زمان بندی روندکاری مقید با مهلت سخت با استفاده از الگوریتم فرا ابتکاری

چکیده

برنامه ریزی کارآمد بخش مهمی‌از پردازش برنامه‌های علمی‌پیچیده در محیط‌های توزیع محاسباتی است. پیچیدگی محاسباتی هم از محیط ناهمگن و هم از ساختار برنامه می‌آید که معمولا به عنوان روند کاری که شامل وظایف مربوطه متفاوتی می‌شود، نشان داده می‌شود. تکنیک‌های شناخته شده‌ی بسیاری توسط گروه‌های علمی‌مختلف پیشنهاد شده است. محبوب‌ترین آنها در فن آوری هوشمند مبتی بر لیست حریص و یا الگوریتم‌های فرا ابتکاری است.در این مقاله قابلیت اجرای الگوریتم فرا ابتکاری از پیش توسعه یافته‌ی الگوریتم ژنتیک (GCA) برای سری‌های زمان بندی در روند کاری با محدودیت شدید زمانی را بررسی می‌کنیم.

کلمات کلیدی: زمانبندی، گردش کار، شبکه، الگوریتم ژنتیک، تکامل همزمان، HEFT

1. مقدمه

امروزه سیستم‌های محاسباتی پیچیده بر اساس شبکه، خوشه یا ابرهای محاسباتی نقش بسیار مهمی‌در تحقیقات علمی‌بازی میکنند، که معمولا از برنامه‌هایمرکب برای اهداف محاسباتی استفاده میکنند. برای اجرای این برنامه‌هادر محیط‌هایتوزیع آنها را به وظایف از هم جدا تقسیم میکنند، که می‌توانند در منابع مختلف با محدودیت سمت چپ بر روی وابستگی بین وظایف اجرا شوند. به طور رسمی‌این برنامه‌ها روندهای کاری نامیده می‌شود و توسط نمودار با وظایف تعریف شده بر روی گره‌ها و وابستگی‌های به عنوان لبه ارائه می‌شوند.

برنامه‌ریزی مناسب برای اجرای برنامه‌های مرکب، بر منابع موجود بخش مهمی‌از حل موثر مشکلات است، که برای ما امکان بررسی مکانیزم بهینه سازی فرآیند برنامه ریزی را به ارمغان می‌آورد. برای اهداف مختلف معیارهای بهینه سازی مختلفی را می‌توان مورد استفاده قرار داد، مانند کل زمان اجرا (makespan)، هزینه، بهره‌وری انرژی و غیره. برای محیط‌های ابر، هزینه محاسباتی اغلب مهمترین معیار است، زیرا کاربران باید برای مدت زمان استفاده از منابع هزینه پرداخت کنند. برای شبکه، محاسبه makespan روند کار مهم ترین است، چرا که نتایج اجرا منجر به پیشرفت تحقیقات و بسیاری از برنامه‌های کاربردی مرکب برای بسیاری از کاربرانی که منتظر فرصتی برای شروع اجرا هستند، می‌شود. در این کار، ما یک مشکل محاسبات فوری که مخصوصا محدودیتهای شدیدی بر زمان اجرا دارد، به نام مهلت سخت را مخاطب قرار داده ایم. برای مثال، اینها برای سیستم‌های جلوگیری از خطر جاری شدن سیل، زلزله، بیماری‌های همه گیر، آتش و غیره بسیار مهم هستند.

دو گروه اصلی از الگوریتم برای ساخت برنامه‌های اجرای روند کار استفاده می‌شوند. گروه اول اکتشافی مبتنی بر لیست است که شامل دو مرحله اصلی رتبه بندی کار و تخصیص منابع بر اساس این رتبه بندی است. یکی از الگوریتم هایی که بیشتر استفاده می‌شود الگوریتم اولین زمان پایان ناهمگن (HEFT) (عرب نژاد، 2013) با توجه به زمان محاسبه کوتاه و توانایی تولید راه حل‌های با کیفیت خوب است. در این کار ما این روش را برای موارد مهلت سخت اتخاذ میکنیم. گروه دوم الگوریتم‌های فوق ابتکاری است. الگوریتم‌های این گروه معمولا نیاز به زمان بیشتری برای اجرا نسبت به فن آوری هوشمند دارند، اما می‌توانند راه حل‌های بسیار بهتری پیدا کنند. محبوب ترین الگوریتم‌هادر این زمینه عبارت هستند از الگوریتم ژنتیک (GA)، بهینه سازی ازدحام ذرات (PSO)، الگوریتم جستجوی گرانشی (GSA)، بهینه سازی کولونی مورچگان (Yu J. R., 2008) (ACO) و به همین ترتیب. در کار ما، الگوریتم ژنتیک مرسوم (CGA) برای موارد مهلت سخت بهبود یافته است.

از سوی دیگر بهینه سازی را نه تنها می‌توان توسط جستجوی کارآمد نقشه ی وظایف در منابع محاسباتی بلکه با بهینه سازی منابع برای مجموعه خاصی از وظایف با استفاده از فن آوری‌های مجازی سازی که اجازه می‌دهد تا

دوباره تعادل پارامترهای پیکربندی منبع اصلی (پردازنده، حافظه و غیره) با توجه به نیازهای وظایف برقرار شود، انجام داد. در این کار ما این بهینه سازی را نیز در الگوریتم ژنتیک مرسوم جذب می‌کنیم.

2. کارهای مرتبط

امروزه، دو روش به طور گسترده پر استفاده وجود دارد که محدودیت سخت مهلت برای توسعه سیستم‌های زمان واقعی را در نظر می‌گیرد - الگوریتم‌های نرخ یکنواخت (RM) و مهلت نخست اولیه (EDF). در (Buttazzo، 2005) مقایسه‌ای جامع از این الگوریتم‌ها به منظور کشف نقاط قوت و ضعف هر دو الگوریتم و برطرف کردن تصورات غلط مربوط به آنها انجام شد. نشان داده شده است که اگر چه RM پیاده سازی ساده تری دارد، به طور کلی اغلب خواصی که به آن نسبت داده شده است را ندارد، مانند کنترل انحراف یا قابل پیش بینی بودن در شرایط اضافه بار. EDF، از سوی دیگر، اجازه استفاده بهتری از منابع وپاسخ به فعالیتهای نامتناوب را می‌دهد. با توجه به کار ما، مفهوم RM از اختصاص دادن اولویت برای انجام وظایف برعکس دوره‌های آنها، دراصلاح HEFT برای پشتیبانی مهلت پیاده سازی شده است.

در حال حاضر، تعداد زیادی از تحقیقات علمی به اجرای روندهای کاری بهینه سازی با توجه به یک یا چند مورد ازپارامترهای کیفیت خدمات (QoS) از جمله مهلت اختصاص داده شده است. به عنوان مثال، Yeoو Buyya در (یئو، 2005) رویکردی برای مسئولیت رسیدگی به توافق سطح خدمات (SLA) در محیطهای توزیع به منظور افزایش کاربرد پذیری خوشه محاسباتی ارائه نمودند. نویسندگان روشی برای وظایف برنامه ریزی با توجه به پارامترهای کیفیت خدمات تعریف شده توسط کاربران، مانند هزینه‌های محاسباتی و مهلت اجرا پیشنهاد دادند. نویسندگان کارایی الگوریتم در مقایسه با الگوریتم‌های قدیمی‌تر برای ویژگی‌هایQoS متفاوت را نشان می‌دهند، اما روش ارائه شده نمی‌تواند به طور مستقیم در کار ما استفاده شود زیرا نمی‌تواند چند روند کاری با مهلت‌های مختلف و گزینه‌های سخت را به حساب آورد.

ابریشمی و همکاران (ابریشمی، 2013) الگوریتمی‌برای برنامه ریزی، با درنظر گرفتن مهلت‌ها با توجه به مهلت و هزینه محاسبات ارائه و مقایسه نمودند. نویسندگان الگوریتم‌های خود را برای به حداقل رساندن هزینه اجرای روندهای کاری و همچنین حفظ مهلت‌ها هدف سازی نمودند. اشکال اصلی الگوریتم پیشنهادی آنها برای استفاده در کار ما این است که آنها تنها یک روند کاری با مهلت‌ها را در نظر گرفتند، حال آنکه در رویکرد ما چند روند کاری با مهلت‌های مختلف با ویژگی مهلت سخت را می‌توان استفاده نمود. با این حال در آینده برای مقایسه با راه حل مان، فرض بر تطبیق کارآمد ترین الگوریتم‌های توصیف شده (به عنوان مثال IC-PCP) برای مورد چند مهلت با گزینه مهلت سخت می نماییم .

در (Bochenina 2014)، Bochenina مطالعه‌ای مقایسه‌ای بر روی الگوریتم‌های زمان‌بندی مختلف برای سیستم‌های ناهمگن با توجه به مهلت انجام داد. نویسنده دو روش برای رفتار با محدودیت‌های زمانی اجرای روند کار پیشنهاد می‌کند، که قادر به محاسبه نمودن مهلت نرم روند کاربه منظور ساخت برنامه‌های زمانی مناسب است.

در (Nasonov D. یک.، 2014) نویسندگان سعی بر مواجهه با سناریوهای مهلت سخت از پیش تعریف شده سیستم هشدار دهنده از طریق بهینه سازی برنامه ریزی پس زمینه با پارامترهای قابلیت اطمینان منابع پذیرفته شده و همچنین تغییرات کم سناریو در حجم کاری هایی که می‌تواند به راحتی با طرح حاضر وفق داده شود، نمودند.

مائو و هامفری در (مائو، 2011) رویکردی برای بهینه سازی اجرای روندهای کاری با مدیریت ماشین‌های مجازی به صورت پویا در محیط ابری توصیف نمودند. هدف اصلی از این تحقیق به حداقل رساندن هزینه اجرا و همچنین سعی بر حفظ مهلت روند کار بود. این کار به پردازش جریان‌های کاری تنها، با مهلت نرم و در یک راه مقرون به صرفه مرتبط است، درحالی که ما سعی برای حفظ چند روند کاری با مهلت سخت داریم.

در (Nebro، 2009) نویسندگان الگوریتم ژنتیک سلولی مبتنی بر تعامل همسایگی پیشنهاد نمودند، که در آن هرکدام از اجزا تنها می‌توانند با همسایگان نزدیکشان در حلقه پرورش همکاری کنند. اگرچه مقایسه این روش در برابر NSGA-II و SPEA2 نتایج خوبی نشان داده است، کاربردپذیری این الگوریتم در سری روندهای‌کاری با امکان مهلت، موضوع تحقیقات آینده است.

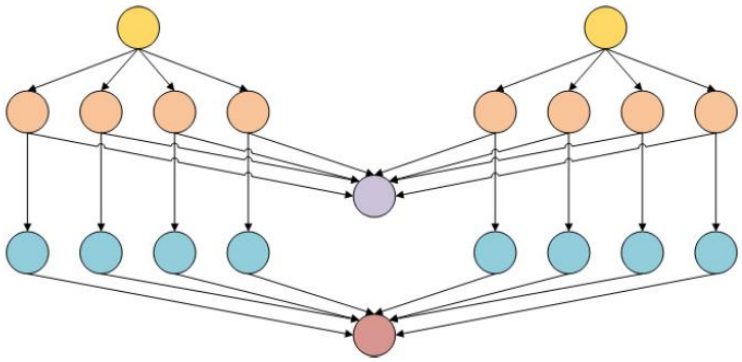
3. شرح مشکل

در این بخش ما مفروضات مان در مورد روندهای کاری، وظایف و منابع محاسباتی آنها، تعریف مساله و راه حل پیشنهاد مان برای آن را توصیف می‌کنیم.

3.1 روندهای کاری

معمولا روندهای کاری در قالب گراف غیر مدور و مستقیم (DAG) هستند که در آن گره‌ها برای نشان دادن وابستگی وظایف و لبه‌ها به وظایف (در اکثر موارد وابستگی داده ها) هستند. شرح مفصل این رویکرد در (Sinnen، 2007) داده شده است. در کار ما کل زمان اجرا (makespan) به عنوان پارامتر اصلی بهینه سازی، محدود شده با قید مهلت تعریف شده است. هر روند کاری می‌تواند مهلت زمانی سخت مشخص برای کاربرداشته باشد، که در آن روندهای کاری باید به همه معنا اجرا شوند. ما نمی‌توانیم اجرا را تا زمانی که همه ی مهلت‌ها در برنامه زمانی نگهداری شوند، شروع کنیم.

در این کار ما از تعدادی روند کاری مصنوعی تولید شده که برنامه‌های علمی واقعی از زمینه‌های مختلف علمی را تکرار می‌کند، استفاده می‌کنیم: Montage (نجوم)، CyberShake (علم زلزله)، Epigenomics (زیست شناسی)، Inspiral (فیزیک گرانشی) و SIPHT (زیست شناسی) (Bharathi، 2008). اینها اطلاعات دقیقی در مورد ساختار روند کار، زمان اجرای نسبی وظایف، ورودی و خروجی داده برای هر وظیفه و وابستگی داده فراهم می‌کنند. Bharathi و همکاران یک ژنراتور روند کار توسعه دادند، که اجازه تولید نسخه‌ای از روند کاری توصیف شده با تعداد مختلف از وظایف در قالب DAX (گراف غیر مدور مستقیم در XML) را می‌دهد. مثالی از ساختار روندکار را می‌توان در شکل 1 یافت.



شکل1: ساختار گردش کار مصنوعی (CyberShake)

زمان اتمام وظیفه در فاز اجرا توسط توزیع نرمال هر وظیفه t با مقدار متوسط m و واریانس d برآورد شده است. در برنامه ریزی زمان اتمام فاز، زمان ثابت و برابر مقدار میانگین در نظر گرفته شده است.

3.2 منابع محاسباتی

همه منابع محاسباتی یک قابلیت اطمینان معین دارند و ما باید این واقعیت را در نظر بگیریم تا احتمال شکست روند کار با مهلت را به کمترین مقدار برسانیم. برای این منظور ما سعی می‌کنیم، از طریق تکرار هر یک از وظایف این روندکاری با 99٪ قابلیت اطمینان اجرا شود. برای مثال، اگر وظایف را با قابلیت اطمینان 0.8 بر روی منابع نگاشت کنیم باید این کار را بر روی یکی دیگر از منابع با قابلیت اطمینان حداقل 0.95 تکرار کنیم تا قابلیت اطمینان اجرای 99٪ حاصل شود، زیرا P(AB)=P(A).P(B) که در آن P(A) میزان شکست از منابع 1 و P(B)- میزان شکست از منابع 2 است.

در این کار ما محیط محاسباتی را به منظورپیاده سازی تکنیک‌های مجازی سازی در نظر گرفتیم. به این معنی که به جای منابع محاسباتی واقعی، وظایف روند کاری را در تعدادی از منابع مجازی که بر اساس منابع واقعی است، نگاشت کردیم.

3.3 شرح مساله

همانطور که در بالا ذکر شد، زمانی که محدودیتهای مهلت سخت برای نتایج نهایی ضروری است و تکنیک‌های محاسبات فوری مورد نیاز است، موارد بسیاری وجود دارد، که ما را وادار به تغییر اولویت در طول بهینه سازی و انطباق الگوریتم موجود به محدودیت‌های قوی در ابتدا و تنها پس از آن بهینه سازی نمودن makespan توسط خودش با توجه به تغییرات پویا در طول فرآیند اجرا در محیط می‌کند. برای چنین الگوریتمی مهم‌ترین نکته دریافت نتایج در زمان ثابت است همچنین نیاز به پردازش چند روندکاری در یک مرتبه است، که در آن هر یک از آنها می‌توانند معیارهای فوری داشته باشند و باید در مقدار مشخصی از زمان تکمیل شوند، به عبارت دیگر مهلت اجرای سختی دارند. تحقیقات ما با هدف بررسی کاربردپذیری الگوریتم‌های فوق ابتکاری، به ویژه، الگوریتم ژنتیک مرسوم (Butakov، 2014)، برای روند کاری برنامه ریزی با مهلت سخت است.

4. راه حل پیشنهادی

در این بخش الگوریتم‌های فوق ابتکاری تکاملی و الگوریتم‌های اکتشافی اصلاح شده ارائه شده است.

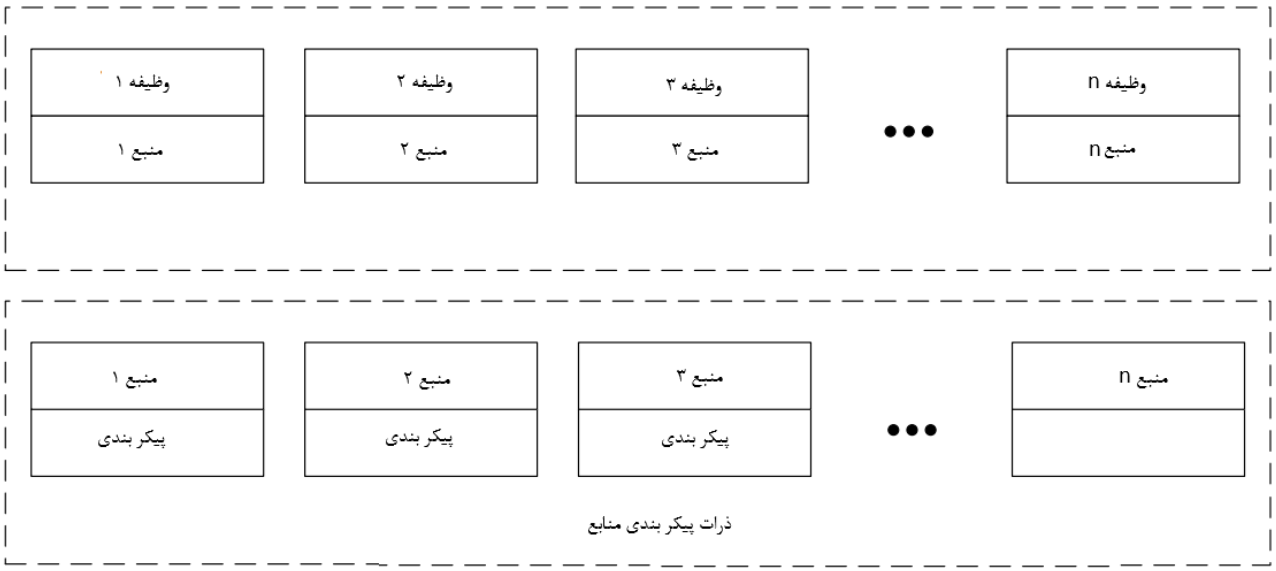
4.1 الگوریتم ژنتیک تکاملی مهلت سخت(HDCGA)

ما مجازی سازی را به عنوان راهی برای افزایش عملکرد زمان بندی در نظر گرفتیم. در (Butakov، 2014) مفهوم رویکرد مرسوم برای برنامه سازی در محیط‌های مجازی ناهمگن بیان شده است. این ایده در مراحل تکاملی به طور همزمان برای زمان بندی وظایف و محیط محاسباتی است. تکامل نگاشت وظایف به روش کلاسیک متقاطع، جهش و انتخاب توسط تابع تناسب سازمان یافته است. تکامل منابع توسط مدیریت منابع مجازی انجام می‌شود. در هر جامعه منابع واقعی به مجموعه‌ای از ماشین‌های مجازی تقسیم می‌شوند. این منابع تشکیل ذرات تکامل می‌دهند، که در شکل 2 نشان داده شده است . طرح اصلی الگوریتم HDCGA در شکل 3 نشان داده شده است. در هر مرحله از فرآیند پیکربندی و برنامه ریزی جمعیت ادغام می‌شوند. سپس برای هر جفت از ذرات زمان بندی ایجاد می‌کنیم، آن

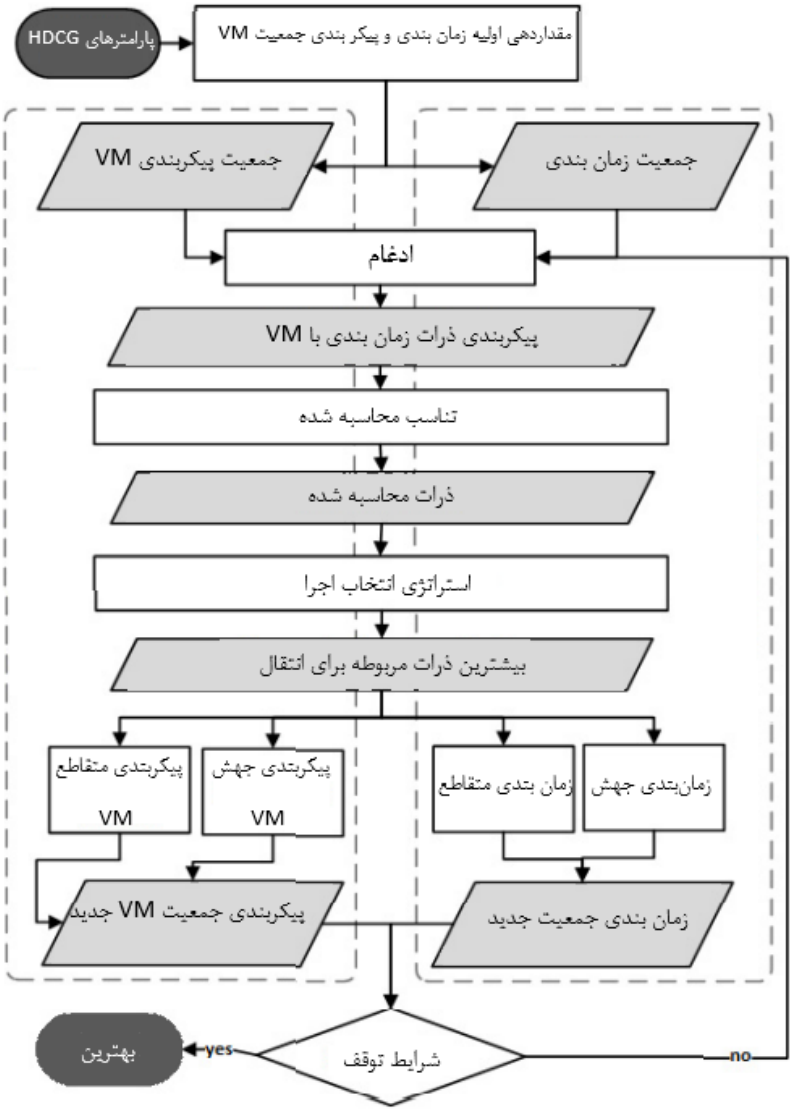
را با تابع تناسب ارزیابی می‌کنیم و استراتژی را بر اساس نتایج تابع تناسب اجرا می‌کنیم. در آزمایش‌های ما توجه شده بود که استفاده از این الگوریتم اجازه بهبود لیست اکتشافی، مانند HEFT، تا 84٪ را می‌دهد.

برای اجرای روندکاری با پشتیبانی مهلت سخت ، محاسبات تابع تناسب به روش زیر اصلاح شد: اگر برآورد makespan از روند کاری بیش از مهلت آن باشد، نتیجه تابع تناسب برای جفت ذره متناظر منجر به مجازات بسیار بالایی می‌شود، که باعث می‌شود شرکت آن جزء در بخشی از روند تکاملی غیر ممکن شود.

برای شبیه سازی محیط‌های پویا ما از مفهوم شرح داده شده در (Nasonov D. B.، 2014) استفاده نمودیم. اصل کلیدی این روش برنامه ترکیبی از روش‌های اکتشافی و فوق ابتکاری برای پردازش رویدادهای سیستم مانند نارسایی منابع است. درهنگام ظهور چنین رویدادی روش تغییر زمان بندی آغاز شده است. این روش از دو مرحله تشکیل شده است. در مرحله اول زمان بندی را با استفاده از HEFT تولید می‌کنیم و بلافاصله آن را بر روی محیط اعمال می‌کنیم. در همان زمان ما فاصله زمانی برای اجرای GA را محاسبه می‌کنیم. اگر GA توانست راه حل بهتر از HEFT پیدا کند، آن را بر محیط اعمال میکنیم.



شکل ۲: طرح ذرات تکاملی



شکل 3: طرح الگوریتم HDCGA

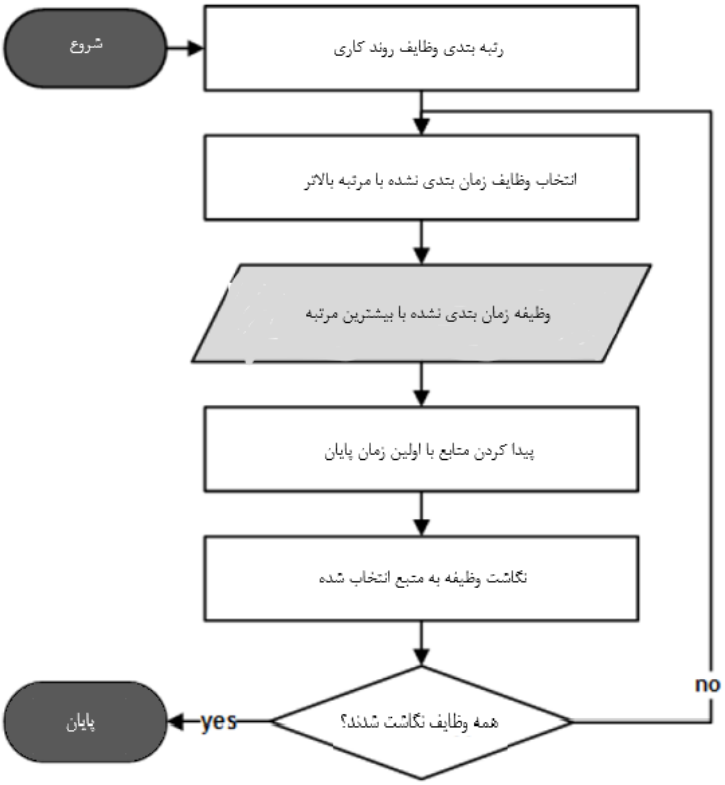
4.2 اولین پایان زمان مهلت ناهمگن (DHEFT)

الگوریتم رقابتی برای این مجموعه آزمایش‌ها تغییر یافته استHEFT - مهلت HEFT (DHEFT). الگوریتم DHEFT در شکل 4 نشان داده شده است. الگوریتم اصلی مهلت روند کار را به حساب نمی‌آورد بنابراین ما تغییراتی جزئی در رتبه بندی اعمال کردیم به این صورت که وظایف روند کار با مهلت کوچکتر رتبه بندی بسیار بالاتری می‌گیرند و زودتر پردازش می‌شوند. برای این منظور برای وظایف روندهای کاری با مهلت، اولویت را معکوس

با مهلت روند کاری اختصاص دادیم. در فاز رتبه بندی، وظایف اولویت هایی اضافه با توجه به فرمول زیر بدست می‌آورند:

$$r_{add} = inc^{p}$$

که در آن r_{add} - رتبه‌هایاضافی، P - اولویت ,وظیفه،inc- افزایش پارامتر می‌باشد. این روش حفظ مهلت را تضمین نمی‌کند اما حداقل مهلت روند کاری را در نظر می‌گیرد و به اندازه کافی برای نسل جمعیت اولیه برای الگوریتم‌های فوق ابتکاری خوب است.



شکل 4: طرح الگوریتم DHEFT

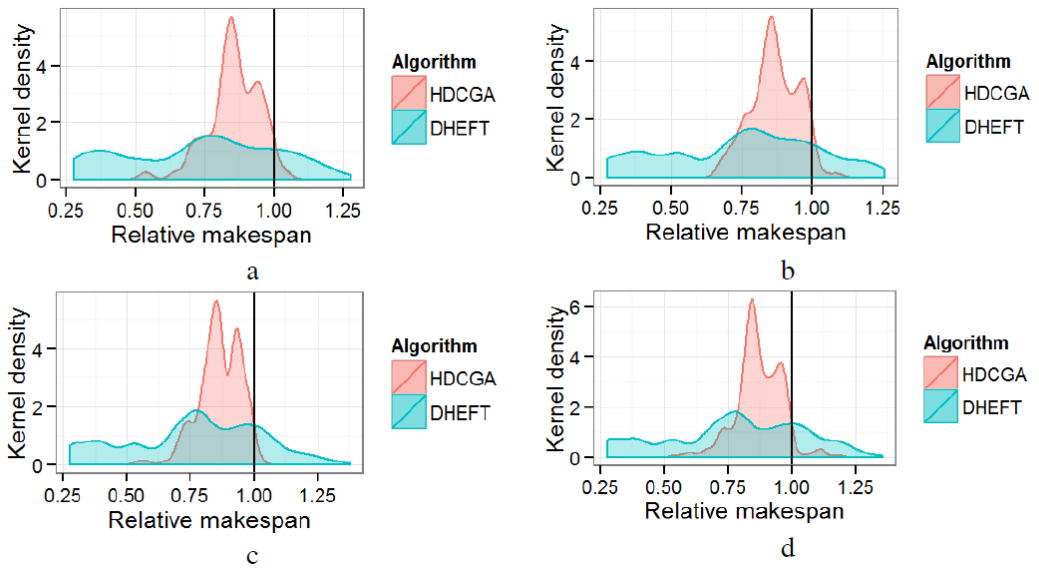
5. آزمایشات

برای قسمت آزمایشی پژوهش ما از شبیه ساز محیط محاسباتی استفاده نمودیم. این امر اجازه شبیه سازی اجرای روندهای کاری بر روی مجموعه‌ای از منابع با استفاده از استراتژی‌های برنامه ریزی مختلف با توجه به قابلیت

اطمینان منابع و مجازی سازی، میدهد. ما هزینه‌های محاسباتی برای هر وظیفه روند کاری را با ضرب نسبت مقدار زمان اجرا فایل و مقدار ثابت از پیش تعریف شده 20 تعیین نمودیم. همچنین هزینه انتقال بین منابع را یک ثابت بزرگتر از صفر فرض نمودیم. هر وظیفه تنها در یک منبع محاسباتی و در یک لحظه خاص از زمان محاسبه شده است. هر منبع محاسباتی دارای ارزش از پیش تعریف شده‌ای از قدرت در فلاپ‌ها است. ما مجموعه‌ای ثابت از منابع شامل چهار نمونه با قدرت محاسباتی شامل: 10، 15، 25، 30 داریم. برای HDCGA ما از پارامترهای زیراستفاده نمودیم: احتمال جهش - 0.8، احتمال متقاطع - 0.6، اندازه جمعیت - 50، تعداد تعاملات جمعیت - 200، تکرار شماره - 200.

برای مطالعات آزمایشی الگوریتم پیشنهادی ما یک سری آزمایشات برای روندهای کاری مصنوعی پیش رو انجام دادیم - Montage، CyberShake و Inspiral . در هر سری ما از چهار روند کاری هم نوع برای ایجاد روند کاری صف استفاده نمودیم. هر سری شامل چهار مجموعه آزمایش: یک، دو، سه و همه چهار روند کاری مهلت اجرای سخت دارند. پس از آن makespans حاصل از روندهای کاری دارای مهلت با توجه به مهلت هایشان نرمال شدند.

در شکل 5 کرنل چگالی makespans نرمال ارائه شده است. همانطور که مشاهده می‌شود، برای همه موارد چگالی HEFT کاملا به طور مساوی از تمام فضاهای راه حل توزیع شده است. جنبه ی منفی این توزیع این است که بیش از حد از مقدار مهلت تجاوز کرده است، که توسط خط عمودی سیاه و سفید در طرح به تصویر کشیده شده است. چگالی CGA، در طرف دیگر، نشان می‌دهد که الگوریتم تولید راه حل‌ایتولید کرده که تنها در موارد کمی‌از مهلت تجاوز کرده است. در جدول 1 نرخ تجاوز از مهلت الگوریتم‌ها برای تعداد مختلفی از مهلت‌ها ارائه شده است.



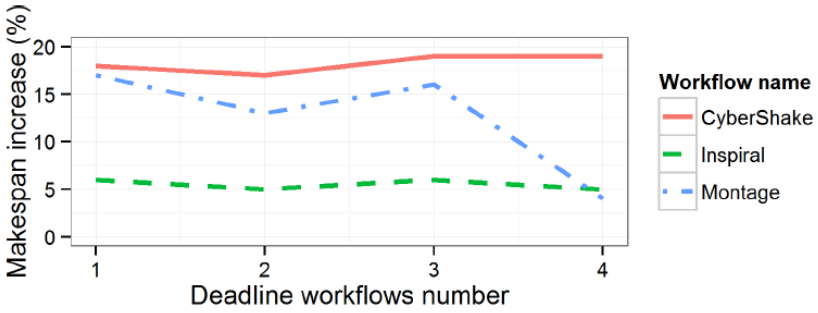
شکل 5: تراکم نسبی makespan برای موارد با یک (a)، دو (b)، سه (C) و چهار (د) مهلت

Number of deadlines	HDCGA deadline exceed rate (%)	DHEFT deadline exceed rate (%)
1	1.53	19.33
2	2.31	18.66
3	1.02	20
4	3.47	23.66

Table 1: Deadline exceed rate

جدول 1: نرخ تجاوز از مهلت

یکی دیگر از معیارهای بسیار مهم برای بررسی عملکرد الگوریتم برنامه ریزی makespan کلی است. افزایش Makespan از HDCGA بیش از DHEFT در شکل 6 نشان داده شده است. میتوان مشاهده نمود که HDCGA نسبت به DHEFT در همه ی موارد حداقل 5٪ بهتر است.



شکل 6: افزایش کلی makespan در HDCGA نسبت به DHEFT

6. نتیجه گیری

در این کار ما کاربرد الگوریتم‌های فوق ابتکاری، برای برنامه ریزی چند روند کاری با امکان مهلت سخت مقید در محیط محاسباتی ناهمگن با اجرای تکنیکهای مجازی سازی را مورد بررسی قرار دادیم. نتایج تجربی نشان دهنده بهره وری الگوریتم پیشنهادی HDCGA در مقایسه با DHEFT از هر دو نقطه نظر نگه داشتن مهلت سخت و تولید راه حل‌های با makespan کلی پایین تر است.

در طول آزمایش تعدادی مشاهدات جالب توجه کشف کردیم: نرخ تجاوز مهلت HDCGA در مورد سه مهلت کمتر از موارد دو و چهار مهلت است, کرنل توزیع چگالی برای همه آزمایشات چند کیفیتی است؛ در makespan روند کاری مونتاژ، افزایش شدید قطره‌ای از 15٪ به 5٪ در مورد چهار روند کاری وجود دارد. در کارهای آینده ما قصد انجام تحقیقاتی مفصل و بدست آوردن توضیحات این پدیده‌ها را داریم.

7. سپاسگذاری

این مقاله توسط بنیاد روسیه برای تحقیقات پایه پشتیبانی می‌شود- 15-29-07034 " فناوری محاسبات توزیع ابری برای شبیه سازی یک شهر بزرگ ".

References

Abrishami, S. M. (2013). Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds. *Future Generation Computer Systems 29.1*, 158-169.

Arabnejad, H. (2013). List Based Task Scheduling Algorithms on Heterogeneous Systems-An overview.

Bharathi, S. e. (2008). Characterization of scientific workflows. *Workflows in Support of Large-Scale Science*.

Bochenina, K. (2014). A Comparative Study of Scheduling Algorithms for the Multiple Deadline-constrained Workflows in Heterogeneous Computing Systems with Time Windows. *Procedia Computer Science 29*, 509-522.

Butakov, N. a. (2014). Co-evolutional genetic algorithm for workflow scheduling in heterogeneous distributed environment. *Application of Information and Communication Technologies (AICT)*.

Buttazzo, G. C. (2005). Rate monotonic vs. EDF: judgment day. *Real-Time Systems 29.1*, 5-26.

Mao, M. a. (2011). Auto-scaling to minimize cost and meet application deadlines in cloud workflows. *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis. ACM*.

Nasonov, D. a. (2014). Hybrid Scheduling Algorithm in Early Warning Systems. *Procedia Computer Science 29*, 1677-1687.

Nasonov, D. B. (2014). Hybrid Evolutionary Workflow Scheduling Algorithm for Dynamic Heterogeneous Distributed Computational Environment. *International Joint Conference SOCO'14-CISIS'14-ICEUTE'14*.

Nebro, A. J. (2009). Mocell: A cellular genetic algorithm for multiobjective optimization. *International Journal of Intelligent Systems 24.7*, 726-746.

Sinnen, O. (2007). Task scheduling for parallel systems. *Wiley-Interscience*, 108.

Son, J. H. (1999). Hard/soft deadline assignment for high workflow throughput. *Database Applications in Non-Traditional Environments*.

Yazir, Y. O. (2010). Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis. *Cloud Computing (CLOUD)*.

Yeo, C. S. (2005). Service level agreement based allocation of cluster resources: Handling penalty to enhance utility. *Cluster Computing*. IEEE International.

Yu, J. a. (2006). A budget constrained scheduling of workflow applications on utility grids using genetic algorithms. *Workflows in Support of Large-Scale Science*.

Yu, J. M. (2007). Multi-objective planning for workflow execution on grids. *Proceedings of the 8th IEEE/ACM International conference on Grid Computing*.

Yu, J. R. (2008). Workflow scheduling algorithms for grid computing. *Metaheuristics for scheduling in distributed computing environments*, 173-214.