مرجان عبدیزدان

نویسنده اصلی

دانشگاه آزاد واحد ماهشهر

Marjanabdeyadan۶۹@yahoo.com

دکتر امیرمسعود رحمانی

استاد راهنما

دانشگاه واحد علوم و تحقیقات تهران

Rahmani۷۴@sr.iau.ac.ir

زمانبندی کارها در سیستم‌های چندپردازنده‌ای با استفاده از یک الگوریتم ژنتیک جدید اولویت بر اساس تعداد فرزندان

چکیده

مسأله زمانبندی ایستای کارها در سیستم‌های چندپردازنده‌ای به دلایل استفاده بهینه از پردازنده‌ها و همچنین صرف زمان کمتر، دارای اهمیت ویژه‌ای است. این مساله از رده مسائل سخت[۱] است و به دست آوردن جواب بهینه دارای پیچیدگی زمانی بالایی است، بنابراین برای حل این مسایل از روش‌های ابتکاری[۲] استفاده می‌شود. الگوریتم‌های ژنتیک، روش مناسبی جهت زمانبندی در سیستم‌های چند پردازنده‌ای است. در این مقاله الگوریتم ژنتیک جدیدی برای زمانبندی در سیستم‌های چندپردازنده‌ای ارایه می‌شود که اولویت[۳] زمانبندی انجام کارها، براساس تعداد فرزندان و نوادگان (Offspring) آنها است. نتایج نشان می‌دهد الگوریتم پیشنهادی جدید در زمان قابل قبول جواب بهینه زمانبندی را نسبت به دیگر روش‌های ژنتیک متداول به‌دست می‌آورد.

کلمات کلیدی

الگوریتم ژنتیک، زمانبندی کارها، سیستم‌های چندپردازنده‌ای، نوادگان.

۱- مقدمه

کارهای محاسباتی پیچیده نمی‌تواند در بازه زمانی قابل قبول بر روی یک پردازنده اجرا شوند و به‌همین دلیل باید به زیرکارهای کوچک‌تر تقسیم شوند و با زمانبندی مناسب و اختصاص دادن آنها به یک سیستم چندپردازنده‌ای زمان انجام کلیه زیرکارها کاهش یابد، به‌عبارتی زمان انجام آخرین کار کمینه شود. برای مدل‌سازی ریاضی مسأله زمانبندی ایستای کارها از گراف جهت‌دار غیرحلقه‌ای (DAG)[۴] استفاده شده است. در این گراف هر گره نشان دهندهٔ کار متناظر خود در مسأله زمانبندی است. وجود یک کمان از گره t_i به گره t_j به این معنی است که تا زمانی که اجرای کار t_i به اتمام نرسد، کار t_j نمی‌تواند شروع به اجرا کند.

مسأله زمانبندی در یک سیستم چندپردازنده‌ای [۱۱, ۱۷, ۲۱, ۲۲] و سیستم‌های توزیع‌شده ناهمگن (۱۴ ,۱۱) از رده مسائل سخت [۶, ۱۳] است. در روش‌های کلاسیک، به‌دست 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 ابتکاری استفاده می‌شود، که در روش‌های ابتکاری لزوماً جواب کاملاً بهینه به‌دست نمی‌آید ولی در یک بازه زمانی معقول، جوابی نزدیک به جواب بهینه به‌دست می‌آید. روش‌های ابتکاری بسیاری ارائه شده است که عبارتند از: [۱،۷،۱۳] Min- min و [۱] MCT و [۱،۷] MET و [۱،۷،۸] OLB و [۱۶،۱۸،۱۹،۲۳،۲۸] SA و [۱۸،۲۵،۲۷] GA و [۱،۷] Duplen و [۱،۷،۱۳] Max-Min [۲۴] DLS و [۱۴] LMT و [۱۵،۱۸،۱۹،۲۰] A× و [۹،۱۸] Tabu search و [۲۳] GSA. یکی از بهترین روش‌های ابتکاری، الگوریتم ژنتیک است [۱۲،۱۸،۲۵،۲۶،۲۷]. کارهای بسیاری در زمینه زمانبندی کارها در سیستم‌های چندپردازنده‌ای به روش الگوریتم ژنتیک [۱۱،۱۷،۲۱،۲۲] انجام شده است برخی از آنها از روش الگوریتم ژنتیک تصادفی [۲،۳،۴،۵] و برخی دیگر از روش الگوریتم ژنتیک و اولویت کارها براساس ارتفاع [۲۹] استفاده نموده‌اند. در این مقاله الگوریتم ژنتیک جدیدی براساس اجرای زودتر کارها با توجه به اولویت آنها بر اساس تعداد فرزندان و نوادگان آنها ارائه و شبیه‌سازی شده است.

ساختار باقیمانده مقاله به صورت زیر است: در بخش دوم الگوریتم زمانبندی اولویت دهی کارها بر حسب ارتفاع[۵] بیان شده و در بخش سوم روش جدید یعنی اولویت دهی کارها بر حسب تعداد نوادگان توضیح داده شده است. در بخش چهارم مراحل انجام الگوریتم ژنتیکی پیشنهادی به‌طور مجزا ارایه شده است و بخش پنجم اختصاص به شبیه‌سازی و نتایج حاصل از آن دارد و در بخش ششم نتیجه‌گیری آمده است.

---

۱ NP_Hard

۲ heuristic

۳ priority

۴ Diagram Acycle Graph

۵ Height

۲- الگوریتم زمانبندی اولویت‌دهی کارها بر حسب ارتفاع

هدف از زمانبندی در یک سیستم چندپردازنده‌ای انتساب n کار به m پردازنده در یک سیستم چندپردازنده‌ای است تا زمان اتمام اجرای آخرین کار در این سیستم کمینه شود. برای سادگی، اگر دو کار، بر روی دو پردازنده مختلف زمانبندی شوند هزینه ارتباطی برای ارسال داده‌ها بین دو کار صفر در نظر گرفته شده است. یکی از روش‌های پرکاربرد در زمانبندی استفاده از اولویت دهی اجرای کارها بر اساس ارتفاع آنها و به صورت زیر است:

اگر ترتیبی از یال‌های جهت‌دار از t_i به t_j وجود داشته باشد آنگاه t_i جد t_j و t_j فرزند t_i است. اگر $PRED(t_i)$ مجموعه کارهایی است که ماقبل t_i قرار گرفته‌اند آنگاه ارتفاع یک کار در گراف کار مطابق فرمول (۱) تعریف می‌شود.

فرمول (۱)

$$height(t_i) = \begin{cases} 0 & If\ PRED(t_i) = \phi \\ 1 + \max\limits_{t_j \in PRED(t_i)} height(t_j) & otherwise \end{cases}$$

تابع ارتفاع در حقیقت یک نوع اولویت اجرا بین کارها را بیان می‌نماید. اگر t_i یک جد برای t_j باشد یعنی t_i باید قبل از t_j انجام شود، آنگاه $height(t_i)<height(t_j)$. اگر ترتیبی از یال‌ها بین دو کار وجود نداشته باشد آنگاه هیچ نوع رابطه اولویتی بین کارها وجود ندارد و کارها می‌توانند به‌ترتیب دلخواه اجرا شوند.

الگوریتم تولید زمانبندی کارها بر حسب ارتفاع به‌صورت زیر است:

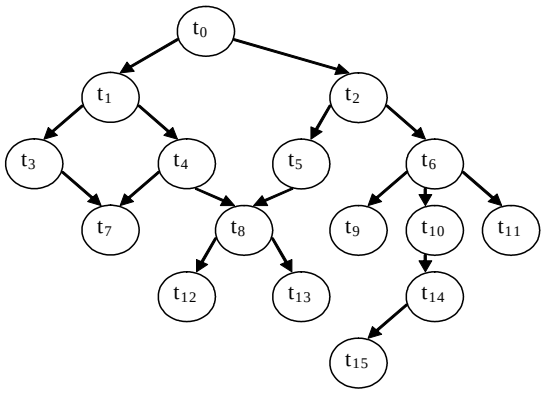
۱. کارها را به ترتیب افزایشی[1] بر حسب ارتفاع آنها در یک صف مرتب نمایید.
۲. مراحل ۳ و ۴ را به تعداد کارها تکرار کنید.
۳. یک عدد تصادفی r به‌طوری که $1 \le r \le m$، تولید کنید (m تعداد پردازنده است).

اولین کار را از صف کارهای مرتب شده انتخاب کرده و به پردازنده r ام تخصیص دهید و سپس آن را حذف نمایید. لازم به ذکر است که اتمام زمان کار برای هرپردازنده، FP[2] آن پردازنده محسوب می‌شود و برای هر کروموزوم، ماکزیمم اتمام زمان کار از بین پردازنده‌های p_1 تا p_m، FC[3] آن کروموزوم است. با استفاده از فرمول(۲) FP و با استفاده از فرمول (۳) FC محاسبه می‌شود.

۴. با تکرار به کار بردن الگوریتم تولید زمانبندی، یک جمعیت اولیه از گره‌های جستجو مورد نیاز تولید می‌شود.

با توجه به گراف وابستگی شکل۱، کارها بر حسب ارتفاع مطابق با جدول شماره ۱ مرتب شده و سپس بر اساس الگوریتم فوق برای یک سیستم چند پردازنده ای با p پردازنده زمانبندی شده‌اند.

شکل ۱: گراف وابستگی کارها



جدول۱: شماره و ارتفاع مربوط به هر کار

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}
۰	۱	۱	۲	۲	۲	۲	۳	۳	۳	۳	۳	۴	۴	۴	۵

جدول۲: شماره کارها و زمان مربوط به هر کار

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}
۳	۲	۴	۱	۱۰	۳	۶	۹	۷	۱۱	۵	۵	۸	۱۰	۱۵	۲

لازم به ذکر است که اتمام زمان کار برای هرپردازنده، FP[1] آن پردازنده محسوب می‌شود و برای هر کروموزوم، ماکزیمم اتمام زمان کار

---

[1] Ascending

[2] Fitness Processor

[3] Fitness Chromosome

از بین پردازنده‌های p_1 تا p_m، FC آن کروموزوم است. با استفاده از فرمول(۲) FP و با استفاده از فرمول (۳) FC محاسبه می‌شود.

فرمول (۲)

$$\text{(FP)} \quad \text{Fitness Processor } (P_j) = \max \{\text{completion-time}(P_j)\}$$

$$\text{Such } 1 \le j \le m$$

فرمول (۳)

$$\text{(FC)} \quad \text{Fitness Cromosome } (C_i) = \max \{\text{completion-time}(P_j)\}$$

$$\text{Such } 1 \le i \le npopsize,\ 1 \le j \le m$$

۳- اولویت‌دهی کارها بر حسب تعداد نوادگان

روش پیشنهاد شده در این مقاله، کارها را بر اساس یک اولویت دهی جدید یعنی تعداد فرزندان و نوادگان (offspring) هر کار به پردازنده‌ها ارجاع می‌دهد تا اجرا شوند. یعنی هر کار که تعداد فرزندان بیشتری دارد، باید زودتر زمانبندی شود. اولویت‌دهی براساس تعداد فرزندان و نوادگان بهتر از اولویت‌دهی براساس ارتفاع است، به‌دلیل این‌که ممکن است تعدادی از کارهای با ارتفاع یکسان برگ باشند و یا این که به برگ‌ها نزدیک باشند و تعدادی از آنها زیرشاخه‌های متعددی داشته باشد و از برگ‌ها دور باشد که در روش اولویت‌دهی براساس ارتفاع زمانبندی بدون توجه به این نکته انجام می‌شود در حالی که در روش اولویت‌دهی براساس تعداد فرزندان و نوادگان به این نکته توجه می‌شود و کارها با زیرشاخه‌ها و نوادگان بیشتر، اولویت بالاتری دارند و بالطبع شاخه‌های شلوغ‌تر یا کارهایی با تعداد نواده بیشتر، زودتر زمانبندی شوند و با اتمام آنها تعداد کارهای بیشتری (فرزندان و نوادگان آنها) امکان اجرا می‌یابند و زمانبندی بهتری به‌دست می‌آید. با توجه به گراف وابستگی، تعداد نوادگان یا offspring برای هر کار با استفاده از فرمول (۴) محاسبه می‌شود.

فرمول (۴)

$$offspring(t_i) = \begin{cases} 0 & ; if\ t_i\ has\ no\ child \\ \sum_{j=1}^{j = number\ of\ outgoing\ edges\ from\ task\ t_i} (1 + N(t_j)) & ; as\ t_j\ has\ the\ direc \\ & incoming\ edge\ from\ t_i \end{cases}$$

الگوریتم تولید زمانبندی کارها بر حسب تعداد نوادگان یا offspring به صورت زیر است:

۱. کارها را به ترتیب کاهشی[۲] بر حسب تعداد نوادگان آنها در یک صف مرتب کنید.

۲. کارها با تعداد offspring یکسان را در یک گروه مجزا قرار دهید و مراحل ۳ و ۴ را تا خالی شدن هر گروه، انجام دهید.

۳. یک کار از بین کارهای گروه به طور تصادفی انتخاب و از گروه حذف کنید.

۴. سپس آن کار را براساس روش EST[۳] به یکی از پردازنده‌های P۱ تا Pm اختصاص دهید به‌طوری‌که زمان شروع آن کار بر روی آن پردازنده از پردازنده‌های دیگر کمتر باشد.

اولویت‌دهی پیشنهادی در این مقاله بر اساس تعداد نوادگان با ذکر یک مثال بیان شده است: با توجه به گراف وابستگی شکل ۱ تعداد نوادگان یا offspring برای هر کار با استفاده از فرمول (۴) محاسبه می‌شود. با توجه به مثال مذکور offspring هر کار در جدول ۳ ظاهر شده است و در جدول ۴ کارها بر حسب offspring و به ترتیب نزولی مرتب شده است.

در ادامه کارها گروه‌بندی می‌شوند، به‌طوری که هر گروه، شامل کارها با offspring یکسان است. سپس اولین گروه .G (بالاترین offspring) انتخاب می‌شود و کارهای گروه به‌طور تصادفی انتخاب شده و سپس براساس روش EST به یکی از پردازنده‌های P_1 تا P_m که زمان شروع آن کار بر روی پردازنده مربوطه از پردازنده‌های دیگر کمتر باشد ارجاع می‌شود. این کار ادامه می‌یابد تا هنگامی که همهٔ کارها در گروه مربوطه به پردازنده‌ها ارجاع شود و این عملیات برای همهٔ گروه‌های بعدی تکرار می‌شود تا همه ی کارها انتخاب شوند (تمامیت) و هر کار فقط یک‌بار انتخاب شود (یکتایی). با این روش یک کروموزوم ایجاد می‌شود و سپس کارها با توجه به جدول ۲ و ۵ زمانبندی می‌شود و زمان پایان همه کارها در پردازنده‌ها محاسبه می‌شود که نتایج کار در

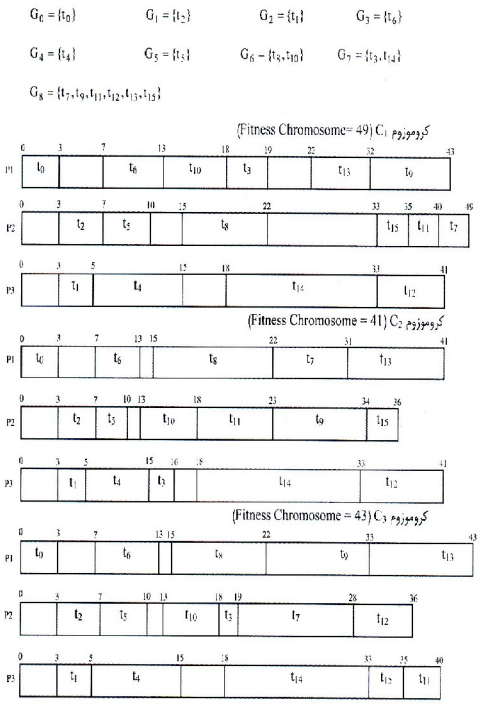
---

[۱] Fitness Processor

[۲] Descending

[۳] Earliest Start Time

شکل ۲ نشان داده شده است. مجدداً همین عملیات به تعداد جمعیت اولیه تکرار می‌شود تا نسل اولیه تولید شود.



شکل۲: روش تولید کروموزوم‌ها اولویت کارها براساس تعداد نوادگان

جدول۳: شماره و offspring مربوط به هر کار

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}
۱۵	۶	۱۰	۱	۴	۳	۵	۰	۲	۰	۲	۰	۰	۰	۱	۰

جدول ۴: کارها مرتب شده براساس offspring

t_0	t_2	t_1	t_6	t_9	t_5	t_8	t_{10}	t_3	t_{14}	t_7	t_4	t_{11}	t_{12}	t_{13}	t_{15}
۱۵	۱۰	۶	۵	۴	۳	۲	۲	۱	۱	۰	۰	۰	۰	۰	۰

جدول۵: شماره کار و زودترین زمان شروع هر کار در گراف وابستگی

t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}
۰	۳	۳	۵	۵	۷	۷	۱۵	۱۵	۱۳	۱۳	۱۳	۲۲	۲۲	۱۸	۳۳

۴- الگوریتم پیشنهادی جدید

الگوریتم پیشنهاد شده در این مقاله دارای مراحل زیر است:

۴-۱- اعتبارسنجی[1] کروموزوم‌ها

در این مرحله عملیات زیر صورت می‌گیرد:

۱. ابتدا تقدم کارها در هر پردازنده بررسی می‌شود تا اولویت‌ها کاملاً رعایت شده باشد. در تولید نسل اولیه بالطبع این ترتیب در هر پردازنده رعایت می‌شود ولی برای نسل‌های جدید ممکن است به علت اجرای عملیات ژنتیکی رعایت نشود که با اجرای مجدد مرحله اعتبارسنجی کروموزوم‌ها، جابجایی‌های لازم کارها در پردازنده‌ها انجام می‌شود تا سیر نزولی offspring و اولویت پیش‌نیازها رعایت شود.

۲. سپس شرط یکتایی و تمامیت بررسی می‌شود تا در هر کروموزوم هیچ کار تکراری نباشد و بیش از n کار وجود نداشته باشد وگرنه کروموزوم مربوطه حذف می‌شود.

۳. پس از اطمینان از وجود کروموزوم‌های معتبر شایستگی هر کروموزوم محاسبه می‌شود. به‌طوری که زمان اتمام اجرای کارها در هر پردازنده محاسبه می‌شود و ماکزیمم زمان اتمام اجرا در پردازنده‌های p_1 تا p_m به‌عنوان شایستگی (fitness) کروموزوم مربوطه منظور می‌شود.

ابتدا با توجه به جدول۲و شکل۱، زودترین زمان شروع هر کار محاسبه می‌شود و سپس با توجه به قرارگیری کارها بر روی هر

---

[1] Evaluation

پردازنده، زودترین زمان شروع هر کار بر روی هر پردازنده محاسبه می‌شود که این زمان، ماکزیمم مقدار بین زودترین زمان شروع کار مربوطه و پایان زمان اجرای کار قبلی است. درنهایت برای هر پردازنده زمان اتمام اجرای کارها محاسبه می‌شود (برطبق فرمول۲) و ماکزیمم زمان اتمام اجرا در پردازنده‌های p_1 تا p_m به عنوان شایستگی کروموزوم مربوطه منظور می‌شود(بر طبق فرمول۳).

سپس کروموزوم‌ها براساس شایستگی به ترتیب صعودی (از شایستگی بهتر به کمتر) مرتب می‌شوند به‌طوری که C_1 (اولین کروموزوم) دارای کمترین زمان اتمام اجرای کارها به‌عبارتی دارای بهترین شایستگی و $C_{npopsize}$ (آخرین کروموزوم) دارای بیشترین زمان اتمام اجرای کارها به‌عبارتی دارای بدترین شایستگی است.

۴-۲- تولید نسل جدید (Initiolazation)

پس از تولید نسل اولیه، برای تولید نسل‌های جدید از عملیات ژنتیکی شامل تبادل (crossover)، جهش (mutation) و متعادل‌سازی (load balancing) به‌صورت تکرار معین استفاده می‌شود. پس از پایان هر تکرار عملیات اعتبارسنجی کروموزوم‌ها مجدداً انجام می‌شود تا کروموزوم‌ها برحسب شایستگی مرتب شده و همچنین تمامیت و یکتایی نیز رعایت شود.

۴-۳- عمل انتخاب (Selection)

عمل انتخاب شامل دو قسمت است:

الف- انتخاب با روش چرخ‌گردان(Roulette wheel) جهت انتخاب دو کروموزوم: پس از مرتب شدن صعودی کروموزوم‌ها براساس fitness، سری چرخ‌گردانی براساس fitness تشکیل می‌شود به‌طوری که کروموزوم‌هایی که fitness کمتری به‌عبارتی شایستگی بهتری دارند، در انتهای سری چرخ‌گردان قرار می‌گیرند یعنی fitness کروموزوم‌ها از انتها ($C_{npopsize}$) تا ابتدا (C_1) جمع شده و سری چرخ‌گردان تشکیل می‌شود و به‌این طریق احتمال انتخاب کروموزوم‌ها با شایستگی بهتر، بیشتر است و سپس دو کروموزوم انتخاب می‌شوند.

ب- انتخاب با روش چرخ‌گردان جهت انتخاب کار: پس از اینکه کارها براساس offspring به‌صورت نزولی مرتب شدند یک سری چرخ‌گردان برحسب offspring کارها تشکیل می‌شود و سپس با تولید یک عدد تصادفی بین صفر تا آخرین عدد سری چرخ‌گردان، یک کار انتخاب می‌شود تا در عملیات ژنتیکی استفاده شود.

عملیات ژنتیکی شامل تغییراتی در نسل جاری و تولید نسل جدید به قرار زیر است: الف) عمل تبادل (crossover) ب) عمل جهش (mutation) ج) عمل متعادل‌کردن (load balancing).

۴-۴- عمل تبادل (Crossover)

در این مرحله ابتدا عدد تصادفی بین صفر و یک ایجاد می‌شود و اگر عدد تولید شده بزرگ‌تر یا مساوی با نرخ تبادل باشد عملیات تبادل به روش زیر انجام می‌شود.

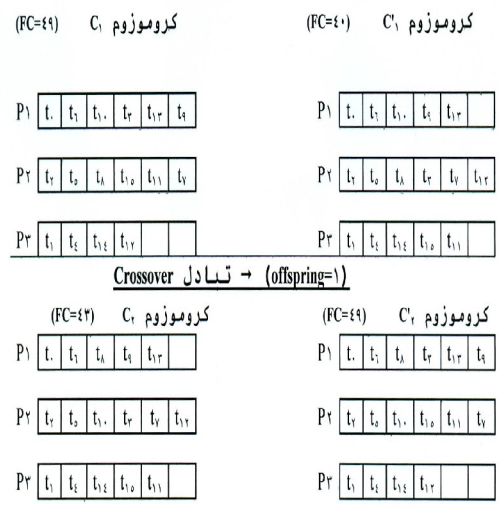
در این مرحله از عملیات ژنتیکی، از دو کروموزوم انتخابی (C_1 و C_2) در مرحله ی Selection کپی‌برداری شده (C'_1 و C'_2) و سپس عملیات زیر بر روی دو کروموزوم جدید انجام می‌شود.

کار انتخاب شده در مرحلهٔ Selection را در نظر گرفته و تمام کارهایی که offspring کمتر یا مساوی با offspring کار مورد نظر دارند، انتخاب می‌شوند و کارهای انتخاب شده بر روی هر پردازنده از کروموزوم انتخابی اول (C_1) با کارهای انتخاب شده بر روی پردازنده نظیر بر روی کروموزوم انتخابی دوم (C_2) مانند شکل ۳ جابه‌جا می‌شوند.

اگر فرض شود که درعمل Selection کروموزوم‌های C_1 و C_2 و کار t_{14} با offspring برابر یک انتخاب شود در حین عمل تبادل از کروموزوم‌های C_1 و C_2 به نام‌های C'_1 و C'_2 کپی‌برداری می‌شود سپس کارهایی که تعداد فرزندان کمتر یا مساوی با یک دارند یعنی کارهای $\{t_3,t_{14},t_7,t_9,t_{11},t_{12},t_{13},t_{15}\}$ بر روی دو کروموزوم انتخاب می‌شوند و سپس بر روی پردازنده‌های نظیر به نظیر این مجموعه کارها جابه‌جا می‌شوند.

لم۱: از آنجا که offspring هر کار در طول اجرا ثابت است و با انتخاب هر کار، offspring آن کار نیز ثابت است و با انتخاب کارها با offspring مساوی و کمتر از کار انتخابی، مجموعه کارهای یکسانی در هر دو کروموزوم انتخاب می‌شوند و چون کروموزوم‌ها قبل از تبادل یکتایی و تمامیت را رعایت می‌کردند بنابراین بعد از تبادل هم با جابجایی و تبادل این دو مجموعه یکسان، یکتایی و تمامیت را رعایت می‌کنند. همچنین در هنگام تبادل بایستی جاگیری کارهای مورد نظر با توجه به نزولی بودن برچسب‌ها انجام شود یعنی در تولید نسل جدید نیز تقدم رعایت می‌شود.

در شکل ۳ مشاهده می‌شود که کروموزم‌های C_1 و C_2 دارای شایستگی ۴۹ و ۴۳ و کروموزم‌های جدید C'_1 و C'_2 دارای شایستگی ۴۰ و ۴۹ هستند. بنابراین پس از عمل تبادل، به‌علت مبادله قسمت‌های پایانی رشته‌ها، نسل جدید تقریباً دارای همان محدوده از شایستگی‌هایی است که نسل قبل دارا است و نیز بهتر شده است. با تکرار عمل تبادل، محدوده شایستگی نسل قبلی تکرار می‌شود و همگرایی، زودتر حاصل می‌شود.



شکل۳: عملیات تبادل بر روی کروموزوم‌های C_1 و C_2 و تولید کروموزوم‌های جدید C'_1 و C'_2

۴-۵- عمل جهش (Mutation)

در این مرحله، ابتدا عدد تصادفی بین صفر و یک ایجاد می‌شود و اگر عدد تولید شده بزرگ‌تر یا مساوی با نرخ جهش باشد، عملیات جهش را به روش زیر انجام می‌شود.

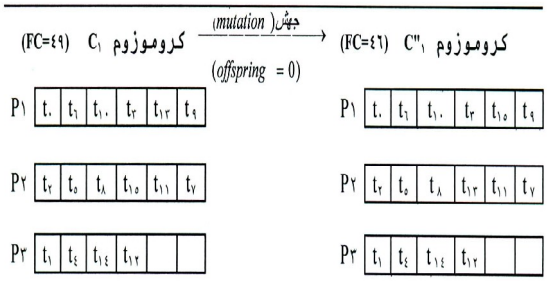
۱. در این مرحله از عملیات ژنتیکی از دو کروموزوم انتخابی (C_1 و C_2) در مرحله Selection کپی‌برداری شده (C''_1 و C''_2) و سپس بر روی دو کروموزوم جدید به‌طور مجزا عملیات زیر انجام می‌شود.

۲. بر روی کروموزوم انتخابی اول (C''_1)، کار انتخاب شده در مرحله ی Selection را در نظر گرفته و سپس کار انتخاب شده بر روی کروموزوم انتخابی اول را با کار دیگری که offspring یکسان با offspring کار انتخاب شده دارد، بر روی همان کروموزوم انتخابی اول مطابق شکل ۴ جابه‌جا می‌شود. همین عملیات بر روی کروموزوم انتخابی دوم (C''_2) نیز انجام می‌شود.

اگر فرض شود که در عمل انتخاب، کروموزم C_1 و کار t_{13} با offspring برابر صفرانتخاب شود جهت عمل جهش، از کروموزوم C_1 به نام C_1' کپی‌برداری می‌شود سپس کار دیگری به جز t_{13} که تعداد offspring مساوی با صفر دارد یعنی t_{15} بر روی کروموزوم C_1' انتخاب می‌شود و دو کار t_{13} و t_{15} با offspring یکسان، بر روی کروموزوم C_1' جابه‌جا می شوند.

لم۲: چون قبل از عمل جهش تمامیت و یکتایی رعایت شده است و پس از عمل جهش، با جابجایی دو کار با offspring یکسان، هیچ کاری از قلم نیفتاده است و هیچ کار دیگری هم به کروموزوم اضافه نشده است پس یکتایی رعایت شده است. چون کار مورد نظر از محل قبلی حذف و در محل جدیدی اضافه شده پس در کل مجموعه کارها تغییری حاصل نشده است و تمامیت نیز رعایت شده است. همچنین با جابجایی دو کار انتخاب شده بر روی یک کروموزوم، ترتیب offspring رعایت شده، بنابراین شرط تقدم نیز رعایت شده است.بنابراین کروموزوم حاصل از عمل جهش نیز معتبر است و نیازی به انجام عملیات اعتبارسنجی کروموزوم نیست.

در شکل ۴ مشاهده می‌شود که پس از عمل جهش در کروموزوم C_1، شایستگی کروموزوم جدید تغییر یافته و عمل جهش از همگرا شدن شایستگی کروموزوم‌ها و همچنین همگرایی زودهنگام پرهیز می‌کند و فضای بیشتری از راه حل‌ها را جستجو می‌کند.



شکل ۴: عملیات جهش بر کروموزوم C_1و تولید کروموزوم جدید C''_1

۴-۶- عمل متعادل‌کردن (Load balancing)

در این قسمت یک روش ابتکاری جدید به نام متعادل‌کردن برای کاهش زمان اتمام ارایه شده است که مراحل آن به‌صورت زیر است:

۱. در این مرحله از عملیات ژنتیکی از دو کروموزوم انتخابی (C_1 و C_2) در مرحلهٔ selection مجدداً کپی‌برداری شده (C'''_1 و C'''_2) و سپس بر روی دو کروموزوم جدید به‌طور مجزا عملیات زیر انجام می‌شود.

۲. ابتدا بر روی یکی از کروموزوم‌ها (مثلاً C'''_1) دو پردازنده‌ای که کمترین و بیشترین زمان اتمام اجرا را دارند از بین پردازنده های P_1 تا P_m انتخاب می‌شوند (به نام‌های P_{min} و P_{max}) ، سپس بر طبق فرمول (۵) تفاوت زمان اتمام اجرا (FP) بر روی دو پردازنده P_{max} و P_{min} محاسبه و حاصل بر دو تقسیم می‌شود.

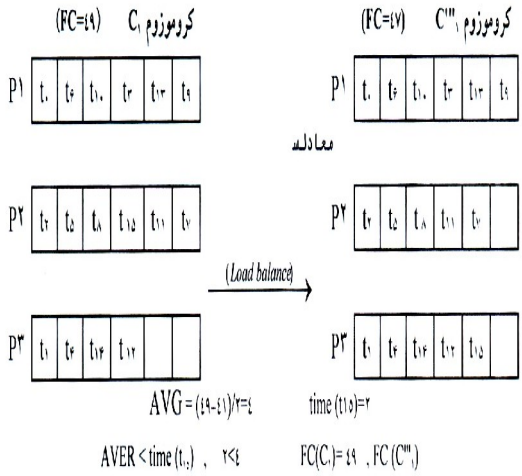
فرمول (۵)

$$AVG= (FP(P_{max})-FP(P_{min}))/۲$$

۳. سپس برای متعادل‌کردن حجم کار پردازنده‌ها مطابق شکل ۵، از روی پردازندهٔ P_{max} کار t_i دارای دو شرط زیر انتخاب می‌شود الف- مدت زمان اجرای آن کمتر یا مساوی AVG باشد و ب- از کارهای پایانی پردازنده P_{max} باشد تا وابستگی و Offspring کمتری داشته باشد و به علت وابستگی‌ها زمان اجرایش به تعویق نیفتد. اگر چنین کاری یافت نشود عمل مرحله ۴ یا متعادل کردن انجام نمی‌شود.

۴. سپس کار t_i از پردازندهٔ P_{max} حذف می‌شود و کار t_i به روی پردازندهٔ P_{min} در محل مناسب با رعایت تقدم ها و ترتیب نزولی offspring ها منتقل می‌شود. بالطبع هدف به دست آوردن fitness کمتر (شایستگی بهتر) است که با خارج کردن کار t_i از روی P_{max} طبعاً زمان اتمام اجرای آن کاهش یافته و چون fitness هر کروموزوم به $FP(P_{max})$ وابسته است، پس fitness کروموزوم $C'''_۱$ کاهش می‌یابد.

در عمل متعادل‌کردن و بر طبق لم ۲، خاصیت یکتایی و تمامیت کاملاً رعایت می‌شود.



شکل۵: عملیات متعادل‌کردن بر کروموزوم $C_۱$ و تولید کروموزوم جدید $C'''_۱$

در شکل ۵ مشاهده می‌شود که در کروموزوم $C_۱$، $P_{max} = P_۲$ و $P_{min} = P_۳$ است.

همچنین $(FP(P_{max})-FP(P_{min}))/۲= (۴۹-۴۱)/۲=۴$, AVG =۴است پس $Execution_Time(t_{۱۵})<AVG$ است و به‌این صورت کار $t_{۱۵}$ به پردازنده $P_۳$ منتقل می‌شود. با اجرای عمل متعادل‌سازی مشاهده می‌شود که پایان زمان اجرای کارها در کروموزوم $C_۱$، کاهش یافته و شایستگی نیز بهتر شده است.

۵- نتایج شبیه‌سازی

به منظور ارزیابی الگوریتم پیشنهادی یک مجموعه شبیه‌سازی با استفاده از نرم افزار Visual Basic.net version ۲۰۰۵ بر روی یک کامپیوتر Pentium IV دارای پردازنده AMD ۲/۸ مگاهرتز با ۵۱۲ مگا بایت حافظه RAM انجام شد.

مقادیر پارامترهای تعداد جمعیت اولیه برای الگوریتم‌ها به ترتیب برابر با تعداد کارها و تعداد تکرار (تعداد نسل) کمتر از ۱۰۰۰ و مقادیر نرخ تبادل و نرخ جهش برای الگوریتم‌ها به ترتیب برابر ۰،۷ و ۰،۰۵ انتخاب شده‌اند.

جهت ارزیابی بهینه بودن الگوریتم پیشنهادی نسبت به الگوریتم‌های قبلی، نتایج شبیه‌سازی با تعداد کارهای متفاوت و تعداد پردازنده‌های ثابت و درصد وابستگی کارها ۵۰٪ و مدت زمان اجرای کارها بین ۱ تا ۱۰۰واحد زمانی، در جداول۶و ۷و ۸و ۹ نشان داده شده است.

با توجه به جدول ۶ مشاهده می‌شود در حالتی که تعداد پردازنده‌ها برابر با ۳ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۶/۲۵ ٪ بهینه شده است.

جدول۶ : نتایج شبیه‌سازی سه الگوریتم ژنتیکی با ۳ پردازنده

تعداد کارها	۳۰	۵۰	۷۰	۹۰
تعدادپردازنده	۳	۳	۳	۳
اتمام زمان روش تصادفی	۱۲۷۵	۱۶۳۹	۲۰۹۱	۲۷۳۸
اتمام زمان روش ارتفاع	۱۱۵۸	۱۴۶۵	۱۹۶۹	۲۸۰۶
اتمام زمان روش پیشنهادی	۱۰۰۲	۱۴۴۸	۱۹۱۵	۲۵۴۹
درصد بهینگی	۱۳	۱	۲	۹

با توجه به جدول ۷ مشاهده می‌شود در حالتی که تعداد پردازنده‌ها برابر با ۵ است شایستگی اولویت بر حسب offspring

نسبت به شایستگی اولویت بر حسب height در حدود ۴/۲۵ ٪ بهینه شده است.

جدول ۷ : نتایج شبیه‌سازی سه الگوریتم ژنتیکی با ۵ پردازنده

تعداد کارها	۳۰	۵۰	۷۰	۹۰
تعدادپردازنده	۵	۵	۵	۵
اتمام زمان روش تصادفی	۹۶۶	۱۷۴۳	۲۷۲۹	۷۲۲۳
اتمام زمان روش ارتفاع	۱۰۰۳	۱۶۸۳	۲۳۹۵	۳۲۴۲
اتمام زمان روش پیشنهادی	۹۱۸	۱۶۳۰	۲۳۰۰	۳۱۹۲
درصد بهینگی	۸	۳	۴	۲

با توجه به جدول ۸ مشاهده می‌شود در حالتی که تعداد پردازنده‌ها برابر با ۷ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۲/۵ ٪ بهینه شده است.

جدول ۸ : نتایج شبیه‌سازی سه الگوریتم ژنتیکی با ۷ پردازنده

تعداد کارها	۳۰	۵۰	۷۰	۹۰
تعدادپردازنده	۷	۷	۷	۷
اتمام زمان روش تصادفی	۲۶۰۴	۳۸۳۶	۲۱۳۵	۳۴۹۸
اتمام زمان روش ارتفاع	۱۲۶۰	۱۷۹۵	۱۹۹۰	۳۳۰۳
اتمام زمان روش پیشنهادی	۱۲۲۲	۱۷۴۷	۱۹۸۱	۳۱۹۷
درصد بهینگی	۳	۳	۱	۳

با توجه به جدول ۹ مشاهده می‌شود در حالتی که تعداد پردازنده‌ها برابر با ۹ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۳/۵ ٪ بهینه شده است.

جدول ۹ : نتایج شبیه‌سازی سه الگوریتم ژنتیکی با ۹ پردازنده

تعداد کارها	۳۰	۵۰	۷۰	۹۰
تعدادپردازنده	۹	۹	۹	۹
اتمام زمان روش تصادفی	۸۵۵	۵۱۱۱	۲۳۱۴	۲۹۱۹
اتمام زمان روش ارتفاع	۸۵۶	۱۷۵۳	۲۱۹۱	۲۷۶۰
اتمام زمان روش پیشنهادی	۸۰۹	۱۷۲۰	۲۰۶۵	۲۷۴۲
درصد بهینگی	۵	۲	۶	۱

بررسی درصد بهینگی الگوریتم پیشنهادی، بیانگر این است که هر چه تعداد پردازنده‌ها کمتر شود، تاحدودی درصد بهینگی بیشتر می‌شود و این ویژگی، مزیتی برای الگوریتم پیشنهادی محسوب می‌شود.

همچنین جهت ارزیابی بهینه بودن الگوریتم پیشنهادی نسبت به الگوریتم‌های قبلی، نتایج شبیه‌سازی جداول قبلی، با تعداد کارهای ثابت و تعداد پردازنده‌های متفاوت و درصد وابستگی کارها ۵۰٪ و مدت زمان اجرای کارها بین ۱ تا ۱۰۰واحد زمانی، از جداول ۶و۷و۸و۹ نتایج زیر حاصل شده است.

در حالتی که تعداد کارها برابر با ۳۰ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۷/۲۵ ٪ بهینه شده است.

در حالتی که تعداد کارها برابر با ۵۰ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۲/۲۵ ٪ بهینه شده است.

در حالتی که تعداد کارها برابر با ۷۰ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۳/۲۵ ٪ بهینه شده است.

در حالتی که تعداد کارها برابر با ۹۰ است شایستگی اولویت بر حسب offspring نسبت به شایستگی اولویت بر حسب height در حدود ۳/۷۵ ٪ بهینه شده است.

بررسی در صد بهینگی الگوریتم پیشنهادی، بیانگر این است که هر چه تعداد کارها بیشتر شود، تاحدودی درصد بهینگی بیشتر می‌شود و این ویژگی، مزیتی برای الگوریتم پیشنهادی محسوب می‌شود. البته در حالتی که تعداد کارها کم است، درصد بهینگی بیشتر از سایر حالت‌ها است. ولی هدف اصلی زمانبندی کارها با تعداد کارهای زیاد و تعداد پردازنده‌های کم است، که با تعداد کارهای بیشتر از ۴۰، درصد بهینگی سیر صعودی دارد.

جدول ۱۰ نتایج شبیه‌سازی با درصد وابستگی‌های متفاوت کارها را نشان می‌دهد.

جدول۱۰ : نتایج شبیه سازی سه الگوریتم ژنتیکی با ۵ پردازنده و ۷۰ کار و زمان کارها بین ۱ تا ۱۰۰

درصدوابستگی	۲۰	۴۰	۶۰	۸۰
اتمام زمان روش تصادفی	۲۰۱۷	۲۳۷۲	۲۱۳۹	۲۰۰۲
اتمام زمان روش ارتفاع	۱۹۷۷	۲۱۸۸	۲۰۱۶	۱۹۱۱
اتمام زمان روش پیشنهادی	۱۹۳۹	۲۱۲۹	۱۹۵۵	۱۸۲۶
درصد بهینگی	۱	۲	۳	۴

با توجه به جدول ۱۰ مشاهده می‌شود در حالتی که تعداد کارها، تعداد پردازنده‌ها و محدوده زمان اجرای کارها ثابت فرض شود، هر چه درصد وابستگی کارها افزایش یابد، متناسب با آن درصد بهینگی نیز افزایش می‌یابد و علت اساسی این ویژگی، اولویت‌دهی کارها بر اساس تعداد نوادگان است، زیرا هر چه درصد وابستگی کارها بالاتر در نظر گرفته شود، تعداد نوادگان برای کارها بیشتر می‌شود و زمانبندی بهتر و اصولی‌تری انجام می‌شود.

۶- نتیجه‌گیری

مسأله زمانبندی کارها در سیستم‌های چندپردازنده‌ای از رده مسائل سخت است. ازاین‌رو استفاده از روش‌های ابتکاری به جای روش‌های کلاسیک، جواب نزدیک به بهینه را در یک بازه زمانی قابل قبول به‌دست می‌دهد. از بین روش‌های ابتکاری، الگوریتم ژنتیک به دلیل برخورداری از پتانسیل بالا، در حل مسائل پیچیده مورد توجه قرار گرفته است. در این مقاله، الگوریتم ژنتیک جدیدی جهت زمانبندی کارها در سیستم‌های چندپردازنده‌ای ارایه شده است. این الگوریتم، اولویت اجرای زودتر کارها را براساس تعداد فرزندان و نوادگان (offspring) قرار داده است، به‌عبارتی هر کاری که offspring بالاتری دارد، زودتر زمانبندی می‌شود. الگوریتم پیشنهادی شبیه‌سازی شده و با مقایسه نتایج مشاهده می‌شود که الگوریتم ژنتیک پیشنهادی یافتن یک زمانبندی بهینه را به نحو چشم‌گیری بهبود می‌دهد و زمان اجرای الگوریتم نیز بهبود اندکی می‌یابد.

مراجع

[۱]. *R. Armstrong, D. Hensgen, and T. Kidd, The relative performance of various mapping algorithms is independent of sizable variances in run-time predictions, in ۷th IEEE Heterogeneous Computing Workshop (HCW, ۹۸), "PP. ۷۹-۸۷, ۱۹۹۸.*

[۲]. *Carnegie- Mellon University, Pittsburgh, PA,Genetic Algorithms and Their Applications, Proc. Ist Int Conf, July ۲٤-۲٦, ۱۹۸٥.*

[۳]. *MIT, Cambridge, MA, Genetic Algorithms and Their applications, Proc. ۲nd Int, conf, July ۲۸-۳۱, ۱۹۸۷.*

[٤]. *George Mason Univ.Washington, DS, Genetic Algorithms, Proc. ۳rd Int, Conf, June ٤-۷, ۱۹۸۹.*

[٥]. *D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, MA: Addison- Wesley. ۱۹۸۹.*

[٦]. *D. Fernandez- Baca, Allocating modules to processors in a distributed system, IEEE Trans. Software Engrg. ۱٥, ۱۱ (Nov. ۱۹۸۹), ۱٤۲۷-۱٤۳٦.*

[۷]. *R. F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith, T. Kidd. M Kussow, J. D. Lima, F. Mirabile, L. Moore, B. Rust, and H. J. Siegel, Scheduling resources in multi- user, heterogeneous, computing environments with SmartNet, in : ۷th IEEE Heterogeneous Computing Workshop (HCW'۹۸), "pp. ۱۸٤_۱۹۹, ۱۹۹۸), ۱۳-۱۷.*

[۸]. *R. F. Freund and H. J. Siegel, Heterogeneous processing, IEEE Comput. ۲٦, ٦ (June ۱۹۹۳), ۱۳_۱۷.*

[۹]. *F. Glover and M. Laguna, Tabu Search, Kluwer Academic, Boston, MA, ۱۹۹۷.*

[۱۰]. *D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, Reading, MA, ۱۹۸۹.*

[۱۱]. *R. L. Haupt, S. E. Haupt, Parallel genetic algorithms, John willy & Sons, ۲۰۰٤.*

[۱۲]. *J. H. Holland, "Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor, MI, ۱۹۷٥.*

[۱۳]. *O. H. Ibarra and C. E. Kim, Heuristic algorithms for scheduling independent tasks on nonidentical processors, J. Assoc. Comput. Mach. ۲٤, ۲(Apr . ۱۹۷۷), ۲۸۰-۲۸۹.*

[۱٤]. *M. Iverson, F. Ozguner, and G. Follen, Parallelizing existing application in a distributed heterogeneous*

environment, in "٤th IEEE Heterogeneous Computing Workshop (HCW'٩٥), "pp. ٩٣_١٠٠, ١٩٩٥.

[١٥]. M. Kafil and I. Ahmad, Optimal task assignment in heterogeneous distributed computing systems, IEEE Concurrency ٦, ٣ (July_ Sept. ١٩٩٨), ٤٢_٥١.

[١٦]. S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, Optimization by simulated annealing, Science ٢٢٠, ٤٥٩٨ (May ١٩٨٣), ٦٧١_٦٨٠.

[١٧]. Lee, Y. H., Chen, C. A Modified Genetic Algorithm for Task Scheduling in Multi processor Systems, the ninth workshop on compiler techniques for high performance computing, ٢٠٠٣.

[١٨]. B. Naharai, A. Youssef, and H. A. Choi, Matching and scheduling in a generalized optimal selection theory, in "Proceedings of the Heterogeneous Computing Workshop," pp. ٣- ٨, April ١٩٩٤.

[١٩]. S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice_ Hall, Englewood Cliffs, NJ, ١٩٩٥.

[٢٠]. C.-C. Shen and W.-H. Tsai, A graph matching approach to optimal task assignment in distributed computing system using a minimax criterion, IEEE Trans. Comput. ٣٤, ٣ (Mar. ١٩٨٥), ١٩٧_٢٠٣.

[٢١]. m. h. Shenassa, m. mahmoodi, a new intelligent method for task scheduling in multiprocessor systems using genetic algorithm, Proceeding of the First International Conference on Modeling, Simulation and Applied Optimization, Sharjah, U.A.E. February ١- ٣, ٢٠٠٥.

[٢٢]. m. h. shenassa, m. mahmoodi, a novel intelligent method for task scheduling in multiprocessor systems using genetic algorithm, journal of Franklin institute, Elsevier, ٢٠٠٦.

[٢٣]. P. Shroff, D. Watson, N. Flann, and R. Freund, Genetic simulated annealing for scheduling data- dependent tasks in heterogeneous environments, in "٥th IEEE Heterogeneous Computing Workshop (HCW '٩٦)," pp. ٩٨_١٠٤, ١٩٩٦.

[٢٤]. G.C. Sih and E.A. Lee, A compile- time scheduling heuristic for interconnection- constrained heterogeneous processor architectures, IEEE Trans. Parallel Distrib. Systems ٤ (Feb. ١٩٩٣), ١٧٥_١٨٦.

[٢٥]. H. Singh and A. Youssef, Mapping and scheduling heterogeneous task graphs using genetic algorithms, in "٥th IEEE Heterogeneous Computing Workshop (HCW '٩٦)," pp. ٨٦_٩٧, ١٩٩٦.

[٢٦]. Y. G. Tirat- Gefen and A. C. Parker, MEGA: An approach to system- level design of application specific heterogeneous multiprocessors, in "٥th IEEE Heterogeneous Computing Workshop (HCW '٩٦)," pp. ١٠٥_١١٧, ١٩٩٦.

[٢٧]. L. Wang, H. J. Siegel, V. P. Roychowdhury, and A. A. Maciejewski, Task matching and scheduling in heterogeneous computing environments using a genetic-algorithm- based approach, J. Parallel Distrib. Comput. ٤٧, ١ (Nov. ١٩٩٧), ١_١٥.

[٢٨]. A. Y. Zomaya and R. Kazman, Simulated annealing techniques, in "Algorithms and Theory of Computation Handbook" (M. J. Atallah, Ed.), pp. ٣٧- ١_٣٧- ١٩, CRC Press, Boca Raton, FL, ١٩٩٩.

[٢٩]. Hou E. S. H, Ansari N. and H. Ren, "A Genetic Algorithm for Multiprocessor Scheduling", IEEE trans. on parallel and distributed systems. vol. ٥, no. ٢, pp. ١١٣- ١٢٠, Feb. ١٩٩٤.

This document was created with Win2PDF available at http://www.daneprairie.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.