1 = p^1, \sum U_i^2 = 0|\sum_{k=1}^{2} \sum_{i=1}^{n_k} q_i^k - \sum_{i=1}^{n_1} L_i^1$$

نشان داد. که در آن OA امکان پذیرش یا رد سفارشات است.

۳. بررسی مسئله

مسئله‌ی $1|OA, p_i^1 = p^1, \sum U_i^2 = 0|\sum_{k=1}^{2} \sum_{i=1}^{n_k} q_i^k - \sum_{i=1}^{n_1} L_i^1$ با فرض صفربودن تعداد سفارش‌های عامل اول، و نیز مشترک‌بودن موعد تحویل سفارش‌های عامل دوم، به یک مسئله‌ی کوله‌پشتی کاهش می‌یابد، و لذا پیچیدگی آن دست کم معادل مسئله‌ی کوله‌پشتی و NP-hard به‌هنجار است. اما از آنجا که الگوریتم برنامه‌ریزی پویای ارائه شده برای این مسئله شبه‌چندجمله‌یی است، پیچیدگی آن به‌شدت NP-hard به‌هنجار است.

در ادامه یک الگوریتم ابتکاری و یک الگوریتم برنامه‌ریزی پویا برای این مسئله ارائه می‌شود. اما قبل از ارائه‌ی الگوریتم‌های مذکور، چند لم ارائه شده که در ادامه مورد استفاده قرارگرفته‌اند.

لم ۱: با فرض معلوم بودن مجموعه‌ی سفارشات پذیرفته شده‌ی عامل اول (A_1) و دوم (A_2) و امکان‌پذیری مسئله براساس این مجموعه‌ها، اگر به‌ازای هر سفارش از عامل دوم (نظیر J_i^2) رابطه‌ی $U_i^2 \left(\sum_{i\in A_1} p_i^1 + \sum_{i\in A_2} p_i^2\right) = 1$ برقرار باشد، آنگاه یک توالی بهینه وجود دارد که در آن هر سفارش دلخواهی از عامل اول می‌تواند در انتهای توالی قرارگیرد.

اثبات: واضح است که براساس فرض لم چنانچه سفارشی از عامل دوم در انتهای توالی قرارگیرد، جواب ناممکن می‌شود. لذا حتماً یک توالی امکان‌پذیر وجود دارد که در آن سفارشی از عامل اول مانند J_h^1 در انتهای توالی قرار دارد. مقدار تابع هدف این توالی برابر $\sum_{i\in A_1} q_i^1 + \sum_{i\in A_2} q_i^2 - \sum_{i\in A_1} L_i^1$ و به‌عبارتی معادل $\sum_{i\in A_1} q_i^1 + \sum_{i\in A_2} q_i^2 - \left(\sum_{i\in A_1} C_i^1 - \sum_{i\in A_1} d_i^1\right)$ است. از آنجا که مقادیر $\sum_{i\in A_1} q_i^1$، $\sum_{i\in A_2} q_i^2$ و $\sum_{i\in A_1} d_i^1$ برای هر توالی امکان‌پذیری از سفارشات پذیرفته شده‌ی درون A_1 و A_2 ثابت است، برای یافتن توالی بهینه باید مقدار $\sum_{i\in A_1} C_i^1$ کمینه باشد. اما به‌دلیل برابری زمان پردازش سفارشات عامل اول، با جابه‌جایی هر سفارش دلخواهی از مجموعه‌ی A_1 با J_h^1 بدون بر هم خوردن امکان‌پذیری، مقدار $\sum_{i\in A_1} C_i^1$ ثابت خواهد ماند. لذا اثبات کامل است. ■

لم ۲: با فرض معلوم بودن مجموعه سفارشات پذیرفته‌شده‌ی عامل اول (A_1) و دوم (A_2) و امکان‌پذیری مسئله براساس این مجموعه‌ها، اگر سفارشی از عامل دوم (مانند J_i^2) وجود داشته باشد به‌طوری که $U_i^2 \left(\sum_{i\in A_1} p_i^1 + \sum_{i\in A_2} p_i^2\right) = 0$، آنگاه یک توالی بهینه وجود دارد که این سفارش در آخر آن قرارگیرد.

اثبات: فرض کنید توالی امکان‌پذیر σ' وجود دارد که در آن کار J_i^2 در انتهای توالی نبوده و از این توالی، توالی σ با انتقال سفارش J_i^2 به انتها، به دست آید. برای هر سفارش J_h^k $(k = 1, 2,\ h = 1, ..., n_k)$ غیر از J_i^2 رابطه‌ی $C_h^k(\sigma) \leq C_h^k(\sigma')$ همواره وجود دارد، یعنی زمان تکمیل سایر سفارشات در توالی σ کوچک‌تر یا مساوی توالی σ' بوده و سفارش J_i^2 در هر دو توالی مذکور بدون دیرکرد است، و از این رو امکان‌پذیری توالی σ نیز برقرار است. ■

در لم سوم یک حد بالا برای حالتی که تعدادی از سفارشات عامل اول از قبل تعیین تکلیف شده، و نیز یک حد بالا برای حالتی که در آن تنها تعدادی از سفارشات عامل دوم تعیین تکلیف نشده‌اند، ارائه شده است.

لم ۳ (حد بالای ۱): با فرض این‌که مجموعه‌ی سفارشات پذیرفته شده و رد شده‌ی عامل اول به‌ترتیب A_1 و R_1 باشد، آنگاه مقدار:

$$\sum_{i\in A_1}\left(q_i^1 - L_i^1\right) + \sum_{i\in S_1 - \{A_1 \cup R_1\}}\left(q_i^1 - \min\left\{0, (|A_1| + 1)p^1 - d_i^1\right\}\right) + \sum q_i^2$$

یک حد بالا برای مقدار تابع هدف مسئله مورد بررسی است، که در آن $|A_1|$ تعداد سفارشات درون مجموعه‌ی A_1 است.

اثبات: قسمت اول از عبارت فوق $\sum_{i\in A_1}(q_i^1 - L_i^1)$ برابر مجموع سود سفارشات از قبل پذیرفته شده‌ی عامل اول است. قسمت دوم $(\sum_{i\in S_1-\{A_1\cup R_1\}}(q_i^1 - \min\{0, (|A_1|+1)p^1 - d_i^1\}))$ نیز شامل دو جزء است که جزء اول مجموع درآمد سفارش‌های تعیین تکلیف نشده از عامل اول بوده و در جزء دوم $(|A_1|+1)p^1 - d_i^1$ برابر با مغایرت زمان تکمیل و موعد تحویل سفارش iام از سفارشات تعیین تکلیف نشده‌ی عامل اول است که در صورت منفی بودن از درآمد کسر می‌شود. همچنین قسمت سوم عبارت فوق $(\sum_{i\in S_2} q_i^2)$ برابر مجموع درآمد کلیه‌ی سفارشات عامل دوم است. لذا این عبارت شامل مجموع سود سفارشات پذیرفته شده، درآمد تمامی سفارشات تعیین تکلیف نشده و پاداش احتمالی حاصل از سفارشات تعیین تکلیف نشده‌ی عامل اول در صورت منفی بودن مغایرت آنها است. برای هر سفارش i درون مجموعه‌ی، در صورتی که سفارش $S_1 - \{A_1 \cup R_1\}$ در جواب بهینه وجود نداشته باشد، مقدار $\min\{0, (|A_1|+1)p^1 - d_i^1\}$ به‌طور ضمنی به حد بالا افزوده شده است و در صورتی که سفارش i در جواب بهینه باشد، مقدار مجموع زمان تکمیل سفارشات $A_1 \cup \{i\}$ در جواب بهینه، یعنی $\sum_{j\in A_1} C_j^1 + C_i^1$، کم‌تر از $\sum_{j\in A_1} C_j^1 + (|A_1|+1)p^1$ نخواهد بود. لذا مقدار پاداش در نظرگرفته شده برای هر سفارش i درون مجموعه‌ی $S_1 - \{A_1 \cup R_1\}$ در عبارت فوق بزرگ‌تر یا مساوی معادل آن در جواب بهینه است. بر این اساس می‌توان نتیجه گرفت که عبارت فوق یک حد بالا برای مسئله با فرض‌های ذکر شده است. ■

لم ۴ (حد بالای ۲): با فرض این‌که مجموعه‌ی سفارشات پذیرفته شده‌ی عامل اول و دوم به‌ترتیب A_1 و A_2 بوده و تنها تعدادی از سفارشات عامل دوم (مجموعه‌ی RS_2) تعیین تکلیف نشده باشد؛ آنگاه مقدار $\sum_{i\in A_1}(q_i^1 - L_i^1) + \sum_{i\in A_2} q_i^2 + \sum_{i\in RS_2} q_i^2$ یک حد بالا برای تابع هدف مسئله‌ی مورد بررسی است.

اثبات: دو جزء اول از عبارت فوق برابر مقدار تابع هدف حاصل از سفارشات پذیرفته شده‌ی هر دو عامل است. بدیهی است که با افزودن مقدار درآمد سفارشات باقی‌مانده از عامل دوم که تنها سفارشات تعیین تکلیف نشده‌اند، این مقدار بزرگ‌تر یا مساوی مقدار بهینه‌ی تابع هدف خواهد بود. ■

۱.۳. الگوریتم SAO برای زمان‌بندی سفارشات پذیرفته شده

براساس لم‌های ۱ و ۲، و با فرض معلوم بودن مجموعه سفارشات پذیرفته‌شده‌ی عامل اول و دوم، توالی بهینه را می‌توان از الگوریتم «ترتیب‌گذاری سفارشات پذیرفته شده (SAO)[17]» پیدا کرد. در این الگوریتم سفارش‌ها از انتهای توالی چیده شده و در هر مرحله درمورد سفارشی که در انتهای توالی قرار می‌گیرد براساس دو لم مذکور تصمیم‌گیری می‌شود. گام‌های این الگوریتم عبارت است از:

گام ۱: توالی جواب را σ نامیده و قرار دهید $\sigma = \phi$.

- سفارشات پذیرفته شده‌ی عامل اول را به ترتیب دلخواه مرتب کرده و با σ_{arb} نشان دهید.
- سفارشات پذیرفته شده‌ی عامل دوم را به‌ترتیب غیرنزولی موعد تحویل (EDD)[18] مرتب کرده و با σ_{EDD} نشان دهید.
- قرار دهید $\tau = \sum_{i\in\sigma_{arb}} p_i^1 + \sum_{i\in\sigma_{\mathrm{EDD}}} p_i^2$.

گام ۲: در صورتی که در توالی σ_{EDD} سفارش دیرکرداری وجود دارد به گام ۷ بروید.

گام ۳: اگر $\tau = 0$ به گام ۸ بروید، در غیر این صورت به گام ۴ بروید.

گام ۴: اگر σ_{EDD} تهی است به گام ۵ بروید؛ در غیر این صورت سفارش انتهای σ_{EDD} را J_h^2 بنامید. اگر $U_h^2(\tau) = 0$، آنگاه این سفارش را از σ_{EDD} حذف کرده و آن را سفارش فعال بنامید و به گام ۶ بروید. در غیر این صورت به گام ۵ بروید.

گام ۵: اگر σ_{arb} تهی است به گام ۸ بروید؛ در غیر این صورت سفارش انتهای σ_{arb} را برابر سفارش فعال قرار داده و پس از حذف آن از σ_{arb} به گام ۶ بروید.

گام ۶: سفارش فعال را در توالی σ قبل از سایر سفارش‌ها زمان‌بندی کنید. از مقدار τ به‌اندازه‌ی زمان پردازش سفارش فعال کم کنید و به گام ۳ بروید.

گام ۷: مسئله امکان‌پذیر نیست؛ به گام ۹ بروید.

گام ۸: توالی σ جواب بهینه‌ی مسئله است.

گام ۹: پایان.

چنان که در گام‌های فوق مشخص است، در هر مرحله‌ی الگوریتم یک سفارش از بین سفارش‌های زمان‌بندی نشده انتخاب، و در انتهای توالی قبل از سفارش‌های زمان‌بندی شده قرار می‌گیرد. در صورت حفظ امکان‌پذیری سفارشی از عامل دوم انتخاب می‌شود و در غیر این صورت به‌دلخواه سفارشی از عامل اول انتخاب می‌شود. اگر تمامی سفارش‌های عامل اول زمان‌بندی شده باشد و سفارشی از عامل دوم را نتوان در انتهای توالی جواب، قبل از سفارش‌های زمان‌بندی شده قرار داد، مسئله امکان‌ناپذیر است. پیچیدگی زمانی گام ۱ الگوریتم فوق به‌دلیل مرتب‌کردن سفارشات عامل دوم برابر $O(n_2 \log n_2)$ است و گام‌های ۵ تا ۷ الگوریتم نیز حداکثر $n_1 + n_2$ بار تکرار می‌شود. لذا درکل پیچیدگی الگوریتم فوق $O(n_1 + n_2 \log n_2)$ است. از این روش ساده در الگوریتم ابتکاری و برنامه‌ریزی پویای ارائه شده برای حل مسئله استفاده شده است.

۲.۳. الگوریتم ابتکاری SOT

در این قسمت الگوریتم ابتکاری[19] SOT برای حل مسئله ارائه شده است. در این الگوریتم ابتدا سفارشات عامل اول به‌ترتیب غیرصعودی از مقدار درآمد ضرب‌در موعد تحویل وارد شده و به انتهای توالی جواب افزوده می‌شود؛ سپس چنانچه مقدار تابع هدف افزایش نیابد سفارش وارد شده از توالی حذف می‌شود. همین روند برای سفارشات عامل دوم با ترتیب ورود EDD به اجرا درآمده و در نهایت توالی جواب به‌عنوان خروجی ارائه می‌شود. گام‌های این الگوریتم عبارت‌اند از:

گام ۱: سفارش‌های عامل اول را به‌ترتیب غیرصعودی از مقدار درآمد ضرب‌در موعد تحویل مرتب کرده و در مجموعه‌ی S_1 قرار دهید؛ سفارش‌های عامل دوم را به‌ترتیب EDD در مجموعه‌ی S_2 قرار دهید. توالی جواب را σ بنامید و قرار دهید $\sigma = \phi$.

گام ۲: در صورتی که $S_1 = \phi$ به گام ۴ بروید؛ در غیر این صورت اولین سفارش از S_1 را O^1 نامیده و آن را از S_1 حذف کنید؛ سپس به گام ۳ بروید.

گام ۳: سفارش O^1 را به انتهای توالی σ بیافزایید. اگر مقدار تابع هدف افزایش یافت توالی جدید را حفظ کنید و در غیر این صورت سفارش O^1 را از توالی σ حذف کنید. به گام ۲ بروید.

گام ۴: در صورتی که $S_2 = \phi$ به گام ۷ بروید و در غیر این صورت اولین سفارش از S_2 را O^2 نامیده و آن را از S_2 حذف کنید. سپس به گام ۵ بروید.

گام ۵: اگر با تشکیل ترتیب EDD از سفارشات عامل دوم در توالی σ و سفارش O^2، دست کم یکی از سفارشات دیرکردار شد، به گام ۴ و در غیر این صورت به گام ۶ بروید.

گام ۶: سفارش O^2 را به توالی σ افزوده و پس از اجرای الگوریتم SAO روی

این توالی، اگر مقدار تابع هدف افزایش یافت توالی جدید را حفظ کنید، در غیر این صورت سفارش O^2 را از توالی σ حذف کنید. به گام ۲ بروید.
گام ۷: توالی σ را به‌عنوان جواب ارائه دهید.
گام ۸: پایان.

پیچیدگی گام ۱ الگوریتم SOT به‌دلیل مرتب‌کردن سفارشات هر عامل برابر با $O(n_1 \log n_1 + n_2 \log n_2)$ است. گام‌های ۲ تا ۳ نیز دارای پیچیدگی $O(n_1)$ هستند. همچنین گام‌های ۴ تا ۶ به‌دلیل اجرای الگوریتم SAO حداکثر که تعداد n_2 مرتبه، و نیز به‌دلیل در اختیار داشتن ترتیب EDD سفارشات عامل دوم در ابتدای الگوریتم برابر با $O(n_2(n_1 + n_2))$ است. لذا پیچیدگی کل الگوریتم $O((n_2)^2 + n_1 n_2 + n_1 \log n_1)$ یا به‌طور خلاصه $O(n^2)$ است.

۳.۳. الگوریتم برنامه‌ریزی پویای DP۱

در اینجا یک برنامه‌ریزی پویای شبه چندجمله‌یی برای مسئله‌ی $1|OA, p_i^1 = p^1, \sum U_i^2 = 0|\sum_{k=1}^{2}\sum_{i=1}^{n_k} q_i^k - \sum_{i=1}^{n_1} L_i^1$ با نام DP۱ ارائه می‌شود. یادآوری می‌شود که این الگوریتم اساساً با دو الگوریتم برنامه‌ریزی پویای ارائه شده توسط قوش[3] متفاوت است و به‌دلیل همین تفاوت ساختار، این الگوریتم کاملاً جدید و بر مبنای ویژگی‌های مسئله‌ی فعلی توسعه داده شده است. الگوریتم DP۱ دارای دو فاز است و به‌طور کلی می‌توان آن را چنین بیان کرد که در فاز اول سفارش‌های عامل اول به‌ترتیب غیرصعودی از موعد تحویل (LDD)[20] وارد شده و فضای حالت شماره ۱ تشکیل و براساس حدود بالا و پایین و اصول غلبه‌ی ارائه شده به‌روز می‌شود. سپس در فاز دوم به‌ازای هریک از حالت‌های نهایی فاز اول، سفارش‌های عامل دوم به‌ترتیب EDD وارد شده و یک فضای حالت موقت به‌نام فضای حالت شماره ۲ تشکیل شده و براساس حدود بالا و پایین و اصل غلبه‌ی مورد نظر به‌روز می‌شود. بهترین حالت امکان‌پذیر از فضای حالت شماره ۲ انتخاب شده و این روند مجدداً برای دیگر حالت‌های فضای حالت فاز اول ادامه می‌یابد. در پایان هر مرحله، بهترین حالت به‌روز می‌شود. هر جواب مسئله، مرتبط با یک حالت است که به‌صورت: $(S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, C^2_{\max\_apt}, P^2_{sum\_apt})$ نشان داده می‌شود و اجزای آن عبارت‌اند از:

S_{apt}: مجموعه‌ی سفارش‌های پذیرفته شده؛
Z_{apt}: مجموع درآمد حاصل از سفارش‌های پذیرفته شده؛
n_1^{apt}: تعداد سفارش‌های پذیرفته‌شده‌ی عامل اول؛
$L^1_{sum\_apt}$:مجموع مغایرت سفارش‌های پذیرفته شده‌ی عامل اول؛
$C^2_{\max\_apt}$:بیشترین زمان تکمیل سفارشات پذیرفته‌شده‌ی عامل دوم؛
$P^2_{sum\_apt}$:مجموع زمان پردازش سفارشات پذیرفته‌شده‌ی عامل دوم.

گام‌های الگوریتم DP۱ نیز عبارت‌است از:

گام ۱: سفارش‌های عامل اول را به‌ترتیب LDD در مجموعه‌ی S_1 قرار دهید. سفارش‌های عامل دوم را به‌ترتیب EDD در مجموعه‌ی S_2 قرار دهید. بهترین جواب فعلی را برابر با جواب الگوریتم ابتکاری SOT در نظر بگیرید. حالت $(\phi, 0, 0, 0, 0, 0)$ را در فضای حالت شماره‌ی ۱ قرار دهید.
گام ۲: در صورتی که $S_1 = \phi$ به گام ۵ بروید و در غیر این صورت سفارش ابتدایی S_1 را j نامیده و آن را از S_1 حذف کرده و به گام ۲-۱ بروید.
گام ۲-۱ (محاسبه‌ی حد بالای ۱): برای هر حالت مانند $(S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, 0, 0)$ موجود در فضای حالت شماره ۱، سفارش j را به S_{apt} افزوده و سایر مقادیر را برای حالت جدید محاسبه کنید. اگر مقدار حد بالای حاصل از لم ۳ بزرگ‌تر از مقدار تابع هدف بهترین جواب فعلی است حالت جدید را به فضای حالت شماره‌ی ۱ بیافزایید.
گام ۲-۲ (اصل غلبه‌ی ۱): برای هر دو حالت مانند $v = (S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, 0, 0)$ و $v' = (S'_{apt}, Z'_{apt}, n_1^{apt'}, L^{1'}_{sum\_apt}, 0, 0)$ موجود در فضای حالت شماره‌ی ۱، در صورتی که شرایط $Z_{apt} \geq Z'_{apt}$، $n_1^{apt} \leq n_1^{apt'}$ و $L^1_{sum\_apt} \leq L^{1'}_{sum\_apt}$ برقرار باشند، آنگاه کافی است تنها حالت v در فضای حالت شماره‌ی ۱ نگه داشته شود. این اصل غلبه را برای فضای حالت شماره‌ی ۱ بررسی کرده و پس از به‌روزکردن آن به گام ۲ بروید.
گام ۳: تعداد کل حالت‌های موجود در فضای حالت شماره‌ی ۱ را برابر NSI در نظرگرفته و قرار دهید $k = 1$.
گام ۴: اگر $k > NSI$ به گام ۷ بروید و در غیر این صورت حالت kام از فضای حالت شماره‌ی ۱ را در فضای حالت شماره‌ی ۲ قرار داده، و با در نظرگرفتن $S'_2 = S_2$ و به گام ۵ بروید.
گام ۵: در صورتی که $S'_2 = \phi$ به گام ۶ بروید و در غیر این صورت سفارش ابتدای $S'_2 = \phi$ را j نامیده و آن را از S'_2 حذف کرده و به گام ۵-۱ بروید.
گام ۵-۱: برای هر حالت موجود در فضای حالت شماره ۲ مانند $(S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, C^2_{\max\_apt}, P^2_{sum\_apt})$، سفارش j را به S_{apt} افزوده و الگوریتم SAO را برای یافتن توالی امکان‌پذیر روی مجموعه‌ی $S_{apt} \cup \{j\}$ اجرا کنید. اگر توالی حاصل امکان‌پذیر است، مقادیر مربوط به حالت جدید را حساب کرده و به گام ۵-۲ بروید.
گام ۵-۲ (محاسبه‌ی حد بالای ۲): مطابق لم ۴ مقدار حد بالا را برای حالت جدید محاسبه کنید. اگر مقدار این حد بالا کوچک‌تر یا مساوی مقدار تابع هدف بهترین جواب فعلی است، قرار دهید $k = k + 1$ و به گام ۴ بروید. در غیر این صورت به گام ۵-۳ بروید.
گام ۵-۳ (اصل غلبه‌ی ۲): برای هر دو حالت مانند:

$$v = (S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, C^2_{\max\_apt}, P^2_{sum\_apt})$$
$$v' = (S'_{apt}, Z'_{apt}, n_1^{apt}, L^{1'}_{sum\_apt}, C^{2'}_{\max\_apt}, P^{2'}_{sum\_apt})$$

در فضای حالت شماره ۲، اگر شرایط $Z_{apt} \geq Z'_{apt}$، $L^1_{sum\_apt} \leq L^{1'}_{sum\_apt}$، $P^2_{sum\_apt} \leq P^{2'}_{sum\_apt}$ و $C^2_{\max\_apt} \leq C^{2'}_{\max\_apt}$ برقرار باشد، آنگاه کافی است تنها حالت v در فضای حالت شماره ۲ نگه داشته شود. این اصل غلبه را برای فضای حالت شماره‌ی ۲ بررسی کرده و پس از به‌روزکردن آن به گام ۵ بروید.
گام ۶: در صورتی که بهترین جواب موجود در فضای حالت شماره ۲ بهتر از بهترین جواب فعلی است آن را جایگزین بهترین جواب فعلی کنید. با در نظرگرفتن $k = k + 1$، فضای حالت شماره ۲ را تهی کرده و به گام ۴ بروید.
گام ۷: بهترین جواب فعلی را به‌عنوان خروجی در نظر بگیرید.
گام ۸: پایان.

اصول غلبه‌ی به‌کار رفته در گام‌های ۲-۲ و ۵-۳ الگوریتم DP۱ در قالب لم‌هایی که در ادامه آمده‌اند، مطرح شده و به اثبات رسیده‌اند.

لم ۵ (اصل غلبه‌ی ۱): در صورتی که در الگوریتم DP۱ در فضای حالت شماره ۱ دو حالت $v = (S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, 0, 0)$ و $v' = (S'_{apt}, Z'_{apt}, n_1^{apt'}, L^{1'}_{sum\_apt}, 0, 0)$ با شرایط زیر وجود داشته باشند، می‌توان بدون از دست دادن جواب بهینه حالت v' را حذف کرد.

$$Z_{apt} \geq Z'_{apt} \qquad (1)$$

$$n_1^{apt} \le n_1^{apt'} \tag{۲}$$

$$L^1_{sum\_apt} \le L^{1'}_{sum\_apt} \tag{۳}$$

اثبات: اگر با حذف حالت v' از فضای حالت شماره ۱، جواب بهینه از دست برود، این بدان معناست که این حالت منجر به رسیدن به جواب بهینه می‌شود. از طرفی از آنجا که در الگوریتم DP۱، هر دو حالت فوق در فاز اول ــ یعنی زمانی که تنها سفارشات عامل اول وارد شده‌اند ــ مورد بررسی قرار می‌گیرد، برای رسیدن به جواب بهینه در گام‌های بعدی الگوریتم، هم می‌توان سفارش‌های باقی‌مانده‌ی عامل اول و هم سفارش‌های عامل دوم را وارد کرد. در این صورت، مجموعه‌ی سفارش‌های عامل اول و دوم که باید برای حصول جواب بهینه به حالت v' افزود به ترتیب OS^1 و OS^2 فرض می‌شود.

از آنجا که جواب بهینه، حتماً یک جواب امکان‌پذیر است و چون فرض شده که جواب بهینه از حالت v' به دست می‌آید؛ با افزودن مجموعه‌های OS^1 و OS^2 به این حالت جواب حاصل امکان‌پذیر است. اگر فرض شود که $\sum_{i\in OS^1} C_i^{1'}$ مجموع زمان تکمیل سفارش‌های مجموعه‌ی OS^1 باشد، وقتی که این مجموعه به سفارشات پذیرفته شده در حالت v' افزوده شود، به‌دلیل امکان‌پذیری جواب بهینه، داریم:

$$L^{1'}_{sum\_apt} + \left(\sum_{i\in OS^1} C_i^{1'} - \sum_{i\in OS^1} d_i^1\right) \le Q_1 \tag{۴}$$

اگر سفارش‌های مجموعه‌ی OS^1 به همان ترتیبی که به حالت v' افزوده شدند به حالت v افزوده شود چون این سفارشات بعد از سفارشات چیده شده از قبل قرار دارند و به‌دلیل برقراری رابطه‌ی ۲ مجموع زمان‌پردازش سفارشات پذیرفته شده‌ی حالت v کم‌تر از حالت v' است. لذا رابطه‌ی $\sum_{i\in OS^1} C_i^1 \le \sum_{i\in OS^1} C_i^{1'}$ برقرار است. از این رو با توجه به روابط ۳ و ۴ داریم:

$$L^1_{sum\_apt} + \left(\sum_{i\in OS^1} C_i^1 - \sum_{i\in OS^1} d_i^1\right) \le Q_1 \tag{۵}$$

بر اساس رابطه‌ی ۵ جواب به دست آمده از حالت v نیز با افزودن سفارش‌های مجموعه‌ی OS^1 به آن، از نظر عامل اول امکان‌پذیر است. اما با افزودن سفارش‌های مجموعه‌ی OS^2 به حالت v' به‌دلیل فرض بهینگی این جواب هیچ سفارشی از عامل دوم در آن دیرکرد نخواهد داشت، و افزودن این سفارشات امکان‌پذیری عامل اول را به هم نمی‌ریزد. باز به‌دلیل برقراری روابط ۲ و ۳، و امکان‌پذیری جواب حاصل از v'، جواب به دست آمده از افزودن سفارش‌های مجموعه‌ی OS^2 به حالت v نیز امکان‌پذیر خواهد بود. همچنین با توجه به برقراری روابط ۱ و ۳ می‌توان نوشت:

$$Z_{apt} - L^1_{sum\_apt} + \sum_{k=1}^{2}\sum_{i\in OS^k} q_i^k \ge Z'_{apt} - L^{1'}_{sum\_apt} + \sum_{k=1}^{2}\sum_{i\in OS^k} q_i^k \tag{۶}$$

لذا هر دو جواب به دست آمده از حالت v و v' امکان‌پذیرند، اما مقدار تابع هدف جواب به دست آمده از حالت v کم‌تر نیست، که این با فرض بهینه بودن جواب به دست آمده از حالت v' در تناقض است. از این رو فرض خلف باطل بوده و حذف حالت v' منجر به از دست رفتن جواب بهینه نمی‌شود. ■

لم ۶ (اصل غلبه‌ی ۲): چنانچه دو حالت:

$$v = (S_{apt}, Z_{apt}, n_1^{apt}, L^1_{sum\_apt}, C^2_{\max\_apt}, P^2_{sum\_apt})$$
$$v' = (S_{apt'}, Z'_{apt}, n_1^{apt}, L^{1'}_{sum\_apt}, C^{2'}_{\max\_apt}, P^{2'}_{sum\_apt})$$

در فضای حالت شماره ۲ با شرایط زیر وجود داشته باشد، می‌توان بدون از دست دادن جواب بهینه حالت v' را حذف کرد.

$$Z_{apt} \ge Z'_{apt} \tag{۷}$$

$$L^1_{sum\_apt} \le L^1_{sum\_apt'} \tag{۸}$$

$$P^2_{sum\_apt} \le P^2_{sum\_apt'} \tag{۹}$$

$$C^2_{\max\_apt} \le C^2_{\max\_apt'} \tag{۱۰}$$

اثبات: مشابه اثبات لم ۵ است. ■

به‌دلیل اصل غلبه‌ی استفاده شده درگام ۲-۲ الگوریتم، تعداد حالات امکان‌پذیر موجود در فضای حالت شماره ۱ حداکثر برابر $(n_1 p^1)(n_1)(n_1 p^1)$ است که در آن $(n_1 p^1)$ بیشینه دامنه‌ی تغییرات $L^1_{sum\_apt}$ در فاز اول است. لذا پیچیدگی کل گام ۲ الگوریتم برابر $O(n_1(n_1)(n_1 p^1))$ است. همچنین از آنجا که درگام‌های ۴ تا ۶ برای هر یک از حالت‌های موجود در فضای حالت شماره ۱ به نوعی یک الگوریتم برنامه‌ریزی پویا انجام می‌شود که پیچیدگی آن به‌ازای هر حالت موجود در فضای حالت ۱، به‌علت استفاده از اصل غلبه‌ی گام ۵-۳ و اجرای الگوریتم SAO برای هر حالت درگام ۵-۱، برابر $O(n_2(n_1+n_2)(P^2_{sum})(n_1 p^1 + P^2_{sum})^2)$ است که در آن $(n_1 p^1 + P^2_{sum})$ دامنه‌ی تغییرات $L^1_{sum\_apt}$ و $C^2_{\max\_apt}$ در فاز دوم بوده و (n_1+n_2) پیچیدگی اجرای الگوریتم SAO برای هر حالت است. یادآور می‌شود در اینجا برای محاسبه‌ی پیچیدگی الگوریتم SAO به‌دلیل در دسترس بودن ترتیب EDD سفارش‌های عامل دوم در الگوریتم DP۱، نیازی به در نظرگرفتن پیچیدگی گام ۱ الگوریتم SAO نیست. لذا پیچیدگی کل را می‌توان با ضرب بیشینه تعداد حالات فضای حالت ۱ در پیچیدگی گام‌های ۴ تا ۶ به دست آورد و آن را به‌صورت $O(n_1^2 n_2(n_1+n_2)p^1 P^2_{sum}(n_1 p^1 + P^2_{sum})^2)$ یا به‌طور خلاصه به‌صورت $O(n^3(P_{sum})^4)$ بیان کرد. پس الگوریتم DP۱ دارای پیچیدگی شبه چندجمله‌یی است و چنان که در ابتدای این قسمت ذکر شد، مسئله NP-hard به‌هنجار است.

در صورتی که زمان پردازش تمامی سفارش‌های هر دو عامل برابر واحد باشد آنگاه مسئله توسط الگوریتم DP۱ در زمان چندجمله‌یی و با پیچیدگی $O(n_1^2 n_2^2(n_1+n_2)^3)$، یا به‌طور خلاصه $O(n^7)$، حل می‌شود.

۴. نتایج محاسباتی

به‌منظور بررسی کارایی الگوریتم‌های پیشنهادی، سعی شده تا کیفیت آنها در حل مسائل نمونه مورد بررسی قرار گیرد. این الگوریتم‌ها در محیط برنامه‌نویسی Visual C# ۲۰۱۰ پیاده‌سازی شده و روی یک دستگاه رایانه با مشخصات Intel® Core™ i۷-۲۶۰۰ CPU ۳٫۴GHz و ۴ GB RAM در محیط سیستم عامل Windows ۷ به اجرا گذاشته شده است. همچنین فاز دوم الگوریتم DP۱ به‌طور موازی کدنویسی شده و محدودیت زمان حل نیز در تمامی مسائل برابر ۳۶۰۰ ثانیه در نظرگرفته شده است.

نظر به جدید بودن مسئله‌ی پذیرش و زمان‌بندی سفارش‌های دو عاملی، سعی شده تا با الگوگیری از نحوه‌ی تولید مسائل نمونه در ادبیات موضوع پذیرش و زمان‌بندی

سفارش‌ها، و نیز مطالعات گذشته در زمینه‌ی زمان‌بندی چندعاملی و اعمال تغییراتی بر اساس شرایط مسئله‌ی جدید، نسبت به طراحی مسائل نمونه اقدام شود. در مطالعه‌ی حاضر زمان‌های پردازش عامل دوم از توزیع یکنواخت گسسته در دو بازه [۱،۱۰] مطابق مطالعات نوبیبون و لئوس[۶] و [۱،۱۰۰] براساس ادبیات موضوع زمان‌بندی چندعاملی[۱۵-۱۷] تولید شده و زمان پردازش عامل اول نیز برابر دو مقدار ۵ و ۵۰ در نظر گرفته شده است. موعد تحویل هر سفارش مانند J_i^k نیز مانند مطالعات قبلی[۶و۱۵-۱۷] از توزیع یکنواخت گسسته‌ی $\max\{p_i^k, U[P_{sum}(1-\tau-R/2), P_{sum}(1-\tau+R/2)]\}$ تولید شده، که در آن τ و R به‌ترتیب عامل دیرکرد و عامل پراکندگی موعد تحویل‌اند. مقادیر در نظر گرفته شده برای پارامترهای مسئله و سطوح مربوط به آنها در جدول ۱ نشان داده شده است. از آنجا که آزمایشات مقدماتی بر مبنای «طرح آزمایش فاکتوریل کامل[۲۱]» نشان دادند که درآمد سفارشات عامل اول و مقدار τ و R مربوط به موعد تحویل‌شان در عملکرد روش‌های ارائه‌شده تأثیر معناداری ندارند لذا در آزمایشات محاسباتی صورت گرفته در این مطالعه، سطوح مختلفی برای آنها در نظر گرفته نشده است.

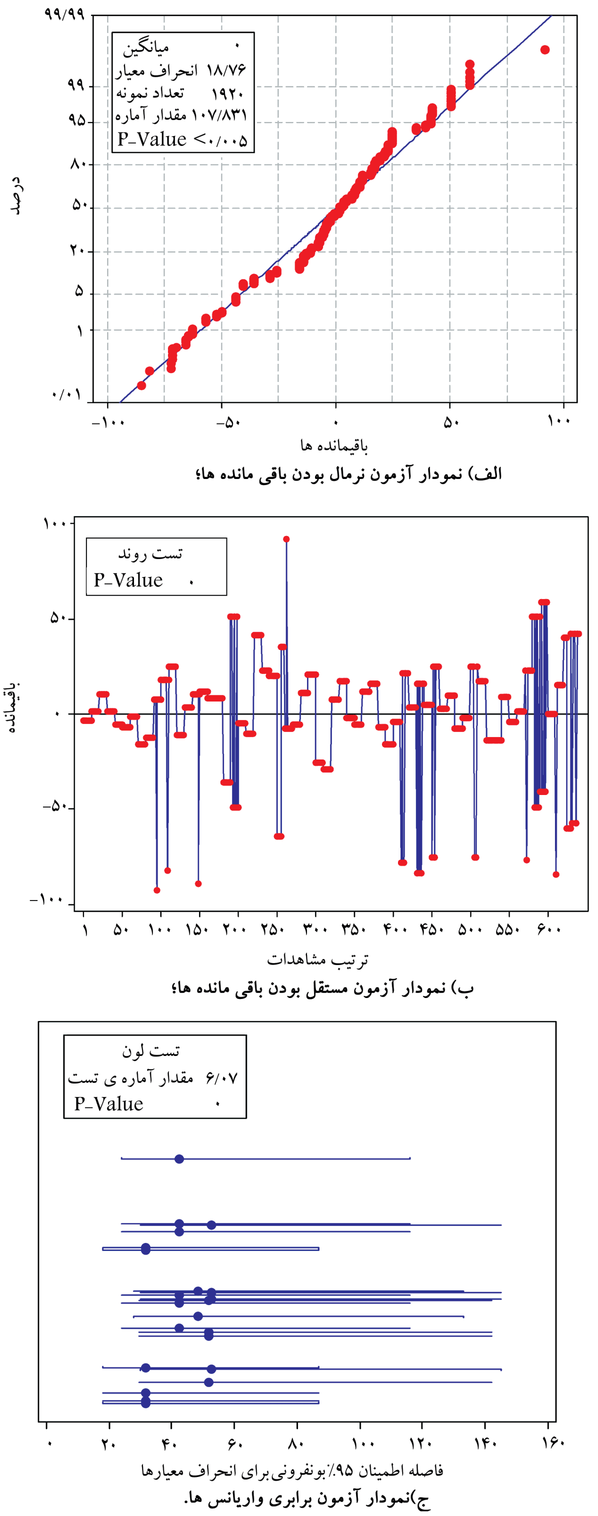
برای بررسی عملکرد الگوریتم DP۱ یک طرح آزمایش فاکتوریل کامل انجام شده است. لذا براساس پارامترهای: تعداد سفارش عامل اول، درآمد عامل دوم، زمان پردازش هر عامل، τ و R عامل دوم، تعداد ۶۴ $(2\times2\times2\times2\times2\times2)$ گروه مختلف با نام‌های G۰۱ تا G۶۴ شکل می‌گیرد که به‌ازای هرگروه سه اندازه مسئله با ۶۰، ۱۰۰ و ۱۵۰ سفارش در نظر گرفته شده و برای هر ترکیب مختلف از گروه و اندازه، ۱۰ مسئله‌ی نمونه تولید و حل شده است.

برای بررسی تأثیر پارامترهای در نظر گرفته شده از تحلیل واریانس[۲۲] استفاده شده است. لذا با توجه به تعداد گروه‌ها و اندازه‌های مختلف مسائل، ۱۹۲ (3×64) تیمار[۲۳] در نظر گرفته شده است. قابل ذکر است که سه پیش‌فرض اصلی تحلیل واریانس ــ شامل نرمال بودن و مستقل بودن باقی‌مانده‌ها، و برابری واریانس‌ها ــ مورد بررسی قرار گرفته و همه‌ی این پیش‌فرض‌ها پذیرفته شده‌اند (شکل ۱).

نتیجه‌ی تحلیل واریانس در جدول ۲ برای متغیر پاسخ «درصد مسائل نمونه‌ی بهینه‌ی حل شده در ۳۶۰۰ ثانیه» به نمایش درآمده است. براساس مقادیر P-Value در جدول ۲، تمامی پارامترها در سطح اطمینان[۲۴] ۵ درصد معنادار هستند. همچنین اثرات متقابل[۲۵] پارامترها تنها در ۱۳ مورد (از ۲۸ مورد) در سطح اطمینان ۵ درصد

جدول ۱. مشخصات پارامترهای استفاده شده در تولید مسائل نمونه.

نام پارامتر	نماد	مقدار	سطح
اندازه	n	۸۰، ۱۰۰ و ۱۵۰	۸۰، ۱۰۰ و ۱۵۰
تعداد سفارش عامل اول	$n1$	$2n_1 = n_2$ $n_1 = 2n_2$	کوچک بزرگ
درآمد عامل دوم	qi_2	$[1, 2\times p_i^2]$ $[1, 20\times p_i^2]$	کوچک بزرگ
زمان پردازش عامل اول	p_1	۵ ۵۰	کوچک بزرگ
زمان پردازش عامل دوم	pi_2	$[1,10]$ $[1,100]$	کوچک بزرگ
مقدار τ عامل دوم	tou_2	۰٫۳ ۰٫۷	کوچک بزرگ
مقدار R عامل دوم	R_2	۰٫۲ ۰٫۶	کوچک بزرگ



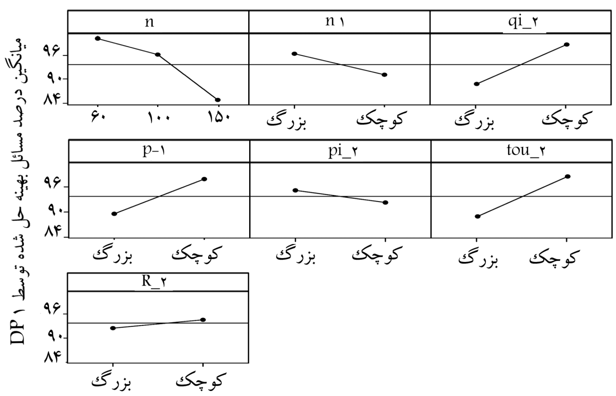
شکل ۱. نمودارهای مربوط به بررسی سه پیش فرض اصلی تحلیل واریانس.

معنادار است. در شکل ۲ نمودار اثرات اصلی پارامترها بر متغیر پاسخ (درصد مسائل بهینه‌ی حل شده) به نمایش درآمده است. بر اساس این شکل اکثر پارامترها اثرات قابل توجهی بر روی میانگین درصد مسائل نمونه‌ی بهینه حل شده در ۳۶۰۰ ثانیه دارند.

نتایج حل برای گروه‌های G۰۱ تا G۶۴، شامل «تعداد نمونه‌ی بهینه» و «میانگین

جدول ۲. نتیجه‌ی تحلیل واریانس برای درصد مسائل حل شده توسط DP۱.

منبع	مجموع مربعات	درجه‌ی آزادی	میانگین مربعات	F	P-Value
Mode	۴۶۶۹۸۹٫۵۸۳	۳۵	۱۳۳۴۲٫۵۶۰	۳۷٫۲۱۴	۰٫۰۰۰
n	۷۹۲۶۰٫۴۱۷	۲	۳۹۶۳۰٫۲۰۸	۱۱۰٫۵۳۲	۰٫۰۰۰
n_1	۱۳۰۲۰٫۸۳۳	۱	۱۳۰۲۰٫۸۳۳	۳۶٫۳۱۶	۰٫۰۰۰
q_i^2	۴۸۰۰۰٫۰۰۰	۱	۴۸۰۰۰٫۰۰۰	۱۳۳٫۸۷۶	۰٫۰۰۰
p^1	۳۳۳۳۳٫۳۳۳	۱	۳۳۳۳۳٫۳۳۳	۹۲٫۹۷۰	۰٫۰۰۰
p_i^2	۳۵۲۰٫۸۳۳	۱	۳۵۲۰٫۸۳۳	۹٫۸۲۰	۰٫۰۰۲
τ_2	۴۲۱۸۷٫۵۰۰	۱	۴۲۱۸۷٫۵۰۰	۱۱۷٫۶۶۵	۰٫۰۰۰
R_2	۲۰۸۳٫۳۳۳	۱	۲۰۸۳٫۳۳۳	۵٫۸۱۱	۰٫۰۱۶
$n * n_1$	۷۰۱۰٫۴۱۷	۲	۳۵۰۵٫۲۰۸	۹٫۷۷۶	۰٫۰۰۰
$n * q_i^2$	۴۰۵۳۱٫۲۵۰	۲	۲۰۲۶۵٫۶۲۵	۵۶٫۵۲۳	۰٫۰۰۰
$n * p^1$	۲۷۹۴۷٫۹۱۷	۲	۱۳۹۷۳٫۹۵۸	۳۸٫۹۷۵	۰٫۰۰۰
$n * p_i^2$	۱۱۸۸۵٫۴۱۷	۲	۵۹۴۲٫۷۰۸	۱۶٫۵۷۵	۰٫۰۰۰
$n * \tau_2$	۳۳۵۹۳٫۷۵۰	۲	۱۶۷۹۶٫۸۷۵	۴۶٫۸۴۸	۰٫۰۰۰
$n * R_2$	۱۰۷۲٫۹۱۷	۲	۵۳۶٫۴۵۸	۱٫۴۹۶	۰٫۲۲۴
$n_1 * q_i^2$	۲۱۳۳۳٫۳۳۳	۱	۲۱۳۳۳٫۳۳۳	۵۹٫۵۰۱	۰٫۰۰۰
$n_1 * p^1$	۲۰۸۳٫۳۳۳	۱	۲۰۸۳٫۳۳۳	۵٫۸۱۱	۰٫۰۱۶
$n_1 * p_i^2$	۵۲۰٫۸۳۳	۱	۵۲۰٫۸۳۳	۱٫۴۵۳	۰٫۲۲۸
$n_1 * \tau_2$	۱۷۵۲۰٫۸۳۳	۱	۱۷۵۲۰٫۸۳۳	۴۸٫۸۶۷	۰٫۰۰۰
$n_1 * R_2$	۸۳٫۳۳۳	۱	۸۳٫۳۳۳	۰٫۲۳۲	۰٫۶۳۰
$q_i^2 * p^1$	۱۵۱۸۷٫۵۰۰	۱	۱۵۱۸۷٫۵۰۰	۴۲٫۳۵۹	۰٫۰۰۰
$q_i^2 * p_i^2$	۷۵۰٫۰۰۰	۱	۷۵۰٫۰۰۰	۲٫۰۹۲	۰٫۱۴۸
$q_i^2 * \tau_2$	۴۴۰۸۳٫۳۳۳	۱	۴۴۰۸۳٫۳۳۳	۱۲۲٫۹۵۲	۰٫۰۰۰
$q_i^2 * R_2$	۲۰٫۸۳۳	۱	۲۰٫۸۳۳	۰٫۰۵۸	۰٫۸۱۰
$p^1 * p_i^2$	۱۳۳۳٫۳۳۳	۱	۱۳۳۳٫۳۳۳	۳٫۷۱۹	۰٫۰۵۴
$p^1 * \tau_2$	۱۲۰۰۰٫۰۰۰	۱	۱۲۰۰۰٫۰۰۰	۳۳٫۴۶۹	۰٫۰۰۰
$p^1 * R_2$	۶۰۲۰٫۸۳۳	۱	۶۰۲۰٫۸۳۳	۱۶٫۷۹۳	۰٫۰۰۰
$p_i^2 * \tau_2$	۵۲۰٫۸۳۳	۱	۵۲۰٫۸۳۳	۱٫۴۵۳	۰٫۲۲۸
$p_i^2 * R_2$	۱۳۳۳٫۳۳۳	۱	۱۳۳۳٫۳۳۳	۳٫۷۱۹	۰٫۰۵۴
$\tau_2 * R_2$	۷۵۰٫۰۰۰	۱	۷۵۰٫۰۰۰	۲٫۰۹۲	۰٫۱۴۸
خطا	۶۷۵۴۸۹٫۵۸۳	۱۸۸۴	۳۵۸٫۵۴۰		
مجموع	۱۷۹۸۰۰۰۰٫۰۰۰	۱۹۲۰			



شکل ۲. نمودار اثرات اصلی پارامترها بر درصد مسائل بهینه حل شده.

مدت زمان حل» به‌تفکیک اندازه مسائل در جدول ۳ نشان داده شده است. در جدول ۴ نیز نتایج حل با جزئیات بیشتری ارائه شده به‌طوری که در این جدول اندازه‌ی مسئله نیز به‌منظور جلوگیری از طولانی شدن جدول ادغام شده است.

براساس جدول ۳ و جدول ۴ مشاهده می‌شود که با افزایش ابعاد مسئله، درصد مسائل نمونه‌ی بهینه حل شده توسط DP۱ کاهش یافته و مدت زمان حل نیز افزایش می‌یابد، به‌طوری که در ابعاد ۶۰ سفارش تمامی مسائل نمونه به‌طور میانگین در ۶٫۰۸ ثانیه، در ابعاد ۱۰۰ سفارش ۹۶٫۰۹٪ از مسائل به‌طور میانگین در ۱۳۳٫۵۹ ثانیه و در ابعاد ۱۵۰ سفارش ۸۴٫۸۴٪ از مسائل به‌طور میانگین در ۳۳۳٫۶۶ ثانیه توسط DP۱ در محدودیت ۳۶۰۰ ثانیه حل شده است. همچنین با افزایش تعداد سفارشات عامل دوم درصد مسائل کم‌تری توسط DP۱ حل شده، به‌طوری که در گروه‌های با $n_1 = 2n_2$ به‌طور میانگین ۹۶٫۲۵٪ از مسائل و در گروه‌های با ویژگی $2n_1 = n_2$ به‌طور میانگین ۹۱٫۰۴٪ از مسائل نمونه حل شده است. این امر به‌دلیل زمان‌بر بودن فاز دوم الگوریتم DP۱ دور از انتظار نیست.

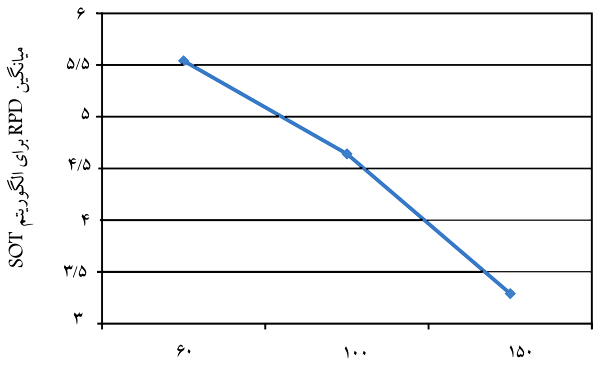
نکته‌ی مهم دیگری که باید به آن اشاره کرد عملکرد مناسب الگوریتم ابتکاری SOT است، به‌طوری که میانگین درصد انحراف نسبی (RPD)[۲۶] این الگوریتم از جواب بهینه در ۱۹۲۰ مسئله‌ی نمونه‌ی بررسی شده برابر ۴٫۸۲٪ است. قابل ذکر است مقدار RPD برای هر مسئله‌ی نمونه از رابطه‌ی ۱۱ محاسبه شده که در آن Z_{opt} و Z_{H_1} به‌ترتیب مقدار تابع هدف حاصل از الگوریتم‌های DP۱ (بهینه) و SOT است.

$$\text{RPD} = \frac{Z_{opt} - Z_{H_1}}{Z_{opt}} \times 100 \qquad (11)$$

میانگین RPD برای الگوریتم ابتکاری SOT به‌ازای ابعاد مختلف مسائل در شکل ۳ به نمایش درآمده است. چنان که مشاهده می‌شود با افزایش ابعاد مسئله، میانگین RPD الگوریتم ابتکاری کاهش یافته است. دلیل این امر آن است که با افزایش ابعاد مسئله مقدار تابع هدف بهینه نیز افزایش می‌یابد و لذا در رابطه‌ی ۱۱ مخرج کسر بزرگ می‌شود و از آنجا که صورت کسر به‌دلیل عملکرد مناسب SOT فاصله‌ی خود را از جواب بهینه حفظ می‌کند، لذا مقدار RPD کاهش می‌یابد.

در شکل ۴ عملکرد الگوریتم ابتکاری SOT به‌ازای سطوح مختلف پارامترهای مسئله قابل مشاهده است. براساس این شکل، به ترتیب کم‌ترین و بیشترین تأثیر بر میانگین RPD مربوط به پارامترهای زمان پردازش عامل اول (p^1) و درآمد عامل دوم (q_i^2) است.

از آنجاکه الگوریتم DP۱ به‌دلیل استفاده از اصول غلبه‌ی ۱ و ۲ دارای پیچیدگی شبه چندجمله‌یی است، این اصول باید کارایی قابل توجهی داشته باشند که داده‌های



شکل ۳. میانگین درصد خطای الگوریتم SOT به‌ازای ابعاد مختلف.

جدول ۳. نتایج کلی حل به‌ازای گروه‌های G۰۱ تا G۶۴ به تفکیک اندازه مسائل.

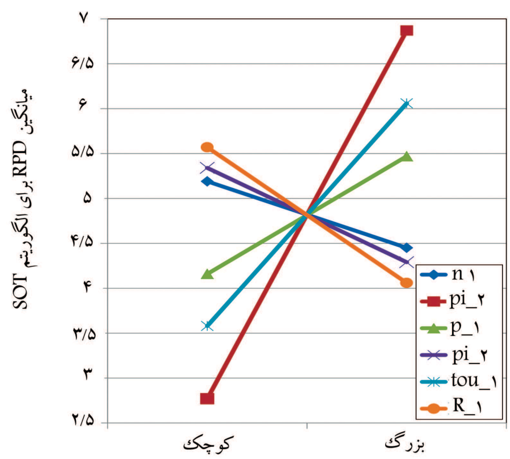
مشخصات مشترک: پارامترها	مشخصات مشترک: n	مشخصات مشترک: تعداد نمونه	گروه با مشخصات $۲n_{۱} = n_{۲}$ $q_{۲}^{i}\square[۱٫۲\times p_{i}^{۲}]$	تعداد نمونه بهینه	میانگین مدت زمان حل (ثانیه)	گروه با مشخصات $۲n_{۱} = n_{۲}$ $q_{۲}^{i}\square[۱٫۲۰\times p_{i}^{۲}]$	تعداد نمونه بهینه	میانگین مدت زمان حل (ثانیه)	گروه با مشخصات $n_{۱} = ۲n_{۲}$ $q_{۲}^{i}\square[۱٫۲\times p_{i}^{۲}]$	تعداد نمونه بهینه	میانگین مدت زمان حل (ثانیه)	گروه با مشخصات $n_{۱} = ۲n_{۲}$ $q_{۲}^{i}\square[۱٫۲۰\times p_{i}^{۲}]$	تعداد نمونه بهینه	میانگین مدت زمان حل (ثانیه)
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۴۴		۱۰	۰٫۴۳		۱۰	۰٫۰۶		۱۰	۰٫۱۴
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۰۱	۱۰	۰٫۵۷	G۱۷	۱۰	۳٫۳۴	G۳۳	۱۰	۰٫۹۲	G۴۹	۱۰	۱٫۸۷
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۳٫۰۳		۱۰	۲۱٫۶۲		۱۰	۱۱٫۴۸		۱۰	۲۱٫۲۱
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۰۲		۱۰	۰٫۰۴		۱۰	۰٫۰۵		۱۰	۰٫۹۴
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۰۲	۱۰	۰٫۰۷	G۱۸	۱۰	۰٫۱۳	G۳۴	۱۰	۳٫۹۹	G۵۰	۱۰	۱۹٫۵۵
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۰٫۳۹		۱۰	۰٫۴۷		۱۰	۵۲٫۸۶		۱۰	۲۰۴٫۵۷
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۱۴		۱۰	۱۰٫۱۷		۱۰	۰٫۰۶		۱۰	۲٫۵۷
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۰۳	۱۰	۲٫۰۸	G۱۹	۹	۷۲۵٫۱۷	G۳۵	۱۰	۲٫۶۰	G۵۱	۱۰	۱۰۵٫۸۱
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۲۵٫۸۰		۰	—		۱۰	۱۰٫۲۰		۱۰	۷۸۵٫۴۹
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۰۴		۱۰	۳۵٫۳۵		۱۰	۰٫۰۷		۱۰	۱٫۸۴
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۰۴	۱۰	۱٫۱۹	G۲۰	۹	۹۰۳٫۷۵	G۳۶	۱۰	۰٫۸۸	G۵۲	۱۰	۵۱٫۸۸
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۱۲٫۴۷		۴	۹۲۹٫۸۷		۱۰	۱۰٫۷۱		۷	۵۰۲٫۳۷
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۰۹		۱۰	۰٫۷۸		۱۰	۰٫۰۳		۱۰	۰٫۱۸
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۰۵	۱۰	۰٫۷۵	G۲۱	۱۰	۱۸٫۱۷	G۳۷	۱۰	۰٫۴۰	G۵۳	۱۰	۴٫۵۸
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۶٫۵۹		۱۰	۸۸٫۰۹		۱۰	۳٫۵۶		۱۰	۶۱٫۰۴
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۰۳		۱۰	۰٫۰۸		۱۰	۰٫۰۱		۱۰	۰٫۰۳
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۰۶	۱۰	۰٫۰۹	G۲۲	۱۰	۰٫۱۹	G۳۸	۱۰	۰٫۰۵	G۵۴	۱۰	۰٫۱۷
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۰٫۳۱		۱۰	۳٫۳۲		۱۰	۰٫۵۱		۱۰	۳٫۷۱
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۱۶		۱۰	۱٫۷۳		۱۰	۰٫۰۵		۱۰	۰٫۷۰
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۰۷	۱۰	۱٫۲۰	G۲۳	۱۰	۳۴٫۲۰	G۳۹	۱۰	۰٫۹۲	G۵۵	۱۰	۱۶٫۶۸
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۸٫۲۱		۱۰	۲۳۴٫۶۴		۱۰	۹٫۵۹		۱۰	۱۳۲٫۷۵
$p^{۱} = ۵$	۶۰	۱۰		۱۰	۰٫۰۸		۱۰	۱٫۳۳		۱۰	۰٫۰۲		۱۰	۰٫۳۱
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۰۸	۱۰	۱٫۳۳	G۲۴	۱۰	۶۵٫۵۹	G۴۰	۱۰	۰٫۲۱	G۵۶	۱۰	۱۸٫۴۰
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۹٫۹۳		۱۰	۴۵۷٫۳۶		۱۰	۶٫۳۷		۱۰	۱۳۱٫۴۳
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۰٫۲۷		۱۰	۰٫۸۱		۱۰	۰٫۳۱		۱۰	۰٫۲۴
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۰۹	۱۰	۷٫۵۴	G۲۵	۱۰	۹٫۳۶	G۴۱	۱۰	۴۱٫۲۶	G۵۷	۱۰	۱۸٫۴۷
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۵۴٫۶۴		۱۰	۸۲٫۳۹		۱۰	۱۵۵٫۳۴		۱۰	۸۳٫۲۶
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۱٫۰۷		۱۰	۱٫۸۹		۱۰	۱٫۱۰		۱۰	۲٫۲۰
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۱۰	۱۰	۷۷٫۱۶	G۲۶	۱۰	۴۰۹٫۹۰	G۴۲	۱۰	۱۲۶٫۲۶	G۵۸	۱۰	۱۰۹٫۹۱
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۹	۱۲۰۴٫۶۹		۵	۱۰٫۲۵		۶	۵۵۷٫۸۵		۱۰	۱۱۱۰٫۷۱
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۱٫۲۸		۱۰	۳٫۴۶		۱۰	۰٫۳۸		۱۰	۲٫۴۱
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۱۱	۱۰	۱۶٫۴۷	G۲۷	۱۰	۶۶۷٫۳۱	G۴۳	۱۰	۱۱٫۸۳	G۵۹	۱۰	۸۷٫۵۵
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۹	۳۰۱٫۵۰		۱	۲۹۰۹٫۵۳		۱۰	۶۳۹٫۸۰		۸	۱۰۲۵٫۴۴
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۱٫۴۳		۱۰	۵۹٫۰۰		۱۰	۰٫۵۷		۱۰	۴٫۰۹
$p_{۲}^{i} \in [۱, ۱۰]$	۱۰۰	۱۰	G۱۲	۱۰	۳۳٫۵۹	G۲۸	۲	۱۱۴۵٫۱۰	G۴۴	۱۰	۳۸٫۴۱	G۶۰	۱۰	۲۸۴٫۹۰
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۳۸۳٫۴۴		۰	—		۶	۱۱۸۶٫۷۷		۶	۱۸۹۲٫۱۷
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۰٫۱۳		۱۰	۰٫۸۸		۱۰	۰٫۱۰		۱۰	۰٫۴۳
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۱۳	۱۰	۲٫۸۶	G۲۹	۱۰	۵۰٫۴۴	G۴۵	۱۰	۵٫۷۳	G۶۱	۱۰	۲۰٫۲۰
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۱۰	۱۹٫۴۴		۱۰	۲۱۷٫۱۳		۱۰	۱۴۳٫۶۱		۵	۳۲۲٫۶۲
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۰٫۰۴		۱۰	۰٫۱۰		۱۰	۰٫۲۲		۱۰	۰٫۴۲
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۱۴	۱۰	۱٫۷۴	G۳۰	۱۰	۰٫۴۸	G۴۶	۱۰	۷٫۲۹	G۶۲	۱۰	۳۷٫۸۴
$\tau_{۲} = ۰٫۳, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۹٫۷۳		۱۰	۲٫۳۷		۸	۱۷۹٫۲۱		۱۰	۴۵۰٫۵۵
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۱٫۱۲		۱۰	۱۱۰٫۴۲		۱۰	۰٫۲۶		۱۰	۵٫۶۹
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۱۵	۱۰	۱۵٫۸۴	G۳۱	۵	۱۴۹۳٫۷۱	G۴۷	۱۰	۸٫۲۵	G۶۳	۱۰	۱۸۴٫۴۷
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۲$	۱۵۰	۱۰		۹	۵۹۲٫۳۹		۰	—		۱۰	۸۳٫۷۸		۸	۱۵۷۱٫۶۵
$p^{۱} = ۵۰$	۶۰	۱۰		۱۰	۰٫۲۴		۱۰	۱۲۷٫۲۶		۱۰	۰٫۱۱		۱۰	۳٫۳۰
$p_{۲}^{i} \in [۱, ۱۰۰]$	۱۰۰	۱۰	G۱۶	۱۰	۳٫۸۵	G۳۲	۲	۱۴۶۴٫۵۰	G۴۸	۱۰	۳٫۳۳	G۶۴	۸	۱۷۷٫۲۰
$\tau_{۲} = ۰٫۷, R_{۲} = ۰٫۶$	۱۵۰	۱۰		۱۰	۱۰۴٫۷۲		۰	—		۱۰	۵۸٫۵۰		۵	۹۱۶٫۱۰

جدول ۴. نتایج جزئی حل به‌ازای گروه‌های G۰۱ تا G۶۴.

گروه	n_i	q_2^i	p_1	p_2^i	τ_2	R_2	تعداد نمونه بهینه از ۳۰ نمونه	میانگین درصد خطای الگوریتم SOT	میانگین مدت زمان حل DP۱ (ثانیه)	فاز اول: میانگین درصد حالاتِ بررسی شده	فاز اول: حذف شده به دلیل حد بالای ۱	فاز اول: حذف شده به دلیل اصل غلبه‌ی ۱	فاز اول: منتقل شده به فاز دوم	فاز دوم: میانگین درصد حالاتِ بررسی شده	فاز دوم: حذف شده به دلیل امکان ناپذیری	فاز دوم: حذف شده به دلیل حد بالای ۲	فاز دوم: حذف شده به دلیل اصل غلبه‌ی ۲	باقی مانده
G۰۱	$n_2 = ۲n_1$	q_2^i □ [۱, ۲ × p_2^i]	۵	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۱٫۷۳	۱٫۳۴	۱٫۱۲	۳۲٫۹۵	۶۲٫۴۱	۴٫۶۴	۹۸٫۸۸	۰٫۰۰	۶۴٫۹۲	۳۱٫۶۴	۳٫۴۴
G۰۲						۰٫۶	۳۰	۰٫۷۱	۰٫۱۶	۹٫۴۸	۳۶٫۵۵	۶۰٫۸۸	۲٫۵۷	۹۰٫۵۲	۰٫۰۰	۷۹٫۸۸	۱۳٫۳۴	۶٫۷۷
G۰۳					۰٫۷	۰٫۲	۳۰	۱٫۷۹	۹٫۳۴	۰٫۵۹	۲۳٫۶۷	۶۸٫۸۸	۷٫۴۵	۹۹٫۴۱	۰٫۰۰	۶۵٫۵۵	۲۹٫۰۵	۵٫۴۱
G۰۴						۰٫۶	۳۰	۱٫۳	۴٫۵۷	۳٫۰۴	۲۶٫۸۲	۶۷٫۳۲	۵٫۸۷	۹۶٫۹۶	۰٫۱۰	۷۸٫۰۵	۱۶٫۸۲	۵٫۰۲
G۰۵				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۱٫۲۵	۲٫۴۸	۰٫۱۱	۷۰٫۲۹	۲۱٫۰۷	۸٫۶۴	۹۹٫۸۹	۰٫۰۶	۵۸٫۶۷	۳۷٫۳۰	۳٫۹۷
G۰۶						۰٫۶	۳۰	۰٫۲۹	۰٫۱۴	۰٫۲۷	۸۴٫۳۷	۹٫۱۹	۶٫۴۴	۹۹٫۷۳	۰٫۰۰	۸۲٫۴۷	۱۳٫۴۰	۴٫۱۳
G۰۷					۰٫۷	۰٫۲	۳۰	۱٫۷۱	۳٫۱۹	۰٫۲۷	۴۷٫۹۲	۴۲٫۰۰	۱۰٫۰۷	۹۹٫۷۳	۰٫۵۸	۴۲٫۲۴	۵۱٫۹۰	۵٫۲۸
G۰۸						۰٫۶	۳۰	۱٫۴۰	۳٫۷۸	۰٫۱۶	۵۶٫۷۴	۳۳٫۰۶	۱۰٫۲۰	۹۹٫۸۴	۰٫۱۷	۵۱٫۱۲	۴۴٫۳۳	۴٫۳۸
G۰۹			۵۰	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۴٫۹۸	۲۰٫۸۱	۰٫۷۳	۱۴٫۲۸	۸۰٫۶۰	۵٫۱۲	۹۹٫۲۷	۰٫۰۰	۲۵٫۶۵	۷۰٫۳۲	۴٫۰۳
G۱۰						۰٫۶	۲۹	۳٫۸۸	۴۰۰٫۸۵	۴٫۲۷	۱۶٫۲۷	۷۹٫۶۵	۴٫۰۸	۹۵٫۷۳	۰٫۰۰	۳۲٫۶۱	۶۱٫۱۳	۶٫۲۶
G۱۱					۰٫۷	۰٫۲	۲۹	۵٫۰۸	۹۹٫۶۹	۰٫۳۱	۱۲٫۰۶	۸۱٫۳۹	۶٫۵۵	۹۹٫۶۹	۰٫۰۰	۳۹٫۴۶	۵۳٫۳۵	۷٫۱۹
G۱۲						۰٫۶	۳۰	۴٫۵۷	۱۳۹٫۴۸	۰٫۲۲	۱۲٫۰۰	۸۲٫۲۹	۵٫۷۱	۹۹٫۷۸	۰٫۰۰	۵۱٫۳۸	۴۲٫۱۷	۶٫۴۵
G۱۳				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۱٫۶۳	۷٫۴۸	۰٫۴۵	۳۳٫۱۶	۶۲٫۹۱	۳٫۹۳	۹۹٫۵۵	۰٫۰۰	۶۲٫۸۵	۳۳٫۱۲	۴٫۰۲
G۱۴						۰٫۶	۳۰	۰٫۸۹	۳٫۸۴	۱٫۶۳	۳۶٫۹۴	۵۹٫۶۶	۳٫۴۰	۹۸٫۳۷	۰٫۰۰	۷۷٫۷۰	۱۸٫۰۸	۴٫۲۲
G۱۵					۰٫۷	۰٫۲	۲۹	۲٫۴۲	۱۸۹٫۷۰	۰٫۳۲	۲۳٫۸۲	۶۷٫۸۷	۸٫۳۲	۹۹٫۶۸	۰٫۰۰	۶۵٫۹۶	۲۷٫۷۵	۶٫۲۹
G۱۶						۰٫۶	۳۰	۱٫۳۳	۳۶٫۲۷	۲٫۰۱	۲۷٫۱۵	۶۶٫۰۶	۶٫۷۹	۹۷٫۹۹	۰٫۱۷	۷۰٫۰۵	۲۳٫۵۵	۶٫۲۳
G۱۷		q_2^i □ [۱, ۲۰ × p_2^i]	۵	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۸٫۴۰	۸٫۴۷	۰٫۲۵	۱۱٫۳۵	۷۸٫۸۸	۹٫۷۷	۹۹٫۷۵	۰٫۰۰	۶۴٫۳۴	۳۲٫۱۳	۳٫۵۳
G۱۸						۰٫۶	۳۰	۱٫۲۰	۰٫۲۱	۱۱٫۶۲	۳۴٫۲۱	۶۱٫۲۴	۴٫۵۵	۸۸٫۳۸	۰٫۰۰	۷۹٫۱۶	۷٫۴۹	۱۳٫۳۵
G۱۹					۰٫۷	۰٫۲	۱۹	۱۵٫۶۷	۳۴۸٫۸۵	۰٫۰۴	۲٫۴۳	۸۳٫۵۹	۱۳٫۹۸	۹۹٫۹۶	۰٫۲۲	۴۰٫۰۰	۵۱٫۵۳	۸٫۲۴
G۲۰						۰٫۶	۲۳	۱۱٫۷۹	۵۳۰٫۷۳	۰٫۰۲	۴٫۵۱	۸۳٫۸۰	۱۱٫۶۹	۹۹٫۹۸	۰٫۰۴	۶۸٫۶۸	۲۵٫۷۳	۵٫۵۵
G۲۱				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۷٫۴۸	۳۵٫۶۸	۰٫۱۱	۲۳٫۴۰	۶۶٫۰۲	۱۰٫۵۸	۹۹٫۸۹	۰٫۱۳	۶۷٫۸۷	۲۷٫۹۴	۴٫۰۵
G۲۲						۰٫۶	۳۰	۱٫۴۹	۱٫۲۰	۶٫۷۸	۶۴٫۲۴	۲۷٫۹۵	۷٫۸۱	۹۳٫۲۲	۰٫۰۰	۸۱٫۶۱	۱۰٫۷۳	۷٫۶۶
G۲۳					۰٫۷	۰٫۲	۳۰	۱۱٫۱۷	۹۰٫۱۹	۰٫۱۱	۶٫۲۲	۸۳٫۶۲	۱۰٫۱۵	۹۹٫۸۹	۲٫۸۴	۵۰٫۱۲	۴۱٫۹۷	۵٫۰۷
G۲۴						۰٫۶	۳۰	۱۰٫۲۷	۱۷۴٫۷۶	۰٫۱۰	۸٫۹۲	۸۱٫۳۴	۹٫۷۴	۹۹٫۹۰	۰٫۶۱	۶۲٫۷۱	۳۲٫۵۰	۴٫۱۸
G۲۵			۵۰	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۵٫۴۸	۳۰٫۸۵	۳٫۹۴	۱۲٫۹۸	۸۱٫۰۶	۵٫۹۶	۹۶٫۰۶	۰٫۰۰	۵۲٫۳۶	۴۰٫۳۱	۷٫۳۳
G۲۶						۰٫۶	۲۵	۴٫۲۰	۱۶۶٫۷۷	۱٫۲۳	۱۳٫۹۱	۷۹٫۷۹	۶٫۳۰	۹۸٫۷۷	۰٫۰۰	۶۲٫۶۵	۳۲٫۶۸	۴٫۶۷
G۲۷					۰٫۷	۰٫۲	۲۱	۷٫۲۵	۴۵۷٫۹۷	۰٫۱۱	۶٫۴۰	۸۳٫۱۷	۱۰٫۴۲	۹۹٫۸۹	۰٫۰۰	۴۲٫۶۲	۴۹٫۸۷	۷٫۵۱
G۲۸						۰٫۶	۱۲	۸٫۰۰	۲۴۰٫۰۲	۰٫۰۴	۶٫۳۵	۸۰٫۵۳	۱۳٫۱۲	۹۹٫۹۶	۰٫۰۰	۵۲٫۶۶	۳۷٫۲۲	۱۰٫۱۱
G۲۹				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۷٫۸۳	۸۹٫۴۸	۰٫۱۲	۱۲٫۲۹	۷۷٫۹۴	۹٫۷۷	۹۹٫۸۸	۰٫۰۰	۶۶٫۶۸	۲۹٫۲۱	۴٫۱۱
G۳۰						۰٫۶	۳۰	۱٫۴۱	۰٫۹۸	۴٫۳۸	۳۳٫۵۵	۶۲٫۷۴	۳٫۷۰	۹۵٫۶۲	۰٫۰۰	۸۳٫۳۶	۹٫۲۱	۷٫۴۴
G۳۱					۰٫۷	۰٫۲	۱۵	۱۶٫۸۹	۵۷۱٫۵۱	۰٫۰۲	۲٫۹۶	۸۲٫۲۵	۱۴٫۷۹	۹۹٫۹۸	۰٫۲۴	۴۰٫۸۵	۴۸٫۴۶	۱۰٫۴۵
G۳۲						۰٫۶	۱۲	۱۲٫۶۷	۳۵۰٫۱۳	۰٫۰۲	۳٫۵۱	۷۹٫۵۹	۱۶٫۹۰	۹۹٫۹۸	۰٫۰۳	۶۳٫۷۸	۲۸٫۰۹	۸٫۱۱
G۳۳	$n_1 = ۲n_2$	q_2^i □ [۱, ۲ × p_2^i]	۵	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۳٫۰۷	۴٫۱۵	۶٫۹۴	۹٫۶۵	۸۸٫۵۶	۱٫۷۹	۹۳٫۰۶	۰٫۰۰	۳۲٫۳۵	۵۹٫۹۸	۷٫۶۷
G۳۴						۰٫۶	۳۰	۲٫۲۲	۱۸٫۹۶	۹٫۱۳	۱۰٫۲۶	۸۷٫۹۹	۱٫۷۵	۹۰٫۸۷	۰٫۰۰	۴۰٫۰۶	۵۲٫۰۸	۷٫۸۶
G۳۵					۰٫۷	۰٫۲	۳۰	۲٫۷۹	۴٫۲۹	۹٫۹۲	۸٫۰۵	۸۹٫۶۰	۲٫۳۴	۹۰٫۰۸	۰٫۰۰	۶۶٫۳۰	۲۰٫۵۷	۱۳٫۱۳
G۳۶						۰٫۶	۳۰	۳٫۱۲	۳٫۸۹	۷٫۴۰	۷٫۲۸	۹۰٫۱۱	۲٫۶۱	۹۲٫۶۰	۰٫۲۳	۵۸٫۲۱	۲۸٫۷۱	۱۲٫۸۵
G۳۷				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۰٫۸۱	۱٫۳۳	۷٫۶۱	۳۲٫۱۱	۶۵٫۰۹	۲٫۸۰	۹۲٫۳۹	۰٫۰۰	۵۶٫۴۸	۳۴٫۸۹	۸٫۶۲
G۳۸						۰٫۶	۳۰	۰٫۲۳	۰٫۱۹	۱۶٫۶۱	۳۶٫۱۵	۶۲٫۵۰	۱٫۳۶	۸۳٫۳۹	۰٫۰۰	۷۸٫۹۴	۱۲٫۵۹	۸٫۴۷
G۳۹					۰٫۷	۰٫۲	۳۰	۰٫۸۷	۳٫۵۲	۹٫۹۳	۲۵٫۶۴	۷۰٫۵۵	۳٫۸۱	۹۰٫۰۷	۰٫۰۴	۵۳٫۲۱	۳۵٫۱۱	۱۱٫۶۵
G۴۰						۰٫۶	۳۰	۰٫۷۵	۲٫۲۰	۱۶٫۰۷	۲۶٫۸۷	۶۹٫۳۰	۳٫۸۳	۸۳٫۹۳	۰٫۶۴	۶۲٫۵۱	۲۴٫۹۷	۱۱٫۸۸
G۴۱			۵۰	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۷٫۰۵	۶۵٫۶۳	۵٫۴۸	۶٫۸۱	۹۰٫۸۰	۲٫۳۹	۹۴٫۵۲	۰٫۰۰	۳٫۳۶	۸۷٫۹۳	۸٫۷۱
G۴۲						۰٫۶	۲۶	۶٫۴۴	۱۷۷٫۷۲	۵٫۰۴	۷٫۸۰	۸۹٫۹۳	۲٫۲۷	۹۴٫۹۶	۰٫۰۰	۴٫۱۵	۸۶٫۹۶	۸٫۸۸
G۴۳					۰٫۷	۰٫۲	۳۰	۶٫۱۵	۲۱۷٫۳۴	۳٫۲۱	۷٫۱۰	۹۰٫۶۷	۲٫۲۳	۹۶٫۷۹	۰٫۰۰	۱۷٫۶۲	۶۶٫۳۰	۱۶٫۰۸
G۴۴						۰٫۶	۲۶	۶٫۰۲	۲۸۸٫۸۶	۳٫۲۲	۷٫۵۰	۸۹٫۹۳	۲٫۵۶	۹۶٫۷۸	۰٫۰۰	۲۳٫۹۷	۶۰٫۴۴	۱۵٫۵۹
G۴۵				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۳٫۱۳	۴۹٫۸۱	۵٫۱۳	۱۰٫۴۳	۸۷٫۵۶	۲٫۰۱	۹۴٫۸۷	۰٫۰۰	۲۹٫۴۵	۶۰٫۹۲	۹٫۶۳
G۴۶						۰٫۶	۲۸	۳٫۲۷	۵۳٫۸۸	۵٫۴۸	۱۰٫۸۷	۸۷٫۱۳	۱٫۹۹	۹۴٫۵۲	۰٫۰۰	۳۲٫۶۴	۵۶٫۹۸	۱۰٫۳۸
G۴۷					۰٫۷	۰٫۲	۳۰	۳٫۸۱	۳۰٫۷۶	۴٫۵۳	۸٫۰۷	۸۸٫۹۵	۲٫۹۸	۹۵٫۴۷	۰٫۰۰	۶۱٫۷۱	۲۴٫۰۹	۱۴٫۱۹
G۴۸						۰٫۶	۳۰	۳٫۹۴	۲۰٫۶۴	۶٫۵۴	۸٫۰۸	۸۹٫۱۸	۲٫۷۴	۹۳٫۴۶	۰٫۱۳	۵۳٫۰۱	۳۱٫۷۸	۱۵٫۰۸

ادامه‌ی جدول ۴. نتایج جزئی حل به‌ازای گروه‌های G۰۱ تا G۶۴.

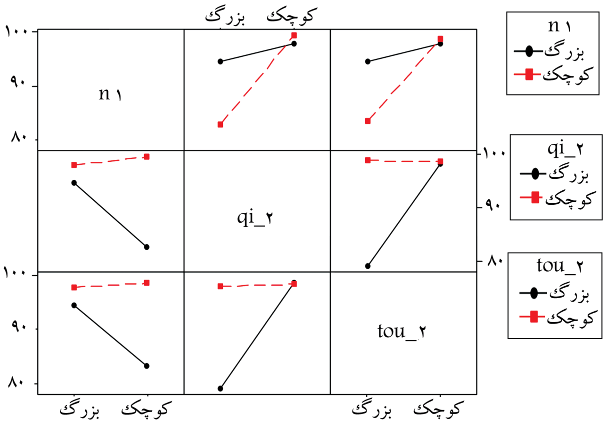
گروه	n_i	$q_i^۲$	$p_۱$	$p_۲^i$	$\tau_۲$	$R_۲$	تعداد نمونه بهینه از ۳۰ نمونه	میانگین درصد خطای الگوریتم SOT	میانگین مدت زمان حل DP۱ (ثانیه)	فاز اول: میانگین درصد حالاتِ بررسی شده	فاز اول: حذف شده به دلیل حد بالای ۱	فاز اول: حذف شده به دلیل اصل غلبه‌ی ۱	فاز اول: منتقل شده به فاز دوم	فاز دوم: میانگین درصد حالاتِ بررسی شده	فاز دوم: حذف شده به دلیل امکان ناپذیری	فاز دوم: حذف شده به دلیل حد بالای ۲	فاز دوم: حذف شده به دلیل اصل غلبه‌ی ۲	فاز دوم: باقی مانده
G۴۹	$n_۱ = ۲n_۲$	$[p_i^۲ \times ۰/۲(]\square q_i^۲$	۵	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۴٫۶۷	۷٫۷۴	۸٫۲۹	۸٫۱۴	۸۹٫۴۳	۲٫۴۳	۹۱٫۷۱	۰٫۰۰	۵۵٫۵۶	۳۵٫۸۰	۸٫۶۴
G۵۰						۰٫۶	۳۰	۳٫۷۰	۷۵٫۰۲	۷٫۶۶	۷٫۴۷	۸۹٫۶۹	۲٫۸۴	۹۲٫۳۴	۰٫۰۰	۵۹٫۰۶	۳۲٫۳۰	۸٫۶۴
G۵۱					۰٫۷	۰٫۲	۲۷	۸٫۰۵	۲۴۳٫۷۹	۰٫۹۷	۲٫۷۲	۹۲٫۸۵	۴٫۴۳	۹۹٫۰۳	۰٫۰۰	۴۲٫۳۰	۴۲٫۲۱	۱۵٫۴۹
G۵۲						۰٫۶	۳۰	۶٫۲۴	۱۸۵٫۳۶	۰٫۹۸	۲٫۵۸	۹۳٫۲۴	۴٫۱۹	۹۹٫۰۲	۰٫۰۷	۶۴٫۴۴	۲۲٫۸۶	۱۲٫۶۳
G۵۳				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۵٫۴۹	۲۱٫۹۳	۱٫۹۷	۱۴٫۵۱	۸۰٫۶۱	۴٫۸۸	۹۸٫۰۳	۰٫۰۱	۶۴٫۱۲	۲۷٫۲۹	۸٫۵۸
G۵۴						۰٫۶	۳۰	۱٫۵۱	۱٫۳۰	۱۴٫۵۲	۲۸٫۲۲	۶۹٫۱۸	۲٫۶۰	۸۵٫۴۸	۰٫۰۰	۸۰٫۸۷	۱۰٫۴۰	۸٫۷۳
G۵۵					۰٫۷	۰٫۲	۳۰	۷٫۴۳	۵۰٫۰۴	۱٫۸۸	۵٫۹۹	۸۹٫۴۳	۴٫۵۸	۹۸٫۱۲	۱٫۴۰	۵۸٫۱۶	۲۸٫۹۶	۱۱٫۴۹
G۵۶						۰٫۶	۳۰	۴٫۶۵	۵۰٫۰۴	۴٫۲۰	۸٫۸۵	۸۶٫۴۲	۴٫۷۳	۹۵٫۸۰	۱٫۲۳	۷۱٫۴۳	۱۶٫۲۵	۱۱٫۰۹
G۵۷			۵۰	[۱, ۱۰]	۰٫۳	۰٫۲	۳۰	۵٫۹۷	۳۳٫۹۹	۷٫۳۷	۷٫۴۲	۹۰٫۵۶	۲٫۰۲	۹۲٫۶۳	۰٫۰۰	۲۳٫۰۷	۶۸٫۰۹	۸٫۸۴
G۵۸						۰٫۶	۲۸	۵٫۸۷	۳۵۷٫۳۹	۷٫۷۵	۷٫۴۳	۹۰٫۴۳	۲٫۱۴	۹۲٫۲۵	۰٫۰۰	۲۵٫۳۲	۶۵٫۵۸	۹٫۱۰
G۵۹					۰٫۷	۰٫۲	۲۶	۶٫۵۲	۲۷۱٫۲۴	۱٫۰۱	۵٫۸۷	۹۰٫۹۹	۳٫۱۴	۹۸٫۹۹	۰٫۰۰	۲۲٫۸۰	۶۰٫۲۳	۱۶٫۹۷
G۶۰						۰٫۶	۲۵	۷٫۱۱	۴۹۴٫۰۳	۰٫۷۰	۵٫۷۳	۹۰٫۸۰	۳٫۴۶	۹۹٫۳۰	۰٫۰۰	۳۰٫۳۷	۵۳٫۸۲	۱۵٫۸۱
G۶۱				[۱, ۱۰۰]	۰٫۳	۰٫۲	۳۰	۴٫۹۹	۱۱۴٫۴۲	۴٫۳۱	۸٫۱۳	۸۸٫۹۳	۲٫۹۴	۹۵٫۶۹	۰٫۰۰	۵۳٫۷۷	۳۶٫۲۵	۹٫۹۸
G۶۲						۰٫۶	۲۸	۳٫۳۰	۱۴۲٫۳۹	۵٫۴۹	۸٫۷۹	۸۸٫۳۷	۲٫۸۴	۹۴٫۵۱	۰٫۰۰	۶۵٫۶۳	۲۴٫۶۰	۹٫۷۸
G۶۳					۰٫۷	۰٫۲	۲۳	۶٫۸۱	۴۰۸٫۳۰	۰٫۹۶	۲٫۹۸	۹۱٫۷۶	۵٫۲۶	۹۹٫۰۴	۰٫۰۰	۴۶٫۹۷	۳۵٫۶۱	۱۷٫۴۳
G۶۴						۰٫۶	۲۷	۶٫۲۷	۳۰۴٫۳۶	۱٫۲۸	۳٫۴۱	۹۱٫۹۴	۴٫۶۵	۹۸٫۷۲	۰٫۰۴	۶۶٫۸۷	۱۹٫۵۴	۱۳٫۵۵
میانگین								۴٫۸۲	۱۲۰٫۷	۳٫۸۴	۱۸٫۰	۷۶٫۴	۵٫۵۶	۹۶٫۱۶	۰٫۱۴	۵۳٫۶	۳۷٫۴	۸٫۷۸



شکل ۴. میانگین درصد خطای الگوریتم SOT به‌ازای پارامترهای مختلف تولید مسئله.

مندرج در جدول ۴ مؤید این مسئله است. براساس این جدول حدود بالای ارائه شده نیز عملکرد مناسبی از خود نشان داده‌اند. لذا در فاز اول و دوم به‌ترتیب به‌طور میانگین ۹۴٫۴۵٪ (۱۸٫۰۲٪ + ۷۶٫۴۳٪) و ۹۱٫۰۸٪ (۵۳٫۶۴٪ + ۳۷٫۴۴٪) از حالت‌های بررسی شده توسط اصول غلبه و حدود بالای مربوطه حذف شده است.

همچنین در شکل ۵، اثرات متقابل سه پارامتر $n_۱$، $q_i^۲$ و $\tau_۲$ به‌دلیل نمود بیشتر در بین سایر اثرات متقابل، نشان داده شده است. بر این اساس هنگامی که تعداد سفارش عامل اول در سطح Low بوده و درآمد عامل دوم در سطح High است مسائل سخت‌تر بوده و درصد کم‌تری از آنها در ۳۶۰۰ ثانیه به‌طور بهینه حل شده‌اند. همین رفتار هنگامی که تعداد سفارش عامل اول و $\tau_۲$ به‌ترتیب در سطح Low و High هستند، تکرار شده است. همچنین با قرارگیری درآمد عامل دوم و $\tau_۲$ هر دو در سطح High، نیز درصد مسائل کم‌تری به‌طور بهینه حل شده‌است. زیرا در این دو مورد، شرایط برقراری اصل غلبه‌ی ۲ کم‌تر شده و از این رو تعداد کم‌تری از حالت‌های فضای حالت شماره‌ی ۲ حذف می‌شوند.



شکل ۵. اثرات متقابل n_1، q_i^2 و τ_2 بر درصد مسائل بهینه‌ی حل شده.

۵. نتیجه‌گیری

در این مقاله مسئله‌ی دوعاملی پذیرش و زمان‌بندی سفارشات با هدف بیشینه‌سازی سود سفارشات پذیرفته شده مطرح شد. این مسئله با فرض برابر بودن زمان پردازش سفارشات عامل اول و مجاز نبودن دیرکرد سفارشات عامل دوم مورد بررسی قرار گرفت. نشان داده شد که این مسئله NP-hard بوده و با فرض معلوم بودن سفارشات پذیرفته شده، برای تعیین توالی بهینه یک الگوریتم چندجمله‌یی ارائه شد. برای حل مسئله‌ی اصلی نیز یک الگوریتم ابتکاری کارآمد و یک برنامه‌ریزی پویای شبه‌چندجمله‌یی ارائه شد.

به‌منظور بررسی عملکرد این الگوریتم‌ها تعدادی مسئله‌ی نمونه تولید و حل شده است. برای تحلیل اثر پارامترهای مختلف حاضر در تولید مسائل نمونه نیز از روش تحلیل واریانس بهره گرفته شده است. نتایج حل حاکی از توانایی الگوریتم DP۱ در حل بهینه‌ی ۹۳٫۶۵٪ از مسائل نمونه در محدوده‌ی ۳۶۰۰ ثانیه است. همچنین الگوریتم ابتکاری نیز عملکرد قابل قبولی از خود به نمایش گذاشته به‌طوری که میانگین درصد انحراف نسبی آن از جواب بهینه در کل مسائل بهینه حل شده برابر ۴٫۸۲٪ است.

برای مطالعات آتی، به‌دلیل NP-hard بودن مسئله‌ی بررسی شده در این نوشتار، می‌توان از الگوریتم‌های فراابتکاری برای حل آن استفاده کرد و از برنامه‌ریزی پویای ارائه شده در این نوشتار برای بررسی کیفیت جواب آنها استفاده کرد. همچنین از آنجا که مسائل پذیرش و زمان‌بندی سفارشات و زمان‌بندی چندعاملی به‌طور جداگانه اهمیت فراوانی در ادبیات موضوع دارند و در این نوشتار نیز بیان شد که ترکیب آنها، یک مسئله‌ی کاربردی‌تر را پدید می‌آورد. لذا حل این مسئله تحت فرضیات متنوع و بیشتری قابل بررسی است. برای مثال می‌توان تعداد عامل‌ها یا در حقیقت انواع مشتریان را بیش از دو عامل در نظر گرفت. بررسی مسئله براساس توابع جریمه‌ی دیگری نظیر دیرکرد وزنی نیز می‌تواند جالب توجه باشد.

پانوشت‌ها

1. order acceptance and scheduling
2. multi-agent scheduling
3. scheduling with competing agents
4. beam search
5. greedy
6. fully polynomial time approximation scheme
7. mixed integer linear programming
8. jobshop
9. branch and price
10. two machine flowshop
11. regular function
12. deteriorating jobs
13. makespan
14. hybrid kangaroo simulated annealing
15. batch
16. rounding procedure
17. sequencing accepted orders
18. earliest due date
19. selecting one order at a time
20. largest due date
21. full factorial experimental design
22. analysis of variance
23. treatment
24. confidence level
25. interaction effects
26. relative percentage deviation

منابع (References)

1. Slotnick, S.A. "Order acceptance and scheduling: A taxonomy and review", *European Journal of Operational Research*, **212**, pp. 1-11 (2011).
2. Slotnick, S.A. and Morton, T.E. "Selecting jobs for a heavily loaded shop with lateness penalties", *Computers and Operations Research*, **23**, pp. 131-140 (1996).
3. Ghosh, J.B. "Job selection in a heavily loaded shop", *Computers and Operations Research*, **24**, pp. 141-145 (1997).
4. Slotnick, S.A. and Morton, T.E. "Order acceptance with weighted tardiness", *Computers and Operations Research*, **34**, pp. 3029-3042 (2007).
5. Rom, W.O., and Slotnick, S.A. "Order acceptance using genetic algorithms", *Computers and Operations Research*, **36**, pp. 1758-1767 (2009).
6. Nobibon, F.T. and Leus, R. "Exact algorithms for a generalization of the order acceptance and scheduling prob-

lem in a single-machine environment", *Computers and Operations Research*, **38**, pp. 367-378 (2011).

7. Yang, B. and Geunes, J. "A single resource scheduling problem with job-selection flexibility, tardiness costs and controllable processing times", *Computers and Industrial Engineering,* **53**, pp. 420-432 (2007).

8. Mestry, S., Damodaran, P. and Chen, C.S. "A branch and price solution approach for order acceptance and capacity planning in make-to-order operations", *European Journal of Operational Research*, **211**, pp. 480-495 (2011).

9. Shabtay, D. and Gasper, N. "Two-machine flow-shop scheduling with rejection", *Computers and Operations Research,* **39**, pp. 1087-1096 (2012).

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

11. Agnetis, A., Mirchandani, P.B., Pacciarelli, D. and Pacifici, A. "Scheduling problems with two competing agents", *Operations Research*, **52**, pp. 229-242 (2004).

12. Cheng, T.C.E., Ng, C.T. and Yuan, J.J. "Multi-agent scheduling on a single machine to minimize total weighted number of tardy jobs", *Theoretical Computer Science*, **362**, pp. 273-281 (2006).

13. Liu, P. and Tang, L. "Two-agent scheduling with linear deteriorating jobs on a single machine", *Computing and Combinatorics, Lecture Notes in Computer Science,* **5092**, X. Hu, and J. Wang (eds.), Springer Berlin Heidelberg, pp. 642-650 (2008).

14. Agnetis, A., Pascale, G.D. and Pacciarelli, D. "A lagrangian approach to single-machine scheduling problems with two competing agents", *Journal of Scheduling*, **12**, pp. 401-415 (2009).

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

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

17. Lee, W.C., Chen, S.K., Chen, C.W. and Wu, C.C. "A two-machine flowshop problem with two agents", *Computers and Operations Research,* **38**, pp. 98-104 (2011).

18. Mor, B. and Mosheiov, G. "Scheduling problems with two competing agents to minimize minmax and minsum earliness measures", *European Journal of Operational Research,* **206**, pp. 540-546 (2010).

19. Mor, B. and Mosheiov, G. "Single machine batch scheduling with two competing agents to minimize total flowtime", *European Journal of Operational Research,* **215**, pp. 524-531 (2011).