زمان‌بندی وظایف در سیستم‌های چندپردازنده‌ای با بهره‌گیری از الگوریتم جستجوی گردابی

مهرداد زیادی[1]، هدیه ساجدی[2]

[1]دانشگاه آزاد اسلامی، واحد قزوین، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، قزوین، ایران، m.ziadi@qiau.ac.ir

[2] دانشکده ریاضی آمار و علوم کامپیوتر، پردیس علوم دانشگاه تهران، hhsajedi@ut.ac.ir

چکیده

امروزه سیستم‌های چندپردازنده‌ای کاربرد وسیعی در محاسبات موازی دارند. یک روش زمان‌بندی مناسب، در کاهش زمان اجرای وظایف و بهره‌وری منابع بسیار تأثیرگذار است. این زمان‌بندی باید به‌گونه‌ای انجام شود که بتواند زمان لازم برای اجرای کل برنامه را با در نظر داشتن زمان اجرای وظایف و بیکاری پردازنده‌ها، کمینه نماید. با توجه به NP کامل بودن مسئله زمان‌بندی گراف وظایف، رویکردهای مبتنی بر روش‌های قطعی در این زمینه کارا نخواهد بود؛ بنابراین استفاده از الگوریتم‌های فرا مکاشفه‌ای و الگوریتم‌های ترکیبی برای حل این مسئله مؤثر خواهد بود لذا بهره‌گیری از الگوریتم جستجوی گردابی، روش مناسبی جهت زمان‌بندی در سیستم‌های چندپردازنده‌ای می‌باشد. در این مقاله، مسئله را به کمک روش ترکیبی جدیدی که مبتنی بر الگوریتم جستجوی گردابی است حل نمودیم. روش ترکیبی با در نظر گرفتن اولویت زمان‌بندی انجام کارها بر اساس زودترین زمان شروع کار و در نظر گرفتن اجرای والد خود و ترکیب آن با الگوریتم جستجوی گردابی برای انتخاب بهترین جواب برای مسئله، می‌تواند به‌طور مؤثری برای مسائل زمان‌بندی در اکثر محیط‌های موازی مورد استفاده قرار گیرد. نتایج حاصل شده نشان می‌دهد الگوریتم پیشنهادی جواب بهینه‌ی زمان‌بندی را در مقایسه با سایر روش‌های متداول نتیجه می‌دهد.

واژه‌های کلیدی

الگوریتم جستجوی گردابی، زمان‌بندی وظایف، زودترین زمان شروع کارها، سیستم‌های چندپردازنده‌ای.

۱- مقدمه

قابلیت دسترسی به سیستم‌های چندپردازنده‌ای قدرتمند، با برقراری ارتباط از طریق لینک‌های پرسرعت مستلزم پیاده‌سازی یک بستر محاسباتی موازی برای برنامه‌های اجرایی با خواسته‌های گوناگون می‌باشد[1]. جهت بهره‌برداری کامل از چنین سیستم‌های محاسباتی برنامه‌های اجرایی آن‌ها به تعدادی کار تجزیه می‌شوند که ممکن است وابسته یا غیر وابسته به یکدیگر باشند[2,3] . طراحی بهینه تکالیف یعنی یکسان‌سازی و سپس زمان‌بندی مناسب در مجموعه‌های متنوعی از دستگاه‌ها، با هدف تنظیم کارآمدی کلی سیستم و گردآوری پتانسیل‌های احتمالی در نتایج توزیعی ازجمله جوانب بحرانی محسوب می‌شود. به‌طورکلی مسئله زمان‌بندی تکالیف در گراف جهت‌دار بدون دور(DAG[1]) مدل‌سازی شده است که در آن تکالیف برنامه اجرایی از طریق گره‌های گراف و وابستگی بین داده‌ها میان کارهای مختلف به تصویر کشیده شده است. هزینه برقراری ارتباط درروی یال‌ها و هزینه‌های محاسبات در نودها نشان داده شده است. مسئله زمان‌بندی موردنظر در اینجا یک مدل استاتیک است، زیرا ویژگی‌های مختلف برنامه اجرایی همچون زمان اجرای کلیه کارها در منابع مجزا و هزینه ارتباطی میان آن‌ها از قبل مشخص هستند. همچنین سیستم پردازش موازی به‌صورت یکنواخت و انحصاری فرض شده است، یعنی اگر یک کار برای پردازش به یک پردازنده تخصیص داده شد تا اجرای کامل آن نمی‌توان پردازنده را از آن کار پس گرفت.

در متون علمی برای حل مسئله زمان‌بندی کارها الگوریتم‌های زیادی مطرح شده است که جز مسائل NP کامل[2] هستند[4] و یافتن راه‌حل بهینه برای این مسائل مستلزم استراتژی‌های زمان‌بندی کارآمدتر است. افزایش سرعت و کارآمدی تنها زمانی قابل حصول خواهد بود که زمان‌بندی کارها در ماشین‌ها به‌طور صحیح و مناسب صورت گیرد زیرا می‌تواند به‌درستی همسانی سیستم را بکار گیرد.

در روش‌های کلاسیک، به دست آوردن جواب کاملاً بهینه یا یک زمان‌بندی کمینه، بسیار زمان‌گیر است و در بسیاری مواقع، اجرای تصادفی کارها به زمان بیشتری نیاز دارد، بنابراین از روش‌های مکاشفه‌ای استفاده می‌شود، که در روش‌های مکاشفه‌ای لزوماً جواب کاملاً بهینه به دست نمی-

[1] Directed Acyclic Graph

[2] NP-complete

آید ولی در یک بازه زمانی معقول، جوابی نزدیک به جواب بهینه به دست می‌آید. روش‌های مکاشفه‌ای بسیاری ارائه‌شده است که عبارتند از: GA[۳] [5,6] و SA[۴] و [7] VS[۵] و ACO[۶] و BA[۷] و [8] TS[۸] و [9] PSO[۹].

یکی از کارآمدترین روش‌های مکاشفه‌ای، الگوریتم جستجوی گردابی[۱۰] است. مطالعه پیشنهادی برحسب عملکرد متفاوت و پارامترهای سرعت نظیر کارآمدی، افزایش سرعت، طول زمان‌بندی و مدت‌زمان تکمیل کارها[۱۱] کارها[۱۱] با مقاله مطرح‌شده در [10] مقایسه می‌شود. بر اساس یافته‌های حاصل از آنالیز شبیه‌سازی گسترده[11,12,13,14]، الگوریتم پیشنهادی از الگوریتم‌های قبلی به‌طور قابل‌توجهی پیشی می‌گیرد.

در ادامه مقاله، ابتدا در بخش ۲ به معرفی مسئله زمان‌بندی پرداخته‌شده، سپس در بخش ۳ الگوریتم جستجوی گردابی و در بخش ۴ رویکرد پیشنهادی با بهره‌گیری از الگوریتم جستجوی گردابی ارائه‌شده است. در بخش ۵ پیاده‌سازی و نتایج شبیه‌سازی شرح داده شده و بخش ۶ شامل نتیجه‌گیری و کارهای آتی می‌باشد.

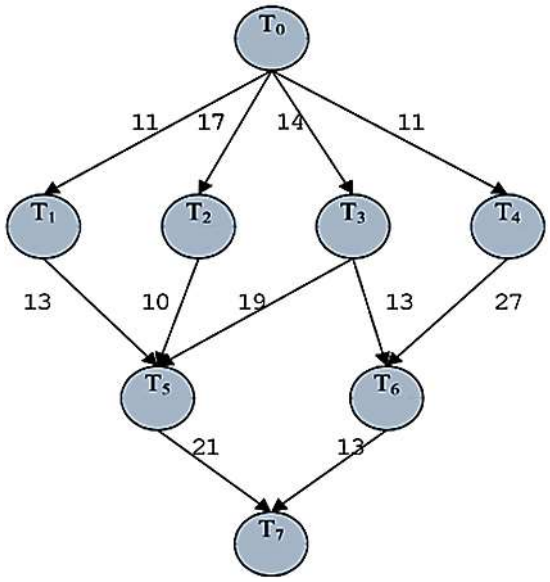
۲- مسئله زمان‌بندی وظایف

هدف یافتن یک زمان‌بندی بهینه برای اجرای گراف وظایف روی یک ساختار چندپردازنده‌ای می‌باشد به‌طوری‌که زمان کل اجرا یعنی زمان پایان آخرین واحد کاری کمینه گردد. فرض بر این است که بین همه پردازنده‌ها یک ارتباط مستقیم وجود دارد و یا به عبارتی توپولوژی اتصال پردازنده‌ها، یک گراف کامل می‌باشد[3,6].

ورودی مسئله به‌صورت گراف جهت‌دار بدون دور به نام گراف وظایف G=(V,E) نشان داده می‌شود. هر گره، عضوی از مجموعه V بوده و یک واحد کاری یا وظیفه از برنامه را نشان می‌دهد به‌طوری‌که وزن این گره‌ها مشخص‌کننده زمان اجرای واحد کاری مربوط خواهد بود. این گراف همچنین مجموعه‌ای از یال‌ها، یعنی E را در بردارد که بیانگر روابط پیش‌نیازی بین واحدهای کاری هستند، به این صورت که با داشتن یالی به‌صورت (t_i,t_j) تازمانی‌که t_i اجرایش را به پایان نرساند، t_j نمی‌تواند اجرایش را آغاز کند. در گراف وظایف، گره‌های بدون والد، گره ریشه و گره‌های فاقد فرزند، گره برگ نام دارند. شکل(۱)

یکی از پارامترهای بسیار تأثیرگذار در زمان‌بندی ایستا هزینه اجرای کارها و زودترین زمان شروع کار(EST[۱۲]) می‌باشد که مشخصات مرتبط با گراف شکل(۱) در جدول(۱) نمایش داده‌شده است.

[3] Genetic Algorithm
[4] Simulated Annealing
[5] Vortex Search
[6] Ant Colony Optimization
[7] Bees Algorithm
[8] Tabu Search
[9] Particle Swarm Optimization
[10] Vortex Search Algorithm
[11] Makespan
[12] Earliest Start Time



شکل ۱: گراف جهت‌دار بدون دور [10]

Task	Cost	EST	Cost	EST	Cost	EST
۰	۱۱	۰	۱۳	۰	۹	۰
۱	۱۰	۱۱	۱۵	۱۳	۱۱	۹
۲	۹	۱۱	۱۲	۱۳	۱۴	۹
۳	۱۱	۱۱	۱۶	۱۳	۱۰	۹
۴	۱۵	۱۱	۱۱	۱۳	۱۹	۹
۵	۱۲	۲۰	۹	۲۵	۵	۲۰
۶	۱۰	۲۲	۱۴	۲۴	۱۳	۱۹
۷	۱۱	۳۲	۱۵	۳۴	۱۰	۲۴

جدول ۱: هزینه‌های ماتریس [10]

۳- الگوریتم جستجوی گردابی

الگوریتم جستجوی گردابی در سال ۲۰۱۵ توسط دوگان و اولمز برای حل مسائل بهینه‌سازی ارائه شد[7]. الگوریتم جستجوی گردابی یک روش غیر جمعیتی است که در هر تکرار از الگوریتم، فضای اطراف بهترین پاسخ یافته شده را با توجه به شعاع، مورد جستجو قرار می‌دهد. سپس شعاع تا نیمه اجرای الگوریتم به‌صورت خطی و پس‌ازآن به‌صورت غیرخطی کاهش می‌یابد. به‌این‌ترتیب جستجوی گردابی در ابتدای عملکرد خود دارای رفتار پویشی[۱۳] و در انتهای اجرایش رفتاری انتفاعی[۱۴] را از خود نشان می‌دهد.

فرآیند جستجو در این الگوریتم با مقداردهی اولیه به مرکز اولیه جستجو μ_0 (پاسخ اولیه) و شعاع اولیه‌ی σ_0 بر اساس روابط (۱) و (۲) آغاز می‌شود.

[13] Exploration
[14] Exploitation

$$\mu_0 = \frac{upperlimit+lowerlimit}{2} \quad (۱)$$

$$\sigma_0 = \frac{\max(upperlimit)-\min(lowerlimit)}{2} \quad (۲)$$

در روابط فوق lowerlimit و upperlimit بردارهایی با انـدازه d×1 می‌باشند که d نشان‌دهنده ابعاد مسئله است. در ادامه‌ی اجـرای الگـوریتم جستجوی گردابی به تعداد N پاسخ به مرکزیت پاسخ اولیه μ_0 و شعاع σ_0 ایجاد می‌شود. چنانچه هرکدام از پاسـخ‌هـای تولیدشـده از محـدوده مجـاز تعریف‌شده خارج شوند با توجه به رابطه‌ی (۳) بازنمایی مجدد می‌گردند:

(۳)

$$S_k^i = \begin{cases} S_k^i \\ , lowerlimit^i \le S_k^i \le upperlimit^i \\ rand \times \left(upperlimit^i - lowerlimit^i\right) + lowerlimit^i , \\ otherwise \end{cases}$$

در رابطه‌ی (۳)، S نشان‌دهنده‌ی پاسخ تولیدشده در الگوریتم می‌باشد. در این رابطه، تابع rand یک عدد تصادفی با توزیع یکنواخـت تولیـد مـی‌کند. اندیس k شامل مقادیر ۱و۲و ... وN می‌باشد و i یکی از مقـادیر ۱و۲و ... وd را اختیار می‌کند.

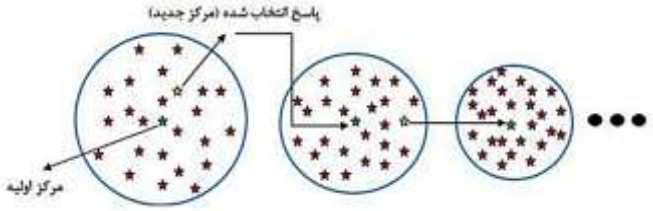
در مرحله بعدی الگوریتم، بهترین پاسخ تولید شده از میان پاسـخ‌هـای تولید شده انتخاب می‌شود. چنانچه این پاسخ از بهترین پاسخ قبـل (مرکـز جستجوی فعلی) برتر بود به‌عنوان مرکز جدید برای دور بعد انتخـاب مـی‌شود؛ بدیهی است که در صورت عـدم برتـری بهتـرین پاسـخ تولیـد شـده، فرایند جستجو در دور بعد حول مرکز فعلـی و البتـه بـا شـعاع همسـایگی کوچک‌تر ادامه می‌یابد. چگونگی کاهش شعاع جستجو r_t با توجه به روابـط (۴) تا (۶) تعیین می‌شود.

$$r_t = \sigma_0.\left(\frac{1}{x}\right).gammaincinv(x, a_t) \quad (۴)$$

$$a_t = a_0 - \frac{t}{MaxItr} \quad (۵)$$

$$gammaincinv(x.a_t) = \left(\int_0^x e^{-t} t^{a_t-1}\, dt \quad a_t > 0\right)^{-1} \quad (۶)$$

در روابط فوق t و MaxItr به ترتیب نشان دهنده‌ی شمارنده حلقـه جاری در الگوریتم و بیشترین تعداد تکرار حلقه می‌باشند. به‌منظـور ایجـاد یک پوشش کامل بر روی فضای جستجو، مقادیر a_0 و x به ترتیب برابر با ۱ و ۰.۱ در نظر گرفته می‌شوند. همچنین به‌منظـور کنتـرل سـرعت کـاهش شعاع جستجو حول بهترین پاسخ، از تابع معکوس ناکامل گاما که در روابط gammaincinv نمایش داده شده است، بهره گرفته می‌شود. به‌این‌ترتیب فضای پاسخ‌های موجود، در ابتدا به شکل یک دایره بزرگ و با دقت پـایین مورد جستجو قرار می‌گیرد. در مراحل بعدی از اجرای الگوریتم، ناحیه مورد جستجو به یک دایره کوچک‌تر که درون دایره‌های قبلی است محدود مـی‌شود. به دلیل ثابت بودن تعداد پاسخ‌های ایجادشده پیرامون پاسخ جاری در هر دور از اجرای الگوریتم، دایره کوچک‌تر به شکلی چگال تر مورد جستجو قرار می‌گیرد. روند کاهش شعاع در شکل(۱) و شبه کد الگوریتم جستجوی گردابی در جدول(۲) ارائه شده است.



شکل ۲: جستجوی چگال تر اطراف هر پاسخ با نزدیک‌تر شدن الگوریتم به تکرارهای آخر [7]

ورودی :

S_0 : جواب بهینه پیش‌فرض و t=0

تکرار :

}

// تولید مجموعه جواب‌های کاندید به کمک S بهینه انتخاب شده از مرحله قبل //

تولید $C(s_t)$

// جایگزینی جواب بهینه یافت شده به‌جای جواب بهینه قبلی از بین مجموعه جواب‌های کاندید C(s) //

$S_{t+1}=Select(C(s_t))$

t=t+1

{

خروجی :

تکرار مراحل تا یافتن جواب بهینه

جدول ۲: شبه کد الگوریتم جستجوی گردابی [7]

٤- رویکرد پیشنهادی

در این روش در ابتدا تعداد P پردازنده و n کار را در نظر گرفته و ترکیب‌های مختلف اختصاص n کار به P پردازنده را برحسب اولویت‌دهی به ارتفاع و زودترین زمان شروع کارها در نظر گرفته و سپس حالات تکراری و غیرممکن را حذف نموده و با در نظر گرفتن دو شرط مذکور، حالت‌های باقیمانده جهت یافتن بهترین حالت اختصاص کارها به پردازنده‌ها توسط الگوریتم جستجوی گردابی انجام شده است.

بدین منظور در ابتدا کمترین و بیشترین محدوده قابل‌قبول برای جواب‌های تولیدی از الگوریتم جستجوی گردابی تعریف شده و سپس در هر مرحله از اختصاص کارها به پردازنده‌ها یک حالت با در نظر گرفتن محدودیت‌های اعمال شده به‌عنوان راه‌حل پیشنهادی به الگوریتم جستجوی گردابی ارسال می‌شود و این الگوریتم با اجرای روش خود جهت یافتن بهینه‌ترین جواب، جواب ارسال شده را مورد مقایسه قرار داده و درصورتی‌که جواب ارسال شده کمینه باشد آن را به‌عنوان جواب فعلی پذیرفته و برای مقایسه در مراحل بعدی از آن استفاده می‌کند. این کار تا جایی ادامه پیدا می‌کند که تمامی حالات قابل‌قبول بررسی شده و جواب بهینه‌تر از سوی الگوریتم جستجوی گردابی پیشنهاد گردد.

الگوریتم فوق را برای تعداد متعددی از گراف‌ها که در محیط GTFF[15] ایجاد کردیم به اجرا گذاشتیم و نتایج بهبودیافته به لحاظ مدت‌زمان تکمیل شدن کارها را به دست آوردیم، همچنین استفاده از الگوریتم جستجوی گردابی در انتخاب حالت مناسب از بین حالت‌های ممکن نشان دهنده‌ی این است که این روش در مقایسه با روش‌های موجود مدت‌زمان کمتری برای اجرا نیاز دارد. در جدول(۳) شبه کد الگوریتم پیشنهادی و در شکل(۳) روال انجام کار ارائه شده است.

ورودی :

- ✓ گراف وظایف
- ✓ تعداد پردازنده
- ✓ تعیین محدوده‌ی اجرای الگوریتم جستجوی گردابی
- ✓ اختصاص مقدار پیش‌فرض به‌عنوان جواب بهینه برای الگوریتم جستجوی گردابی

پیش‌پردازش :

- ✓ یافتن تمام حالت‌های ممکن برای اختصاص n کار به P پردازنده
- ✓ حذف حالات تکراری و غیرممکن

تکرار :

{

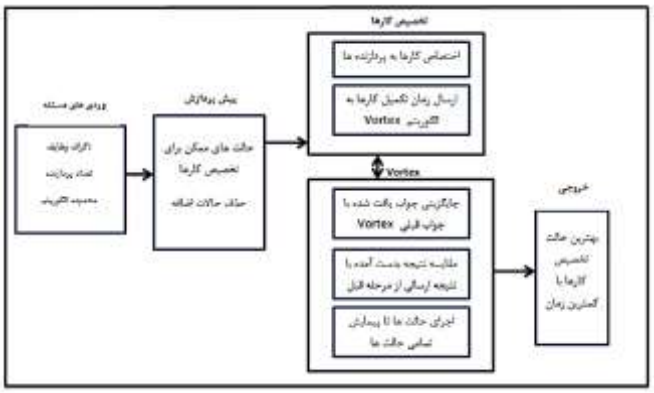
- ✓ اختصاص کارها به پردازنده‌ها بر روی یک حالت از حالات ممکن در هر تکرار
- ✓ ارسال زمان تکمیل‌شده کارها به‌عنوان ورودی برای الگوریتم جستجوی گردابی
- ✓ جایگزینی جواب یافت شده به‌جای جواب قبلی در الگوریتم جستجوی گردابی
- ✓ مقایسه‌ی نتیجه‌ی به‌دست‌آمده با نتیجه ارسال شده از مرحله قبل
- ✓ اجرای حالت‌ها تا پیمایش تمامی حالات

{

خروجی :

- ✓ یافتن بهترین حالت تخصیص کارها با کمینه‌ترین زمان اجرای کارها.

جدول ۳: شبه کد الگوریتم پیشنهادی



شکل ۳: دیاگرام الگوریتم پیشنهادی

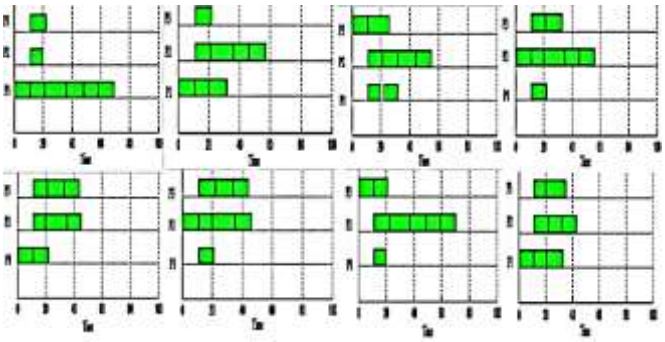
۵ – پیاده‌سازی و نتایج حاصل

رویکرد پیشنهادی در محیط شبیه‌سازی MATLAB پیاده‌سازی شده است، به‌طوری‌که در مرحله اول از اجرای رویکرد پیشنهادی تمامی ترکیبات مختلف از اختصاص n کار به P پردازنده توسط الگوریتم پیشنهادی به‌دست‌آمده و سپس حالت‌های تکراری و غیرقابل‌قبول حذف و نتایج باقی از بین ۵۶ حالت یافت می‌شود و تنها ۸ حالت قابل‌قبول با دو شرط ارتفاع گره و زودترین زمان شروع کار مربوط به هرکدام در نظر گرفته‌شده است که نتیجه آن در شکل(۴) نشان داده شده است. همچنین پارامترهای مورداستفاده در الگوریتم در جدول(۴) ارائه شده است.

نام پارامتر	مقدار
تعداد پردازنده	۳
تعداد کار	۸
کمترین محدوده الگوریتم جستجوی گردابی	۰
بیشترین محدوده الگوریتم جستجوی گردابی	۱۰۰
جواب پیش‌فرض اولیه	۵۵
زمان شروع کارها	T=0

جدول ۴: پارامترهای موردنیاز الگوریتم پیشنهادی

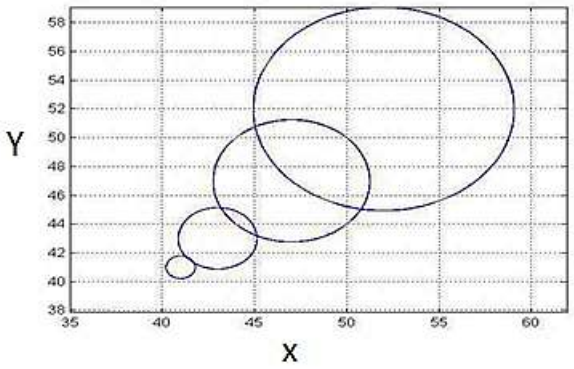
[15] Graph Task For Free



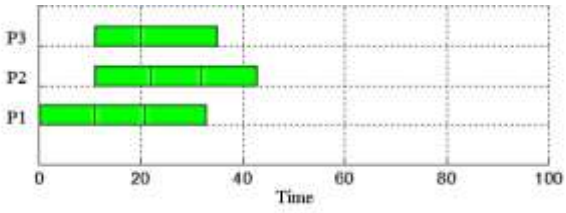
شکل ۴: حالت‌های ممکن و قابل‌قبول از بین تمام ترکیبات مختلف اختصاص کارها به پردازنده‌ها

متناسب با شکل فوق زمان اجرای لازم برای هر حالت به ترتیب ۷۰، ۶۹، ۵۷، ۵۶، ۵۵، ۴۶، ۴۵، ۴۳ به دست آمده است. این نتایج به‌صورت گام‌به‌گام تولید شده و در ۸ مرحله (تعداد گره) توسط الگوریتم پیشنهادی برای یافتن جواب بهینه مورد مقایسه و بررسی قرار گرفته‌اند. این روند تا پیمایش کامل حالت‌ها و به دست آوردن کمینه‌ترین جواب ادامه یافته است تا نتیجه‌ی مطلوب حاصل گردد.

شکل(۵) نشان‌دهنده خروجی حاصل از الگوریتم پیشنهادی می‌باشد که هر مرحله از اجرای رویکرد ترکیبی جستجوی گردابی را تا رسیدن به جواب بهینه نشان می‌دهد. همان‌طوری که از شکل پیداست در مرحله اول جواب یافت شده به مرکزیت جواب پیش‌فرض اجرا شده که از محدوده‌ی انتخابی بزرگ‌تری برخوردار است و در مراحل بعدی این محدوده جواب کمینه‌تر شده و ما را جهت دستیابی به جواب بهینه‌تر هدایت می‌نماید. نحوه‌ی اجرای این مراحل توسط شکل(۵) نشان داده شده و بهترین حالت نتیجه شده توسط اجرای الگوریتم پیشنهادی ، در شکل(۶) بازمان اتمام کارها در ۴۳ ثانیه نشان داده شده است.

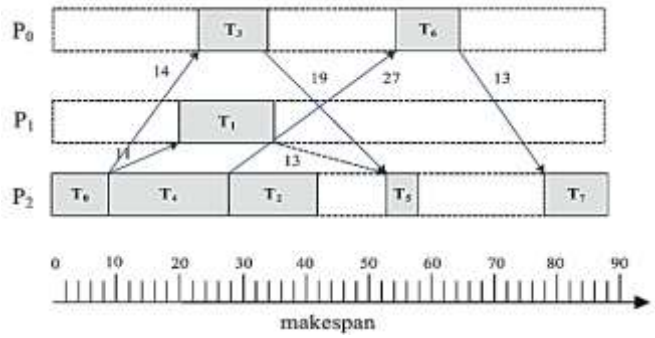


شکل ۵ : مراحل اجرای الگوریتم پیشنهادی و نحوه انتخاب جواب‌ها توسط الگوریتم جستجوی گردابی

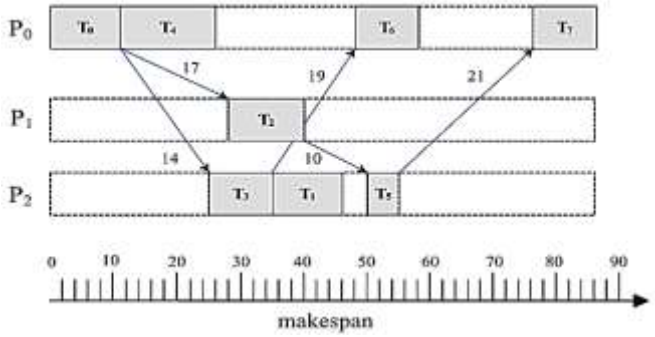


شکل ۶: بهینه‌ترین حالت ارائه شده برای تخصیص کارها به پردازنده‌ها توسط الگوریتم پیشنهادی

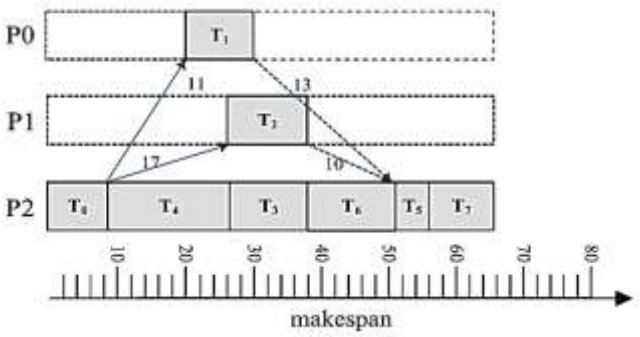
این در حالی است که نتایج به‌دست‌آمده از کارهای انجام شده در [10] به ترتیب برای الگوریتم‌های HEFT-T، CPOP، MPQGA، BAG مقادیر ۸۰، ۸۷، ۶۶، ۶۹ را به دست آورده است.



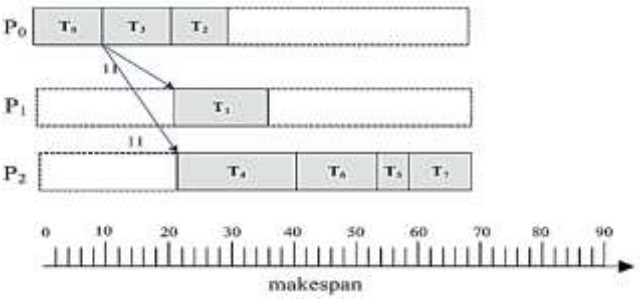
شکل ۷: استفاده از الگوریتم HEFT-T برای زمان‌بندی گراف وظایف با مدت‌زمان ۸۸ برای تکمیل کارها [10]



شکل ۸: استفاده از الگوریتم CPOP برای زمان‌بندی گراف وظایف با مدت‌زمان ۸۷ برای تکمیل کارها [10]



شکل ۹: استفاده از الگوریتم MPQGP برای زمان‌بندی گراف وظایف با مدت‌زمان 66 برای تکمیل کارها [10]



شکل ۱۰: استفاده از الگوریتم BGA برای زمانبندی گراف وظایف با مدت زمان ۶۹ برای تکمیل کارها [10]

مقایسه‌ی نتیجه راهکار پیشنهادی با نمودارهای فوق نشان‌دهنده این موضوع می‌باشد که راهکار پیشنهادی نتیجه‌ی بسیار مطلوبی ارائه نموده است.

٦- نتیجه‌گیری و کارهای آتی

نتایج حاصل نشان‌دهنده بهبود چشمگیر الگوریتم پیشنهادی نسبت به سایر الگوریتم‌هاست، همچنین رویکرد پیشنهادی به دلیل استفاده الگوریتم جستجوی گردابی به‌جای الگوریتم ژنتیک دارای مزایای بسیار زیادی می-باشد که از آن جمله می‌توان به پیچیدگی کمتر زمان اجرا و تعداد تکرارهای کمتر الگوریتم برای یافتن جواب بهینه اشاره کرد. الگوریتم ژنتیک برای رسیدن به جواب، نیازمند تولید تعداد بسیار زیادی نسل می-باشد و این در حالی است که الگوریتم جستجوی گردابی تنها به تعداد گره‌های گراف برای یافتن جواب بهینه به تکرار نیاز دارد.

جهت بهبود بیشتر این الگوریتم می‌توان الگوریتم پیشنهادی را با اعمال محدودیت‌های بیشتر و تعریف پارامترهای دیگر پیاده‌سازی نمود تا به نتایج مطلوب‌تر دست‌یافت.

مراجع

[1] S. Braun, D. Siegel, J. Maciejewski, A. Beck, N. Boloni, L. Maheswaran, M. Reuther, I. Robertson, P. Theys and D. Yao , "Characterizing Source Allocation Heuristics for Heterogeneous Computing Systems", Advances in Computers, vol. 63, pp. 91-128, 2005.

[2] U. Munir, Z. Li, S. Shi, Z. Zou and Q. Rasool, "A New Heuristic for Task Scheduling in Heterogeneous Computing Environment", Journal of Zhejiang University Science, vol. 9, no. 12, pp. 1715-1723, 2008.

[3]L. Xiaoping, J. Tianze and R. Rubén, "Heuristics for periodical batch job scheduling in a MapReduce computing framework", Information Sciences, vol. 326, pp. 119–133, 2016.

[4] I. Daoud and N. Kharma, "A High Performance Algorithm for Static Task Scheduling in Heterogeneous Distributed Computing Systems", Journal of Parallel and Distributed Computing, vol. 68, no. 4, pp. 399-409, 2008.

[5] D.E. Goldberg, "Genetic Algorithms in Search Optimization and Machine Learning", Addision-Wesley, 1989.

[6] O. Engin, G. Ceran and M.k.Yilmaz, "An Efficient Genetic Algorithm for Hybrid Flow Shop Scheduling with Multiprocessor Task Problems", Applied Soft Computing, vol. 11, pp. 3056-3065, 2011.

[7] B. Dog˘an and T. Ölmez, "A New Metaheuristic for Numerical Function Optimization: Vortex Search Algorithm, Information Sciences", vol. 293, pp. 125-145, 2015.

[8] J. Shuai and H. Zhi-Hua, "Path-relinking Tabu search for the multi-objectiveflexible job shop scheduling problem", Computers & Operations Research, vol. 47, pp. 11–26, 2014.

[9] M. S. Fakhar, A. R. Kashif, M. A. Saqib and H. Tehzeeb, "Non Cascaded Short-term Hydro-thermal Scheduling using Fully-informed Particle Swarm Optimization", Electrical Power and Energy Systems, vol. 73, pp. 983-990, 2015.

[10] X. Yuming , L. Kenli , H. Jingtong and L. Keqin, "A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues", Information Sciences, vol. 270, pp. 255-287, 2014.

[11] M. Iverson, F. Ozguner and G. Follen, "Parallelizing Existing Applications in a Distributed Heterogeneous Environment", in Proceedings of the 4th Heterogeneous Computing Workshop, pp. 93-100, 1995.

[12] H. EI-Rewini and G. Lewis, "Scheduling Parallel Program Tasks onto Arbitrary Target Machines", Journal of Parallel and Distributed Computing, vol. 9, no. 2, pp. 138-153, 1990.

[13] H. Mahamat and A. Azween, "A New Grid Resource Discovery Framework", The International Arab Journal of Information Technology, vol. 8, no. 1, pp. 99-107, 2011.

[14] H. Topcuglou, S. Hariri and Y. Wu, "Performance Effective and Low-Complexity Task Scheduling for Heterogeneous Computing", IEEE Transaction on Parallel and Distributed Systems, vol. 13, no. 3, pp. 260-274, 2002.