زمانبندی کارها در محیط گرید محاسباتی با استفاده از الگوریتم Simulated annealing

سعید پارسا*، حمید سعدی†، حمید محمدی‡

چکیده

انگیزه گرید محاسباتی، مجتمع کردن منابع توزیع شده ناهمگون جهت حل مسائل پیچیده علمی، صنعتی و تجاری است. جهت رسیدن به این هدف یک سیستم زمانبندی کارآمد به عنوان یک بخش حیاتی برای گرید لازم است. متاسفانه پویایی و ناهمگونی منابع گرید باعث پیچیدگی زمانبندی وظایف می‌شوند. بعلاوه با معرفی مدل اقتصادی گرید، علاوه بر زمان اتمام کار، هزینه اجرای کار نیز به نگرانی‌های کاربران اضافه شد. اکثر سیستم‌های زمانبندی موجود در محیط گرید، زمان اتمام و هزینه اجرای کارها را به صورت مجزا بهینه می‌کنند. در این مقاله یک سیستم زمانبندی ارائه شده که به کاربر اجازه می‌دهد اهمیت زمان اتمام و اهمیت هزینه اجرای کارهایش را به صورت "وزن زمان" و "وزن هزینه" مشخص کند. بر اساس این دو وزن یک تابع هدف وزن‌دار و دو متغیره برای زمانبند ارائه شده است. زمانبند پیشنهادی از الگوریتم Simulated annealing (SA) برای بهینه سازی انتصاب وظایف به منابع بهره گرفته است. در استفاده از SA زمان و هزینه به همراه وزن‌شان در نظر گرفته شده‌اند. ما نشان دادیم زمانبند SA، زمانبندی بهینه‌تری ایجاد می‌کند.

کلمات کلیدی

گرید محاسباتی، زمانبندی، تابع وزن‌دار شایستگی کارایی، simulated annealing.

Scheduling jobs on computational grid using simulated annealing

Saeed parsa, Hamid saadi, Hamid mohamadi

Abstract

Computational Grids enable the coordinated and aggregated use of geographically distributed resources for solving large-scale problems in science, engineering, and commerce. Achieving high performance in a grid system requires effective resource scheduling. The heterogeneous and dynamic nature of the grid, as well as the differing demands of applications run on the grid, makes grid scheduling complicated. Also, after introduction of grid economical models, the execution cost, besides the completion time, has become of great concern to the grid users. Many of grid scheduling systems optimize completion time and cost separately. In this article a new simulated annealing scheduling algorithm which considers both the completion time and execution cost is introduced. The algorithm applies a weighted objective function that takes into account both the completion time and execution cost of the tasks. It is shown that the simulated annealing scheduler produces a comparatively better result.

Keywords

Computational grid, scheduling, weighted performance fitness function, simulated annealing.

* دانشیار نرم افزار، عضو هیأت علمی دانشگاه، دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر، parsa@iust.ac.ir
† دانشجوی کارشناسی ارشد نرم افزار، دانشگاه علم و صنعت ایران، دانشکده مهندسی کامپیوتر، saadi@comp.iust.ac.ir
‡ دانشجوی کارشناسی ارشد نرم افزار، دانشگاه صنعتی شریف، دانشکده مهندسی کامپیوتر، hamid_m@ce.sharif.edu

۱- مقدمه

گرید محاسباتی زیرساختاری برای استفاده از منابع توزیع شده و ناهمگون مهیا کرده تا مسائل بزرگ علمی، صنعتی و تجاری را حل کند. منابع گرید در مکان‌های مختلف جغرافیایی و توسط سازمانهای مختلف و تحت سیاست‌های مختلفی اداره می‌شوند [1]. امروزه افزایش کارایی گرید یک مساله است. جهت افزایش کارایی گرید به یک زمانبندی درست و کارآمد احتیاج است. متاسفانه طبیعت پویای منابع گرید و همچنین تقاضاهای مختلف کاربران، باعث پیچیدگی مساله زمانبندی گرید شده‌اند. پویایی کارایی منابع ناشی از ناهمگونی، خودمختاری و اشتراکی بودن منابع گرید است. هدف مساله زمانبندی گرید، انتصاب بهینه کارها به منابع است. مطالعه در زمانبندی گرید از این جهت حایز اهمیت است که سیستم‌های دنیای واقعی معمولا به طور فیزیکی یا عملیاتی، توزیع شده و ناهمگون می‌باشند. سیستم‌هایی مثل کنترل ترافیک هوایی، نجوم، پزشکی، زیست شناسی، نظامی و موبایل از این دسته هستند.

بویا در سال ۲۰۰۱ یک چارچوب اقتصادی برای گرید معرفی کرد [4]. به این ترتیب که کاربران گرید به ازای استفاده از منابع، به دارندگان آن منابع هزینه مالی پرداخت می‌کنند. این چارچوب باعث ایجاد انگیزه در دارندگان منابع می‌شود تا منابع خود را به اشتراک بگذارند. مدل اقتصادی بویا باعث اهمیت هزینه و سود اقتصادی از دید کاربران و دارندگان منابع شد. به طوری که از آن پس، در بیشتر الگوریتم‌های زمانبندی، هزینه و سود اقتصادی در تابع هدف زمانبند مدنظر قرار داده می‌شد [9]. اما مشکل این الگوریتم‌ها اینجاست که برای بهبود زمان اتمام کار و هزینه اجرای کار، زمانبند توابع هدف مختلف و سیاست‌های مجزایی را بکار می‌گیرد.

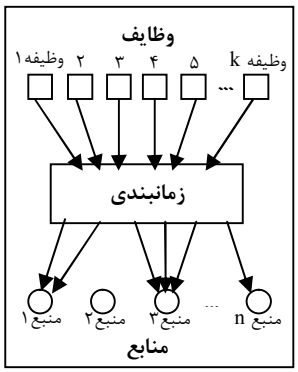
برای بهبود زمانبندی در گرید روشهای مکاشفه‌ای با الهام از قوانین طبیعی پیشنهاد شده‌اند [2,5,9]. این روشها عبارتند از الگوریتم‌های ژنتیکی، Simulated Annealing (SA)، Tabu Search، تئوری بازی و روشهای ترکیبی آنها. در [9] ثابت شده، SA هم در بهبود زمان و هم در بهبود هزینه، نسبت به تئوری بازی راه‌حل بهینه‌تری تولید می‌کند. دلیل آن هم محدودیتهای موجود در تئوری بازی غیر مشارکتی[1] است. در [2] نشان داده‌اند که سرعت اجرای SA از همه الگوریتم‌های فوق بیشتر است. مراجع [2,5] الگوریتم SA را فقط برای بهبود زمان اتمام کارها بکار گرفته‌اند. البته در [9] الگوریتم SA برای بهبود زمان اتمام و هزینه اجرای کارها به صورت مجزا، بکار گرفته شده است. هیچکدام از [2,5,9] برای مواقعی که اهمیت (وزن) هزینه و زمان برای کاربر متفاوت باشد روشی ارائه نکرده‌اند. مشکل دیگر الگوریتم‌های فوق این است که راه‌حل اولیه در آنها یک نگاشت تصادفی از وظایف به منابع است.

از آنجا که ارزانی منبع و سرعت منبع دو فاکتور متضاد هستند، یا به عبارت دیگر منبع سریع معمولا گران است، زمانبند باید در انتصاب وظایف به منابع نظر کاربر را اعمال کند. مثلا اگر کاربر نگران هزینه اجرای کار خود باشد وظایف آن کار را روی یک منبع ارزان منتصب کند و اگر کاربر نگران زمان اتمام کار خود باشد، زمانبند باید وظایف آن کار را روی یک منبع سریع منتصب کند.

زمانبند ما به کاربر آزادی بیشتری برای اعمال نظرش می‌دهد، به این صورت که کاربر می‌تواند ۷۰٪ نگران هزینه و ۳۰٪ نگران زمان اتمام کار خود باشد. ما وزن زمان[2] و وزن هزینه‌ی[3] کارهای کاربر را با استفاده از دو عدد بین ۰ و ۱ که مجموع آنها ۱ است (W_t و W_c)، مدل کرده‌ایم. این دو مقدار توسط کاربر مشخص شده و در تابع هدف زمانبند تاثیر داده شده‌اند. به دلیل NP-complete بودن مساله زمانبندی در گرید، ما از الگوریتم غیر قطعی SA برای بهینه سازی زمانبندی استفاده کرده‌ایم. در استفاده از SA دو فاکتور زمان و هزینه به همراه وزن آنها مدنظر قرار داده شده‌اند، همچنین راه‌حل اولیه در SA پیشنهادی یک راه‌حل تقریبا بهینه است نه یک راه‌حل تصادفی. در نهایت الگوریتم پیشنهادی را با دو الگوریتم که یکی به صورت تصادفی منابع را اختصاص می‌دهد و دیگری از SA استفاده نمی‌کند، مقایسه کرده‌ایم. نتایج این مقایسه که کارا بودن روش پیشنهادی را می‌رساند در بخش ۵ آمده‌اند.

۲- زمانبندی

سیستم مدیریت منابع گرید[4] مسئول کنترل منابع گرید است و هدف آن بالا بردن کارایی سیستم گرید می‌باشد. سرویس اطلاعات گرید[5] یک بخش از سیستم مدیریت منابع گرید است که اطلاعات ایستا و پویای منابع گرید را مهیا می‌کند. زمانبند گرید یک بخش دیگر از سیستم مدیریت منابع گرید است. زمانبند با استفاده از اطلاعات منابع، که توسط سرویس اطلاعات گرید مهیا می‌شوند، و اطلاعات کارها و وظایف آنها، یک انتصاب از وظایف به منابع ایجاد می‌کند. به این انتصاب زمانبندی گفته می‌شود. مفهوم زمانبندی در شکل زیر نشان داده شده است. یک زمانبندی کارا باعث کاهش زمان اتمام و کاهش هزینه اجرای کارها می‌شود.



شکل(۱): زمانبندی وظایف در محیط گرید

موضوع مساله زمانبندی گرید انتصاب یک مجموعه از وظایف به یک مجموعه از منابع است. ما یک مجموعه از k وظیفه

$T=\{t_1,t_2,...,t_k\}$ و یک مجموعه از n منبع $R=\{r_1,r_2,...,r_n\}$ را فرض کردیم. مجموعه T شامل همه کارهایی است که باید زمانبندی شوند. این کارها قبلا توسط کاربر به k وظیفه تقسیم شده‌اند. هدف زمانبندی، تولید یک نگاشت انتصاب از وظایف به منابع است. همانطور که در شکل (۱) مشاهده می‌شود، در این انتصاب چندین وظیفه ممکن است به یک منبع منتصب شوند.

۳- Simulated Annealing

Simulated Annealing یک روش جستجوی محلی است که جهت یافتن جواب بهینه کلی برای یک مساله پیچیده بکار می‌رود. ایده اولیه این روش در سال ۱۹۵۳ مطرح شد [3]. در این روش ابتدا یک شیی تا دمای بالایی حرارت داده می‌شود. سپس به آهستگی سرد می‌شود به طوری که سیستم تقریباً در هر زمان در تعادل ترمودینامیکی قرار داشته باشد. در حالت تعادل، شیی پیکربندی‌های زیادی دارد که برای هر کدام یک شایستگی مشخص می‌شود. از پیکربندی فعلی به پیکربندی بعدی یک آشفتگی تصادفی اعمال می‌شود و شایستگی متناظر با آن محاسبه می‌شود.

فرض کنید F_c و F_n شایستگی فعلی و شایستگی جدید را نمایش دهند. اگر $F_n < F_c$ آنگاه پیکربندی جدید جایگزین پیکربندی فعلی می‌شود، در غیر این صورت پیکربندی جدید با احتمال $\exp(-(F_n-F_c)/T)$ جایگزین پیکربندی فعلی می‌شود. T در این رابطه، دمای مربوط به آن تعادل است. بنابراین احتمال پذیرفته شدن یک پیکربندی بدتر هنگامی که T بالا باشد بیشتر و هنگامی که T پایین باشد کمتر است. دلیل پذیرفتن پیکربندیهای بدتر، فرار از ماکزیمم و مینیمم‌های نسبی است. به دلیل کاهش تدریجی دما و پذیرفته شدن پیکربندی‌های بهتر در دماهای پایین‌تر، این الگوریتم به سمت بهترین پیکربندی همگرا می‌شود [6].

در هر دما بهترین پیکربندی که تا کنون دیده شده و شایستگی آن (F_b) نگهداری می‌شود. اگر $F_n < F_b$ آنگاه پیکربندی جدید به عنوان بهترین پیکربندی پذیرفته می‌شود. با کاهش تدریجی دما تا هنگامی که بهبود بیشتری ممکن نباشد پیکربندی‌های جدیدتری بدست خواهند آمد. در نهایت بهترین پیکربندی، خروجی الگوریتم SA است. از سال ۱۹۸۳ به بعد، SA وارد مسائل بهینه سازی شد [6].

۴- طرح پیشنهادی

در این بخش یک زمانبند وظایف گرید با بهینه کردن زمان و هزینه بر اساس وزن مشخص شده توسط کاربر، پیشنهاد شده است. برای بهینه سازی از الگوریتم SA استفاده شده است.

۴-۱- تابع هدف زمانبند

هنگام معرفی وظایف بعضی از کاربران نگران زمان اتمام و بعضی دیگر نگران هزینه اجرای کارهای خود هستند. به همین علت در روش پیشنهادی، کاربر هنگام معرفی کارهای خود می‌تواند وزن‌های زمان اتمام و هزینه اجرای آن کارها (W_t و W_c) را مشخص کند. این دو وزن در محدوده [۱و۰] بوده و مجموع آنها ۱ است. مثلا اگر برای یک کار W_c برابر ۰/۷ باشد به این معناست که کاربر ۷۰٪ نگران هزینه مالی اجرا و ۳۰٪ نگران زمان اتمام آن کار است. پس زمانبند باید منابعی برای اجرای وظایف آن کار پیدا کند که آن منابع هزینه را ۷۰٪ و زمان را ۳۰٪ بهبود بخشند. معمولا بیشتر کاربران گرید نگران هزینه اجرای کارهای خود هستند.

هر زمانبند برای انجام زمانبندی از یک تابع شایستگی کارایی[۶] (PFF) استفاده می‌کند. زمانبند با استفاده از PFF می‌تواند شایستگی منابع مختلف برای اجرای یک وظیفه را معین کند. ما از یک تابع وزن‌دار و دو متغیره برای این منظور استفاده کرده‌ایم. تابع پیشنهادی ما از دو پارامتر زمان و هزینه تشکیل شده است. در این تابع وزن زمان اتمام و وزن هزینه اجرای کار (W_t و W_c) از دید کاربر، در نظر گرفته شده‌اند. زمان و هزینه از یک جنس نبوده و در یک محدوده نیستند، برای رفع این مشکل آنها را در محدوده صفر تا یک قرار داده و در اصطلاح نرمال کرده‌ایم. تابع هدف به شرح ذیل است:

$$PFF\ (t_i,r_j) = w_{ti}\times\frac{t_{i,j}-t_{\min}}{t_{\max}-t_{\min}} + w_{ci}\times\frac{c_{i,j}-c_{\min}}{c_{\max}-c_{\min}} \qquad (۱)$$

تابع هدف فوق میزان شایستگی منبع j ام را برای اجرای وظیفه i ام مشخص می‌کند. همانطور که در تابع فوق مشاهده می‌شود نرمال شده زمان و هزینه در وزن آنها ضرب شده است. در تابع فوق $t_{\max}$ و $t_{\min}$ حداکثر و حداقل زمان اتمام وظیفه بر روی منابع مناسب است. منابع مناسب برای اجرای وظیفه در بخش ۴-۲ شرح داده شده‌اند. $c_{\max}$ و $c_{\min}$ حداکثر و حداقل هزینه اجرای وظیفه بر روی منابع مناسب است. $t_{i,j}$ و $c_{i,j}$ زمان اتمام و هزینه اجرای وظیفه i ام بر روی منبع j ام است.

باید متذکر شد که ضرایب وزن، توسط کاربر در حین معرفی کار مشخص می‌شوند. W_t و W_c باعث می‌شوند که کاربر آزادی بیشتری در معرفی کارهای خود به گرید داشته باشد. هدف زمانبند این است که برای هر وظیفه x یک منبع y را پیدا کند به طوری که $PFF(t_x,r_y)$ مینیمم باشد. طریقه یافتن این منبع در بخش ۴-۲ آمده است.

۴-۲- زمانبند

اولین عملی که توسط زمانبند در زمانبندی وظایف گرید باید صورت گیرد، مشخص کردن مجموعه منابع مناسب برای اجرای هر وظیفه است. منابعی برای اجرای وظیفه مناسب هستند که:

- سیستم عامل و معماری مورد نیاز برای اجرای وظیفه را دارا باشند.

- هزینه تخمینی اجرای وظیفه در آنها از هزینه‌ای که کاربر برای اجرای آن وظیفه مشخص کرده، بیشتر نباشد.
- کاربر مجوز استفاده از آن منابع را داشته باشد.

پس از مشخص کردن مجموعه منابع مناسب برای اجرای هر وظیفه، زمانبند باید بهترین منبع را از بین این مجموعه مشخص کند. بهترین منبع همان منبعی است که تابع (۱) به ازای آن مینیمم باشد. مجموعه منابع مناسب ممکن است در یکی از دو حالت زیر باشند، که در هر کدام از حالت‌ها زمانبند سیاست‌های مختلفی را بکار می‌گیرد. این سیاست‌ها در زیر شرح داده شده‌اند.

الف: پردازنده‌های همه منابع مناسب برای اجرای وظیفه مشغول هستند. در این حالت زمانبند وظیفه را در انتهای صف انتظار بهترین منبع مناسب قرار می‌دهد. سپس الگوریتم SA را فراخوانی می‌کند تا با جابه جا کردن وظایف موجود در صف انتظار منابع مختلف، بهترین زمانبندی حاصل شود. الگوریتم SA پیشنهادی در بخش ۴-۳ شرح داده شده است.

ب: بعضی از منابع مناسب دارای پردازنده آزاد هستند. در این حالت زمانبند ابتدا وظایف موجود در صف انتظار منابع مختلف را به پردازنده‌های آزاد آنها (در صورت وجود) منتصب می‌کند. این عمل تا زمانی که صف انتظار یک منبع خالی نباشد و پردازنده آزاد در آن منبع موجود باشد ادامه می‌یابد. حال در صورتی که پردازنده آزاد روی بعضی از منابع مناسب، باقی بماند (به شرط خالی بودن صف انتظار آن منابع)، زمانبند وظیفه را روی بهترینِ آنها منتصب می‌کند. در غیر این صورت، زمانبند وظیفه را در انتهای صف انتظار بهترین منبع مناسب قرار می‌دهد. سپس الگوریتم SA را فراخوانی می‌کند تا با جابه‌جا کردن وظایف موجود در صف انتظار منابع مختلف، بهترین زمانبندی حاصل شود.

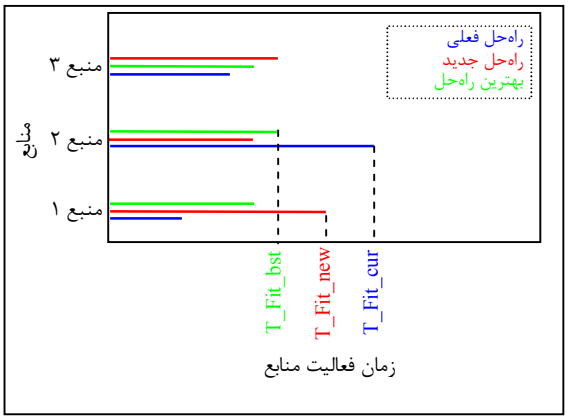
اگر یکی از منابع پردازش یک وظیفه را به اتمام برساند، پردازنده منتصب شده به آن وظیفه آزاد می‌شود. زمانبند پردازنده آزاد شده را به اولین وظیفه موجود در صف انتظار آن منبع اختصاص می‌دهد. در این صورت پس از اتمام هر وظیفه یک وظیفه جدید، در صورت وجود، منتصب می‌شود و از آنجا که الگوریتم SA پیشنهادی ما باعث توازن صفهای منابع مختلف می‌شود، دیگر نگران خالی بودن یک صف و شلوغی صف‌های دیگر نیستیم. در حقیقت با استفاده از SA هم باعث کاهش زمان اتمام و هزینه اجرای کارها و هم باعث افزایش بهره‌وری منابع می‌شود.

۴-۳- الگوریتم SA پیشنهادی

برای بهبود زمان اتمام و هزینه اجرای وظایف در گرید الگوریتم نوین SA در این بخش ارائه شده است. این الگوریتم در شکل‌های (۳) و (۴) نشان داده شده و روند اجرای آن نیز در بخش ۴-۳-۱ آمده است.

در این الگوریتم زمان و هزینه از وزن‌های مختلفی برخوردارند. هر پیکربندی در SA معادل یک نگاشت از وظایف به منابع است که در واقع یک راه‌حل مساله زمانبندی است. در هر کدام از سه راه‌حل فعلی، جدید و بهترین، دو شایستگی یکی برای زمان و یکی برای هزینه در نظر گرفته می‌شود. هدف الگوریتم SA کاهش مجموع شایستگی زمان و شایستگی هزینه زمانبندی است. البته این دو شایستگی در میانگین وزن‌های زمان و هزینه وظایف موجود در زمانبندی ضرب می‌شوند.

همانگونه که در شکل (۲) مشاهده می‌شود، شایستگی زمان در هر راه‌حل، حداکثر زمان پایان فعالیت منابع در آن راه‌حل است. زمان فعالیت هر منبع مجموع زمان اتمام وظایف موجود در صف انتظار آن منبع یعنی مجموع زمان اجرا و زمان مصرف شده جهت زمانبندی وظایف آن منبع است.



شکل (۲): شایستگی زمان در راه‌حل‌های مختلف الگوریتم SA

شایستگی هزینه در هر راه‌حل، مجموع کل هزینه اجرای وظایف موجود در صف انتظار، در آن راه‌حل است. شایستگی کلی در هر راه‌حل ترکیبی از شایستگی زمان و شایستگی هزینه در آن راه‌حل است. اما نمی‌توان آن را به راحتی مشخص کرد چون نرمال کردن زمان و هزینه پیچیده می‌شود.

```
Algorithm Simulated_ Annealing (S0, Tmax , Tmin , k);
Begin
  T = Tmax ;   // initial temperatu re
  CurS = S0;   // initial solution
  BstS = Cur S; //BstS is the best solution seen so far
  T_Fit_cur = TimeFitn ess(CurS); //time fitness of current solution
  C_Fit_cur = CostFitn ess(CurS); //cost fitness of current solution
  T_Fit_bst = TimeFitn ess(BstS); //time fitness of best solution
  C_Fit_bst = CostFitn ess(BstS); //cost fitness of best solution
  Repeat
        Call Metropolis (CurS, T_Fi t_cur, C_F it_cur,
                         BstS, T_F it_bst, C_ Fit_bst, T , k);
        T = α × T  // α is cool rate
  Until (T > Tmin ); // Tmin is final temperatu re
  Return (BestS );
End.
```

شکل (۳): الگوریتم Simulated annealing

```
Algorithm Metropolis (CurS, T_Fit_cur, C_Fit_cur, BstS, T_Fit_bst, C_Fit_bst, T, k);
Begin
  Repeat
    NewS = Neighbor (CurS); // new solution
    T_Fit_new = TimeFitness (NewS); //time fitness of new solution
    C_Fit_new = CostFitness (NewS); //cost fitness of new solution
    Δfitness_cur = avg(w_t) × (T_Fit_new − T_Fit_cur) / Max(T_Fit_new, T_Fit_cur) + avg(w_c) × (C_Fit_new − C_Fit_cur) / Max(C_Fit_new, C_Fit_cur)
    Δfitness_bst = avg(w_t) × (T_Fit_new − T_Fit_bst) / Max(T_Fit_new, T_Fit_bst) + avg(w_c) × (C_Fit_new − C_Fit_bst) / Max(C_Fit_new, C_Fit_bst)
    If (Δfitness_cur > 0) then
      CurS = NewS;
      T_Fit_cur = T_Fit_new;
      C_Fit_cur = C_Fit_new;
      If (Δfitness_bst > 0) then
        BstS = NewS;
        T_Fit_bst = T_Fit_new;
        C_Fit_bst = C_Fit_new;
      End if;
    Else if (Random < e^(−Δfitness_cur / T)) then
      CurS = NewS;
      T_Fit_cur = T_Fit_new;
      C_Fit_cur = C_Fit_new;
    End if;
    k = k-1;    // k is the amount time for which annealing …
  Until (k=0);  // must be applied at temperature T.
End. //of Metropolis
```

شکل (۴): الگوریتم Metropolis

برای رفع مشکل نرمال کردن زمان و هزینه در الگوریتم SA، مستقیما $\Delta fitness_cur$ و $\Delta fitness_bst$ را محاسبه کردیم. $\Delta fitness_cur$ اختلاف شایستگی راه‌حل جدید و راه‌حل فعلی است. $\Delta fitness_bst$ اختلاف شایستگی راه‌حل جدید و بهترین راه‌حل است. همانطور که در شکل (۴) مشاهده می‌شود در محاسبه این دو Δ، میانگین وزن زمان اتمام و وزن هزینه اجرای وظایف در نرمال شده اختلاف شایستگی زمان و هزینه ضرب شده است. ما از این دو Δ برای قبول یا رد راه‌حل جدید به عنوان راه‌حل فعلی یا به عنوان بهترین راه‌حل، استفاده کرده‌ایم.

۴-۳-۱- روند اجرای الگوریتم SA

باید متذکر شد که الگوریتم SA پیشنهادی، وظایف موجود در صفهای انتظار منابع مختلف را جا به جا می‌کند تا به یک زمانبندی بهینه دست یابد. این الگوریتم به صورت زیر عمل می‌کند:

۱- در ابتدا یک راه‌حل تقریبا بهینه داریم. در این راه‌حل هر وظیفه درون صف انتظار بهترین منبع مناسب قرار گرفته است. یعنی منبعی که تابع (۱) به ازای آن مینیمم باشد.

۲- الگوریتم از دمای $T_{\max}$ شروع می‌شود.

۳- راه‌حل فعلی به عنوان بهترین راه‌حل در نظر گرفته می‌شود.

۴- شایستگی زمان و هزینه برای راه‌حل فعلی و بهترین راه‌حل محاسبه می‌شود.

۵- راه‌حل جدید به روش زیر ایجاد می‌شود. متد $Neighbor(CurS)$:

۵-۱ در راه‌حل فعلی دو منبع مختلف به تصادف انتخاب می‌شوند.

۵-۲ در صورتی که صف انتظار منبع اول خالی نباشد یک وظیفه به صورت تصادفی از صف انتظار منبع اول حذف می‌شود. این وظیفه ممکن است در هر مکانی از صف قرار داشته باشد.

۵-۳ وظیفه حذف شده، به انتهای صف انتظار منبع دوم اضافه می‌شود.

۶- شایستگی زمان و هزینه راه‌حل جدید محاسبه می‌شود.

۷- $\Delta fitness_cur$ محاسبه می‌شود. این مقدار اختلاف شایستگی راه‌حل جدید و راه‌حل فعلی است.

۸- اگر $\Delta fitness_cur$ مثبت باشد، به معنی بهینه‌تر بودن راه‌حل جدید از راه‌حل فعلی است. پس راه‌حل جدید را جایگزین راه‌حل فعلی می‌کنیم.

۹- $\Delta fitness_bst$ محاسبه می‌شود. این مقدار اختلاف شایستگی راه‌حل جدید و بهترین راه‌حل است.

۱۰- اگر $\Delta fitness_bst$ مثبت باشد، به معنی بهینه‌تر بودن راه‌حل جدید از بهترین راه‌حل است. پس راه‌حل جدید را جایگزین بهترین راه‌حل می‌کنیم.

۱۱- جهت فرار از راه‌حل‌های بهینه نسبی، در صورتی که $\Delta fitness_cur$ مثبت نباشد راه‌حل جدید را با احتمال $\exp(-(\Delta fitness_cur/T))$ جایگزین راه‌حل فعلی می‌کنیم. این احتمال با دمای فعلی رابطه مستقیم دارد. یعنی هرچه دما پایین‌تر باشد احتمال پذیرفتن راه‌حل‌های جدیدی که بهینه نیستند کمتر می‌شود. در الگوریتم SA دما به تدریج پایین می‌آید.

۱۲- مراحل ۵ تا ۱۱، k بار تکرار می‌شوند.

۱۳- دما در نرخ سردی ضرب شده و دمای جدید تولید می‌شود.

۱۴- مراحل ۵ تا ۱۳، تا دمای $T_{\min}$ تکرار می‌شوند.

۱۵- بهترین راه‌حل، خروجی الگوریتم است.

۱۶- پایان الگوریتم.

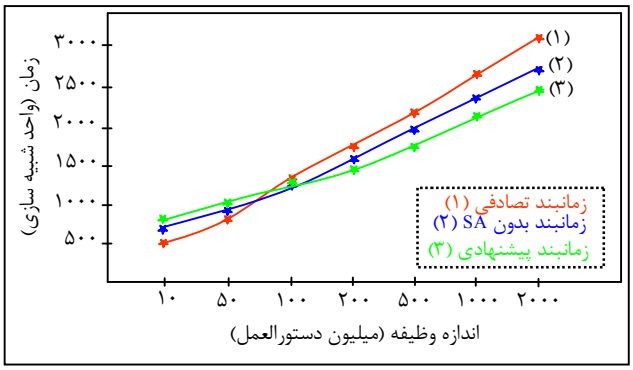
۵- ارزیابی نتایج

در این بخش نتایج حاصل از بکارگیری الگوریتم پیشنهادی برای زمانبندی وظایف مستقل بر روی شبکه گرید محاسباتی ارائه شده است. از آنجا که تابع هدف زمانبند پیشنهادی یک تابع وزن‌دار و دو متغیره است، نتایج حاصل از زمانبند پیشنهادی را هم از لحاظ زمان و هم از لحاظ هزینه، با نتایج دو زمانبند دیگر مقایسه کرده‌ایم. زمانبند اول به طور تصادفی منابع را اختصاص می‌دهد و زمانبند دوم همان زمانبند پیشنهادی، بدون استفاده از الگوریتم SA است. در شکل‌های (۵) تا (۸) زمانبند اول با شماره (۱)، زمانبند دوم با شماره (۲) و زمانبند پیشنهادی با شماره (۳) مشخص شده‌اند. تفاوت نتایج زمانبندهای (۲) و (۳) تاثیر الگوریتم SA را در بهبود زمانبند پیشنهادی می‌رساند.

از آنجا که روش ما غیر قطعی بوده و نتایج هر اجرا با اجرای قبلی می‌تواند تا حدی متفاوت باشد، جهت اطمینان از نتایج، هر سه زمانبند را حداقل ۳۰ بار اجرا کردیم و میانگین نتایج اجراها را درنظر گرفتیم. انحراف از میانگین در اجراهای مختلف نیز ۴/۷ درصد محاسبه شد.

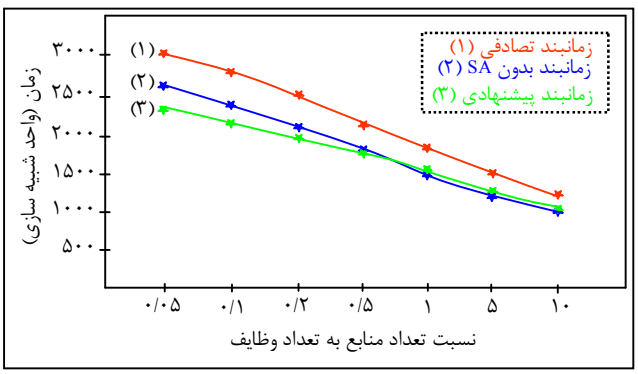
در بحث زمان، هدف کاهش زمان کلی زمانبندی است. هر وظیفه ابتدا باید زمانبندی و سپس اجرا شود. زمان اتمام هر وظیفه از ارائه آن تا اتمام اجرای آن طول می‌کشد. زمانبند سعی در کاهش زمان اجرای وظیفه دارد ولی ممکن است زمان مصرف شده جهت زمانبندی آن را افزایش دهد. جهت ارزیابی زمانبندهای مختلف از لحاظ زمان، زمان اتمام وظیفه در نظر گرفته شده است. زمان اتمام وظیفه مجموع زمان مصرف شده برای زمانبندی وظیفه و زمان اجرای آن است.

شکل (۵) و (۶) تفاوت زمانبندها از لحاظ زمان کلی، را در دو وضعیت نشان می‌دهند. شکل (۵) تاثیر اندازه وظایف در زمان کلی زمانبندی را نشان می‌دهد. همانطور که در شکل (۵) مشاهده می‌شود هنگامی که اندازه وظایف پایین باشد زمانبند (۱) بهتر عمل می‌کند. زیرا زمان مصرف شده برای زمانبندی وظیفه با اندازه پایین، در زمانبندهای (۲) و (۳) بیشتر از زمان اجرای آن است، و از آنجا که زمان مصرف شده برای زمانبند (۱) بسیار ناچیز است، این زمانبند برای وظایف کم اندازه بهتر عمل می‌کند. اما با افزایش اندازه وظیفه، زمانبند (۲) و در نهایت زمانبند (۳) عملکرد بهتری خواهد داشت. علت نتایج فوق این است که زمان مصرف شده برای زمانبندی در زمانبند (۳) بیشتر از زمانبند (۲) و در زمانبند (۲) بیشتر از زمانبند (۱) است. به همین علت زمانبند (۳) برای مواقعی که اندازه وظیفه زیاد باشد، یعنی زمان اجرای آن بالا باشد، بهتر عمل می‌کند. معمولا وظایف گرید از اندازه بالایی برخوردارند.



شکل (۵): تاثیر اندازه وظایف بر زمان کلی زمانبندی

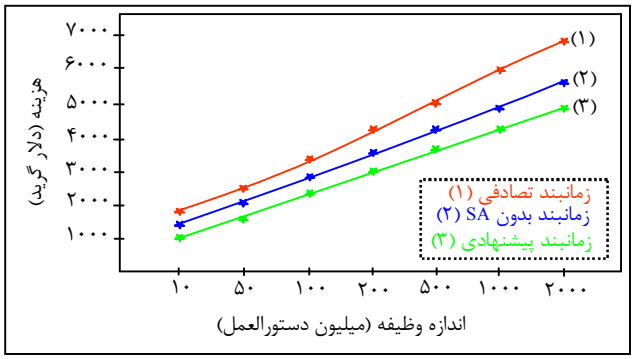
شکل (۶) تاثیر نسبت تعداد منابع به تعداد وظایف در زمان کلی زمانبندی را نشان می‌دهد. همانطور که در شکل (۶) مشاهده می‌شود هنگامی که تعداد منابع کمتر از تعداد وظایف باشد زمانبند (۳) بهتر عمل می‌کند. اگر تعداد منابع بیشتر از تعداد وظایف باشد، عملکرد زمانبندهای (۲) و (۳) تقریبا یکسان است. علت این است که با افزایش تعداد منابع، صف انتظار آنها تقریبا خالی از وظایف می‌شود و اگر صف انتظار منابع خالی باشد الگوریتم SA هیچ بهبودی در زمان کلی زمانبندی ایجاد نمی‌کند و باعث می‌شود که عملکرد زمانبند (۳) شبیه زمانبند (۲) شود. در گرید تعداد منابع محدود بوده و تعداد آنها از تعداد وظایف کمتر است.



شکل (۶): تاثیر نسبت تعداد منابع به تعداد وظایف بر زمان کلی زمانبندی

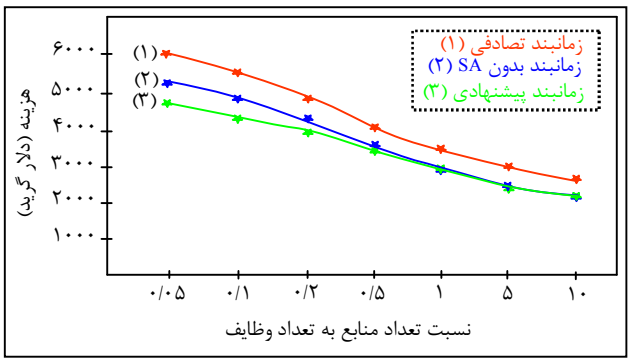
در بحث هزینه، هدف کاهش هزینه زمانبندی وظایف است. هزینه اجرای هر وظیفه از ضرب "زمان اجرای آن روی منبع اختصاص داده شده" در "بهای آن منبع به ازای یک واحد زمان پردازش" بدست می‌آید. هزینه کلی زمانبندی، مجموع هزینه اجرای کل وظایف است.

شکلهای (۷) و (۸) تفاوت زمانبندها از لحاظ هزینه کلی زمانبندی را در دو وضعیت نشان می‌دهند. در شکل (۷) تاثیر اندازه وظایف در هزینه کلی زمانبندی نشان داده شده است. همانطور که در شکل (۷) مشاهده می‌شود زمانبند (۳) در هر اندازه‌ای از وظایف، عملکرد بهتری دارد. علت این است که در هزینه پردازش وظایف فقط زمان اجرای آنها مهم است و زمان مصرف شده برای زمانبندی در نظر گرفته نمی‌شود و از آنجا که زمانبند (۳) زمان اجرای وظایف را کاهش می‌دهد باعث کاهش هزینه زمانبندی نیز می‌شود.



شکل (۷): تاثیر اندازه وظایف بر هزینه کلی زمانبندی

شکل (۸) تاثیر نسبت تعداد منابع به تعداد وظایف در هزینه کلی زمانبندی را نشان می‌دهد. همانطور که در شکل (۸) مشاهده می‌شود هنگامی که تعداد منابع کمتر از تعداد وظایف باشد زمانبند (۳) عملکرد بهتری دارد. ولی اگر تعداد منابع بیشتر از تعداد وظایف باشد هزینه زمانبندی تولید شده توسط زمانبندهای (۲) و (۳) تقریبا یکسان است. علت این است که با افزایش تعداد منابع صف انتظار آنها خالی از وظایف شده و الگوریتم SA بهبودی در زمان اجرا و هزینه اجرای وظایف حاصل نمی‌کند و باعث می‌شود که عملکرد زمانبند (۳) شبیه عملکرد زمانبند (۲) شود. متذکر می‌شویم که تعداد منابع در گرید محدود بوده و تعداد آنها از تعداد وظایف کمتر است.



شکل (۸): تاثیر نسبت تعداد منابع به تعداد وظایف بر هزینه کلی زمانبندی

۶- نتیجه‌گیری

ما در این مقاله یک الگوریتم برای زمانبندی وظایف مستقل از هم در گرید محاسباتی ارائه دادیم. الگوریتم پیشنهادی به کاربر آزادی بیشتری برای مشخص کردن اهمیت زمان و اهمیت هزینه کارهای خود می‌دهد. ما یک تابع هدف وزن‌دار برای زمانبند پیشنهادی با در نظر گرفتن اهمیت زمان و اهمیت هزینه کارهای کاربر ارائه دادیم. هم زمان اتمام و هم هزینه اجرای وظایف در این تابع مدنظر قرار داده شده‌اند. هدف زمانبند پیشنهادی کمینه کردن این تابع برای وظایف مختلف کارهای کاربران است. برای بهبود زمانبندی وظایفی که منبع آزاد برای آنها موجود نمی‌باشد و در صف انتظار منابع مختلف قرار می‌گیرند از الگوریتم Simulated Annealing استفاده کردیم. در استفاده از این الگوریتم همانند تابع هدف، زمان و هزینه به همراه وزن آنها از دید کاربر، در نظر گرفته شده‌اند.

زمانبند پیشنهادی را با دو زمانبند دیگر که یکی وظایف را به صورت تصادفی به منابع اختصاص می‌دهد و دیگری از الگوریتم SA استفاده نمی‌کند، مقایسه کردیم و نشان دادیم که اگر تعداد منابع محدود و اندازه وظایف زیاد باشد، که در واقعیت اینگونه است، زمانبند پیشنهادی در کاهش زمان و هزینه کل زمانبندی، بهترین عملکرد را نسبت به دو زمانبند دیگر دارد. در آینده می‌توان این زمانبند را برای زمانبندی وظایف وابسته به هم بسط داد، همچنین در محاسبه هزینه می‌توان هزینه شبکه را به حساب آورد.

مراجع

[1] I. Foster and C. Kesselman (editors), *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann Publishers, USA, 1999.

[2] R. Braun, H. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. Reuther, J. Robertson, M. Theys, B. Yao, D. Hensgen and R. Freund, *A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems*, in J. of Parallel and Distributed Computing, vol.61, No. 6, pp. 810-837, 2001.

[3] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, "*Equation of State Calculations by Fast Computing Machines*", J. Chem. Phys., vol. 21, no. 6, pp. 1087-1092, 1953.

[4] R. Buyya, J. Giddy, D. Abramson, *A case for economy grid architecture for service-oriented grid computing*, in: 10th IEEE Internat. Heterogeneous Computing Workshop (HCW 2001), San Francisco, CA, April 2001.

[5] A. Abraham, R. Buyya and B. Nath, *Nature's Heuristics for Scheduling Jobs on Computational Grids*, in Proc. of 8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000), pp. 45-52, Cochin, India, December 2000.

[6] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "*Optimization by Simulated Annealing*", Science, 220, 4598, pp. 671-680, 1983.

[7] M, Arora, S.K. Das, R. Biswas, *A Decentralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments*, in Proc. of International Conference on Parallel Processing

Workshops (ICPPW'02), pp.:499 – 505, Vancouver, British Columbia Canada, August 2002.

[8] D. P. Silva, W. Cirne and F. V. Brasileiro, *Trading Cycles for Information: Using Replication to Schedule Bag-of-Tasks Applications on Computational Grids*, in Proc of Euro-Par 2003, pp.169-180, Klagenfurt, Austria, August 2003.

[9] L. Young, S. McGough, S. Newhouse, and J. Darlington, *Scheduling Architecture and Algorithms within the ICENI Grid Middleware*, in Proc. of UK e-Science All Hands Meeting, pp. 5-12, Nottingham, UK, September 2003.

زیرنویس‌ها

[1] Uncooperative game theory
[2] Time weight
[3] Cost weight
[4] Grid Resource Management System
[5] Grid Information Service
[6] Performance Fitness Function