Archive of SID

فصلنامه علمی- پژوهشی
فناوری اطلاعات و ارتباطات ایران

سال دهم، شماره‌های ۳۷ و ۳۸، پاییز و زمستان ۱۳۹۷
صص: ۷۱ -۸۴

زمان‌بندی کارها در محیط‌های ابری با استفاده از چارچوب نگاشت – کاهش و الگوریتم ژنتیک

*سید نیما خضر *نیما جعفری نویمی‌پور
*کارشناس ارشد، گروه مهندسی کامپیوتر، واحد تبریز، دانشگاه آزاد اسلامی، تبریز، ایران
**دانشیار، گروه مهندسی کامپیوتر، واحد تبریز، دانشگاه آزاد اسلامی، تبریز، ایران
تاریخ دریافت: ۱۳۹۸/۱۰/۰۴ تاریخ پذیرش:۱۳۹۹/۰۳/۱۶

چکیده

زمان‌بندی وظایف یک جزء حیاتی هر سیستم توزیع‌شده همچون گرید، ابر و شبکه‌های نظیر به نظیر می‌باشد که وظایف را برای اجرا به منابع مناسب ارجاع می‌دهد. روش‌های رایج در زمان‌بندی دارای معایبی از قبیل پیچیدگی زمانی بالا، هم‌زمان اجرا نشدن کارهای ورودی و افزایش زمان اجرای برنامه است. الگوریتم‌های زمان‌بندی بر پایه اکتشاف جهت اولویت‌دهی به وظایف از سیاست‌های متفاوتی استفاده می‌کنند که باعث به وجود آمدن زمان‌های اجرای بالا بر روی سیستم‌های رایانش توزیع‌شده ناهمگن می‌شود. بنابراین، روشی مناسب است که اولویت‌دهی آن باعث تولید زمان اجرای کل کمینه گردد. الگوریتم ژنتیک به‌عنوان یکی از روش‌های تکاملی به‌منظور بهینه کردن مسائل NP-کامل[1] به کار گرفته می‌شود. در این مقاله الگوریتم ژنتیک موازی با استفاده از چارچوب نگاشت-کاهش برای زمان‌بندی وظایف بر روی رایانش ابری با استفاده از صف‌های اولویت چندگانه ارائه‌شده است. ایده اصلی این مقاله، استفاده از چارچوب نگاشت-کاهش برای کاهش زمان اجرای کل برنامه می‌باشد. نتایج آزمایش‌ها بر روی مجموعه‌ای از گراف‌های جهت‌دار بدون دور تصادفی حاکی از آن است که روش پیشنهادی زمان اجرای کل دو روش موجود را با سرعت همگرایی بالا بهبود داده است.

واژه‌های کلیدی: رایانش ابری، زمان‌بندی وظایف، کاهش زمان اجرا، الگوریتم ژنتیک، نگاشت-کاهش، هادوپ.

[1]. NP-Completeness

نویسندهٔ عهده‌دار مکاتبات: نیما جعفری نویمی‌پور jafari@iaut.ac.ir

www.SID.ir

Archive of SID

۱- مقدمه

سیستم‌های رایانش توزیع‌شده ناهمگن[۲] نوعی از سیستم‌های رایانش توزیع‌شده است که وظایف را روی چندین پردازنده‌های اجرا میکند [۱]. زمان‌بندی وظایف در محیط‌های ابری به مجموعه‌ای از زیر وظایف کوچک‌تر اولویت‌دار، جهت پردازش تجزیه می‌شود [۲]. این زیروظایف نشان‌دهنده محدودیت تقدم[۳] می‌باشند به این معنی که نتیجه دیگر وظایف قبل از اجرای زیر وظیفه فعلی نیاز است [۲]. با تجزیه یک کار رایانش به زیر وظایف و اجرای آن‌ها بر روی چندین پردازنده، زمان اتمام وظیفه می‌تواند کاهش پیدا کند. بنابراین هدف از این کار، زمان‌بندی زیروظایف بر روی تعدادی پردازنده در دسترس جهت کاهش زمان اتمام وظایف بدون نقض محدودیت‌های تقدم می‌باشد [۳]. توسعه[۴] الگوریتم‌های زمان‌بندی وظایف که زیروظایف یک برنامه را به پردازنده‌ها تخصیص می‌دهند، در سیستم‌های ابری یک چالش می‌باشد. باوجود تلاش‌های بسیاری که اخیراً در زمینهٔ زمان‌بندی وظایف صورت گرفته است، همچنان این مسئله در محیط‌های رایانش ناهمگن به‌عنوان یکی چالش مهم باقی‌مانده است [۴]. تعدادی از چالش‌های مهم زمان‌بندی وظایف در سیستم‌های ابری در زیر آورده شده است:

- ناهمگن بودن منابع
- زمان اجرای الگوریتم[۵]
- زمان اجرای کل[۶]
- سرعت همگرایی[۷] راه‌حل‌ها در روش‌های تکاملی
- بهره‌وری[۸] الگوریتم زمان‌بندی

با توجه به افزایش حجم داده‌ها در محیط ابری کاهش زمان اجرای الگوریتم زمان‌بندی همچنین به‌عنوان یک چالش مهم باقی‌مانده است. بنابراین الگوریتم‌های گوناگونی جهت کاهش زمان اتمام وظایف با موازی‌سازی زیروظایف و رعایت رابطه‌های تقدم آن‌ها ارائه‌شده‌اند. معمولاً رابطه‌های تقدم با گراف جهت‌دار بدون دور[۹] نشان داده می‌شوند که از رأس‌ها که نشان‌دهنده کارها و یال‌های جهت‌دار نشانگر وابستگی بین کارها می‌باشد. گراف‌های جهت‌دار بدون دور برای نشان دادن برنامه‌ها با حجم زیاد و مختلف مناسب و گویا است [۵]. از طرف دیگر سیستم هادوپ، اجرا وظایف را در یک مرکز داده مشترک را فراهم می کند[۱۸]. نگاشت-کاهش یک ابزار مؤثر در هادوپ برای پردازش داده‌های بزرگ در رایانش ابری می‌باشد که با استفاده از پردازنده‌ها و یا کامپیوترها به‌صورت موازی دستورات و برنامه‌ها را اجرا می-نماید [۶]. این چهارچوب برنامه نویسی اجازه اجرای پردازش موازی و توزیع شده بروی مجموعه بزرگی از داده‌ها در یک محیط توزیع‌یافته را می‌دهد [۱۶]. این چارچوب با رویکردها مختلفی ارائه‌شده است و در زمینه‌های مختلف کاربردهای مختلفی دارد ازجمله الگوریتم‌هایی که در داده‌های بزرگ دارای پیچیدگی زمانی بالایی است که با بهبود پردازش به‌صورت موازی زمان اجرای الگوریتم را کاهش می‌دهد [۷]. لذا این مقاله به چگونگی، زمان‌بندی کارهای اولویت‌دار بر روی گراف‌های مستقیم بدون دور بزرگ به کمک الگوریتم ژنتیک و چارچوب نگاشت-کاهش می‌پردازد.

به‌صورت خلاصه اهداف مقاله به‌صورت زیر است:

- استفاده از الگوریتم ژنتیک برای زمان‌بندی در محیط‌های ابری ناهمگن با استفاده چارچوب نگاشت-کاهش
- بهبود زمان اجرای کل برنامه
- افزایش سرعت همگرایی راه‌حل‌ها و جلوگیری از همگرایی زودرس

ادامه مقاله به صورت زیر سازماندهی خواهد شد: در بخش ۲ برخی از روش‌های مهم در زمان‌بندی کارها در محیط‌های ابری مورد مطالعه و بررسی قرار خواهد گرفت. در بخش ۳ ساختار کلی و روش پیشنهادی با استفاده از چارچوب نگاشت-کاهش و الگوریتم ژنتیک نشان داده خواهد شد. در بخش ۴ هم به بررسی نتایج بدست آمده و مقایسه روش

---

[2] .Heterogeneous distributed computing systems
[3] .Precedence constraint
[4] .Development
[5] .Execution time
[6] .Makespan
[7] .Convergence speed
[8] .Efficiency
[9] .Directed acyclic graph

www.SID.ir

Archive of SID

پیشنهادی با روش‌های دیگر در زمان اجرای کل برنامه پرداخته خواهد شد. و در نهایت نتیجه گیری و پیشنهاد کارهای آتی نشان داده خواهد شد.

۲-پیشینهٔ تحقیق

در این بخش روش‌های موجود در زمینه زمان‌بندی وظایف و بررسی مزایا و معایب این روش‌ها دارد. همچنین بعد از بررسی روش‌های موجود ایده اصلی روش زمان‌بندی پیشنهادی این مقاله شرح داده می‌شود. طبیعت چندهدفه در مسئله زمان‌بندی در ابر، حل آن مخصوصاً درزمانی که کارها زیاد و پیچیده می‌باشند را سخت می‌کند. در میان مسائل مختلف زمان‌بندی، زمان‌بندی ماشین با پردازنده‌های موازی یکی از مسائل بهینه‌سازی NP-سخت است. در مسئله زمان‌بندی پردازنده موازی، مجموعه‌ای از n کار وجود دارد ($T_1, T_2, \ldots, T_n$)؛ هرکدام می‌توانند بر روی m پردازنده ($P_1, P_2, \ldots, P_m$) پردازش شوند، هر کار باید با رعایت تقدم بر روی پردازنده پردازش شود. هر پردازنده فقط می‌تواند یک کار را در یک‌زمان معین پردازش کند. در ادامه چندین روش برای زمان‌بندی کارها در محیط‌های ابری بررسی خواهد شد.

پژوهشگران در مقاله [۸] جهت رسیدن همزمان به دو معیار کارآیی بالا و زمان‌بندی سریع دو الگوریتم زمان‌بندی بر پایه لیست[۱۰] HEFT وCPOP ارائه داده‌اند. الگوریتم HEFT در هر مرحله یک وظیفه با بیشترین ارزش رتبه رو به بالا[۱۱] را انتخاب و با استفاده از روش درج[۱۲] به پردازنده‌ای تخصیص می‌دهد که زودترین زمان شروع را کاهش دهد. درحالی‌که الگوریتم CPOP برای اولویت‌دهی از تجمیع مقدارهای رو به بالا و رو به پائین[۱۳] به وظایف استفاده می‌کند. تفاوت دیگر این دو الگوریتم در قسمت انتخاب پردازنده به وظایف می‌باشد به‌طوری که الگوریتم CPOP وظایف موجود بر روی مسیر بحرانی را بر روی پردازنده‌ای اجرا می‌کند که زمان اجرای کل وظایف مسیر بحرانی را کمینه کند. در مقاله [۹] یک الگوریتم زمان‌بندی مبتنی بر الگوریتم کلونی مورچه ارائه‌شده است. هدف این زمان‌بند کمینه کردن زمان گردش مجموعه‌ی کارها همزمان با کمینه کردن حداکثر زمان اتمام انجام کل کارها می‌باشد. این روش ارائه‌شده اجازه پرداختن به کارهای چابک‌تر درحالی‌که زمان تکمیل کاهش می‌یابد را می‌دهد. همچنین نتایج ارزیابی نشان می‌دهد که الگوریتم کلونی مورچه بهتر از الگوریتم‌های تصادفی و بهترین تلاش عمل می‌کند. در مقاله [۱۰] روشی در زمینه زمان‌بندی کارها در ابر خبره ارائه‌شده است. این روش دارای عملکرد سریع‌تر با دقت بالاتر و نرخ شکست پایین‌تر در مقایسه با روش‌های بررسی‌شده می‌باشد. نتایج پیاده‌سازی نشانگر بهبود حداکثر ۵۶ درصدی، ۱۴ درصدی و ۷,۵ درصدی به ترتیب بر اساس معیارهای زمان کلی اجرا، نرخ شکست و کارایی منابع انسانی در مقایسه با روش‌های دیگر می‌باشد. در مقاله [۱۱] یک الگوریتم زمان‌بندی سطوح پویا ارائه‌شده است. نتایج شبیه‌سازی‌ها اثبات کرده‌اند که این الگوریتم می‌تواند به‌صورت کارا نیازهای جریان کار رایانش ابری را ازنظر اعتماد، کاهش هزینه‌های زمانی و اطمینان از اجرای کارها در یک حالت امن را تأمین کند. پژوهشگران در مقاله [۱۲] یک روش زمان‌بندی وظایف در سیستم‌های رایانش ناهمگن با استفاده از الگوریتم ژنتیک و صف‌های اولویت‌دار چندگانه به نام MPQGA را ارائه کرده‌اند.

ایده اصلی روش ارائه‌شده به‌کارگیری مزیت‌های هر دو نوع الگوریتم اکتشافی و تکاملی و اجتناب از ایرادات آن‌ها می‌باشد. روش ارائه‌شده الگوریتم ژنتیک را جهت تخصیص اولویت به هر وظیفه و روش اکتشافی EFT را جهت نگاشت[۱۴] وظایف به پردازنده‌ها به کار گرفته است. همچنین روش MPQGA عملگرهای ترکیب، جهش و تابع برازندگی مناسبی را جهت زمان‌بندی گراف جهت‌دار بدون دور طراحی کرده است. نتایج شبیه‌سازی انجام‌شده بر روی گراف‌های دنیای واقعی[۱۵] و تصادفی نشان می‌دهد که

[10] .List-based scheduling
[11] .Upward rank
[12] .Insertion-based approach
[13] .Downward rank
[14]. Mapping
[15] .Real world graphs

www.SID.ir

Archive of SID

الگوریتم MPQGA دو روش غیرتکاملی و یک روش جستجوی تصادفی به نام BGA [16]را بهبود بخشیده است. در این ادامه به مقایسه روش‌های موجود پرداخته‌شده است. در الگوریتم HEFT وظایف با استفاده از ارزش رو بالا انتخاب می‌شود و دارای همگرای متوسط می‌باشد، در روش CPOP وظایف با استفاده از ارزش رو بالا و پایین انتخاب می‌شود و مانند HEFT دارای سرعت همگرایی متوسط می‌باشد. در الگوریتم MPQGA که بهبودیافته الگوریتم HEFT است برای بهبود زمان‌بندی از الگوریتم ژنتیک استفاده‌شده است. جدول ۱ الگوریتم‌های موجود را از جنبه روش مورداستفاده، مزایا و معایب آن‌ها مورد مقایسه قرار می‌دهد.

جدول۱: مقایسه روش‌های بررسی‌شده برای زمان‌بندی کارها در محیط‌های ابری

الگوریتم‌های زمان‌بندی	معایب	مزایا	زمان اجرا
[۸]	همگرایی کند، عدم تولید نتایج پایدار در مسائل با طیف گسترده	پیچیدگی کمتر	متوسط
[۹]	نتایج نامناسب در گراف‌های بر پایه محاسبات	پیچیدگی و هزینه ارتباطی کمتر	بالا
[۱۰]	همگرایی کند	هزینه ارتباطی کمتر	متوسط
[۱۱]	مناسب برای محیط‌های همگن	هزینه ارتباطی و اجرای کمتر	متوسط
[۱۲]	همگرایی کند و پیچیدگی بیشتر	کارایی بهتر	متوسط

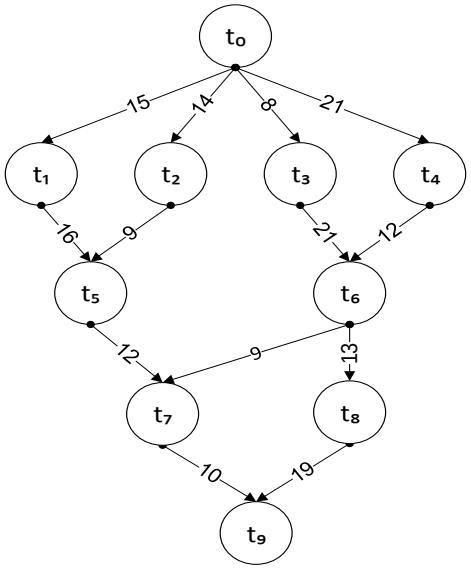
در روش‌های بررسی‌شده، زمان اجرای کل برنامه‌ها بالا است. در صورتی که تعداد زیر وظایف افزایش پیدا کند زمان اجرای کل برنامه افزایش می‌یابد. مطالعات اخیر نشان می‌دهد که الگوریتم‌های ژنتیک بر روی انواع مسائل طبیعی عملکرد موفقی داشته‌اند. الگوریتم ژنتیک یک روش جستجوی مؤثر برای پیدا کردن یک جواب بهینه یا نزدیک به بهینه هست. با توجه به‌سرعت پایین الگوریتم ژنتیک در مسائل بزرگ، دراین مقاله برای افزایش سرعت همگرایی و دوری از همگرایی زودرس و کاهش زمان اجرای برنامه از چارچوب نگاشت-کاهش استفاده شده است.

۳-روش پیشنهادی

در این بخش به روش پیشنهادی پرداخته شده است. در ابتدا تعاریف متغییرهای مسئله بیان شده است در ادامه به معرفی کردن الگوریتم ژنتیک برای ترکیب کردن خدمات ابری پرداخته شده است. در ادامه یک الگوریتم ژنتیک موازی جدید زمان‌بندی وظیفه با استفاده از چارچوب نگاشت-کاهش بر روی رایانش ابری با استفاده از صف‌های اولویت چندگانه ارائه‌شده است.

۳-۱-تعاریف اولیه

روش پیشنهادی دارای یک مجموعه از پردازنده‌های ناهمگن P می‌باشد که توسط شبکه پرسرعت به همدیگر متصل شده‌اند. در این روش مسئله زمان‌بندی وظایف توسط گراف جهت‌دار بدون دور G(V,E) نمایش داده‌شده است که شکل ۱ نمونه ارائه‌شده در این مقاله را نشان می‌دهد.



شکل ۱: گراف جهت‌دار بدون دور با ۱۰ زیروظیفه [۵]

[16] Basic genetic algorithm

www.SID.ir

Archive of SID

به‌طوری که V نشان‌دهنده زیروظایف برنامه و E نشان‌دهنده لبه‌های گراف است که وابستگی‌های بین زیروظایف را مشخص می‌کنند. هر زیر وظیفه در گراف فقط بر روی یک پردازنده می‌تواند اجرا شود. مقدارهای لبه‌ها نشانگر مقدار هزینه ارتباطی بین دو زیر وظیفه است و میانگین هزینه محاسباتی هر زیر وظیفه در پردازنده‌ها بر روی هر گره مقدارگذاری شده است. جدول ۲ نمادهای استفاده‌شده در رابطه‌ها و الگوریتم‌های این مقاله را توضیح می‌دهد.

جدول ۲ :نمادهای استفاده‌شده در روش پیشنهادی

نماد	شرح
ti	i امین زیر وظیفه در گراف
pk	k امین پردازنده در سیستم
E	مجموعه لبه‌ها در گراف
T	مجموعه زیروظایف در گراف
P	مجموعه پردازنده‌ها در سیستم
tentry	زیر وظیفه ورودی در گراف
texit	زیر وظیفه خروجی در گراف
Succ(ti)	مجموعه تأخرهای زیر وظیفه ti
Pred(ti)	مجموعه تقدم‌های زیر وظیفه ti
$\overline{W(t_i)}$	میانگین هزینه محاسباتی زیر وظیفه ti
$W(t_i, p_k)$	هزینه محاسباتی زیر وظیفه ti بر روی پردازنده
$C(t_i, t_j)$	مقدار هزینه ارتباطی بین وظایف ti و t_j
$rank_b(t_i)$	رتبه رو به بالای زیروظیفه ti
$rank_t(t_i)$	رتبه رو به پایین زیروظیفه ti
Rankb+t(ti)	جمع رتبه روبه بالا و پایین زیروظیفه ti
$EST(t_i, p_k)$	زودترین زمان شروع زیروظیفه ti بر روی
$EFT(t_i, p_k)$	زودترین زمان پایان زیروظیفه ti بر روی
$AST(ti, pk)$	زمان شروع واقعی زیروظیفه ti بر روی
$AFT(ti, pk)$	زمان پایان واقعی زیروظیفه ti بر روی پردازنده
$avail\{k\}$	زمان بیکار و در دسترس بودن پردازنده pk
pop	مخفف کلمه انگلیسی جمعیت
PopSize	مخفف کلمه انگلیسی اندازه جمعیت
ChSize	مخفف کلمه انگلیسی اندازه کروموزوم

درصورتی‌که دو زیر وظیفه متصل به یکدیگر بر روی پردازنده یکسان زمان‌بندی شوند هزینه ارتباطی بین آن‌ها صفر در نظر گرفته می‌شود. گره‌های شروع و خروجی گراف به ترتیب نشان‌دهنده شروع و پایان برنامه هستند. مدل استفاده برای الگوریتم پیشنهادی نامرتبط[17] می‌باشد، بدین معنی که یک پردازنده ممکن است بعضی از وظایف را در زمان کمتر و بعضی دیگر را در زمان بیشتر اجرا کند که در جدول ۳ نشان داده شده است. همچنین برای شروع اجرای یک زیروظیفه می‌بایست تمام زیر وظایف تقدم[18] آن زمان‌بندی و اجراشده باشند

جدول ۳: هزینه‌های محاسباتی زیروظایف بر روی پردازنده‌ها [۱۷]

Task	p_0	p_1	p_2	$\bar{\omega}$
t_0	۸	۹	۱۰	۹
t_1	۹	۱۰	۱۱	۱۰
t_2	۱۰	٦	۱۱	۹
t_3	۱۲	۸	۱٦	۱۲
t_4	۲۵	۱۸	۱۷	۲۰
t_5	۱۳	۱۰	۱٦	۱۳
t_6	۷	۱۲	۱۷	۱۲
t_7	۱۷	۱۰	۱۲	۱۳
t_8	۱۲	۸	۱۳	۱۱
t_9	۱۳	۱۰	۱۳	۱۲

۲-۳-روش پیشنهادی

در این بخش ابتدا ساختار الگوریتم ژنتیک ارائه می‌شود. سپس به تشریح جزئیات روش پیشنهادی پرداخته خواهد شد. همچنین برای بهبود الگوریتم ژنتیک پیشنهادی از چارچوب برنامه‌نویسی نگاشت-کاهش استفاده‌شده است. الگوریتم تکاملی ژنتیک از عملگرهای مختلفی استفاده می-کند که عملگرهای ژنتیک نامیده می‌شوند. این عملگرها باعث تولید کروموزوم‌هایی بهتر از جمعیت اولیه می‌شوند. این عملگرها جهت همگرایی راه‌حل‌های مسئله لازم و ضروری هستند [۱۳]. یک الگوریتم ساده ژنتیک دارای سه عملگر اساسی ژنتیک است. این عملگرها شامل عملگرهای انتخاب، ترکیب و جهش هستند. عملگر انتخاب بعضی از راه حل ها را به عنوان والد از میان جمعیت انتخاب می کند. عملگر ترکیب والد های انتخاب شده را جهت تولید فرزندان

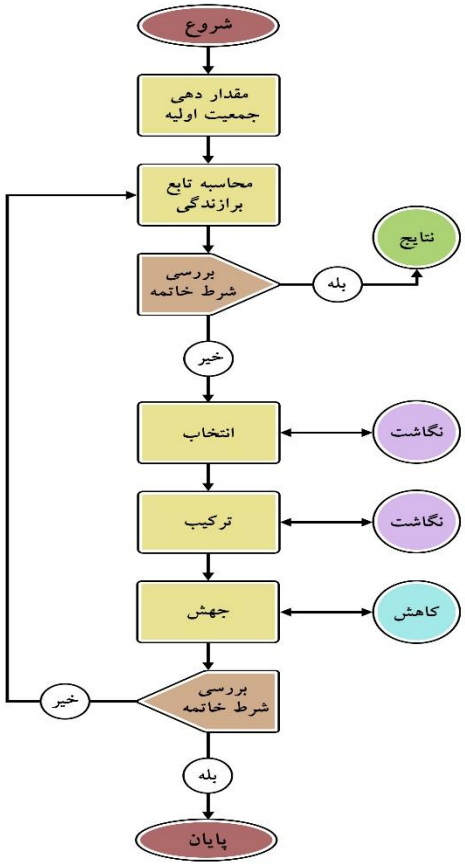
[17] .Unrelated
[18] .Precedence

www.SID.ir

Archive of SID

جدید ترکیب می کند و عملگر جهش مطابق قوانین مشخص شده فرزندان را تغییر می دهد.

برای توسعه و پیاده‌سازی یک الگوریتم جدید زمان‌بندی وظایف، از ترکیب الگوریتم ژنتیک و چارچوب نگاشت-کاهش استفاده‌شده است. با تکنیک نگاشت-کاهش می‌توان الگوریتم‌ها را در فضای موازی پیاده‌سازی نمود. هادوپ ابزاری است که این تکنیک را پیاده‌سازی نموده است و برای هر الگوریتم تنها کافی است دو قسمت نگاشت و کاهش آن را پیاده‌سازی کنیم [۲۲]. مراحل الگوریتم پیشنهادی در شکل ۲ نمایش داده‌شده‌اند. الگوریتم پیشنهادی با مقداردهی جمعیت اولیه شروع می‌شود.



شکل ۲: مراحل روش پیشنهادی

بعد از تولید جمعیت اولیه هر کروموزوم ارزیابی‌شده و با مقدار زمان اجرای کل رتبه‌بندی می‌شوند و این رتبه‌بندی به‌عنوان مقدار برازندگی هر کروموزوم شناخته می‌شود. رتبه بندی انجام‌شده از روش تخصیص پردازنده HEFT استفاده می‌کند. سپس تعدادی از کروموزوم‌ها جهت انجام عمل ترکیب انتخاب و ترکیب می‌شوند. بعد از مرحله ترکیب، عمل موازی‌سازی روی کروموزم‌ها اعمال می‌شود. الگوریتم پیشنهادی تا زمان وقوع شرط خاتمه تکرار می‌شود. در ادامه این مقاله به جزئیات تک‌تک مراحل روش پیشنهادی و شبه کدهای آن پرداخته می‌شود.

روش پیشنهادی زمان‌بندی وظایف را در دو مرحله انجام می دهد: در مرحله یکم به‌منظور تخصیص وظایف به پردازنده‌ها از روش ژنتیک به همراه روش‌های اکتشافی استفاده‌شده است و در مرحله دوم تخصیص وظایف به پردازنده‌ها از روش ژنتیک به همراه چارچوب نگاشت-کاهش استفاده‌شده است. تابع اصلی الگوریتم پیشنهادی جهت زمان‌بندی کردن وظایف بر روی منابع ناهمگن در ابر است.

۳-۳-مقداردهی جمعیت اولیه

ساختن یک کروموزوم توسط روش رمزنگاری[19] مرحله ابتدایی در مقداردهی جمعیت اولیه می‌باشد [۱۹]. ساختمان کروموزوم الگوریتم پیشنهادی دارای ژن[20] می باشد که نشان‌دهنده یک راه‌حل از مسئله زمان‌بندی وظایف می‌باشد [۲۰]. ساختمان یک کروموزوم شامل یک جایگشت از اعداد طبیعی ۰ تا n-1 می‌باشد که نشانگر ترتیب اولویت‌های وظایف در گراف جهت‌دار بدون دور با تعداد n می‌باشد. علاوه براین ترتیب اولویت وظایف در کروموزوم می‌بایست یک ترتیب توپولوژیکی معتبر باشد بطوریکه گره شروع بایستی در اول کروموزوم و گره پایانی در آخر کروموزوم قرار گیرد و همچنین سایر وظایف در مکان‌هایی از کروموزوم قرار گیرند که محدودیت‌های تقدم وظایف را نقض نکنند. به‌منظور عملی بودن زمان‌بندی باید تمام زیر وظایف موجود در گراف زمان‌بندی شوند.

[19] Encoding.
[20] Genes.

www.SID.ir

Archive of SID

جدول ۳: اولویت‌های وظایف

Subtask (ti)	Rankb	Rankt	Rankb+t	level
t0	۱۲۹	۰	۱۲۳	۰
t1	۸۱	۲۴	۱۰۵	۱
t2	۷۸	۲۳	۱۰۱	۱
t3	۱۰۰	۱۷	۱۱۷	۱
t4	۹۹	۳۰	۱۲۹	۱
t5	۶۰	۵۰	۱۱۰	۲
t6	۶۷	۵۲	۱۱۹	۲
t7	۳۵	۷۵	۱۱۰	۳
t8	۴۲	۷۷	۱۱۹	۳
t9	۱۲	۱۰۷	۱۱۹	۴

اندازه جمعیت اولیه الگوریتم پیشنهادی با نماد PopSize نمایش داده می‌شود و مقدار آن ۴ برابر تعداد وظایف موجود در گراف است (n×۴). اندازه جمعیت الگوریتم تا پایان یافتن الگوریتم ثابت است یعنی در طول تولید نسل‌های جدید این مقدار تغییری نمی‌کند. در الگوریتم پیشنهادی به‌منظور داشتن تخم‌ریزی[21] خوب در مقداردهی جمعیت اولیه از سه سیاست رتبه‌بندی اکتشافی بنام‌های رتبه رو به بالا(رابطه (۱))، رتبه رو به پایین (رابطه (۲)) و ترکیبی از این دو روش با رتبه سطح (رابطه (۳)) برای مقداردهی سه کروموزوم اول جمعیت استفاده‌شده است [۱۲]. در جدول ۳ با استفاده از سه سیاست رتبه‌بندی شده است. اولویت‌های کروموزوم‌های باقیمانده با جایگشت‌های تصادفی تولید می‌شوند. کروموزوم‌های تولیدشده تصادفی به دلیل اینکه اولویت‌هایشان معتبر نیست از سمت چپ به راست به‌طوری‌که زیروظایف محدودیت‌های اولویت را نقض نکنند مرتب‌سازی می‌شوند. فرآیند تولید جمعیت اولیه درالگوریتم ۱ نشان داده شده است.

$$rank_b(t_i) = \overline{W(t_i)} + \max_{t_j \in succ(t_i)} \left(C(t_i, t_j) + rank_b(t_j) \right) \quad (۱)$$

در این رابطه) $\overline{W(t_i)}$ میانگین هزینه محاسباتی وظیفه ti، $C(t_i, t_j)$ مقدار هزینه ارتباطی بین وظایف ti وtj و $rank_b(t_j)$ رتبه رو به بالای تاخر زیر وظیفه ti است.

$$rank_t(t_i) = \max_{t_j \in pred(t_i)} \left(rank_t(t_j) + \overline{W(t_j)} + C(t_i, t_j) \right) \quad (۲)$$

در این رابطه) $rank_t(t_j)$ رتبه رو به پایین زیر وظیفه تقدم ti است.

$$Level(t_i) = \begin{cases} 0, & if\ t_i = t_{entry}; \\ \max\limits_{t_j \in pred(t_i)} \left(Level(t_j) \right) + 1, & otherwise \end{cases} \quad (۳)$$

در این رابطه) $Level(t_j)$ رتبه سطح تقدم زیر وظیفه t_i است.

Algorithm 1. Initial population
1: Initial three of the chromosomes by using three heuristic rank policies 2: for i=3 to PopSize-1 do 3: for j=0 to ChSize-1 do 4: generate a genes j ∈ $(0, ChSize - 1)$ randomly that not generated in pervious genes 5: move chromosome i from left to right in first valid place in the queue in order to has a valid topological order 6: end for 7: end for

الگوریتم ۱: مقداردهی جمعیت اولیه

۳-۴-تخصیص زیروظایف به پردازنده‌ها

در جمعیت اولیه تولیدشده هر کروموزوم باید دارای ترتیب اولویت معتبر باشد یعنی محدودیت‌های تقدم در کروموزوم نقض نشود. در روش پیشنهادی به‌منظور تخصیص وظایف به پردازنده‌ها از سیاست تخصیص پردازنده HEFT استفاده‌شده است [۸]. در این روش زیروظیفه با بالاترین اولویت از بین زیروظایف کروموزوم انتخاب و به پردازنده‌ای تخصیص داده می‌شود که زمان اجرای آن از سایر پردازنده‌ها کمینه باشد. این روش تخصیص پردازنده از سیاست زمان‌بندی بر اساس درج استفاده می‌کند. در ادامه این بخش به تشریح این روش پرداخته‌شده است. زودترین زمان شروع زیر وظیفه ti بر روی پردازنده pk به‌صورت $EST(t_i, p_k)$ نمادگذاری می‌شود و از رابطه (۴) به دست می‌آید.

[21] .Seeding

www.SID.ir

Archive of SID

$$EST(t_i, p_k) = max\left\{ \max_{t_j \in pred(t_i)} \left(AFT(t_j) + C(t_j, t_i) \right) \right\} \quad (۴)$$

زمان شروع واقعی زیر وظیفه ti بر روی پردازنده pk با $AST(t_i, p_k)$ نمادگذاری می‌شود و از رابطه (۵) به دست می‌آید.

$$AST(t_i, p_k) = max(EST(t_i, p_k), Avail(p_k)) \quad (۵)$$

که Avail (pk) زمانی است که پردازنده pk بیکار و آماده اجرای وظایف باشد. زودترین زمان پایان زیر وظیفه ti بر روی پردازنده pk با $EFT(t_i, p_k)$ نمادگذاری شده است و از رابطه (۶) محاسبه می‌شود.

$$EFT(t_i, p_k) = W(t_i, p_k) + EST(t_i, p_k) \quad (۶)$$

که $W(t_i, p_k)$ هزینه محاسباتی زیر وظیفه ti بر روی پردازنده pk است. زمان واقعی پایان زیر وظیفه ti بر روی پردازنده pk با $AFT(t_i, p_l)$ نشان داده می‌شود و از رابطه (۷) به دست می‌آید.

$$AFT(t_i, p_k) = \min_{1 \le l \le P} EFT(t_i, p_l) \quad (۷)$$

که pk در رابطه بالا مناسب‌ترین[۲۲] پردازنده برای زیر وظیفه ti می‌باشد. شبه کد تخصیص وظایف به پردازنده‌ها در الگوریتم ۲ در تشریح شده است.

Algorithm 2. Assigning subtasks to processors function
1: Fill the priority queue with subtasks 2: while the priority queue is not empty do 3: Select the first subtask ti from the priority queue 4: for each processor pk in the processor set do 5: Compute $EFT(t_i, p_k)$ value using the insertion-based HEFT scheduling policy 6: Assign subtask ti to the processor pk that minimizes $EFT(t_i, p_k)$ 7: end for 8: Remove ti from the priority queue 9: end while 10: Return makespan= $AFT(t_{exit})$.

الگوریتم ۲) تخصیص وظایف به پردازنده‌ها

۳-۵-تابع برازندگی

مقدار برازندگی کروموزوم‌ها در تصمیم‌گیری اینکه کدام کروموزوم‌ها به نسل آینده انتقال یابند بسیار مهم است. زمان پایان واقعی آخرین زیر وظیفه در گراف جهت‌دار بدون دور یا به عبارتی گره خروجی گراف زمان اجرای کل برنامه می‌باشد. زمان اجرای کل برنامه از رابطه (۸) به دست می‌آید. که همان مقدار برازندگی کروموزوم i است.

$$makespan\ i = max\{AFT(t_{exit})\} \quad (۸)$$

مقدار برازندگی کوچک بیانگر کوتاه بودن زمان اجرای کل است. بنابراین بیشترین و کمترین مقدار برازندگی در میان جمعیت به ترتیب بدترین و بهترین کروموزوم‌ها هستند.

۳-۶-ترکیب

ترکیب، یک عملگر مهم در الگوریتم‌های ژنتیک است که به‌منظور ایجاد تغییر در کروموزوم‌های جمعیت مورداستفاده قرار می‌گیرد [۲۱]. همچنین عملگر ترکیب به تکامل جمعیت در الگوریتم‌های ژنتیک کمک می‌کند.

عملگر ادغام بیش از یک کروموزوم را برای تولید نسل جدید کروموزوم‌ها ترکیب می‌کند. کروموزوم جدید برخی ویژگی‌ها را از والد نخست و بقیه ویژگی‌ها را از والد دوم به ارث می‌برد. الگوریتم ۳ جزئیات عملگر ترکیب الگوریتم پیشنهادی را تشریح می‌کند.

Algorithm 3. Crossover operation
Input: Two parents from the current population. Output: Two new offsprings 1: Choose randomly a suitable crossover point i 2: Cut the father's chromosome and the mother's chromosome into left and right segments

[22]. Fittest

www.SID.ir

Archive of SID

3: Generate a new offspring, namely the son 4: Inherit the left segment of the father's chromosome to the left segment of the son's chromosome 5: Copy genes in mother's chromosome that do not appear in the left segment of father's chromosome to the right segment of son's chromosome 6: Generate a new offspring, namely the daughter 7: Inherit the left segment of the mother's chromosome to the left segment of the daughter's chromosome 8: Copy genes in father's chromosome that do not appear in the left segment of mother's chromosome to the right segment of daughter's chromosome 9: Return the two new offsprings

الگوریتم ۳: ترکیب تک نقطه‌ای

در الگوریتم ۴ عملگر تک نقطه‌ای به‌صورت چارچوب نگاشت-کاهش در آمده است به صورتی که هر عمکلر ترکیب به یک نگاشت تخصیص داده می شود.

Algorithm 4. Crossover Mapper
1: Run HDFS engine 2: Assign algorithm 5 to each mapper 3: Run the MapReduce job tracker 4: Start the calculation 5: Write the result into M blocks 6: Send the result to the shuffle phase 7: if all individuals have been processed successfully then Write best individual to global file in DFS 8: Copy and send the result to the Reducer

الگوریتم ۴: ترکیب تک نقطه‌ای با استفاده از چارچوب نگاشت-کاهش

۳-۷-جهش

عملگر جهش در الگوریتم ژنتیک برای نگه داشتن تنوع جمعیت بوسیله تغییر کروموزوم بکار می‌رود. بعد از اعمال عملگر ترکیب به منظور اجتناب از همگرایی به بهینه محلی و ایجاد تنوع و گوناگونی درجمعیت با استفاده از عملگر جهش یک تعداد از کروموزومهای به دست آمده تغییر داده می‌شوند. همچنین دو ژن برای انجام عملیات جهش تعریف می‌شوند. الگوریتم ۵ جزئیات عملگر جهش الگوریتم پیشنهادی را تشریح می‌کند.

Algorithm 5. Mutation operation
Input: A randomly chosen chromosome. Output: A new chromosome. 1: Choose randomly a gene Ti 2: Find the first successor Tj member Succ(i)
3: Choose randomly a gene Tk in the interval [i+1, j-1] 4: if l < i for all Tl member Pred(k) then 5: Generate a new offspring by interchanging gene Ti and gene Tk 6: return the new offspring 7: else 8: Go to Step 1 9: end if

الگوریتم ۵) عملگر جهش

در الگوریتم ۶ عملگر جهش به‌صورت چارچوب نگاشت-کاهش درآمده است به صورتی که هر عملگر ترکیب به یک نگاشت تخصیص داده می‌شود.

Algorithm 6. Mutation Reduce
1: Check the HDFS engine 2: Assign the result of algorithm 6 to each Reducer 3: Run the MapReduce job tracker 4: Start the calculation 5: Write the result into M blocks 6: Check the termination 7: if all individuals have been processed successfully then Write best individual to global file in DFS 8: Send the result to the cluster

الگوریتم ۶) جهش با استفاده از چارچوب نگاشت-کاهش

۳-۸-انتخاب

در الگوریتم پیشنهادی به‌منظور انتخاب بعضی از کروموزوم-ها برای الگوریتم جستجوی محلی از روش انتخاب چرخ رولت با پذیرش اتفاقی[۲۳] استفاده‌شده است [۱۴]. این روش دارای پیچیدگی زمانی o(۱) می‌باشد. این الگوریتم انتخاب دارای دو مرحله اصلی است: در مرحله اول یک کروموزوم به‌طور تصادفی از میان جمعیت انتخاب می‌شود که این انتخاب با احتمال (۱/n) صورت می‌گیرد. در مرحله بعدی مقدار برازندگی کروموزوم ارزیابی‌شده و اگر کروموزوم انتخاب‌شده یکی از نخبه‌های جمعیت باشد برای جستجوی محلی انتخاب می‌شود در غیر این صورت کروموزوم انتخاب‌شده پذیرفته نمی‌شود و تابع انتخاب دوباره از مرحله

[23] .Roulette-wheel selection via stochastic acceptance

www.SID.ir

Archive of SID

۱ تکرار می‌شود. شبه کد روش انتخاب چرخ رولت با پذیرش اتفاقی در الگوریتم ۷ آورده شده است.

Algorithm 7. Roulette-wheel selection via stochastic acceptance
Input: A current population chromosome. Output: A selected chromosome 1:do Generate a random number R∈ [0,PopSize-1]; 2: if $\frac{makespan_R}{\text{makespan}_{min}} == 1$ then 3: return chromosome$_R$; 4: end if 6: while (founding a one of the fittest solution).

الگوریتم ۷: انتخاب چرخ رولت با پذیرش اتفاقی

۹-۳- شرط خاتمه

تفاوت اصلی بین تکامل طبیعت و مسئله در تکامل طبیعی می‌باشد که گونه‌ها در طبیعت تمایل به خاتمه ندارند ولی در حل کردن یک مسئله با توجه به بودجه مشخص‌شده فرآیند را بایستی متوقف کرد [۱۵]. روش‌های معمول مثل تنظیم یک محدودیت بر روی تعداد ارزیابی برازندگی‌ها یا زمان ساعت کامپیوتر و یا دنبال کردن تنوع و توقف آن موقع کم شدن تنوع در جمعیت برای بررسی شرط خاتمه الگوریتم‌ها وجود دارد [۱۲]. در الگوریتم پیشنهادی شرط خاتمه موقعی اتفاق می‌افتد که تمام کروموزوم‌ها یا به عبارتی راه‌حل‌ها به یک مقدار برازندگی یکسان همگرا شوند.

۴-نتایج شبیه سازی

در این قسمت روش پیشنهادی با دو الگوریتم MPQGA و جعفری نویمی پور و همکاران از جنبه‌های زمان کل اتمام کار و مصرف انرژی مقایسه شده است. نتایج آزمایش‌ها مقایسه الگوریتم‌ها با استفاده از گراف جهت‌دار بدون دور و همچنین گراف‌های جهت‌دار بدون دور تولیدشده تصادفی به‌دست‌آمده‌اند.

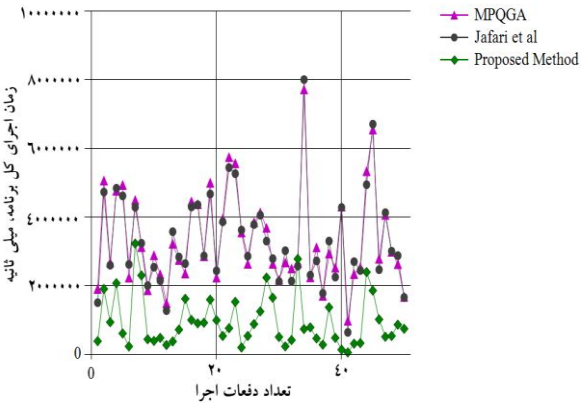
جدول ۴: مشخصات خوشه‌های سخت‌افزاری و نرم‌افزاری در هادوپ

نوع	سیستم‌عامل	پردازنده	حافظه
خوشه ۱	ubuntu-14.04.2	4-core, 3.07 GHZ	4G
خوشه ۲	ubuntu-14.04.2	4-core, 2.07 GHZ	4G
خوشه ۳	ubuntu-14.04.2	4-core, 2.07 GHZ	4G
خوشه ۴	ubuntu-14.04.2	4-core, 2.07 GHZ	4G

روش پیشنهادی و سایر الگوریتم‌ها در زبان برنامه‌نویسی جاوا در محیط MapR پیاده‌سازی شده‌اند و شبیه‌سازی و اجرای الگوریتم‌ها بر روی چهار خوشه در جدول ۴ با مشخصات زیر اجراشده است.

تولید گراف‌های جهت‌دار بدون دور تصادفی این امکان را می‌دهد که الگوریتم‌های مقایسه‌ای تحت شرایط مختلف بررسی و ارزیابی شوند. الگوریتم پیشنهادی با سایر الگوریتم‌ها ازلحاظ زمان اجرای کل برنامه و مصرف انرژی مقایسه شده است. زمان اجرای کل وظایف از لحظه ورود به سیستم تا زمان پاسخ دهی، چه مدت در سیستم میماند تا پاسخ خود را دریافت کند. مقدار زمان اجرای کل کارها بر حسب میلی ثانیه در نظر گرفته شده است.

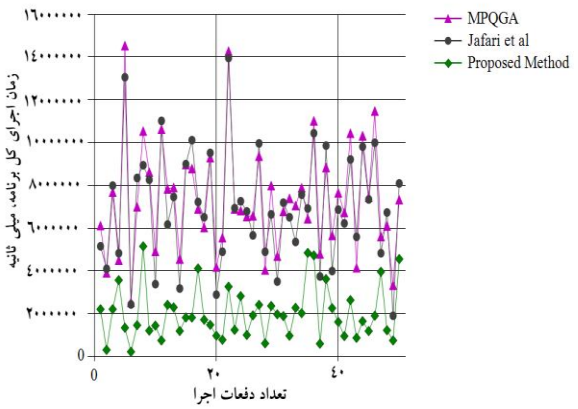
هزینه‌های محاسباتی و ارتباطی گراف‌ها به‌صورت تصادفی از میان یک بازه انتخاب می‌شوند. شکل ۳ زمان‌های اجرای کل گراف تصادفی را با ۵۰ زیر وظیفه و شکل ۴ زمان‌های اجرای کل گراف تصادفی را با ۱۰۰ زیر وظیفه را نشان می‌دهند. با توجه به نتایج نمودارها در شکل‌ها، مشهود است که روش پیشنهادی زمان‌های اجرای دو الگوریتم MPQGA و جعفری نویمی پور و همکاران را همواره بهبود می‌بخشد. بهبود زمان‌های اجرای کل گراف‌ها با افزایش تعداد زیر وظایف مشهود است.



شکل ۳: زمان‌های اجرای کل برنامه با ۵۰ کار

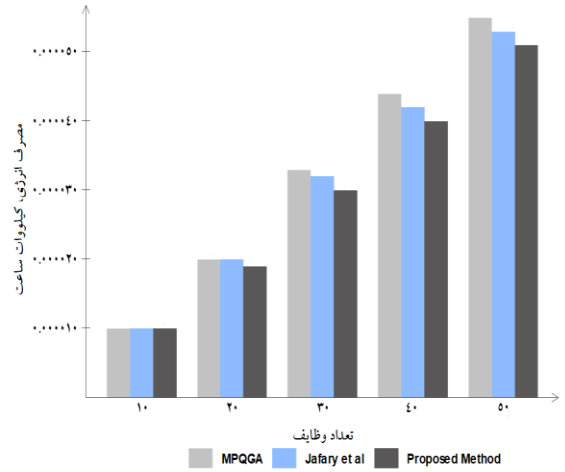
www.SID.ir

Archive of SID



شکل ۴: زمان‌های اجرای کل برنامه با ۱۰۰ کار

سدر روش پیشنهادی با استفاده از هادوپ و مدل نگاشت-کاهش کاهش انرژی مصرفی کل را در پی دارد. مصرف انرژی با دو الگوریتم MPQGA و جعفری نویمی پور و همکاران مقایسه شده است. شکل ۵ مصرف انرژی با تعداد وظایف مختلف وظیفه را نشان می‌دهند. محور افقی برابر با تعداد وظایف و محور عمودی برابر مصرف انرژی وظایف در نظر گرفته می شود. در هر باراجرا، تعداد وظایف از ۱۰ تا ۵۰ متغیر فرض شده است که در هر بازه که سیستم اجرا میشود، مقادیر متفاوتی به دست میآید که در شکل ۵ قابل مشاهده است.



شکل ۵: مصرف انرژی با تعداد وظایف مختلف

۵-نتیجه گیری و پیشنهاد کارهای آتی

زمان‌بندی در سیستم‌های رایانش توزیع‌شده یک مسئله مهم می‌باشد و الگوریتم‌های زمان‌بندی گوناگونی در این زمینه ارائه‌شده است. در این مقاله یک الگوریتم ژنتیک موازی با استفاده از چارچوب نگاشت-کاهش برای وظایف ایستا در سیستم‌های رایانش ابری ارائه گردید. این الگوریتم با ترکیب الگوریتم‌های ژنتیک استفاده از الگوریتم HEFT برای تخصیص زیروظایف به پردازنده‌ها توسعه داده شد. نتایج بدست آمده بر روی یک گراف ساده و همچنین گراف‌های تولیدشده تصادفی حاکی از آن است که الگوریتم ارائه‌شده از دو الگوریتم MPQGA و جعفری نویمی پور و همکاران در زمینه زمان اجرای کل بهینه عمل کرده است. لازم به ذکر است که هزینه‌های محاسباتی و ارتباطی گراف‌ها به‌صورت تصادفی از بین یک بازه انتخاب می‌شوند از معایب این روش محسوب می‌شود. از طرف دیگر، نگاشت-کاهش برنامه نویسی سطح پایین است و برای بدست آوردن تابع نگاشت و کاهش باید کدهای زیادی استفاده کرد. در این مقاله مقایسه‌های الگوریتم پیشنهادی با سایر الگوریتم‌ها بر روی گراف‌های جهت‌دار بدون دور تصادفی و یک گراف ساده صورت گرفته است. به همین دلیل مطالعات بعدی در مورد الگوریتم پیشنهادی می‌تواند بر روی گراف‌های جهان واقعی مانند FFT[۲۴]، Molecular dynamics code و غیره صورت گیرد. در شیوه نگاشت-کاهش که از سوی شرکت گوگل در سال ۲۰۰۴ معرفی‌شده است به‌طور مختصر همپوشانی در قسمت فازهای نگاشت و ارتباط دیده می‌شود. متأسفانه این همپوشانی[۲۵] در نرم‌افزارهای نوشته‌شده به شیوه نگاشت-کاهش که دارای فاز ارتباط حجیم‌تری نسبت

منابع

1. Khemka, B., et al., Utility maximizing dynamic resource management in an oversubscribed energy-constrained heterogeneous computing system. Sustainable Computing: Informatics and Systems, 2015. 5: p. 14-30.
2. Arunarani, A.R., Manjula, D. and Sugumaran, V., 2019. Task scheduling techniques in cloud computing: A literature survey. Future Generation Computer Systems, 91, pp.407-415.
3. Zang, X., Sun, J., Zhao, J. and Zeng, W., 2018, May. Research Progress of Cloud Computing Task Scheduling Technology. In 2018 3rd International Conference on Automation, Mechanical

[۲۴] Fast Fourier transformation
[۲۵] Overlapping

www.SID.ir

Archive of SID

به فاز نگاشت می‌باشند تأثیری چندانی روی کاهش زمان پاسخ این نرم‌افزارها ندارد. اگر بتوان راهکارهایی ارائه دهیم که توانایی همپوشانی فازهای حجیم ارتباط و کاهش را در این‌گونه نرم‌افزارها را داشته باشد می‌توان زمان بحرانی اجرای برنامه را کاهش داد و همچنین از منابع در دسترس حداکثر استفاده را نمود.

tool. Communications of the ACM, 2010. 53(1): p. 72-77.
7. Khezr, S.N. and Navimipour, N.J., 2017. MapReduce and its applications, challenges, and architecture: a comprehensive review and directions for future research. Journal of Grid Computing, 15(3), pp.295-321.
8. Topcuoglu, H., S. Hariri, and W. Min-You, Performance-effective and low-complexity task scheduling for heterogeneous computing. Parallel and Distributed Systems, IEEE Transactions on, 2002. 13(3): p. 260-274.
9. Mateos, C., E. Pacini, and C.G. Garino, An ACO-inspired algorithm for minimizing weighted flowtime in cloud-based parameter sweep experiments.

, 2014. 270(0): p. 255-287.
13. Abed, I., et al., Optimization of the Time of Task Scheduling for Dual Manipulators using a Modified Electromagnetism-Like Algorithm and Genetic Algorithm. Arabian Journal for Science and Engineering, 2014. 39(8): p. 6269-6285.
14. Lipowski, A. and D. Lipowska, Roulette-wheel selection via stochastic acceptance. Physica A: Statistical Mechanics and its Applications, 2012. 391(6): p. 2193-2196.
15. Ong, B. and M. Fukushima, Genetic algorithm with automatic termination and search space rotation. Memetic Computing, 2011. 3(2): p. 111-127.
16. Hashem, Ibrahim Abaker Targio, Nor Badrul Anuar, Mohsen Marjani, Ejaz Ahmed, Haruna Chiroma, Ahmad Firdaus, Muhamad Taufik Abdullah et al. "MapReduce scheduling algorithms: a

www.SID.ir

Archive of SID

scheduling algorithm. In 2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud) (pp. 71-76). IEEE.

20. Panahi, V., & Navimipour, N. J. (2019). Join query optimization in the distributed database system using an artificial bee colony algorithm and genetic operators. Concurrency and Computation: Practice and Experience, e5218.

21. Mirjalili, S. (2019). Genetic algorithm. In Evolutionary Algorithms and Neural Networks (pp. 43-55). Springer, Cham.

22. Malik, M., Neshatpour, K., Rafatirad, S., Joshi, R. V., Mohsenin, T., Ghasemzadeh, H., & Homayoun, H. (2019). Big vs little core for energy-efficient Hadoop computing. Journal of Parallel and Distributed Computing, 129, 110-124.

review." The Journal of Supercomputing (2018): 1-31.

17. Keshanchi B, Souri A, Navimipour NJ. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: formal verification, simulation, and statistical testing. Journal of Systems and Software. 2017 Feb 1;124:1-21.

18. Shabestari, F., Rahmani, A. M., Navimipour, N. J., & Jabbehdari, S. (2019). A taxonomy of software-based and hardware-based approaches for energy efficiency management in the Hadoop. Journal of Network and Computer Applications, 126, 162-177.

19. Ashouraei, M., Khezr, S. N., Benlamri, R., & Navimipour, N. J. (2018, August). A new SLA-aware load balancing method in the cloud using an improved parallel task

www.SID.ir

Archive of SID

www.SID.ir