انجمن کامپیوتر ایران
Computer Society of Iran

مرکز توسعه فناوری نیرو (متن)

زمان‌بندی بار محاسباتی تقسیم‌پذیر با در نظرگرفتن زمان بازگشت نتایج در سیستم‌های ناهمگن با استفاده از الگوریتم‌های تپه‌نوردی و شبیه‌سازی حرارتی

رضا منصفی،
عضو هیئت علمی دانشگاه،
دانشگاه فردوسی مشهد، دانشکده مهندسی، گروه کامپیوتر
monsefi@um.ac.ir

جواد حمیدزاده،
دانشجوی دکتری کامپیوتر،
دانشگاه فردوسی مشهد، دانشکده مهندسی، گروه کامپیوتر و عضو هیئت علمی موسسه آموزش عالی سجاد
Ja_ha47@stu-mail.um.ac.ir

چکیده: امروزه مسئله زمان‌بندی کارها در سیستم‌های ناهمگن به دلیل لزوم استفاده بهینه از ماشین‌های محاسباتی موجود و همچنین صرف زمان کمتر برای اجرای الگوریتم‌های زمان‌بندی، از اهمیت خاصی برخوردار است. در این مقاله زمان‌بندی بار محاسباتی تقسیم‌پذیر با درنظرگرفتن زمان بازگشت نتایج در یک سیستم ناهمگن دارای شبکه ارتباطی ستاره‌ای بررسی شده است. یکی از اهداف زمان‌بندی در این‌گونه سیستم‌ها، کمینه‌سازی زمان کل پاسخ است. تاکنون الگوریتمی معین با پیچیدگی زمانی چند جمله‌ای که بتواند در تمام حالت‌ها جواب بهینه را تولید کند، برای این منظور ارائه نشده است. این مسئله مانند مسائل ترکیباتی، پیچیده به نظر می‌رسد و راه حل‌های موجود برای آن، راه حل‌های ابتکاری است. در این مقاله الگوریتم‌های تپه‌نوردی و شبیه‌سازی حرارتی و ترکیب آنها با الگوریتم ژنتیک به عنوان راه حل‌های مسئله پیشنهاد شده است. با انجام شبیه‌سازی و مقایسه نتایج مشاهده می‌شود که این راه حل‌ها، در مقایسه با سایر روش‌های موجود جواب‌های بهتری تولید می‌کنند. در میان روش‌های موجود، الگوریتم‌های پیشنهادی از میانگین کل درصد خطای نسبی کمتری برخوردار هستند.

واژه‌های کلیدی: زمان‌بندی، بار محاسباتی تقسیم‌پذیر، زمان بازگشت نتایج، سیستم‌های ناهمگن، الگوریتم تپه‌نوردی، الگوریتم شبیه‌سازی حرارتی، الگوریتم‌های ترکیبی.

۱- مقدمه

امروزه کارهای[۱] محاسباتی حجیم و زمان‌بندی اجرای آنها در سیستم‌های ناهمگن[۲] بسیار مورد توجه قرار گرفته است. در این گونه سیستم‌ها، کارایی ماشین‌ها، هم‌بندی شبکه ارتباطی و سرعت خطوط ارتباطی شبکه می‌تواند متفاوت باشد بطوری که یک سیستم ناهمگن تشکیل شود. مسئله تطبیق و زمان‌بندی کارها در این سیستم‌ها، از جمله مسائل سخت[۳] است. یکی از مدل‌های محاسباتی مطرح در این سیستم‌ها، مدل بار محاسباتی تقسیم‌پذیر[۴] است. تئوری این دسته از محاسبات برای اولین بار در سال ۱۹۸۸ میلادی توسط آقای «ربرتازی» در مقاله [۷] معرفی شد که با چاپ کتاب [۲] توسط ایشان و همکاران، باعث توجه بیشتر محققین به این تئوری شد.

در تئوری بار محاسباتی تقسیم‌پذیر، بار محاسباتی از تعداد زیادی واحدهای محاسباتی[۵] ریزدانه[۶] تشکیل شده است که پردازش این واحدها می‌تواند به طور کاملاً مستقل و موازی با هم انجام گیرد. فرض بر این است که این واحدهای محاسباتی قابل تقسیم نباشند. درنتیجه، حجم زیادی از محاسبات، می‌تواند به هر نسبتی برحسب واحدهای محاسباتی ریزدانه تقسیم شده و سپس برای اجرای موازی به ماشین‌های موجود ارسال شود.

این تئوری ابزار مناسبی جهت مدل‌سازی کاربردهای بسیاری با داده‌های حجیم[۷] است [۴و۱۵]. از جمله کاربردهایی که در این دسته از محاسبات می‌توان به آنها اشاره نمود عبارتند از: پردازش تصویر[۸]، پردازش سیگنال، جستجو در بانک‌های اطلاعاتی[۹]، داده‌کاوی[۸]، محاسبات جبرخطی [۱۰] و کاربردهای چند رسانه‌ای[۱۱].

در این تئوری، از مدل ارباب و کارگر[۹] جهت پیاده‌سازی استفاده می‌شود. کل بار محاسباتی در این تئوری بر روی کامپیوتر ارباب قرار دارد که از طریق یک شبکه ارتباطی دارای هم‌بندی مشخص، با کامپیوترهای کارگر در ارتباط است. کامپیوتر ارباب موظف است بار محاسباتی تقسیم‌پذیر را بین کامپیوترهای کارگر تقسیم نماید و کامپیوترهای کارگر تا دریافت کامل سهم بار محاسباتی خود منتظر مانده و سپس پردازش خود را آغاز نمایند. هر کامپیوتر کارگر پس از اتمام پردازش، موظف است طی ترتیبی که در زمان‌بندی مشخص شده نتیجه را به کامپیوتر ارباب بازگرداند. هدف از زمان‌بندی در این سیستم، تعیین سهم هر کامپیوتر کارگر، ترتیب توزیع سهم‌ها و ترتیب بازگشت نتایج پردازش شده به کامپیوتر ارباب است، به‌گونه‌ای که زمان کل پاسخ[۱۰] کمینه شود.

در اغلب تحقیقات انجام شده در این زمینه، حجم نتایج حاصل از پردازش در کامپیوترهای کارگر بقدری کم فرض شده است که می‌توان از در نظرگرفتن تاخیر زمانی لازم جهت ارسال این داده‌ها

به کامپیوتر ارباب صرفنظر نمود. این در حالی است که می‌توان کاربردهای بسیاری را در نظر گرفت که در آنها، حجم نتایج بدست آمده در کامپیوترهای کارگر، ضریب مشخصی از حجم بار تحویلی به آنهاست. بنابراین در این گونه موارد نمی‌توان از در نظرگرفتن تاخیر زمانی مورد نیاز جهت بازگشت نتایج به کامپیوتر ارباب صرفنظر نمود. متأسفانه تحقیقات اندکی در این زمینه انجام شده و نتایج بدست آمده در این باره، بسیار محدود است.

امروزه تحقیقات در این زمینه با در نظرگرفتن تاخیر زمانی برای بازگشت نتیجه پردازش‌ها مطرح گردیده است، به‌این ترتیب که اگر m، تعداد کامپیوترهای کارگر باشد، با در نظرگرفتن ترتیب‌های مختلف جهت توزیع و بازگشت نتیجه، پیچیدگی زمانی راه حل بهینه، از مرتبه $O(m!^2)$ خواهد شد و چنانچه تعداد کامپیوترهای کارگر زیاد باشد، بدست آوردن زمان‌بندی بهینه، بسیار زمانبر خواهد بود. تاکنون نیز راه حلی بهینه با پیچیدگی زمانی چند جمله‌ای برای این مسئله ارائه نشده است. اما روش‌های ابتکاری موجود برای حل این مسئله عبارتند از: روش‌های LIFOC[۱۱]، FIFOC[۱۲] [۱۴]، ITERLP[۱۳] [۱۳]، SPORT[۱۴] [۵] و GA[۱۵] [۱].

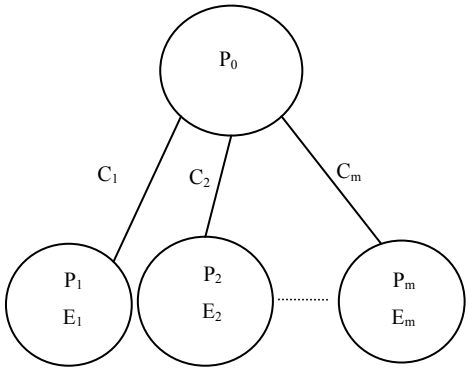
نتایج تحقیقات، نشان دهنده این است که در یک سیستم همگن[۱۶] روش FIFOC عملکرد بهتری نسبت به روش LIFOC داشته و به جواب بهینه نزدیک‌تر است ولی در یک سیستم ناهمگن، عکس این مسئله صادق است، یعنی عملکرد روش LIFOC بهتر از روش FIFOC است [۶]. آخرین روش قابل بررسی در این زمینه، روش SPORT است که در مقالات [۵] و [۶] معرفی و تحلیل گردیده است.

یکی از روش‌های ابتکاری هوشمند در حل این مسئله، استفاده از روش‌های محاسبات تکاملی، از جمله الگوریتم‌های تپه‌نوردی و شبیه‌سازی حرارتی است. الگوریتم‌های محاسبات تکاملی ارائه شده در این زمینه الگوریتم[۱۲] و الگوریتم[۱] است که در اولی، تاخیر زمانی بازگشت نتیجه در نظر گرفته نشده است. هدف ما در این مقاله، پیشنهاد الگوریتم‌های تپه‌نوردی و شبیه‌سازی حرارتی برای حل این مسئله است که در آن، تاخیر زمانی بازگشت نتیجه درنظر گرفته شود.

ادامه این مقاله از بخش‌های زیر تشکیل شده است. در بخش دوم مسئله زمان‌بندی بار تقسیم‌پذیر در یک شبکه دارای هم‌بندی ستاره بیان شده و مدل ریاضی برای آن ارائه گردیده است. در بخش سوم، الگوریتم تپه‌نوردی جهت این زمان‌بندی بیان شده است. در بخش چهارم، الگوریتم شبیه‌سازی حرارتی جهت این زمان‌بندی بیان شده است. در بخش پنجم، روش‌های ترکیبی الگوریتم‌های تکاملی جهت این زمان‌بندی بیان شده است. در بخش ششم، نتایج شبیه‌سازی‌های انجام شده و مقایسه‌ها بیان گردیده است و در نهایت، در بخش پایانی نتیجه‌گیری و پیشنهاد ادامه کار بیان شده است.

۲- بیان مدل مسئله زمان‌بندی بار تقسیم‌پذیر

در شکل (۱)، هم‌بندی ستاره‌ای مورد استفاده در شبکه ارتباطی مدل ارباب و کارگر نشان داده شده است. دراین شکل، کامپیوتر ارباب در ریشه و کامپیوترهای کارگر در برگ‌ها واقع شده‌اند. در ابتدا کل بار محاسباتی L بر روی کامپیوتر ارباب ذخیره گردیده است.



شکل (۱): شبکه ستاره‌ای ناهمگن [۵]

در این مدل، $P=\{P_0,\cdots,P_m\}$ بیانگر مجموعه m+1 کامپیوتر است که در آن P_0 کامپیوتر ارباب و بقیه، کامپیوترهای کارگر هستند؛ $E=\{E_1,\cdots,E_m\}$ بیانگر شاخص سرعت پردازش کامپیوترهای کارگر است؛ E_k بیانگر مدت زمان لازم جهت پردازش یک واحد محاسباتی بر روی کامپیوتر کارگر شماره k است؛ مجموعه $C=\{C_1,\cdots,C_m\}$ بیانگر پارامترهای مربوط به سرعت خطوط ارتباطی شبکه است؛ C_k بیانگر مدت زمان لازم جهت ارسال یک واحد داده محاسباتی از کامپیوتر ارباب به کامپیوتر کارگر شماره k است. دراین مدل، L بیانگر کل حجم بار محاسباتی اولیه موجود در کامپیوتر ارباب است. همچنین با توجه به اینکه به کلیت مسئله لطمه‌ای وارد نخواهد آمد، فرض می‌کنیم L=1 باشد.

در این مدل، $\alpha=\{\alpha_1,\cdots,\alpha_m\}$ بیانگر درصد سهم بار محاسباتی هر یک از کامپیوترهای کارگر است؛ δ نیز بیانگر نسبت حجم داده حاصل از پردازش در هر کامپیوتر کارگر به سهم بار محاسباتی آن است.

دو مجموعه σ_a و σ_c، به‌ترتیب بیانگر ترتیب توزیع بارمحاسباتی به کامپیوترهای کارگر و ترتیب بازگشت نتایج از آنها به کامپیوتر ارباب است. در واقع اولین مجموعه، ترتیب تخصیص بار[۱۷] و دومین مجموعه، بیانگر ترتیب جمع آوری نتایج[۱۸] است؛ $\sigma_a[k]$ بیانگر شماره کامپیوتر کارگری است که به عنوان kاُمین کامپیوتر جهت ارسال یا تخصیص بار از سوی کامپیوتر ارباب انتخاب شده است؛ $\sigma_c[k]$ بیانگر شماره کامپیوتر کارگری است که

به عنوان kاُمین کامپیوتر، نتیجه را به کامپیوتر ارباب برمی‌گرداند؛ $\sigma_a(k)$ بیانگر اندیس محل کامپیوتر کارگر شماره k در ترتیب تخصیص بار محاسباتی است؛ $\sigma_c(k)$ بیانگر اندیس محل کامپیوتر کارگر شماره k در ترتیب بازگشت نتایج است. در این مدل فرض براین است که قبل از زمان‌بندی، مقادیر مجموعه‌های C، E و پارامترهای m و δ مشخص هستند.

در اینجا هدف از زمان‌بندی، تعیین مجموعه‌های σ_a، σ_c و α است به‌گونه‌ای که زمان کل پردازش کمینه شود. زمان کل پردازش عبارت است از زمان شروع توزیع بار محاسباتی تا زمان دریافت آخرین نتیجه پردازش شده توسط کامپیوتر ارباب.

در صورتی‌که دو مجموعه σ_a و σ_c مشخص باشند، مجموعه α را می‌توان با حل برنامه ریزی خطی ارائه شده در شکل (۲) محاسبه کرد. در شکل(۲)، متغیر T، بیانگر زمان کل پاسخ است که قرار است کمینه شود؛ روابط (۱) تا (۴)، بیانگر محدودیت‌های مسئله زمان‌بندی هستند؛ بخش اول محدودیت شماره (۱) بیانگر زمان لازم جهت تخصیص بار محاسباتی کامپیوترهای کارگر یکم تا kاُم، طبق ترتیب σ_a است؛ بخش دوم محدودیت شماره (۱) بیانگر مدت زمان لازم جهت پردازش در کامپیوتر kاُم است؛ بخش سوم محدودیت مزبور نیز در برگیرنده مدت زمان لازم جهت بازگشت نتایج بدست آمده از کامپیوترهای کارگر شماره k تا m، طبق ترتیب σ_c است. محدودیت شماره (۲) بیانگر جمع زمان‌های لازم برای تخصیص کل بار محاسباتی به کامپیوترهای کارگر و جمع‌آوری تمامی نتایج از آنها است. محدودیت شماره (۳) نیز یک رابطه نرمال‌سازی به این منظور است که تخصیص کل بار محاسباتی را به کامپیوترهای کارگر تضمین کند.

مسئله کمینه‌سازی نشان داده شده در شکل (۲) با در نظرگرفتن محدودیت‌های (۱) تا (۴)، یک مسئله برنامه‌ریزی خطی است که می‌تواند به کمک روش‌های استاندارد برنامه‌ریزی خطی، با پیچیدگی زمانی چند جمله‌ای حل شود [۳].

Minimize T

Subject To:

$$\sum_{j=1}^{\sigma_a(k)} \alpha_{\sigma_a[j]} C_{\sigma_a[j]} + \alpha_k E_k + \sum_{j=k}^{m} \delta\alpha_{\sigma_c[j]} C_{\sigma_c[j]} \le T \qquad k=1,\cdots,m \quad (۱)$$

$$\sum_{j=1}^{m} \alpha_{\sigma_a[j]} C_{\sigma_a[j]} + \sum_{j=1}^{m} \delta\alpha_{\sigma_c[j]} C_{\sigma_c[j]} \le T \quad (۲)$$

$$\sum_{j=1}^{m} \alpha_j = 1 \quad (۳)$$

$$T \ge 0,\ \alpha_k \ge 0 \qquad k=1,\cdots,m \quad (۴)$$

شکل(۲): مسئله برنامه‌ریزی خطی [۵]

هدف ما از به کارگیری الگوریتم‌های پیشنهادی، تعیین ترتیب‌های بهینه σ_a و σ_c است. برای حل این مسئله در حالت معمول براساس روش «جستجوی همه حالت‌ها[۱۹]»، نیاز به بررسی تمامی جایگشت‌های دو مجموعه σ_a و σ_c است که برابر است با: $m!\times m! = m!^2$. این روش که جواب بهینه سراسری را تولید می‌کند، در این مقاله الگوریتم بهینه نامیده می‌شود.

در حال حاضر الگوریتم معینی برای حل این مسئله وجود ندارد که بتواند در زمان چند جمله‌ای، جواب بهینه را بیابد. در این زمینه فقط راه حل‌های ابتکاری یا حریصانه[۲۰] وجود دارد. روش‌های LIFOC، FIFOC، ITERLP، SPORT و GA نیز جزو راه حل‌های ارائه شده و مطرح هستند [۵]. الگوریتم‌های تکاملی پیشنهادی برای حل مسئله زمان‌بندی فوق، در مقایسه با روش‌های مطرح توانسته است جواب‌های بهتر و نزدیک‌تری به جواب بهینه تولید کند که نتایج حاصل از اجرای این الگوریتم‌ها و مقایسه آنها با جواب‌ها در سایر روش‌ها، در بخش شبیه‌سازی این مقاله بیان شده است.

۳- الگوریتم تپه‌نوردی برای بهینه‌سازی زمان‌بندی بار محاسباتی تقسیم‌پذیر

تپه‌نوردی را گاهی جستجوی حریصانه محلّی می نامند زیرا یک حالت همسایه خوب را بدون این که از قبل بداند که از آن جا به کجا خواهد رفت، انتخاب می‌کند. با این وجود الگوریتم‌های حریصانه اغلب خوب عمل می‌کنند. تپه‌نوردی اغلب پیشرفت سریعی به سمت راه حل دارد، زیرا معمولاً بهبود بخشیدن یک حالت بد، خیلی ساده است.

موفقیت تپه‌نوردی بستگی بسیار زیادی به شکل دور نمای فضای حالت دارد. اگر تعداد کمی بیشینه محلّی و فلات وجود داشته باشد، تپه‌نوردی با شروع مجدد تصادفی، خیلی سریع یک راه حل خوب را پیدا می کند. مسائل NP-Hard معمولاً دارای تعداد نمایی بیشینه محلی هستند که در آنها گیر می‌افتند. با این وجود یک بیشینه محلی نسبتاً خوب را اغلب می‌توان پس از تعداد کمی شروع مجدد پیدا کرد.

در این مقاله، برای حل مسئله زمان‌بندی، دو روش متفاوت تپه‌نوردی پیشنهاد شده است. این روش‌ها عبارتند از: ۱)تپه‌نوردی ساده ۲)تپه‌نوردی تصادفی

برای حل مسئله، این روش‌ها، با یک مجموعه از راه حل‌های تصادفی آغاز می‌گردند که جمعیت اولیه نامیده می‌شود. هر عضو این جمعیت، یک کروموزوم (فرد) نامیده می‌شود. هر کروموزوم نشان دهنده یک راه حل مورد بررسی است و شامل دو بخش از ژن‌ها است. یک بخش، بیانگر ترتیب σ_a و بخش دیگر بیانگر ترتیب σ_c است. تعداد ژن‌های هر بخش برابر m است که در این مسئله، برابر است با تعداد کامپیوترهای کارگر.

۳-۱ تپه‌نوردی ساده[۲۱]

در این روش ابتدا لیست‌های تصادفی مربوط به ترتیب ارسال و ترتیب دریافت تولید می‌شود. سپس برای هر کدام از لیست‌های m تایی

مذکور، تعداد $\frac{m(m-1)}{2}$ همسایه بررسی می‌شود. همان‌طور که قبلا گفته شد برای دو لیست m تایی دریافت و ارسال، $m! \times m!$ همسایه وجود دارد. ولی در این روش فقط $\frac{m(m-1)}{2} \times \frac{m(m-1)}{2}$ همسایه بررسی می‌شود. در حین این بررسی در هر مرحله همسایه‌ای که وضعیت بهتری دارد انتخاب می‌شود. این عمل تا زمان رسیدن به حالتی که هیچ یک از همسایه‌های آن نتیجه بهتری تولید نکنند ادامه می‌یابد.

۳-۲ تپه‌نوری تصادفی[۲۲]

در این روش حالت بعدی حالت فعلی به صورت کاملاً تصادفی تولید می‌شود. هر حالت بعدی با یک جابه‌جایی در ترتیب ارسال و ترتیب دریافت تولید می‌شود. در این روش فقط $1/10$ تعداد حالات تپه‌نوردی ساده تولید می‌شود.

با پایان یافتن الگوریتم‌های بالا، کروموزوم دارای بهترین مقدار تابع برازندگی، بیانگر مجموعه‌های σ_a، σ_c و α خواهد بود. بطور کلی الگوریتم‌های فوق‌الذکر، به دنبال یافتن نقطه بهینه بیشینه است ولی هدف روش پیشنهادی در زمان‌بندی، یافتن نقطه بهینه کمینه است. در نتیجه تابع برازندگی را به صورت رابطه (۴) در نظر گرفته‌ایم.

$$F=\frac{1}{T} \qquad (1)$$

۴- الگوریتم شبیه‌سازی حرارتی برای بهینه‌سازی زمان‌بندی بار محاسباتی تقسیم‌پذیر

شبیه‌سازی حرارتی یک روش جستجوی تصادفی است که با استفاده از تشابه پروسه سرد شدن تدریجی یک جسم آهنی گداخته و رسیدن به یک ساختار انرژی کریستالی ثابت، مسائل بهینه‌سازی را که در آن هدف یافتن یک مینیمم سراسری است حل می‌کند. این روش در سال ۱۹۸۳ و برای حل مسائل غیرخطی مشکل ابداع شد.

اگر فرض کنیم مسئله کمینه‌سازی باشد، این الگوریتم با به کارگیری یک روش جستجوی تصادفی نه تنها تغییراتی را که منجر به کاهش مقدار تابع هزینه (f) مسئله شود را می‌پذیرد، بلکه با احتمال رابطه (۲) تغییراتی را که منجر به افزایش آن شود را نیز می‌پذیرد:

$$P = exp\left(\frac{-\delta f}{T}\right) \qquad (2)$$

در رابطه (۲)، δf و T دو پارامتر کنترلی هستند. δf مقدار افزایش تابع هزینه (f) و T درجه حرارت را نشان می‌دهد. ثابت شده است که با دقت در کاهش تدریجی درجه حرارت و با داشتن یک زمان بسیار زیاد در حد بی‌نهایت، شبیه‌سازی حرارتی همیشه جواب بهینه را تولید می‌نماید. در ابتدای حل مسئله احتمال پذیرش جواب‌های غیربهینه زیاد ولی با کاهش درجه حرارت این احتمال نیز کاهش می‌یابد.

دمای ابتدایی (T_0) باید به حدی باشد که مسئله را به اصطلاح به خوبی ذوب کند و سپس تا حد منجمد شدن کاهش یابد. بنابراین تعیین T_0 تا حد زیادی به مسئله مورد نظر بستگی دارد.

یک دمای ابتدایی مناسب دمایی است که موجب افزایش احتمال میانگین پذیرش به ۸۰ درصد شود. به عبارتی ۸۰ درصد از تغییراتی که موجب افزایش تابع هزینه مسئله شود، پذیرفته شوند. می‌توان با انجام یک جستجوی ابتدایی که در آن تمامی حالاتی که موجب افزایش f شده و پذیرفته شده‌اند با محاسبه میانگین افزایش حاصل در تابع f (δf^+)، دمای ابتدایی (T_0) را از رابطه (۳) محاسبه نمود:

$$T_0 = \frac{-\delta f^+}{\ln(P_0)} \quad , \quad P_0 = 0.8 \qquad (3)$$

دمای نهایی می‌تواند بر اساس تعداد جواب‌های بدست آمده و یا یک مقدار ثابت محدود شود. دما براساس رابطه (۴) کاهش می‌یابد.

$$Temperature=Temperature*0.95 \qquad (4)$$

۵- روش‌های ترکیبی الگوریتم‌های تکاملی برای حل بهینه زمان‌بندی بار تقسیم‌پذیر

در این بخش، الگوریتم‌های ترکیبی ژنتیک با الگوریتم‌های معرفی شده در بخش ۴، بیان می شوند.

۵-۱ ابتدا تپه‌نوردی ساده، سپس الگوریتم ژنتیک[۲۳] (HCGA)

این روش در واقع ترکیبی از روش الگوریتم ژنتیک و روش تپه‌نوردی ساده است. استدلال مورد نظر برای این روش در این واقعیت نهفته است که اگر جمعیت (نسل) اولیه الگوریتم ژنتیک نسل مناسبی باشد، با احتمال خوبی جواب نهایی الگوریتم ژنتیک بهبود خواهد یافت.

در این روش ابتدا تعداد ۱۰ ترتیب اولیه لیست ارسال و لیست دریافت به همان روشی که در الگوریتم تپه‌نوردی ساده گفته شد تولید می‌شوند، این لیست‌ها به عنوان جمعیت اولیه الگوریتم ژنتیک مورد استفاده قرار می گیرد.

۵-۲ ابتدا الگوریتم ژنتیک، سپس تپه‌نوردی ساده[۲۴] (GAHC)

در این روش الگوریتم ژنتیک ابتدا از یک سری نقاط تصادفی شروع کرده و در نهایت جواب نزدیک به بهینه را تولید می‌کند. سپس این جواب به الگوریتم تپه‌نوردی ساده داده می شود. این الگوریتم نیز با یک بار شروع از همان نقطه داده شده سعی می‌کند جواب یافته شده توسط الگوریتم ژنتیک را بهبود ببخشد (به اصطلاح به قلّه برساند).

۵-۳ ابتدا الگوریتم ژنتیک، سپس تپه‌نوردی تصادفی[۲۵] (GARHC)

این روش نیز دقیقاً مانند روش GAHC است با این تفاوت که جواب الگوریتم ژنتیک توسط تپه‌نوردی تصادفی بهبود می‌یابد. انتظار می‌رود پاسخ‌های این روش نسبت به GAHC تفاوت چندانی نداشته باشد یا حتّی بدتر شود، زیرا حرکات در الگوریتم فوق به‌صورت تصادفی انجام می‌شوند.

۵-۴ ابتدا الگوریتم ژنتیک، سپس الگوریتم شبیه‌سازی حرارتی[۲۶] (GASA)

در این شیوه، الگوریتم ژنتیک پاسخ مربوط به خود را تولید می‌کند، سپس شبیه‌سازی حرارتی در ابتدا به روشی که گفته شد دمای اولیه (T_0) را تخمین می‌زند سپس عملیات خود را با پاسخی که الگوریتم ژنتیک تولید کرده آغاز می‌نماید.

۶- نتایج شبیه‌سازی و مقایسه‌ها

در آزمایش‌های انجام شده، کارایی الگوریتم‌های پیشنهادی را به کمک الگوریتم بهینه، با سایر الگوریتم‌ها مقایسه کرده‌ایم. پیاده‌سازی‌ها، در محیط MatLab و بر روی کامپیوتری با پردازنده Intel Core 2 Due 2.4 GHz و دارای یک گیگا بایت حافظه اصلی انجام شده است. مقادیر پارامترهای C و E جهت انجام آزمایش‌ها، برای نمایش یک سیستم ناهمگن، مشابه روش ارائه شده در مقاله [۵] تولید شده‌اند که در آن ۲۵ مورد مختلف در نظر گرفته‌شده‌است.

برای هر یک از ۲۵ مورد، m مقدار برای پارامترهای C و E، در محدوده‌های تعیین شده، بصورت تصادفی و با توزیع یکنواخت تولید شده‌اند. برای هر مورد فوق، تولید پارامترهای C و E، صد بار تکرار شده است که منجر به صد بار اجرای الگوریتم‌های پیشنهادی و سایر الگوریتم‌ها شده است. برای هر کدام از صد نمونه فوق، اجرای الگوریتم‌های پیشنهادی ده مرتبه تکرار شده است و نتیجه (T) در نظر گرفته شده برای الگوریتم‌های پیشنهادی، میانگین این ده مرتبه از اجرا است.

در تمام آزمایش‌ها، زمان اتمام پردازش کل بار محاسباتی، برای هر الگوریتم محاسبه شده است. اگر متغیر T_{opt} بیانگر این زمان برای الگوریتم بهینه باشد و متغیر T_v نیز بیانگر زمان فوق‌الذکر برای سایر الگوریتم‌ها باشد آنگاه درصد خطای نسبی (ΔT_v) طبق رابطه (۵) محاسبه می‌شود.

$$\Delta T_v = \frac{T_v - T_{opt}}{T_{opt}} \times 100\% \qquad (۵)$$

با توجه به اینکه برای هر مورد از جدول شماره (۱)، صد داده آزمایشی تولید شده است، میانگین رابطه (۵)، بصورت فرمول (۶) محاسبه می‌گردد.

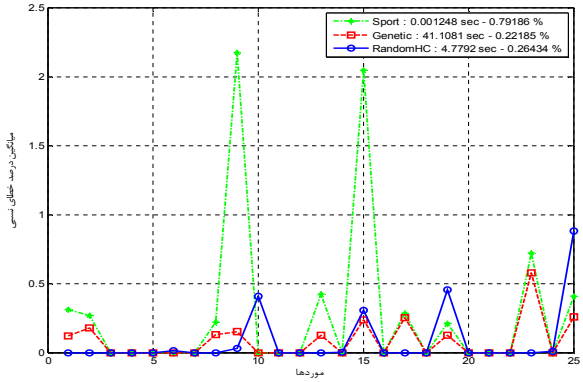
$$\overline{\Delta T_v} = \frac{\sum_{k=1}^{100} \Delta T_v^k}{100} \qquad (۶)$$

فرمول (۶) را میانگین درصد خطای نسبی می‌نامیم.

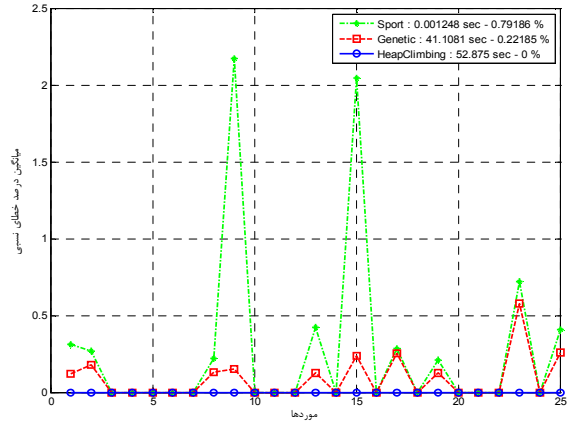
در شکل‌های(۳)، (۴) و(۵)، به‌ترتیب میانگین درصد خطای نسبی برای الگوریتم‌های تپه‌نوردی تصادفی، تپه‌نوردی ساده و شبیه‌سازی حرارتی به همراه الگوریتم ژنتیک و الگوریتم SPORT نشان داده شده است. همانطور که در این نمودارها دیده می شود، میانگین درصد خطای نسبی الگوریتم ژنتیک حدود ۰/۲۲ است و این الگوریتم یک مسئله را در زمان حدود ۴۱ ثانیه حل می‌کند. الگوریتم تپه‌نوردی تصادفی با زمان اجرای تقریباً 1/10 نسبت به الگوریتم ژنتیک، دارای درصد خطای نسبی کمتری در حدود ۰/۲۶ است.

الگوریتم شبیه‌سازی حرارتی با زمان اجرای حدود ۱۷ ثانیه نتایج بهتری را نسبت به الگوریتم ژنتیک و حتی تپه‌نوردی تصادفی تولید کرده است. میانگین درصد خطای نسبی در این روش حدود ۰/۱۴ درصد است.

بهترین عملکرد متعلق به تپه‌نوردی ساده است که حدود ۰/۰۰۰۱ درصد خطا دارد. البته در بین الگوریتم‌های پنج‌گانه فوق، تپه‌نوردی ساده با زمان اجرای حدود ۵۲ ثانیه بیشترین زمان را داراست، در مقابل الگوریتم SPORT با زمان اجرای حدود ۰/۰۰۱ ثانیه سریع‌ترین الگوریتم است، اما همین الگوریتم با درصد خطای نسبی حدود ۰/۷۹ درصد بدترین نتایج را تولید کرده است.



شکل(۳): نمودار میانگین درصد خطای نسبی الگوریتم‌های تپه‌نوردی تصادفی، ژنتیک و SPORT برای m=5, δ=0.5

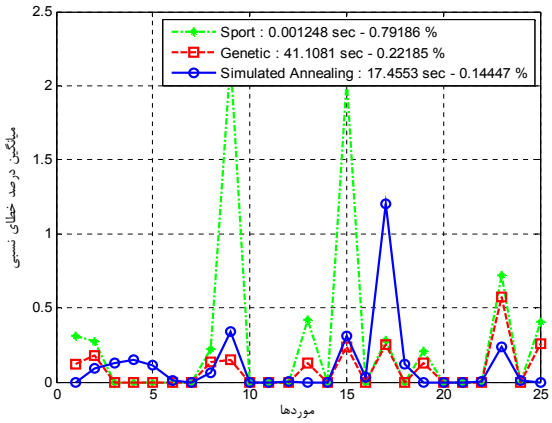


شکل(۴): نمودار میانگین درصد خطای نسبی الگوریتم‌های تپه‌نوردی ساده، ژنتیک و SPORT برای m=5, δ=0.5

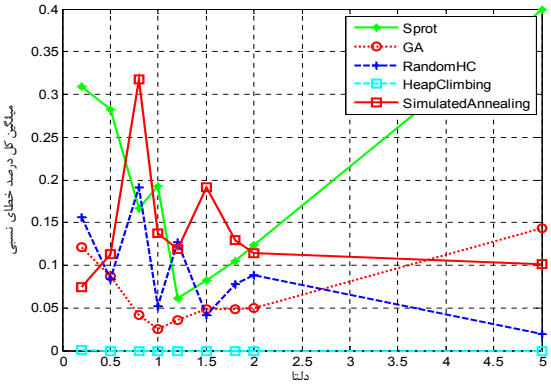
به‌منظور در نظرگرفتن میزان تاثیر پارامتر دلتا بر روی زمان پاسخ‌های بدست آمده در الگوریتم‌های ذکر شده، آزمایش‌های قبلی را برای m=5 و مقادیر مختلف δ تکرار کرده، سپس میانگین مقدار $\overline{\Delta T_v}$ را برای ۲۵ مورد محاسبه نموده‌ایم که حاصل آن در نمودار شماره (۶) نشان داده شده است. رابطه (۷) را میانگین کل

درصد خطای نسبی نامیده‌ایم.

$$\overline{\overline{\Delta T_v}}=\frac{\sum_{i=1}^{25}\overline{\Delta T_v^i}}{25} \quad (۷)$$



شکل(۵): نمودار میانگین درصد خطای نسبی الگوریتم‌های شبیه‌سازی حرارتی، ژنتیک و SPORT برای m=5, δ=0.5



شکل(۶): نمودار میانگین کل درصد خطای نسبی برای m=5

در نمودار (۶)، میانگین کل درصد خطای نسبی برای پنج الگوریتم SPORT، ژنتیک، شبیه‌سازی حرارتی، تپه‌نوردی تصادفی و ساده برای پنج کامپیوتر کارگر ترسیم شده است.

همان‌طور که در نمودار (۶) نشان داده شده است افزایش مقدار δ تقریباً هیچ تأثیری بر روی تپه‌نوردی ساده ندارد و میانگین درصد خطای نسبی آن در حد صفر باقی می ماند. در مورد الگوریتم ژنتیک مقدار خطا تا $\delta = 1.0$ با شیب آرامی کاهش می‌یابد و سپس با شیب ملایمی افزایش می‌یابد، تا این که در $\delta = 5.0$ به حداکثر خود در حدود ۰/۱۵ درصد می‌رسد. الگوریتم شبیه‌سازی حرارتی تا $\delta = 2.0$ فراز و نشیب های زیادی دارد، مانند الگوریتم تپه‌نوردی تصادفی، ولی از $\delta = 2.0$ به بعد میانگین درصد خطای نسبی هر دو الگوریتم با شیب ملایمی کاهش می‌یابد.

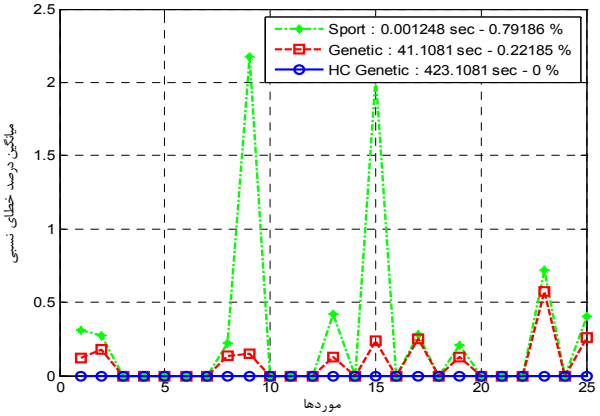
در الگوریتم‌های تپه نوردی تصادفی و شبیه‌سازی حرارتی بیشترین تأثیر δ بر افزایش خطای نسبی مربوط به $\delta = 0.8$ است.

در نمودارهای (۷) تا (۱۰) میزان کارایی الگوریتم‌های ترکیبی نشان داده شده است. ترکیب الگوریتم‌های تپه‌نوردی ساده و ژنتیک جواب‌های بسیار خوبی تولید کرده است، به‌طوری که میانگین درصد خطای نسبی صفر است. الگوریتم ترکیبی فوق با زمان اجرای حدود ۴۲۳ ثانیه توانسته است جواب کاملاً بهینه را تولید نماید. علت کاملاً بهینه بودن این الگوریتم به این دلیل است که نسل اولیه مورد استفاده توسط الگوریتم ژنتیک، جوابهای نسبتاً بهینه الگوریتم تپه‌نوردی ساده بوده‌اند. از طرفی برای آنکه اندازه جمعیت‌ها ۱۰ بوده است و الگوریتم تپه‌نوردی ساده باید برای تولید نسل اولیه ۱۰ بار تکرار شود، مدت زمان اجرای الگوریتم را بسیار طولانی کرده است.

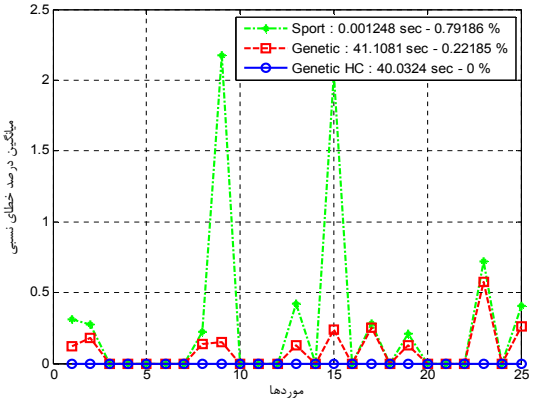
با توجه به نمودارها، استفاده از الگوریتم ابتدا ژنتیک و سپس تپه‌نوردی ساده منطقی‌تر به نظر می‌رسد، زیرا این الگوریتم با زمان اجرای حدود ۴۰ ثانیه جواب‌های کاملاً بهینه را تولید کرده است. علت کاملاً بهینه بودن جواب های این الگوریتم، این است که جواب‌های نسبتاً بهینه الگوریتم ژنتیک توسط تپه‌نوردی ساده به قله می‌رسد و از آنجا که از الگوریتم تپه‌نوردی ساده تنها یکبار و پس از الگوریتم ژنتیک استفاده شده است، زمان اجرا به نحو چشمگیری کاهش پیدا کرده است.

ترکیب الگوریتم ژنتیک و شبیه‌سازی حرارتی نیز نتایج الگوریتم ژنتیک را در زمان تقریبی ۵۱ ثانیه بهبود بخشیده و درصد خطای نسبی آنرا به حدود ۰/۰۴ درصد رسانده است.

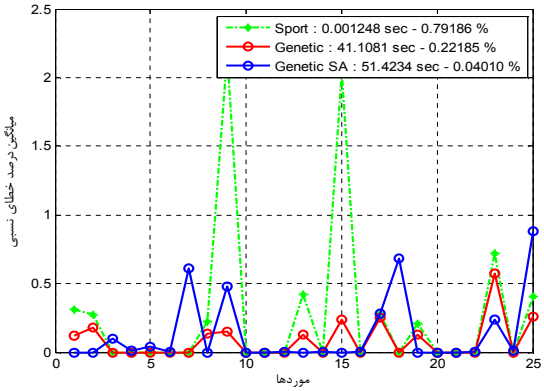
همچنین ترکیب الگوریتم ژنتیک و تپه‌نوردی تصادفی با زمان اجرای حدود ۳۸ ثانیه مسائل را با درصد خطای نسبی ۰/۰۰۰۵ حل کرده است. این الگوریتم ترکیبی عملکرد بهتری نسبت به الگوریتم ژنتیک دارد ولی به خوبی ابتدا ژنتیک و سپس تپه‌نوردی ساده نیست.



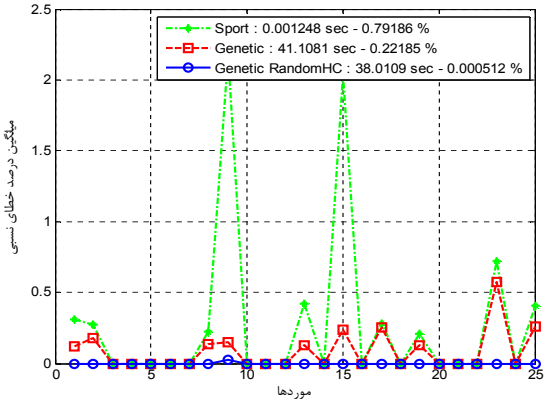
شکل(۷): نمودار میانگین درصد خطای نسبی الگوریتم‌های تپه‌نوردی- ژنتیک، ژنتیک و SPORT برای m=5, δ=0.5



شکل(۸): نمودار میانگین درصد خطای نسبی الگوریتم‌های ژنتیک-تپه‌نوردی، ژنتیک و SPORT برای m=5, δ=0.5

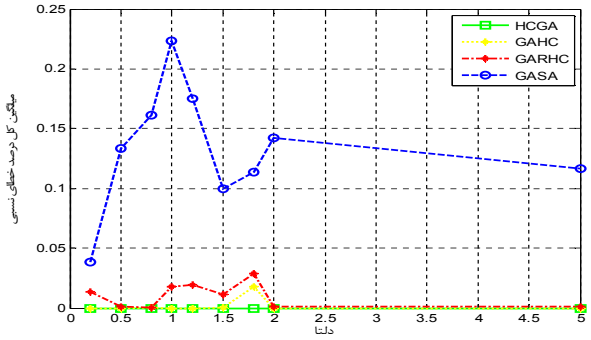


شکل(۹): نمودار میانگین درصد خطای نسبی الگوریتم‌های ژنتیک-شبیه‌سازی حرارتی، ژنتیک و SPORT برای m=5, δ=0.5



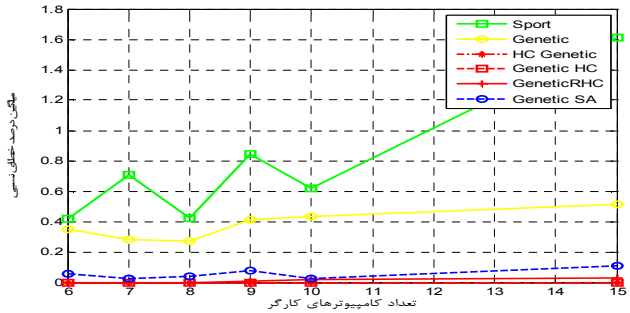
شکل(۱۰): نمودار میانگین درصد خطای نسبی الگوریتم‌های ژنتیک-تپه‌نوردی تصادفی، ژنتیک و SPORT برای m=5, δ=0.5

در شکل (۱۱)، نمودار میانگین کل درصد خطای نسبی برای الگوریتم‌های ترکیبی پیشنهادی، برای تعداد پنج کامپیوتر کارگر و مقادیر مختلف δ نشان داده شده است. با توجه به این نمودار، کمترین میزان خطای نسبی مربوط به الگوریتم ترکیبی تپه‌نوردی ساده و الگوریتم ژنتیک است. الگوریتم ترکیبی ژنتیک و شبیه‌سازی حرارتی از بیشترین میزان خطای نسبی برخوردار است.



شکل(۱۱): نمودار میانگین کل درصد خطای نسبی برای m=5

در شکل (۱۲)، تاثیر افزایش تعداد کامپیوترهای کارگر بر روی میزان خطای الگوریتم‌های پیشنهادی نشان داده شده است. بیشترین تاثیر از نظر بزرگی خطا، الگوریتم ترکیبی ژنتیک-شبیه‌سازی حرارتی داشته است ولی الگوریتم‌های ترکیبی ژنتیک و تپه‌نوردی کمترین تاثیر را داشته‌اند.



شکل(۱۲): بررسی افزایش تعداد کامپیوترهای کارگر برای δ=0.5

در جدول شماره (۱)، الگوریتم‌ها از نظر میانگین مدت زمان اجرا و میانگین کل درصد خطای نسبی با یکدیگر مقایسه شده‌اند.

جدول(۱): مقایسه زمان اجرا و میانگین کل درصد خطای نسبی الگوریتم‌ها برای m=5, δ=0.5

نام الگوریتم	میانگین زمان اجرا (ثانیه)	$\overline{\overline{\Delta T_v}}=\frac{\sum_{i=1}^{25}\overline{\Delta T_v^i}}{25}$
الگوریتم بهینه	501.8594	0
الگوریتم ژنتیک	41.1081	0.22185
الگوریتم SPORT	0.001248	0.79186
الگوریتم تپه‌نوردی ساده	52.875	0.000001
الگوریتم تپه‌نوردی تصادفی	4.7792	0.26434
الگوریتم شبیه‌سازی حرارتی	17.4553	0.14447
الگوریتم تپه‌نوردی ساده- ژنتیک	423.1081	0.000001
الگوریتم ژنتیک- تپه‌نوردی ساده	40.0324	0.000001
الگوریتم ژنتیک- تپه‌نوردی تصادفی	38.0109	0.000512
الگوریتم ژنتیک- شبیه‌سازی حرارتی	51.4234	0.04010

همان‌طور که در جدول (۱) دیده می‌شود، میانگین کل درصد خطای نسبی دو الگوریتم ترکیبی تپه‌نوردی ساده و ژنتیک نسبت به بقیه الگوریتم‌ها کمتر است. اما زمان لازم برای اجرای الگوریتم ترکیبی تپه‌نوردی ساده- ژنتیک نسبت به سایر الگوریتم‌های غیر بهینه، بیشتر است.

۷- نتیجه‌گیری و کارهای آینده

مسئله زمان‌بندی بار محاسباتی تقسیم‌پذیر با درنظرگرفتن زمان بازگشت نتیجه در یک سیستم ناهمگن، جزو مسائل ترکیباتی با درجه پیچیدگی بالا و از مرتبه $O(m!^2)$ است که هنوز الگوریتمی بهینه از مرتبه چند جمله‌ای برای حل آن ارائه نشده است. روش‌های موجود نیز روش‌هایی ابتکاری هستند که نمی‌توانند در تمامی حالات، جواب قابل قبول تولید کنند. در میان الگوریتم های تکاملی محض (بدون ترکیب با الگوریتم ژنتیک) افزایش δ کمترین تأثیر را بر تپه نوردی ساده و در الگوریتم های تکاملی ترکیبی کمترین تأثیر را بر الگوریتم ترکیبی تپه‌نوردی-ژنتیک و الگوریتم ترکیبی ژنتیک-تپه‌نوردی می‌گذارد.

با توجه به مقایسه‌های صورت گرفته، الگوریتم ترکیبی ژنتیک-تپه‌نوردی در مورد تمام مقادیر δ در زمان معقول جواب‌های بسیار خوبی تولید می‌کند.

در پایان و در راستای این تحقیق، می‌توان در نظرگرفتن مواردی عملی مانند محدودیت حافظه در کامپیوترهای کارگر و همچنین در نظرگرفتن سایر مدل‌های تاخیر در شبکه را به عنوان کارهای آتی در نظر داشت.

مراجع

[۱] منصفی، رضا، حمی‌دزاده، جواد، "زمان‌بندی بار محاسباتی تقسیم‌پذیر با در نظرگرفتن زمان بازگشت نتایج در سیستم‌های ناهمگن با استفاده از الگوریتم ژنتیک"، چهاردهمین کنفرانس ملی سالانه انجمن کامپیوتر ایران، دانشگاه صنعتی امیرکبیر، ایران، تهران، اسفند ماه ۱۳۸۷.

[2] Bharadwaj, V., Ghose, D., Mani, V., Robertazzi, T. G., *Scheduling Divisible Loads in parallel and Distributed Systems*, IEEE CS Press, 1996.

[3] Vanderbei, R. J., *Linear Programming: Foundations and Extensions*, 2nd Ed., International Series in Operations Research & Management, vol. 37, Kluwer Academic Publishers, 2001.

[4] Robertazzi, T. G., "*Ten Reasons to Use Divisible Load Theory*", Computer, pp. 63-68, May 2003.

[5] Ghatpande, A., Nakazato, H., Beaumont, O., Watanabe, H., "*SPORT: An Algorithm for Divisible Load Scheduling With Result Collection on Heterogeneous Systems*", IEICE Transactions on Communications, vol. E91-B, no. 8 August 2008.

[6] Ghatpande, A., Nakazato, H., Beaumont, O., Watanabe, H., "*Analysis of Divisible Load Scheduling with Result Collection on Heterogeneous Systems*", IEICE Transactions on Communications, vol. E91-B, no. 7, July 2008.

[7] Cheng, Y. C., Robertazzi, T. G., "*Distributed Computation with Communication Delays*", IEEE Transactions on Aerospace and Electronic Systems, vol. 24, no. 6, pp. 700-712, Nov. 1988.

[8] Li, X., Bharadwaj, V., Ko, C. C., "*Distributed Image Processing on a Network of Workstations*", Int'l J. Computers and Applications, vol. 25, no. 2, pp. 1-10, 2003.

[9] Blazewicz, J., Drozdowski, M., Markiewicz, M., "*Divisible Task Scheduling: Concept and Verification*", Parallel Computing, vol. 25, pp. 87-98, 1990.

[10] Chan, S., Bharadwaj, V., Ghose, D., "*Large Matrix-Vector Products on Distributed Bus Networks with Communication Delays Using the Divisible Load Paradigm: Performance and Simulation*", Math. and Computers in Simulation, vol. 58, pp. 71-92, 2001.

[11] Altilar, D. Paker, Y., "*Optimal Scheduling Algorithms for Communication Constrained Parallel Processing*", Proc. Eighth Int'l Euro-Par Conf. pp. 197-206, 2002.

[12] Suresh, S., Mani,V., Omkar, S. N., Kim, H. J., "*Divisible Load Scheduling in Distributed Systems with Buffer Constraints: Genetic Algorithm and Linear Programming Approach*", International Journal of Parallel, Emergent and Distributed Systems, Vol. 21, No. 5, pp. 303-321, Oct. 2006.

[13] Ghatpande, A., Nakazato, H., Watanabe, H., Beaumont, O., "*Divisible Load Scheduling with Result Collection on Heterogeneous Systems*", Proc. Heterogeneous Computing Workshop(HCP 2008), April 2008.

[14] Beaumont, O., Marchal, L., Rehn, V., Robert Y., "*FIFO Scheduling of Divisible Loads with Return Messages Under the One Port Model*", Proc. Heterogeneous Computing Workshop HCW'06, April 2006.

[15] Robertazzi, T., http://www.ece.sunysb.edu/~tom/dlt.html.

آخرنویس‌ها

[1] Jobs
[2] Heterogeneous
[3] NP_Hard
[4] Divisible Load
[5] Load Units
[6] Fine-Granularity
[7] Massive-Data
[8] Data Mining
[9] Master-Worker
[10] Makespan
[11] Last In First Out Channel
[12] First In First Out Channel
[13] Iteratively Solving Linear Programs
[14] System Parameters based Optimized Result Transfer
[15] Genetic Algorithm
[16] Homogeneous
[17] Allocation Sequence
[18] Collection Sequence
[19] Exhaustive Search
[20] Greedy
[21] Simple Hill Climbing
[22] Random Hill Climbing
[23] Hill Climbing–Genetic Algorithm
[24] Genetic Alghorithm–Hill Climbing
[25] GA–Random Hill Climbing
[26] GA – Simulated Annealing