بهينه‌سازي زمان در گريدهاي محاسباتي اقتصادي با استفاده از اتوماتاهاي يادگير

ياسر مهدوي‌فر[*] محمدرضا ميبدي[†]

چكيده

كاربران در گريدهاي محاسباتي اقتصادي بايستي براي اجراي برنامه‌هاي كاربردي خود هزينه پرداخت كنند. هر كاربر، مهلت زماني و بودجه مورد نظر خود را تعيين كرده و مي‌تواند بهينه‌سازي هزينه يا زمان را درخواست كند. يک الگوريتم زمانبندي با هدف بهينه‌سازي زمان، بايد با توجه به قيمت و توانمندي منابع گريد، عمل تخصيص آنها به كارهاي ناهمگون كاربر را طوري انجام دهد كه اجراي كارها با صرف بودجه تعيين شده و در حداقل زمان پايان يابد. در اين مقاله، يک الگوريتم مبتني بر اتوماتاهاي يادگير، براي اين منظور پيشنهاد شده است. با استفاده از شبيه‌سازي نشان داده شده است كه الگوريتم پيشنهادي در مقايسه با الگوريتم‌هاي مكاشفه‌اي گزارش شده، از كارايي بالاتري برخوردار بوده و درخواست‌هاي كاربر را در زمان كمتري انجام مي‌دهد.

كلمات كليدي: گريد محاسباتي، زمانبندي اقتصادي، بهينه‌سازي زمان، اتوماتاي يادگير

Time Optimization in Economic Computational Grids Using Learning Automata

Yasser Mahdavifar Mohammad Reza Meybodi

Soft Computing Laboratory
Computer Engineering and Information Technology Department
Amirkabir University of Technology, Tehran, Iran

Abstract

In economic computational grids, resources have price and the users must pay for executing their applications. The user determines his deadline and budget and then requests cost or time optimization. A scheduling algorithm that adopts time optimization strategy, should allocate heterogeneous grid resources to heterogeneous user jobs so that their execution finishes with specified budget and in minimum time. In this paper, a new algorithm is introduced for this purpose that uses learning automata. It is shown by using simulation that suggested algorithm has higher performance and performs users' requests in less time with respect to the reported heuristics.

Keywords: Computational Grid, Economic Scheduling, Time Optimization, Learning Automata

۱- مقدمه

گريد محاسباتي[۱] يک زيربناي سخت‌افزاري و نرم‌افزاري مي‌باشد كه دسترسي قابل‌اعتماد، پايدار، فراگير و ارزان را به توانايي‌هاي محاسباتي ديگران فراهم مي‌كند [3-6]. يک گريد محاسباتي با مجموعه‌اي از منابع ناهمگون[۲] (كامپيوترهاي شخصي، ايستگاه‌هاي كاري[۳]، كلاسترها و ابركامپيوترها)، در مقياس وسيع در ارتباط است. گريدهاي محاسباتي به تدريج به سوي تجاري شدن پيش مي‌روند و دارندگان منابع با انگيزه‌هاي مالي، منابع خود را در اختيار ديگران قرار مي‌دهند. مشتريان گريد نيز با پرداخت هزينه درخواست خود، مي‌توانند از اين منابع استفاده كنند. صاحبان و

---

[*] دانشجوي كارشناسي ارشد، دانشگاه صنعتي اميركبير، دانشكده مهندسي كامپيوتر و فناوري اطلاعات، mahdavifar@ce.aut.ac.ir

[†] عضو هيأت علمي دانشگاه، دانشگاه صنعتي اميركبير، دانشكده مهندسي كامپيوتر و فناوري اطلاعات، meybodi@ce.aut.ac.ir

استفاده‌کنندگان منابع داراي اهداف، استراتژي‌ها و الگوهاي عرضه و تقاضاي متفاوتي هستند. در چنين شرايطي نمي‌توان از راهکارهاي متداول براي مديريت منابع که سعي مي‌کنند ميزان کارايي کل سيستم را بهبود دهند، استفاده کرد. براي اين منظور، در سال‌هاي اخير از رويکردهاي اقتصادي براي مديريت تخصيص منابع در گريد استفاده شده است [7-10]. يکي از مدل‌هاي اقتصادي که بدين منظور استفاده شده است، مدل بازار کالا[4] مي‌باشد. در اين مدل، هر منبع داراي قيمت مشخصي است که بر اساس عرضه، تقاضا و ارزش در سيستم اقتصادي تعيين شده است.

منابع در گريد به صورت ناهمگون و توزيع شده هستند و به طور مشترک مورد استفاده قرار مي‌گيرند. از طرف ديگر، کاربران، محدوديت‌هاي مهلت[5] (زمان اتمام اجراي برنامه) و بودجه (هزينه محاسبات) را براي گريد تعيين مي‌کنند. اين مسائل، باعث پيچيدگي عمل زمانبندي برنامه کاربر مي‌گردد. با توجه به نياز کاربر، يکي از استراتژي‌هايي که يک الگوريتم زمانبندي مي‌تواند اتخاذ کند، مينيمم کردن زمان در محدوده بودجه (بهينه‌سازي زمان) مي‌باشد. تاکنون چندين الگوريتم مکاشفه‌اي[6] به منظور بهينه‌سازي زمان گزارش شده است. الگوريتم BTO توسط بويا در [11,12] پيشنهاد شده است. همچنين، براي برطرف کردن ضعف‌هاي اين الگوريتم، الگوريتم‌هاي مکاشفه‌اي ABTO، EBTO و AEBTO در [۲] ارائه و ارزيابي شده‌اند.

در اين مقاله با به‌کارگيري مجموعه‌اي از اتوماتاهاي يادگير، يک الگوريتم جديد به نام LATO[7] براي زمانبندي کارهاي مستقل از هم و با هدف بهينه‌سازي زمان در گريدهاي محاسباتي اقتصادي با مدل بازار کالا پيشنهاد مي‌شود. الگوريتم پيشنهادي، با استفاده از جعبه‌ابزار GridSim [13] شبيه‌سازي شده و کارايي آن مورد بررسي قرار گرفته است. همچنين، نتايج بدست آمده با نتايج الگوريتم‌هاي گزارش شده، مقايسه شده است. الگوريتم پيشنهادي بدليل انجام فرايند يادگيري، در مقايسه با الگوريتم‌هاي مکاشفه‌اي به زمان بيشتري نياز دارد. شبيه‌سازي‌ها نشان داده است که زمان لازم براي زمانبندي ۲۰۰ کار تقريباً ۱ ثانيه مي‌باشد. اين مقدار زمان در مقابل کاهشي که در زمان اجراي کارها بدست مي‌آيد قابل ملاحظه نمي‌باشد. همچنين الگوريتم‌هاي مکاشفه‌اي، در شرايط خاص مانند ناهمگوني بالاي کارها، داراي عملکرد مناسبي نمي‌باشند. به همين دليل استفاده از الگوريتم يادگير پيشنهادي به منظور زمانبندي کارها در گريد، داراي توجيه اقتصادي بوده و با جلب مشتريان بيشتر، موجب استفاده بهتر از منابع گريد و سودآوري بالاتر مي‌شود. ادامه مقاله بدين صورت سازماندهي شده است. در بخش ۲ اتوماتاهاي يادگير به اختصار معرفي مي‌گردد. در بخش ۳ زمانبندي اقتصادي در گريد، شرح داده مي‌شود و در بخش ٤ الگوريتم‌هاي زمانبندي پيشنهادي ارائه مي‌گردد. بخش ۵ اختصاص به نتايج شبيه‌سازي‌ها دارد و بخش ٦ نتيجه‌گيري مي‌باشد.

۲- اتوماتاهاي يادگير

یک اتوماتاي يادگير يك ماشين با حالات محدود[8] است که مي‌تواند تعداد محدودي عمل را انجام دهد. هر عمل انتخاب شده توسط محيطي احتمالي ارزيابي مي‌گردد و پاسخي به اتوماتاي يادگير داده مي‌شود. اتوماتاي يادگير از اين پاسخ استفاده نموده و عمل بعدي خود را انتخاب مي‌کند. در طي اين فرايند، اتوماتاي يادگير ياد مي‌گيرد که چگونه بهترين عمل را از بين اعمال مجاز خود انتخاب نمايد. شکل (۱) ارتباط بين اتوماتاي يادگير و محيط را نشان مي دهد. محيط را مي‌توان توسط سه تايي $E \equiv \{\alpha, \beta, c\}$ نشان داد که در آن $\alpha = \{\alpha_1, ..., \alpha_r\}$ مجموعه ورودي‌ها، $\beta = \{\beta_1, ..., \beta_m\}$ مجموعه خروجي‌ها و $c = \{c_1, ..., c_r\}$ مجموعه احتمال‌هاي جريمه مي‌باشد. هرگاه β مجموعه دو عضوي باشد، محيط از نوع P مي‌باشد. در چنين محيطي $\beta_1 = 1$ به عنوان جريمه و $\beta_2 = 0$ به عنوان پاداش در نظر گرفته مي‌شود. در محيط Q، $\beta(n)$ مي‌تواند به طور گسسته يک مقدار از مقادير محدود در فاصله [0,1] و در محيط از نوع S، هر مقدار در فاصله [0,1] را اختيار کند. c_i احتمال اينکه عمل α_i نتيجه نامطلوب داشته باشد، مي‌باشد. در محيط ايستا مقادير c_i در طي زمان بدون تغيير مي‌مانند؛ حال آنکه در محيط غير ايستا اين مقادير در طي زمان تغيير مي‌کنند. اتوماتاي يادگير به دو گروه اتوماتاهاي يادگير با ساختار ثابت و اتوماتاهاي يادگير با ساختار متغير تقسيم مي‌شود. در اين مقاله از اتوماتاهاي يادگير با ساختار متغير استفاده شده است که در ادامه اين بخش به شرح مختصري از آن مي‌پردازيم.

شکل (۱): رابطه بين اتوماتاي يادگير و محيط

اتوماتاهاي يادگير با ساختار متغير[۹]: اتوماتـاي يـادگير بـا سـاختار متغيـر توسـط ۴تـايي $\{\alpha, \beta, p, T\}$ نـشان داده مـي‌شـود کـه در آن $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ مجموعة عمل‌هاي اتوماتا، $\beta = \{\beta_1, \beta_2, \ldots, \beta_m\}$ مجموعة ورودي‌هاي اتوماتـا، $p = \{p_1, p_2, \ldots, p_m\}$ بـردار احتمال انتخاب هريک از عمل‌ها و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوريتم يادگيري مي‌باشد. در اين نوع از اتوماتاهاي يادگير، اگـر عمـل α_i در مرحلة nام انجام شود و پاسخ مطلوب از محيط دريافت نمايد، احتمال $p_i(n)$ کاهش يافته و ساير احتمال‌ها افـزايش مـي‌يابنـد. در هـر حـال، تغييرات به گونه‌اي صورت مي‌گيرد تا حاصل جمع $p_i(n)$ها همواره ثابت و مساوي يک باقي بماند. الگوريتم زير يک نمونـه از الگـوريتم‌هـاي يـادگير خطي است.

الف- پاسخ مطلوب

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)]$$
$$p_j(n+1) = (1-a)p_j(n) \qquad \forall j \quad j \neq i$$

ب- پاسخ نامطلوب

$$p_i(n+1) = (1-b)p_i(n)$$
$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad j \neq i$$

در روابط فوق، a پارامتر پاداش و b پارامتر جريمه است. با توجه به مقادير a و b سه حالت را مي‌توان در نظر گرفت. زمانيکه a و b با هم برابر باشند، الگوريتم را L_{RP}[۱۰] مي‌ناميم. زمانيکه b از a خيلي کوچکتر باشد، الگوريتم را $L_{R\varepsilon P}$[۱۱] مي‌ناميم. زمانيکه b مساوي صفر باشد، الگـوريتم را L_{RI}[۱۲] مي‌ناميم. براي اطلاعات بيشتر درباره اتوماتاهاي يادگير و کاربردهاي آن مي‌توان به [14,15] مراجعه نمود.

۳- زمانبندي اقتصادي در گريد

طول يک کار، تعداد دستورالعمل‌هاي آن مي‌باشد و بر حسب واحد MI[13] (ميليون دستورالعمل) اندازه‌گيري مي‌شود. اگر کارهـاي يـک برنامـه کاربردي، کاملاً همگون[۱۴] باشند (طول يکسان داشته باشند)، مي‌توان الگوريتم‌هاي زمانبندي با پيچيدگي چندجمله‌اي براي مينيمم کردن زمان در محدوده بودجه، طراحي کرد؛ ولي هنگامي که کارها داراي ناهمگوني باشند (طول‌هاي متفاوت داشته باشند)، مسأله زمانبندي با هدف بهينـه‌سـازي زمان به يک مسأله NP-Complete تبديل مي‌شود.

قيمت اعلام شده براي يک منبع، مقدار هزينه استفاده از آن در واحد زمان مي‌باشد و قيمت مفيد يک منبع بـه مقـدار هزينـه اسـتفاده از آن براي اجراي هر MI گفته مي‌شود. در اين مقاله، واژه‌هاي قيمت، ارزاني و گراني با توجه به قيمت مفيد منبع به کار مي‌روند.

کاربر گريد پس از اينکه محدوديت‌هاي مهلت زماني و بودجه خود را مشخص کرد، ممکن است بهينه‌سازي هزينه و يا زمان را درخواست کند. در صورتي که کاربر خواستار بهينه‌سازي زمان باشد، الگوريتم زمانبندي بايد با صرف بودجه‌اي که کاربر تعيين کرده است، کمتـرين زمـان را بـراي اجراي کارها بدست آورد و در صورتي که کاربر خواستار بهينه‌سازي هزينه باشد، الگوريتم زمانبندي بايد در محدوده مهلتي که کاربر تعيـين کـرده است، کمترين هزينه را براي اجراي کارها بدست آورد.

الگوريتم‌هاي زمانبندي با هدف بهينه‌سازي هزينه و يا زمان از دو نوع مي‌باشند: زمانبندي مرحله‌اي و زمانبندي يکباره. الگـوريتم‌هـايي کـه از روش زمانبندي مرحله‌اي استفاده مي‌کنند، در طول اجراي خود به تدريج کارها را به منابع موجود در گريد واگذار مي‌کنند. در مقابل، الگوريتم‌هايي که رويکرد زمانبندي يکباره را اتخاذ مي‌کنند، يک نگاشت از کارهاي برنامه کاربر به منابع موجود را توليد مي‌کنند که براي بررسي به بخش کنترل پذيرش داده مي‌شود. پس از بررسي، اگر نگاشت بدست آمده، نيازمندي‌هاي کاربر را برآورده کند، کارها طبق اين نگاشت به منابع براي اجرا واگـذار مي‌گردد. براي توضيحات بيشتر مي‌توان به [۱] مراجعه کرد.

الگوريتم‌هاي بهينه‌سازي هزينه و يا زمان، داراي چند مرحله مشترک ابتدايي مي‌باشند. اين مراحل عبارتند از يـافتن، تجـارت و مرتـب‌سـازي منابع. الگوريتم‌هايي که از روش زمانبندي يکباره استفاده مي‌کنند، در پايان نيز داراي دو مرحله مشترک کنترل پذيرش و توزيع هـستند. در ادامـه توضيحاتي درباره اين مراحل داده مي‌شود [۱].

- يافتن منابع[۱۵]: شناسايي منابعي که مي‌توانند در اجراي کارها مورد استفاده قرار بگيرند و همچنين بدست آوردن توانمندي‌هـا و ويژگـي‌هـاي آنها که از طريق سرويس اطلاعاتي گريد انجام مي‌گيرد.
- تجارت منابع[۱۶]: شناسايي هزينه هر يک از منابع در واحد زمان (G$/sec) ، شناسايي ميـزان توانمنـدي منبـع در واحـد زمـان (نـرخ اجـراي ميليون دستورالعمل در ثانيه، MI/sec) و محاسبه قيمت واقعي و مفيد منبع که همان هزينه اجراي يک ميليون دستورالعمل در ثانيه مي‌باشد (G$/MI).

- مرتب‌سازی منابع: یک الگوریتم بهینه‌سازي، حتماً به یک ترتیب خاص از منابع بر حسب قیمت آنها نیاز دارد. در این مرحله، منابع به ترتیب صعودي از نظر قیمت مفید مرتب‌سازي مي‌شوند. در واقع، اولویت واگذاري کار با منابع ارزان‌تر مي‌باشد. اگر دو منبع داراي قیمت مفید یکسان باشند، منبعي که توانمندي بیشتري دارد در اولویت قرار مي‌گیرد.
- کنترل پذیرش[17]: هزینه کل اجراي کارها بر روي منابع بر طبق زمانبندي مشخص و با بودجه تعیین شده توسط کاربر مقایسه مي‌گردد. همچنین زمان تقریبي اجراي برنامه بر طبق زمانبندي محاسبه و سپس با مهلت تعیین شده توسط کاربر مقایسه مي‌شود. در صورتي که هزینه‌ها در محدوده بودجه باشد و اجراي برنامه قبل از مهلت زماني پایان مي‌یابد، کاربر پذیرفته مي‌شود.
- توزیع[18]: در این مرحله در صورت پذیرش کاربر، کارها بر طبق نگاشت تعیین شده، به منابع واگذار مي‌شود.

بدلیل اینکه منابع در گرید، کارها را به طور موازي اجرا مي‌کنند، زماني که براي اجراي تمامي کارها صرف مي‌شود، مدت زماني است که پرکارترین منبع (منبعي که بیشترین زمان را براي اجراي کارهاي خود نیاز دارد) براي اجراي کارهاي واگذار شده به او نیاز دارد. در واقع، پس از اینکه پرکارترین منبع، پردازش کارهاي خود را به پایان رساند، گرید مي‌تواند نتیجه اجراي برنامه کاربردي را به کاربر تحویل دهد. به این ترتیب، منظور از اینکه یک الگوریتم بهینه‌سازي زمان باید زمان اجراي کارها را تا حد ممکن کاهش دهد، این است که الگوریتم باید زمان اجراي کارها را در پرکارترین منبع کاهش دهد. این عمل با انتقال کارهاي نگاشت شده به این منبع به منابعي که بتوانند زودتر کار را اجرا کنند، انجام مي‌گیرد. البته انجام این عمل تا زماني ممکن است که بودجه کافي براي انتقال کارها به منابع سریع‌تر موجود باشد.

الگوریتم پیشنهادي در این مقاله، از روش زمانبندي یکباره استفاده مي‌کند و بنابراین امکان تضمین محدودیت‌هاي بودجه و مهلت براي چندین کاربر به طور همزمان وجود خواهد داشت. براي توضیحات بیشتر مي‌توان به [۱] مراجعه کرد.

۴- الگوریتم پیشنهادي (LATO)

الگوریتم LATO الگوریتمي است مبتني بر اتوماتاهاي یادگیر که عمل پیچیده بهینه‌سازي زمان را با کارایي بالاتري در مقایسه با الگوریتم‌هاي مکاشفه‌اي موجود انجام مي‌دهد. این الگوریتم براي زمانبندي کارهاي برنامه کاربردي متعلق به کاربر، هر کدام از کارها را به یک اتوماتاي یادگیر، مجهز مي‌کند. هر یک از این اتوماتاهاي یادگیر، داراي ساختار متغیر بوده و داراي تعداد اعمال برابر با تعداد منابع در گرید مي‌باشد. عمل iام یک اتوماتاي یادگیر، انتخاب منبع iام در گرید مي‌باشد.

یک تکرار از الگوریتم LATO بدین قرار است. در ابتدا هر کدام از اتوماتاهاي یادگیر، یکي از اعمال خود را که یکي از منابع موجود در گرید مي‌باشد، انتخاب مي‌کند. سپس کار متناظر با این اتوماتاي یادگیر، در صف موقت منبع به نام صف انتساب که در ابتداي هر تکرار خالي مي‌شود، قرار داده مي‌شود. از این طریق، یک نگاشت از کارها به منابع در گرید ایجاد مي‌شود. سپس هزینه نگاشت ایجاد شده، محاسبه مي‌گردد. فرایند دادن پاداش و یا جریمه به اعمال انتخابي اتوماتاهاي یادگیر با توجه به هزینه محاسبه شده و نکات زیر که بیان کننده شرایطي است که بایستي در یک زمانبندي مورد توجه قرار گیرد، انجام مي‌گیرد. شرایطي که بایستي در یک زمانبندي مورد توجه قرار گیرد عبارتند از: ۱- مجموع هزینه اجراي کارها نباید از بودجه تعیین شده توسط کاربر بیشتر باشد، ۲- زمان اجراي کارهاي واگذار شده به پرکارترین منبع نباید با زمان اجرا در منابع دیگر اختلاف زیادي داشته باشد و ۳- هر کار باید به منبعي واگذار شود که بتواند اجراي آن کار را زودتر از بقیه منابع به اتمام برساند. در هر مرحله از الگوریتم، فرایند دادن پاداش و یا جریمه به شرح زیر انجام مي‌گیرد.

اگر مجموع هزینه از مقدار بودجه تعیین شده، فراتر رود، محیط، اعمال انتخابي اتوماتاهاي یادگیر کارهایي را که هزین هزیادي براي اجرا لازم دارند، بر طبق الگوریتم یادگیري جریمه مي‌کند. عمل جریمه کردن، از انتهاي صف کارهاي گران‌ترین منبع استفاده شده (گران‌ترین منبعي که حداقل یک کار به آن منتسب شده است) شروع مي‌شود. بعد از جریمه کردن هر کار و حذف آن از صف انتساب منبع انتخابي، تفاوت هزینه اجراي کار بر روي منبع انتخابي و هزینه اجراي آن بر روي ارزان‌ترین منبع موجود، از مقدار کل هزینه‌هاي اجرا کسر مي‌شود. عمل جریمه دادن تا زماني ادامه مي‌یابد که مقدار کل هزینه از مقدار بودجه تعیین شده، بیشتر شود. در واقع با این جریمه، کارها به سمت استفاده از منابع ارزانتر هدایت مي‌شوند.

در هر تکرار همچنین اگر یکي از منابع استفاده شده در زمانبندي (منابعي که حداقل یک کار به آنها منتسب شده دارند)، به غیر از آخرین (گران‌ترین) منبع استفاده شده، بتواند یک کار را زودتر از پرکارترین منبع، پردازش کند، عمل انتخاب شده اتوماتاي یادگیر متناظر با این کار بر طبق الگوریتم یادگیري جریمه مي‌شود. این عمل به توزیع متعادل کارها بین منابع کمک مي‌کند. در آخرین مرحله در یک تکرار، کارهایي که توانسته‌اند زمان اجراي کمتر یا برابر با زمان اجراي تکرار قبلي براي خود بدست بیاورند، پاداش مي‌گیرند؛ البته به شرطي که در همین تکرار، جریمه نشده باشند. از این طریق، یک کار سعي مي‌کند منبعي را براي اجراي خود انتخاب کند که در کمترین زمان، اجرایش را به اتمام برساند.

تعداد تکرارهاي الگوریتم بایستي با توجه به پارامترهاي گرید تعیین شود. در شبیه‌سازي‌هاي این مقاله، این تعداد ۱۰۰۰۰ در نظر گرفته شده است. الگوریتم LATO در شکل (۲) آورده شده است.

۱. مرتب‌سازي: کارها را از نظر طول به صورت نزولي مرتب کن.
۲. يادگيري: به تعداد ۱۰۰۰۰ بار تکرار کن:
 a. صف‌هاي انتساب منابع را خالي کن.
 b. انتخاب: براي هر کار به ترتيب انجام بده:
  i. با استفاده از اتوماتاي يادگير مربوطه، منبعي را براي واگذاري، انتخاب کن.
  ii. کار را در صف انتساب منبع انتخابي قرار بده.
 c. جريمه (۱): تا هنگامي که مجموع هزينه اجراي کارها بر روي منابع انتخاب شده از بودجه تعيين شده بيشتر است، تکرار کن:
  i. از انتهاي صف انتساب در گران‌ترين منبع استفاده شده، يک کار را انتخاب کن.
  ii. تفاوت هزينه اجراي کار بر روي اين منبع و هزينه اجرا بر روي ارزان‌ترين منبع موجود را از مجموع هزينه اجراي کارها کم کن.
  iii. عمل انتخابي اتوماتاي يادگير اين کار را جريمه کن.
 d. جريمه (۲): آخرين کار منتسب شده به پرکارترين منبع را در نظر بگير. اگر يکي از منابع استفاده شده به جز منبع آخر (گران‌ترين منبع استفاده شده) مي‌تواند اجراي اين کار را زودتر به اتمام برساند، اين کار را جريمه کن.
 e. پاداش: براي هر کار انجام بده:
  اگر عمل انتخابي اتوماتاي يادگير اين کار جريمه نشده است و اگر زمان اجراي اين کار بر روي منبع انتخابي کمتر و يا برابر با زمان اجراي آن در تکرار قبلي مي‌باشد، عمل انتخابي اتوماتاي يادگير اين کار را پاداش بده.
۳. زمانبندي: براي هر کار به ترتيب نزولي طول، با استفاده از اتوماتاي يادگير منتسب به آن، منبعي را انتخاب کرده و کار را به آن منبع واگذار کن.

شکل (۲): الگوريتم **LATO**

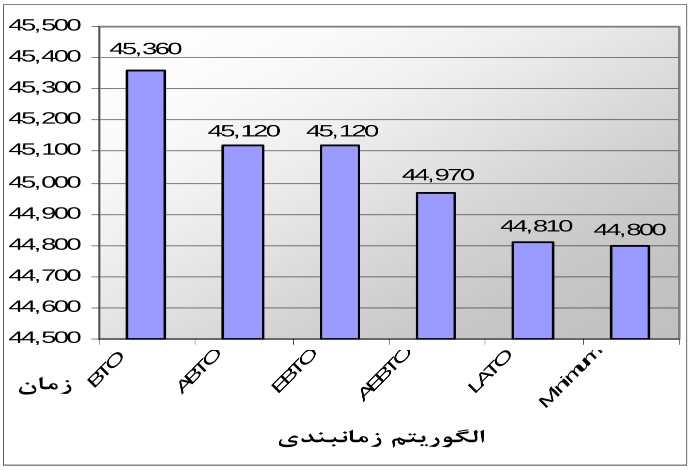
۵- نتايج شبيه‌سازي‌ها

الگوريتم پيشنهادي با استفاده از جعبه‌ابزار GridSim [13] شبيه‌سازي شده و نتايج آن با نتايج بدست آمده از الگوريتم‌هاي مکاشفه‌اي مقايسه شده است. نتايج گزارش شده، ميانگين ۲۰ بار شبيه‌سازي مي‌باشد. محيط شبيه‌سازي شده براي گريد، شامل تعدادي منبع و يک کاربر است. کليه منابع محاسباتي داراي يک پردازنده هستند که مشخصات آنها در جدول (۱) داده شده است. همانطور که مشاهده مي‌شود، ناهمگوني بالايي براي منابع در نظر گرفته شده است. برنامه کاربر از ۲۰۰ کار مستقل از هم تشکيل شده است که طول هر کار به صورت تصادفي از محدوده (۲۰۰۰۰...۱۰۰۰۰) انتخاب مي‌شود. گستردگي اين محدوده، منجر به ناهمگوني کارها مي‌گردد. کاربر مقدار ۱۲۰۰۰۰ را براي بودجه تعيين کرده و بهينه‌سازي زمان را درخواست مي‌کند. براي هر کار، اتوماتاي يادگير از نوع L_{RP} با مقادير $a = b = 0.01$ مي‌باشد.

جدول (۱): پيکربندي منابع

نام منبع	نرخ اجرا (MI/sec)	قيمت (G$/sec)	قيمت مفيد (G$/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 شکل (۳)، الگوريتم‌هاي مکاشفه‌اي BTO، ABTO، EBTO و AEBTO [۲] و الگوريتم پيشنهادي LATO با يکديگر مقايسه شده‌اند. همانطور که مشاهده مي‌شود، الگوريتم LATO کمترين زمان را براي اجراي کارها بدست آورده و در مقايسه با بهترين الگوريتم مکاشفه‌اي گزارش شده (AEBTO)، بيش از ۱۵۰ واحد (۰,۳ درصد) کاهش زمان را نشان مي‌دهد. اختلاف زمان بدست آمده توسط الگوريتم LATO با حداقل زمان مورد نياز براي اجراي کارها بسيار کم است. براي بدست آوردن حداقل زمان مورد نياز براي اجراي کارها، از روش ارائه شده در [۲] استفاده شده است.



شکل (۳): نتايج الگوريتم‌هاي زمانبندي با هدف بهينه‌سازي زمان

الگوريتم پيشنهادي بدليل انجام فرايند يادگيري، در مقايسه با الگوريتم‌هاي مكاشفه‌اي به زمان بيشتري نياز دارد. شبيه‌سازي‌ها نشان داده است كه زمان لازم براي زمانبندي ۲۰۰ كار تقريباً ۱ ثانيه مي‌باشد. اين مقدار زمان در مقابل كاهش زمان اجراي كارها كه در صورت استفاده از اين الگوريتم‌ها بدست مي‌آيد، قابل ملاحظه نمي‌باشد. همچنين الگوريتم‌هاي مكاشفه‌اي، در شرايط خاص مانند ناهمگوني بالاي كارها، داراي عملكرد مناسبي نمي‌باشند. به همين دليل، استفاده از الگوريتم پيشنهادي به منظور زمانبندي كارها در گريد، داراي توجيه اقتصادي مي‌باشد؛ زيرا كه حتي يک كاهش پايين در زمان اجراي كارها منجر به جلب مشتريان بيشتر، استفاده بهتر از منابع در گريد و سودآوري بالاتر براي گريد مي‌شود.

۶- نتيجه‌گيري

در اين مقاله، الگوريتمی مبتني بر اتوماتاهاي يادگير براي زمانبندي در گريدهاي محاسباتي اقتصادي با مدل بازار كالا به منظور مينيمم كردن زمان در محدوده بودجه تعيين شده، پيشنهاد گرديد. اين الگوريتم با استفاده از جعبه‌ابزار GridSim شبيه‌سازي شده و كارايي آن در محيط گريد با منابع و كارهاي ناهمگون، مورد بررسي قرار گرفت. نشان داده شد كه الگوريتم پيشنهادی (LATO) در مقايسه با الگوريتم‌هاي مكاشفه‌اي گزارش شده، از كارايي بالاتري برخوردار بوده و درخواست‌هاي كاربر را در زمان كمتري اجرا مي‌كند.

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.

[۲] ياسر مهدوي‌فر و محمدرضا ميبدي، الگوريتم‌هاي زمانبندي براي بهينه‌سازي زمان در گريدهاي محاسباتي اقتصادي، گزارش فني، آزمايشگاه سيستم‌هاي نرم‌افزاري، دانشكده مهندسي كامپيوتر و فناوري اطلاعات، دانشگاه صنعتي اميركبير، تهران، ارديبهشت ۱۳۸۵.

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

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

[5] 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.

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.
[13] 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.
[14] K. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.
[15] K. Najim and A. S. Poznyak, *Learning Automata: Theory and Application*, Elsevier Science Ltd., Tarrytown, NY, 1994.

زیرنویس‌ها

[1] Computational Grids
[2] Heterogeneous
[3] Workstations
[4] Commodity Market Model
[5] Deadline
[6] Heuristic
[7] Learning Automata Time Optimization
[8] Finite State Machine
[9] Variable structure
[10] Linear reward penalty
[11] Linear reward epsilon penalty
[12] Linear reward inaction
[13] Million Instruction
[14] Homogeneous
[15] Resource Discovery
[16] Resource Trading
[17] Admission Control
[18] Dispatching