علوم و مهندسی کامپیوتر
نشریه علمی پژوهشی انجمن کامپیوتر ایران
مجلد ۱۲، شماره ۱ (الف)، ۱۳۹۳
صفحات ۲۱-۳۰
مقاله عادی

زمانبندی کارا، سریع و متوازن در گریدهای محاسباتی با یک الگوریتم مورچگان جدید

مصطفی صفرپور غلامحسین دستغیبی‌فرد سیدعلی میرسلیمانی

دانشکده مهندسی برق و کامپیوتر، دانشگاه شیراز، شیراز، ایران

چکیده

در محیط‌های محاسباتی ناهمگون همانند گرید محاسباتی، زمانبندی یا نگاشت کارآمد کارهای محاسباتی مستقل مسأله‌ای بسیار مهم است. این مسأله از نوع مسائل NP-سخت به‌شمار رفته و برای حل آن از الگوریتم‌های اکتشافی و فرااکتشافی استفاده می‌شود. در این میان، بهینه‌سازی کلونی مورچگان به عنوان روشی کارآمد در حل مسائل NP- سخت شناخته شده است. در میان انواع الگوریتم‌هایی که در زیرمجموعه‌ی این روش قرار می‌گیرند، الگوریتم سیستم کلونی مورچگان به عنوان یک روش توانا و قدرتمند شناخته می‌شود که از قانون انتساب شبه‌تصادفی استفاده کرده و دارای یک رویکرد اضافی در اعمال مکانیزم‌های بهنگام‌سازی فرومن، به‌نام بهنگام‌سازی محلی می‌باشد.

در این مقاله، نخستین‌بار یک الگوریتم سیستم کلونی مورچگان جدید که در آن تکنیک‌های نوینی برای حل مسأله‌ی زمانبندی در گرید محاسباتی به‌کار رفته ارائه شده است. قابل توجه است که این الگوریتم رویکردی ساده داشته و از تکنیک‌هایی چون جستجوی محلی، ترکیب یا بذرپاشی استفاده نمی‌کند. نتایج حاصل از آزمایش‌ها نشان می‌دهند که الگوریتم پیشنهادی در مقایسه با الگوریتم‌های پیشین حداقل درپنج نمونه از محک مورد مطالعه به نتایج بهتری در مدت زمان بسیار کمتری دست می‌یابد و در بقیه‌ی موارد نیز در زمان بسیار کمتری نسبت به کارهای پیشین، الگوریتم پیشنهادی در رده‌ی دوم بوده و بر الگوریتم ژنتیک در تمامی موارد غلبه می‌کند.

کلمات کلیدی: محاسبات ناهمگون، گرید محاسباتی، زمان‌بندی، بهنیه‌سازی کلونی مورچگان، زمان اجرای کل.

۱- مقدمه

در دنیای امروز، مسائل پیچیده‌ای وجود دارند که حل آن‌ها نیازمند پردازش سریع و کارآمد است. رویکرد رایج محققان برای حل این‌گونه مسائل، استفاده از محیط‌های پردازش توزیع شده است. یک بستر مشترک برای پردازش توزیع شده معمولا شامل مجموعه‌ای ناهمگون از کامپیوترها است. محاسبات مشبک (گرید محاسباتی)، مجموعه‌ای از تکنیک‌های پردازش توزیع شده است که بر روی یک فرا رایانه‌ی بزرگ مجازی کار می‌کنند. این محیط مجازی از ترکیب بسیاری از بسترهای ناهمگون با ویژگی‌های مختلف ساخته شده است. این زیرساخت، امکان دسترسی فراگیر و مقرون به صرفه به منابع پردازشی توزیع‌شده برای حل مسائل سخت را فراهم می‌کند [۱، ۲].

هنگامی که از محیط‌های پردازش توزیع شده ناهمگون همچون گرید محاسباتی استفاده می‌کنیم، یک مشکل مهم پیدا کردن یک استراتژی زمانبندی برای اجرای مجموعه‌ای از کارها بر روی مجموعه‌ای از منابع محاسباتی (ماشین) می‌باشد. هدف این زمانبندی اختصاص منابع محاسباتی به کارها با برآورده کردن برخی معیارهای بهره‌وری مانند زمان اجرای کل می‌باشد. مسائل زمانبندی در سیستم‌های چند پردازنده‌ای همگون به طور گسترده مورد مطالعه قرار گرفته‌اند [۳، ۴]. با این حال، مسأله‌ی زمانبندی گرید محاسباتی به دلیل محبوبیت پردازش توزیع‌شده و رشد روزافزون در استفاده از خوشه‌های ناهمگون از سال‌ها پیش مورد توجه قرار گرفته است [۵، ۶].

مسائل زمانبندی سنتی NP-سخت هستند [۷]، یعنی روش‌های کلاسیک مانند برنامه‌ریزی پویا فقط برای حل نمونه‌هایی از مسأله با اندازه‌ی کوچک مفید

می‌باشند. الگوریتم‌های فرااکتشافی روش‌های امید بخشی برای حل مسأله‌ی زمانبندی هستند، چرا که قادر به تولید زمانبندی‌های کارآمد در زمان قابل قبول بوده و برای نمونه‌های بزرگی از مسأله هم این امر صادق است. بهینه‌سازی کلونی مورچگان به عنوان یک روش فرااکتشافی انعطاف‌پذیر و قوی برای حل مسأله‌ی زمانبندی پردازش ناهمگون مطرح شده و سطح بالایی از کارایی را در حل مسائل مربوط به بسیاری زمینه‌های کاربردی دیگر نیز نشان داده است [۸، ۱۹].

در این مقاله ما یک الگوریتم سیستم کلونی مورچگان را با ابتکار و قوانین بهنگام‌سازی فرومن جدید برای حل مسأله‌ی زمانبندی پردازش ناهمگون و گرید محاسباتی ارائه می‌کنیم. بخش‌های این مقاله که در ادامه می‌آیند به این ترتیب سازماندهی شده‌اند: بخش دوم به مروری بر مطالعات اخیر انجام شده در این حوزه می‌پردازد. در بخش سوم به فرموله‌سازی مساله‌ی زمانبندی در گرید محاسباتی پرداخته شده است. بخش چهارم کاربرد الگوریتم‌های کلونی مورچگان را برای مساله‌ی زمانبندی در گرید محاسباتی نشان می‌دهد. الگوریتم سیستم کلونی مورچگان جدید به همراه فرمول‌های مسأله در بخش پنجم آورده شده است. نتایج حاصل از آزمایشات در بخش ششم و جمع‌بندی و کارهای آینده نیز در بخش هفتم بیان شده‌اند.

۲- کارهای پیشین

در این بخش به بررسی مجموعه‌ای از الگوریتم‌های اکتشافی همچون توازن‌بار فرصت‌طلبانه، کمترین زمان اجرا، کمترین زمان اتمام، کمینه-کمینه و بیشینه-کمینه و نیز الگوریتم‌های فرااکتشافی نظیر الگوریتم ژنتیک، جستجوی ممنوعه و کلونی مورچگان که برای زمانبندی کارها در محیط پردازش ناهمگون مانند گرید محاسباتی طراحی شده‌اند می‌پردازیم. کارها در گرید محاسباتی مستقل و بدون هیچ وابستگی بوده و به ماشین‌های موجود به صورت ایستا منتسب می‌شوند (زمانبندی دسته‌ای یا یکجا). همچنین هر ماشین در گرید محاسباتی یک کار را در یک زمان اجرا می‌کند. برای این انتساب، فرض بر این است که تعداد ماشین‌ها، m و تعدادکارها، t از پیش معلوم است. واضح است که مسأله‌ی زمانبندی پردازش ناهمگون یک مسأله‌ی پیچیده است. بنابراین، بسیاری از محققان تحقیقات خود را در این زمینه انجام می‌دهند. هدف اصلی این پژوهش‌ها پیدا کردن راه‌حل بهینه و بهبود عملکرد کلی سیستم است. الگوریتم‌های اکتشافی و فرااکتشافی در یک محیط ایستا عمل کرده و باید زمان اجرا و حجم پردازش را از پیش بدانند. الگوریتم های متداولی که به این منظور استفاده می‌شوند به شرح زیر است:

توازن بار فرصت طلبانه (OLB): انتساب هر کار به ترتیب دلخواه به ماشینی با کمترین کارهای منتسب شده، بدون در نظر گرفتن زمان به اتمام رسیدن کارها بر روی آن ماشین. (OLB) سعی می‌کند تا کارهای منتسب شده به ماشین‌ها را به تعادل برساند، اما از آنجا که، زمان اجرا را به حساب نمی‌آورد، راه‌حل‌های ضعیفی را پیدا می‌نماید [۱۱]. اما از منابع به صورت متعادل استفاده می‌کند.

کمترین زمان اجرا (MET) اختصاص هر کار به ترتیب دلخواه به ماشینی که انتظار می رود کار بر روی آن سریع‌ترین سرعت اجرا را داشته باشد، بدون در نظر گرفته بار حال حاضر بر روی آن ماشین. (MET) برای پیدا کردن جفت‌های (کار- ماشین) خوب تلاش می‌کند، اما از آنجا که بار فعلی بر روی یک ماشین را در نظر نمی‌گیرد اغلب باعث عدم تعادل بار بین ماشین‌ها می‌شود.

کمترین زمان اتمام (MCT) اختصاص هر کار به ترتیب دلخواه به ماشینی که انتظار می‌رود حداقل زمان اتمام برای آن کار را دارا باشد. زمان اتمام کار j بر روی ماشین m برابر است با زمان اتمام کار j وقتی به مجموعه‌ی کارهای زمانبندی شده بر روی ماشین m افزوده می‌شود. اینیک الگوریتم اکتشافی بسیار موفق‌تر است، زیرا هم زمان‌های اجرا و هم بارهای ماشین در نظر گرفته می‌شود.

کمینه-کمینه (Min-Min) حداقل زمان اتمام هر کار زمانبندی نشده را محاسبه می نماید (همانند MCT)، و سپس حداقل موجود در این حداقل‌ها را به ماشینی که آن زمان را برآورده کرده است منتسب می‌کند. (Min-Min) از ایده‌ای شبیه به (MCT) استفاده می‌کند، اما از آنجا که این الگوریتم زمان اتمام حداقل برای همه‌ی کارها را در هر تکرار در نظر می‌گیرد در نتیجه می‌تواند زمان اجرای کلی را به میزان حداقلی افزایش دهد، که موجب می‌شود تا تعادل بار ماشین‌ها بهتر از (MCT) حفظ شود [۱۲].

بیشینه-کمینه (Max-min) بسیار شبیه به الگوریتم (Min-Min) است. باز هم حداقل زمان اتمام برای هر کار تولید می‌شود، اما کار با حداکثر، حداقل زمان اتمام به ماشین مربوطه انتساب داده می‌شود. (Max-min) براساس این نظر استوار است که بهتر است کارهای بزرگ‌تر زودتر زمانبندی شوند تا موجب عدم‌تعادل بار ماشین در پایان عملیات نشوند [۱۲].

الگوریتم ژنتیک (GA)، یکی از بهترین روش‌ها برای جستجوی فضاهای جواب بزرگ است. این روش بر روی جمعیتی از کروموزوم‌ها برای یک مسأله‌ی مشخص عمل می‌کند. ابتدا جمعیت اولیه را به صورت تصادفی تولید می‌نماید. جمعیت اولیه ممکن است توسط هر الگوریتم اکتشافی دیگری نیز تولید شود؛ برای مثال اگر جمعیت توسط (Min-Min) تولید شود، آن را "بذرپاشی" جمعیت با (Min-Min) می‌نامیم [۱۲، ۱۳].

جستجوی ممنوعه (TS) نیز یک روش جستجو برای فضای جواب است. این روش ناحیه‌هایی از فضای جواب را که قبلا مورد جستجو واقع شده‌اند را نگهداری می‌کند. بدین طریق جستجوی جدید در نزدیکی این مناطق تکرار نمی‌شود [۱۲].

بهینه‌سازی کلونی مورچگان (ACO) مطالعات بسیاری در مورد زمانبندی کارها با استفاده از (ACO) در محیط گرید محاسباتی وجود دارد مانند [۱۱، ۱۸]. این روش به عنوان یک روش کارآمد در حل مسائل زمانبندی در گرید محاسباتی به کار رفته است. بهینه‌سازی کلونی مورچگان دارای انواع مختلفی است که هر کدام دارای رویکردهای متفاوتی برای جستجوی فضای مساله هستند. از آنجا که این پژوهش کاربرد بهینه‌سازی کلونی مورچگان را برای مساله‌ی زمانبندی گرید محاسباتی بررسی می‌کند، الگوریتم‌های مختلف آن برای مساله فرموله و پیاده‌سازی گردید که توضیحات جامع در بخش چهارم بیان شده است.

۳- فرموله‌سازی مسأله‌ی زمانبندی دسته‌ای در گرید محاسباتی

گریدهای محاسباتی از تعداد زیادی ماشین؛ که منابع محاسباتی یا پردازشگر نیز نامیده می‌شوند، به همراه تعداد زیادی کار یا برنامه که باید بر روی ماشین‌ها اجرا شوند، تشکیل می‌شوند. ماشین‌ها قدرت محاسباتی متفاوتی دارند. کارها غیرقابل تجزیه بوده و نمی‌توان آن‌ها را به تکه‌های کوچک‌تر تقسیم نمود، همچنین نمی‌توان پس از اختصاص یک کار به یک ماشین، انجام آن را متوقف کرد. از آن‌جا که زمان اجرای هر کار در یک ماشین نسبت به ماشین دیگر متفاوت است،رقابتی در میان کارها برای استفاده از آن دسته از ماشین‌ها که قادرند آن‌ها را در کوتاه‌ترین زمان اجرا کنند وجود دارد.

به طور کلی الگوریتم‌های زمانبندی موجود را می‌توان به دو دسته تقسیم کرد: یکی حالت بر خط است و دیگری حالت دسته‌ای یا یکجا. در حالت برخط، زمانبند همیشه در حالت آماده است. هرگاه یک کار جدید به زمانبند می‌رسد، آن کار به

سرعت به یکی از ماشین‌های موجود منتسب می‌شود. در این حالت هر کار تنها یک بار برای تطبیق و زمانبندی در نظر گرفته می‌شود. در حالت دسته‌ای که موضوع این تحقیق نیز هست، کارها و ماشین‌ها قبل از زمانبندی به یکدیگر منتسب می‌شوند. در این حالت، تصمیم‌گیری بهتری انجام می‌شود، چرا که زمانبند جزئیات کاملی از کارها و ماشین‌های در دسترس را در اختیار دارد.

زمان، چالش اصلی در مسائل زمانبندی است، به گونه‌ای که یافتن راه حل در این مسائل تلاشی برای کمینه کردن زمان صرف شده برای اجرای تمام کارها خواهد بود. در این مدل، رایج‌ترین معیار برای کمینه‌کردن، زمان اجرای کل یا makespan نام دارد که عبارت است از فاصله‌ی زمانی بین شروع اولین کار تا تکمیل اجرای آخرین کار [۹]. فرمول پیش‌رو، بیان ریاضی مساله‌ی زمانبندی دسته‌ای در گرید محاسباتی را با هدف به کمینه کردن زمان اجرای کل نشان می‌دهد:

۱. گرید محاسباتی تشکیل شده از یک مجموعه از ماشین‌ها $P=\{m_1, m_2, \ldots, m_M\}$، و یک مجموعه از کارها $T=\{t_1, t_2, \ldots, t_N\}$ که باید بر روی سیستم اجرا شود.

۲. یک ماتریس زمان تخمینی اجرا ETC با ابعاد T×P وجود دارد، که عنصر $ETC[t_i][m_j]$ نشان دهنده‌ی زمان مورد نیاز برای اجرای کار t_i بر روی ماشین m_j است.

۳. یک سطر از ماتریس ETC شامل زمان اجرای تخمینی یک کار بر روی هر ماشین می‌باشد. به صورت مشابه یک ستون از ماتریس ETC شامل تخمینی از زمان اجرای یک ماشین برای هر کار است.

فرض کنید که $E_{ij\ (i \in T, j \in P)}$ زمان اجرای کار iام بر روی ماشین jام است و $W_{j\ (j \in p)}$ کارهای قبلی منتسب شده به m_j می‌باشد، آنگاه معادله‌ی (۱) زمان مورد نیاز برای m_j جهت به اتمام رساندن کارهای منتسب به آن را نشان می‌دهد. بنابر تعریف ذکر شده، زمان اجرای کل می‌تواند با استفاده از معادله‌ی (۲) تعیین شود:

$$\sum_{\forall \text{ task i allocated to machine j}} (E_{ij} + W_j) \qquad (۱)$$

$$\text{makespan} = \max \{ \sum_{\forall \text{ task i allocated to machine j}} (E_{ij} + W_j) \}, i \in T \text{ and } j \in P \qquad (۲)$$

از منظری دیگر، می‌توان زمان اجرای کل را بر حسب زمان تکمیل کار iام بر روی ماشین jام تعریف کرد که بنابر فرمول (۳) بیان می‌شود:

$$CT_{ij} = E_{ij} + W_j \qquad (۳)$$

اکنون زمان اجرای کل برابر خواهد بود با بیشنه‌ی زمان تکمیل کارها بر روی ماشین‌ها، در یک لیست زمانبندی:

$$\text{makespan} = max_{\ (i,j) \in SchedulingList} (CT_{ij}) \qquad (۴)$$

زمان اجرای کل معیاری برای اندازه‌گیری توان سیستم گرید محاسباتی است و هدف اصلی یک زمانبند، کمینه کردن زمان اجرای کل است.

۴- مساله‌ی زمانبندی در گرید محاسباتی و بهینه‌سازی کلونی مورچگان

بهینه‌سازی کلونی مورچگان از رفتار طبیعی مورچگان هنگامی که برای یافتن منبع غذایی کوتاه‌ترین مسافت را می‌یابند، الهام گرفته شده است [۸]. رفتار طبیعی مورچگان برای یافتن کوتاه‌ترین مسافت، از طریق ترشح فرومن (جوهر مورچه) در مسیر رفت و برگشت بوده و مسیری با مسافت کمینه، فرومن بیشتری خواهد داشت. در شبیه‌سازی این رفتار طبیعی برای مساله‌ی زمانبندی در گرید محاسباتی، هر مورچه برای انتساب کار iام به ماشین jام، از طریق دو مولفه راهنمایی می‌شود [۱۹]:

- اطلاعات اکتشافی: میزان اولویت یک انتساب را تعیین کرده و در طول اجرای الگوریتم مقدار آن ثابت است.
- دنباله‌ی فرومنی: میزان اولویت به دست آمده برای یک انتساب از ابتدای اجرا تا کنون را مشخص کرده و مقدار آن در طول اجرای الگوریتم توسط مورچگان در حال تغییر است.

الگوریتم (۱) شمای کلی بهینه‌سازی کلونی مورچگان را نشان می‌دهد. در فاز ساخت راه‌حل (سطر ۳)، هر مورچه براساس قانون گذر یا انتقال، سعی در ساختن یک راه‌حل برای مساله خواهد داشت. هر مورچه در حافظه‌ی خود راه‌حل ساخته شده را نگه‌داری می‌کند (در مساله‌ی زمانبندی گرید محاسباتی، به این راه‌حل، لیست زمانبندی می‌گوییم). سپس دنباله‌ی فرومنی باید به‌هنگام شود (سطر ۶). این به‌هنگام‌سازی می‌تواند بعد از هر حرکت مورچه (انتساب یک کار به یک ماشین) یا در پایان ساخت یک راه‌حل توسط مورچه (مجموعه‌ای از انتساب‌ها که به یک راه‌حل منتهی می‌شود) انجام شود. در واقع این عملیات به تقویت دنباله‌ی فرومنی خواهد انجامید. برای جلوگیری از حالت رکود، عملیاتی به نام تبخیر یا فراموشی نیز توسط مورچگان انجام می‌شود، به این ترتیب، انتساب‌های بد را با کاهش مقدار فرومن می‌توان از حافظه‌ی مورچگان پاک کرد.

الگوریتم (۱): بهینه‌سازی کلونی مورچگان

```
1: parameter initialization
2: while ( stopping criterion not satisfied ) do
3:    construct solutions
4:    local search    %optional
5:    update statistics
6:    update pheromone trails
7: end while
```

تا کنون انواع مختلفی از بهینه‌سازی کلونی مورچگان توسط محققان ارائه شده‌اند، به عنوان مثال، سیستم کلونی مورچگان (ACS) [۱۰]، سیستم مورچگان بیشینه-کمینه (MMAS) [۱۵]، سیستم مورچگان براساس رتبه (RAS) [۱۶]، سیستم مورچگان سریع (FANT) [۱۷] و سیستم مورچگان نخبه‌گرا (EAS) [۱۸]. هر کدام از این روش‌ها دارای رویکردهای متفاوتی به‌منظور به‌هنگام‌سازی دنباله‌ی فرومنی، ساخت راه‌حل و تبخیر دارند که آنها را از هم متمایز ساخته است. در این میان، سیستم مورچگان بیشینه-کمینه و سیستم کلونی مورچگان، برای مسائل مختلف رفتار بهتر از خود نشان داده‌اند. یک چالش در این تحقیق، یافتن بهترین نوع این الگوریتم برای مساله‌ی زمانبندی گرید محاسباتی است. در این راستا، پیش تحقیقی انجام گرفت تا از میان چهار نمونه از الگوریتم‌های بهینه‌سازی کلونی مورچگان، الگوریتمی که بهترین رفتار را برای زمانبندی در گرید محاسباتی دارد شناسایی شده، سپس با بهبود رفتار این الگوریتم، ضمن کاهش زمان اجرای کل، سعی در متوازن کردن بار ماشین‌ها نیز داشته باشیم.

چهار الگوریتم مورد مطالعه عبارتند از: سیستم مورچگان، سیستم مورچگان نخبه‌گرا، سیستم مورچگان بیشینه-کمینه و سیستم کلونی مورچگان. گر چه رقابت اصلی بین دو الگوریتم اخیر است، مطالعه‌ی دو الگوریتم اول به‌منظور ایجاد خط پایه در جهت مقایسه انجام شده است. جدول (۱) نتایج این مقایسه را نشان می‌دهد.

۴-۱- سیستم مورچگان (AS)

اولین الگوریتم مورچگان کـه توسـط مـارکو دوریگـو و همکـاران [۲۱] ارائـه شـد، سیستم مورچگان نام گرفت. این الگـوریتم دارای دو فـاز اصـلی سـاخت راه‌حـل و به‌هنگام‌سازی فرومن است که در ادامه تشریح شده است.

در نخستین فاز، هر مورچه پـس از آنکـه انتسـاب نخسـت خـود را بـه‌صـورت تصادفی انجام داد، بقیه‌ی انتساب‌های خود بـه منظـور سـاخت یـک راه‌حـل را بـه وسیله‌ی قانون انتساب تصادفی انجام می‌دهد.

$$p_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in allowed_k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} \text{ if } j \in allowed_k \quad (۵)$$

به‌طوری که:

- η_{ij} میزان جـذابیت یـک انتسـاب را براسـاس اطلاعـات اکتشـافی نشـان می‌دهد. کارهای پیشین در زمینه‌ی گرید محاسباتی، این پارامتر را مطـابق بـا روابط زیر ارائه کرده‌اند [۱۱ و ۱۸]:

$$\eta_{ij} = \frac{1}{W_j} \quad (۶)$$

و

$$\eta_{ij} = \frac{1}{CT_{ij}} \quad (۷)$$

- τ_{ij} دنباله‌ی فرومنی را برای یک انتساب نشان می‌دهد. در واقع یـک پسـا عامل برای میزان جذابیت یک انتساب است، چرا کـه اطلاعـات گذشـته بـرای انتساب بعدی را در بر دارد.
- α و β دو پارامتر هستند که به‌ترتیب میزان تاثیر نسبی دنباله‌ی فرومنـی و اطلاعات اکتشافی را تعیین می‌کنند.
- $allowed_k$ مجموعه‌ی انتساب‌هایی است که مورچه kام می‌توانـد انجـام دهد (تاکنون انجام نداده است).

بعد از آن‌که تمامی مورچه‌ها راه‌حل‌هـای خـود را سـاختند، دنبالـه‌ی فرومنـی به‌هنگام خواهد شد. به‌هنگام‌سازی شامل دو فاز است: تبخیر یا فراموشـی و انتشـار فرومن. در فاز تبخیر، دنباله‌ی فرومنی مربوط به راه‌حل ساخته شده توسط تمـامی مورچگان به اندازه ثابت نرخ تبخیر (ρ) کاسته می‌شود (رابطه ۸).

در فاز انتشار، دنباله‌ی فرومنی مربوط به راه‌حل سـاخته شـده توسـط تمـامی مورچگان، بسته به کیفیت راه‌حل (زمان اجرای کل)، افزایش مـی‌یابـد (روابـط ۹ و ۱۰).

$$\tau_{ij} = (1 - \rho)\tau_{ij} \quad (۸)$$

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad \forall(i,j) \in SchedulingList \quad (۹)$$

به طوری که:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{makespan^{k}} & \text{if } (i,j) \text{ belong to } SchedulingList^{k} \\ 0, & \text{else} \end{cases} \quad (۱۰)$$

۴-۲- سیستم مورچگان نخبه‌گرا (EAS)

نخبه‌گرایی، تقویت بیشتر بهترین راه‌حل‌های ساخته شده توسط مورچگان است. این ایده، اولین بهبود روی سیستم مورچگان بود [۸ و ۱۸]. بنابراین در فاز انتشار، مقدار بیشتری فرومن که بسته به کیفیت راه‌حل ساخته شده توسط بهترین مورچه و ضریب نخبه‌گرایی (e) است، به مسیر انتساب (جفت کار- ماشین) بهترین مورچه افزوده خواهد شد (روابط ۱۱ و ۱۲).

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{best} \quad \forall(i,j) \in SchedulingList^{best} \quad (۱۱)$$

به طوری که:

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{makespan^{best}} & \text{if } (i,j) \text{ belong to } SchedulingList^{best} \\ 0, & \text{else} \end{cases} \quad (۱۲)$$

۴-۳- سیستم مورچگان بیشینه-کمینه (MMAS)

سیستم مورچگان بیشینه-کمینه [۱۵] با سیستم مورچگان دارای دو تفاوت عمده است؛ نخست، در فاز انتشار فقط بهترین مورچه می‌توانـد افـزایش فـرومن داشـته باشد، سپس، انتشار فرومن دارای محدودیت بیشینه و کمینـه اسـت (روابـط ۱۳ و ۱۴).

$$\tau_{ij} = [(1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{best}]_{\tau_{min}}^{\tau_{max}} \quad \forall(i,j) \in SchedulingList \quad (۱۳)$$

به طوری که:

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{makspan^{best}} & if\ (i,j)\ belong\ to\ SchedulingList^{best} \\ 0, & else \end{cases} \quad (۱۴)$$

$\boldsymbol{\tau_{min}}$ و $\boldsymbol{\tau_{max}}$ به‌ترتیب حد بالا و حـد پـایین فـرومن را نشـان مـی‌دهنـد و از طریق رابطه (۱۵) اعمال می‌شوند:

$$[eq.(13)]_{y}^{x} = \begin{cases} x & \text{if eq.}(13) > x \\ y & \text{if eq.}(13) < y \\ eq.(13) & \text{else} \end{cases} \quad (۱۵)$$

۴-۴- سیستم کلونی مورچگان (ACS)

سیستم کلونی مورچگان [۱۰]، از سه جنبه با سیسـتم مورچگـان متفـاوت اسـت. نخست (قانون انتساب شبه تصادفی)، جستجویی قدرتمندتر از سیسـتم مورچگـان است که با اعمال یک قانون انتخاب پر تکاپو میسر شده است. دوم (نخبـه‌گرایـی)، فازهای تبخیر و انتشار فقط برای راه‌حل ساخته شده توسط بهترین مورچه اعمـال می‌گردد. سوم (به‌هنگام سازی محلی فرومن)، هنگامی که مورچه‌ای انتساب انجـام دهد، از مسیر آن انتساب مقداری فرومن کاسته خواهد شـد تـا احتمـال جسـتجو پیرامون دیگر انتساب‌ها افزایش یابد. فازهای مختلف این الگـوریتم بـه‌صـورت زیـر فرموله می‌شود. ساخت راه‌حل در این الگوریتم توسط قانون انتساب شـبه تصـادفی است (رابطه‌ی ۱۶):

$$P_{ij}^{k}=\begin{cases}\text{argmax}_{l\in \text{allowed}_k}\left\{[\tau_{ij}][\eta_{ij}]^{\beta}\right\} & \text{if } q<q_0\\ \text{Eq. (3)} & \text{else}\end{cases} \qquad (۱۶)$$

در رابطه‌ی (۱۶) q یک متغیر تصادفی در بازه [۰و۱] بوده و $0\le q_0\le 1$ یک پارامتر است.

بعد از آن‌که تمامی مورچگان راه‌حل خود را ساختند، فقط بهترین مورچه خواهد توانست به‌هنگام‌سازی فرمون را انجام دهد (به‌هنگام سازی سراسری).

$$\tau_{ij}=(1-\rho)\tau_{ij}+\rho\,\Delta\tau_{ij}^{\text{best}} \qquad \forall(i,j)\in \text{SchedulingList}^{\text{best}} \qquad (۱۷)$$

به طوری که

$$\Delta\tau_{ij}^{\text{best}}=\frac{1}{\text{makspan}^{\text{best}}} \qquad (۱۸)$$

در سیستم کلونی مورچگان، پس از هر انتسابی که مورچه‌ای انجام می‌دهد، مقدار فرومن در آن مسیر کمی کاهش می‌یابد تا بقیه‌ی مورچه‌ها مسیرهای دیگر (انتساب‌های ممکن دیگر) را نیز جستجو کنند. این ایده که به‌هنگام‌سازی محلی نام دارد، از حالت رکود جلوگیری می‌کند.

$$\tau_{ij}=(1-\xi)\tau_{ij}+\xi\tau_0 \qquad (۱۹)$$

که $0<\xi<1$ و τ_0 که مقدار اولیه‌ی فرومن است، دو پارامتر هستند.

نتایج این پیش مطالعه در جدول (۱) نشان داده شده است. خانه‌هایی از جدول که به رنگ خاکستری هستند، بهترین نتایج به‌دست آمده بعد از پیاده‌سازی را نشان می‌دهند. همان‌گونه که قابل مشاهده است، سیستم کلونی مورچگان توانسته است بر رقیب اصلی خود که سیستم مورچگان بیشینه-کمینه است، در تمامی موارد مورد مطالعه چیره شود. تحلیل این رفتار در بخش ششم قابل مشاهده است.

جدول ۱- مقایسه چهار نوع الگوریتم مورچگان براساس زمان اجرای کل

Instance	AS	EAS	MMAS	ACS
u_c_hihi.0	8624432.0	8495633.0	8456773.0	**8354392.5**
u_c_hilo.0	169761.2	162595.5	161695.1	**160921.3**
u_c_lohi.0	287122.7	275554.1	272233.2	**270934.3**
u_c_lolo.0	5561.1	5498.7	5435.0	**5411.2**
u_i_hihi.0	3721896.4	3516375.6	3510673.1	**3498233.2**
u_i_hilo.0	81233.2	80963.5	80698.0	**79651.9**
u_i_lohi.0	122388.9	121031.2	120233.5	**118927.3**
u_i_lolo.0	2798.4	2792.1	2767.6	**2691.5**
u_s_hihi.0	5201945.0	5192391.5	5076129.0	**4955398.5**
u_s_hilo.0	104893.1	104723.8	104083.5	**102195.2**
u_s_lohi.0	144012.8	141013.2	139451.0	**135275.3**
u_s_lolo.0	3881.6	3810.1	3715.4	**3691.5**

۵- الگوریتم پیشنهادی

با توجه به پیش مطالعه‌ی صورت‌گرفته در بخش چهارم، مشخص گردید که سیستم کلونی مورچگان در میان انواع دیگر الگوریتم‌های مورچگان، بهترین رفتار را برای مساله‌ی زمانبندی در گرید محاسباتی از خود نشان می‌دهد. سیستم کلونی مورچگان برای کاهش زمان محاسبه‌ی انتخاب مسیر ها از قانون انتساب شبه تصادفی به‌منظور جایگزینی روش انتقال وضعیت استفادهمی‌کند، این روش همچنین تنها فرومن مسیر مطلوب یا بهینه را بروزرسانی می‌نماید که این امر به مورچه‌ها برای جستجوی مسیر بهینه کمک می‌کند.

بنابراین، در این قسمت سعی خواهد شد الگوریتم ACS را به‌گونه‌ای بهبود دهیم که هم بتواند زمان اجرای کل را کاهش دهد، و هم میزان بار منابع محاسباتی را تا حد مطلوبی متوازن نماید. قسمت‌های پیش‌رو مطالعه‌ی انجام گرفته در این راستا را بیان می‌کنند.

۵-۱- اطلاعات اکتشافی

برخی از پژوهش‌ها، اطلاعات اکتشافی را با این ایده که اگر ماشین m_j زودتر آماده باشد (بار محاسباتی کمتری داشته باشد)، بنابراین این ماشین کارآمدتر است، به کار می‌برند [۱۱]. ایراد این روش، نادیده گرفتن زمان اجرای کار بر روی ماشینی با بار محاسباتی کمتر است. برخی دیگر از زمان تکمیل یک کار بر روی یک ماشین بعنوان اطلاعات اکتشافی استفاده کرده و انتسابی را جذاب‌تر دانسته‌اند که زمان تکمیل کار بر روی ماشین موردنظر، کمترین باشد [۱۸]. اطلاعات اکتشافی پیشنهادی در رابطه‌ی (۲۰) آورده شده است:

$$\eta_{ij}=\frac{1}{W_j\times \text{ETC}_{ij}} \qquad (۲۰)$$

در این حالت، به‌منظور انتساب کار iام به ماشین jام، اگر زمان اجرای کار iام بر روی ماشین jام از بقیه کمتر باشد و همچنین ماشین jام بار محاسباتی کمتری داشته باشد، آنگاه جذابیت این انتساب از بقیه انتساب‌های ممکن بیشتر خواهد بود. به منظور پی‌بردن به کارایی این روش و مقایسه با روش‌های پیشین، هرکدام از این سه روش در سیستم کلونی مورچگان آزمایش شده و نتایج حاصل، از برتری کامل اطلاعات اکتشافی پیشنهادی حکایت دارد. نتایج در جدول (۳) قابل مشاهده است.

۵-۲- اطلاعات کارآمدی

با توجه به اینکه در این مقاله هدف کمینه کردن زمان اجرای کل و تعادل بار در ماشین‌ها است، اطلاعات جدیدی به نام اطلاعات کارآمدی در رابطه (۲۱) معرفی شده است:

$$\omega_{ij}=\frac{1}{\text{CT}_{ij}} \qquad (۲۱)$$

در این رابطه CT_{ij} زمان تکمیل iامین کار بر روی jامین ماشین است. این رابطه به این معنی است که اگر زمان تکمیل iامین کار بر روی jامین ماشین کمتر باشد، آنگاه کارآمدی ماشین jام از بقیه ماشین‌های موجود بیشتر خواهد بود. بنابراین، قانون انتساب شبه تصادفی مطابق با روابط (۲۲) و (۲۳) تغییر می‌کند:

$$P_{ij}^{k}=\begin{cases}\text{argmax}_{l\in \text{allowed}_k}\left\{[\tau_{ij}][\eta_{ij}]^{\beta}[\omega_{ij}]^{\gamma}\right\} & \text{if } q<q_0\\ \text{Eq. (9)} & \text{else}\end{cases} \qquad (۲۲)$$

$$P_{ij}^{k}=\frac{[\tau_{ij}][\eta_{ij}]^{\beta}[\omega_{ij}]^{\gamma}}{\sum_{l\in \text{allowed}_k}[\tau_{il}][\eta_{il}]^{\beta}[\omega_{il}]^{\gamma}} \qquad \text{if } j\in \text{allowed}_k \qquad (۲۳)$$

دو رویکرد اخیر با در نظر گرفتن اطلاعات ETC_{ij} و CT_{ij} سعی در حفظ تعادل بار در ماشین‌ها و کمینه‌کردن زمان اجرای کل دارند.

۵-۳- به‌هنگام سازی محلی فرومن

بعد از آن‌که هر مورچه توانست یک انتساب انجام دهد (جفت کار- ماشین را بسازد)، بهنگام‌سازی محلی مطابق با رابطه (۲۴) انجام خواهد شد:

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \tau_0 \times \omega_{ij} \qquad (۲۴)$$

بنابراین، افزایش فرومن در بهنگام‌سازی محلی به جای آن‌که ثابت باشد، متغیر و وابسته به زمان تکمیل کار iام بر روی ماشین jام (CT_{ij}) خواهد بود، به این معنی که اگر CT_{ij} کمتر باشد، انتساب کار i به ماشین j کیفیت مطلوبی داشته و به میزان بیشتری تقویت می‌شود.

۵-۴- به‌هنگام سازی سراسری فرومن

ایده‌ی بهنگام‌سازی سراسری، تقویت بیشتر جفت (کار، ماشین) متعلق به بهترین لیست زمانبندی یافته شده از زمان شروع الگوریتم تاکنون است. با این وجود، بهنگام سازی سراسری مطابق با رابطه زیر تغییر می‌کند:

$$\tau_{ij} = \tau_{ij} + e\,\Delta\tau_{ij}^{gbest} \qquad \forall (i,j) \in \text{SchedulingList}^{best} \qquad (۲۵)$$

به‌طوری‌که

$$\Delta\tau_{ij}^{gbest} = \begin{cases} \dfrac{1}{makspan^{gbest}} & if\ (i,j) \in SchedulingList^{gbest} \\ 0, & else \end{cases} \qquad (۲۶)$$

e پارامتری است که وزن را با توجه به بهترین لیست زمانبندی تاکنون (best-so-far) و زمان اجرای کل آن makspan^{gbest} تعیین می‌کند و ضریب نخبه پروری نامیده می‌شود.

الگوریتم (۲) شبه کد الگوریتم پیشنهادی برای مساله زمانبندی در گریدهای محاسباتی را نشان می‌دهد.

الگوریتم (۲): سیستم کلونی مورچگان پیشنهادی

```
parameter initialization
  m, ρ, β, τ0, q0, ξ, γ, e
while ( stopping criterion not satisfied ) do
  construct solutions phase
    for all ants do
      place ants in random chosen machine
      apply local update for (task, machine) pair Eq.(24)
    end for
    while ( all task not assigned ) do
      apply ACS pseudorandom proportional rule Eq.(22)
      apply local update for (task, machine) pair Eq.(24)
    end while
  end of construct solution phase
  update statistics
  update iteration-best and best-so-far
  evaporate
  apply global pheromone updateEq.(25)
end while
```

۶- نتایج عملی

این بخش به معرفی مدل شبیه‌سازی، نمونه‌های مساله زمانبندی در گریدهای محاسباتی و بستر توسعه به‌منظور ارزیابی نتایج الگوریتم پیشنهادی می‌پردازد.

۶-۱- مدل شبیه‌سازی

سیستم‌های واقعی محاسبات ناهمگون، مانند گرید محاسباتی، ترکیب پیچیده‌ای از اجزای سخت‌افزاری، نرم‌افزاری و شبکه‌ای بوده و اغلب مقایسه منصفانه بین تکنیک‌های مختلف که بر روی سیستم‌های متفاوت استفاده می‌شوند سخت می‌گردد.

برای حل این مشکل براون و همکاران [۱۲] یک مدل شبیه‌سازی برای مقایسه الگوریتم زمانبندی ایستا برای محیط‌های محاسبات ناهمگون ارائه کرده‌اند. آنها مفهوم فراکار را به‌عنوان یک مجموعه از کارهای مستقل و بدون هیچ وابستگی تعریف کرده، که هدف از یک الگوریتم زمانبندی در اینجا به حداقل رساندن زمان اجرای کل فراکار می‌باشد. از آنجا که زمانبندی به صورت ایستا انجام می‌گیرد فرض بر آن است که به تمام اطلاعات لازم در مورد کارها موجود در فراکار و ماشین‌های موجود در سیستم از پیش دسترسی وجود دارد. از آنجا که زمان اجرای مورد انتظار هر کار برروی هر ماشین باید شناخته شده باشد، این اطلاعات در ماتریس زمان تخمینی اجرا (ETC) ذخیره می‌شوند. البته، در یک زمانبند واقعی، ممکن است برخی از تفاوت‌ها بین زمان تخمینی اجرا و زمان واقعی انجام کار وجود داشته باشد، اما در این مقاله ما فرض می‌کنیم که مقدار موجود در ماتریس (ETC) زمان اجرای کار می‌باشد.

براون و همکاران [۱۲] مسائل زمانبندی در محیط‌های محاسباتی ناهمگون را تا حد امکان به صورت واقع‌گرایانه در نظر گرفته و انواع مختلف ماتریس ETC را با توجه به سه معیار تعریف کردند: ناهمگونی کار، ناهمگونی ماشین و سازگاری. ناهمگونی کار به عنوان مقدار واریانس (تغییرات) ممکن در میان زمان اجرای کارها تعریف می‌شود، به این منظور دو مقدار ممکن تعریف شده است: قوی و ضعیف. از سوی دیگر، ناهمگونی ماشین، نشان دهنده تغییرات احتمالی زمان اجرای یک کار خاص بر روی تمام ماشین‌ها است، و دوباره دارای دو مقدار: قوی و ضعیف است. به منظور تلاش برای درنظر گرفتن برخی از ویژگی‌های دیگر مسائل زمانبندی واقعی، سه نوع سازگاری ETC مختلف مورد استفاده قرار گرفت که عبارتند از: سازگار، ناسازگار و نیم‌سازگار. یک ماتریس ETC سازگار گفته می‌شود اگر هرگاه که ماشین m_j وظیفه‌ی j_i را سریعتر از ماشین m_k اجرا نماید، آنگاه ماشین m_j تمامی وظایف دیگر را سریع تر از m_k اجرا می‌کند. بنابراین یک ماتریس ETC سازگار می‌تواند به عنوان مدلی از یک سیستم ناهمگون که در آن ماشین‌ها فقط در سرعت پردازش متفاوت هستند دیده شود. در یک ماتریس ETC ناسازگار ماشین p_j ممکن است برخی از وظایف سریع ترزا از p_k سریعتر و برخی را کندتر اجرا کند. بنابراین ماتریس ناسازگار ETC می‌تواند یک شبکه را که در آن انواع مختلفی از ماشین‌ها در دسترس هستند شبیه‌سازی نماید، به عنوان مثال، یک ماشین لینوکس ممکن است وظایفی را که شامل مقدار زیادی از محاسبات نمادین می‌باشد سریع‌تر از یک ماشین ویندوز اجرا کند، اما وظایفی را که شامل تعداد زیادی عملیات ممیز شناور است را کندتر انجام دهد.

یک ماتریس ETC نیم‌سازگار یک ماتریس ناسازگار است که زیر ماتریسی سازگار با اندازه از پیش تعریف شده دارد، و به‌همین ترتیب می‌تواند برای مثال، یک گرید محاسباتی که شامل شبکه‌های جزیی از ماشین‌های لینوکس مشابه (اما با سرعت ماشین‌های مختلف) است را شبیه‌سازی نماید، اما شامل مجموعه‌ای از ماشین‌های مختلف محاسباتی نیز می‌باشد. این فرضیات مختلف ۱۲ نوع متفاوت از ماتریس ETC ممکن را در اختیار قرار می‌دهند (به عنوان مثال کار با حجم محاسباتی بالا، ناهمگونی ضعیف ماشین در یک ماتریس ناسازگار، و غیره) که طیف وسیعی از سیستم‌های ناهمگون موجود را شبیه‌سازی می‌کند.

همه‌ی این نمونه‌ها دارای ۵۱۲ کار و ۱۶ ماشین بوده و ترکیبی از سه خاصیت مدل ETC هستند (ناهمگونی کار و ماشین و سازگاری) تا سناریوهای مختلفی از مساله را مدل کنند. مجموعه نمونه‌های ارایه شده توسط براون و همکاران، یک معیار استاندارد عملی برای ارزیابی الگوریتم‌هایی است که برای حل مساله زمانبندی در محیط‌های پردازش ناهمگون به‌کار می‌روند. در مطالعاتی که توسط براون و همکاران [۱۲] انجام شد، این ماتریس‌ها به‌صورت تصادفی و با محدودیت‌های خاصی تولید شدند تا خواص ماتریس‌های توصیف شده در بالا را شبیه‌سازی کنند. روش تولید این ماتریس‌ها به اختصار در ادامه بیان شده‌اند. در ابتدا، یک بردار $m\times1$ که بردار پایه (B) نامیده می‌شود، به‌صورت تصادفی و یکنواخت در بازه‌ی اعشاری $[1, \Phi_b]$ تولید می‌شود. سپس ماتریس ETC از طریق ضرب B[i] با عدد تصادفی و یکنواخت $X_r^{i,k}$ که دارای حد بالای Φ_r است، تولید می‌شود. $X_r^{i,k}$ با نام مضرب سطری شناخته می‌شود. بنابراین هر ردیف از ماتریس ETC با فرمول زیر محاسبه می‌شود:

$$ETC\,[j_i, p_k] = B[i] \times X_r^{i,k} \quad , 0 \le k \le n \qquad (۲۷)$$

این فرآیند برای تمام ردیف‌ها تکرار می‌شود تا آن‌که سرانجام ماتریس $m\times n$ پر شود. هر کدام از ناهمگونی‌های کاری و ماشینی که قبلا توصیف شدند، با استفاده از مقادیر پایه‌ی (B) مختلف مدل می‌شوند: $\Phi_b = 3000$ برای ناهمگونی کاری قوی و $\Phi_b = 100$ برای ناهمگونی کاری ضعیف. $\Phi_r = 1000$ برای ناهمگونی ماشینی قوی و $\Phi_b = 10$ برای ناهمگونی ماشینی ضعیف. برای مدل‌سازی سازگاری، هر ردیف از ماتریس به‌صورت جداگانه مرتب می‌شود؛ به‌گونه‌ای که ماشین p_1 همواره سریع‌ترین و ماشین p_m همواره کندترین خواهد بود. برای مدل‌سازی ناسازگاری، ماتریس‌ها مرتب نشده و در حالت اصلی و تصادفی خود باقی می‌مانند. برای مدل کردن حالت نیم‌سازگار، ردیف‌های زوج مرتب می‌شوند (سازگار) و ردیف‌های فرد به حالت تصادفی و اصلی خود باقی می‌مانند (ناسازگار).

الگوی نامی که براون و همکاران [۱۲] برای نمونه‌های مساله‌ی زمانبندی در محیط‌های پردازش ناهمگون ارایه کردند به صورت d_c_MHTH.0 است، به‌طوری‌که d تابع توزیع استفاده شده برای تولید مقادیر ETC است (u به معنی توزیع یکنواخت است) و c نوع سازگاری را مشخص می‌کند (c برای سازگار، i برای ناسازگار و s برای نیم‌سازگار). MH و TH به‌ترتیب درجه‌ی ناهمگونی کارها و ماشین‌ها را مشخص می‌کند (lo برای ناهمگونی ضعیف و hi برای ناهمگونی قوی). عدد آخر که بعد از نقطه آمده (0) موارد تست شده را مشخص می‌کند (در ابتدا، مجموعه‌های زیادی توسط براون و همکاران تولید شد که در نهایت فقط کلاس 0 عمومیت یافت).

۶-۲- بستر توسعه و اجرا

الگوریتم پیشنهادی به‌صورت ترتیبی به زبان c و با استفاده از کتابخانه‌های استاندارد آن بر روی یک رایانه‌ی $core^{i3}$ با سیستم عامل لینوکس ابونتو پیاده‌سازی و تست شده است.

۶-۳- نمایش راه‌حل

لیست زمانبندی ساخته شده توسط هر مورچه به صورت یک بردار به اندازه‌ی تعداد کارها است که هر اندیس بردار یک کار را نشان داده و هر عنصر این بردار ماشینی است که کار منتسب شده را اجرا خواهد کرد. شکل (۱) این نمایش را نشان می‌دهد.

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	...	...	t_i	...	t_k	...
m_1	m_3	m_1	m_0	m_4	m_8	m_2	m_7	...	...	m_j	...	m_k	...

شکل ۱- نمایش لیست زمانبندی

۶-۴- تنظیم پارامترها

با توجه به خواص مدل مساله (ناهمگونی و پایداری و ...)، عملیات تنظیم پارامترها با استفاده از سه نمونه مساله با خواص متفاوت، صورت گرفته است (u_s_lohi.0،u_i_hilo.0،u_c_lolo.0). پارامترهای مورد مطالعه عبارتند از تعداد مورچه‌ها (m)، نرخ تبخیر (ρ)، فرومن اولیه (τ_0)، تاثیر نسبی اطلاعات اکتشافی (β)، پارامتر (q_0)، و نرخ نخبه‌پروری (e). بهترین پارامترها در جدول (۲) نمایش داده شده‌اند.

جدول ۲- تنظیم پارامترهای الگوریتم پیشنهادی

Parameters	Values
m	256
ρ	0.5
τ_0	1.0 / (task × machine)
ξ	0.1
β	2.0
γ	1.0
q_0	0.7
e	100

۶-۵- مقایسه‌ی اطلاعات اکتشافی پیشنهادی و روش‌های پیشین

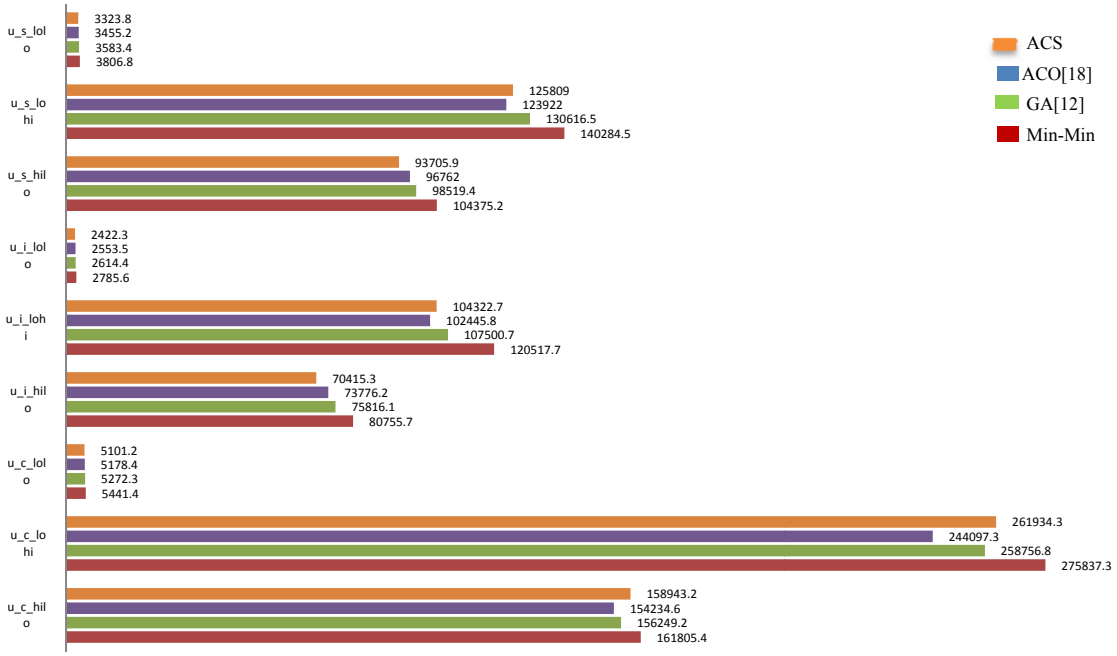
جدول (۳) نتایج مقایسه‌ی اطلاعات اکتشافی پیشنهادی با روش‌های پیشین نشان می‌دهد. مطابق با نتایج، روش پیشنهادی در تمامی موارد دو روش پیشین را شکست می‌دهد. برای مقایسه عادلانه، هر سه روش به عنوان اطلاعات اکتشافی در یک الگوریتم سیستم کلونی مورچگان ساده به‌صورت جداگانه تست شده و پس از مدت زمان ۲۰۰ ثانیه، نتایج برحسب زمان اجرای کل به دست آمده است.

جدول ۳- مقایسه اطلاعات اکتشافی (HI) پیشنهادی و روش‌های پیشین براساس زمان اجرای کل به‌دست آمده

Instance	**HI[12]**	**HI[18]**	**our HI**
	$1/ready_j$	$1/CT_{ij}$	$1/(ready_j * ET_{ij})$
u_c_hihi.0	9026851	8595233	**8201934**
u_c_hilo.0	179802	172943	**166298**
u_c_lohi.0	292366	288391	**267191**
u_c_lolo.0	5413	5339	**5298**
u_i_hihi.0	3604367	3319023	**3249103**
u_i_hilo.0	80982	79812	**78023**
u_i_lohi.0	118023	112561	**110384**
u_i_lolo.0	2805	2709	**2612**
u_s_hihi.0	5102760	4699103	**4558092**
u_s_hilo.0	102566	99071	**98141**
u_s_lohi.0	138926	131610	**129677**
u_s_lolo.0	3695	3531	**3491**

جدول ۴- مقایسه زمان اجرای کل در الگوریتم پیشنهادی و الگوریتم‌های فرااکتشافی پیشین

Instance	min-min	GA[12]	ACO+TS+LS[18]	Our ACS	Time
		12600 sec	12600 sec		
u_c_hihi.0	8460674.0	8050844.5	7497200.9	8045322.5	1500sec
u_c_hilo.0	161805.4	156249.2	154234.6	158943.2	1500sec
u_c_lohi.0	275837.3	258756.8	244097.3	261934.3	1500sec
u_c_lolo.0	5441.4	5272.3	5178.4	**5101.2**	700sec
u_i_hihi.0	3513919.3	3104762.5	2947754.1	3052630.3	1500sec
u_i_hilo.0	80755.7	75816.1	73776.2	**70415.3**	350sec
u_i_lohi.0	120517.7	107500.7	102445.8	104322.7	1500sec
u_i_lolo.0	2785.6	2614.4	2553.5	**2422.3**	320sec
u_s_hihi.0	5160343.0	4566206	4162547.9	4295853.5	1500sec
u_s_hilo.0	104375.2	98519.4	96762	**93705.9**	300sec
u_s_lohi.0	140284.5	130616.5	123922	125809.0	1500sec
u_s_lolo.0	3806.8	3583.4	3455.2	**3323.8**	150sec



شکل ۲- مقایسه زمان اجرای کل بین الگوریتم پیشنهادی و روش‌های پیشین

۶-۶- مقایسه‌ی الگوریتم پیشنهادی و دیگر الگوریتم‌های فرااکتشافی

در این قسمت، الگوریتم پیشنهادی با الگوریتم‌های پیشین نظیر min-min، الگوریتم ژنتیک (GA) [۱۲] و الگوریتم ترکیبی مورچگان و جستجوی ممنوعه و جستجوی محلی (ACO+TS+LS) [۱۸] مقایسه خواهد شد. جدول (۴) و شکل (۲) نتایج را بر حسب زمان اجرای کل نشان می‌دهد. نتایج نشان می‌دهد که برای پنج نمونه، الگوریتم پیشنهادی از دیگر الگوریتم‌ها عملکرد بهتری از خود نشان می‌دهد؛ چه از لحاظ کاهش زمان اجرای کل و چه از لحاظ زمان مصرف شده توسط الگوریتم. نکته‌ی قابل توجه این است که الگوریتم پیشنهادی از ترکیب یا جستجوی محلی [۱۸] و یا بذرپاشی [۱۲، ۱۳] استفاده نمی‌کند و بسیار ساده و روان است. بنابراین الگوریتم‌های GA و ACO+TS+LS که الگوریتم‌های پیچیده و یا ترکیبی هستند دارای زمان مصرفی ۱۲۶۰۰ ثانیه، یعنی ۳ ساعت و نیم هستند، در حالی‌که الگوریتم پیشنهادی دارای زمان مصرفی بسیار کمتری است که در جدول (۴) نشان داده شده است. ملاک توقف الگوریتم پیشنهادی رسیدن به زمان ۱۵۰۰ ثانیه و یا دستیابی به نتایج بهتر است.

۶-۷- تحلیل نتایج

در این قسمت به تحلیل نتایج به‌دست آمده توسط الگوریتم پیشنهادی می‌پردازیم. مطابق با جدول (۴)، الگوریتم پیشنهادی در مواردی که درجه‌ی ناهمگونی کارها ضعیف است، به بهترین نتایج دست‌یافته است. در اکثر موارد نیز، الگوریتم پیشنهادی، الگوریتم ژنتیک [۱۲] را شکست داده و در رده‌ی دوم قرار می‌گیرد. عواملی که در غالب شدن سیستم کلونی مورچگان پیشنهادی موثر هستند، نخست استفاده از قانون انتساب شبه تصادفی است که زمان بسیار کمی برای یک انتساب مصرف می‌کند، دیگری به‌هنگام‌سازی محلی فرومن است که از حالت رکود جلوگیری کرده و باعث می‌شود انتساب‌های دیگری هم در نظر گرفته شوند. در نهایت یکی از عوامل اصلی موفقیت روش پیشنهادی با توجه به جدول (۲)، اطلاعات اکتشافی به‌کار رفته در الگوریتم ارزیابی می‌شود.

۷- نتیجه‌گیری

این مقاله حل مساله زمانبندی در محیط‌های پردازش ناهمگون همچون گرید محاسباتی، که از مسایل بسیار مهم در سیستم‌های توزیع شده‌ی ناهمگون است را با الگوریتم سیستم کلونی مورچگان نشان می‌دهد. الگوریتم پیشنهادی طوری طراحی شده است که نتایج دقیقی را به‌صورت کارا در جهت کمینه کردن زمان اجرای کل بیابد. با استفاده از چندین نمونه مسأله که به عنوان معیار ارزیابی مطرح هستند، مشخص شد که الگوریتم پیشنهادی از دیگر روش‌های پیشین عملکرد بهتری دارد، چه از لحاظ کمینه کردن زمان اجرای کل و چه از لحاظ زمان مصرفی الگوریتم. همچنین با توجه به خاصیت توازنی افزوده شده به الگوریتم، میزان بار ماشین‌ها نیز متوازن خواهد بود. عقیده‌ی ما بر این است که با به‌کارگیری جستجوی محلی در این الگوریتم، نتایج بهتری حاصل خواهد شد. همچنین با توجه به خاصیت ذاتی موازی بودن الگوریتم مورچگان، پیاده‌سازی موازی این الگوریتم بسیار کارامد خواهد بود.

مراجع

[1] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *IEEE Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200-222, 2001.

[2] I. Foster, and C. Kesselman, *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann, 1998.

[3] H. El-Rewini, T. Lewis, and H. Ali, *Task Scheduling in Parallel and Distributed Systems*, Prentice-Hall, 1994.

[4] J. Leung, L. Kelly, and J. Anderson, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, CRC Press, 2004.

[5] M. Eshaghian, *Heterogeneous Computing*, Artech House, 1996.

[6] R. Freund, V. Sunderam, A. Gottlieb, K. Hwang, and S. Sahni, "Special issue on heterogeneous processing," *IEEE Journal of parallel and distributed computing*, vol. 21, no. 3, pp. 12-24, 1994.

[7] M. Garey, and D. Johnson, *Computers and Intractability*, Freeman, 1979.

[8] M. Dorigo, and T. Stutzle, *Ant Colony Optimization*, MIT Press, 2004.

[9] J. Leung, L. Kelly, and J. Anderson, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, CRC Press, 2004.

[10] M. Dorigo, and L. M. Gambardella, "Ant Colony System: A cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Trans. Evolutionary Computation*, vol. 1, no. 3, pp. 53-66, 1999

[11] S. Fidanova, and M. Durchova, *Ant Algorithm for Grid Scheduling Problem*, *Large Scale Computing,* Lecture Notes in Computer Science, Springer, Germany, 2006.

[12] T. D. Braun, and et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *IEEE Journal of parallel and distributed computing*, vol. 61, no. 6, pp. 810-837, 2001.

[13] L. Wang, and et al., "Task matching and scheduling in heterogeneous computing environments using a genetic-algorithm-based approach," *IEEE Journal of Parallel and Distributed Computing*, vol. 47, no.1, pp. 8-22, 1997.

[14] L. Wang, and et al., "Environments using a genetic-algorithm-based approach," *IEEE Journal of parallel and distributed computing*, vol. 47, no. 1, pp. 2-18, 1997.

[15] T. Stutzle, *MAX-MIN Ant System for Quadratic Assignment Problems*, Technical Report of Intellectics Group, Darmstadt, Germany, 1997.

[16] B. Bullnheimer, R. F. Hartl, and C. Strauss, "A new rank-based version of the ant system: A computational study," *IEEE Journal of Central European for Operations Research and Economics*, vol. 7, no. 1, pp. 25-38, 1999.

[17] E. D. Taillard, and L. M. Gambardella, *Adaptive memories for the quadratic assignment problem*, Technical Report, Lugano, Switzerland, 1997.

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 26, no. 1, pp. 29-41, 1996.

[19] G. Ritchie, and J. Levine, "A hybrid ant algorithm for scheduling independent jobs in heterogeneous computing environments," *IEEE Int'l Workshop of the UK Planning and Scheduling Special Interest Group*, pp. 178-183, 2004.

[20] R. J. Mullen, and et al., "A review of ant algorithms," *Proc, IEEE Int'l Conf. Expert Systems with Applications*, pp. 36-44, 2009.

[21] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a colony of cooperating agents," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 26, no. 1, pp. 29-41, 1996.

مصطفی صفرپور کارشناسی‌ارشد خود را در رشته مهندسی کامپیوتر از دانشگاه شیراز در سال ۱۳۹۲ دریافت کرد. پایان‌نامه وی در زمینه طراحی الگوریتم‌های فرااکتشافی موازی برای حل مسئله زمانبندی در گریدهای محاسباتی است. زمینه تحقیقات وی گریدهای محاسباتی، زمانبندی، الگوریتم‌های فرا اکتشافی، الگوریتم‌های فرااکتشافی موازی، پردازش موازی و شبکه‌های نظیر به نظیر است.

آدرس پست‌الکترونیکی ایشان عبارت است از:

safarpour.mostafa@gmail.com

غلامحسین دستغیبی‌فرد کارشناسی‌ارشد و دکترای خود را در رشته علوم کامپیوتر از دانشگاه آکلاهمای آمریکا در سال‌های ۱۹۷۹ و ۱۹۹۱ دریافت کرد. وی اکنون استادیار دانشکده مهندسی برق و کامپیوتر دانشگاه شیراز است. زمینه تحقیقات ایشان گریدهای محاسباتی، پردازش موازی، سیستم‌های توزیع شده و پردازش ابری می‌باشد.
آدرس پست‌الکترونیکی ایشان عبارت است از:
dstghaib@shirazu.ac.ir

سیدعلی میرسلیمانی مدرک کارشناسی‌ارشد در مهندسی کامپیوتر خود را از دانشگاه شیراز در سال ۱۳۹۱دریافت نمود. تحقیقات وی بر روی معماری‌های موازی و الگوریتم‌های موازی و توزیع شده متمرکز است. پایان‌نامه ارشد وی پیرامون تحلیل کارآیی پردازنده‌های گرافیکی می‌باشد. او هم‌اکنون در حال کار کردن بر روی الگوریتم‌های کاوش در فضای طراحی برای پردازنده‌های گرافیکی می‌باشد.
آدرس پست الکترونیکی ایشان عبارت است از:
ali.mirsoleimani@gmail.com

اطلاعات بررسی مقاله:

تاریخ ارسال: ۹۲/۱/۳۰
تاریخ اصلاح: ۹۲/۶/۷
تاریخ قبول شدن:۹۲/۷/۷
نویسنده مرتبط: مصطفی صفرپور، دانشکده مهندسی برق و کامپیوتر، دانشگاه شیراز، شیراز، ایران.