



ارائه شده توسط:

سایت ترجمه فا

مرجع جدیدترین مقالات ترجمه شده

از نشریات معتبر

موازی سازی یک الگوریتم بهینه سازی غیر خطی چند هدفه: کاربردی برای یک

مسئله مکانی

چکیده

مسائل زندگی واقعی معمولاً شامل اهداف ضد و نقیض می شوند. حل مسائل چند هدفه (به عنوان مثال، به دست آوردن مجموعه ای کارآمد و کامل مربوط به پارتو-جلو) از طریق روش های دقیق در بسیاری از موارد در برابر چاره جویی مقاومت می کند. به منظور مقابله با این مسائل، چند (فرا) روش اکتشافی در طول دهه گذشته توسعه یافته اند که هدف آنها به دست آوردن یک تقریب گسسته خوب از پارتو-جلو است. در این راستا، یک الگوریتم چند هدفه تکاملی جدید، به نام FEMOEA، که می توان برای بسیاری از مسائل بهینه سازی چند هدفه غیر خطی استفاده نمود، به تازگی پیشنهاد شده است. از طریق مقایسه با یک الگوریتم دقیق فاصله شاخه و حد، نشان داده شده است که FEMOEA تقریب بسیار خوبی از پارتو-جلو فراهم می کند. علاوه بر این، با الگوریتم های مرجع NSGA-II، SPEA2 و MOEA / D مقایسه شده است. مطالعات محاسباتی جامع نشان داده است که، در میان الگوریتم های مورد مطالعه، FEMOEA، به طور متوسط، بهترین نتایج را برای همه شاخص های کیفیت تجزیه و تحلیل ارائه می دهد. با این حال، هنگامی که مجموعه تخمین پارتو-جلو باید بسیاری از نقاط را داشته باشند (چون یک دقت بالا مورد نیاز است)، زمان محاسبات مورد نیاز توسط FEMOEA می تواند هرگز ناچیز نباشند. علاوه بر این، الزامات حافظه مورد نیاز به وسیله الگوریتم هنگام حل آن موارد می تواند آنقدر زیاد باشد که حافظه موجود به اندازه کافی نباشد. در آن موارد، موازی سازی الگوریتم و اجرای آن در یک معماری موازی می تواند بهترین روش مستقیم باشد. در این کار، یک موازی سازی FEMOEA، به نام FEMOEA-Paral، ارائه شده است. برای نشان دادن کاربرد آن، یک محل تسهیلات رقابتی دو هدفه و طراحی مشکل حل شده است. نتایج نشان می دهد که FEMOEA-Paral قادر به حفظ اثربخشی نسخه های ترتیبی است و این مورد با کاهش هزینه محاسباتی همراه می شود. علاوه بر این، نسخه موازی،

مقیاس پذیری خوبی را نشان می دهد. نتایج بهره وری با استفاده از یک ابزار پروفایل بندی و ردیابی برای تجزیه و تحلیل عملکرد تحلیل شده است.

کلمات کلیدی: بهینه سازی چندهدفه غیرخطی، الگوریتم تکاملی چندهدفه، موازی سازی، بازده، اثربخشی، سیستم عملکرد کاربردهای تنظیم و تحلیل، مسئله موقعیت تسهیلات

۱. مقدمه

مسائل بهینه سازی چندهدفه، همه جا وجود دارند. بسیاری از مسائل زندگی واقعی به اتخاذ چندین دیدگاه متناقض نیاز دارند [2,3,11,12,20,29]. بنابراین، الگوریتم های بهینه سازی چندهدفه کلی قادر به حل مسائل بهینه سازی دارای مشکل در حل کردن هستند.

در این مقاله، ما با مسئله بهینه سازی چندهدفه غیرخطی کلی (MOP) سرو کار داریم که می توانند به صورت زیر فرموله شوند:

$$\begin{aligned} \min \quad & \{f_1(\mathbf{y}), \dots, f_m(\mathbf{y})\}, \\ \text{s.t.} \quad & \mathbf{y} \in S \subseteq \mathbb{R}^n, \end{aligned} \quad (1)$$

که در آن $f_1, \dots, f_m: \mathbb{R}^n \rightarrow \mathbb{R}$ توابع مقدار-حقیقی هستند. اجازه دهید، بردار توابع هدف را توسط $f(\mathbf{y}) = (f_1(\mathbf{y}), \dots, f_m(\mathbf{y}))$ و تصویر ناحیه عملی را توسط $Z=f(S)$ نشان دهیم.

در هنگام برخورد با مسائل چند هدفه ما نیاز به روشن نمودن این مورد داریم که حل یک مسئله چه معنی می دهد. در ادامه، برخی از تعاریف شناخته شده برای توضیح مفهوم راه حل (۱) ارائه شده است.

تعریف ۱: در صورتی یک بردار عملی $\mathbf{y}^* \in S$ کارآمد نامیده می شود، اگر و فقط اگر بردار عملی دیگر $\mathbf{y} \in S$ وجود نداشته باشد به طوری که $f_l(\mathbf{y}) \leq f_l(\mathbf{y}^*)$ برای تمام $l = 1, \dots, m$ و $f_j(\mathbf{y}) < f_j(\mathbf{y}^*)$ برای حداقل یک شاخص $j (j \in \{1, \dots, m\})$. مجموعه SE از تمام نقاط کارآمد، مجموعه کارآمد یا مجموعه پارتو نامیده می شود.

اگر y_1 و y_2 دو نقطه عملی باشند و $f_l(y_1) \leq f_l(y_2)$ برای تمام $l = 1, \dots, m$ با حداقل یکی از نامعادلات محدود، آنگاه می‌گوییم که y_1 بر y_2 مسلط است.

بازده در فضای تصمیم‌گیری تعریف می‌شود. تعریف متناظر در فضای معیار به شرح زیر است:

تعریف ۲. بردار $z^* = f(y^*) \in Z$ ، غیرمسلط نامیده می‌شود اگر و فقط اگر y^* کارآمد باشد. مجموعه Z_N از تمام حامل‌های غیر تحت سلطه، مجموعه غیر تحت سلطه یا پارتو-جلو نامیده می‌شود. اگر y_1 و y_2 دو نقطه امکان‌پذیر باشند و Y_1 بر Y_2 مسلط باشند، آنگاه می‌گویند $f(y_1)$ بر $f(y_2)$ تسلط دارد.

در حالت ایده‌آل، حل (۱) به معنی به دست آوردن مجموعه کلی کارآمد است، یعنی، تمام نقاطی که موثر و متناظر با پارتو-جلو آن هستند. با این حال، برای اکثریت MOPها، به دست آوردن توضیحات دقیق از مجموعه کارآمد و یا پارتو-جلو آسان نیست، زیرا آن مجموعه‌ها معمولاً شامل یک تعداد نامحدود از نقاط (معمولاً مجموعه زنجیره‌ای) می‌شوند. تا حدی از دانش ما، فقط سه روش کلی دقیق، یعنی، سه روش شاخه و حد (نگاه کنید به [۹،۱۰،۲۴]) در نوشته‌ها مطرح شده است که یک حوزه از روش‌های تنظیم شده تا یک دقت از پیش مشخص شده را به دست می‌آورد.

به طور خاص، آنها یک لیست از جعبه‌ها (فواصل چند بعدی) را ارائه می‌دهند که اتحاد آنها شامل مجموعه کامل کارآمد (و تصاویر آنها، پارتو-جلو متناظر) به عنوان یک راه حل می‌شود. با این حال، آنها وقت‌گیر می‌باشند. علاوه بر این، آنها نیاز به حافظه‌های بزرگ دارند، به طوری که تنها نمونه کوچک را می‌توان با آنها حل نمود. دیگر روش‌های جالب را می‌توان در [7,28] یافت.

در مقابل، استفاده از فن‌آوری‌های (فرا) هوشمند می‌تواند برای ما به دست آوردن «تقریب خوبی از پارتو-جلو، حتی برای مسائل با متغیرها و اهداف را میسر سازد. منظور ما از به واسطه یک تخمین خوب به معنی یک مجموعه گسسته از نقاط گسترده روی پارتو-جلو یا به طور مساوی توزیع شده روی آن است. است مجموعه‌ای از روش‌ها با آن هدف در نوشته‌ها وجود دارند، اگرچه بسیاری از آنها برای مقابله با MOPs ترکیبی طراحی شده‌اند (چند استثنا [5,14,17,21,25] هستند). با این حال، رایج‌ترین روش مورد استفاده در نوشته‌ها برای مقابله با (۱)، استفاده از

الگوریتم های تکاملی چند هدفه (MOEAs) است. این به خاطر توانایی آن برای پیدا کردن راه حل های متعدد کارآمد در یک شبیه سازی تک اجراست (برای یک مقدمه عالی در این مورد موضوع نگاه کنید به [۴]).

به تازگی، یک الگوریتم تکاملی جدید سریع و کارآمد چند هدفه (FEMOEA)، با هدف کسب سریع یک تقریب خوب اندازه ثابت از پارتو-جلو ارائه شده است [۲۲،۲۳]. این روش، ایده ها را از دیگر MOEAs معمولی با برخی از مفاهیم از دیگر الگوریتم های تکاملی (EAها) ابداع شده برای مقابله با مسائل بهینه سازی تک هدفه ترکیب می کند. علاوه بر این، به منظور سرعت بخشیدن به همگرایی خود را به سمت بهینه پارتو-جلو، FEMOEA شامل دو دستگاه جدید: یک روش جدید و بهبود یک قاعده توقف های جدید است. این دو سهم می تواند در هر MOEA گنجانده شده است. در [۲۲]، FEMOEA با iB & B مقایسه شد، یک الگوریتم شاخه و حد بازه توانمند در به دست آوردن یک محدوده از پارتو-جلو صحیح، زمانی که تسهیلات رقابتی دارای مشکل در حل کردن شرح داده شده در [۱۰] حل می شوند. یک مطالعه محاسباتی جامع نشان داد که FEMOEA رقابتی است و به طور متوسط قادر به کاهش زمان محاسبات از روش دقیق تا حدود ۹۹٪ است، و این راه حل های نهایی با کیفیت خوب را ارائه می دهد، برای مثال، تمام نقاط ارائه شده توسط الگوریتم (تقریباً) کارآمد بودند و آنها به طور مساوی بر پارتو-جلو کامل توزیع شدند. علاوه بر این، در [۲۳]، FEMOEA با الگوریتم های به خوبی شناخته شده NSGA-II و SPEA2 مقایسه شد [۳۳]، که به الگوریتم های مرجع در جامعه محاسبات تکاملی چند هدفه تبدیل شده اند. علاوه بر این، با MOEA / D نیز مقایسه شد، یک الگوریتم جدیدتر که ثابت شده است یکی از رقابتی ترین الگوریتم های تکاملی چند هدفه [۳۰،۳۱] است. چندین شاخص کیفیت، در نظر گرفته شدند، یعنی، هایپرولوم، فاصله متوسط، شاخص اپسیلون افزودنی، گسترش و فاصله گذاری. با توجه به نتایج محاسباتی و تجزیه و تحلیل آماری انجام شده، الگوریتم جدید، به طور متوسط، عملکرد بهتری نسبت به سه الگوریتم دیگر در تمام شاخص های کیفیت دارد.

هرچند زمانی که مجموعه تخمین زدن پارتو-جلو باید دارای چندین نقطه باشد (چون دقت بالایی نیاز دارد)، زمان محاسباتی مورد نیاز توسط FEMOEA نمی تواند هرگز قابل چشمپوشی باشد. علاوه بر این، الزامات حافظه مورد نیاز توسط الگوریتم در هنگام حل کردن آن نمونه ها می تواند آنقدر بالا باشد که حافظه در دسترس نمی تواند کافی

باشد. در آن موارد، موازی سازی الگوریتم و اجرای آن در یم معماری موازی می تواند بهترین روش مستقیم باشد. تا آنجا که به دانش نویسندگان مرتبط می شود، توسعه الگوریتم های تکاملی چندهدفه موازی یک میدان در حال انفجار است به که به اندازه کافی کاوش نشده است ([1,15,16,26] را ببینید).

بقیه مقاله به شرح زیر سازماندهی شده است. در بخش ۲، الگوریتم ترتیبی به اختصار شرح داده شده است. در بخش ۳، نسخه موازی FEMOEA، FEMOEA-Paral به تفصیل بیان شده است. برای نشان دادن قابلیت کاربرد آن، یک محل تسهیلات رقابتی دو-هدفه و مسئله طراحی حل شده است. مدل همراه و مطالعه محاسباتی مربوطه در بخش ۴. نشان داده شده است. در نهایت در بخش ۵، نتیجه گیری های اصلی خلاصه شده است.

۲. روش FEMOEA ترتیبی

این بخش به تبیین برخی از مفاهیم FEMOEA که برای توضیح موازی سازی آن مورد نیاز است اختصاص داده شده است. جزئیات در مورد رویه های نسخه ترتیبی حذف می شوند. برای کسب اطلاعات بیشتر، خواننده علاقه مند به [۲۲،۲۳] ارجاع داده می شود.

مهم ترین مفهوم در FEMOEA، مفهوم یک فرد است. یک فرد توسط یک مرکز و شعاع تعریف شده است. مرکز، یک راه حل است و شعاع یک عدد مثبت است که منطقه فرعی از فضای جستجوی تحت پوشش آن فرد را تعیین می کند. هدف اصلی از شعاع، تمرکز بر اپراتورهای جستجو در نواحی فرعی مربوطه است. برای کسب اطلاعات بیشتر در مورد مفهوم و همچنین محاسبه شعاع مربوط به هر فرد، [۱۹،۲۲،۲۳] را ببینید.

به غیر از مرکز و شعاع، یک فرد دارای دو ویژگی است که به فضای معیار مرتبط می شود: رتبه تسلط (drank) و فاصله تراکم (cdist)، و [۶] را ببینید. رتبه غیر سلطه نشان دهنده تعداد افرادیست که تسلط بر آن فرد خاص دارند. در این معنا، مقدار صفر بدان معنی است که چنین فردی تحت سلطه هر یک از افراد باقی مانده در جامعه کنونی نیست. فاصله ازدحام، یک برآورد از چگالی راه حل های پیرامون یک راه حل ویژه در یک جمعیت است.

در طول فرآیند بهینه سازی، دو فهرست از افراد توسط FEMOEA حفظ می شوند که حداکثر اندازه آن، M ، که برای هر دو لیست یکسان است، یک پارامتر ورودی معین می باشد. پارامتر M به تعداد مورد نظر از راه حل ها در فینال پارتو-جلو اشاره دارد. لیست اول، به نام `population_list`، از M افراد گوناگون با ویژگی های مختلف، به عنوان مثال شعاع های مختلف، صفوف غیر سلطه و فاصله ازدحام تشکیل شده است. فهرست دوم، به نام `external_list`، می تواند به عنوان سپرده برای نگه داشتن راه حل های غیر تحت سلطه قابل درک باشد.

تعریف ۳. یک راه حل i قابل ترجیح به راه حل i' یا $i', i \succ i'$ ، if $d_{rank}^i < d_{rank}^{i'}$ and $c_{dist}^i > c_{dist}^{i'}$ است.

رابطه قبلی به عنوان عملگر مقایسه شلوغ شناخته می شود (نگاه کنید به [۶]). برای سرعت بخشیدن به فرآیند انتخاب، هر دو لیست همیشه با توجه به چنین اپراتوری طبقه بندی می شوند.

در FEMOEA، در نظر گرفته می شود که هر فرد یک راه حل کارآمد را اشغال کند. برای این منظور، FEMOEA در طول فرآیند جستجو به سمت مناسب ترین مناطق، افراد را هدایت می کند. بنابراین، توجه کنید که یک فرد خاص، بخشی ثابت از دامنه جستجو نیست، بلکه می توانید از طریق فضا، با اقدام به جستجو حرکت کند. «مدیریت-فردی یکی از بخش های اصلی FEMOEA است. این روش شامل ایجاد و انتخاب افراد در طول فرآیند بهینه سازی کلی می شود. علاوه بر این، FEMOEA شامل یک روش بهبود می شود، که به لحاظ منطقی از مدیریت فرد جدا شده است. این به این معنی است که FEMOEA به راحتی می تواند برای حل هر گونه مشکل چند هدفه دیگر سازگار باشد و تنها روش بهبود را تطبیق می دهد. در [۲۳]، یک روش جدید برای بهبود بهره وری نقاط، جایی که در آن هیچ اطلاعات گرادیان استفاده نمی شود، پیشنهاد شد، در حالی که در [۲۲]، یک روش مبتنی بر گرادیان طراحی شده است.

ذکر این مورد مهم است که یک فرد منفرد در `population_list` می تواند فرزندان جدید ایجاد کند و یا بدون مشارکت فرزندان باقی مانده بهبود یابد. در نتیجه، یک موازی سازی ذاتی وجود دارد که متشکل از تقسیم افراد در میان تعداد پردازنده های در دسترس است. با این وجود، اگر چه هیچ رابطه ای میان افراد در `population_list` وجود ندارد، تکامل

جمعیت عمدتاً به راه حل های ذخیره شده در external_list بستگی دارد. علاوه بر این، external_list قابل تغییر (اضافه کردن، حذف و یا به روز رسانی افراد) با رویه های اولیه کاربردی برای population_list است. سپس، اثربخشی نسخه موازی می تواند کاهش یابد، اگر رابطه استقبال شده بین external_list و population_list به حساب آورده نشود.

۳. الگوریتم موازی FEMOEA-Paral

الگوی برنامه نویسی مورد استفاده برای موازی سازی FEMOEA می تواند به عنوان یک مدل درشت دانه در نظر گرفته شود که در آن هر پردازنده، FEMOEA را به طور مستقل از باقی مانده ها در بیشتر زمان ها اجرا می کند، اما با توجه به population_list کوچکتر. به طور دقیقتر، طول چنین فهرستی برابر با $M' = M/P$ خواهد شد، با این فرض که P پردازنده در دسترس خواهند بود. این لیست، local_population_list در زیر نامیده خواهد شد. بنابراین، این ایده که پردازنده های مختلف با یک لیست کوچکتر و متفاوت از افراد در چنین روشی، در زمان ادغام همه لیست های محلی کار می کنند، یک لیست جمعیت شبیه به لیست نسخه ترتیبی را می توان به دست آورد. با این وجود، اگر چه هیچ رابطه ای میان افراد در population_list وجود ندارد، و از این رو می توان آنها را در میان پردازنده بدون مشکل توزیع نمود، external_list در میان پردازنده ها برای جلوگیری از اثربخشی ضعیف تقسیم نمی شود. در مقابل، هر پردازنده دارای یک نسخه محلی از آن است که در سراسر این کار local_external_list نامیده خواهد شد.

یک مسئله مهم که باید برجسته شود این است که، بر خلاف نسخه ترتیبی، این دو لیست توسط اپراتور مقایسه شلوغ طبقه بندی نمی شود، بلکه فقط توسط اولین مقدار تابع هدف طبقه بندی می شوند. از آنجا که انتخاب، به صورت موازی انجام خواهد شد، تعمیر و نگهداری لیست های مرتب شده توسط اپراتور مقایسه شلوغ، از نظر بازده ضد-بهره وری است.

به غیر از این دو لیست، لیست دیگری، به نام auxiliary_external_list در طول فرآیند بهینه سازی حفظ می شود. چنین لیستی تنها در پردازنده با شماره شناسایی 0، P0 ذخیره می شود و افراد بیشتر ترجیح داده شده پیدا شده در طول فرآیند بهینه سازی کل را نگه می دارد.

الگوریتم ۱ طرح های ساختار الگوریتم موازی. در زیر، مراحل کلیدی مختلف توصیف می شوند.

Algorithm 1. FEMOEA-Paral

```
1: Init_individuals_lists_paral
2: while termination criteria are not satisfied
3:   Create_new_individuals_paral (e vals)
4:   Select_individuals_paral (local_population_list)
5:   Improve_individuals_paral (local_population_list)
6:   Update_local_external_list
7:   Select_individuals_paral (local_external_list)
8:   Improve_individuals_paral (local_external_list)
9: end while
10: if length (auxiliary_external_list) < M
11:   Compose_pareto
12: end if
```

• Init_individuals_lists_paral: در ابتدا، افراد بسیار به اندازه پارامتر M' نشان داده شده در هر پردازنده ایجاد می شوند.

همانند نسخه ترتیبی، مرکز افراد به صورت تصادفی محاسبه می شوند، در حالی که شعاع آنها شعاع مرتبط در سطح

۱ خواهد بود. local_population_list از این مجموعه از افراد مقداره‌ی اولیه می شود، در حالی که local_external_list

تنها شامل افراد غیر تحت سلطه می شود

پس از این روش، یک حلقه شروع می شود، که اساساً افراد را ایجاد، انتخاب و بهبود می بخشد. این حلقه تا زمانی

اجرا می شود که بهبود قابل توجهی از پارتو-جلو (قرار داده شده در auxiliary_external_list) در سه تقریب متوالی به

دست می آید و یا حداکثر سطح L به دست می آید. L یک پارامتر ورودی است. توجه داشته باشید که معیارهای

اتمام، توسط PO پردازنده، کنترل می شود.

• Create_new_individuals_paral (evals): رویه Create_new_individuals_paral ترتیبی، فضای جستجو را برای شناسایی

مناطق با راه حل های خوب بررسی می کند. برای رسیدن به یک تعادل بین اکتشاف و بهره برداری، روش ایجاد،

یک دنباله از اپراتورهای ژنتیکی را به کار می برد که با مقایسه با راه حل های ذخیره شده در external_list، از تجارب

انباشته استفاده می کند. روش Create_new_individuals_paral مشابه هفرای ترتیبی آن است. تنها تفاوت در طول

فهرست جمعیت به نظر می رسد که برابر با M برای مورد ترتیبی و برای نسخه موازی برابر با $M' = M/P$ است.

اندازه های لیست خارجی، در هر دو مورد، برابر با M هستند، اگر چه در نسخه موازی، `local_external_list` به جای `external_list` از نسخه ترتیبی استفاده می شود.

- `Select_individuals_paral` (فهرست): رویه `Select_individuals` ترتیبی، هنگامی که به حداکثر ظرفیت مجاز خود می رسد، طول لیست را کاهش می دهد. بنابراین، ارجح ترین افراد انتخاب خواهند شد (کیفیت ۳). توجه داشته باشید که، در نسخه موازی، هر پردازنده یک لیست محلی را حفظ می کند. انجام انتخاب ها با توجه به در نظر گرفتن اطلاعات محلی می تواند بهره وری معکوس از نظر اثربخشی داشته باشد، زیرا یک راه حل که در یک پردازنده خاص قابل ترجیح نیست، نمی تواند هنگام در نظر گرفتن همه لیست های محلی ترجیح شود. سپس، به منظور جلوگیری از کاهش در اثربخشی جهانی، انتخاب ها به صورت محلی در هر پردازنده انجام نمی شوند. در عوض، یک انتخاب جهانی با توجه به تمام لیست ها در P پردازنده انجام می شود. این می تواند نشان دهد که مقادیر زیادی از داده ها باید مکرر در میان پردازنده ها منتقل شوند. برای تسریع سربارهای ارتباطی و از این رو روند انتخاب، یک طرح ارتباطی درخت سلسله مراتبی طراحی شده است. اجازه دهید فرض کنیم که ریشه درخت، P_0 پردازنده است. آنگاه، P_0 پردازنده می تواند به عنوان جمع آوری کننده تمام اطلاعات منتقل قابل درک باشد. حداکثر عرض درخت سلسله مراتبی توسط تعداد پردازنده های موجود P داده شود که توسط P_{id} با $id \in [0, P - 1]$ شناسایی می شود. حداکثر تعداد مراحل آن توسط $stg^{max} = \log_2(P)$ داده می شود. هر مرحله مرتبط با یک رقم $stg \in [0, stg^{max}]$ است.

سه نوع پردازنده وجود دارد: فرستنده، گیرنده و پردازنده های بیکار. همانطور که می توان در شکل ۱ مشاهده نمود، نقش هر پردازنده از طریق مدل های ارتباطی متفاوت است. زمانی که یک پردازنده یک گیرنده است، یک لیست از یک پردازنده فرستنده را به دست می آورد، شامل یک لیست مشترک با توجه به لیست خود افراد و لیست دریافت شده می شود و `drank` مربوط به هر فرد را محاسبه می کند. سپس، یک انتخاب که وابسته به لیست منتقل شده و مرحله مدل ارتباطی تغییر، انجام می گیرد.

- اگر فهرست به local_population_list اشاره کند. در این مورد، اگر $stg < stg_{max}$ ، انتخاب تنها از لحاظ رتبه سلطه

انجام می شود. بطور دقیقتر، حداقل رتبه غیر سلطه d_{rank}^{min} به گونه ای که حداقل $2 \cdot M' \cdot 2^{stg+1}$ فرد با

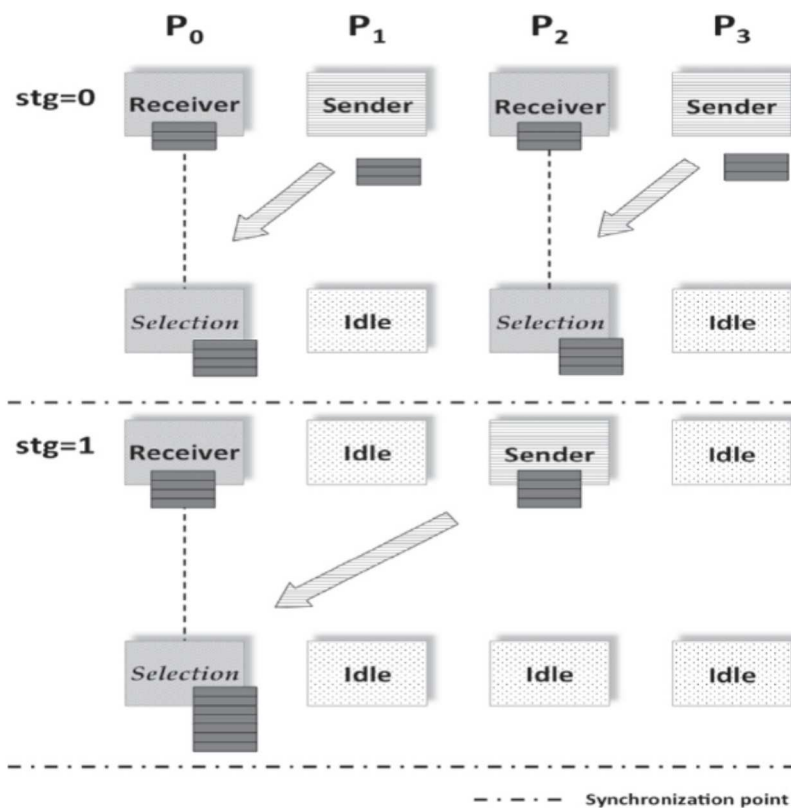
رتبه غیر سلطه کوچکتر از یا برابر با رتبه d_{rank}^{min} محاسبه شود. افراد با $d_{rank} \geq d_{rank}^{min}$ از فهرست مشترک

حذف می شوند. افراد باقی مانده از طریق درخت سلسله مراتبی ارتباطی منتقل خواهند شد. در آخرین مرحله از

رویه انتخاب، یعنی $stg = stg_{max}$ ، پردازنده گیرنده (P_0)، ریشه درخت سلسله مراتبی است، و M فرد ارجح

تر را با استفاده از عملگر مقایسه شلوع انتخاب می کند. سپس population_list حاصل در میان تمام پردازنده ها

توزیع خواهد شد، به گونه ای که M' فرد به طور مستقیم به هر پردازنده ارسال خواهند شد.



شکل ۱. رویه Select_individuals_parallel.

- اگر فهرست اشاره به local_external_list و $STG < stg_{max}$ اشاره کند، تنها افراد غیر تحت سلطه در فهرست مشترک

حفظ خواهند شد و به مرحله بعدی از درخت سلسله مراتبی ارتباطات منتقل خواهند شد. اگر

$stg = stg^{max}$ ، P0 پردازنده (گیرنده) به فهرست مشترک برای M افراد ارجح تر کاهش می یابد و در

local_external_list خود آن ذخیره خواهد کرد. سپس، P0، local_external_list خود را به طور مستقیم در میان تمام پردازنده های توضیح داده شده قبلی توزیع خواهد کرد.

- Improve_individuals_parallel (فهرست): این روش، موازی سازی رویه Improve_individuals (لیست) است، که در آن هر فرد در لیست ورودی فراخوانی یک روش بهبود است. توجه داشته باشید که پیاده سازی هر نوع بهینه ساز محلی به منظور کار با فرد می تواند. هنگام نامبردن فرد متعلق به population_list، روش محلی، از یک سو، هل دادن فرد به سمت پارتو-تنظیم شده و، از سوی دیگر، مطالعه اطراف آن را برای به دست آوردن راه حل های نامعین میسر می سازد که می توان در external_list درج نمود. گنجاندن نقاط نامشخص در external_list می تواند کیفیت نهایی پارتو-جلو را بهبود بخشد، اما تلاش محاسباتی (عناصر بیشتر در لیست به زمان محاسبات بیشتری مورد نیاز برای مدیریت external_list منجر می شود) را افزایش می دهد. با جستجو برای یک مصالحه بین کیفیت نهایی پارتو-جلو و تلاش محاسباتی، راه حل های نامعین در یک List_external قرار داده می شوند، زمانی که external_list به عنوان ورودی در Improving_method در نظر گرفته می شود.

Improve_individuals_parallel روش ترتیبی را در لیست های محلی مربوطه اجرا می کند. سپس، در مرحله ۵ از الگوریتم ۱، هر پردازنده، local_population_list خود را با طول برابر با M' بهبود می بخشد و به عنوان یک نتیجه، افراد می توانند با راه حل هایی جایگزین شوند که آنها بر آن تسلط دارند و نقاط جدید می توانند در local_external_list در مورد مشخص نمودن افراد نامشخص گنجانده شوند.

در مقابل، در مرحله ۸ الگوریتم ۱، هر پردازنده، روش بهبود برای local_external_list (با طول برابر با M') که فقط توسط P0 ارسال می شود، به کار می برد. توجه داشته باشید که افراد دریافت شده متعلق به local_external_list از P0 هستند که در این لحظه شامل ارجح ترین ها راه حل از همه لیست های خارجی می شوند. هنگامی که تمام افراد بهینه سازی شده باشند، آنها به پردازنده P0 با استفاده از مدل ارتباطی پیشنهاد فرستاده می شوند. اگر $stg < stg^{max}$ ،

روند انتخاب توسط پردازنده های گیرنده، افراد غیر سلطه را حفظ خواهد کرد. در مقابل، اگر $stg = stg^{max}$ ،

پردازنده P0, auxiliary_external_list را به افراد بهبود یافته دریافتی متصل خواهد کرد و یک روش انتخاب را اعماد خواهد کرد که در آن تنها M فرد ارجح تر حفظ می شوند. در نهایت، P0 به طور مستقیم، local_external_list را به هر پردازنده ارسال خواهد کرد که به auxiliary_external_list جدید آنها تبدیل می شود.

- Update_local_external_list: بعد از این رویه قبلی، local_population_list می تواند شامل افرادی باشد که شایسته گنجاندن در local_external_list است. سپس، شبیه برای نسخه ترتیبی، local_external_list با کپی برداری از افراد غیر تحت سلطه از local_population_list برای آن به روز می شود. البته، این به معنی این است که نقاط در local_external_list تحت سلطه نقاط جدید باید حذف شوند. در نهایت، یک رویه انتخاب روی local_external_list انجام می شود که در آن M راه حل ارجح تر انتخاب می شوند.
- Compose_pareto: راه حل ارائه شده توسط الگوریتم باید افراد M باشد. اگر تعداد افراد در auxiliary_external_list به این مقدار برسد، مجموعه ارائه شده به عنوان تقریبی از پارتو-تنظیم شده، مجموعه حفظ شده در آن فهرست خواهد بود. در غیر این صورت، لیست های جمعیت محلی به P0 با استفاده از طرح ارتباطات سلسله مراتب درخت توضیح داده شده قبلی فرستاده می شوند. که این لیست به auxiliary_external_list می پیوندد فرستاده شده، و M افراد بیشتر ترجیح داده شده به عنوان یک نتیجه ارائه شده است.

۴. تجزیه و تحلیل تجربی

برای نشان دادن قابلیت کاربرد FEMOEA-Paral، ما از آن برای حل یک مسئله محل تسهیلات رقابتی (و طراحی) که حل آن مشکل است استفاده نمودیم. به طور خاص، ما مسئله دو هدفه شرح داده شده در [۱۰] بازبینی نمودیم که به خاطر کامل بودن، به طور مختصر در بخش ۴,۱ شرح داده شده است.

۴,۱. یک مسئله طراحی و موقعیت یابی تسهیلات حق امتیازدهنده-حق امتیازگیرنده سطحی دو هدفه

یک امتیازدهنده می خواهد حضور خود در یک منطقه جغرافیایی خاص را با افتتاح یک مرکز جدید افزایش دهد. هر دوی حق امتیازدهنده (صاحب امتیاز) و حق امتیازگیرنده (مالک واقعی امکانات جدید که باید افتتاح شوند) دارای یک هدف هستند: به حداکثر رساندن سود خود. با این حال، به حداکثر رساندن سود به دست آمده توسط حق

امتیازدهنده معمولاً در تضاد با حداکثر سازی سود به دست آمده توسط حق امتیازگیرنده است، همانطور که خواهیم دید. در مدل، قرار است تقاضا، غیرکششی (ثابت) و متمرکز در برخی از نقاط تقاضا باشد که قدرت خرید آنها را در میان تمام امکانات متناسب با جذب آنها تقسیم می کند (نگاه کنید به [۱۳]). تابع جاذبه (و یا مطلوبیت) یک مشتری نسبت به یک مرکز معین، به فاصله بین مشتری و تاسیسات و همچنین به دیگر ویژگی های این تاسیسات که کیفیت آن را تعیین می کنند بستگی دارد.

نماد زیر در سراسر این مقاله استفاده خواهد شد. متغیرهای تسهیلات جدید (یعنی، متغیرهای مسئله) توسط $nf = (x, \alpha)$ نشان داده می شوند که در آن x نشاندهنده مکان تسهیلات جدید: $x = (x_1, x_2)$ است و

α کیفیت آن ($\alpha > 0$) است. در نظر بگیرید که q شاخص نقاط تقاضا باشد، $q = 1, \dots, q_{max}$ ، و r

شاخص تسهیلات موجود $r = 1, \dots, r_{max}$ است. pq مکان نقطه تقاضای q ام است؛ wq به تقاضا (یا قدرت

خرید) در pq اشاره می کند؛ erf مکان تسهیلات موجود r ام است؛ dqr ، فاصله بین pq و erf است؛ و α_{qr} ، کیفیت

erf تصور شده توسط pq است. با نماد قبلی، $\alpha_{qr}/g_q(d_{qr})$ ، جذابیت است که pq برای erf احساس می کند که

در آن $g_q(\cdot)$ ، یک تابع غیرکاهشی غیرمنفی است. علاوه بر این، γ_q ، وزن کیفیت تسهیلات جدید به صورت

درک شده توسط pq است و d_{qx} ، فاصله بین pq و تسهیلات جدید nf است. بنابراین $\gamma_q \alpha / g_q(d_{qx})$ ، جذابیت

را ارائه می دهد که pq برای nf احساس می کند. در نظر بگیرید که k تعداد تسهیلات موجود باشد که بخشی از حق

امتیاز (اولین k از تسهیلات r_{max} در این رده، $0 < k < r_{max}$ ، فرض می شوند) هستند. از مفروضات قبلی،

سهم بازار جذب شده توسط حق امتیازدهنده برابرست با

$$MK(nf) = \sum_{q=1}^{q_{max}} w_q \frac{\frac{\gamma_q \alpha}{g_q(d_{qx})} + \sum_{r=1}^k \frac{\alpha_{qr}}{g_q(d_{qr})}}{\frac{\gamma_q \alpha}{g_q(d_{qx})} + \sum_{r=1}^{r_{max}} \frac{\alpha_{qr}}{g_q(d_{qr})}}$$

ما فرض می کنیم که هزینه های عملیاتی برای حق امتیازدهنده مربوط به امکانات جدید، ثابت هستند. به این ترتیب، سود به دست آمده توسط حق امتیازدهنده یک تابع افزایشی از سهم بازار است. بنابراین، به حداکثر رساندن سود به دست آمده توسط حق امتیازدهنده معادل به حداکثر رساندن سهم بازار آن است. این هدف اول مسئله خواهد بود.

هدف دوم مسئله، به حداکثر رساندن سود به دست آمده توسط حق امتیاز، به صورت تفاوت بین درآمد به دست آمده از سهم بازار گرفته شده توسط امکانات جدید منهای هزینه های عملیاتی تصور شده آن است. سهم بازار گرفته شده توسط امکانات جدید (حق امتیازگیرنده) برابر است با

$$mk(nf) = \sum_{q=1}^{q_{\max}} w_q \frac{\frac{\gamma_q \alpha}{g_q(d_{qx})}}{\frac{\gamma_q \alpha}{g_q(d_{qx})} + \sum_{r=1}^{r_{\max}} \frac{\alpha_{qr}}{g_q(d_{qr})}}$$

و سود توسط عبارت زیر ارائه می شود،

$$\pi(nf) = F(mk(nf)) - G(nf),$$

که در آن $F(\cdot)$ ، یک تابع شدیداً افزایشی است که فروش مورد انتظار را تعیین می کند (یعنی درآمد اعطا شده) برای یک سهم معین بازار، $mk(nf)$ و $G(nf)$ ، یک تابع است که هزینه های عملیاتی یک تسهیلات واقع در x با کیفیت α را ارائه می دهد. در مطالعات محاسباتی ما، ما F به صورت خطی و G را به صورت قابل تفکیک به شکل $G(nf) = G_1(x) + G_2(\alpha)$ در نظر گرفته ایم که در آن $G_1(x) = \sum_{q=1}^{q_{\max}} \Phi_q(d_{qx})$ ، با $\Phi_q(d_{qx}) = w_q / ((d_{qx})^{\phi_{q0}} + \phi_{q1})$ ، $\phi_{q0}, \phi_{q1} > 0$ ، $G_2(\alpha) = e^{\alpha_0 + \alpha_1} - e^{\alpha_1}$ با $\alpha_0 > 0$ و α_1 با توجه به مقادیر (دیگر عبارات ممکن برای $G(nf)$ را می توان در [81] یافت).

مسئله در نظر گرفته شده به صورت زیر است

$$\begin{cases} \max & MK(nf), \\ \max & \pi(nf), \\ \text{s.t.} & d_{qx} \geq d_q^{\min} \quad \forall q, \\ & \alpha \in [\alpha_{\min}, \alpha_{\max}], \\ & x \in FR \subset \mathbb{R}^2, \end{cases} \quad (2)$$

که در آن پارامترهای $d_q^{\min} > 0$ و $\alpha_{\min} > 0$ آستانه های معین هستند که تضمین می کند که تسهیلات جدید روی یک نقطه تقاضا واقع شده است و دارای مینیمم سطح کیفیت است. پارامتر α_{\max} ، ماکزیمم مقداری است که کیفیت یک تسهیلات می تواند در عمل داشته باشد. با FR، ناحیه ای از صفحه را نشان می دهیم که تسهیلات جدید می توانند واقع شوند. توجه داشته باشید که (۲) یک مورد خاص از (۱) است که در آن $y = nf, f_1(y) = -MK(nf), f_2(y) = -\pi(nf)$ و مجموعه عملی S توسط محدودیت ها در (۲) داده می شود.

۴،۲ نتایج محاسباتی

تمام مطالعات محاسباتی در یک خوشه با ۱۸ گره اجرا شده اند. هر گره دارای ۱۶ گره (Intel Xeon E5 2650)، ۶۴ گیگابایت حافظه مشترک و ۱۲۸ گیگ درایو حالت-جامد است. در کل، ۲۸۸ هسته، ۱۱۵۱ گیگ حافظه و ۲۳۰۴ گیگ SSD. در این کار، هر نمونه با $P = 1, 2, 4, 8, 16, 32, 64$ پردازنده حل شده است. اجراها با $P = 1, 2, 4, 8, 16$ پردازنده در حالت تک اجرا شدند که در هر مورد، یک تعداد هسته ها برابر با P انتخاب می شوند. این مطالعات ۳۲ پردازنده با استفاده از ۲ گره ها و ۱۶ هسته در هر گره اجرا شدند. در نهایت، برای $P=64$ ، ۴ گره با ۱۶ هسته در هر گره انتخاب شدند. شبکه های ارتباط داخلی Infiniband و Ethernet هستند.

به منظور داشتن دید کلی در مورد عملکرد الگوریتم، انواع مختلف مسائل با تغییر q_{\max} ، عدد r_{\max} از تسهیلات موجود و تعداد k از تسهیلات موجود متعلق به زنجیره تولید شده اند. محیط های استفاده شده

$$(q_{\max} = 25, 50, r_{\max} = 2, k = 1), (q_{\max} = 25, 50, r_{\max} = 5, k = 1, 2)$$
 و $(2, 4)$

. برای هر محیط، یک نمونه با استفاده از انتخاب تصادفی از مسائل به طور یکنواخت درون بازه های پیش-تعریف شده تولید شد ([10] را ببینید). این مجموعه از ده مسئله را می توان از

<http://www.um.es/geloca/gio/AMCtestproblems>.

zip دانلود نمود. تعداد نقاط انتخاب شده برای تقریباً پارتو-جلو در $M = 400, 800, 1600, 3200, 6400$.

تنظیم شد. برای مقابله با اثر تصادفی بودن، هر مسئله ۱۰ بار اجرا می شود و مقادیر متوسط محاسبه شده اند.

اثر بخشی از یک سو، با چک کردن این مورد که آیا نقاط در مجموعه راه حل ارائه شده توسط FEMOEA- Paral تخمین مجموعه کارآمد، قطعاً نقاط کارآمد هستند، آزمایش شده است (و یا بسیار نزدیک به نقاط کارآمد هستند). برای این هدف، شبیه به آنچه که در [۲۲] با نسخه های ترتیبی انجام شد، بررسی می کنیم که آیا نقاط راه حل در کادرهای راه حل مربوطه ارائه شده توسط روش شاخه و حد بازه $iB\&B$ [۱۰] گنجانده می شوند. از سوی دیگر، برای اندازه گیری خوبی یک تقریب برای طیف پارتو-جلو، به اصطلاح شاخص های پروولوم [32] نیز محاسبه می شود. این شاخص، های پروولوم بخشی از فضای معیار را اندازه گیری می کند که به طور ضعیف توسط مجموعه تقریب تحت سلطه است (شکل ۲- سمت چپ). هر چه های پروولوم بالاتر باشد، تقریب بهتر است. به منظور اندازه گیری این مقدار، یک نقطه مرجع که تحت تسلط تمام نقاط قرار می گیرد، مورد نیاز است (نقطه RP در شکل ۲). برای یک مسئله معین، همان نقطه مرجع باید برای تمام الگوریتم ها و همه اجراها استفاده شود. در مطالعات محاسباتی ما، ما نقاط تمام مجموعه های تقریبی پارتو-جلو را با هم (برای مقادیر ثابت P و M ، هر مسئله را ۱۰ بار حل می کنیم، در نتیجه برای هر مسئله، که ما ۳۵۰ مجموعه تقریب داریم) در نظر می گیریم و ما باید به عنوان نقطه مرجع استفاده کنیم که مولفه لام آن، حداکثر تمام اجزاء لام از آن نقطه است. این یک تقریب از نقطه Nadir است.

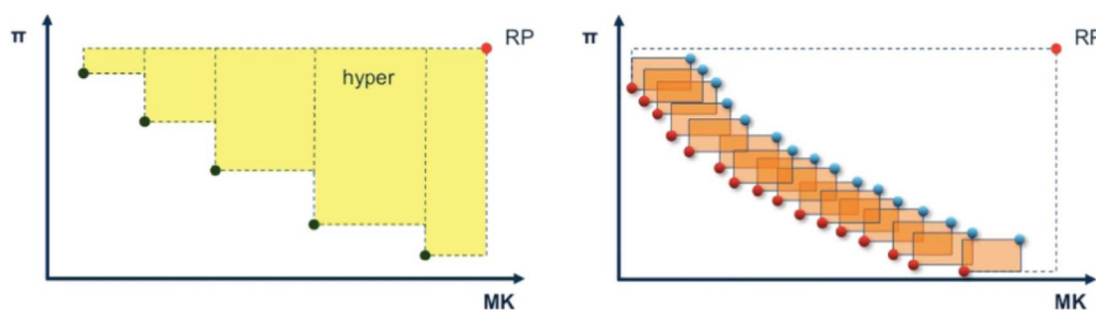
برای هر مسئله $(j = 1, \dots, 10)$ ، $IB \& B$ ، به عنوان یک راه حل، یک لیست از جعبه ها را ارائه می دهد (فواصل چند بعدی) که اتحاد آنها شامل مجموعه کامل کارآمد و تصاویر آنها شامل پارتو-جلو متناظر می شود. برای محاسبه های پروولوم آن راه حل، گوشه بالای سمت راست و پایین سمت چپ جعبه در فضای تصویر به دست می آید (شکل ۲- سمت راست). آن نقاط، دو مجموعه تقریب را تشکیل می دهند. یک فاصله $[lowH_j, uppH_j]$ که شامل های پروولوم دقیق پارتو-جلو می شود، به شرح زیر قابل دستیابی است. حد پایین تر آن $lowH_j$ ، های پروولوم به دست آمده با مجموعه ای تشکیل شده از گوشه های بالای سمت راست از جعبه هاست، و مرز بالاتر آن، $uppH_j$ ، های پروولوم به دست آمده با مجموعه ای متشکل از گوشه پایین سمت چپ است. ذکر این مورد مهم است که قبل از محاسبه آن های پروولوم ها، نقاط تحت سلطه از هر مجموعه حذف می شوند. به خاطر اختصار، فواصل خاص به دست آمده برای هر مسئله نشان داده نشده اند. اما به منظور داشتن یک دید کلی، میانگین های حدود بالا و پایین برای

مجموعه ای کامل از ده مسئله شده محاسبه شده است، $[lowH, uppH] = [361.9153, 364.0888]$. زمان

محاسبه متوسط صرف شده توسط iB&B در حل نمونه ها، ۲۳،۵ ساعت است.

توجه داشته باشید که هایپروولوم با افزایش تعداد نقاط نزدیک به افزایش پارتو-جلو بزرگتر می شود. در این معنا، رفتار هر دوی FEMOEA و FEMOEA-Paral زمانی که تعداد نقاط نزدیک پارتو-جلو تخمین زده می شوند، M ، مطالعه شده است.

لازم به ذکر است که FEMOEA-Paral پارتو-جلو را با موفقیت ۱۰۰٪ تخمین زده است، به عنوان مثال، در تمام اجراها و برای تمام مسائل، تمام نقاط در جعبه های مربوطه ارائه شده توسط IB & B قرار گرفتند. علاوه بر این، مقادیر هایپروولوم همواره در فاصله $[lowH_j, uppH_j]$ برای هر نمونه خاص $j, j = 1, \dots, 10$ گنجانده شده است. همانطور که انتظار می رود، می توان مشاهده کرد (جدول ۱) که هایپروولوم با تعداد نقاط M در پارتو-جلو افزایش می یابد. علاوه بر این، هایپروولوم به دست آمده توسط نسخه موازی شبیه به مقدار به دست آمده با روش ترتیبی است. این یک چالش بوده است، زیرا نسخه ترتیبی دارای کنترل کلی بر جمعیت و لیست های خارجی است، در حالی که نسخه موازی، برخی تصمیمات را مبتنی بر دانش محلی می گیرد. رویه انتخاب طراحی شده به مقابله با این اشکال کمک نمود.



شکل ۲. محاسبه مقدار هایپروولوم برای یک مجموعه تخمین (سمت چپ) و مجموعه های تخمین در نظر گرفته

شده برای الگوریتم iB&B (سمت راست)

نتایج متوسط به دست آمده توسط FEMOEA گروه بندی شده مطابق با مقادیر M و P

P	M = 400			M = 800			M = 1600			M = 3200			M = 6400		
	Hyper	SD	Eff	Hyper	SD	Eff	Hyper	SD	Eff	Hyper	SD	Eff	Hyper	SD	Eff
1	363.0677	0.0073	1007	363.4107	0.0023	3528	363.5803	0.0005	14532	363.6609	0.0001	65219	363.6997	0.0000	343761
2	363.1025	0.0064	1.41	363.4256	0.0014	1.48	363.5852	0.0004	1.72	363.6626	0.0001	2.05	363.7003	0.0000	2.66
4	363.0996	0.0035	1.69	363.4261	0.0020	1.80	363.5851	0.0006	2.24	363.6626	0.0002	2.86	363.7003	0.0000	4.12
8	363.1018	0.0050	1.60	363.4264	0.0016	1.66	363.5846	0.0007	2.60	363.6625	0.0002	3.79	363.7003	0.0000	4.99
16	363.1018	0.0060	1.18	363.4252	0.0023	1.43	363.5850	0.0004	2.02	363.6624	0.0002	2.98	363.7003	0.0000	3.92
32	363.1029	0.0062	0.71	363.4257	0.0014	0.92	363.5851	0.0005	1.23	363.6625	0.0001	1.83	363.7002	0.0000	2.66
64	363.1029	0.0042	0.37	363.4256	0.0018	0.48	363.5846	0.0004	0.65	363.6623	0.0002	0.96	363.7002	0.0000	1.46

برای اندازه گیری تلاش محاسباتی در به دست آوردن راه حل، معیار بازده، Eff، که تخمین می زند که استفاده از پردازنده در حل مسئله چقدر خوب است، محاسبه شده است. بازده یک نسخه موازی (روی P پردازنده اجرا می شود)

با توجه به مورد ترتیبی بدین صورت محاسبه می شود $Eff(P) = \frac{T(1)}{P \cdot T(P)}$ ، که در آن T(1)، زمان اجرا بر روی یک پردازنده تک است و T(P)، زمان اجرا روی P پروسسور است. توجه داشته باشید که بازده ایده آل، 1 است.

علاوه بر این، مقیاس پذیری نسخه موازی نیز آزمایش شده است. به طور کلی، این مفهوم را می توان به عنوان توانایی یک سیستم، الگوریتم، و یا روند، در حل یک مقدار رو به رشد کار به شیوه ای توانمند و یا توانایی آن برای بزرگنمایی برای جای دادن رشد درک نمود. استفاده از معیارهای عملکرد مانند بهره وری به ما اجازه می دهد تا تعیین کنیم که آیا یک الگوریتم مقیاس پذیر است: یک الگوریتم که مقیاس ها به خوبی قادر به حفظ تعادل و یا حتی افزایش سطح آن از عملکرد و یا بهره وری باشد، زمانی که در موارد بیشتر خواستار آزمایش خواهد بود.

به خاطر اختصار، نتایج خاص برای هر مسئله خاص نشان داده نشده است. اما به منظور داشتن یک دید کلی،

متوسط نتایج بهره وری برای مقادیر مختلف P و گروه بندی آنها با توجه به مقدار M، تفصیل می شوند (ستون \overline{Eff} در جدول 1 ببینید). توجه داشته باشید که برای $P = 1$ ، زمان محاسبات (برحسب ثانیه) به جای آن به تصویر کشیده شده است. هایپروولوم متوسط و متوسط مربوطه از انحرافات استاندارد نیز گنجانده شده است. به عبارت دقیق تر، برای مقادیر M و P ثابت، برای هر یک از 10 مسائل (هر یک از آنها 10 بار اجرا می شوند) ما متوسط هایپروولوم و متناظر با انحراف استاندارد آن $(\overline{Hyper}_j, SD_j), j = 1, \dots, 10$ را محاسبه نمودیم. مقادیر در جدول 1،

$$\overline{SD} = \sum_{j=1}^{10} SD_j / 10 \text{ و } \overline{Hyper} = \sum_{j=1}^{10} \overline{Hyper}_j / 10 \text{ هستند.}$$

توجه داشته باشید که، طبق انتظار، \overline{Hyper} همیشه در بازه $[lowH, uppH]$ گنجانده می شود همچنین توجه کنید که هر قدر M بالاتر باشد، انحراف استاندارد کوچکتر است. این تعجب آور نیست، زیرا نقاط بیشتر در مجموعه تخمین پارتو-جلو، هر قدر تقریب بهتر باشد، تفاوت بین مجموعه تقریب می تواند کوچکتر باشد، و تفاوت های پروولوم متناظر آن کوچکتر است. در مورد بهره وری، تا ۸ پردازنده افزایش می یابد، و سپس شروع به کاهش می نماید. با این حال، نتایج امیدوار به دست آمده است و مقادیر بسیار برتر نسبت به حالت ایده آل در بسیاری از موارد به دست آمده اند. (با مقدار M).

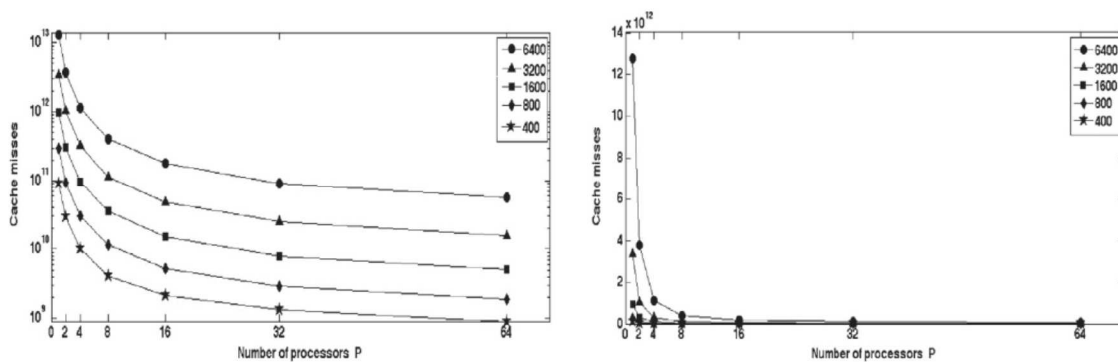
رفتار نسخه موازی و از اینرو بازده های به دست آمده عمدتاً به دو دلیل هستند: i: توزیع فهرست ها در میان پردازنده های در دسترس که موجب کاهش خطاهای کش و از اینرو زمانی دسترسی به حافظه می شود؛ و ii: سربارهای تحمیل شده توسط تبادل اطلاعات در میان پردازنده ها در روش `Select_individuals_parallel (list)` رخ می دهد. این نتایج از یک مطالعه انجام شده توسط و استنباط شده است `TAU (Tuning and Analysis Utilities) Performance` [27] `System [18] and PAPI (Performance Application Programming Interface)`. ذکر این مورد مهم است که تمایلات مشاهده شده در جدول ۱، نتیجه مقادیر متوسط محاسبه شده نیستند. در مقابل، این رفتار برای تمام نمونه های خاص مشابه است. بدین دلیل است که تنها نتایج به دست آمده برای یک مسئله خاص (یکی با محیط $(50, 10, 4)$) بعداً نشان داده خواهد شد، زمانی که با استفاده از `TAU Performance System` تحلیل شود.

یکی از نقاط کلیدی مطالعه شده با استفاده از `TAU` و `PAPI`، خطاهای کش کلی به دست آمده توسط `FEMOEA (P = 1)` و `FEMOEA-Paral (P > 1)` برای هر مقدار M است. شکل ۳، آن دسته از نتایج را برای نمونه $(50, 10, 4)$ نشان می دهد. همانطور که می توان دید، با توجه به یک مقدار M ، تعداد خطاهای کش، به طور نمایی تا حدود ۸ پردازنده کاهش می یابد. از آن نقطه به بعد، چنین کاهش تدریجی چندان مهم نیست، اما بسیار کندتر است. در کنار اینها، توجه داشته باشید که هر قدر مقدار M بزرگتر باشد، تعداد خطاهای کش در نسخه ترتیبی و بنابراین صرفه جویی های زمانی به دست آمده توسط نسخه های موازی ناشی از کاهش خطاهای

کش بزرگ نیست. البته این حقایق به طور مشخص به بازده های به دست آمده مرتبط می شوند که نسبت به مورد ایده آل و برتر هستند و با افزایش مقدار M بالاتر باشند.

زمانی محاسباتی مرتبط با روش `Select_individuals_parallel` نیز با استفاده از TAU محاسبه شده است. رفتار چنین روشی و از اینرو نتایجی که می توان به دست آورد قابل برون یابی به هر فهرست ورودی (با هر طول) مطالعه شده است. بنابراین، برای جستجوی شجاعت، تنها یک نمونه خاص نشان داده خواهند شد. به طور دقیق تر، این مورد با تنظیم $(50, 10, 4), M = 3200$ و فهرست به نام `local_population_list` در نظر گرفته خواهند شد.

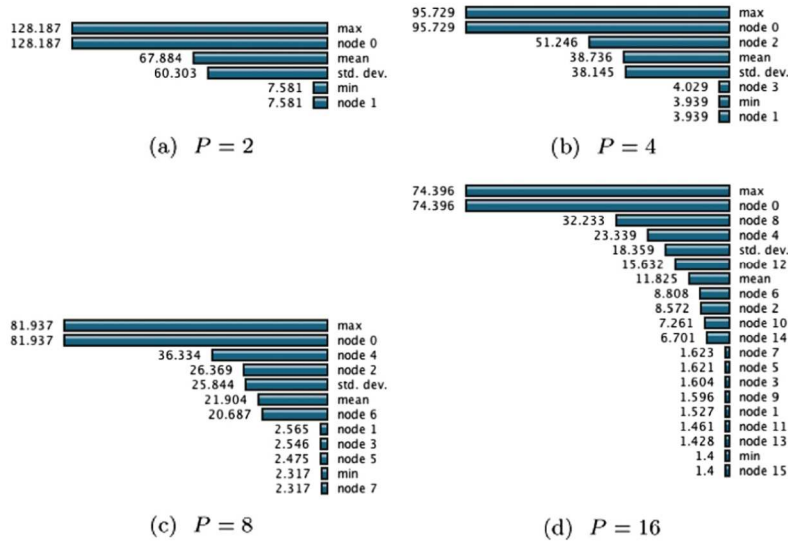
شکل ۴ نشاندهنده زمان محاسباتی به کار گرفته شده توسط هر پردازنده (گره نامبرده در شکل) برای اجرای این روش، برای آن نمونه خاص و زمانی که $P = 2, 4, 8, 16$ هستند. گرافیک های مرتبط با $P = 32, 64$ برای کاهش طول مقاله حذف شده اند، هرچند همین تمایل مشاهده شد. علاوه بر این، توجه داشته باشید که برخی از ارقام اضافی ارائه شده اند، یعنی، ماکزیمم، متوسط و مینیمم زمان های اجرای صرف شده توسط پردازنده ها نشان داده می شوند. انحراف استاندارد نیز به تصویر کشیده شده است. همانطور که می توان دید، یک عدم تعادل بالاتر در محاسبه زمان های مرتبط با هر پردازنده وجود دارد



شکل ۳: تنظیم $(50, 10, 4)$ ، متوسط از دست رفتن های کلی کش به دست آمده توسط FEMOEA و

نقطه در پارتو-جلو (a) در سمت چپ در مقیاس FEMOEA-Paral با $400, 800, 1600, 3200$ and 6400

خطی، (b) در سمت راست در مقیاس لگاریتمی



شکل ۴. تنظیم $(50, 10, 4)$. زمان محاسباتی به کار گرفته شده در
 با `Select_individuals_paral (local_population_list)`

$P = 2, 4, 8, 16$ and $M = 3200$.

طرح درخت ارتباط را برای درک دلایل پشت چنین عدم تعادلی به یاد داشته باشید. پیروی نماد قبلاً تعریف شده در این مقاله، پردازنده ها با عدد هویت فرد در ابتدا دریافت کننده خواهند بود، در حالیکه پردازنده ها با عدد زوج، فرستنده ها خواهند بود. پردازنده P0 (گره ۰ در شکل) نشاندهنده کلکتور است. همانطور که انتظار می رود، همیشه فعال است و از اینرو زمان محاسبه آن همیشه ماکزیمم زمان است. در کنار اینها، تعداد پردازنده های بیکار آماده به کار موجب افزایش مقدار P می شود. این نتایج به طور مشخص نشان می دهند که رویه انتخاب یک تنگنا است و یک کاهش در بازده نسخه موازی را موجه می شود. با این حال، توجه داشته باشید که برخلاف هزینه های زمانی (ناشی از زمان های ارتباطات و انتظار تحمیل شده توسط این روش)، مقادیر بازده خوب به دست می آیند. این بدان دلیل است که این سربارها توسط صرفه جویی های زمانی مرتبط با دسترسی های حافظه قبلاً اظهار شده جبران می شوند. توجه داشته باشید که زمانی که $P > 8$ ، آن صرفه جویی ها چندان چشمگیر نیستند و بنابراین اثرات هر دوی زمان های ارتباطات و انتظار دارای تاثیر بیشتر بر بازده هستند. این حقیقت، وجود یک نقطه عدم انعطاف در مقادیر

بازده را نشان می دهد زمانی که $P=8$, یعنی, بازده برای حدود ۸ پردازنده افزایش می یابد و بنابراین کاهش آغاز می شود. رویه انتخاب برای توانایی در حفظ اثربخشی نسخه ترتیبی طراحی شد؛ دیگر تصمیمات که می توانند این کمبود را بهبود بخشند به نوبه خود موجب کاهش کیفیت راه حل های به دست آمده می شوند.

۵. نتایج

برای حل مسائل بهینه سازی دارای مشکل حل کردن و چندهدفه, همانند بسیاری از مسائل موقعیت یابی رقابتی, یک نسخه موازی از FEMOEA به نام FEMOEA-Paral توسعه و تحلیل شده است. یک مطالعه محاسباتی جامع نشان داده است که FEMOEA-Paral, اثربخشی نسخه ترتیبی را حفظ می کند, یعنی هر دو نسخه, پارتو-جلو را با ۱۰۰٪ موفقیت در تمام نمونه ها تخمین می زنند, مقادیر هایپرولوم آنها همیشه در بازه فراهم شده توسط iB&B گنجانده می شوند و آنها هر دو, مقادیر هایپرولوم مشابه را برای هر نمونه خاص به دست می آورند. نگهداری مقادیر اثربخشی به لطف رویه انتخاب پیاده سازی شده ممکن می شود که به طور همزمان انتخاب ارجح ترین راه حل ها را ممکن می سازد. بازده نسخه موازی نیز تست شده است. توزیع بار محاسباتی انجام شده توسط FEMOEA-Paral تسریع زیاد زمان های محاسباتی ترتیبی را برای ما میسر می سازد, به گونه ای که FEMOEA-Paral قادر به کسب مقادیر بازده بسیار بالاست. علاوه بر این, مقیاس پذیری نسخه موازی نیز توسط حل نمونه ها با بار محاسباتی بیشتر نشان داده شده است.

نتایج بهره وری با استفاده از TAU تحلیل شده است. می توان نتیجه گرفت که سربار موازی سازی با تعداد پردازنده افزایش می یابد. علاوه بر این, مشخص شد که در حالی که توزیع جمعیت در میان پردازنده ها موجب صرفه جویی در زمان CPU نسبت به نسخه های ترتیبی می شود (تعداد خطاهای کش کوچکتر است), مخالف آن با رویه انتخاب برقرار است, که به عنوان یک تنگنا در الگوریتم عمل می کند. روش های انتخاب جدید, قادر به حفظ اثربخشی هستند, اما بدون ایجاد عدم تعادل بار, باید مورد بررسی قرار گیرند. علاوه بر این, روش ها برای کاهش خطاهای کش

برای نسخه ترتیبی و همچنین برای نسخه موازی باید مطالعه شوند.. یک مطالعه تحلیلی نیز باید برای توصیف بهبودی که می توان با موازی سازی کسب نمود انجام شود.

References

- [1] E. Alba, G. Luque, S. Nesmachnow, Parallel metaheuristics: recent advances and new trends, *Int. Trans. Oper. Res.* 20 (1) (2013) 1–48.
- [2] E.E. Ammar, E.A. Youness, Study on multiobjective transportation problem with fuzzy numbers, *Appl. Math. Comput.* 166 (2) (2005) 241–253.
- [3] C.A.C. Coello, G.B. Lamont (Eds.), *Applications of Multi-objective Evolutionary Algorithms*, World Scientific, Singapore, 2004.
- [4] C.A.C. Coello, G.B. Lamont, D.A. Van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems*, second edition, Genetic and Evolutionary Computation, Springer, New York, 2007.
- [5] A.L. Custódio, J.F.A. Madeira, A.L.F. Vaz, L.N. Vicente, Direct multisearch for multiobjective optimization, *SIAM J. Optim.* 21 (3) (2011) 1109–1140.
- [6] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.* 6 (2) (2002) 182–197.
- [7] Y.G. Evtushenko, M.A. Pospypkin, Nonuniform covering method as applied to multicriteria optimization problems with guaranteed accuracy, *Comput. Math. Math. Phys.* 53 (2) (2013) 144–157.
- [8] J. Fernández, B. Pelegrín, F. Plastria, B. Tóth, Solving a Huff-like competitive location and design model for profit maximization in the plane, *Eur. J. Oper. Res.* 179 (3) (2007) 1274–1287.
- [9] J. Fernández, B. Tóth, Obtaining an outer approximation of the efficient set of nonlinear biobjective problems, *J. Global Optim.* 38 (2) (2007) 315–331.
- [10] J. Fernández, B. Tóth, Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods, *Comput. Optim. Appl.* 42 (3) (2009) 393–419.
- [11] J. Figueira, S. Greco, M. Ehrgott (Eds.), *Multiple Criteria Decision Analysis: State of the Art Surveys*, Kluwer, New York, 2005.
- [12] D. Fotakis, E. Sidiropoulos, A new multi-objective self-organizing optimization algorithm (MOSOA) for spatial optimization problems, *Appl. Math. Comput.* 218 (9) (2012) 5168–5180.
- [13] S.L. Hakimi, Locations with spatial interactions: competitive locations and games, in: R.L. Francis, P.B. Mirchandani (Eds.), *Discrete Location Theory*, Wiley/Interscience, 1990, pp. 439–478.
- [14] D. Jaeggi, G. Parks, T. Kipouros, J. Clarkson, A multi-objective tabu search algorithm for constrained optimisation problems, in: C.A.C. Coello, A.H. Aguirre, E. Zitzler (Eds.), *Evolutionary Multi – A Criterion Optimization*, Third International Conference, EMO 2005, Lecture Notes in Computer Science, vol. 3410, Springer, 2005, pp. 490–504.
- [15] A. Lančinskas, J. Žilinskas, Approaches to parallelize Pareto ranking in NSGA-II algorithm, in: R. Wyrzykowski, J. Dongarra, K. Karczewski, J. Waśniewski (Eds.), *Parallel Processing and Applied Mathematics*, Lecture Notes in Computer Science, vol. 7204, Springer, Berlin, Heidelberg, 2012, pp. 371–380.
- [16] A. Lančinskas, J. Žilinskas, Solution of multi-objective competitive facility location problems using parallel NSGA-II on large scale computing systems, in: Pekka Manninen, Per Öster (Eds.), *Applied Parallel and Scientific Computing*, Lecture Notes in Computer Science, vol. 7782, Springer, Berlin, Heidelberg, 2013, pp. 422–433.
- [17] A.J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J.J. Durillo, A. Beham, AbYSS: adapting scatter search to multiobjective optimization, *IEEE Trans. Evol. Comput.* 12 (4) (2008) 439–457.
- [18] Dpt. of Computer and Information Science University of Oregon, Advanced Computing Laboratory LANL NM, and Research Centre Julich ZAM Germany. TAU (Tuning and Analysis Utilities). <<http://www.cs.uoregon.edu/research/tau/home.php>>.
- [19] P.M. Ortigosa, I. García, M. Jelásity, Reliability and performance of UEGO, a clustering-based global optimizer, *J. Global Optim.* 19 (3) (2001) 265–289.
- [20] M.S. Osman, M.A. Abo-Sinna, A.A. Mousa, An effective genetic algorithm approach to multiobjective routing problems (MORPs), *Appl. Math. Comput.* 163 (2) (2005) 769–781.
- [21] P.M. Pardalos, I. Steponavičiute, A. Žilinskas, Pareto set approximation by the method of adjustable weights and successive lexicographic goal programming, *Optim. Lett.* 6 (4) (2012) 665–678.
- [22] J.L. Redondo, J. Fernández, J.D. Álvarez, A.G. Arrondo, and P.M. Ortigosa, Approximating the Pareto-front of a planar bi-objective competitive facility location and design problem, *Comput. Oper. Res.* doi:<http://dx.doi.org/10.1016/j.cor.2014.02.013>.
- [23] J.L. Redondo, J. Fernández, P.M. Ortigosa, FEMOEA: a fast and efficient multi-objective evolutionary algorithm, submitted for publication. <http://www.um.es/geloca/gio/FEMOEA.pdf>.
- [24] D. Scholz, *Deterministic global optimization: geometric branch-and-bound methods and their applications*, Optimization and its Applications, Springer, 2012.
- [25] P. Shelokar, V. Jayaraman, B. Kulkarni, Ant algorithm for single and multiobjective reliability optimization problems, *Qual. Reliab. Eng. Int.* 18 (6) (2002) 497–514.
- [26] R.G. Strongin, Y.D. Sergeyev, *Global optimization with non-convex constraints: sequential and parallel algorithms*, Nonconvex Optimization and its Applications, Kluwer Academic Publishers, Dordrecht, 2000.
- [27] Innovative Computing Laboratory University of Tennessee (U.S.A.), PAPI (Performance Application Programming Interface). <<http://icl.cs.utk.edu/papi/>>.
- [28] A. Žilinskas, A one-step worst-case optimal algorithm for bi-objective univariate optimization, *Optim. Lett.* (2013), <http://dx.doi.org/10.1007/s11590-013-0712-8>.
- [29] D.J. White, A bibliography on the applications of mathematical programming multiple-objective methods, *J. Oper. Res. Soc.* 41 (8) (1990) 669–691.
- [30] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, *IEEE Trans. Evol. Comput.* 11 (6) (2007) 712–731.
- [31] Q. Zhang, H. Li, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, *IEEE Trans. Evol. Comput.* 13 (2) (2009) 284–302.
- [32] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: methods and applications* [Master's thesis], Swiss Federal Institute of Technology (ETH) Zurich, Shaker Verlag, Germany, 1999.
- [33] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization, in: K.C. Giannakoglou, D.T. Tsahalis, J. Périaux, K.D. Papailiou, T. Fogarty (Ed.), *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, International center for numerical methods in engineering (CIMNE), Athens, Greece, 2002, pp. 95–100.



این مقاله، از سری مقالات ترجمه شده رایگان سایت ترجمه فا میباشد که با فرمت PDF در اختیار شما عزیزان قرار گرفته است. در صورت تمایل میتوانید با کلیک بر روی دکمه های زیر از سایر مقالات نیز استفاده نمایید:

لیست مقالات ترجمه شده ✓

لیست مقالات ترجمه شده رایگان ✓

لیست جدیدترین مقالات انگلیسی ISI ✓

سایت ترجمه فا ؛ مرجع جدیدترین مقالات ترجمه شده از نشریات معتبر خارجی