فصلنامه علمی – پژوهشی مطالعات مدیریت صنعتی – سال سیزدهم، شماره ۳۶، بهار ۹۴
صفحات ۱۷۱-۱۴۳

# زمانبندی دو معیاره برای حداقل سازی زمان دیرکرد کل و واریانس زمان انتظار بر روی یک ماشین با استفاده از الگوریتم ژنتیک

مقصود امیری،* مهدی کشاورز قرابایی،**

(تاریخ دریافت:۹۱/۸/۳ – تاریخ پذیرش: ۹۲/۲/۴)

## چکیده

*مسائل عملی زمانبندی معمولاً تصمیم‌گیرنده را وادار به در نظر گرفتن تعداد زیادی از معیارها قبل از اتخاذ تصمیم می‌نمایند. این تحقیق یک مسئله زمانبندی تک ماشین را مورد بررسی قرار می‌دهد که هدف در آن حداقل کردن ترکیبی از دو معیار دیرکرد کل و واریانس زمان انتظار می‌باشد به طوری که زمان بیکاری در ماشین مجاز نیست. حداقل کردن دیرکرد کل همیشه به عنوان یک معیار عملکرد مهم در سیستم‌های عملی، که می‌توان با استفاده از آن از تحمیل هزینه‌های جریمه دیرکرد اجتناب نمود، مطرح می‌باشد و واریانس زمان انتظار نیز یک معیار مهم در پیاده‌سازی کیفیت خدمات (QoS) در بسیاری از سیستم‌ها می‌باشد. هر کدام از این دو معیار از نوع NP-hard می‌باشند و بنابراین ترکیب خطی آن‌ها نیز NP-hard خواهد بود. برای این مسئله الگوریتمی ژنتیک طراحی شده که از ساختار معمول آن استفاده می‌کند. دو نوع جمعیت هیوریستیک و تصادفی برای جمعیت اولیه و دو نوع تابع برازش در الگوریتم به کار رفته است. کارایی الگوریتم ژنتیک ارائه شده به وسیله تست روی تعداد زیادی از مسائل نشان داده می‌شود.*

**واژگان کلیدی**:
زمانبندی دو معیاره؛ تک ماشین؛ الگوریتم فرا ابتکاری؛ دیرکردکل؛ واریانس زمان انتظار

* استاد دانشگاه علامه طباطبایی
** کارشناسی ارشد مدیریت صنعتی دانشگاه علامه طباطبایی(نویسنده مسئول) m.keshavarz_gh@yahoo.com

**مقدمه**

یکی از وظایف بسیار مهم برنامه ریزی تولید، تعیین توالی و زمانبندی قطعات بر روی ماشین‌ها می‌باشد. ساده‌ترین مسئله زمانبندی، مسئله‌ای است که در آن تنها یک منبع یا یک ماشین وجود دارد. محیط تک ماشین بسیار ساده، دارای اهمیت زیاد و یک حالت خاص برای تمام محیط‌های ماشینی دیگر می‌باشد. در عمل مسائل زمانبندی در محیط‌های پیچیده تر اغلب اوقات به زیر مسئله‌هایی که مرتبط با سیستم تک ماشین است تجزیه می‌شوند (Baker & Trietsch, 2009; Pinedo, 2008).

از زمان اولین مقاله زمانبندی که در سال ۱۹۵۴ نوشته شد ، متغیرهای بسیاری از مسئله پایه زمانبندی، با تمایز قائل شدن بین محیط‌های مختلف ماشین، محدودیت‌های جانبی و تابع‌های مختلف هدف، تعریف شد. تا سال ۱۹۸۰ هنوز معمول بود که تابع هدف فقط یک معیار عملکرد را مورد توجه قرار دهد اما برای رسیدن به یک تعادل قابل قبول باید کیفیت راه حل روی تمام معیارهای مهم سنجیده شود و این نکته منجر به گسترش زمانبندی چند معیاره شد (Gupta et al., 1990).

دیرکرد کل در مواقعی که تحویل کارها بعد از موعد مقرر موجب تحمیل هزینه‌هایی به سیستم شود معیاری بسیار مهم می‌باشد. وقتی که این معیار را حداقل می‌کنیم در عین حال که به رضایت مشتری فکر می‌کنیم، از هزینه‌هایی که در اثر تحویل دیرهنگام سفارشات به سیستم تحمیل می‌شود نیز پرهیز خواهیم کرد (Holsenback & Russell, 1992). واریانس زمان انتظار به عنوان معیاری دیگر در این تحقیق، در سیستم‌هایی که لزوم پاسخ‌دهی یکسان به درخواست‌ها احساس می‌شود، نقشی کلیدی دارد (Merten & Muller, 1972). این معیار در سیستم‌ها منجر به عملکرد قابل پیش بینی سیستم خواهد شد و همچنین در فراهم کردن کیفیت خدمات (QoS) در بسیاری از صنایع مهم است (Ye et al., 2007). حداقل کردن این معیار باعث می‌شود کارهایی که وارد سیستم می‌شوند، در یک محدوده زمانی مشخص در انتظار ورود به سیستم برای پردازش بمانند و این منجر به پاسخگویی مناسب به

کارهایی است که همه در داخل سیستم تولیدی از اولویت یکسانی در پردازش برخوردارند. با تحقیقات انجام گرفته روی این معیارها مشخص شده که هرکدام از آن‌ها به طور مجزا به صورت مسئله‌ای از نوع NP-Hard هستند (Du & Leung, 1990; Wieslaw, 1993) و می‌توان گفت اگر مسئله‌ای به صورت NP-hard باشد، الگوریتم بهینه برای مسئله به زمان محاسباتی احتیاج دارد که با افزایش اندازه مسئله به صورت نمایی افزایش می‌یابد (Ferrolho & Crisostomo, 2006; Madureira et al., 2001; Potts & Van Wassenhove, 1991). برای حل این نوع از مسائل معمولاً از رویکردهای فرا ابتکاری استفاده می‌شود (Aryanezhad, et al., 2009).

محققین بسیاری روی معیارهای استفاده شده در این تحقیق، الگوریتم‌های ژنتیک و همچنین زمانبندی چند معیاره تحقیق نموده‌اند. در این میان مرتن[1] و مولر[2] (۲۰۰۱) حداقل‌سازی واریانس را در مسائل زمانبندی تک ماشینه مورد تحقیق قرار دادند. در این تحقیق، آن‌ها واریانس زمان در جریان و واریانس زمان انتظار را به عنوان مقیاس عملکرد در مسائل زمانبندی تک ماشین آنالیز کردند و نشان دادند توالی که واریانس زمان در جریان را حداقل می‌کند ، متضاد توالی است که واریانس زمان انتظار را حداقل می‌کند. ایلون[3] و چادوری[4] (۱۹۷۷) مسئله تک ماشین را در حالتی که واریانس زمان‌های انتظار حداقل شوند بررسی کردند. آن‌ها این تئوری را مطرح کردند که توالی بهینه باید به صورت V-shaped باشد و پنج روش ابتکاری برای حداقل کردن واریانس زمان‌های انتظار ارائه کردند. کنت[5] (۱۹۸۱) مسئله واریانس زمان در جریان ساخت را در سیستم تک ماشینه بررسی کرد. او یک روش ابتکاری برای حل مسئله ارائه داد که نسبت به روش‌های ارائه شده قبلی از کارایی خوبی برخوردار بود. گوپتا[6] و دیگران (۱۹۹۰) مسئله واریانس زمان در جریان ساخت را در سیستم

[1] Merten
[2] Muller
[3] Eilon
[4] Chowdhury
[5] Kanet
[6] Gupta

تک ماشین بررسی کردند. آن‌ها یک روش ابتکاری ارائه دادند که پیچیدگی آن در سطح ( n log n) است و این روش ابتکاری را از نظر کارایی با روش مرتن و مولر (۲۰۰۱) و روش‌های ابتکاری ایلون و چادوری (۱۹۷۷) و همچنین روش کنت (۱۹۸۱) مقایسه کردند. سریرانگاچاریولو[1] و سرینیواسان[2] (۲۰۱۰) مسئله حداقل کردن واریانس زمان تکمیل را بررسی کردند. آن‌ها یک روش ابتکاری جدید پیشنهاد دادند. همچنین یک روش بر مبنای الگوریتم ژنتیک برای حل مسئله فراهم نموده و در بدترین حالت ممکن عملکرد روش پیشنهادی را بررسی کردند و نشان دادند که روش ابتکاری از نظر عملکرد بسیار بهتر از روش‌های متداول است و همچنین برای حالت چند ماشینه کاربرد دارد. یی[3] و دیگران (۲۰۰۷) روش‌های کاهش واریانس زمان انتظار را بررسی کردند. آن‌ها دو روش ابتکاری BS ( Balanced Spiral) و VS (Verified Spiral) را ارائه کردند و روش‌های خود را با چهار روش زمانبندی دیگر روی مسائل با اندازه‌های کوچک و بزرگ مقایسه کردند. بگچی[4] و دیگران (۱۹۸۷) مسئله زمانبندی را برای حداقل کردن میانگین مربع انحراف‌ها حول یک موعد تحویل مشترک بررسی کردند. آن‌ها نشان دادند که حالت خاصی از مسئله MSD ( Mean Squared Deviation) که مسئله MSD نامقید گفته می‌شود، معادل مسئله CTV (Completion Time Variance) است، که به وسیله مرتن و مولر (۲۰۰۱) بررسی شده است. آن‌ها در تحقیق خود به بهبود روش‌های حل مسائل WTV ( Waiting Time Variance) وCTV و ارائه یک رویه جدید پرداختند. هلسنبک[5] و راسل[6](۱۹۹۲) مسئله حداقل کردن دیرکرد کل را مورد تحقیق قرار دادند و برای اجتناب از محاسبه تمام توالی‌های ممکن یک روش ابتکاری ارائه دادند. این روش ابتکاری از تحلیل NBR ( Net Benefit of Relocation) برای تعیین اینکه کدام کار باید در آخر بیاید تا دیرکرد کل حداقل شود،

[1] Srirangacharyulu
[2] Srinivasan
[3] Ye
[4] Bagchi
[5] Holsenback
[6] Russell

استفاده کرده است. آن‌ها در عمل نشان دادند که روش ابتکاری بر پایه تحلیل NBR بسیار کارامد تر از روش ابتکاری بر پایه API (Adjacent Pairwise Interchange) ارائه شده توسط فرای[۱] و دیگران (۱۹۸۹) است. همچنین روش ابتکاریشان را با روش پاتس[۲] (۱۹۸۲) مقایسه کردند و نشان دادند که از نظر عملکرد بهتر از آن است. کولاماس[۳] (۱۹۹۴) پژوهشی کلی روی مسئله زمان دیرکرد کل انجام داد. او بررسی خود را روی روش‌های ابتکاری متمرکز کرد و در نهایت در تحقیق خود یک روش ابتکاری برای دو حالت تک ماشینه و ماشین موازی ارائه داد. بگچی (۱۹۸۹) مسئله زمانبندی تک ماشینه را در دو مسئله دو هدفه بررسی کرد، مسئله اول حداقل کردن همزمان میانگین و واریانس زمان تکمیل و مسئله دوم حداقل کردن همزمان میانگین و واریانس زمان انتظار بود. او در نهایت یک الگوریتم کارا برای حداقل کردن این توابع هدف ارائه داد. هن[۴] (۲۰۰۵) روی مهم‌ترین نتایج زمانبندی چند هدفه تمرکز کرد. او در تحقیق خود مفاهیم اولیه زمانبندی دو هدفه را بررسی کرد و همچنین زمانبندی با زودکرد و دیرکرد و زمانبندی با زمان‌های پردازش کنترل شده را مورد بحث قرار داد. سو[۵] و تین[۶] (۲۰۱۱) مسئله حداقل سازی میانگین مربع انحرافات زمان‌های تکمیل حول یک موعد تحویل مشترک را با محدودیت حداکثر دیرکرد مورد بررسی قرار دادند و حالتی را در نظر گرفتند که موعد تحویل به اندازه‌ای بزرگ است که حداقل سازی MSD بدون محدودیت انجام شود. آن‌ها در تحقیقشان راه حلی برای این مسئله ارائه کردند و با مثال‌های زیادی آن را مورد ارزیابی قرار دادند. کوکسالان[۷] و بوراک[۸] (۲۰۰۳) تحقیقی روی استفاده از الگوریتم ژنتیک برای حل مسائل دو هدفه انجام و دو مسئله، حداقل کردن زمان در جریان ساخت و تعداد کارهای دارای دیرکرد و همچنین، حداقل کردن زمان در

[1] Fry
[2] Potts
[3] Koulamas
[4] Han
[5] Su
[6] Tien
[7] Köksalan
[8] Burak

جریان ساخت و حداکثر زودکرد را بررسی و الگوریتمی ژنتیک برای حل این مسائل ارائه کردند. جولای[1] و دیگران (۲۰۰۷) یک الگوریتم ژنتیک برای حداقل کردن همزمان حداکثر زودکرد و تعداد کارهای دارای دیرکرد ارائه دادند. آن‌ها مثال‌هایی محاسباتی از الگوریتم پیشنهادی خود ارائه دادند و عملکرد آن را ارزیابی کردند.

این تحقیق به دنبال ارائه الگوریتمی برای حل مسئله زمانبندی در محیط تک ماشین می‌باشد که دو معیار دیرکرد کل و واریانس زمان انتظار را به صورت همزمان حداقل کند. به این منظور الگوریتمی فرا ابتکاری بر مبنای ساختار معمول الگوریتم‌های ژنتیک توسعه داده شده است که کارایی آن روی تعداد زیادی از مسائل نشان داده می‌شود.

## روش تحقیق

این تحقیق مسئله زمانبندی $n$ کار روی یک ماشین را مورد بررسی قرار می‌دهد به نحوی که دیر کرد کل و واریانس زمان انتظار در آن حداقل می‌شود. مفروضات مسئله عبارتند از:

- تمام کارها در زمان صفر در دسترس هستند.
- کارها از یکدیگر مستقل می‌باشند.
- هر کار فقط یک بار روی ماشین پردازش می‌شود.
- حق شفعه[2] مجاز نمی‌باشد.
- زمان‌های پردازش و موعدهای تحویل هر کار در زمان صفر مشخص می‌باشند.

مسئله مورد نظر تحقیق را در دو دسته متمایز از مسائل با توجه به تعداد کارها مورد بررسی قرار می‌دهیم. دسته اول مربوط به مسائلی می‌شود که تعداد کارها در آن‌ها $n$ =5, 10, 15 خواهد بود و دسته دوم مربوط به مسائلی می‌شود که تعداد کارها در آن‌ها $n$ =20, 50, 100 خواهد بود. چهار الگوریتم ژنتیک با ویژگی‌های متفاوت ارائه می‌شود که برای بررسی

[1] Jolai
[2] Preemption

کارایی این الگوریتم‌ها تابع هدفی در تحقیق مورد استفاده قرار میگیرد که به صورت ترکیبی خطی وزنی از دو هدف مسئله و به صورت LP-metric می‌باشد. همان طور که قبلاً گفته شد هرکدام از اهداف مسئله NP-hard می‌باشند، بنابراین می‌توان گفت که ترکیب خطی آن‌ها نیز مسئله ای NP-hard است. در مسائل مورد بررسی توابع هدف با وزن‌های مختلف (w=0.3, 0.5, 0.7) با یکدیگر ترکیب می‌شوند. با استفاده از وزن‌های مختلف می‌توانیم نقاط بیشتری در مرز پارتو را بررسی کنیم. زمان‌های پردازش در دو حالت بالا و پایین، و موعدهای تحویل در چهار حالت مختلف که در بخش‌های بعد به آن‌ها اشاره می‌شود، ارزیابی می‌شوند. در دسته اول انحراف نتایج حاصل از الگوریتم‌های ژنتیک طراحی شده از نتایج بهینه که به وسیله نرم افزار LINGO حاصل شده، بدست می‌آید. در دسته دوم انحراف نتایج الگوریتم‌های ژنتیک طراحی شده از بهترین نتایج که از همان الگوریتم‌ها حاصل شده، بدست می‌آید (برای هر مسئله یکی از چهار الگوریتم، که بهترین نتیجه را ارائه کرده مبنا قرار گرفته و انحراف بقیه نتایج از آن بدست می‌آید). لازم به ذکر است که زمان‌های پردازش و موعدهای تحویل روی توزیع‌های یکنواخت گسسته تولید خواهند شد. تمامی کدها در نرم افزار MATLAB نوشته شده و روی رایانه شخصی با پردازشگر Intel ( 2.1 MHz Core 2 Duo) و ۴ گیگا بایت حافظه رم، اجرا شده است.

## مدل ریاضی مسئله

در این تحقیق از برنامه ریزی عدد صحیح (صفر و یک) برای مدل سازی مسئله مورد نظر استفاده کرده‌ایم. مدل ریاضی مسئله به صورت زیر می‌باشد:

$$\min Z_1 = \sum_{k=1}^{n} max\left\{\left(\sum_{i=1}^{n}\sum_{j=1}^{k} X_{ij}p_i - \sum_{i=1}^{n} X_{ik}d_i\right), 0\right\} \quad (۱)$$

$$\min Z_2 = \frac{1}{n}\sum_{k=1}^{n}\left(W_{[k]} - \frac{1}{n}\sum_{k=1}^{n} W_{[k]}\right)^2 \quad (۲)$$

$$W_{[k]} = 0 \quad if \quad k = 1 \quad (۳)$$

$$W_{[k]} = \sum_{i=1}^{n}\sum_{j=1}^{k-1} X_{ij}p_i \quad if \quad k \neq 1 \quad (۴)$$

$$\sum_{i=1}^{n} X_{ij} = 1 \quad \forall j \qquad (۵)$$

$$\sum_{j=1}^{n} X_{ij} = 1 \quad \forall i \qquad (۶)$$

$$X_{ij} \in \{0,1\} \qquad (۷)$$

$Z_1$: دیر کرد کل

$Z_2$: واریانس زمان انتظار

$W_{[k]}$: زمان انتظار کاری که در توالی $k$ ام قرار می‌گیرد

$d_i$: موعد تحویل کار $i$ ام

$p_i$: زمان پردازش کار $i$ ام

$X_{ij}$: متغیر تخصیص کار $i$ ام به توالی $j$ام ( اگر کار $i$ ام به توالی $j$ام اختصاص یابد ۱ و در غیر این صورت صفر خواهد بود)

همان طور که در بخش قبل اشاره شد برای مسائل دسته اول ($n$ =5, 10, 15) انحراف نتایج الگوریتم‌های طراحی شده از نتایج بهینه بدست می‌آید. برای دستیابی به نتایج بهینه تابع هدف زیر را در LINGO حداقل می‌کنیم. این تابع از نوع LP-metric است و آریانژاد[1] و دیگران (۲۰۰۹) در تحقیق خود از آن استفاده نموده‌اند:

$$f(Z_1, Z_2) = w\left[\left|Z_1 - Z_{1opt}^*\right|/\left(1 + Z_{1opt}^*\right)\right] + (1-w)\left[\left|Z_2 - Z_{2opt}^*\right|/\left(1 + Z_{2opt}^*\right)\right] \qquad (۸)$$

لازم به ذکر است که در این مدل $Z_{1opt}$ و $Z_{2opt}$ به ترتیب نشان‌دهنده مقادیر بهینه برای توابع دیرکرد کل و واریانس زمان انتظار هستند که برای هر مسئله از مسائل دسته اول، از قبل و به صورت مجزا توسط نرم افزار LINGO بدست آمده است. همچنین باید گفت قرار

[1] Aryanezhad

دادن عدد ۱ در مخرج صرفا برای جلوگیری از بوجود آمدن $\frac{Z_1}{0}$ یا $\frac{Z_2}{0}$ در حالات خاصی است که $Z_1$ یا $Z_2$ مقادیر بهینه برابر صفر اتخاذ می‌کنند.

## تضاد اهداف در تحقیق

برای هر مسئله MO[1] توابع هدف می‌توانند به سه صورت: کاملاً متضاد، نا متضاد و نسبتاً متضاد تقسیم‌بندی شوند. در یک مجموعه راه حل داده شده φ ، یک بردار از توابع هدف $F = \{f_1, f_2, \dots, f_m\}$ ، کاملاً متضاد گفته می‌شود، اگر هیچ دو راه حل $P_b$ و $P_a$ در مجموعه φ یافت نشود ، طوریکه $F_a \prec F_b$ یا $F_b \prec F_a$ (که $F_a \prec F_b$ به این معنی است که راه حل $P_b$ بر راه حل $P_a$ غلبه دارد )، در چنین مسئله‌ای بهینه یابی لازم نیست ، زیرا مجموعه جواب φ نشان دهنده مجموعه جواب بهینه global پارتو می‌باشد. در حالت دیگر توابع هدف نامتضاد گفته می‌شود ، اگر به ازای تمام زوج راه حل‌های $P_b$ و $P_a$ از مجموعه φ شرایط $F_a \prec F_b$ یا $F_b \prec F_a$ بر قرار باشد. این گروه از مسائل MO به راحتی می‌توانند تبدیل به مسائل تک هدفه شوند که می‌تواند با در نظر گرفتن دلخواه یکی از اهداف و یا ترکیبی از توابع به صورت یک تابع اسکالر باشد. بنابراین هر بهبودی در یک هدف منجر به بهبود در اهداف باقیمانده می‌شود. در حالت سوم یک بردار از توابع هدف $F = \{f_1, f_2, \dots, f_m\}$ ، نسبتاً متضاد گفته می‌شود اگر مجموعه‌های ناتهی از راه‌حل‌های $P_b$ و $P_a$ وجود داشته باشد طوریکه $F_a \prec F_b$ یا $F_b \prec F_a$ باشد. باید گفت بسیاری از مسائل واقعی بهینه سازی MO متعلق به این دسته‌اند که در آن مجموعه بهینه پارتو نشان‌دهنده تبادل (Trade off) بین اهداف متضاد مورد نظر است (Tan et al., 2005). برای اینکه نشان دهیم که دو تابع دیر کرد کل و واریانس زمان انتظار از نوع سوم در طبقه‌بندی فوق هستند مسئله ای را با ۶ کار طراحی کرده و با حل آن نشان می‌دهیم که این دو تابع از نظر طبقه بندی در دسته سوم قرار می‌گیرند. در این مسئله موعد تحویل همه کارها مشترک و برابر $d = 6$ و

[1] Multi Objective

زمان‌های پردازش کارها $p_1 = 5$، $p_2 = 7$، $p_3 = 6$، $p_4 = 3$، $p_5 = 11$ و $p_6 = 2$ می‌باشد. حال ۳ راه حل (توالی) را در نظر گرفته و مقدار دو تابع هدف مورد نظر تحقیق را برای آن‌ها محاسبه می‌کنیم.

$S_1 = (1,2,3,4,5,6)$ ➔ $Z_1 = 87\,, Z_2 = 111.09$
$S_2 = (5,3,2,1,6,4)$ ➔ $Z_1 = 110\,, Z_2 = 116.09$
$S_3 = (2,5,1,6,4,3)$ ➔ $Z_1 = 99\,, Z_2 = 101.8$

با توجه به اعداد بدست آمده برای توابع هدف، می‌توانیم بگوییم $F_2 \prec F_3$ (راه حل سوم به راه حل دوم غلبه دارد) و $F_2 \prec F_1$ ( راه حل اول به راه حل دوم غلبه دارد). اما همان‌طور که دیده می‌شود $F_1 \prec F_3$ یا $F_3 \prec F_1$ بر قرار نیست. زیرا در یکی از توابع ( واریانس زمان انتظار) $S_3$ به $S_1$ و در تابع هدف دیگر (متوسط دیرکرد) $S_1$ به $S_3$ برتری دارد. با نتایج به دست آمده می‌توان گفت که این توابع شرایط لازم برای قرار گرفتن در دسته سوم را دارند و تمام شرایط برای قرار گرفتن در دسته اول و دوم را نقض می‌کنند، بنابراین از نوع نسبتاً متضاد هستند.

## تولید داده‌های مسئله

همان طور که در بخش قبل اشاره شد، این مسئله در دو دسته بررسی می‌شود. دسته اول که مربوط به کارهای با تعداد $n = 5, 10, 15$ و دسته دوم مربوط به کارهای با تعداد $n = 20, 50, 100$ می‌باشد. زمان‌های پردازش در هر دو دسته در دو حالت «زمان‌های پردازش بالا» و «زمان‌های پردازش پایین» تولید می‌گردد. زمان‌های پردازش بالا در بازه $[1,100]$ و زمان‌های پردازش پایین در بازه $[1,25]$ و روی توزیع یکنواخت گسسته تولید می‌شود. موعدهای تحویل در چهار بازه مختلف و به صورت ضرایبی از مجموع زمان‌های پردازش(P) تولید می‌شوند که این چهار بازه در جدول ۱ آمده است. بازه‌های مختلف موعدهای تحویل

به این دلیل استفاده می‌شود که بتوانیم فشردگی[1] موعدهای تحویل را به وسیله تولید داده‌ها در این بازه‌ها تنظیم کنیم. باید گفت بازه‌های زمان‌های پردازش و موعدهای تحویل مورد استفاده در این تحقیق همان است که در تحقیق کوکسالان[2] و بوراک[3] (۲۰۰۳) بیان شده است.

| بازه | حدود موعدهای تحویل |
|---|---|
| I | [0.0P,0.4P] |
| II | [0.1P,0.3P] |
| III | [0.25P,0.45P] |
| IV | [0.3P,1.3P] |

جدول ۱- بازه‌های موعدهای تحویل

همان طور که قبلاً گفته شد توابع هدف با سه وزن مختلف $w = 0.3, 0.5, 0.7$ با یکدیگر ترکیب خواهند شد و در نهایت چهار الگوریتم ژنتیک برای آن‌ها ارائه می‌شود. با توجه به این گفته‌ها $2 \times 3 \times 4 \times 3 \times 4 = 288$ (۲حالت در زمان پردازش، ۳ حالت در تعداد کارها، ۴ حالت در موعد تحویل، ۳ حالت در وزن توابع هدف، ۴ الگوریتم ژنتیک) کلاس از مسائل برای هر دسته بررسی شده است. برای هر کدام از این کلاس‌ها ۱۰ مسئله به صورت تصادفی تولید می‌شود. به این صورت برای هر دسته ۲۸۸۰ مسئله مورد ارزیابی قرار گرفته است. علاوه بر این برای دسته اول و به منظور بدست آوردن انحراف نتایج الگوریتم‌ها از مقادیر بهینه، $2 \times 3 \times 4 \times 3 \times 10 = 720$ ( ۲حالت در زمان پردازش، ۳ حالت در تعداد کارها، ۴ حالت در موعد تحویل، ۳ حالت در وزن توابع هدف، ۱۰ مسئله برای هر حالت) مسئله به وسیله LINGO و بدین ترتیب در کل ۶۴۸۰ مسئله حل شده است.

[1] Tightness
[2] Köksalan
[3] Burak

## کاربرد الگوریتم‌های ژنتیک در تحقیق

الگوریتم فرا ابتکاری مورد استفاده در این تحقیق الگوریتم ژنتیک می‌باشد. الگوریتم ژنتیک یک روش فرا ابتکاری بر مبنای جستجو با استفاده از قواعد ژنتیک و انتخاب طبیعی می‌باشد. عمومی‌ترین شکل الگوریتم ژنتیک دارای سه عملگر انتخاب، تقاطع وجهش می‌باشد که در این تحقیق نیز از آن استفاده شده است. در اینجا دو سطح برای معرفی الگوریتم ژنتیک ارائه شده تعریف می‌گردد. سطح اول الگوریتم ژنتیک اصلی که برای حل مسئله ارائه می‌شود و سطح دوم الگوریتم ژنتیک فرعی که در بخشی از الگوریتم‌های اصلی از آن استفاده می‌شود.

## الگوریتم ژنتیک اصلی

این سطح شامل الگوریتم‌های اصلی برای حل مسئله می‌باشد و ویژگی‌ها و عملگرهای آن در ادامه توضیح داده خواهد شد.

## کدینگ راه حل (طرح کروموزوم)

برای مسئله تک ماشین مورد نظر این تحقیق، از نمایش جایگشتی راه حل که یک جایگشت از اعداد صحیح $1,\ldots,n$ می‌باشد استفاده شده است، که در آن $n$ نشان دهنده تعداد کارها می‌باشد.

## تابع برازش

تابع برازش الگوریتم ژنتیک ارائه شده در این تحقیق، یک ترکیب خطی از دیرکرد کل و واریانس زمان انتظار می‌باشد. این تابع نیز مشابه تابع ۸ و از نوع LP-metric است که آن را به صورت زیر تعریف می‌کنیم:

$$fitness = w[|Z_1(S) - Z_1^*|/(1 + Z_1^*)] + (1 - w)[|Z_2(S) - Z_2^*|/(1 + Z_2^*)] \quad (۹)$$

در تابع بالا $Z_1$ و $Z_2$ به ترتیب نشان‌دهنده دیرکرد کل و واریانس زمان انتظار می‌باشند و $S$ نشان‌دهنده یک کروموزوم (توالی) در جمعیت می‌باشد. به این دلیل که $Z_1^*$ و $Z_2^*$ در ابتدای امر موجود نیستند آن‌ها را به دو روش که در ادامه توضیح داده می‌شود جایگزین می‌کنیم و توابع ایجاد شده از این روش‌ها را پویا و ایستا می‌نامیم.

## تابع برازش پویا

جایگزین شدن $Z_1^*$ و $Z_2^*$ در این نوع تابع برازش مشابه روشی است که در الگوریتم WBGA برای به دست آوردن مقادیر مرزی اهداف استفاده می‌شود (Deb, 2009)، که در تحقیق حاضر با کاربردی جدید ارائه شده است. بدین ترتیب که مقادیر بسیار بزرگی در تکرار اول به $Z_1^*$ و $Z_2^*$ تعلق می‌گیرد. اگر در تکرارهای بعد مقادیر بهتری در جمعیت برای این مقادیر پیدا شد، آنگاه $Z_1^*$ و $Z_2^*$ با مقادیر بهتر جایگزین می‌شوند. بنابراین مقادیر $Z_1^*$ و $Z_2^*$ می‌توانند در طی نسل‌ها چندین بار جایگزین شوند.

## تابع برازش ایستا

در این حالت، مقادیر $Z_1^*$ و $Z_2^*$ را قبل از شروع الگوریتم ژنتیک اصلی به وسیله الگوریتم‌های ژنتیک فرعی که در ادامه توضیح داده خواهد شد، بدست می‌آوریم. بعد از این مرحله الگوریتم ژنتیک اصلی شروع به کار کرده و مقادیر $Z_1^*$ و $Z_2^*$ تا پایان تغییر نمی‌کنند.

## جمعیت اولیه

از دو نوع جمعیت اولیه در این تحقیق استفاده شده است و اندازه جمعیت برابر ۳۰ (POP=30) می‌باشد. نوع اول یک جمعیت اولیه کاملاً تصادفی است و نوع دوم جمعیت اولیه هیوریستیک می‌باشد. در نوع اول جایگشت‌های تصادفی برای تولید جمعیت اولیه تصادفی استفاده می‌شود. در نوع دوم جمعیت اولیه به ۷ بخش تقریباً برابر تقسیم می‌شود. شش

بخش اول ( بخش ۱ تا بخش ۶) به وسیله روش‌های هیوریستیک ایجاد شده و اندازه هر بخش برابر است با $PS = \lfloor POP/7 \rfloor$ و بخش آخر (بخش ۷) همانند جمعیت اولیه نوع اول، به صورت جایگشت‌های تصادفی ایجاد شده و اندازه آن $RS = POP - 6PS$ می‌باشد. برای بدست آوردن کروموزوم‌ها در هر یک از ۶ بخش اول از مراحل زیر استفاده می‌شود (C نشان‌دهنده شمارنده کروموزوم‌ها در هر بخش می‌باشد):

۱. بدست آوردن توالی با استفاده از هیوریستیک و قرار دادن آن به عنوان کروموزوم اول در بخش ، **C←1**

۲. انتخاب دو مکان در کروموزوم اول

۳. جابجایی ژن‌های موجود در مکان‌های انتخاب شده

۴. قرار دادن کروموزوم بدست آمده به عنوان کروموزوم بعدی در بخش، **C←C+1**

۵. اگر **PS-C=0** آنگاه توقف کن، در غیر این صورت به مرحله ۲ برو

برای هر کدام از بخش‌های ۱ تا ۶ از یک هیوریستیک استفاده می‌کنیم. بنابراین در کل از ۶ هیوریستیک استفاده می‌شود که این ۶ هیوریستیک عبارتند از: ۱- قاعده EDD که اگر در توالی بیش از یک کار دیرکرد دار نباشد، مقدار دیرکرد کل را کمینه می‌کند. ۲- قاعده AU (Apparetn Urgency) که توسط مورتون[1] و دیگران (۱۹۸۴) برای مسئله دیرکرد کل ارائه شده است. ۳- قاعده ای هیوریستیک که توسط یون[2] و لی[3] (۲۰۱۱) با عنوان Heuristic 1 برای مسئله دیرکرد کل معرفی شده است. ۴- هیوریستیکی برای حداقل کردن دیرکرد کل که توسط پانیرسلوام[4] (۲۰۰۶) ارائه شده است. ۵- هیوریستیکی برای حداقل کردن واریانس زمان انتظار که توسط ایلون[5] و چادوری[6] (۱۹۷۷) با نام Method

[1] Morton
[2] Yoon
[3] Lee
[4] Panneerselvam
[5] Eilon
[6] Chowdhury

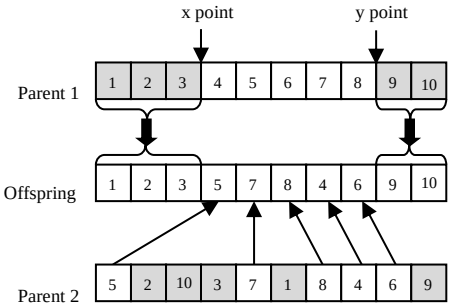
1.1 ارائه شده است. ۶- هیوریستیکی که بازهم توسط ایلون و چادوری (۱۹۷۷) برای حداقل کردن واریانس زمان انتظار با نام Method 1.2 معرفی شده است.

## انتخاب والد

در هر تکرار الگوریتم نیمی از جمعیت که دارای مقادیر بهتر برازش می‌باشند به تکرار بعدی (استخر جفت‌گیری) راه پیدا می‌کنند. می‌توان گفت که نرخ انتخاب در این الگوریتم ۰,۵ می‌باشد. برای اینکه فرزندی به وجود آید باید دو والد از استخر جفت‌گیری انتخاب شوند. در این تحقیق هر دوی والد اول و والد دوم با روش انتخاب مسابقه‌ای انتخاب می‌شوند. در انتخاب مسابقه ای T کروموزوم به صورت تصادفی انتخاب می‌شود و هرکدام از آن‌ها که مقدار برازش بهتری داشته باشد به عنوان والد انتخاب می‌شود. در تحقیق حاضر T=4 به عنوان مقداری مناسبی برای اندازه مسابقه تعیین شده است.

## عملگر تقاطع

در الگوریتم ارائه شده از عملگر تقاطع دو نقطه‌ای با یک فرزند استفاده شده است. برای مثال اگر والد اول و والد دوم به ترتیب (1,2,3,4,5,6,7,8,9,10) و (5,2,10,3,7,1,8,4,6,9) باشند و نقاط تقاطع ۳ و ۸ باشند فرزند به صورت شکل ۱ به وجود می‌آید.




شکل ۱- عملگر تقاطع

تفاوت عمده‌ای که روش تقاطع تحقیق حاضر با روش تقاطع معمولی دارد این است که در روش معمول برای تعیین نقاط تقاطع از اعداد تصادفی استفاده می‌شود و تولید اعداد تصادفی در الگوریتم باعث افزایش زمان اجرای الگوریتم در CPU خواهد شد، اما در این روش، نقاط تقاطع، تابعی از تعداد کارها در مسئله ($n$) خواهد بود و این امر باعث کاهش زمان اجرای الگوریتم و کاراتر شدن آن می‌شود. نقاط تقاطع "x" و "y" به صورت زیر تعریف می‌شوند:

$$x = \lfloor n/k \rfloor + 1 \qquad (۱۰)$$

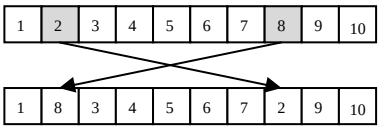
$$y = n - x \qquad (۱۱)$$

در رابطه‌های بالا $k$ پارامتری است با شرط ($k > 2$) که کنترل کننده نقاط تقاطع می‌باشد. اگر $k$ افزایش یابد و به سمت $n$ میل کند، آنگاه تعداد ژن‌هایی که فرزند از والد اول به ارث می‌برد کاهش می‌یابد. و اگر k به سمت ۲ برود باعث کاهش تعداد ژن‌های به ارث رسیده از والد دوم می‌شود. در این تحقیق $k$=6 به عنوان مقداری مناسب برای بررسی مسائل انتخاب شده است.

## عملگر جهش

نرخ جهش در این الگوریتم ۰٫۱ می‌باشد. به این معنی که ۱۰ درصد از ژن‌های موجود در هر تکرار جهش خواهند یافت. برای جهش، کروموزومی به صورت تصادفی انتخاب شده و دو

مکان در کروموزوم به صورت تصادفی انتخاب می‌شوند، آنگاه ژن‌های موجود در این دو مکان با هم جابجا می‌شوند. لازم به ذکر است که بهترین کروموزوم (کروموزومی که مقدار برازش بهتری دارد) در هر نسل به علت نخبه گرایی (elitism) مورد جهش واقع نمی‌شود.



شکل ۲- عملگر جهش

## شرایط توقف

در این بخش دو پارامتر زیر را برای شرایط توقف تعریف می‌کنیم:

a: تعداد تکرارهای پی در پی که در آن بهترین کروموزوم در جمعیت تغییر نمی‌کند.

b: تعداد کل تکرارها

حال اگر بهترین کروموزوم در جمعیت بعد از a=40 نسل متوالی تغییر نکند و یا در صورتی که به تکرار b=100 در کل برسیم،الگوریتم متوقف می‌شود.

## الگوریتم‌های ارائه شده اصلی

با توجه به ویژگی‌ها و عملگرهای ذکر شده، ۴ الگوریتم ژنتیک اصلی به صورت زیر ارائه می‌شود:

GA-A : این الگوریتم ژنتیک از تابع برازش پویا و جمعیت اولیه کاملاً تصادفی بهره می‌برد.

GA-B : این الگوریتم ژنتیک از تابع برازش ایستا و جمعیت اولیه هیوریستیک استفاده می‌کند.

GA-C : این الگوریتم ژنتیک از تابع برازش ایستا و جمعیت اولیه کاملاً تصادفی استفاده می‌کند.

GA-D : این الگوریتم ژنتیک از تابع برازش ایستا و جمعیت اولیه هیوریستیک استفاده می‌کند.

اگر ITR نشان‌دهنده شمارنده تکرار و NC نشان‌دهنده شمارنده تکرارهای پیاپی‌ای باشد که در آن بهترین کروموزوم تغییر نکند، آنگاه مراحل الگوریتم‌های ژنتیک GA-A و GA-B به صورت زیر باشد:

۱- ایجاد جمعیت اولیه، **ITR←1** و **NC←0**

۲- تخصیص مقادیر بزرگ به $Z_1^*$ و $Z_2^*$

۳- ارزیابی $Z_1$ برای همه کروموزوم‌های جمعیت و اگر $Z_1 < Z_1^*$ آنگاه $Z_1^* \leftarrow Z_1$

۴- ارزیابی $Z_2$ برای همه کروموزوم‌های جمعیت و اگر $Z_2 < Z_2^*$ آنگاه $Z_2^* \leftarrow Z_2$

۵- ارزیابی به وسیله تابع برازش برای همه کروموزوم‌های جمعیت

۶- انتخاب دو والد به وسیله انتخاب مسابقه ای

۷- به کار بردن عملگر تقاطع و جایگزینی کروموزوم‌های حذف شده به وسیله فرزندان ایجاد شده.

۸- به کار بردن عملگر جهش

۹- اگر بهترین مقدار برازش در تکرار **ITR** برابر مقدار برازش در تکرار **ITR-1** باشد، آنگاه **NC←NC+1** در غیر این صورت **NC←0**

۱۰- **ITR←ITR+1**

۱۱-اگر **ITR>b** یا **NC≥a** آنگاه توقف کن در غیر این صورت به گام ۳ برو

و مراحل الگوریتم‌های ژنتیک **GA-C** و **GA-D** به صورت زیر خواهد بود:

۱- ایجاد جمعیت اولیه، **ITR←1** و **NC←0**

۲- بدست آوردن مقادیر $Z_1^*$ و $Z_2^*$ به وسیله الگوریتم‌های ژنتیک فرعی (**S1-GA** و **S2-GA** که این الگوریتم‌ها در بخش بعدی توضیح داده خواهد شد)

۳- ارزیابی به وسیله تابع برازش برای همه کروموزوم‌های جمعیت

۴- انتخاب دو والد به وسیله انتخاب مسابقه ای

۵- به کار بردن عملگر تقاطع و جایگزینی کروموزوم‌های حذف شده به وسیله فرزندان ایجاد شده.

۶- به کار بردن عملگر جهش

۷- اگر بهترین مقدار برازش در تکرار **ITR** برابر مقدار برازش در تکرار **ITR-1** باشد، آنگاه **NC←NC+1** در غیر این صورت **NC←0**

۸- **ITR←ITR+1**

۹- اگر **ITR>b** یا **NC≥a** آنگاه توقف کن در غیر این صورت به گام ۳ برو

## الگوریتم ژنتیک فرعی

این سطح شامل دو الگوریتم ژنتیک S1-GA و S2-GA می‌شود که به ترتیب برای به دست آوردن $Z_1^*$ و $Z_2^*$ در حالت تابع برازش ایستا استفاده می‌شوند. این الگوریتم‌ها بسیار شبیه به الگوریتم‌های ژنتیک اصلی می‌باشند. اما تفاوت‌هایی دارند در زیر بیان می‌شوند:

۱- جمعیت اولیه: اندازه جمعیت اولیه در الگوریتم‌های ژنتیک فرعی نصف اندازه جمعیت در الگوریتم‌های ژنتیک اصلی هستند (POP=15). برای تولید جمعیت اولیه در این حالت جمعیت اولیه را به دو قسمت تقسیم می‌کنیم (بخش ۱ و بخش ۲). اولین بخش (بخش ۱) بخش هیوریستیک جمعیت اولیه و اندازه آن $PS = \lfloor POP/7 \rfloor$ می‌باشد. دومین بخش (بخش ۲) بخش تصادفی جمعیت اولیه و اندازه آن $RS = POP - PS$ می‌باشد. برای تولید کروموزوم‌های بخش ۱ از همان گام‌هایی که در قسمت جمعیت اولیه در سطح الگوریتم‌های ژنتیک اصلی آمده استفاده می‌کنیم. هیوریستیک ۳ و هیوریستیک ۶ به ترتیب برای S1-GA و S2-GA استفاده می‌شود. بخش ۲ نیز به صورت جایگشت‌های تصادفی تولید می‌شود.

۲- تابع برازش: در الگوریتم S1-GA که برای بدست آوردن $Z_1^*$ بکار می‌رود تابع برازش همان تابع دیرکرد کل ($Z_1$) و در الگوریتم S2-GA که برای بدست آوردن $Z_2^*$ بکار می‌رود تابع برازش همان تابع واریانس زمان انتظار ($Z_2$) می‌باشد.

۳- شرط توقف: اگر بهترین کروموزوم در a=20 تکرار متوالی تغییر نکند و یا در صورتی که به تکرار b=50 در کل برسیم، الگوریتم متوقف می‌شود.

عملگرهای انتخاب، تقاطع و جهش در این دو الگوریتم همانند الگوریتم‌های ژنتیک اصلی و گام‌های S1-GA و S2-GA مطابق زیر می‌باشد:

۱- ایجاد جمعیت اولیه، ITR←1 و NC←0

۲- ارزیابی به وسیله تابع برازش برای همه کروموزوم‌های جمعیت

۳- انتخاب دو والد به وسیله انتخاب مسابقه‌ای

۴- به کار بردن عملگر تقاطع و جایگزینی کروموزوم‌های حذف شده به وسیله فرزندان ایجاد شده.

۵- به کار بردن عملگر جهش

۶- اگر بهترین مقدار برازش در تکرار ITR برابر مقدار برازش در تکرار ITR-1 باشد، آنگاه NC←NC+1 در غیر این صورت NC←0

۷- ITR←ITR+1

۸- اگر ITR>b یا NC≥a آنگاه توقف کن در غیر این صورت به گام ۲ برو

## روش ارزیابی نتایج

در این تحقیق در هر دسته از مسائل گفته شده دو شاخص برای ارزیابی الگوریتم‌های ژنتیک ارائه شده استفاده می‌شود. شاخص اول در دسته اول مسائل، انحرافات نتایج از مقادیر بهینه و در دسته دوم انحرافات نتایج از بهترین نتایج بدست آمده می‌باشد. و شاخص دوم در هر دو دسته زمان‌های اجرای الگوریتم خواهد بود. برای روشن‌تر شدن شاخص انحرافات آن‌ها را برای هر دو دسته از مسائل در ادامه توضیح می‌دهیم.

**شاخص انحرافات در مسائل دسته اول**

این دسته مربوط مسائلی می‌شود که تعداد کارها در آن $n$=5, 10, 15 می‌باشد. در این دسته انحراف نتایج حاصل شده از الگوریتم‌های ژنتیک از مقادیر حاصله توسط نرم افزار LINGO 8 به دست می‌آید. به دلیل اینکه مقادیر حاصل از توابع برازش الگوریتم‌های ژنتیک و تابع هدف بهینه شده در LINGO دارای یک مقیاس نمی‌باشند، برای بدست آوردن انحرافات ابتدا آن‌ها را هم مقیاس می‌کنیم. برای این کار $Z_1$ و $Z_2$ حاصل از الگوریتم‌های GA-A, GA-B, GA-C, GA-D برای هر مسئله را در رابطه شماره ۸ و با پارامترهای متناظر قرار می‌دهیم و مقدار $f(Z_1, Z_2)$ را برای هرکدام به دست می‌آوریم. سپس مقدار انحراف مقادیر هم مقیاس شده را از مقدار بهینه به‌صورت زیر به دست می‌آوریم:

$$\%deviation = 100 \times \left[\left(f - f_{opt}\right)/f_{opt}\right] \quad (۱۲)$$

لازم به ذکر است که $f_{opt}$ نشان‌دهنده مقدار بهینه هر مسئله می‌باشد که با توجه به تابع شماره ۸ و بوسیله LINGO حاصل شده است.

**شاخص انحرافات در مسائل دسته دوم**

این دسته مربوط به مسائلی می‌شود که تعداد کارها در آن n=20, 50, 100 می‌باشد. در این دسته انحراف نتایج حاصله از الگوریتم‌های ژنتیک از بهترین مقادیر حاصل شده از چهار الگوریتم به دست می‌آید. به دلیل اینکه مقادیر حاصل از توابع برازش الگوریتم‌های ژنتیک مقیاس مشترکی ندارند برای بدست آوردن انحرافات ابتدا آن‌ها را هم مقیاس می‌کنیم. در اینجا نیز $Z_1$ و $Z_2$ حاصل از الگوریتم‌های GA-A, GA-B, GA-C, GA-Dبرای هر مسئله را در تابع زیر و با پارامترهای متناظر قرار می‌دهیم و مقدار $f(Z_1, Z_2)$ را برای هرکدام به دست می‌آوریم.

$$f(Z_1, Z_2) = w[|Z_1 - Z_{1GA}^*|/(1 + Z_{1GA}^*)] + (1 - w)[|Z_2 - Z_{2GA}^*|/(1 + Z_{2GA}^*)] \quad (۱۳)$$

لازم به ذکر است $Z^*_{1GA}$ و $Z^*_{2GA}$ به ترتیب نشان‌دهنده مقادیر نزدیک بهینه برای تابع دیرکرد کل و واریانس زمان انتظار می‌باشند. این مقادیر برای هر مسئله و به وسیله الگوریتم‌های ژنتیک فرعی ارائه شده، با شرایط توقف a=80 و b=200 به دست می‌آیند.سپس مقدار درصد انحراف نتایج هم مقیاس شده را از بهترین مقدار به صورت زیر بدست می‌آوریم:

$$\%deviation = 100 \times [(f - f_{best})/f_{best}] \quad (۱۴)$$

لازم به ذکر است که $f_{\text{best}}$ نشان‌دهنده بهترین مقدار به دست آمده برای هر مسئله از چهار الگوریتم طراحی شده می‌باشد.

## یافته‌های پژوهش

در این بخش میانگین انحرافات در دو جدول ارائه می‌شود. جدول اول (جدول ۲) نشان‌دهنده میانگین انحرافات تمام کلاس‌های دسته اول مسائل از مقادیر بهینه و جدول دوم (جدول ۳) نشان‌دهنده میانگین انحرافات تمام کلاس‌های دسته دوم مسائل، از بهترین مقادیر می‌باشد. هر عدد در این جداول نشان‌دهنده میانگین انحرافات ۱۰ مسئله تولید شده برای هر کلاس است. در قسمت بعد این بخش برای کلاس‌های موجود در وزن‌های مختلف و تعداد کارهای مختلف میانگین زمان‌های اجرای هر الگوریتم را به دست می‌آوریم و در این صورت می‌توانیم برای مسائل با تعداد کارهای مختلف، زمان‌های اجرای الگوریتم‌ها در وزن‌های مختلف را مقایسه کنیم.

## بررسی میانگین انحرافات

برای مقایسه کلی الگوریتم‌های ارائه شده در جداول ۲ و ۳ سطری با عنوان "میانگین" تعریف شده است. این سطر میانگینی برای میانگین انحرافات تمام کلاس‌های مسائل در ستون خود می‌باشد. با استفاده از این میانگین می‌توانیم الگوریتم‌ها را به صورت کلی‌تر و در وزن‌های

مختلف و زمان‌های پردازش بالا و پایین، مقایسه کنیم.

بر این اساس می‌توان از جدول ۲ نتیجه گرفت که به طور کلی الگوریتم GA-D در وزن $w=0.3$ و الگوریتم GA-B در وزن‌های $w=0.5$ و $w=0.7$ چه در زمان‌های پردازش بالا و چه در زمان‌های پردازش پایین در این دسته از مسائل از نظر میانگین انحرافات بهتر از الگوریتم‌های دیگر هستند. و نیز مطابق جدول ۳ می‌توان گفت به طور کلی الگوریتم GA-B در این دسته از مسائل در تمامی وزن‌ها و زمان‌های پردازش بالا و پایین از نظر میانگین انحرافات بهتر از الگوریتم‌های دیگر است.

## بررسی میانگین زمان‌های اجرا

برای مقایسه الگوریتم‌ها از نظر زمان‌های اجرا، مطابق جدول ۴، میانگین زمان‌های اجرا بر حسب وزن توابع هدف و تعداد کارها، در مسائل به دست آمده است. همان‌طور که در جدول دیده می‌شود به جز در وزن $w=0.7$ متعلق به $n=100$ که الگوریتم GA-C میانگین زمان اجرای کمتری دارد در سایر حالات الگوریتم GA-D است که کمترین متوسط زمان اجرا را به خود اختصاص داده است. بنابراین می‌توانیم به طورکلی الگوریتم GA-D را از لحاظ زمان اجرا، بهتر از بقیه الگوریتم‌های ارائه شده بدانیم.

جدول ۲- میانگین انحرافات نتایج الگوریتم‌ها از مقادیر بهینه برای دسته اول مسائل

| | n | d | w = 0.3 | | | | w = 0.5 | | | | w = 0.7 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | GA-A | GA-B | GA-C | GA-D | GA-A | GA-B | GA-C | GA-D | GA-A | GA-B | GA-C | GA-D |
| زمان های پردازش پایین | 5 | I | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.061 | 0.000 | 0.000 | 0.064 | 0.000 |
| | | II | 0.029 | 0.029 | 0.054 | 0.054 | 0.000 | 0.000 | 0.025 | 0.000 | 0.000 | 0.000 | 0.021 | 0.093 |
| | | III | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.094 | 0.020 | 0.071 | 0.000 | 0.071 | 0.000 |
| | | IV | 0.000 | 0.000 | 0.053 | 0.097 | 0.000 | 0.000 | 0.105 | 0.019 | 0.000 | 0.000 | 0.080 | 0.088 |
| | 10 | I | 0.493 | 0.301 | 0.127 | 0.109 | 0.069 | 0.370 | 0.043 | 0.164 | 0.053 | 0.101 | 0.184 | 0.308 |
| | | II | 0.012 | 0.014 | 0.083 | 0.316 | 0.020 | 0.000 | 0.500 | 0.022 | 0.106 | 0.000 | 0.378 | 0.000 |
| | | III | 1.211 | 0.245 | 0.149 | 0.025 | 0.392 | 0.531 | 0.849 | 0.785 | 0.107 | 0.252 | 0.598 | 0.610 |
| | | IV | 0.067 | 0.500 | 0.654 | 0.628 | 0.357 | 0.500 | 0.599 | 0.500 | 0.346 | 0.747 | 0.730 | 0.500 |
| | 15 | I | 0.436 | 0.193 | 0.671 | 0.141 | 5.042 | 0.030 | 7.232 | 0.861 | 0.519 | 0.554 | 1.605 | 0.522 |
| | | II | 0.453 | 0.112 | 0.544 | 0.191 | 0.826 | 0.281 | 0.632 | 0.439 | 1.400 | 0.364 | 1.182 | 0.450 |
| | | III | 0.211 | 0.090 | 0.320 | 0.076 | 0.349 | 0.096 | 0.858 | 0.309 | 0.659 | 0.447 | 1.859 | 0.799 |
| | | IV | 0.694 | 0.562 | 1.223 | 0.340 | 0.786 | 0.215 | 1.169 | 0.491 | 0.673 | 0.430 | 1.043 | 0.548 |
| میانگین | | | 0.301 | 0.171 | 0.323 | 0.165 | 0.653 | 0.169 | 1.009 | 0.306 | 0.328 | 0.241 | 0.651 | 0.326 |
| زمان های پردازش بالا | 5 | I | 0.000 | 0.000 | 0.028 | 0.014 | 0.000 | 0.000 | 0.054 | 0.035 | 0.000 | 0.000 | 0.000 | 0.000 |
| | | II | 0.055 | 0.055 | 0.065 | 0.116 | 0.011 | 0.011 | 0.011 | 0.011 | 0.000 | 0.000 | 0.000 | 0.000 |
| | | III | 0.023 | 0.023 | 0.031 | 0.028 | 0.000 | 0.000 | 0.017 | 0.006 | 0.000 | 0.000 | 0.000 | 0.000 |
| | | IV | 0.000 | 0.000 | 0.025 | 0.000 | 0.000 | 0.000 | 0.008 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| | 10 | I | 0.051 | 0.084 | 0.314 | 0.223 | 0.057 | 0.071 | 0.310 | 0.298 | 0.455 | 0.015 | 0.269 | 0.015 |
| | | II | 0.645 | 0.644 | 0.000 | 0.017 | 0.087 | 0.032 | 0.410 | 0.032 | 0.310 | 0.360 | 1.113 | 0.677 |
| | | III | 0.122 | 0.000 | 0.175 | 0.137 | 0.184 | 0.200 | 0.936 | 0.413 | 0.186 | 0.451 | 0.688 | 0.498 |
| | | IV | 0.210 | 0.000 | 0.099 | 0.029 | 0.554 | 0.000 | 0.695 | 0.020 | 0.266 | 0.027 | 0.195 | 0.000 |
| | 15 | I | 0.726 | 0.403 | 0.723 | 0.323 | 0.334 | 0.161 | 0.646 | 0.200 | 0.944 | 0.130 | 0.789 | 0.077 |
| | | II | 0.621 | 0.396 | 1.312 | 0.358 | 0.843 | 0.212 | 0.751 | 0.240 | 1.205 | 0.356 | 1.762 | 0.902 |
| | | III | 0.334 | 0.098 | 0.213 | 0.081 | 0.305 | 0.031 | 0.578 | 0.115 | 0.520 | 0.234 | 1.059 | 0.447 |
| | | IV | 1.017 | 0.423 | 0.933 | 0.170 | 0.618 | 0.329 | 1.218 | 0.530 | 0.690 | 0.633 | 1.172 | 0.446 |
| میانگین | | | 0.317 | 0.177 | 0.326 | 0.125 | 0.249 | 0.087 | 0.469 | 0.158 | 0.381 | 0.184 | 0.587 | 0.255 |

جدول ۳- میانگین انحرافات نتایج الگوریتم‌ها از بهترین نتایج برای دسته دوم مسائل

| | n | d | w = 0.3 | | | | w = 0.5 | | | | w = 0.7 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | GA-A | GA-B | GA-C | GA-D | GA-A | GA-B | GA-C | GA-D | GA-A | GA-B | GA-C | GA-D |
| زمان های پردازش پایین | 20 | I | 1.340 | 0.268 | 1.107 | 0.213 | 0.821 | 0.270 | 0.963 | 0.411 | 1.691 | 0.179 | 1.760 | 0.170 |
| | | II | 1.026 | 0.383 | 1.163 | 0.321 | 0.697 | 0.129 | 0.783 | 0.299 | 1.788 | 0.349 | 1.392 | 1.000 |
| | | III | 0.258 | 0.092 | 0.332 | 0.085 | 0.458 | 0.135 | 0.377 | 0.079 | 0.757 | 0.006 | 0.885 | 0.340 |
| | | IV | 0.670 | 0.824 | 0.982 | 0.578 | 1.149 | 0.094 | 1.066 | 0.442 | 0.496 | 0.282 | 1.159 | 0.236 |
| | 50 | I | 2.837 | 0.086 | 2.610 | 0.202 | 4.031 | 0.179 | 3.742 | 0.088 | 9.061 | 0.054 | 8.380 | 0.003 |
| | | II | 3.080 | 0.196 | 3.362 | 0.081 | 3.135 | 0.177 | 3.685 | 0.323 | 7.771 | 0.182 | 7.558 | 1.054 |
| | | III | 8.337 | 0.183 | 7.445 | 0.878 | 9.323 | 0.171 | 8.770 | 1.670 | 10.772 | 0.201 | 12.463 | 0.719 |
| | | IV | 9.275 | 0.952 | 9.352 | 1.107 | 7.265 | 0.779 | 9.190 | 0.728 | 6.637 | 0.760 | 9.125 | 1.064 |
| | 100 | I | 6.753 | 0.131 | 6.064 | 0.083 | 7.505 | 0.085 | 8.202 | 0.078 | 15.043 | 0.003 | 14.808 | 0.060 |
| | | II | 6.987 | 0.082 | 6.965 | 0.088 | 7.132 | 0.135 | 6.650 | 0.079 | 13.437 | 0.041 | 14.617 | 0.101 |
| | | III | 8.823 | 0.316 | 9.593 | 0.201 | 12.679 | 0.245 | 12.774 | 0.743 | 15.137 | 0.174 | 16.360 | 0.803 |
| | | IV | 13.365 | 0.557 | 11.575 | 0.450 | 12.850 | 0.291 | 11.683 | 0.963 | 12.165 | 0.375 | 15.904 | 0.334 |
| میانگین | | | 5.229 | 0.339 | 5.046 | 0.357 | 5.587 | 0.224 | 5.657 | 0.492 | 7.896 | 0.217 | 8.701 | 0.490 |
| زمان های پردازش بالا | 20 | I | 1.140 | 0.518 | 0.544 | 0.297 | 0.870 | 0.220 | 1.052 | 0.069 | 2.997 | 0.213 | 1.869 | 0.038 |
| | | II | 0.756 | 0.260 | 1.348 | 0.176 | 0.825 | 0.189 | 0.692 | 0.356 | 1.674 | 0.331 | 2.399 | 0.515 |
| | | III | 0.325 | 0.125 | 0.368 | 0.116 | 0.660 | 0.022 | 0.546 | 0.138 | 0.563 | 0.128 | 1.138 | 0.433 |
| | | IV | 3.350 | 0.531 | 1.274 | 1.096 | 1.124 | 0.461 | 0.925 | 0.625 | 1.488 | 0.392 | 2.364 | 0.590 |
| | 50 | I | 3.633 | 0.173 | 2.846 | 0.252 | 4.060 | 0.180 | 3.355 | 0.151 | 9.082 | 0.029 | 7.511 | 0.010 |
| | | II | 3.346 | 0.131 | 2.727 | 0.155 | 4.050 | 0.187 | 4.393 | 0.142 | 8.225 | 0.011 | 7.847 | 0.253 |
| | | III | 6.646 | 0.850 | 6.379 | 0.065 | 9.580 | 0.280 | 9.666 | 0.719 | 11.934 | 0.304 | 11.744 | 1.316 |
| | | IV | 11.104 | 0.137 | 5.978 | 1.140 | 9.286 | 0.979 | 7.501 | 1.003 | 10.683 | 0.630 | 9.539 | 1.901 |
| | 100 | I | 6.348 | 0.158 | 5.967 | 0.165 | 7.258 | 0.300 | 7.245 | 0.031 | 12.692 | 0.013 | 12.198 | 0.005 |
| | | II | 6.573 | 0.062 | 6.957 | 0.149 | 6.457 | 0.184 | 6.479 | 0.047 | 10.910 | 0.066 | 12.111 | 0.123 |
| | | III | 10.930 | 0.186 | 11.379 | 0.260 | 13.491 | 0.282 | 12.652 | 0.850 | 17.623 | 0.369 | 16.450 | 0.955 |
| | | IV | 10.797 | 0.318 | 10.154 | 0.112 | 9.499 | 0.733 | 8.927 | 0.716 | 11.943 | 0.408 | 12.283 | 0.168 |
| میانگین | | | 5.412 | 0.287 | 4.660 | 0.332 | 5.597 | 0.335 | 5.286 | 0.404 | 8.318 | 0.241 | 8.121 | 0.526 |

جدول ۴- میانگین زمان‌های اجرای الگوریتم‌ها

| $n$ | $w$ / GAs | 0.3 | 0.5 | 0.7 | میانگین |
|---|---|---|---|---|---|
| 5 | GA-A | 2.282 | 2.339 | 2.325 | 2.315 |
| | GA-B | 2.071 | 2.125 | 2.113 | 2.103 |
| | GA-C | 2.080 | 2.110 | 2.094 | 2.095 |
| | GA-D | 1.962 | 1.974 | 1.978 | 1.971 |
| 10 | GA-A | 8.530 | 8.707 | 8.660 | 8.632 |
| | GA-B | 5.999 | 6.944 | 6.916 | 6.620 |
| | GA-C | 7.108 | 7.267 | 7.091 | 7.155 |
| | GA-D | 5.504 | 6.199 | 6.130 | 5.944 |
| 15 | GA-A | 13.684 | 13.585 | 13.645 | 13.638 |
| | GA-B | 10.716 | 12.293 | 12.534 | 11.848 |
| | GA-C | 12.041 | 12.109 | 12.133 | 12.094 |
| | GA-D | 9.461 | 10.748 | 11.457 | 10.555 |
| 20 | GA-A | 17.927 | 17.974 | 17.934 | 17.945 |
| | GA-B | 15.003 | 16.608 | 17.618 | 16.410 |
| | GA-C | 15.987 | 15.888 | 16.021 | 15.962 |
| | GA-D | 13.266 | 15.539 | 15.861 | 14.889 |
| 50 | GA-A | 48.750 | 48.748 | 49.148 | 48.882 |
| | GA-B | 42.141 | 48.640 | 49.491 | 46.757 |
| | GA-C | 43.469 | 43.491 | 43.613 | 43.524 |
| | GA-D | 37.600 | 41.798 | 43.524 | 40.974 |
| 100 | GA-A | 103.471 | 102.686 | 102.898 | 103.018 |
| | GA-B | 86.804 | 101.090 | 104.666 | 97.520 |
| | GA-C | 91.448 | 90.723 | 91.412 | 91.194 |
| | GA-D | 74.855 | 90.634 | 92.562 | 86.017 |

## نتیجه گیری

در این تحقیق یک مسئله زمانبندی دو معیاره در محیط تک ماشین مورد بررسی قرار گرفت. معیار اول دیرکرد کل و معیار دوم واریانس زمان انتظار بود. برای این مسئله الگوریتمی ژنتیک توسعه داده شد. در این الگوریتم دو نوع تابع برازش که آن‌ها را پویا و ایستا نامیدیم برای مسئله ارائه گردید. علاوه‌بر جمعیت اولیه کاملاً تصادفی، یک جمعیت اولیه هیوریستیک نیز با توجه به معیارهای مورد نظر مسئله طراحی شد. عملگر تقاطع در این تحقیق طوری طراحی گردید که از تولید اعداد تصادفی، که باعث افزایش زمان اجرای الگوریتم‌ها می‌شوند، برای تعیین نقاط تقاطع اجتناب شود. در نهایت با توجه به دو نوع تابع برازش ارائه شده و دو نوع جمعیت اولیه مورد استفاده، چهار الگوریتم ژنتیک تعریف شد. برای سنجش کارایی این چهار الگوریتم نتایج آن‌ها روی تعداد زیادی از مسائل در دو دسته و از نظر دقت و زمان اجرا مورد ارزیابی قرار گرفت.

می‌توان از این تحقیق نتیجه گرفت که استفاده از یک تابع برازش پویا و جمعیت اولیه هیوریستیک می‌تواند کمک زیادی به بهبود الگوریتم‌های ژنتیک برای حل مسئله مورد نظر نماید. مسلماً انتخاب یکی از این الگوریتم‌ها کاملاً به شرایط تصمیم‌گیرنده بستگی دارد. با این حال بنا بر نتایج تحقیق حاضر اگر شرایط زمان کمتری به تصمیم‌گیرنده تحمیل کرد آنگاه الگوریتم GA-D گزینه بهتری خواهد بود اما اگر در شرایطی قرار داشتیم که دقت بالا مد نظر تصمیم گیرنده باشد می‌توانیم الگوریتم GA-B را به عنوان الگوریتم برتر انتخاب کنیم.

## منابع

Aryanezhad, M. B., Jabbarzadeh, A., & Zareei, A. (2009). Combination of Genetic Algorithm and LP-metric to solve single machine bi-criteria scheduling problem. Paper presented at the Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on.

Bagchi, U. (1989). Simultaneous Minimization of Mean and Variation of Flow Time and Waiting Time in Single Machine Systems. Operations Research, 37(1), 118-125.

Bagchi, U., Sullivan, R. S., & Chang, Y.-L. (1987). Minimizing Mean Squared Deviation of Completion Times about a Common Due Date. Management Science, 33(7), 894-906.

Baker, K. R., & Trietsch, D. (2009). Principles of Sequencing and Scheduling: John Wiley & Sons.

Deb, K. (2009). Multi-Objective Optimization Using Evolutionary Algorithms: John Wiley & Sons.

Du, J., & Leung, J. Y. T. (1990). Minimizing Total Tardiness on One Machine Is NP-Hard. Mathematics of Operations Research, 15(3), 483-495.

Eilon, S., & Chowdhury, I. G. (1977). Minimising Waiting Time Variance in the Single Machine Problem. Management Science, 23(6), 567-575.

Ferrolho, A., & Crisostomo, M. (2006). Genetic Algorithm for the Single Machine Total Weighted Tardiness Problem. Paper presented at the E-Learning in Industrial Electronics, 2006 1ST IEEE International Conference on.

Fry, T. D., Vicens, L., Macleod, K., & Fernandez, S. (1989). A Heuristic Solution Procedure to Minimize $\bar{T}$ on a Single Machine. The Journal of the Operational Research Society, 40(3), 293-297.

Gupta, M. C., Gupta, Y. P., & Bector, C. R. (1990). Minimizing the Flow-Time Variance in Single-Machine Systems. The Journal of the Operational Research Society, 41(8), 767-779.

Han, H. (2005). Multicriteria scheduling. European Journal of Operational Research, 167(3), 592-623.

Holsenback, J. E., & Russell, R. M. (1992). A Heuristic Algorithm for Sequencing on One Machine to Minimize Total Tardiness. The Journal of the Operational Research Society, 43(1), 53-62.

Jolai, F., Rabbani, M., Amalnick, S., Dabaghi, A., Dehghan, M., & Parast, M. Y. (2007). Genetic algorithm for bi-criteria single machine scheduling problem of minimizing maximum earliness and number of tardy jobs. Applied Mathematics and Computation, 194(2), 552-560.

Kanet, J. J. (1981). Minimizing Variation of Flow Time in Single Machine Systems. Management Science, 27(12), 1453-1459.

Köksalan, M., & Burak Keha, A. (2003). Using genetic algorithms for single-machine bicriteria scheduling problems. European Journal of Operational Research, 145(3), 543-556.

Koulamas, C. (1994). The Total Tardiness Problem: Review and Extensions. Operations Research, 42(6), 1025-1041.

Madureira, A., Ramos, C., & do Carmo Silva, S. (2001). A GA based scheduling system for dynamic single machine problem. Paper presented at the Assembly and Task Planning, 2001, Proceedings of the IEEE International Symposium on.

Merten, A. G., & Muller, M. E. (1972). Variance Minimization in Single Machine Sequencing Problems. Management Science, 18(9), 518-528.

Morton, T. E., Rachamadugu, R. V., & Vepsalainen, A. (1984). Accurate myopic heuristics for tardiness scheduling (GSIA technical report ed.): Carnegie-Mellon University.

Panneerselvam, R. (2006). Simple heuristic to minimize total tardiness in a single machine scheduling problem. The International Journal of Advanced Manufacturing Technology, 30(7), 722-726.

Pinedo, M. (2008). Scheduling: Theory, Algorithms, and Systems: Springer.

Potts, C. N., & Van Wassenhove, L. N. (1982). A decomposition algorithm for the single machine total tardiness problem. Operations Research Letters, 1(5), 177-181.

Potts, C. N., & Van Wassenhove, L. N. (1991). Single Machine Tardiness Sequencing Heuristics. IIE Transactions, 23(4), 346-354.

Srirangacharyulu, B., & Srinivasan, G. (2010). Completion time variance minimization in single machine and multi-machine systems. Computers & Operations Research, 37(1), 62-71.

Su, L.-H., & Tien, Y.-Y. (2011). Minimizing mean absolute deviation of completion time about a common due window subject to maximum tardiness for a single machine. International Journal of Production Economics, 134(1), 196-203.

Tan, K. C., Khor, E. F., & Lee, T. H. (2005). Multiobjective evolutionary algorithms and applications: Springer.

Wieslaw, K. (1993). Completion time variance minimization on a single machine is difficult. Operations Research Letters, 14(1), 49-59.

Ye, N., Li, X., Farley, T., & Xu, X. (2007). Job scheduling methods for reducing waiting time variance. Computers & Operations Research, 34(10), 3069-3083.

Yoon, S. H., & Lee, I. S. (2011). New constructive heuristics for the total weighted tardiness problem. J Oper Res Soc, 62(1), 232-237.