

ارائه الگوریتم ژنتیک وفقی بهبود یافته برای مسئله زمان بندی کارها بر روی ربات های موازی

گوهر وحدتی^۱، مریم حبیبی پور^۲، سعید طوسی زاده^۳، مهدی یعقوبی^۴

۱- دانشجوی کارشناسی ارشد هوش مصنوعی، دانشگاه آزاد اسلامی، واحد مشهد، elham62.vahdati@gmail.com

۲- دانشجوی کارشناسی ارشد هوش مصنوعی، دانشگاه آزاد اسلامی، واحد مشهد، habibi.maryam@gmail.com

۳- استادیار گروه برق، دانشگاه آزاد اسلامی، واحد مشهد، s.toosi@mshdiau.ac.ir

۴- استادیار گروه کامپیوتر، دانشگاه آزاد اسلامی، واحد مشهد، yaghobi@mshdiau.ac.ir

چکیده

یکی از مسائل مهم زمان بندی، زمان بندی کارها بر روی ربات های موازی است به گونه ای که میانگین تأخیر کارها مینیمم شود. در این مسئله، مجموعه ای از n کار مستقل که در زمان صفر، آماده زمان بندی هستند، وجود دارد. متناظر با هر کار، یک زمان پردازش و یک زمان انقضاء تعیین شده است. هدف این مسئله پیدا کردن ترتیبی از کارها بر روی ربات ها می باشد به نحوی که میانگین تأخیر کارها مینیمم شود. این مسئله، یکی از مسائل ترکیبی NP-Hard است. الگوریتم ژنتیک یک ابزار مناسب برای حل مسائل بهینه سازی ترکیبی می باشد. در این مقاله، یک الگوریتم ژنتیک وفقی غیرخطی به همراه دو عملگر ادغام و جهش جدید اکتشافی، بکار گرفته شده است. در این الگوریتم، تابع برازندگی برحسب میانگین تأخیر کارها در نظر گرفته شده است که با تنظیم احتمال جهش و ادغام، به صورت وفقی و غیرخطی با میانگین تأخیر، از مشکلاتی از قبیل همگرایی نابهنگام، همگرایی کند و همگرایی با پایداری کم جلوگیری می شود. نتایج آزمایشات نشان می دهند که الگوریتم ژنتیک پیشنهادی به آسانی در بهینه های محلی قرار نمی گیرد و از سرعت خوبی در همگرایی به جواب جامع برخوردار می باشد. علاوه بر این، پیاده سازی الگوریتم پیشنهادی، از پیچیدگی برخوردار نمی باشد.

کلمات کلیدی

زمان بندی ربات های موازی، الگوریتم ژنتیک، میانگین تأخیر

۱- مقدمه

توجه قرار گرفته است. جمعیت الگوریتم ژنتیک متشکل از یک گروه افراد می باشد که هر یک از آنها یک راه حل را ارائه می کنند. افراد می توانند به واسطه عملیات انتخاب، ادغام و جهش، مقدار برازندگی بالاتری بدست آورند تا بهترین راه حل مسئله را جستجو کنند [۴]. در این مقاله، الگوریتم ژنتیک وفقی به همراه دو عملگر ادغام و جهش اکتشافی بکار گرفته شده است. همچنین، کارایی الگوریتم

تاکنون روش های بسیاری برای حل مسئله زمان بندی کارها ارائه شده است که می توان روش های الگوریتم ژنتیک [۵ - ۱]، [۶ و ۷] SA^۱، جستجوی تابو [۸] و شاخه و حد [۹] را نام برد. از میان این روش ها، الگوریتم های ژنتیک به علت کارایی حاصل از این روش در پیدا کردن راه حل های نزدیک به جواب بهینه و زمان محاسباتی کم برای مسائل بهینه سازی ترکیبی بزرگ، بیشتر مورد

```

begin GA
  create initial population
  while generation_count < k do
    /* k = max.numberof generation.*/
    begin
      Selection and Elitism
      Crossover
      Mutation
      Increment generation_count
    end
    Output the best individual found
  end GA

```

شکل ۱- شبه کد الگوریتم ژنتیک

۳-۱- کروموزوم

برای حل مسئله توسط الگوریتم ژنتیک، یک کروموزوم که دارای اطلاعات ژن می باشد در نظر گرفته می شود. الگوریتم های ژنتیک کلاسیک از رشته های باینری برای ایجاد کروموزوم استفاده می کنند که برای مسائلی از این قبیل مناسب نمی باشند زیرا یک راه حل مستقیم و موثر برای نگاشت جواب های ممکن به صورت یک رشته باینری وجود ندارد. هر کروموزوم در این مسئله به صورت $c_1, c_2, c_3, \dots, c_j, \dots, c_n$ می باشد که c_j یک عدد صحیح مثبت است. توالی ژن ها در کروموزوم، ترتیب اجرای کارها را مشخص می کند [۱]. به عنوان مثال، کروموزوم برای مسئله ای با ۷ کار در شکل (۲) نمایش داده شده است.

۵	۱	۲	۴	۶	۳	۷
---	---	---	---	---	---	---

شکل ۲- نمایش کروموزوم

۳-۲- عملگر ادغام

پس از اینکه مجموعه ای از بهترین رشته ها انتخاب شد عملگر ادغام با هدف تولید رشته های بهتر بر روی استخر تولید مثل اعمال می شود. هدف از ادغام، جستجوی فضای پارامتر و تا حد امکان حفظ اطلاعات نهفته در رشته ها است.

۳-۲-۱- عملگر ادغام دو نقطه ای نسخه اول $(2PX-V1)^2$

در این عملگر، از والد اول دو نقطه ادغام به تصادف انتخاب می کنیم. ژن هایی را که بین دو نقطه ادغام نیستند را مستقیماً به

پیشنهادی و الگوریتم ژنتیک استاندارد (به همراه عملگرهای ادغام و جهش ژنتیکی رایج)، بر روی سه مسئله استاندارد [۱۰] مورد مقایسه قرار گرفته اند.

نتایج نشان می دهند که الگوریتم پیشنهادی از سرعت خوبی در همگرایی به جواب جامع برخوردار است همچنین پیاده سازی الگوریتم پیشنهادی از پیچیدگی برخوردار نمی باشد.

۲- توصیف مسئله

در این مسئله، n تعداد کار، بر روی m تعداد ربات موازی به گونه ای زمان بندی می شوند که میانگین تأخیر کارها مینیمم شوند. مجموعه ای از n کار مستقل که در زمان صفر، آماده زمان بندی هستند، وجود دارد. متناظر با هر کار، یک زمان پردازش p_i ، و یک زمان انقضاء d_i تعیین شده است [۱].

- یک ربات در هر زمان، فقط یک کار را می تواند انجام دهد.
- پردازش از نوع انحصاری (غیرقابل پس گرفتنی) است.
- T_j ، تأخیر کار j ام می باشد.
- C_j ، زمان اتمام و d_j ، زمان انقضاء کار j ام می باشد.
- $R(i, j)$ ، پردازش یا عدم پردازش کار j بر روی ربات i را نشان می دهد. اگر کار j بر روی ربات i پردازش شود $R(i, j) = 1$ ، در غیر این صورت $R(i, j) = 0$.

$$c_i = \sum_{j=1}^i p_j \quad (۱)$$

$$T_j = \max\{0, C_j - d_j\} \quad (۲)$$

$$MeanTardiness = \frac{\sum_{i=1}^m \sum_{j=1}^n R(i, j) T_j}{n} \quad (۳)$$

۳- الگوریتم ژنتیک

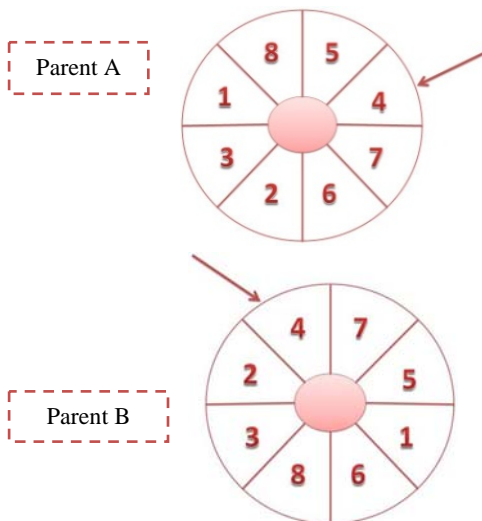
الگوریتم ژنتیک، کارآیی خوبی برای جستجوی سراسری از خود نشان داده است که با جمعیت و چندین نقطه در فضای جستجو شروع به جستجو می کند. الگوریتم ژنتیک دارای عملگرهای انتخاب، ادغام و جهش می باشد که تنظیم پارامترهایی از قبیل احتمال ادغام و احتمال جهش برای الگوریتم ژنتیک نیاز می باشد. در این مسئله، برازندگی الگوریتم، میانگین تأخیر کارها در نظر گرفته شده است به طوری که هر چه میانگین تأخیر کمتر باشد مقدار برازندگی بیشتر خواهد بود. همچنین تابع هدف، به حداقل رساندن میانگین تأخیر کارها در نظر گرفته شده است.

۳-۲-۶- عملگر ادغام پیشنهادی

در این عملگر، کروموزوم‌های پدر به صورت صف چرخشی در نظر گرفته می‌شوند.

فرآیند تولید کروموزوم‌های فرزند به شرح زیر می‌باشد:

۱. یک کار به صورت تصادفی انتخاب می‌شود و دو اشاره‌گر در کروموزوم‌های پدر A و B به کار مورد نظر اشاره می‌کنند.
۲. کار مورد نظر در کروموزوم فرزند اول درج می‌شود سپس به جای این کار در کروموزوم‌های پدر صفر قرار می‌گیرد و اشاره‌گرها یک واحد در جهت عقربه‌های ساعت جلو می‌روند.
۳. زمان پردازش کارهایی که اشاره‌گرها به آنها اشاره می‌کنند، PA و PB نامیده می‌شوند. اگر PA کوچکتر یا مساوی PB باشد آنگاه کاری که اشاره‌گر در کروموزوم پدر A به آن اشاره می‌کند در فرزند اول درج می‌شود و در کروموزوم‌های پدر A و B به جای کار مورد نظر صفر قرار می‌گیرد و اشاره‌گر مربوط به کروموزوم A، به کار غیر صفر بعدی اشاره می‌کند. ولی اگر PA بزرگتر از PB باشد آنگاه کاری که اشاره‌گر در کروموزوم B به آن اشاره می‌کند در فرزند اول درج می‌شود و در کروموزوم‌های پدر به جای کار مورد نظر صفر قرار می‌گیرد و اشاره‌گر مربوط به کروموزوم B، به کار غیر صفر بعدی اشاره می‌کند. شرط توقف هنگامی خواهد بود که تعداد کارهای غیر صفر کروموزوم A برابر صفر شود. برای تولید کروموزوم فرزند دوم، مراحل بالا انجام می‌شود با این تفاوت که اشاره‌گرها در خلاف جهت عقربه‌های ساعت حرکت می‌کنند. در شکل (۳)، عملگر ادغام بر روی کروموزوم‌های پدر، در مسئله‌ای با ۸ کار (که کار ۴ به عنوان نقطه شروع می‌باشد)، اعمال شده است و کروموزوم‌های فرزند تولید شده‌اند.



شکل ۳ الف) - نمایش کروموزوم‌های پدر و فرزند در عملگر ادغام

فرزند اول منتقل می‌کنیم و این ژن‌ها را از والد دوم حذف می‌کنیم. ژن‌های بین دو نقطه ادغام را با ژن‌های باقیمانده از والد دوم پر می‌کنیم [۳].

۳-۲-۳- عملگر ادغام مبتنی بر ترتیب^۳ (OBX)

در این عملگر، تعدادی از ژن‌های والد اول انتخاب می‌شوند. موقعیت ژن‌های انتخاب شده از والد دوم حذف می‌شوند و این ژن‌ها به موقعیت‌های حذف شده متناظر در فرزند اول منتقل می‌شوند و بقیه ژن‌ها مستقیماً از والد دوم از چپ به راست به ارث می‌رسند [۳].

۳-۲-۳- عملگر ادغام چرخشی^۴ (CX)

در این عملگر، هدف این است که بین دو والد یک چرخه پیدا شود. برای این کار اولین ژن از والد اول، به اولین ژن در والد دوم مرتبط می‌شود. محتوای ژن اول والد دوم را در والد اول پیدا کرده و آن را به ژن متناظر در والد دوم مرتبط می‌کنیم و آنقدر این کار را ادامه می‌دهیم تا در والد دوم محتوایی برابر با ژن اول والد اول پیدا کنیم. در این لحظه چرخه ساخته شده است. ژن‌های منتخب از والد اول به فرزند اول منتقل می‌شوند و بقیه ژن‌ها از والد دوم به ارث می‌رسند [۳].

۳-۲-۴- عملگر ادغام مرتبه‌ای^۵ (OX)

این عملگر توسط Davis ارائه شده است. ابتدا دو نقطه ادغام از والد دوم انتخاب می‌شود. ژن‌های بین این دو نقطه به فرزند اول منتقل می‌شوند و نشانگر فرزند اول روی نقطه دوم باقی می‌ماند این ژن‌ها از والد اول حذف می‌شوند. از نقطه دوم ادغام روی والد اول به سمت راست شروع به حرکت می‌کنیم در صورت وجود ژن جدید آن را در محل نشانگر فرزند اول قرار می‌دهیم و نشانگر را جلو می‌بریم. اگر به آخرین ژن فرزند اول یا والد اول رسیدیم به اول آن بر می‌گردیم، تا ژن‌های فرزند اول کامل شوند [۳].

۳-۲-۵- عملگر ادغام با نگاشت جزئی^۶ (PMX)

در این عملگر، دو نقطه ادغام به صورت تصادفی از والد دوم انتخاب می‌شوند و ژن‌های بین این دو نقطه به فرزند اول به ارث می‌رسد. موقعیت‌های متناظر در والد اول نیز علامت‌گذاری می‌شوند. به عبارت دیگر از ژن‌های والد دوم به والد اول نگاشتی ایجاد می‌شود. این نگاشت در جدولی ذخیره می‌شود. ژن‌های بدون علامت از والد اول در جدول نگاشت جستجو می‌شوند تا ژن متناظرشان را در جدول بیابند، پاسخ این جستجو در ژن‌های خالی فرزند اول از چپ به راست به ارث می‌رسد و اگر ژنی در جدول نگاشت وجود نداشت خودش منتقل می‌شود [۳].

۴-۳- تنظیم وفقی غیر خطی احتمال جهش و ادغام

معمولاً احتمال‌های جهش و ادغام در الگوریتم ژنتیک استاندارد ثابت در نظر گرفته می‌شوند بنابراین حتی اگر الگوریتم بارها و بارها تکرار شود، راه‌حل‌های بهتر مختلفی ظاهر نخواهد شد. بنابراین الگوریتم ژنتیک وفقی که می‌تواند احتمال ادغام و جهش را به صورت وفقی و غیر خطی تنظیم کند بکار گرفته شده است.

تابع هدف، به حداقل رساندن میانگین تأخیر کارها می‌باشد. در الگوریتم ژنتیک وفقی منحنی‌های تنظیم احتمال جهش و ادغام به آهستگی با T_{avg} تغییر می‌کنند و احتمال جهش و ادغام افرادی که برازندگی آنها تا حد زیادی به میانگین برازندگی جمعیت نزدیک است افزایش می‌یابد و رنج افزایش یافته در الگوریتم ژنتیک وفقی غیر خطی، بیشتر از الگوریتم ژنتیک وفقی خطی و سایر الگوریتم‌ها می‌باشد [۴].

$$P_c = \begin{cases} p_{c1} - \frac{p_{c1} - p_{c2}}{1 + \exp(a(\frac{T_{avg} - T'}{T_{avg} - T_{min}}))} & T' \leq T_{avg} \\ p_{c1} & T' > T_{avg} \end{cases} \quad (4)$$

$$P_m = \begin{cases} p_{m1} - \frac{p_{m1} - p_{m2}}{1 + \exp(a(\frac{T_{avg} - T}{T_{avg} - T_{min}}))} & T \leq T_{avg} \\ p_{m1} & T > T_{avg} \end{cases} \quad (5)$$

- T_{min} : کمترین تأخیر جمعیت
- T_{avg} : میانگین تأخیر جمعیت
- T' : کمترین تأخیر بین دو فرد در عمل ادغام
- T : تأخیر فرد در عمل جهش

۴- نتایج تجربی

```

begin GA
create initialpopulation
while generation_count < k do
/* k = max.numberof generation*/
begin
Selection and Elitism
Heuristic Crossover ← Nonlinear adjusting Crossover Probabiliy
Heuristic Mutation ← Nonlinear adjusting Crossover Probabiliy
Increment generation_count
end
Output thebest individual found
end GA

```

شکل ۵- شبه کد الگوریتم ژنتیک پیشنهادی

کار	۱	۲	۳	۴	۵	۶	۷	۸
زمان پردازش	۲	۴	۵	۳	۸	۷	۱۰	۱۲
زمان انقضاء	۱	۳	۲	۸	۷	۴	۱۲	۱۴
پدر	۵	۴	۷	۶	۲	۳	۱	۸
پدر	۷	۵	۱	۶	۸	۳	۲	۴
فرزند	۴	۷	۶	۲	۳	۱	۵	۸
فرزند	۴	۲	۳	۵	۸	۱	۶	۷

شکل ۳ (ب) - وضعیت کروموزوم‌های پدر و فرزند در عملگر ادغام

۳-۳- عملگر جهش

این عمل برای جلوگیری از همگرایی ناپهنگام و کمک به الگوریتم جستجو برای فرار از به دام افتادن در بهینه‌های محلی مفید است. از سوی دیگر این عمل برای حفظ حالت تمایز کروموزوم‌ها در یک جمعیت بکار می‌رود.

۳-۳-۱- عملگر جهش معکوس (IM)

فرآیند تولید کروموزوم فرزند به شرح زیر می‌باشد:
 گام ۱) به صورت تصادفی، یک مقدار بین ۳ تا ۱- L ، به عنوان طول بلاک جهش در نظر گرفته می‌شود که L ، طول کروموزوم می‌باشد.
 گام ۲) به صورت تصادفی یک موقعیت در کروموزوم پدر در نظر گرفته می‌شود. موقعیت دوم از جمع موقعیت اول با طول بلاک جهش بدست می‌آید. بلاک جهش، از ژن‌های مابین و مشمول اولین و دومین موقعیت تشکیل شده است.
 گام ۳) به منظور تولید بلاکی با ژن‌های جدید، بلاک جهش، معکوس خواهد شد و جایگزین بلاک جهش در گام ۲ خواهد شد. کروموزوم جدید، کروموزوم فرزند می‌باشد [۴].

۳-۳-۲- عملگر جهش پیشنهادی

این عملگر به این صورت عمل می‌کند که یک کار به صورت تصادفی انتخاب می‌شود سپس از میان بقیه کارها، کاری که کمترین زمان پردازش را دارد انتخاب می‌شود سپس ترتیب کارهای مشمول و مابین این دو کار عکس می‌شود. به عنوان مثال در مسئله‌ای با ۸ کار و انتخاب کار ۱ به عنوان نقطه شروع داریم:

کار	۱	۲	۳	۴	۵	۶	۷	۸
زمان پردازش	۲	۴	۵	۳	۸	۷	۱۰	۱۲
زمان انقضاء	۱	۳	۲	۸	۷	۴	۱۲	۱۴

پدر	۵	۴	۷	۶	۲	۳	۱	۸
فرزند	۵	۱	۳	۲	۶	۷	۴	۸

شکل ۴- نمایش کروموزوم پدر و فرزند در عملگر جهش

همگرا شده است که در مقایسه با سایر روشها بهبود قابل توجهی داشته است.

همچنین الگوریتم پیشنهادی کمترین میانگین تأخیر را نسبت به سایر روشها داشته است. نتایج نشان می دهند که الگوریتم ژنتیک پیشنهادی با تعداد نسل های کم به جواب های بسیار نزدیکی به جواب بهینه، دست می یابد.

جدول ۱- پارامترهای الگوریتم ژنتیک پیشنهادی

سایز جمعیت	تعداد نسل	a	Pm2	Pm1	Pc2	Pc1
۲۰	۱۰۰	۲۰	۰,۰۵	۰,۳	۰,۶	۰,۹۹

جدول ۲- پارامترهای الگوریتم ژنتیک استاندارد

سایز جمعیت	تعداد نسل	نرخ جهش	نرخ ادغام
۲۰	۱۰۰	۰,۱	۰,۹

مسائل استاندارد بکار گرفته شده به صورت تصادفی تولید شده اند. فرآیند تولید این مسائل به شرح زیر می باشد [۱۰]:
 برای هر کار $j (j=1,2,3,\dots,n)$ ، زمان پردازش صحیح $\{p_j\}$ از توزیع یکنواخت $[1,100]$ به دست می آید.
 دامنه نسبی زمان انقضاء (RDD) و میانگین ضریب تأخیر (TF) از مجموعه $[0.2,0.4,0.6,0.8,1]$ انتخاب می شوند.
 برای هر کار j ، زمان انقضاء صحیح $p(j)$ ، از توزیع یکنواخت (۶) بدست می آید.

$$[P(1-TF-RDD/2), P(1-TF+RDD/2)] \quad (۶)$$

$P = \sum_{j=1,2,\dots,n} p(j)$
 جدول (۳)، نتایج حاصل از مقایسه الگوریتم ژنتیک وفقی بهبود یافته $(IAGA)^8$ و الگوریتم ژنتیک استاندارد (به همراه عملگرهای ادغام و جهش ژنتیکی رایج)، بر روی سه مسئله استاندارد را در ۱۰ بار اجرا نشان می دهد. اعداد داخل جدول (۳)، نشان دهنده میانگین تأخیر کارها می باشند. n بیانگر تعداد کارها و m بیانگر تعداد رباتها است.

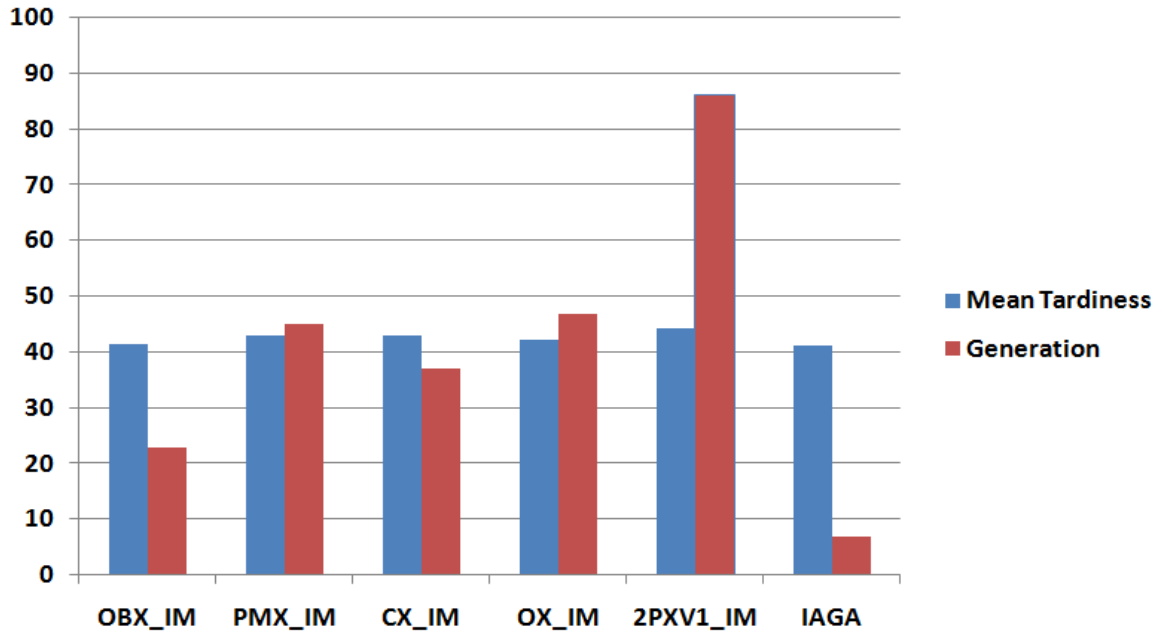
شکل (۶)، تعداد نسل های مورد نیاز برای رسیدن به جواب بهینه و میانگین تأخیر کارها در الگوریتم های مختلف برای مسئله زمان بندی ۴۰ کار و ۴ ربات را نشان می دهد. همان طور که در شکل مشاهده می شود الگوریتم پیشنهادی در نسل هفتم به جواب بهینه

جدول ۳- میانگین تأخیر کارها برای مسائل استاندارد مختلف در الگوریتم پیشنهادی و الگوریتم ژنتیک استاندارد پس از ۱۰ بار اجرا

Problem	Obx_IM	Pmx_IM	Cx_IM	Ox_IM	2pxv1_IM	IAGA
m=2 n=12	23.41	23.58	23.66	23.5	23.5	23.41
n=40 m=4	41.5	42.92	43.02	42.2	44.37	41.35
n=50 m=3	37.58	38.86	39.98	37.54	38.5	37.3

جدول ۴- میانگین تعداد تکرار نسلها برای رسیدن به جواب بهینه در الگوریتم پیشنهادی و الگوریتم ژنتیک استاندارد پس از ۱۰ بار اجرا

Problem	Obx_IM	Pmx_I M	Cx_IM	Ox_IM	2pxv1_IM	IAGA
m=2 n=12	24	52	85	62	50	6
n=40 m=4	23	45	37	47	86	7
n=50 m=3	23	55	43	92	80	8



شکل ۶- مقایسه تعداد تکرار نسل‌ها و میانگین تأخیر کارها در الگوریتم‌های مختلف برای مسئله ۴*۴۰

one Machine Total Weighted Tardiness Problem; Elsevier, Applied Mathematics and Computation, Vol. 199, No. 2, pp. 590 - 598, 2008.

- [4] X. Yingjie, C. Zhentong, Jing Sun, **An Improved Adaptive Genetic Algorithm for Job-Shop Scheduling Problem**, Proceedings of the Third International Conference on Natural Computation, Vol. 04, pp. 287-291, 2007.
- [5] D.H.J. Biskup, J.N.D. Gupta, **Scheduling Identical Parallel Machines to Minimize Total Tardiness**, Elsevier, International Journal of Production Economics, Vol. 115, No. 1, pp. 134 - 142, 2008.
- [6] C. Koulamas, **Decomposition and Hybrids Simulated Annealing Heuristics for the Parallel Machine Total Tardiness Problem**, Naval Research Logistics, Vol. 44, No. 1, pp. 109 - 125, 1997.
- [7] K.H Kim, D.W. Kim, **Unrelated Parallel Machine Scheduling with Setup Times Using Simulated Annealing**, Robotics and Computer-Integrated Manufacturing, Vol. 18, No. 3, pp. 223 -231, 2002.
- [8] V.A. Armentano, D.S. Yamashita, **Tabu Search for Scheduling on Identical Parallel Machines to Minimize Mean Tardiness**, Journal of Intelligent Manufacturing, Vol. 11, No. 5, pp. 453 - 460, 2000.
- [9] M. Azizog˘lu, O. Kirca, **Tardiness Minimization on Parallel Machines**, International Journal of Production Economics, Vol. 55, No. 2, pp. 163 - 168, 1998.
- [10] OR-Library
<http://people.brunel.ac.uk/~mastjbjb/jeb/info.html>

۵- نتیجه گیری

الگوریتم‌های ژنتیک به علت کارایی در جستجوی راه‌حل‌های نزدیک به جواب بهینه و زمان محاسباتی کم، برای مسائل بهینه‌سازی ترکیبی، مورد استفاده قرار می‌گیرند. در این مقاله، یک الگوریتم ژنتیک وفقی بکار گرفته شده است که با تنظیم احتمال ادغام و جهش به صورت وفقی غیرخطی، از مشکلاتی مثل همگرایی نابهنگام، همگرایی با سرعت کم و پایداری کم جلوگیری می‌کند. همچنین با در نظر گرفتن دو عملگر جدید ادغام و جهش اکتشافی، می‌تواند ویژگی‌های خیلی خوب را از کروموزوم‌های پدر به ارث ببرد. نتایج تجربی نشان می‌دهند که الگوریتم ژنتیک پیشنهادی به‌آسانی در بهینه‌های محلی قرار نمی‌گیرد و در تعداد نسل‌های کمتری به جواب بهینه جامع دست پیدا می‌کند. همچنین شایان ذکر است که پیاده‌سازی الگوریتم پیشنهادی از پیچیدگی برخوردار نمی‌باشد.

۶- مراجع

- [1] Cakar, T., Koker, R., Demir, H.I.; **Parallel Robot Scheduling to Minimize Mean Tardiness with Precedence Constraints Using a Genetic Algorithm**, Elsevier, Advances in Engineering Software, Vol. 39, No. 1, pp. 47 - 54, 2008.
- [2] S.Z Jun, Z.J. Ying; **A Genetic Algorithm Based Approach to Intelligent Optimization for Scheduling Dual Resources with Robots**, Robot, Vol. 24, No. 4, pp. 342 - 357, 2002.
- [3] T. Kellegoz, B. Toklu, J. Wilson; **Comparing Efficiencies of Genetic Crossover Operators for**

۷- پی نوشتها

- 1- Simulated Annealing (SA)
- 2- Two Point Crossover Operator Version 1(2px_v1)
- 3- Order Based Crossover Operator (OBX)
- 4- Cycle Crossover Operator (CX)
- 5- Order operator (OX)
- 6- Partially Mapped Crossover Operator (PMX)
- 7- Inverse Mutation (IM)
- 8- Improved Adaptive Genetic Algorithm (IAGA)