بهينه‌سازي هزينه 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 هزينه، بايد با توجه به قيمت و توانمندي منابع گريد، عمل تخصيص آنها به كارهاي ناهمگون كاربر را طوري انجام دهد كه اجراي كارها با حداقل هزينه و قبل از مهلت تعيين شده، پايان يابد. در اين مقاله، دو الگوريتم مبتني بر اتوماتاهاي يادگير، براي اين منظور پيشنهاد شده است. با استفاده از شبيه‌سازي نشان داده شده است كه الگوريتم‌هاي پيشنهادي در مقايسه با الگوريتم‌هاي مكاشفه‌اي گزارش شده، از كارايي بالاتري برخوردار بوده و درخواست‌هاي كاربر را با هزينه كمتري انجام مي‌دهند.

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

Cost Optimization in Economic Computational Grids Using Learning Automata

Yasser Mahdavifar Mohammad Reza Meybodi

Soft Computing Lab.
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 cost optimization strategy, should allocate heterogeneous grid resources to heterogeneous user jobs so that their execution finishes in the specified deadline with minimum cost. In this paper, two new algorithms are introduced for this purpose that use learning automata. It is shown by using simulation that suggested algorithms have higher performance and perform users' requests with less cost with respect to the reported heuristics.

Keywords: Computational Grid, Economic Scheduling, Cost 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 [5-2]. 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 استفاده‌كنندگان منابع داراي اهداف، استراتژي‌ها و الگوهاي عرضه و تقاضاي متفاوتي هستند. در چنين شرايطي نمي‌توان از راهكارهاي متداول

* 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

براي مديريت منابع كه سعي مي‌كنند ميزان كارايي كل سيستم را بهبود دهند، استفاده كرد. براي اين منظـور، در سـال‌هـاي اخيـر از رويكردهـاي اقتصادي براي مديريت تخصيص منابع در گريد استفاده شده است [6-9]. يكي از مدل‌هاي اقتصادي كه بدين منظور استفاده شده است، مدل بـازار كالا[4] مي‌باشد. در اين مدل، هر منبع داراي قيمت مشخصي است كه 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، كاربران، محدوديت‌هـاي مهلت[5] (زمان اتمام اجراي برنامه) و بودجه (هزينه محاسبات) را براي گريد تعيين مي‌كنند. اين مسائل، باعث پيچيدگي عمل زمانبندي برنامه كاربر مي‌گردد. با توجه به نياز كاربر، يكي از استراتژي‌هايي كه يک الگوريتم زمانبندي مي‌تواند اتخـاذ كنـد، مينـيمم كـردن هزينـه در محـدوده مهلـت (بهينه‌سازي هزينه) مي‌باشد. تاكنون چندين الگوريتم مكاشفه‌اي[6] به منظور بهينه‌سازي هزينه گزارش شده اسـت. الگـوريتم BCO توسـط بويـا در [10,11] پيشنهاد شده است. همچنين، براي برطرف كردن ضعف‌هاي اين الگـوريتم، الگـوريتم‌هـاي مكاشـفه‌اي ABCO، EBCO و AEBCO در [1] ارائه و ارزيابي شده‌اند.

در اين مقاله با به‌كارگيري مجموعه‌اي از اتوماتاهاي يادگير، دو الگوريتم جديد به نـام‌هـاي LACO[7] و ALACO[8] بـراي زمانبنـدي كارهـاي مستقل از هم و با هدف بهينه‌سازي هزينه در گريدهاي محاسباتي اقتصادي با مدل بازار كالا پيشنهاد مي‌شود. الگوريتم‌هاي پيشنهادي، با اسـتفاده از جعبه‌ابزار GridSim [12] شبيه‌سازي شده و كارايي آنها مورد بررسي قرار گرفته است. همچنين، نتايج بدست آمده با نتايج الگوريتم‌هاي گزارش شده، مقايسه شده است. الگوريتم‌هاي پيشنهادي بدليل انجام فرايند يادگيري، در مقايسه با الگوريتم‌هاي مكاشفه‌اي به زمـان بيـشتري نيـاز دارنـد. شبيه‌سازي‌ها نشان داده است كه زمان لازم براي زمانبندي ۲۰۰ كار تقريباً ۱ ثانيه مي‌باشد. اين مقدار زمان در مقابل كاهشي كه در هزينه اجـراي كارها بدست مي‌آيد قابل ملاحظه نمي‌باشد. همچنين الگوريتم‌هاي مكاشفه‌اي، در شرايط خاص مانند ناهمگوني بالاي كارها، داراي عملكرد مناسبي نمي‌باشند. به همين دليل استفاده از الگوريتم‌هاي يادگير پيشنهادي به منظور زمانبندي كارها در گريـد، داراي توجيـه اقتـصادي بـوده و بـا جلـب مشتريان بيشتر، موجب استفاده بهتر از منابع گريد و سودآوري بالاتر مـي‌شـود. ادامـه مقالـه بـدين صـورت سـازماندهي شـده اسـت. در بخـش ۲ اتوماتاهاي يادگير به اختصار معرفي مي‌گردد. در بخش ۳ زمانبندي اقتصادي در گريد، شرح داده مي‌شـود و در بخـش ٤ الگـوريتم‌هـاي زمانبنـدي پيشنهادي ارائه مي‌گردد. بخش ٥ اختصاص به نتايج شبيه‌سازي‌ها دارد و بخش ٦ نتيجه‌گيري مي‌باشد.

۲- اتوماتاهاي يادگير

یک اتوماتاي يادگير يك ماشين با حالات محدود[9] است كه مي‌تواند تعداد محدودي عمل را انجام دهـد. هـر عمـل انتخـاب شـده توسـط محيطـي احتمالي ارزيابي مي‌گردد و پاسخي به اتوماتاي يادگير داده مي‌شود. اتوماتاي يادگير از اين پاسـخ اسـتفاده نمـوده و عمـل بعـدي خـود را انتخـاب مي‌كند. در طي اين فرايند، اتوماتاي يادگير ياد مي‌گيرد كه چگونه بهترين عمل را از بين اعمال مجاز خود انتخـاب نمايـد. شـكل (۱) ارتبـاط بـين اتوماتاي يادگير و محيط را نشان مي دهد. محـيط را مـي‌تـوان توسـط سـه تـايي $E \equiv \{\alpha, \beta, c\}$ نـشان داد كـه در آن $\alpha = \{\alpha_1, ..., \alpha_r\}$ مجموعـه 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، $\beta(n)$ مـي‌توانـد بـه 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 كند. c_i احتمال اينكه عمـل α_i نتيجه نامطلوب داشته باشد، مي‌باشد. در محيط ايستا مقادير c_i در طي زمان بدون تغيير مي‌مانند؛ حال آنكه در محيط غيـر ايـستا ايـن مقـادير در طي زمان تغيير مي‌كنند. اتوماتاي يادگير به دو گروه اتوماتاهاي يادگير با ساختار ثابت و اتوماتاهاي يادگير با ساختار متغير تقسيم مي‌شود. در ايـن مقاله از اتوماتاهاي يادگير با ساختار متغير استفاده شده است كه در ادامه اين بخش به شرح مختصري از آن مي‌پردازيم.

شكل (۱): رابطه بين اتوماتاي يادگير و محيط

اتوماتاهاي يادگير با ساختار متغير: اتوماتاي يادگير با ساختار متغير توسط ۴تـايي $\{\alpha, \beta, p, T\}$ نـشان داده مـي‌شـود كـه در آن $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_r\}$ مجموعة عمل‌هاي اتوماتا، $\beta = \{\beta_1, \beta_2, ..., \beta_m\}$ مجموعة ورودي‌هاي اتوماتـا، $p = \{p_1, p_2, ..., p_m\}$ بـ[illegible] 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، اگـ[illegible]

α_i در مرحلهٔ nام انجام شود و پاسخ مطلوب از محیط دریافت نماید، احتمال $p_i(n)$ کاهش یافته و سایر احتمال‌ها افـزایش مـی‌یابنـد. در هـر حـال، تغییرات به گونه‌اي صورت مي‌گیرد تا حاصل جمع $p_i(n)$ها همواره ثابت و مساوي یک باقي بماند. الگوریتم زیر یک نمونـه از الگـوریتم‌هـاي یـادگیر 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 جریمه است. با توجه به مقادیر a و b سه حالت را مي‌توان در نظر گرفت. زمانیکه a و b با هم برابر باشند، الگوریتم را L_{RP}[۱۱] مي‌نامیم. زمانیکه b از a خیلي کوچکتر باشد، الگوریتم را $L_{R\varepsilon P}$[۱۲] مي‌نامیم. زمانیکه b مساوي صفر باشد، الگـوریتم را L_{RI}[۱۳] مي‌نامیم. براي اطلاعات بیشتر درباره اتوماتاهاي یادگیر و کاربردهاي آن مي‌توان به [13,14] مراجعه نمود.

۳- زمانبندي اقتصادي در گرید

طول یک کار، تعداد دستورالعمل‌هاي آن مي‌باشد و بر حسب واحد MI[14] (میلیون دستورالعمل) اندازه‌گیري مي‌شود. اگر کارهـاي یـک برنامـه کاربردي، کاملاً همگون[۱۵] باشند (طول یکسان داشته باشند)، مي‌توان الگوریتم‌هاي زمانبندي با پیچیدگي چندجمله‌اي براي مینـیمم کـردن هزینـه در محدوده مهلت، طراحي کرد؛ ولي هنگامي که کارها داراي نـاهمگوني باشـند (طـول‌هـاي متفـاوت داشـته باشـند)، مـسأله زمانبنـدي بـا هـدف بهینه‌سازي هزینه به یک مسأله NP-Complete تبدیل مي‌شود.

قیمت اعلام شده براي یک منبع، مقدار هزینه استفاده از آن در واحد زمان مي‌باشد و قیمت مفید یک منبع بـه مقـدار هزینـه اسـتفاده از آن براي اجراي هر MI گفته مي‌شود. در این مقاله، واژه‌هاي قیمت، ارزاني و گراني با توجه به قیمت مفید منبع به کار مي‌روند.

کاربر گرید پس از اینکه محدودیت‌هاي مهلت زماني و بودجه خود را مشخص کرد، ممکن است بهینه‌سازي هزینه و یا زمان را درخواست کند. در صورتي که کاربر خواستار بهینه‌سازي زمان باشد، الگوریتم زمانبندي باید با صرف بودجه‌اي که کاربر تعیین کرده است، کمتـرین زمـان را بـراي اجراي کارها بدست آورد و در صورتي که کاربر خواستار بهینه‌سازي هزینه باشد، الگوریتم زمانبندي باید در محدوده مهلتي که کاربر تعیـین کـرده است، کمترین هزینه را براي اجراي کارها بدست آورد.

الگوریتم‌هاي زمانبندي با هدف بهینه‌سازي هزینه و یا زمان از دو نوع مي‌باشند: زمانبندي مرحله‌اي و زمانبندي یکباره. الگـوریتم‌هـایي کـه از روش زمانبندي مرحله‌اي استفاده مي‌کنند، در طول اجراي خود به تدریج کارها را به منابع موجود در گرید واگذار مي‌کنند. در مقابل، الگوریتم‌هایي که رویکرد زمانبندي یکباره را اتخاذ مي‌کنند، یک نگاشت از کارهاي برنامه کاربر به منابع موجود را تولید مي‌کنند که براي بررسي به بخش کنترل پذیرش داده مي‌شود. پس از بررسي، اگر نگاشت بدست آمده، نیازمندي‌هاي کاربر را برآورده کند، کارها طبق این نگاشت به منابع براي اجرا واگـذار مي‌گردد. براي توضیحات بیشتر مي‌توان به [۱] مراجعه کرد.

الگوریتم‌هاي بهینه‌سازي هزینه و یا زمان، داراي چند مرحله مشترک ابتدایي مي‌باشند. این مراحل عبارتند از یـافتن، تجـارت و مرتـب‌سـازي منابع. الگوریتم‌هایي که از روش زمانبندي یکباره استفاده مي‌کنند، در پایان نیز داراي دو مرحله مشترک کنترل پذیرش و توزیع هـستند. در ادامـه توضیحاتي درباره این مراحل داده مي‌شود [۱].

- یافتن منابع[۱۶]: شناسایي منابعي که مي‌توانند در اجراي کارها مورد استفاده قرار بگیرند و همچنین بدست آوردن توانمنديهـا و ویژگـي‌هـاي آنها که از طریق سرویس اطلاعاتي گرید انجام مي‌گیرد.
- تجارت منابع[۱۷]: شناسایي هزینه هر یک از منابع در واحد زمان (G$/sec) ، شناسایي میـزان توانمنـدي منبـع در واحـد زمـان (نـرخ اجـراي میلیون دستورالعمل در ثانیه، MI/sec) و محاسبه قیمت واقعي و مفید منبع که همان هزینه اجراي یک میلیون دستورالعمل در ثانیه مي‌باشد (G$/MI).
- مرتب‌سازي منابع: یک الگوریتم بهینه‌سازي، حتماً به یک ترتیب خاص از منابع بر حسب قیمت آنهـا نیـاز دارد. در ایـن مرحلـه، منـابع بـه 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 پذيرش[18]: هزينه كل اجراي كارها بر روي منابع بر طبق زمانبندي مشخص و با بودجه تعيين شده توسط كاربر مقايسه مي‌گردد. همچنين زمان تقريبي اجراي برنامه بر طبق زمانبندي محاسبه و سپس با مهلت تعيين شده توسط كاربر مقايسه مي‌شود. در صورتي كه هزينه‌ها در محدوده بودجه باشد و اجراي برنامه قبل از مهلت زماني پايان مي‌يابد، كاربر پذيرفته مي‌شود.
- توزيع[19]: در اين مرحله در صورت پذيرش كاربر، كارها بر طبق نگاشت تعيين شده، به منابع واگذار مي‌شود.

الگوريتم‌هاي پيشنهادي در اين مقاله، از روش زمانبندي يكباره استفاده مي‌كنند و بنابراين امكان تضمين محدوديت‌هاي بودجه و مهلت براي چندين كاربر به طور همزمان وجود خواهد داشت. براي توضيحات بيشتر مي‌توان به [۱] مراجعه كرد.

۴- الگوريتم‌هاي پيشنهادي

الگوريتم اول (**LACO**): الگوريتم LACO با به‌كارگيري مجموعه‌اي از اتوماتاهاي يادگير، عمل زمانبندي كارها را با هدف مينيمم كردن هزينه محاسبات در محدوده مهلت تعيين شده انجام مي‌دهد. اين الگوريتم براي زمانبندي كارهاي برنامه كاربردي متعلق به كاربر، هر كدام از كارها را به يک اتوماتاي يادگير مجهز مي‌كند. هر كدام از اتوماتاهاي يادگير، داراي ساختار متغير بوده و تعداد اعمال آن برابر با تعداد منابع در گريد مي‌باشد. عمل iام يک اتوماتاي يادگير، انتخاب منبع iام در گريد مي‌باشد. احتمال انتخاب عمل iام اتوماتاي يادگير يک كار در ابتداي الگوريتم 1/r مي‌باشد كه r برابر با تعداد منابع در گريد مي‌باشد.

الگوريتم LACO از روش زمانبندي يكباره براي زمانبندي كارها استفاده مي‌كند. در ابتداي هر تكرار، يک زمانبندي جديد با استفاده از بردارهاي احتمالات اعمال اتوماتاهاي يادگير كارها ايجاد و سپس ارزيابي مي‌گردد. بر اساس اين ارزيابي، اعمال انتخابي اتوماتاهاي يادگير كارها، پاداش و يا جريمه داده مي‌شود. تكرارهاي الگوريتم، ادامه مي‌يابد تا اينكه تعداد تكرارها از يک مقدار از پيش تعيين شده بيشتر شود و يا اينكه در چند تكرار متوالي، زمانبندي ايجاد شده تغيير نكند. كارها بر طبق زمانبندي نهايي ايجاد شده توسط الگوريتم، به منابع واگذار مي‌گردد. در اين الگوريتم، براي هر منبع، يک صف موقت به نام صف انتساب در نظر گرفته مي‌شود كه در ابتداي هر تكرار خالي مي‌شود. كارها به ترتيب نزولي طول آنها، با استفاده از اتوماتاهاي يادگير خود، منبعي را انتخاب مي‌كنند و سپس در صف انتساب منبع انتخاب شده قرار مي‌گيرند. در شكل (۲) الگوريتم LACO و مراحل آن آورده شده است.

۱. مرتب‌سازي: كارها را بر حسب طول به صورت نزولي مرتب كن.
۲. يادگيري: به تعداد ۱۰۰۰۰ بار تكرار كن:
   a. صف‌هاي انتساب منابع را خالي كن.
   b. انتخاب: براي هر كار به ترتيب نزولي طول انجام بده:
      i. بر طبق بردار احتمال انتخاب اعمال اتوماتاي يادگير كارانتخاب شده، منبعي را براي واگذاري به اين كار انتخاب كن.
      ii. كار را در صف انتساب منبع انتخابي قرار بده.
   c. جريمه: براي هر منبع انجام بده:
      از انتهاي صف انتساب به ترتيب: عمل انتخابي اتوماتاي يادگير هر كار را كه اجرايش پيش از مهلت تعيين شده باتمام نميرسد جريمه كن.
   d. پاداش: براي هر كار انجام بده:
      اگر عمل انتخابي اتوماتاي يادگير كار جريمه نشده است و اگر منبعي كه انتخاب كرده است ارزان‌تر يا هم‌قيمت با منبع انتخابي در تكرار قبل است، به عمل انتخابي اتوماتاي يادگير كار پاداش بده
۳. زمانبندي: براي هر كار به ترتيب، با استفاده از اتوماتاي يادگير متعلق به آن، منبعي را انتخاب كرده و كار را به آن منبع، نگاشت كن.

شكل (۲): الگوريتم **LACO**

الگوريتم دوم (**ALACO**): الگوريتم ALACO با تغيير در نحوه جريمه كردن اعمال انتخابي اتوماتاهاي يادگير در الگوريتم LACO بدست مي‌آيد. در الگوريتم LACO و همچنين در اين الگوريتم، صف انتساب منابع از نظر طول كارها به صورت نزولي مرتب است. محيط در الگوريتم LACO، از انتهاي صف منابع، كارهايي را كه قبل از مهلت تمام نمي‌شوند، حذف كرده و عمل انتخابي اتوماتاي يادگير آنها را جريمه مي‌كند. در حالي كه پس از حذف اين كارها ممكن است بتوان كارهاي كوچكي را در بين آنها دوباره به منبع برگرداند به طوري كه اجراي آنها قبل از مهلت تمام شود.

محيط در الگوريتم ALACO براي يک منبع، تمام كارهاي اضافه (كارهايي كه اجراي آنها پيش از مهلت تمام نمي‌شود) را حذف كرده و در صفي به نام صف كارهاي اضافه قرار مي‌دهد. سپس بزرگترين كار اضافه‌اي را كه بتواند در شرايط كنوني (كه همه كارهاي اضافه، حذف شده‌اند) توسط منبع تا قبل از مهلت اجرا شود، دوباره در صف انتساب قرار مي‌دهد. سپس در صف كارهاي اضافه، همه آنهايي را كه بزرگتر از آن كار بوده،

جريمه مي‌كند. كارهاي كوچكتر جريمه نمي‌شوند؛ ولي پاداش نيز نخواهند گرفت. به اين ترتيب، زمان بيكاري منابع تا مهلت تعيين شده به طور قابل ملاحظه‌اي كاهش مي‌يابد؛ زيرا كارهاي كوچكي كه بر روي منابع گران اجرا مي‌شدند، اكنون در الگوريتم ALACO در زمان بيكاري بزرگي كه در منابع ارزان وجود دارد اجرا مي‌شوند و هزينه محاسبات كاهش مي‌يابد. مراحل زمانبندي با استفاده از الگوريتم ALACO در شكل (۳) آمده است.

١. مرتب‌سازي: كارها را از نظر طول به صورت نزولي مرتب كن.
٢. يادگيري: به تعداد ١٠٠٠٠ بار تكرار كن:
   a. صف‌هاي انتساب منابع را خالي كن.
   b. انتخاب: براي هر كار به ترتيب انجام بده:
      i. با استفاده از اتوماتاي يادگير مربوطه، منبعي را براي واگذاري، انتخاب كن.
      ii. كار را در صف انتساب منبع انتخابي قرار بده.
   c. جريمه: براي هر منبع انجام بده:
      i. از انتهاي صف انتساب به ترتيب: هر كار را كه اجرايش پيش از مهلت تمام نمي‌شود در يک صف به نام صف كارهاي اضافه قرار بده و از صف انتساب حذف كن.
      ii. از انتهاي صف كارهاي اضافه به ترتيب: اگر كار مي‌تواند در شرايط كنوني روي منبع اجرا شده و قبل از مهلت تمام شود، آن كار را به صف انتساب برگردان و كارهاي كوچكتر را از صف كارهاي اضافه نيز حذف كن (تا نه جريمه شوند و نه پاداش بگيرند). در اين صورت بررسي كارهاي اضافه متوقف مي‌شود.
      iii. اعمال انتخابي اتوماتاهاي كارهاي باقيمانده در صف كارهاي اضافه را جريمه كن
   d. پاداش: براي هر كار انجام بده:
      اگر عمل انتخابي اتوماتاي يادگير آن جريمه نشده است و در صف انتساب يكي از منابع حضور دارد و اگر منبعي كه انتخاب كرده است، ارزان‌تر يا هم‌قيمت با منبع انتخابي در تكرار قبل است، به عمل انتخابي اتوماتاي يادگير آن كار پاداش بده.
٣. زمانبندي: براي هر كار به ترتيب، با استفاده از اتوماتاي يادگير متعلق به آن، منبعي را انتخاب كرده و كار را به آن منبع، نگاشت كن.

شكل (٣): الگوريتم **ALACO**

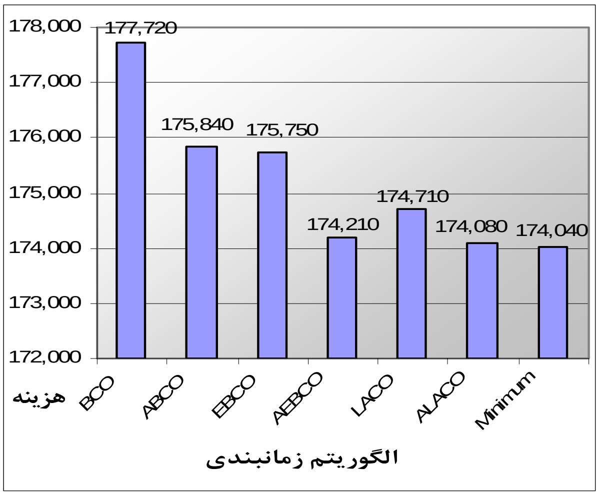
۵- نتايج شبيه‌سازي‌ها

الگوريتم‌هاي پيشنهادي با استفاده از جعبه‌ابزار GridSim [11] شبيه‌سازي شده و نتايج آنها با نتايج بدست آمده از الگوريتم‌هاي مكاشفه‌اي مقايسه شده است. نتايج گزارش شده، ميانگين ٢٠ بار شبيه‌سازي مي‌باشد. محيط شبيه‌سازي شده براي گريد، شامل تعدادي منبع و يک كاربر است. كليه منابع محاسباتي داراي يک پردازنده هستند كه مشخصات آنها در جدول (١) داده شده است. همانطور كه مشاهده مي‌شود، ناهمگوني بالايي براي منابع در نظر گرفته شده است. برنامه كاربر از ٢٠٠ كار مستقل از هم تشكيل شده است كه طول هر كار به صورت تصادفي از محدوده (٢٠٠٠٠٠...١٠٠٠٠٠) انتخاب مي‌شود. گستردگي اين محدوده، منجر به ناهمگوني كارها مي‌گردد. كاربر مقدار ١٢٠٠٠ را براي مهلت زماني تعيين كرده و بهينه‌سازي هزينه را درخواست مي‌كند. براي هر كار، اتوماتاي يادگير از نوع L_{RP} با مقادير $a = b = 0.01$ مي‌باشد.

جدول (١): پيكربندي منابع

نام منبع	نرخ اجرا (MI/sec)	قيمت (G$/sec)	قيمت مفيد (G$/1000MI)
R1	١٠٠	٠,۵	۵
R2	١٨٠	١,٠	۵,۵۵
R3	٢۴٠	١,۵	۶,٢۵
R4	٢٨٠	٢	٧,١۴
R5	٣٠٠	٢,۵	٨,٣٣
R6	۴٠٠	۴	١٠
R7	۵٠٠	۶	١٢
R8	۶٠٠	٩	١۵

در شکل (۴)، الگوريتم‌هاي مكاشفه‌اي BCO، ABCO، EBCO و AEBCO [۱] و الگوريتم‌هاي پيشنهادي LACO و ALACO با يكـديگر مقايسه شده‌اند. همانطور كه مشاهده مي‌شود، از بين دو الگوريتم پيشنهادي، الگوريتم ALACO كمترين هزينه را براي اجراي كارها بدسـت آورده و در مقايسه با بهترين الگوريتم مكاشفه‌اي گزارش شده (AEBCO)، بيش از ۱۰۰ واحد (۰,۱ درصد) كاهش هزينه را نشان مي‌دهد. اختلاف هزينه بدست آمده توسط الگوريتم ALACO با حداقل هزينه مورد نياز براي اجراي كارها بسيار كم است. براي بدست آوردن حـداقل هزينـه مـورد نيـاز براي اجراي كارها، از روش ارائه شده در [۱] استفاده شده است.



شكل (۴): نتايج الگوريتم‌هاي زمانبندي با هدف بهينه‌سازي هزينه

الگوريتم‌هاي پيشنهادي بدليل انجام فرايند يادگيري، در مقايسه با الگوريتم‌هاي مكاشفه‌اي به زمان بيشتري نياز دارند. شبيه‌سـازي‌هـا نـشان داده است كه زمان لازم براي زمانبندي ۲۰۰ كار تقريباً ۱ ثانيه مي‌باشد. اين مقدار زمان در مقابل كاهش هزينه اجراي كارها كه در صورت استفاده از اين الگوريتم‌ها بدست مي‌آيد، قابل ملاحظه نمي‌باشد. همچنين الگوريتم‌هاي مكاشفه‌اي، در شرايط خاص ماننـد نـاهمگوني بـالاي كارهـا، داراي عملكرد مناسبي نمي‌باشند. به همين دليل، استفاده از الگوريتم‌هاي پيشنهادي به منظور زمانبندي كارها در گريد، داراي توجيه اقتصادي مي‌باشـد؛ زيرا كه حتي يک كاهش پايين در هزينه اجراي كارها منجر به جلب مشتريان بيشتر، استفاده بهتر از منابع در گريد و سودآوري بـالاتر بـراي گريـد مي‌شود.

۶- نتيجه‌گيري

در اين مقاله، دو الگوريتم مبتني بر اتوماتاهاي يادگير براي زمانبندي در گريدهاي محاسباتي اقتصادي با مدل بازار كالا بـه منظـور مينـيمم كـردن هزينه در محدوده مهلت تعيين شده، پيشنهاد گرديد. اين الگوريتم‌ها با استفاده از جعبه‌ابزار GridSim شـبيه‌سـازي شـده و كـارايي‌هـاي آنهـا در محيط گريد با منابع و كارهاي ناهمگون، مورد بررسي قرار گرفت. نشان داده شد كه يكي از الگوريتم‌هاي پيشنهاد شده (ALACO) در مقايسـه بـا الگوريتم‌هاي مكاشفه‌اي گزارش شده، از كارايي بالاتري برخوردار بوده و درخواست‌هاي كاربر را با هزينه 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.

[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.

[13] K. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[14] 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 Cost Optimization
[8] Advanced Learning Automata Cost Optimization
[9] Finite State Machine
[10] Variable structure
[11] Linear reward penalty
[12] Linear reward epsilon penalty
[13] Linear reward inaction
[14] Million Instruction
[15] Homogeneous
[16] Resource Discovery
[17] Resource Trading
[18] Admission Control
[19] Dispatching