الگوریتم‌های زمانبندی برای بهینه‌سازی زمان در گریدهای محاسباتی اقتصادی

یاسر مهدوی‌فر و محمدرضا میبدی

آزمایشگاه سیستم‌های نرم‌افزاری، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر

E-mail: mahdavifar@gmail.com, meybodi@ce.aut.ac.ir

چکیده - کاربران در گریدهای محاسباتی اقتصادی برای اجرای برنامه‌های کاربردی خود هزینه پرداخت می‌کنند. هر کاربر، مهلت زمانی و بودجه مورد نظر خود را تعیین کرده و بهینه‌سازی هزینه یا زمان را درخواست می‌کند. یک الگوریتم زمانبندی با هدف بهینه‌سازی زمان، باید با توجه به قیمت و توانمندی منابع گرید، عمل تخصیص آنها به کارهای ناهمگون کاربر را طوری انجام دهد که اجرای کارها با صرف بودجه تعیین شده و در حداقل زمان پایان یابد. در این مقاله، سه الگوریتم مکاشفه‌ای جدید برای این منظور پیشنهاد شده است. با استفاده از شبیه‌سازی نشان داده شده است که الگوریتم‌های پیشنهادی در مقایسه با تنها الگوریتم گزارش شده از کارایی بالاتری برخوردار بوده و درخواستهای کاربر را در زمان کمتری انجام می‌دهند.

کلید واژه- گرید محاسباتی، زمانبندی اقتصادی، بهینه‌سازی زمان.

۱- مقدمه

گرید محاسباتی[۱] یک زیربنای سخت‌افزاری و نرم‌افزاری می‌باشد که دسترسی قابل‌اعتماد، پایدار، فراگیر و ارزان را به توانایی‌های محاسباتی دیگران فراهم می‌کند [2-5]. یک گرید محاسباتی با مجموعه‌ای از منابع ناهمگون[۲] (کامپیوترهای شخصی، ایستگاه‌های کاری[۳]، کلاسترها و ابرکامپیوترها)، در مقیاس وسیع در ارتباط است. گریدهای محاسباتی به تدریج به سوی تجاری شدن پیش می‌روند و دارندگان منابع با انگیزه‌های مالی، منابع خود را در اختیار دیگران قرار می‌دهند. مشتریان گرید نیز با پرداخت هزینه درخواست خود، می‌توانند از این منابع استفاده کنند. صاحبان و استفاده‌کنندگان منابع دارای اهداف، استراتژی‌ها و الگوهای عرضه و تقاضای متفاوتی هستند. در چنین شرایطی نمی‌توان از راهکارهای متداول برای مدیریت منابع که سعی می‌کنند میزان کارایی کل سیستم را بهبود دهند، استفاده کرد. برای این منظور، در سال‌های اخیر از رویکردهای اقتصادی برای مدیریت تخصیص منابع در گرید استفاده شده است [6-9]. یکی از مدل‌های اقتصادی که بدین منظور استفاده شده است، مدل بازار کالا[۴] می‌باشد. در این مدل، هر منبع دارای قیمت مشخصی است که بر اساس عرضه، تقاضا و ارزش در سیستم اقتصادی تعیین شده است.

منابع در گرید به صورت ناهمگون و توزیع شده هستند و به طور مشترک مورد استفاده قرار می‌گیرند. از طرف دیگر، کاربران، محدودیت‌های مهلت[۵] (زمان اتمام اجرای برنامه) و بودجه (هزینه محاسبات) را برای گرید تعیین می‌کنند. این مسائل، باعث پیچیدگی عمل زمانبندی برنامه کاربر می‌گردد. یکی از استراتژی‌هایی که یک الگوریتم زمانبندی می‌تواند اتخاذ کند، مینیمم کردن زمان در محدوده بودجه تعیین شده (بهینه‌سازی زمان) می‌باشد. تا آنجا که نگارندگان این مقاله مطلع می‌باشند، تاکنون تنها یک الگوریتم مکاشفه‌ای[۶] که آن را BTO[7] می‌نامیم، برای مینیمم کردن زمان در محدوده بودجه در مدل اقتصادی بازار کالا توسط بویا در [10,11] گزارش شده و در [12] مورد ارزیابی قرار گرفته است. این الگوریتم برای زمانبندی برنامه‌های کاربردی پارامتروروب[۸] که شامل تعداد زیادی

کارهای همگون مستقل از هم می‌باشند، طراحی شده است و به همین دلیل برای کارهای ناهمگون، جوابهای قابل قبولی را به کاربر ارائه نمی‌کند.

در این مقاله، ۳ الگوریتم مکاشفه‌ای جدید که آنها را ABTO[9]، EBTO[10] و AEBTO[11] می‌نامیم و بر اساس الگوریتم BTO طراحی شده‌اند، برای زمانبندی کارهای مستقل از هم با هدف بهینه‌سازی زمان در گریدهای محاسباتی اقتصادی با مدل بازار کالا پیشنهاد می‌شوند. الگوریتم‌های پیشنهادی با استفاده از جعبه‌ابزار GridSim [12] شبیه‌سازی شده و کارایی‌های آنها مورد بررسی قرار گرفته است. نتایج بدست آمده با نتایج الگوریتم بویا مقایسه شده است. ادامه مقاله بدین صورت سازماندهی شده است. در بخش ۲ زمانبندی اقتصادی در گرید شرح داده می‌شود. در بخش ۳ الگوریتم‌های زمانبندی پیشنهادی ارائه می‌گردد. بخش ۴ اختصاص به نتایج شبیه‌سازی‌ها دارد و بخش ۵ نتیجه‌گیری می‌باشد.

۲- زمانبندی اقتصادی در گرید

طول یک کار، تعداد دستورالعمل‌های آن می‌باشد و بر حسب واحد MI[12] (میلیون دستورالعمل) اندازه‌گیری می‌شود. اگر کارهای یک برنامه کاربردی، کاملاً همگون[۱۳] باشند (طول یکسان داشته باشند)، می‌توان الگوریتم‌های زمانبندی با پیچیدگی چندجمله‌ای برای مینیمم کردن زمان در محدوده بودجه، طراحی کرد؛ ولی هنگامی که کارها دارای ناهمگونی باشند (طول‌های متفاوت داشته باشند)، مسأله زمانبندی با هدف بهینه‌سازی زمان به یک مسأله NP-Complete تبدیل می‌شود.

قیمت اعلام شده برای یک منبع، مقدار هزینه استفاده از آن در واحد زمان می‌باشد و قیمت مفید یک منبع به مقدار هزینه استفاده از آن برای اجرای هر MI گفته می‌شود. در این مقاله، واژه‌های قیمت، ارزانی و گرانی با توجه به قیمت مفید منبع به کار می‌روند.

کاربر گرید پس از اینکه محدودیت‌های مهلت زمانی و بودجه خود را مشخص کرد، ممکن است بهینه‌سازی هزینه و یا زمان را درخواست کند. در صورتی که کاربر خواستار بهینه‌سازی زمان باشد، الگوریتم زمانبندی باید با صرف بودجه‌ای که کاربر تعیین کرده است، کمترین زمان را برای اجرای کارها بدست آورد و در صورتی که کاربر خواستار بهینه‌سازی هزینه باشد، الگوریتم زمانبندی باید در محدوده مهلتی که کاربر تعیین کرده است، کمترین هزینه را برای اجرای کارها بدست آورد.

اجرای یک برنامه کاربردی بر روی گرید اقتصادی با صرف بودجه‌ای که توسط کاربر فراهم شده است، دارای یک کران پایین یا مینیمم برای زمان محاسبات می‌باشد. البته در بیشتر موارد، هیچ سیستم زمانبندی وجود ندارد که بتواند کارها را در مدت زمان مینیمم به کاربر تحویل دهد و الگوریتم‌های زمانبندی سعی می‌کنند تا حد ممکن زمان محاسبات را به زمان مینیمم نزدیک کنند. بهترین روش برای بدست آوردن مینیمم زمان محاسبات این است که از روش ارائه شده برای بدست آوردن مینیمم هزینه استفاده کنیم [۱]. برای این کار، روش مینیمم هزینه را با در نظر گرفتن مهلت، اجرا می‌کنیم و هزینه بدست آمده را بررسی می‌کنیم. سپس مهلت را مقداری کاهش داده و دوباره روش مینیمم هزینه را اجرا می‌کنیم. این عمل آنقدر تکرار می‌گردد تا به مهلتی برسیم که مینیمم هزینه انجام کارها در آن مهلت، با بودجه تعیین شده تقریباً برابر شود. در این حالت، مهلت استفاده شده در روش مینیمم هزینه، همان مینیمم زمان محاسبات خواهد بود.

الگوریتم‌های زمانبندی با هدف بهینه‌سازی هزینه و یا زمان از دو نوع می‌باشند: زمانبندی مرحله‌ای و زمانبندی یکباره. الگوریتم‌هایی که از روش زمانبندی مرحله‌ای استفاده می‌کنند، در طول اجرای خود به تدریج کارها را به منابع موجود در گرید واگذار می‌کنند. در مقابل، الگوریتم‌هایی که رویکرد زمانبندی یکباره را اتخاذ می‌کنند، یک نگاشت از کارهای برنامه کاربر به منابع موجود را تولید می‌کنند که برای بررسی به بخش کنترل پذیرش داده می‌شود. پس از بررسی، اگر نگاشت بدست آمده، نیازمندی‌های کاربر را برآورده کند، کارها طبق این نگاشت به منابع برای اجرا واگذار می‌گردد. برای توضیحات بیشتر می‌توان به [۱] مراجعه کرد.

الگوریتم‌های بهینه‌سازی هزینه و یا زمان، دارای چند مرحله مشترک ابتدایی می‌باشند. این مراحل عبارتند از یافتن، تجارت و مرتب‌سازی منابع. الگوریتم‌هایی که از روش زمانبندی یکباره استفاده می‌کنند، در پایان نیز دارای دو

مرحله مشترک کنترل پذیرش و توزیع هستند. در ادامه توضیحاتی درباره این مراحل داده می‌شود [۱].

یافتن منابع[۱۴]: شناسایی منابعی که می‌توانند در اجرای کارها مورد استفاده قرار بگیرند و همچنین بدست آوردن توانمندی‌ها و ویژگی‌های آنها که از طریق سرویس اطلاعاتی گرید انجام می‌گیرد.

تجارت منابع[۱۵]: شناسایی هزینه هر یک از منابع در واحد زمان (G$/sec) ، شناسایی میزان توانمندی منبع در واحد زمان (نرخ اجرای میلیون دستورالعمل در ثانیه، MI/sec) و محاسبه قیمت واقعی و مفید منبع که همان هزینه اجرای یک میلیون دستورالعمل در ثانیه می‌باشد (G$/MI).

مرتب‌سازی منابع: یک الگوریتم بهینه‌سازی، حتماً به یک ترتیب خاص از منابع بر حسب قیمت آنها نیاز دارد. در این مرحله، منابع به ترتیب صعودی از نظر قیمت مفید مرتب‌سازی می‌شوند. در واقع، اولویت واگذاری کار با منابع ارزان‌تر می‌باشد. اگر دو منبع دارای قیمت مفید یکسان باشند، منبعی که توانمندی بیشتری دارد در اولویت قرار می‌گیرد.

کنترل پذیرش[۱۶]: هزینه کل اجرای کارها بر روی منابع بر طبق زمانبندی مشخص و با بودجه تعیین شده توسط کاربر مقایسه می‌گردد. همچنین زمان تقریبی اجرای برنامه بر طبق زمانبندی محاسبه و سپس با مهلت تعیین شده توسط کاربر مقایسه می‌شود. در صورتی که هزینه‌ها در محدوده بودجه باشد و اجرای برنامه قبل از مهلت زمانی پایان می‌یابد، کاربر پذیرفته می‌شود.

توزیع[۱۷]: در این مرحله در صورت پذیرش کاربر، کارها بر طبق نگاشت تعیین شده، به منابع واگذار می‌شود.

بدلیل اینکه منابع در گرید، کارها را به طور موازی اجرا می‌کنند، زمانی که برای اجرای تمامی کارها صرف می‌شود، مدت زمانی است که پرکارترین منبع (منبعی که بیشترین زمان را برای اجرای کارهای خود نیاز دارد) برای اجرای کارهای واگذار شده به او نیاز دارد. در واقع، پس از اینکه پرکارترین منبع، پردازش کارهای خود را به پایان رساند، گرید می‌تواند نتیجه اجرای برنامه کاربردی را به کاربر تحویل دهد. به این ترتیب، منظور از اینکه یک الگوریتم بهینه‌سازی زمان باید زمان اجرای کارها را تا حد ممکن کاهش دهد، این است که الگوریتم باید زمان اجرای کارها را در پرکارترین منبع کاهش دهد. این عمل با انتقال کارهای نگاشت شده به این منبع به منابعی که بتوانند زودتر کار را اجرا کنند، انجام می‌گیرد. البته انجام این عمل تا زمانی ممکن است که بودجه کافی برای انتقال کارها به منابع سریع‌تر موجود باشد.

الگوریتم‌های پیشنهادی در این مقاله، از روش زمانبندی یکباره استفاده می‌کنند و بنابراین امکان تضمین محدودیت‌های بودجه و مهلت برای چندین کاربر به طور همزمان وجود خواهد داشت. برای توضیحات بیشتر می‌توان به [۱] مراجعه کرد.

۲-۱- الگوریتم BTO

الگوریتم BTO تنها الگوریتم زمانبندی گزارش شده با هدف مینیمم کردن زمان در محدوده بودجه تعیین شده در گریدهای محاسباتی اقتصادی با مدل بازار کالا تا پیش از نگارش این مقاله می‌باشد. این الگوریتم توسط بویا در [10,11] گزارش شده و در [12] مورد ارزیابی قرار گرفته است. الگوریتم بویا با در نظر گرفتن مقداری از بودجه برای هر کار، منبعی را برای یک کار انتخاب می‌کند که بتواند اجرای آن کار را زودتر از منابع دیگر به اتمام برساند و هزینه انجام کار بر روی آن، از بودجه اختصاص یافته به کار، بیشتر نباشد. در الگوریتم بویا، تخصیص بودجه به کارها به نسبت طول آنها انجام می‌گیرد.

الگوریتم BTO ابتدا کارها را به طور صعودی بر حسب طول مرتب می‌کند. سپس کوتاهترین کار، انتخاب شده و متناسب با طول این کار به مجموع طول همه کارها به آن بودجه تخصیص داده می‌شود. بودجه تخصیص داده شده از کل بودجه کسر می‌شود. سپس از بین منابع موجود منابعی که می‌توانند کار را با توجه به بودجه اختصاص داده شده انجام دهند، مشخص می‌شوند. از بین این منابع، منبعی که می‌تواند کار را در کوتاهترین زمان انجام دهد به این کار تخصیص داده می‌شود. در صورتیکه هزینه انجام این کار توسط منبع از بودجه تخصیص یافته کمتر باشد، تفاوت به کل بودجه اضافه می‌شود و طول این کار از مجموع طول کارها کسر می‌گردد. سپس الگوریتم زمانبندی، عملیات فوق

را برای دومین کوتاهترین کار، سومین کوتاهترین و... تکرار می‌کند تا همه کارها به منابع نگاشت شوند. در مرحله توزیع، کارها به منابع منتسب شده، واگذار می‌شوند؛ البته به هر منبع، حداکثر به تعداد پردازنده‌های بیکار آن، کار واگذار می‌شود. کارهای باقیمانده، دوباره در صف کارهای پردازش نشده، قرار می‌گیرند. دو مرحله زمانبندی و توزیع، تا پایان واگذاری کارها به منابع، تکرار می‌شوند (زمانبندی مرحله‌ای). الگوریتم زمانبندی BTO به طور دقیق‌تر در شکل (۱) آورده شده است.

تا زمانی که کارهای پردازش نشده وجود دارد و از محدودیت‌های بودجه و مهلت تجاوز نکرده‌ایم، تکرار کن:

۱. **زمانبندی:** برای هر کار در صف کارهای پردازش نشده انجام بده:
   a. با توجه به نسبت طول کار به مجموع طول کارهای باقیمانده، بودجه‌ای به این کار اختصاص داده و سپس این بودجه را از بودجه کل کسر کن.
   b. گروهی از منابع که هزینه اجرای کار بر روی آنها از بودجه اختصاص داده شده، کمتر است، از بقیه منابع جدا کن.
   c. در گروه منابع انتخاب شده، منبعی را که بتواند زودتر از منابع دیگر گروه، کار را اجرا کند، انتخاب کرده و کار را به آن منتسب کن.
   d. هزینه اجرای کار بر روی منبع انتخابی را از بودجه اختصاص داده شده به این کار کسر کرده و مابقی را به بودجه کل برگردان.
   e. کار را از صف کارهای پردازش نشده حذف کن.
۲. **توزیع:** تعداد کارهایی را که می‌توان به یک منبع واگذار کرد، بدون اینکه با حالت اضافه‌بار مواجه شود، مشخص کرده و آنها را واگذار کن. سپس بقیه کارها را به صف کارهای پردازش نشده برگردان. سیاست پیش‌فرض این است که تعداد کارهای واگذار شده به یک منبع از تعداد پردازنده‌های آن بیشتر نباشد.
۳. برای مدت زمان مشخصی الگوریتم را متوقف کن. در این مدت زمان باید احتمال بیکار شدن حداقل یکی از پردازنده‌های منابع وجود داشته باشد.

شکل (۱): الگوریتم BTO

۳- الگوریتم‌های پیشنهادی

الگوریتم اول (ABTO): زمان بدست آمده برای اجرای مجموعه کارهای کاربر، زمانی است که پرکارترین منبع برای اجرای کارهای واگذار شده به او، نیاز دارد. الگوریتم بهینه‌سازی زمان باید سعی کند تا زمان اجرای کارها در پرکارترین منبع را به حداقل برساند. این عمل می‌تواند با انتقال کارها از پرکارترین منبع به منابع دیگر انجام شود. هنگامی که کارها به ترتیب در حال زمانبندی هستند، به تدریج در صف کارهای منابع قرار می‌گیرند. هر کار با توجه به بودجه خود به منبعی واگذار می‌شود که بتواند زودتر از منابع دیگر، آن را اجرا کند. اگر در ابتدا به جای کارهای بزرگتر، کارهای کوچکتر را زمانبندی کنیم، اختلاف زمان اجرای کارها در بین منابع زیاد می‌شود و در نتیجه اختلاف زمان اجرا در پرکارترین منبع و منابع دیگر، افزایش می‌یابد؛ زیرا به علت محدودیت بودجه باقیمانده، نمی‌توان به راحتی شرایط انتقال یک کار بزرگ را از پرکارترین منبع به یک منبع سریع‌تر فراهم کرد. اگر در ابتدا به جای کارهای کوچکتر، کارهای بزرگتر را توزیع کنیم، در اواخر فرایند زمانبندی، با استفاده از بودجه باقیمانده، می‌توان کارهای کوچکتر باقیمانده را به طریقی بین منابع توزیع کنیم که زمان لازم برای اجرای کارهای واگذار شده در پرکارترین منبع نسبت به منابع دیگر اختلاف زیادی نداشته باشد. بنابراین برای کاهش زمان اجرای کارهای برنامه کاربر، بهتر است صف کارها بر خلاف الگوریتم BTO که به صورت صعودی بر حسب طول مرتب می‌شود، به صورت نزولی بر حسب طول مرتب شود. الگوریتم ABTO در ابتدای مرحله زمانبندی، صف کارهای پردازش نشده را بر حسب طول کارها به صورت نزولی مرتب می‌کند. این تغییر، بهبود قابل ملاحظه‌ای را در الگوریتم BTO ایجاد می‌کند.

الگوریتم دوم (EBTO): الگوریتم زمانبندی EBTO که از روش زمانبندی یکباره استفاده می‌کند، با انجام تغییراتی از الگوریتم BTO حاصل می‌شود. الگوریتم EBTO بدلیل اینکه از زمانبندی یکباره استفاده می‌کند، پس از نگاشت هر کار به یک منبع، بلافاصله عمل واگذاری آن را به منبع انتخابی برای اجرا انجام می‌دهد. به این ترتیب با زمانبندی یکباره، امکان سرویس‌دهی همزمان به چندین کاربر فراهم می‌شود. این الگوریتم در ابتدا کارها را از نظر طول به صورت نزولی مرتب می‌کند و سپس با استفاده از روشی که بویا برای اختصاص بودجه به کارها مطرح کرده بود، اولین کار را از مجموعه کارها انتخاب و بودجه‌ای به آن اختصاص می‌دهد. از بین منابعی که می‌توانند این کار را با هزینه‌ای کمتر از بودجه اختصاص یافته به آن اجرا کنند، منبعی را انتخاب می‌کند که بتواند اجرای کار را زودتر از بقیه تمام کند و سپس این کار را به منبع برای اجرا واگذار می‌کند. این عمل برای همه کارها به ترتیب انجام می‌شود تا

واگذاری همه آنها به اتمام برسد. مراحل این الگوریتم، به طور دقیق‌تر در شکل (۲) آمده است.

۱. کارها را به صورت نزولی بر حسب طول مرتب کن.
۲. برای هر کار (به ترتیب) انجام بده:
a. با توجه به نسبت طول کار به مجموع طول کارهای باقیمانده، بودجه‌ای به کار اختصاص داده می‌شود و سپس بودجه اختصاص داده شده، از بودجه کل کسر می‌شود.
b. گروهی از منابع که هزینه اجرای کار بر روی آنها از بودجه اختصاص داده شده، کمتر است، تعیین گردد.
c. در گروه منابع انتخاب شده، کار را به منبعی واگذار کن که بتواند زودتر از منابع دیگر گروه، آن را به اتمام برساند.
d. هزینه اجرای کار بر روی منبع انتخابی را از بودجه اختصاص داده شده کم کرده و مابقی بودجه را به بودجه کل برگردان.
e. کار را از صف کارها خارج کن.

شکل (۲): الگوریتم EBTO

الگوریتم سوم (AEBTO): برای اینکه بتوان زمان اجرا در پرکارترین منبع را کاهش داد، باید نوعی تعادل بین منابع استفاده شده از نظر زمان صرف شده برای اجرای کارها ایجاد کرد. کارهای بزرگ برای اجرا بر روی منابع گران در مقایسه با کارهای کوچک نیاز به بودجه بیشتری دارند. بنابراین اگر بتوانیم به کارهای بزرگتر بودجه بیشتری تخصیص بدهیم، می‌توان امیدوار بود که کارهای بزرگ و کوچک تا حد امکان بطور یکسان برروی منابع ارزان و گران توزیع می‌شوند. الگوریتم AEBTO از الگوریتم EBTO با تغییر روش تخصیص بودجه به کارها بدست می‌آید. در روش تخصیص بودجه به کارها در الگوریتم AEBTO سعی می‌شود بودجه اختصاص یافته به هر کار که باقی می‌ماند، برای کارهای بزرگ استفاده شود. الگوریتم AEBTO در ابتدا صف کارها را بر حسب طول به صورت نزولی مرتب می‌کند. سپس بر خلاف الگوریتم EBTO، هنگامی که در حال زمانبندی کارها می‌باشد، به جای اینکه مقدار اضافی بودجه اختصاص یافته به یک کار را به بودجه کل برگرداند، به بودجه کار بعدی اضافه می‌کند. از آنجا که در ابتدا کارهای بزرگتر زمانبندی می‌شوند، این عمل باعث می‌شود کارهای بزرگ بودجه بیشتری نسبت به طولشان دریافت کنند و در نتیجه بتوانند بر روی منابع گرانتر نیز اجرا شوند. جزئیات الگوریتم AEBTO در شکل (۳) آمده است.

۱. کارها را به صورت نزولی بر حسب طول مرتب کن.
۲. برای هر کار به ترتیب انجام بده
a. با توجه به نسبت طول کار به مجموع طول کارهای باقیمانده، بودجه‌ای را به آن کار اختصاص داده و سپس این بودجه را از بودجه کل کسر کن.
b. مقداری را که از بودجه کار قبلی باقیمانده است، به بودجه اختصاص یافته به این کار، اضافه کن.
c. گروهی از منابع را که هزینه اجرای کار بر روی آنها از بودجه اختصاص داده شده، کمتر است، از بقیه منابع جدا کن.
d. در گروه منابع انتخاب شده، کار را به منبعی واگذار کن که بتواند زودتر از منابع دیگر گروه، آن را به اتمام برساند.
e. هزینه اجرای کار بر روی منبع انتخابی را از بودجه اختصاص داده شده، کم کرده و مابقی را برای کار بعدی ذخیره کن.
f. کار را از صف کارها خارج کن.

شکل (۳): الگوریتم AEBTO

۴- نتایج شبیه‌سازی‌ها

الگوریتم‌های پیشنهادی با استفاده از جعبه‌ابزار GridSim [12] شبیه‌سازی شده و نتایج آنها با نتایج بدست آمده از الگوریتم بویا (BTO) مقایسه شده است. نتایج گزارش شده، میانگین ۲۰ بار شبیه‌سازی می‌باشد. محیط شبیه‌سازی شده برای گرید، شامل تعدادی منبع و یک کاربر است. کلیه منابع محاسباتی دارای یک پردازنده هستند که مشخصات آنها در جدول (۱) داده شده است. همانطور که مشاهده می‌شود، ناهمگونی بالایی برای منابع در نظر گرفته شده است. برنامه کاربر از ۲۰۰ کار مستقل از هم تشکیل شده است که طول هر کار به صورت تصادفی از محدوده (۲۰۰۰۰۰...۱۰۰۰۰) انتخاب می‌شود. گستردگی این محدوده، منجر به ناهمگونی کارها می‌گردد. کاربر مقدار ۱۲۰۰۰۰ را برای بودجه تعیین کرده و بهینه‌سازی زمان را درخواست می‌کند.

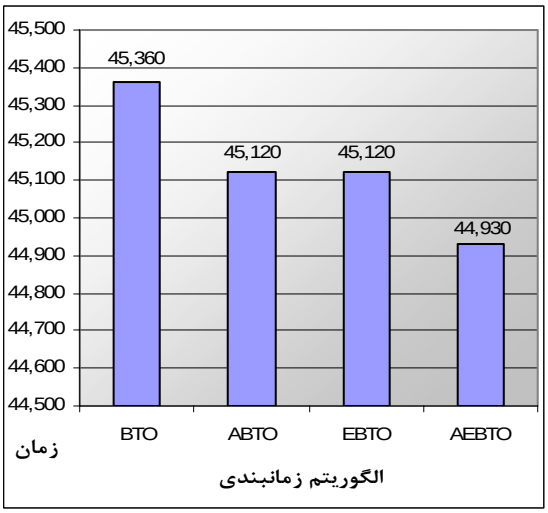
جدول (۱): پیکربندی منابع

نام منبع	نرخ اجرا (MI/sec)	قیمت (G$/sec)	قیمت مفید (G$/1000MI)
R1	۱۰۰	۰,۵	۵
R2	۱۸۰	۱,۰	۵,۵۵
R3	۲۴۰	۱,۵	۶,۲۵
R4	۲۸۰	۲	۷,۱۴
R5	۳۰۰	۲,۵	۸,۳۳
R6	۴۰۰	۴	۱۰
R7	۵۰۰	۶	۱۲
R8	۶۰۰	۹	۱۵

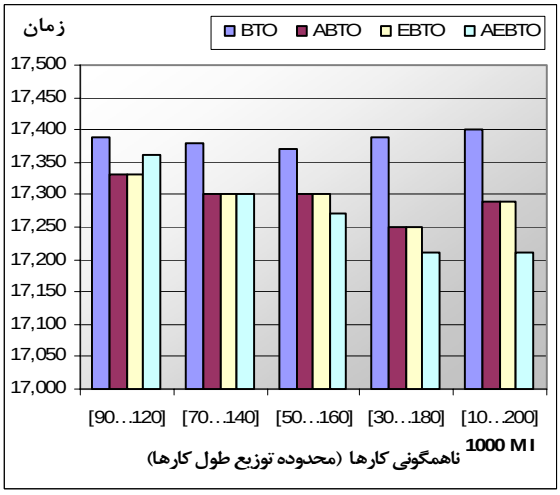
در اولین آزمایش که نتایج آن در شکل (۴) نشان داده شده است، الگوریتم‌های پیشنهادی و الگوریتم BTO با یکدیگر مقایسه شده‌اند. همانطور که مشاهده می‌شود، الگوریتم AEBTO در مقایسه با سایر الگوریتم‌ها، از عملکرد بهتری برخوردار بوده و توانسته است زمان را در مقایسه با الگوریتم BTO تا بیش از ۴۰۰ واحد (۱ درصد) کاهش دهد.

در دومین آزمایش، تأثیر ناهمگونی کارها را بر زمان اجرای برنامه کاربر بررسی می‌کنیم. برای این منظور، محدوده توزیع طول کارها را تغییر می‌دهیم. محدوده‌های آزمایش شده و نتایج بدست آمده در شکل (۵) نشان داده شده است. همانطور که مشاهده می‌شود با افزایش ناهمگونی کارها، الگوریتم AEBTO توانسته است زمان اجرای برنامه کاربر را کاهش دهد؛ در حالی که الگوریتم BTO در هنگام وجود ناهمگونی زیاد در کارها، نمی‌تواند تغییری در عملکرد خود داشته باشد.

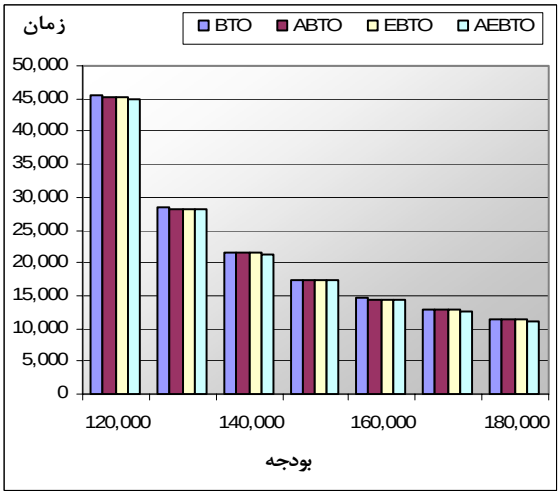
در آخرین آزمایش، تأثیر تغییرات بودجه بر روی زمان اجرای برنامه کاربر را بررسی می‌کنیم. نتایج این آزمایش در شکل (۶) نشان داده شده است. مشاهده می‌شود که با افزایش بودجه، زمان اجرای برنامه کاهش می‌یابد؛ زیرا الگوریتم‌های زمانبندی می‌توانند از منابع توانمندتر (که قیمت بیشتری دارند) استفاده کرده و کارها را با سرعت بیشتری اجرا کنند.



شکل (۴): مقایسه الگوریتم‌های بهینه‌سازی زمان



شکل (۵): مقایسه الگوریتم‌ها در ناهمگونی‌های مختلف



شکل (۶): نتایج الگوریتم‌ها با بودجه‌های مختلف

۵- نتیجه‌گیری

در این مقاله، سه الگوریتم مکاشفه‌ای جدید برای زمانبندی در گریدهای محاسباتی اقتصادی با مدل بازار کالا، به منظور مینیمم کردن زمان در محدوده بودجه تعیین شده پیشنهاد گردید. این الگوریتم‌ها با استفاده از جعبه‌ابزار GridSim شبیه‌سازی شده و کارایی آنها در شرایط مختلف، مورد بررسی قرار گرفت. نشان داده شد که الگوریتم‌های پیشنهادی در مقایسه با تنها الگوریتم گزارش شده، از کارایی بالاتری برخوردار بوده و در همه شرایط آزمایش شده، برنامه کاربر را در زمان کمتری اجرا می‌کنند. در بین الگوریتم‌های پیشنهادی، الگوریتم AEBTO نتایج بهتری را تولید کرده است.

مراجع

[۱] ياسر مهدوي‌فر و محمدرضا ميبدي، الگوريتم‌هاي زمانبندي براي بهينه‌سازي هزينه در گريدهاي محاسباتي اقتصادي، گزارش فني، آزمايشگاه سيستم‌هاي نرم‌افزاري، دانشكده مهندسي كامپيوتر و فناوري اطلاعات، دانشگاه صنعتي اميركبير، تهران، دي ۱۳۸۴.

[2] I. Foster and C. Kesselman, *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann, San Francisco, 1999.

[3] I. Foster, C. Kesselman and S. Tuecke, *"The anatomy of the Grid: Enabling scalable virtual organizations"*, International Journal of Supercomputer Applications, 2001.

[4] M. Baker, R. Buyya and Domenico Laforenza, *"Grids and Grid technologies for wide-area distributed computing"*, The Journal of Concurrency and Computation: Practice and Experience, Vol 14, Issue 13-15, Nov. 2002.

[5] V. Berstis, *Fundamentals of Grid Computing*, IBM Redbooks, November 2002.

[6] R. Buyya, D. Abramson, and J. Giddy, *"A Case for Economy Grid Architecture for Service-Oriented Grid Computing"*, Proceedings of the 10th IEEE International Heterogeneous Computing Workshop, April 2001.

[7] R. Buyya, D. Abramson, and J. Giddy, *"An Economy Driven Resource Management Architecture for Global Computational Power Grids"*, Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications, June 2000.

[8] R. Buyya, D. Abramson, and J. Giddy, *"Nimrod-G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid"*, The 4th International Conference on High Performance Computing in Asia-Pacific Region, May 2000.

[9] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, *"Economic Models for Resource Management and Scheduling in Grid Computing"*, The Journal of Concurrency and Computation: Practice and Experience, May 2002.

[10] R. Buyya, J. Giddy, D. Abramson, *"An Evaluation of Economy-based Resource Trading and Scheduling on Computational Power Grids for Parameter Sweep Applications"*, Proceedings of the 2nd International Workshop on Active Middleware Services, August 2000.

[11] R. Buyya, *Economic-based Distributed Resource Management and Scheduling for Grid Computing*, Ph.D. Thesis, School of Computer Science and Software Engineering, Monash University, Melbourne, Australia, April 2002.

[12] R. Buyya and M. Murshed, *"GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing"*, Journal of Concurrency and Computation: Practice and Experience, pp. 1-32, May 2002.

زیرنویس‌ها

[1] Computational Grids
[2] Heterogeneous
[3] Workstations
[4] Commodity Market Model
[5] Deadline
[6] Heuristic
[7] Buyya Time Optimization
[8] Parameter Sweep Applications
[9] Advanced Buyya Time Optimization
[10] Extended Buyya Time Optimization
[11] Advanced Extended Buyya Time Optimization
[12] Million Instruction
[13] Homogeneous
[14] Resource Discovery
[15] Resource Trading
[16] Admission Control
[17] Dispatching