

# الگوریتم ترکیبی یادگیری افزایشی مبتنی بر جمعیت و بهینه سازی حدی (PBIL-EO , PBIL-GEO)

میترا هاشمی<sup>1</sup>، محمد رضا میبیدی<sup>2</sup>

<sup>1</sup> دانشگاه آزاد اسلامی قزوین، دانشکده برق رایانه و فن آوری اطلاعات، قزوین

Mitra.hash@yahoo.com

<sup>2</sup> آزمایشگاه سیستم‌های نرم دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیر کبیر، تهران

Mmeybodi@aut.ac.ir

## چکیده

الگوریتم PBIL یکی از انواع الگوریتم‌های تخمین توزیع است. این الگوریتم در مقایسه با الگوریتم‌هایی مانند الگوریتم ژنتیک از نظر سرعت، حافظه مصرفی و دقت جواب‌های بدست آمده، کارایی بهتری دارد؛ و به خوبی می‌تواند نقاط ناشناخته فضای جستجو را با به کار بردن روش‌های آماری جستجو نماید. این الگوریتم از یک بردار احتمال استفاده کرده و اعضای جمعیت را از طریق نمونه برداری از آن ایجاد می‌کند. از طرفی الگوریتم EO برای جستجوهای محلی و نزدیک بهینه مناسب می‌باشد و می‌تواند بهترین جواب را از میان جواب‌های محلی بدست آورده و در بهینه محلی گیر نکند. لذا با ترکیب این دو الگوریتم می‌توان تعاملی بین دو مفهوم اساسی مطرح در الگوریتم‌های تکاملی، اکتشاف و بهره برداری، ایجاد نمود و نتایج بهتری بدست آورد. نتایج موجود در این مقاله نشان دهنده کارایی الگوریتم پیشنهادی بر روی دو مسئله بگرنج، دو تکه کردن گراف و زمان‌بندی وظایف بر روی چند پردازنده می‌باشد.

## کلمات کلیدی

الگوریتم یادگیری افزایشی مبتنی بر جمعیت، بهینه سازی حدی، بهینه سازی حدی تعمیم یافته، الگوریتم‌های تخمین توزیع، مسائل بهینه سازی.

## 1- مقدمه

در این الگوریتم‌ها یک جمعیت ممکن است با یک توزیع احتمال تخمین زده شود و راه‌حل‌های کاندید جدید توسط نمونه‌برداری از این توزیع بدست آیند. این روش مزایای زیادی دارد مانند، اجتناب از همگرایی زودرس و استفاده از یک نمایش فشرده و کوتاه. الگوریتم یادگیری افزایشی مبتنی بر جمعیت<sup>[1]</sup> یک الگوریتم بهینه‌سازی و تخمین توزیع است. الگوریتم توسط بالوجا در سال 1994 بدست آمد. این الگوریتم ساده‌تر از یک الگوریتم ژنتیک استاندارد است و در اکثر مواقع به سمت نتایج بهتر از آن هدایت می‌کند.

در الگوریتم یادگیری افزایشی، ژن‌ها به صورت مقادیر حقیقی در رنج  $[0,1]$  نمایش داده می‌شود که مشخص کننده احتمال هر الی خاص در آن ژن است. در این الگوریتم به سوی بهترین فرد عمل یادگیری با تغییر بردار احتمال انجام می‌گیرد و نهایتاً جوابهای مناسبی ارائه می‌دهد.

در طول دهه 1990 اکثر مسائل بهینه‌سازی ترکیبی با موفقیت توسط الگوریتم‌های ژنتیک حل می‌شدند. اما وجود مسائل فریبنده و سخت و بدلیل عملکرد ضعیف الگوریتم ژنتیک باعث جست‌وجوی الگوریتم‌های بهینه‌سازی جدید شد. به خاطر وجود این مشکلات یک تعداد از محققان گروهی از الگوریتم‌های جدید که الگوریتم‌های تخمین توزیع<sup>[1,2,3]</sup> نامیده می‌شود را پیشنهاد دادند.

این الگوریتم‌ها توسط ماهلن بین و پاب مطرح شدند، الگوریتم‌های تخمین توزیع یک نمونه الهام گرفته از مکاشفه تصادفی مبتنی بر جمعیت‌هایی از افراد هستند که هر کدام یک راه‌حل ممکن از مسأله بهینه‌سازی را کد می‌کنند. این جمعیت‌ها در نسل‌های بعدی با پیشرفت جستجو تکامل می‌یابند.

تفاوت اساسی آن با الگوریتم ژنتیک حفظ آمارهای جمعیت به جای حفظ یک تعداد زیاد از نمونه‌ها می‌باشد. چون الگوریتم یادگیری افزایشی مجبور نیست یک جمعیت بزرگ را نگه‌داری کند، بنابراین حافظه کمتری را نیاز دارد و اجراهایش سریعتر و نتایج در اکثر موارد بهتر از یک الگوریتم ژنتیک معادل است.

ایرادی که در مورد این الگوریتم وجود دارد، آمارهای مرتبه دوم (وابستگی متقاطع) جمعیت را حفظ نمی‌کند. بنابراین این الگوریتم بطور موثر در فضا‌هایی که آمارهای مرتبه دوم وجود دارد و یک امر مهم در بهینه‌سازی رشته بیتی است دچار مشکل می‌شود، بنابراین با ترکیب این روش با روشی مانند بهینه‌سازی حدی می‌توان تا حدی این مشکل را حل کرده و به جوابهای بهینه و با دقت بیشتر دست یافت.

الگوریتم  $\tau$ -EO [8,9] یک الگوریتم تکاملی جدید است که بر اساس ویژگی بحران‌های خود سازمانده [5,6] عمل می‌کند. این الگوریتم که برای حل مسائل بهینه‌سازی استفاده شده و جواب‌های مطلوبی ارائه داده است، بر خلاف بسیاری از الگوریتم‌های تکاملی که در هر لحظه از چندین جواب تشکیل شده‌اند تنها از یک جواب تشکیل شده و در هر مرحله آن را بهبود می‌دهد. بهبود جواب در این الگوریتم با انتخاب یکی از اجزای جواب (جزء تعویضی) و جایگزین کردن مقدار آن با مقداری جدید انجام می‌شود.

در این مقاله با ترکیب این دو روش به گونه‌ای که بهترین فرد بدست آمده با الگوریتم یادگیری افزایشی مبتنی بر جمعیت، توسط الگوریتم بهینه‌سازی حدی دوباره بهبود می‌یابد و مانع از غیر افتادن الگوریتم در بهینه‌های محلی و دست‌یابی به بهینه سراسری می‌شود، عمل شده است. این الگوریتم بر روی دو مسئله؛ دو تکه کردن گراف و زمانبندی کارها روی چند پردازنده انجام گرفته، نتایج نشان دهنده عملکرد مناسب الگوریتم پیشنهادی در مقایسه با الگوریتم‌های مطرح همانند: الگوریتم ژنتیک، بهینه‌سازی حدی، بهینه‌سازی حدی و اتوماتای یادگیر و الگوریتم یادگیری افزایشی مبتنی بر جمعیت استاندارد است.

ادامه مقاله بدین صورت سازماندهی شده است: در بخش 2 و 3 الگوریتم PBIL و  $\tau$ -EO به اختصار معرفی می‌شوند. بخش 4 به معرفی الگوریتم جدید PBIL-GEO و PBIL-EO اختصاص دارد. در بخش 5 مسائل آزمایشی و روش حل آنها توسط الگوریتم‌های مقایسه شده شرح داده می‌شوند. در بخش 6 نتایج حاصل از اجرای الگوریتم‌ها بر روی مسائل انتخاب شده ارائه شده و با هم مقایسه می‌شوند. بخش 7 جمع‌بندی می‌باشد.

## 2- الگوریتم یادگیری افزایشی مبتنی بر جمعیت (PBIL)

الگوریتم یادگیری افزایشی [1,2] یک ترکیبی از بهینه‌سازی تکاملی و یادگیری رقابتی است. این الگوریتم کمک می‌کند تا یک

بردار احتمال با مقادیر حقیقی  $\vec{P} = \{P_1, \dots, P_L\}$  (طول کدگذاری باینری) ایجاد شود، بطوریکه وقتی نمونه برداری می‌شود، راه‌حل‌های با کیفیت بالا و با احتمال بالا را تولید کند.

از یک بردار احتمال که مقدار 0.5 برای هر مکان بیت در نظر می‌گیرد، شروع می‌شود. این بردار احتمال، بردار احتمال مرکزی به خاطر اینکه در نقطه مرکز فضای جستجو می‌افتد، نامیده می‌شود. نمونه‌برداری از این بردار احتمال اولیه راه‌حل‌های تصادفی تولید می‌کند. یک راه‌حل از یک بردار احتمال  $\vec{P}$  به این صورت نمونه‌برداری می‌شود، برای هر محل ژن  $i$ ، اگر یک عدد تصادفی ایجاد شده  $p_i = \text{rand}(0,1) < r$  (1) می‌شود، در غیر اینصورت آن محل ژن (0) قرار داده می‌شود.

در تکرار  $t$ ، یک مجموعه  $S(t)$  متشکل از  $n$  راه‌حل از بردار احتمال  $\vec{p}(t)$  نمونه‌برداری می‌شود. نمونه‌ها با استفاده از تابع شایستگی مبتنی بر مسأله ارزیابی می‌شوند. سپس بردار احتمال برای یافتن بهترین راه‌حل  $\vec{B}(t)$  از مجموعه  $S(t)$  یادگیری انجام می‌دهد، به صورت زیر:

(1)

$$P_i(t+1) = (1 - \alpha) * P_i(t) + \alpha * B_i(t),$$

بطوریکه  $\alpha$  نرخ یادگیری است که فاصله بردار احتمال برای هر تکرار را معین می‌کند. بعد از اینکه بردار احتمال برای رسیدن به بهترین نمونه بروز رسانی می‌شود، به منظور حفظ تنوع نمونه‌ها ممکن است نیاز به انجام عمل جهش بیتی باشد.

عمل جهش به این صورت انجام می‌شود. برای محل ژن  $i = \{1, \dots, L\}$ ، اگر یک عدد تصادفی  $r = \text{rand}(0,1) < P_m$  (احتمال جهش) تولید شود، پس  $P_i$  با رابطه زیر تغییر داده می‌شود:

(2)

$$P'_i = \begin{cases} P_i * (1.0 - \delta_m) & P_i > 0.5 \\ P_i & P_i = 0.5 \\ P_i * (1.0 - \delta_m) + \delta_m & P_i < 0.5 \end{cases}$$

بطوریکه  $\delta_m$ ، شیف‌ت جهش است که تعداد عملیات جهش را کنترل می‌کند. بعد از عمل جهش، یک مجموعه جدید از نمونه‌ها توسط بردار احتمال جدید تولید می‌شوند و این حلقه تکرار می‌شود.

با پیشروی فرآیند جستجو، عناصر در بردار احتمال از مقدار اولیه 0.5 به سمت 0 یا 1 می‌روند و راه‌حل‌های با کیفیت بالا را نشان می‌دهند. روند جستجو وقتی که تعدادی از شرایط پایانی ارضاء می‌شود متوقف می‌گردد. برای مثال، رسیدن به حداکثر تعداد دفعات تکرار  $t_{\max}$  یا وقتی که بردار احتمال به مقادیر 0 یا 1 در هر محل از بیتیها همگرا می‌شود.

### 3- الگوریتم بهینه سازی حدی (τ-EO)

الگوریتم τ-EO [9و8] یک الگوریتم اکتشافی مبتنی بر پدیده بحران‌های خود سازمانده [5,6] است. در این الگوریتم، جواب مساله به آرایه‌ای تبدیل می‌شود که هر یک از عناصر آن معادل یکی از اجزای جواب است.

این الگوریتم در هر مرحله (هر تکرار حلقه اصلی) با انتخاب یکی از اجزای جواب (جزء تعویضی) و تغییر مقدار آن جواب را بهبود می‌دهد. در این الگوریتم، برای پیدا کردن جزء تعویضی، ابتدا اجزای جواب به صورت صعودی بر اساس شایستگی محلی آنها مرتب می‌شوند. سپس به هر جزء با توجه به مکان آن در فهرست مرتب شده ( $n$ ) احتمالی متناسب با  $P(n)$  (رابطه 1) نسبت داده شده و در انتها یکی از اجزا با توجه به این احتمال‌ها انتخاب می‌شود؛ انتخاب جزء تعویضی در این روش به نحوی است که اجزا با شایستگی کمتر با احتمال بالاتری انتخاب می‌شوند. این روش موجب می‌شود تا الگوریتم τ-EO بتواند از بسیاری از نقاط بهینه محلی فرار کند.

(3)

$$P(n) \sim n^{-\tau}$$

الگوریتم بهینه‌سازی حدی نیازمند تعریف تابعی برای محاسبه شایستگی محلی سلول‌های جواب است، از طرف دیگر چگونگی تعویض سلول تعویضی به شکل مساله وابسته است. بنابراین نمی‌توان به راحتی و بدون نیاز به پیاده‌سازی مجدد، از آن برای حل دامنه وسیعی از مسائل استفاده کرد. به این منظور الگوریتم بهینه‌سازی حدی تعمیم یافته (GEO) [10] بر اساس الگوریتم بهینه‌سازی حدی با پارامتر  $\tau$  معرفی شده است.

برای استفاده از الگوریتم بهینه‌سازی حدی تعمیم یافته، ابتدا متغیرهای مساله باید به یک رشته دودویی نگاشت شوند. مساله دیگری که در الگوریتم مفروض مطرح است، محاسبه شایستگی سلول‌ها است. محاسبه شایستگی محلی مهمترین مرحله در الگوریتم بهینه‌سازی حدی است. در هر مرحله بر اساس جواب جاری و به تعداد سلول‌های آن، جواب‌هایی تولید می‌شود که به عنوان فرزندان جواب جاری شناخته می‌شوند. این جواب‌ها از هم متمایز بوده و هر کدام تنها در مقدار یک سلول با جواب جاری اختلاف دارند. بعد از تولید شدن جواب‌های جدید، شایستگی هر کدام از آنها بوسیله تابع شایستگی کلی محاسبه می‌شود؛ شایستگی هر سلول برابر خواهد بود با شایستگی فرزندی از جواب جاری، که در مقدار آن سلول با جواب جاری اختلاف دارد، منهای شایستگی جواب جاری.

### 4- الگوریتم پیشنهادی (PBIL-EO , PBIL-GEO)

با توجه به مزایا و معایب ذکر شده از الگوریتم یادگیری افزایشی مبتنی بر جمعیت و برای رفع معایب آن الگوریتم مفروض را با

الگوریتم بهینه‌سازی حدی ترکیب می‌کنیم. در الگوریتم بهینه‌سازی حدی نیز قدرت جستجوی کل فضای جستجو در مقایسه با الگوریتم‌های دیگر کمتر می‌باشد لذا با ترکیب این دو الگوریتم از قدرت مکاشفه کل فضای جستجوی الگوریتم یادگیری افزایشی مبتنی بر جمعیت و نیز قدرت الگوریتم بهینه‌سازی حدی در یافتن جوابهای دقیق در قسمتی از فضای جستجو عمل می‌کنیم.

در ابتدا بردار احتمال را با مقادیر اولیه 0.5 مقدار دهی می‌نماییم و سپس یک تعداد افراد بر اساس اینکه سائز جمعیت چقدر باشد با استفاده از بردار احتمال نمونه برداری می‌شوند.

در الگوریتم پیشنهادی از نخبه‌گزینی نیز استفاده و بهترین فرد به عنوان نخبه انتخاب شده و بعد از تغییر، در یادگیری بردار احتمال مطابق رابطه 1 شرکت می‌نماید. با انتخاب بهترین فرد از قسمتی از فضای جستجو سعی داریم با اعمال بهینه‌سازی حدی بر روی بهترین فرد و اعمال یک جستجوی محلی در فضای بهترینها، شایسته‌ترین فرد را بدست آورده و از آن در عمل یادگیری استفاده نماییم. می‌توان مراحل الگوریتم پیشنهادی را به صورت زیر نمایش داد:

- 1- مقداردهی اولیه
- 2- مقداردهی بردار احتمال با مقدار 0.5 (بردار احتمال مرکزی)
- 3- نمونه برداری  $N$  فرد از بردار احتمال
- 4- پیدا کردن و حفظ (نخبه‌گزینی) بهترین فرد در جمعیت
- 5- اعمال الگوریتم EO داخلی یا GEO داخلی (با توجه به مسئله) بر روی بهترین فرد
- 6- عمل یادگیری بردار احتمال (رابطه 1) بر اساس بهترین فرد حاصل از مرحله قبل
- 7- اعمال جهش بر روی بردار احتمال (رابطه 2)
- 8- تکرار از مرحله 3 تا توقف الگوریتم
- 9- بر گرداندن بهترین جواب

الگوریتم EO داخلی:

- 1- محاسبه شایستگی اجزای جواب با توجه به مسئله
- 2- مرتب کردن اجزای جواب بر اساس شایستگی بصورت صعودی
- 3- انتخاب یکی از اجزای جواب با استفاده از توزیع احتمال رابطه 3
- 4- انتخاب مقدار جدید برای جزء تعویضی با توجه به مسئله
- 5- جایگزین کردن مقدار جدید جزء تعویضی و تولید جواب جدید
- 6- اجرا از مرحله 1 تا زمانی که بهبودی حاصل می‌شود.

الگوریتم GEO داخلی:

- 1- تولید فرزندان جواب جاری و محاسبه شایستگی آنها
- 2- مرتب کردن جوابهای تولید شده با توجه به شایستگی به صورت صعودی
- 3- انتخاب یکی از فرزندان به عنوان جواب جاری با توجه به رابطه 3
- 4- تکرار مراحل تا وقتی که بهبودی ایجاد می شود.

نتایج بر روی هر دو مسئله آزمایش نشان دهنده کارایی روش مفروض در مقایسه با روش های دیگر است.

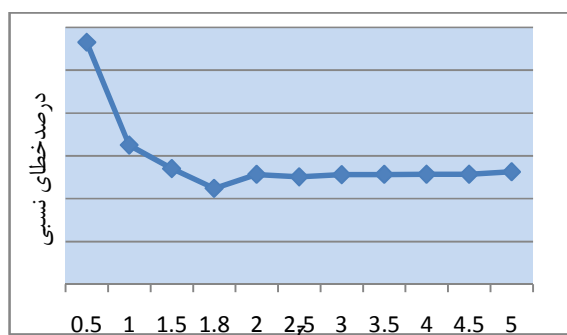
## 5- مسائل آزمایشی و روش حل آنها

برای ارزیابی کارایی الگوریتم پیشنهادی و مقایسه آن با روش های دیگر از دو مساله بگرنج زمانبندی کارها روی چند پردازنده<sup>۲</sup> و دو تکه کردن گراف<sup>۴</sup> استفاده شده است. هدف در مساله زمانبندی، توزیع وظایف بین پردازنده ها است بطوریکه اولویت وظایف حفظ و زمان پاسخ (اجرای کامل) مینیمم مقدار شود. نمونه هایی از مساله که برای مقایسه عملکرد الگوریتم ها استفاده شده از مرجع [11] آورده شده اند. در مساله دو تکه کردن گراف نیز هدف، تقسیم کردن گراف به دو قسمت است به شکلی که تعداد گره های دو قسمت برابر باشد (در صورت فرد بودن تعداد گره های گراف، اختلاف تعداد گره های دو قسمت برابر 1 باشد) و ثانیاً تعداد یال های بین دو قسمت کمترین مقدار ممکن باشد. نمونه های استفاده شده برای مقایسه عملکرد الگوریتم ها در حل این مساله از مرجع [7] آورده شده اند.

### 5-1- حل مسئله دو تکه کردن گراف

برای حل مساله دو تکه کردن گراف توسط الگوریتم PBIL-EO از رشته ای از بیتها با مقادیر 0 و 1 به منظور تعیین دو بخش متفاوت در گراف استفاده می شود و نیز از روش حل این مساله توسط الگوریتم  $\tau$ -EO در [8] و [9] استفاده شده است در این مراجع از یک خوشه بندی اولیه نیز استفاده شده است؛ در این روش برای محاسبه شایستگی هر جزء جواب، از نسبت تعداد گره های همسایه گره در دو تکه گراف استفاده می شود. برای اینکه جوابها با مفروضات مسئله مطابقت داشته باشند (تعداد گره های مساوی در هر بخش) از الگوریتم KL [12] نیز استفاده شده است. در پیاده سازی انجام شده برای حل مساله دو تکه کردن گراف، با توجه به اهمیت نرخ یادگیری موجود در رابطه 1 الگوریتم مورد نظر با مقادیر مختلف نرخ یادگیری بر روی نمونه مسائل انجام شده و مقداری که خطای نسبی کمتری در مقایسه با مقادیر دیگر داشت (0.25) انتخاب گردید. در الگوریتم مفروض احتمال جهش (0.02) و شیفت جهش (0.2) نیز به روش گفته شده و محاسبه خطای نسبی بدست آمده است.

همچنین برای مقایسه الگوریتم های EO-PBIL، EO-LA و  $\tau$ -EO و مشاهده تاثیر تغییرات اعمال شده، مقدار پارامتر  $\tau$  در تمامی اجراها برای 1.8 که بهترین مقدار این پارامتر برای الگوریتم  $\tau$ -EO می باشد قرار داده شده است. شکل 1 نشان دهنده مجموع خطای نسبی به ازای مقادیر مختلف  $\tau$  می باشد. همانطور که مشاهده می شود الگوریتم به ازای مقدار 1.8 برای پارامتر  $\tau$  کمترین خطا را تولید کرده است.



شکل 1: تعیین مقدار مناسب پارامتر  $\tau$

### 5-2- حل مسئله زمانبندی کارها روی چند پردازنده

برای پیاده سازی الگوریتم PBIL-GEO برای حل مساله زمانبندی کارها روی چند پردازنده از روش حل این مساله توسط الگوریتم GEO در [10] استفاده شده است. در حل این مسئله از نمونه مسائل مطرح در این زمینه [11] استفاده شده است. برای بررسی نتایج زمانبندی با در نظر گرفتن اولویت و بدون در نظر گرفتن آن انجام شده است. دو نمونه مسئله مورد آزمایش با 50 و 100 کار و در نظر گرفتن سیستم موازی با 2، 4، 8 و 16 پردازنده فرض شده است. نحوه نمایش افراد جمعیت و توضیح کامل در مرجع [10] ذکر شده است.

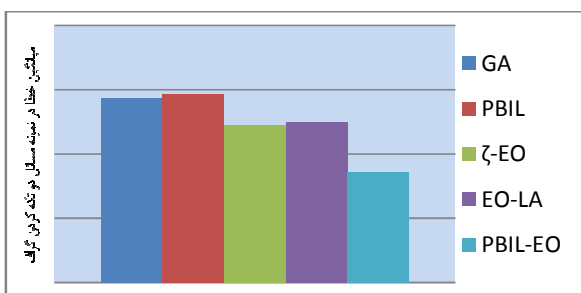
در این مسئله مقادیر نرخ یادگیری (0.25)، احتمال جهش (0.02) و شیفت جهش (0.05) با محاسبه خطای نسبی و روش ذکر شده بدست آمده است. همچنین برای مقایسه الگوریتم های PBIL-GEO، GEO و مشاهده تاثیر تغییرات اعمال شده، مقدار پارامتر  $\tau$  در تمامی اجراها برای 1.2 که بهترین مقدار این پارامتر برای الگوریتم GEO می باشد قرار داده شده است. هم چنین سائز جمعیت 60 و تعداد دفعات تکرار برابر 100 در نظر گرفته شده است.

## 6- نتایج

برای مقایسه الگوریتم ها در حل مساله زمانبندی کارها روی چند پردازنده هر کدام از این الگوریتم ها به تعداد 10 مرتبه بر روی نمونه مسائل آزمایشی اجرا و کمینه نتایج تولید شده توسط آنها جمع آوری شده است. در این مقایسه ها، مقدار پارامتر  $\tau$  برای الگوریتم ها برابر 1.2 در نظر گرفته شده اند؛ علت برابر قرار دادن پارامتر  $\tau$  برای این دو

اتوماتای یادگیر) نیز عملکرد بهتری داشته و این بدلیل تاثیر اتوماتا در بهبود عملکرد جستجوی محلی EO بوده و هنوز امکان اینکه نقاط مکاشفه نشده وجود داشته باشد است. شکل 3 نیز نشان می‌دهد، میانگین خطا در نمونه مسائل دو تکه کردن گراف در الگوریتم پیشنهادی کمتر از بقیه الگوریتم‌ها است.

نتایج مناسب این الگوریتم بدلیل استفاده از مزایای هر دو الگوریتم و رفع عیوب آنها از طریق ترکیب با یکدیگر است. الگوریتم PBIL عمل جستجوی نقاط ناشناخته در فضا را انجام می‌دهد و به نوعی می‌توان گفت بر روی عمل اکتشاف<sup>۵</sup> تاکید دارد، الگوریتم EO روی استفاده از تجربیات قبلی<sup>۶</sup> و جستجو در نزدیکی نقاط بهینه و یافتن نقطه بهینه سراسری تاکید دارد.



شکل 3: نمودار میله ای مقایسه روش‌ها با در نظر داشتن میانگین خطا نمونه مسائل دو تکه کردن گراف

## 7- جمع‌بندی

با توجه به نتایج حاصل ملاحظه می‌شود الگوریتم پیشنهادی (PBIL-EO و PBIL-GEO) بر روی هر دو مسئله جهان واقعی، دو تکه کردن گراف و زمانبندی وظایف بر روی چند پردازنده، عملکرد خوبی داشته است و این بدلیل ترکیب دو روش و استفاده از مزایای هر دو روش و ایجاد تعادل بین دو مفهوم مکاشفه و بهره برداری از اطلاعات قبلی است. الگوریتم PBIL عمل کشف نقاط ناشناخته در فضای جستجو و EO جستجو در نزدیکی نواحی بهینه را برای یافتن بهینه سراسری انجام می‌دهند. می‌توان این الگوریتم را مثلاً با اتوماتای یادگیر ترکیب نمود (در مقایسه نتایج EO و EO-LA) و به نتایج بهتری دست یافت.

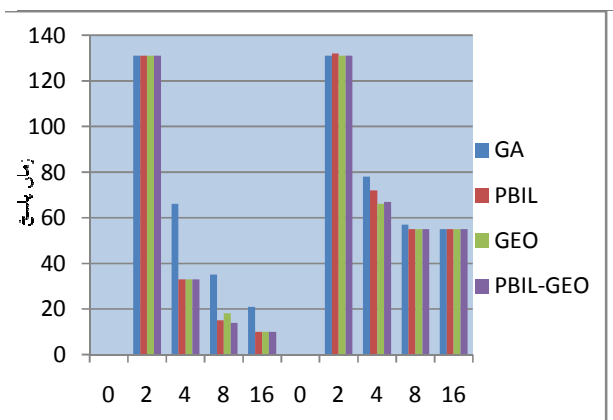
## مراجع

- [1] Yang S. and Yao X., "Population-Based Incremental Learning with Associative Memory for Dynamic Environments", IEEE Trans. Evolutionary Computation, vol. 12, No. 5, pp. 542-561, 2008.
- [2] Tianshi Chen, Tang Ke, Guoliang Chen, Xin Yao, "Analysis of Computational Time of Simple Estimation of Distribution Algorithms", IEEE Trans., Evolutionary computation, vol. 14, No. 1, 2010

الگوریتم بررسی تاثیر تغییرات ایجاد شده در الگوریتم GEO و PBIL-GEO است.

نتایج اجرای الگوریتم‌های مختلف برای این مسئله در جداول 2 و 3 آمده است. اجرای الگوریتم‌ها در دو حالت با اولویت و بدون اولویت برای انجام کارها در نظر گرفته شده است.

در مسئله زمانبندی با توجه به نتایج ملاحظه می‌شود که تقریباً در تمام موارد الگوریتم مفروض در مقایسه با بقیه الگوریتم‌ها (PBIL, GEO, GA)، عملکرد بهتری داشته و توانسته به بهترین جواب ممکن یعنی کوتاهترین زمان پاسخ دست یابد. در تعداد پردازنده کم اکثر الگوریتم‌ها به جواب مطلوب رسیده‌اند، کارایی الگوریتم مفروض در توزیع وظایف بر روی تعداد پردازنده زیاد مشخص‌تر می‌شود. از شکل 2 نیز مشخص می‌شود که در هر دو وضعیت، با نظر گرفتن اولویت برای وظایف (مجموعه نمودارهای سمت راست) و بدون در نظر گرفتن اولویت (مجموعه نمودارهای سمت چپ) تقریباً در تمام موارد روش پیشنهادی کمترین مقدار (زمان پاسخ) ممکن را دارد.



شکل 2: نمودار میله ای مقایسه روش‌ها در زمانبندی با تعداد پردازنده مختلف و زمان پاسخ در دو حالت با و بدون اولویت

برای بررسی و مقایسه الگوریتم‌ها در حل مساله دو تکه کردن گراف نیز از نمونه مسائل انتخاب شده در بخش قبل استفاده شده است. برای مشاهده تغییر عملکرد الگوریتم‌های EO-LA و PBIL-EO و مقدار پارامتر  $\tau$  برای آنها برابر 1.8 در نظر گرفته شده است. مقایسه الگوریتم‌ها بر اساس مجموعه جواب‌های تولید شده توسط آنها برای نمونه مسائل انتخاب شده در 10 اجرا انجام شده است. نتایج آماری جواب‌های تولید توسط این الگوریتم‌ها در جدول 1 نمایش داده شده است. همانطور که در این جدول مشاهده می‌شود الگوریتم PBIL-EO تقریباً در تمامی موارد از بقیه الگوریتم‌ها بهتر می‌باشد.

با توجه به جدول نتایج همانطور که ذکر شد ملاحظه می‌شود، الگوریتم PBIL نسبت به GA کارایی بهتری دارد. در مقایسه PBIL و EO تقریباً در نصف موارد PBIL و نصف دیگر EO عملکرد بهتری داشته است. در مقایسه با EO-LA (ترکیب بهینه سازی حدی با

تعداد پردازنده	GA	PBIL	GEO	PBIL- GEO
		بدون اولویت		
2	291	291	291	291
4	146	71	72	71
8	77	24	24	23
16	23	23	23	22
		با اولویت		
2	291	291	291	291
4	189	171	146	146
8	153	106	98	94
16	150	99	97	94

جدول 3: نتایج آزمایشات زمانبندی وظایف با 50 وظیفه

تعداد پردازنده	GA	PBIL	GEO	PBIL- GEO
		بدون اولویت		
2	131	131	131	131
4	66	33	33	33
8	35	15	18	14
16	21	10	10	10
		با اولویت		
2	131	132	131	131
4	78	72	66	67
8	57	55	55	55
16	55	55	55	55

[3] Hons Robin, Estimation of Distribution Algorithms and Minimum Relative Entropy, phd. Thesis, university of Bonn, 2005.

[4] Boettcher, S. and Percus, A. G., "Extremal Optimization: An Evolutionary Local-Search Algorithm", <http://arxiv.org/abs/cs.NE/0209030>

[5] [http://en.wikipedia.org/wiki/Self-organized\\_criticality](http://en.wikipedia.org/wiki/Self-organized_criticality)

[6] Per Bak, Chao Tang and Wiesenfeld, K. "Self-organized Criticality", Physical Review A, Vol. 38, No. 1, 1988.

[7] <http://staffweb.cms.gre.ac.uk/~c.walshaw/partition>

[8] Boettcher, S., "Extremal Optimization of Graph Partitioning at the Percolation Threshold", Physics A, vol 32, No.28, pp.5201-5211, 1999.

[9] Boettcher, S. and Percus, A. G., "Extremal Optimization for Graph Partitioning", Physical Review E, vol. 64, pp. 021114, 2001.

[10] Switalski Piotr, Seredynski Franciszek, "Solving multiprocessor scheduling problem with GEO metaheuristic", IEEE International Symposium on Parallel&Distributed Processing, 2009.

[11] <http://www.kasahara.elec.waseda.ac.jp>

[12] Mühlenbein H. and Mahnig Th., "Evolutionary Optimization and the Estimation of Search Distributions with Applications to Graph Bipartitioning", Journal of Approximate Reasoning, Vol. 31, 2002.

جدول 2: نتایج آزمایشات زمانبندی وظایف با 100 وظیفه

Estimation of Distribution Algorithm<sup>1</sup>  
Population Based Incremental Learning<sup>2</sup>  
Multiprocessor Scheduling Problem<sup>3</sup>  
Graph Bipartitioning Problem<sup>4</sup>  
exploration<sup>5</sup>  
exploitation<sup>6</sup>

جدول 1: نتایج آزمایشات مسئله دو تکه کردن گراف

نام گراف	GA	PBIL	ζ-EO	EO-LA	PBIL- Eo
3elt	335.5	332	330.5	327	267
4elt	772	734.5	728	722.5	666.5
Add20	4287	4774	4942	4683	3168
Add32	2179.5	2425	254.5	216	583.5
Crack	445	442	380	378.5	382
CS4	1016	1018	769	790	643.5
CTI	2433.5	2294	1747	1794	1442
data	510	504	672	674	483
FE-4elt2	196	200	494	502	196
FE-sphere	665	664	677	678	654
UK	67	66	35	31	37
Whitaker3	224.5	217.5	442	453	205
Wing-nodal	9863	9636	9287	9263.5	6710