طراحی جدول زمان‌بندی خودکار برای دروس دانشگاهی با استفاده از الگوریتم های ژنتیک

سید امیرحسن منجمی[۱]، سولماز مسعودیان[۲]، افسانه استکی[۳] و ناصر نعمت بخش[۴]

چکیده: طراحی جدول زمان‌بندی، اساساً از وظایف پیچیده و وقت گیر برای پرسنل مسئول می‌باشد که از طرفی انجام خودکار آن گامی در جهت کاهش بار کاری پرسنل و از سوی دیگر یک نمونه مطلوب برای امتحان روش‌های برنامه‌ریزی و ارضای محدودیت‌ها در هوش مصنوعی است. در این پژوهش، ابتدا الگوریتم‌های ژنتیک مطالعه و بررسی شده، سپس در مسأله بهینه سازی جدول زمانی دروس برای یک دانشکده فرضی مورد استفاده قرار گرفته است. در این رویکرد روند تکاملی پاسخ‌ها طی تکرار نسل‌ها در یک الگوریتم ژنتیک، نهایتاً منجر به تولید یک جدول زمان‌بندی دروس خوش کیفیت خواهد گردید. در مرحله پیاده سازی، به کمک تغییراتی که در روند معمول الگوریتم‌های ژنتیک صورت داده شد، نتایج بسیار خوبی در زمینه طراحی جداول زمان‌بندی دروس دانشگاهی حاصل گردیده است. اساس کار الگوریتم طراحی شده، حفظ کروموزوم‌های بهتر جمعیت و اعمال عملگرهای ژنتیکی بر روی بقیه کروموزوم‌ها به منظور بهبود آنها می‌باشد. در آزمون‌ها، مقایسه بین الگوریتم ژنتیک عادی و الگوریتم پیشنهادی، طی چند مرحله، نقاط قوت الگوریتم پیشنهادی را مشخص کرد. ایده‌های مطرح شده در این تحقیق قابل تسری به کاربردهای مشابه نیز خواهد بود.

کلمات کلیدی: جداول زمان‌بندی، برنامه‌ریزی، الگوریتم‌های ژنتیک، بهینه سازی، ارضای محدودیت‌ها

۱– مقدمه

مسأله جدول‌بندی زمانی دروس در اصل، شامل تخصیص دروس هفتگی به بازه‌های زمانی و اتاق‌های برگزاری کلاس‌ها است. اگر شروطی مانند عدم تخصیص یک کلاس در یک ساعت به چند درس، یا عدم تلاقی زمانی در دروس یک استاد را به مسأله اضافه کنیم، مشخص می‌شود که مسأله طراحی جداول زمان‌بندی، می‌تواند یک مسأله ارضای محدودیت (CSP)[۱] قلمداد شود. با توجه به تعداد روزافزون دانشجویان، رشته‌های جدید، کمبود کلاس‌ها، اتاق‌های کنفرانس و آزمایشگاه و تعداد رو به افزایش درس‌های ارائه شده برای دانشجویان، با محدودیت‌های بسیاری برای ساخت یک جدول مناسب مواجه هستیم و طراحی جداولی که بتواند به طور گسترده به وسیله استادان و دانشجویان مورد قبول واقع شود بسیار مشکل است [۱]. شاید بتوانیم چنین جدولی را یک جدول *خوش کیفیت* بنامیم.

روش‌های متعددی برای حل این نوع مسائل به کار گرفته شده که برخی از آنها عبارتند از [۱]:

- الگوریتم رنگ‌آمیزی گراف[۲]
- استفاده از توابع هیوریستیک[۳] یا توابع تجربی
- روش‌های جمعیت‌گرا یا تکاملی

در این روش‌ها عموماً از ایده‌های تکاملی و بهبود جمعیتی استفاده می‌شود. انواع معمول این روش‌ها در کاربرد طراحی جدول زمان‌بندی عبارتند از [۲]:

- الگوریتم‌های ژنتیکی
- الگوریتم‌های کلنی مورچه‌ها
- الگوریتم‌های ممتیک[۴]

الگوریتم مورد استفاده در پروژه حاضر، الگوریتم ژنتیک بوده که یکی از قوی‌ترین و پرکاربردترین الگوریتم‌ها در مسائل جستجو و بهینه سازی است [۳و۴].

تاریخ دریافت مقاله ۸۸/۸/۱۴ ، تاریخ تصویب نهایی ۸۸/۱۰/۲۷

[۱] **استادیار،** گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه اصفهان (نویسنده مسئول)، پست الکترونیکی: monadjemi@eng.ui.ac.ir

[۲] **دانشجوی کارشناسی،** گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه اصفهان

[۳] **دانشجوی کارشناسی،** گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه اصفهان

[۴] **استادیار،** گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه اصفهان

یکی از دلایل محبوبیت الگوریتم‌های ژنتیکی عدم نیاز به مدل ریاضی سطح بالا و پیشرفته می‌باشد. این الگوریتم‌ها برای اولین بار توسط جان هلند[۵] و بر پایه نظریه تکامل طبیعی داروین پیشنهاد گردیدند [۵و۶]. الگوریتم‌های ژنتیکی از قانون تکامل پیروی می‌کنند. برای شروع این الگوریتم‌ها نیاز به یک نسل اولیه تصادفی می‌باشد که مجموعه‌ای معمولاً تصادفی از جواب‌های مسأله است. سپس این جواب‌ها تا رسیدن به یک سطح بهینه عمومی تکامل پیدا می‌کنند. عمل تکامل توسط آمیزش کروموزوم‌ها و عمل جهش بر روی آنها انجام می‌شود و کروموزوم‌هایی که دارای برازندگی[۶] بیشتری هستند شانس بیشتری برای انتقال به نسل‌های بعد (تجدید نسل) را دارند [۵و۶]. در مسأله مورد بحث ما کروموزوم ها در اصل همان جداول زمانی می‌باشند. در این تحقیق سعی شده که با محور قرار دادن ایده اصلی الگوریتم‌های ژنتیکی و با توجه به خصوصیات مسأله طراحی جداول زمان‌بندی، با تغییراتی در این الگوریتم‌ها بازدهی آنها را در حل مسأله طراحی جداول زمان‌بندی افزایش دهیم و نتایج عملی را نیز به صورت مقایسه‌ای بیازماییم. سازماندهی مقاله حاضر بدین ترتیب است که در بخش ۲ به کلیات الگوریتم‌های ژنتیک پرداخته‌ایم. بخش ۳ به مرور کارهای قبلی و بخش ۴ به تشریح الگوریتم پیشنهادی اختصاص یافته است. در بخش ۵ به بیان آزمون‌ها و نتایج آنها پرداخته و نهایتاً مقاله در بخش ۶ جمع‌بندی گردیده است.

از کارهای انجام شده در این زمینه می‌توان به موارد متعدد اشاره نمود. پیچر و رانکین[۷] روشی مبتنی بر الگوریتم های ممتیک همراه با یک جستجوی محلی موثر ارائه دادند که بر روی داده‌های اولین مسابقه جدول‌بندی زمانی در سال ۱۹۹۴ اعمال شد [۷]. بورک، الیمن و ویر[۸] در دانشکدهٔ علوم کامپیوتر دانشگاه ناتینگهام، الگوریتم ژنتیکی را پیشنهاد کرده‌اند که تنها روی بازه های زمانی قابل قبول اعمال می شود و به طور تعاملی با کاربر ارتباط دارد [۲]. اربن و کپلر[۹] نیز جداولی بهینه برای دروس هفتگی را به کمک الگوریتم های ژنتیکی طراحی و پیاده سازی نموده‌اند [۸]. کارتر[۱۰] در روش پیشنهادی خود مسأله را به بخش‌هایی تقسیم کرده و از یک الگوریتم حریصانه برای تخصیص بازه‌های زمانی و از یک الگوریتم مبتنی بر تابع لاگرانژ، برای تخصیص کلاس های درس استفاده می‌کند [۱].

با توجه به پر محاسبه بودن الگوریتم‌های ژنتیکی، تلاش‌هایی در جهت افزایش سرعت آنها نیز صورت گرفته است. در این راستا آبرامسون و ابلا در تحقیق خود از الگوریتم‌های ژنتیک موازی در حل مسأله طراحی جداول زمان‌بندی برای مدارس سود برده‌اند و مزیت زمانی نسبتاً بالایی را نسبت به الگوریتم‌های ژنتیک عادی گزارش کرده‌اند [۹]. از بین کارهای متاخر، عبدالله و همکاران روش‌های ترکیبی و هایبرید را برای حل این مسأله پیشنهاد کرده و آزموده‌اند [۱۰ . وراک و همکاران نیز مروری کامل بر کاربردهای الگوریتم ژنتیک را در طراحی جداول زمان‌بندی در تحقیق خود ارائه نموده‌اند [۱۱]. تحقیق کوپر و کینگستون و گزارش فنی ایشان به بیان پیچیدگی مسأله ساخت جداول زمان‌بندی می‌پردازد و آن را تحلیل می‌کند [۱۲].

همچنین در بلیگیانیس و همکاران برای طراحی جداول زمان‌بندی دبیرستان‌ها در کشور یونان، الگوریتمی وفقی بر مبنای محاسبات تکاملی پیشنهاد و پیاده سازی کرده‌اند [۱۳]. نتایج عملی کار ایشان روی تعداد زیادی از دبیرستان‌های یونان و در مقایسه با چند الگوریتم دیگر مطلوب گزارش شده است. به طور مشابه، پیلای و بن ژاف نیز در تحقیق خود از یک الگوریتم ژنتیک مطلع و هوشمند برای طراحی جداول زمان‌بندی امتحانات سود برده‌اند. دیدگاه آنان یک دیدگاه دو مرحله‌ای بوده است که در مرحله اول محدودیت‌های سخت و در مرحله دوم محدودیت‌های نرم به چالش کشیده می‌شوند. در این تحقیق به طور خاص مزایای الگوریتم ژنتیک آشکار شده است و نتایج امیدوار کننده‌ای از آن گزارش شده است [۱۴]. برای اطلاعات مفصل‌تر، خواننده علاقمند می‌تواند به تحقیق مروری ویلکه و اوستلر در سال ۲۰۰۸ که در آن چهار الگوریتم جستجوی تابو، آبدیده سازی فلزات، ژنتیک، و B&B باهم مقایسه شده‌اند- مراجعه نماید [۱۵].

۲- کلیات الگوریتم های ژنتیک

تجربیات دهه‌های اخیر نشان دادند که الگوریتم‌های ژنتیک یکی از قوی‌ترین روش‌های برگرفته از طبیعت است که با الهام از علم ژنتیک و پدیده انتخاب طبیعی، یکی از بهترین اشکال بهینه‌سازی عددی در مسائل علوم و مهندسی را ارائه می‌کند. در این الگوریتم‌ها با جستجوی تصادفی- و البته هوشمندانه و هدفمند- مناسب‌ترین جواب‌ها از میان اطلاعات کد شده (همان کروموزوم‌ها) به دست خواهد آمد. اجزای الگوریتم‌های ژنتیکی عبارتند از [۶]:

- کروموزوم و ژن
- جمعیت ژنتیکی
- تابع برازش
- عملگرهای ژنتیکی
- پارامترها

در جدول ۱ مقایسه فشرده‌ای بین ژنتیک در سیستم‌های طبیعی و الگوریتم‌های ژنتیک نشان داده شده است [۵].

در یک الگوریتم ژنتیک جمعیت ژنتیکی (جامعه) همان مجموعه کروموزوم‌ها، عملگرهای ژنتیکی همان تزویج از طریق ادغام و جهش، و پارامترها کمیت‌هایی مانند اندازه جامعه، نرخ جهش و مانند آن هستند. سایر اجزا در جدول ۱ توضیح داده شده‌اند.

۳- روش تحقیق

۳-۱ مسأله جدول بندی زمانی دروس در یک نگاه

در این قسمت و با فرض سود بردن از یک الگوریتم ژنتیک استاندارد در طراحی جداول زمان‌بندی به بیان فرضیات مسأله می‌پردازیم.

- ورودی‌ها:

- لیست دروس اصلی و آزمایشگاهی دانشکده
- لیست کلاس‌های درس و آزمایشگاه‌های موجود در دانشکده
- لیست اساتید دانشکده

- خروجی: جدول بهینه برنامه ریزی دروس

- پارامترها:

- اندازه جمعیت ژنتیکی: ۱۰۰۰
- طول کروموزوم (تعداد خانه‌های جدول):

برای محاسبه طول کرومزوم از رابطه زیر استفاده می‌شود:

$$L_c = H \times (C + A) \qquad (۱)$$

که در اینجا H تعداد کل بازه‌های زمانی یک ساعتی در یک هفته، C تعداد کلاس‌ها، و A تعداد آزمایشگاه‌ها می‌باشد.

جدول ۱ مقایسه الگوریتم ژنتیک با سیستم‌های طبیعی

سیستم‌های طبیعی	الگوریتم ژنتیک
کروموزوم : بسته‌های ژنی هستند که اطلاعات وراثتی را از نسلی به نسل دیگر انتقال می دهند.	کروموزوم : پاسخ‌های ممکن مسأله که به صورت رشته‌های عددی رمزگذاری شده‌اند.
محیط : شرایط محیطی که جمعیت در آن قرار دارد و دیکته کننده نحوه تحول است.	تابع برازش: محک کیفیت یک کروموزوم که به صورت یک رابطهٔ ریاضی در آمده که آن را تابع برازش می‌نامند.
اصل انتخاب طبیعی : معیار بقای موجود زنده و تکثیر آن، سازش با محیط است.	تکثیر : هر رشته جمعیت را به عنوان متغیر تابع برازش در نظر گرفته و مقدار تابع برازش هر رشته محاسبه می‌شود. متناسب با مقدار تابع برازش، رشته‌های والدین برای تولید جمعیت جدید انتخاب می‌شود.
تقاطع: در نتیجه تقاطع یا تبادل قسمتی از کروموزوم‌ها، مبادله ژن‌های پیوسته صورت می‌گیرد.	ادغام : رشته‌های جمعیت به صورت دو به دو مزدوج می‌شوند. زوج رشته‌ها از یک نقطه قطع می‌شوند. نیم بخش‌های بین دو رشته تعویض می‌شوند.
جهش: جانشین شدن ژنی به جای ژن دیگر در طول زنجیره DNA یا تغییرات ایجاد شده در ژن. گاهی قسمتی از یک ژن جانشین ژن دیگری می‌شود.	جهش : یک بیت از رشته عددی به صورت تصادفی انتخاب می‌شود و دچار تغییر می‌گردد.
تجدید نسل: ایجاد نسل‌های جدید و تکامل موجودات	تجدید نسل: تکرار مراحل الگوریتم بعد از مرحلهٔ تکثیرتا حصول پاسخ بهینه یا رسیدن به حد توقف

- حداکثر تعداد نسل‌ها:

$$G_m = \frac{500000}{P} \qquad (۲)$$

که در آن p اندازه جمعیت یا تعداد کرومزم‌ها می باشد.

- احتمال عملگر ادغام : ۸۰٪ ۲۵

- احتما ل عملگر جهش : این احتمال بین دو مقدار ۲۰٪ و ۵۰٪ بسته به شرایط الگوریتم متغیر می‌باشد. این پارامترهای مرزی با مقداری سعی و خطا به دست آمده است.

۳-۱-۱ خصوصیات

- برای کدگذاری کروموزوم‌ها از روش جای‌گشتی استفاده شده است. در کدگذاری جای‌گشتی اعداد صحیح نمایانگر ژن‌ها در کروموزوم است. به تعداد ژن‌ها عدد غیر تکراری در کروموزوم داریم و عمل ادغام باید به گونه‌ای باشد که پس از ترکیب دو کروموزوم ژن تکراری در فرزند ایجاد نشود [۶].

- نوع عملگر انتخاب تورنمنت با اندازه ۳ است که برای نیمه بدتر جمعیت به کار می رود. نیمه بهتر مستقیماً از نسل قبل کپی می‌شود. در تورنمنت با اندازه k، تعداد k کروموزوم به طور تصادفی انتخاب شده و یکی از آنها بر اساس میزان برازش به پیروزی می‌رسد [۶].

- عملگر ادغام از نوع پاره- نگاشته[11] می‌باشد. ادغام پاره- نگاشته در حقیقت همان عملگر ادغام دونقطه ای است که برای حالت جایگشتی خاص شده است. در این روش دو عدد به صورت تصادفی به عنوان نقاط برش انتخاب شده، سپس قسمت مابین دو نقطه برش در دو کروموزوم تعویض شده و آن گاه قسمت‌های دو طرف طوری مقدار دهی می‌شوند که در هیچ کدام از دو کروموزوم، تکراری صورت نگیرد [۵].

- نوع عملگر جهش : تعویض مقدار دو یا چند موقعیت از جدول (کروموزم)

۳-۱-۲ تابع برازش

رابطه (۳) تابع برازش به کار رفته در این تحقیق را نشان می‌دهد.

$$Fitness = \frac{K}{0.8\times\sum H_i \times hard_i + 0.2\times\sum S_i \times soft_i + 1} \qquad (۳)$$

که در این رابطه K ضریب ثابت است و در آزمون‌های ما ۱۰۰۰ فرض شده است، Hi وزن نسبت داده شده به محدودیت سخت i ام، $hard_i$ تعداد دفعات نقض محدودیت سخت i ام، S_i وزن نسبت داده شده به محدودیت نرم i ام و $soft_i$ تعداد دفعات نقض محدودیت نرم i ام می‌باشد. بدین ترتیب تابع برازش مقداری بین ۱ تا ۱۰۰۰ خواهد داشت.

۳-۱-۳ شروط خاتمه الگوریتم

۱- رسیدن به برازش ۱۰۰۰ (جدولی که هیچ نقض محدودیت سخت یا نرمی ندارد)

۲- تکرار تعداد مشخص نسل (حداکثر تعداد نسل‌ها)

۳- همگرا شدن جمعیت (در الگوریتم طراحی شده احتمال وقوع این شرط در جمعیت های بزرگ نزدیک به صفر است).

الگوریتم ژنتیکی روش بهینه سازی کارآمدی است که خصایص مثبت روش‌های تصادفی و حریصانه را تا حد زیادی در خود جمع کرده است. لیکن همواره می‌توان با توجه به خصوصیات مسأله، تغییراتی را در یک الگوریتم ژنتیک استاندارد پیشنهاد نمود. در بخش‌های بعدی به ویژگی‌های خاص الگوریتم ژنتیک پیشنهادی در این تحقیق می‌پردازیم و جزئیات محقق سازی یک مسأله طراحی جداول زمان‌بندی روی این الگوریتم را تشریح خواهیم کرد.

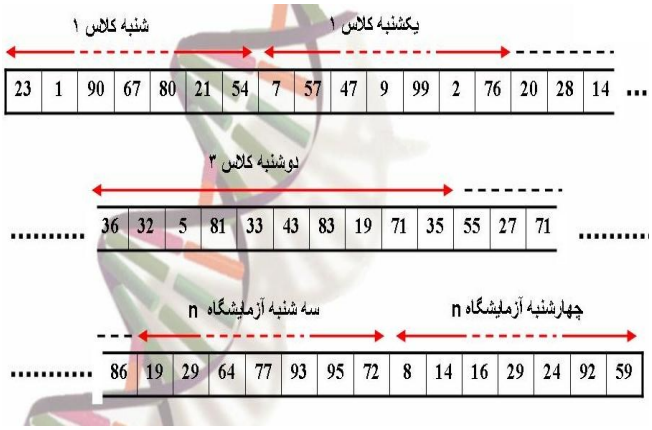
۳-۲ نمایش کروموزوم‌ها

در این تحقیق هر کرومزوم معادل یک جدول زمانی فرض شده است. در نمایش دو بعدی ستون‌های این جدول بازه‌های زمانی و سطرهای آن مکان‌ها (کلاس‌ها) را مشخص می‌کنند. شکل‌های ۱ و ۲ نمایش یک بعدی و دو بعدی این کروموزوم‌ها (یا جدول‌ها) را نشان می‌دهد.

همان طور که پیشتر ذکر شد روش کدگذاری جای‌گشتی می‌باشد. در روش کدگذاری جای‌گشتی کروموزوم یک آرایه

n عضوی صحیح است که اعداد ۱ تا n به عناصر مختلف آن منسوب می‌شوند. چون معمولاً یک کلاس در تمام ساعات هفته اشغال نیست، تعدادی از خانه‌های جدول خالی می‌ماند که با صفر پر می‌شوند.

در ساخت این جدول، ابتدا کلاس‌های درس و سپس آزمایشگاه‌ها درج می‌شوند. هدف از این کار این است که نقض محدودیت‌های سخت را کم کنیم چون یک کلاس درس نمی‌تواند در آزمایشگاه تشکیل شود و بر عکس. توابع ادغام و جهش ما نیز برای هربخش به طور جداگانه عمل خواهند کرد. در کد نوشته شده دو پارامتر ورودی به توابع جهش و ادغام اضافه شد که ابتدا و انتهای بازه‌ای که باید ادغام یا جهش روی آن صورت گیرد را مشخص می‌کند. یعنی هرگاه قرار است ادغام یا جهش روی کروموزومی صورت گیرد، این عمل با توجه به این بازه مجاز انجام خواهند شد. در حقیقت در روش به کار گرفته شده، هرگز یک کلاس درسی در یک آزمایشگاه و یا یک آزمایشگاه در یک کلاس درسی واقع نخواهد شد. به همین دلیل است که مقدار دهی اولیه را شبه تصادفی، و نه کاملاً تصادفی در نظر می‌گیریم. رخدادها در مکان‌ها اتفاق می‌افتند. (کلاس‌های درس و آزمایشگاه‌ها) و کروموزوم‌ها همان جداول زمانی هستند که هر یک دارای برازش مربوط به خود می‌باشند. نکته قابل توجه این است که نوع مکان‌ها و رخدادها باید همیشه یکسان باشد و اندازه مکان‌ها نباید کوچک‌تر از تعداد دانشجویان درس مربوطه در آن کلاس باشد. در روش پیشنهادی بازه‌های زمانی را تک ساعتی در نظر گرفته‌ایم.



شکل ۱ نمایش واقعی کروموزوم/جدول

	چهارشنبه 18-19	چهارشنبه 17-18	.....	دوشنبه 13-14	.....	شنبه 9-10	شنبه 8-9
کلاس ۱	89	8	.....	56	.....	24	91
کلاس ۲	69	39	.....	53	.....	12	76
.....	75	86	.....	28	.....	59	29
کلاس n	5	33	.....	15	.....	42	18
آزمایشگاه ۱	49	72	.....	96	.....	9	89
.....	86	57	.....	78	.....	30	41
آزمایشگاه n	80	10	.....	71	.....	60	79

شکل ۲ تصور ذهنی کروموزوم/جدول

۳-۳ اندازه جمعیت

چنان چه اندازه جمعیت خیلی بزرگ باشد، سرعت اجرای الگوریتم به شدت کاهش خواهد یافت. از طرفی اگر اندازه جمعیت کوچک باشد، مسأله خیلی زود به جوابی که لزوماً بهینه نیست همگرا خواهد شد. راه حل پیشنهادی این است که اندازه جمعیت تابعی از تعداد رخدادها و طول کروموزوم‌ها باشد. یعنی هر چقدر تعداد رخدادها بیشتر باشد جمعیت را بزرگتر، و هر چه طول کروموزوم‌ها بیشتر باشد جمعیت را کوچکتر در نظر می‌گیریم.

$$S_p = k \times \frac{m \times n}{(E-1)} \qquad (۴)$$

که در این رابطه Sp سایز جمعیت، m مجموع تعداد کلاس‌ها و آزمایشگاه‌ها، n برابر تعداد کل بازه‌های زمانی یک ساعتی در یک هفته، E برابر با تعداد وقایع و k یک ضریب ثابت می‌باشد. در آزمون‌ها به کمک سعی و خطا مقدار k=50 مناسب تشخیص داده شد.

پس از اجراهای متفاوت و تست نتایج با مقادیر مختلف به این نتیجه رسیدیم که k=50 مقدار مناسبی است.

۳-۴ مقدار اولیه[۱۲]

برای مقداردهی اولیه یک جمعیت، از راه حل‌های ممکن به طور تصادفی و مستقل از برازش استفاده می‌شود. جداول جمعیت اولیه عموماً از برازش بسیار پایین رنج می‌برند؛ زیرا در مرحله مقدار دهی اولیه توجهی به محدودیت‌های نرم و

تعدادی از محدودیت‌های سخت نداریم. علت این است که اگر بخواهیم محدودیت‌های نرم و تمامی محدودیت‌های سخت را اعمال کنیم، تولید جمعیت اولیه کاری بسیار دشوار و تقریباً غیرعملی خواهد شد. در حقیقت مسأله جدول‌بندی زمانی به عنوان یک مسأله چند جمله‌ای غیرقطعی سخت[13] شناخته شده است. مسأله چندجمله‌ای غیر قطعی، نوعی مسأله تصمیم‌گیری است که توسط الگوریتم‌های غیر قطعی زمانی چند جمله‌ای قابل حل است [۱۲و۱۶]. شروطی که در مرحله مقداردهی اولیه در نظر گرفته شدند عبارتند از:

الف: دروس اصلی همه در کلاس‌های درس و دروس آزمایشگاهی همه در آزمایشگاه‌ها قرار داده شوند.

ب: هر سه ساعت دروس آزمایشگاهی به طور پیوسته و در یک روز قرار داده شوند.

ج: دروس سه واحدی به صورت یک تک ساعت و یک دو ساعت پیوسته در نظر گرفته شوند.

د: ظرفیت کلاس‌ها و آزمایشگاه‌ها در نظر گرفته شود.

بنابراین این نتیجه حاصل شد که جایگذاری به صورت تصادفی- خطی دارای مزیت است. دلایل این انتخاب و مزایای نسبی این شیوه در قیاس با الگوریتم‌های ژنتیکی استاندارد در ادامه مقاله بحث خواهد شد. در روش خطی انتخاب خانه‌ها کاملاً به صورت تصادفی نیست، بلکه هرگاه خانه‌ای برای جایگذاری و قابل قبول نبود به جای انتخاب خانه بعدی به صورت تصادفی، به ترتیب خانه‌های جدول را پیمایش کنیم تا به یک خانه قابل قبول برسیم. همچنین در روش اتخاذ شده، قبل از هر جایگذاری دروس آزمایشگاه، امتحان می‌شود که آیا فضای کافی برای کل درس آزمایشگاهی در همسایگی خانه انتخابی وجود دارد یا خیر. ضمناً چون کامپیوتر جدول زمانی را به صورت خطی می‌بیند، باید به خانه‌هایی که در موقع تبدیل به یک جدول دوبعدی از هم جدا می‌شوند توجه کرد. علاوه بر این، در مرحله مقداردهی اولیه، وقتی یک عدد تصادفی به عنوان مکان یک درس انتخاب می‌شود دقت می‌شود که اگر درس اصلی باشد، عدد تصادفی در محدوده کلاس‌های درسی و اگر درس آزمایشگاه باشد، عدد تصادفی در محدوده آزمایشگاه‌ها انتخاب گردد.

۳-۵ ادغام[14] یا جابه‌جایی ضربدری

در الگوریتم پیشنهادی، عمل ادغام نیز تنها روی نیمه بدتر جمعیت (نیمه دارای برازش‌های کمتر) انجام می‌شود. در هر ادغام یک کروموزوم فرزند تولید می‌شود و در نهایت این کروموزوم‌ها به جای نیمه بدتر جمعیت قرار می‌گیرند، در نتیجه اندازه جمعیت ثابت می ماند.

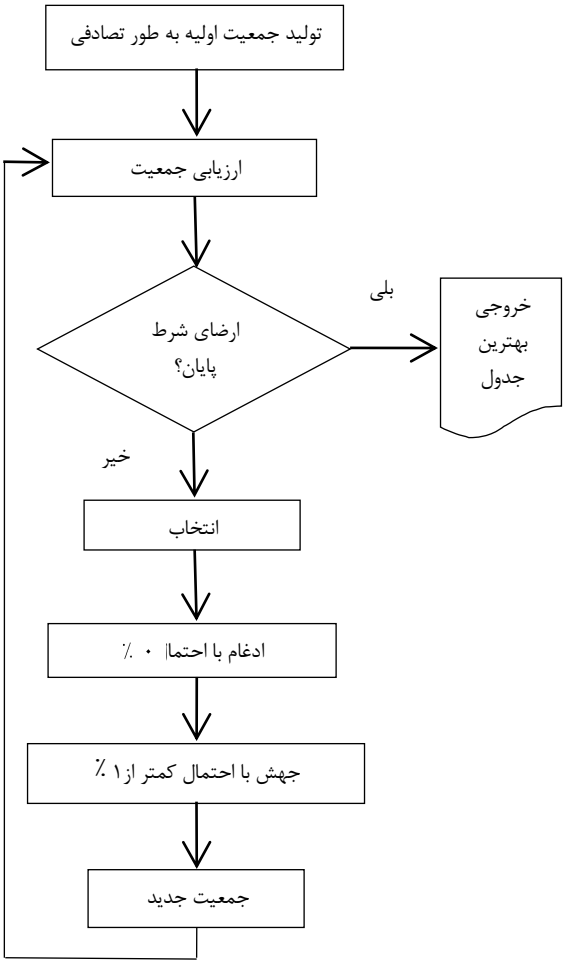
در عمل ادغام، مرز بین کروموزوم‌های مربوط به کلاس‌ها و آزمایشگاه‌ها در نظر گرفته شده تا از تولید جداول بی معنی اجتناب شود. نرخ ادغام بهینه روی ۰/۹ تنظیم گردیده است.

۳-۶ جهش

همان گونه که گفته شد وظیفه اپراتور جهش جلوگیری از یکدستی جامعه و افتادن الگوریتم بدام نقاط بهینه محلی است. در روش پیشنهادی جهت پیاده سازی اپراتور جهش در این تحقیق، سه ایده ابتکاری مورد استفاده قرار گرفت:

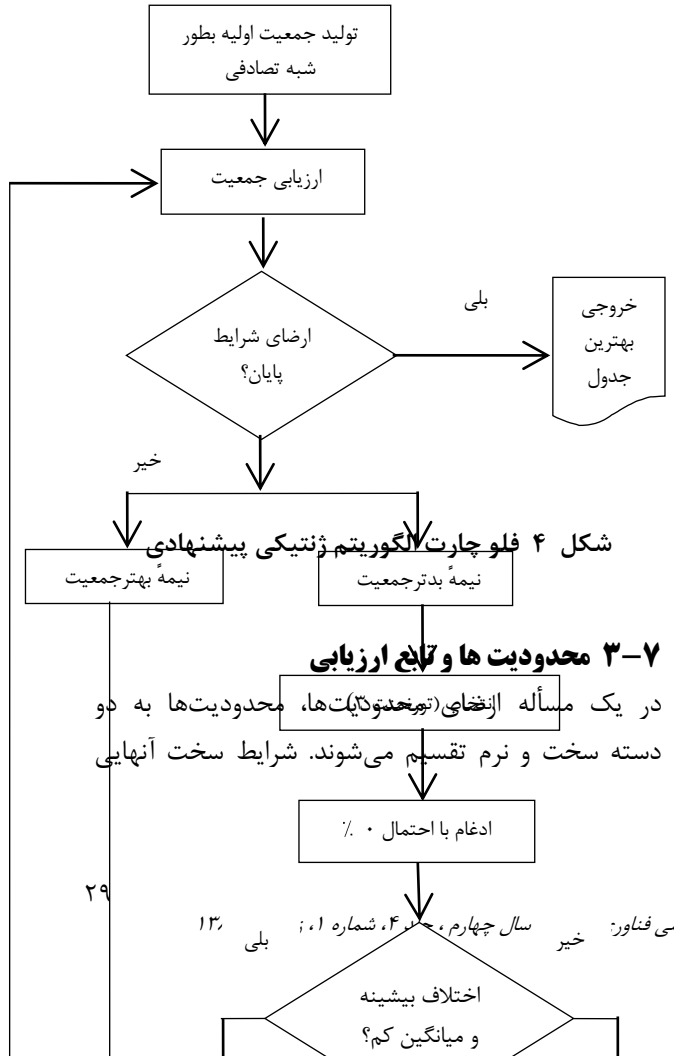
- جهش از طریق جابه‌جایی تعدادی از ژن‌ها (دروس) در زیر بازه مجاز خود (کلاس یا آزمایشگاه) محقق می گردد تا از تولید کروموزومهای نامعتبر اجتناب شود.
- چون جامعه با تکرار نسل‌ها به سمت یکنواختی می‌رود، نرخ جهش برای جلوگیری از یکنواختی تدریجاً افزایش می‌یابد.
- فضاهای خالی جدول جابه‌جا نمی‌شوند.

در این تحقیق نرخ جهش معمولاً از ۰/۲ شروع و به ۰/۵ ختم می‌شود که این مقادیر به کمک سعی و خطا به دست آمدند. فلوچارت‌های مربوط به روند یک الگوریتم ژنتیک نوعی و الگوریتم ژنتیکی طراحی شده به ترتیب در شکل‌های ۳ و ۴ آمده است.



شکل ۳ فلو چارت الگوریتم ژنتیکی عادی و استاندارد

اگر نقض تمام محدودیت‌های سخت و نرم به صفر برسد، مفهوم این است که جدولی بهینه به دست آورده‌ایم و الگوریتم متوقف می‌شود. یکی از شروط خاتمه معمول در الگوریتم‌های ژنتیک این است که اگر مقادیر برازش جداول به یک مقدار یکسان همگرا شود، الگوریتم متوقف گردد. البته در این روش خطر به دام افتادن در یک ماکزیمم محلی وجود دارد. برای اجتناب از این مشکل در روش پیشنهادی به محض کاهش اختلاف مابین برازش‌ها که نشانه یکنواختی جامعه است، نرخ جهش افزایش می‌یابد. بدین ترتیب شرط توقف الگوریتم در رسیدن به جدول بهینه یا تکرار تعداد مشخصی از نسل‌ها محدود می‌شود.



شکل ۴ فلو چارت الگوریتم ژنتیکی پیشنهادی

۳-۷ محدودیت ها و توابع ارزیابی

در یک مسأله زمان‌بندی، محدودیت‌ها به دو دسته سخت و نرم تقسیم می‌شوند. شرایط سخت آنهایی

هستند که در صورت عدم برآورده شدن آنها، پاسخ (در اینجا جدول) قابل قبول نخواهد بود. محدودیت‌های نرم نیز مواردی هستند که رعایت آنها موجب بهتر شدن پاسخ خواهد شد [۵]. تابع ارزیابی مورد استفاده جهت محاسبه برازش پاسخ‌ها، شامل دو بخش است. بخش اول جریمه مربوط به نقض محدودیت‌های سخت و بخش دوم جریمه مربوط به نقض محدودیت‌های نرم را محاسبه می‌کند. وزن جریمه محدودیت‌های سخت بسیار بالاتر انتخاب شده است (رابطه ۳).

محدودیت‌های سخت به شرح زیر در نظر گرفته شدند:

الف) تداخل ساعات یک استاد: اگر به دو درس مختلف در یک زمان، یک استاد اختصاص داده شده باشد جریمه‌ای نسبتاً بالا برای آن در نظر می‌گیریم.

ب) تداخل دروس دانشجویان یک ورودی خاص: از آنجا که دروس ارائه شده برای دانشجویان یک ورودی خاص نباید دارای تداخل باشند، برای این محدودیت نیز جریمه ای نسبتاً بالا در نظر گرفته‌ایم.

پ) مکان نامناسب برای کلاس‌ها و یا آزمایشگاه‌ها: یک کلاس درس نمی‌تواند در یک آزمایشگاه تشکیل شود و یا بر عکس، هر چند با تمهیدات به کار رفته احتمال آن ناچیز است.

ث) عدم اتصال ساعات دروس: چون هر سه ساعت دروس آزمایشگاهی باید پشت سرهم ارائه شود، همچنین برای درس‌های سه واحدی، دو ساعت از سه ساعت آنها باید پیوسته باشد، در صورت عدم اتصال این ساعات، جریمه‌ای خواهیم داشت. ضمناً دروسی نیز وجود دارند که تنها دو ساعت در هفته هستند که این دو ساعت نیز باید پیوسته باشد.

ج) در نظر گرفتن امکانات لازم برای دروس آزمایشگاهی: هر درس آزمایشگاهی باید در آزمایشگاه خاص خودش برگزار شود.

محدودیت‌های نرم نیز به شرح زیر در نظر گرفته شدند:

الف) ارائه دروس سه واحدی غیر آزمایشگاهی در ۳ ساعت متوالی: بهتر است دروس مهم و تخصصی در سه ساعت متوالی ارائه نشود؛ زیرا موجب کاهش کیفیت آموزش می‌گردد.

ب) ارائه دروس سه واحدی غیر آزمایشگاهی در یک روز: بهتر است دروس تخصصی و مهم علاوه بر رعایت شرط قبلی، در یک روز نیز ارائه نشود و ترجیحاً در دو روز ارائه گردد.

پ) ارائه دروس سه واحدی غیر آزمایشگاهی در دو روز پشت سرهم: ترجیحاً باید ساعات دروس سه واحدی در طول هفته پخش شده باشند (شامل یک تک ساعت و یک دو ساعت).

ت)عدم فاصله بین دروس یک گروه در یک روز: بهتر است در برنامه دانشجویان بین ساعات برگزاری کلاس‌های آنها ساعات خالی وجود نداشته باشد.

ث) ترجیحات اساتید: هر استاد ترجیحات خود را به این صورت که صبح یا بعد از ظهر روزهای خاصی را می‌تواند در دانشکده حضور داشته باشد اعلام می‌کند.

ج) ظرفیت مکان‌ها: اندازه مکان‌ها نباید کمتر از تعداد دانشجویان درسی باشد که در این مکان تشکیل می‌شود.

در نهایت جریمه کلی شامل مجموع وزن جریمه‌های سخت و نرم می‌باشد که جریمه‌های سخت، وزن بسیار بیشتری نسبت به جریمه‌های نرم دارند. نسبت معکوس بین جریمه‌ها و تابع ارزیابی در رابطه ۳ مشخص شده است.

۴- آزمون‌ها و نتایج

۴-۱ نتایج حاصل از اجرای الگوریتم ژنتیک عادی

همان طور که قبلاً ذکر شد در الگوریتم ژنتیکی پیشنهادی، مقدار دهی اولیه به صورت شبه تصادفی- خطی است و نرخ ادغام و جهش بیش از نرخ ادغام و جهش در الگوریتم‌های ژنتیک معمولی در نظر گرفته شده‌اند. ضمناً نرخ جهش بسته به شرایط متغیر می‌باشد. همچنین به منظور از دست ندادن کروموزوم‌های بهتر، نیمه بهتر جمعیت به صورت مستقیم در نسل بعد کپی می‌شود، یعنی روی نیمه بهتر عمل ادغام صورت نمی‌گیرد، همچنین اگر مقدار برازش یک کروموزوم کمتر از برازش میانگین نباشد هیچ عمل جهشی روی آن انجام نمی‌شود.

به منظور مقایسه بین الگوریتم پیشنهادی و یک الگوریتم ژنتیک معمولی، آزمون‌هایی انجام شدند که نتایج حاصل از آنها در زیر تشریح و در جدول ۴ نیز خلاصه شده‌اند. در

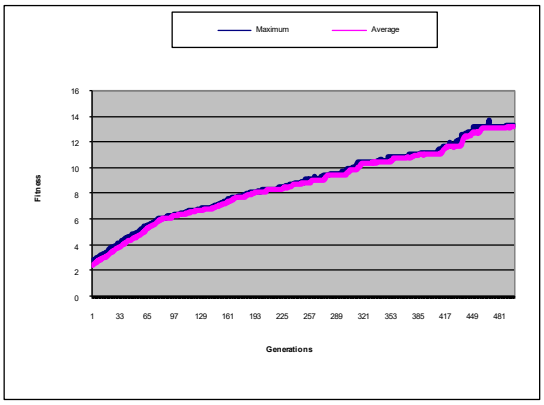
نمودارهای زیر محور افقی تعداد تکرار نسل و محور عمودی مقدار تابع برازش را نشان می دهد. همچنین خطوط تیره (مشکی) ماکزیمم برازش حاصل از تابع ارزیابی و خطوط روشن‌تر (خاکستری) میانگین برازش جامعه را مشخص می‌کنند. در کلیه تست ها امکانات برابر و دانشکده دارای ۴ کلاس، ۳ آزمایشگاه و ۱ سایت کامپیوتر فرض شده است. همچنین مشخصات دروس و تعداد ثبت نام مشابه بوده، از مجموعه دروس دانشکده مهندسی دانشگاه اصفهان در یک ترم تحصیلی استخراج شده‌اند. *آزمون‌ها روی برنامه درسی پنج ترم متوالی دو گروه آموزشی که از کلاس‌ها و آزمایشگاه‌های مشترکی استفاده می‌کنند متمرکز شده است و نتایج و منحنی‌های ارائه شده، حاصل میانگین امتیازات جداول این پنج ترم می‌باشد.* همچنین عمل ادغام و جهش با احتمال‌های مستقل خود روی تمام کروموزوم‌ها انجام می‌شوند.

۱-۱-۴ اجرای الگوریتم ژنتیک عادی (آزمون ۱)

- مقدار دهی اولیه: کاملاً تصادفی
- نرخ ادغام: ٪۶۰
- نرخ جهش: ٪۵
- عمل ادغام و جهش بسته به احتمال ادغام و جهش روی تمام کروموزوم‌ها انجام می شود.

همان طور که در شکل ۵ مشخص است جداول اولیه دارای برازش بسیار پایینی بوده و حتی پس از گذشت ۵۰۰ نسل برازش بیشینه به مقدار ۱۳/۲۶ رسیده است، که در مقایسه با روش پیشنهادی حتی از مقدار برازش بیشینه در نسل اول این روش نیز کمتر است. باید دقت شود که بر خلاف روش پیشنهادی، نمودار برازش بیشینه کاملاً صعودی نیست؛ زیرا به علت کپی نکردن مستقیم کروموزوم‌های بهتر، تضمینی برای از دست ندادن کروموزوم بهینه در هر نسل وجود ندارد. *در ادامه خواهیم دید که تغییر در الگوریتم ژنتیک عادی، کیفیت جداول زمان‌بندی تولید شده را به شدت بالا می‌برد که دلیل آن می‌تواند عدم تطبیق خصایص الگوریتم ژنتیک به عنوان یک ابزار بهینه سازی عمومی با انتظارات همه مسائل باشد.*

\\جدول بهینه نهایی حاصل از اجرای این الگوریتم با عنوان «آزمون شماره ۱» در جدول شماره ۴ آمده است.



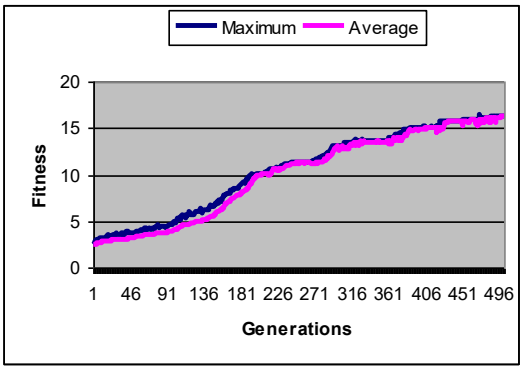
شکل ۵ منحنی برازش- نسل مربوط به اجرای الگوریتم ژنتیک عادی و استاندارد برای جمعیتی با اندازه ۵۰۰ (آزمون ۱)

۲-۱-۴ اجرای الگوریتم ژنتیک عادی با افزایش نرخ جهش و ادغام (آزمون ۲)

نمونه دیگر از اجرای الگوریتم ژنتیک معمولی در شکل ۶ نشان داده شده است. شرایط این اجرا کاملاً مانند حالت اول است، با این تفاوت که نرخ ادغام و جهش افزایش داده شده است:

- مقدار دهی اولیه: کاملاً تصادفی
- نرخ ادغام: ٪۹۰
- نرخ جهش: متغیر ٪۲۰ یا ٪۵۰

همان طور که در شکل ۶ مشخص است، تنها تفاوت با حالت قبلی این است که حد آستانه برازش‌ها کمی بالاتر از حالت قبلی است که دلیل آن افزایش نرخ ادغام و جهش می‌باشد و باعث اجتناب از یکنواختی جامعه پاسخ‌ها و بدام افتادن در نقاط ماکزیمم محلی شده است. نتایج جدول بهینه نهایی حاصل از اجرای این الگوریتم با عنوان «آزمون شماره ۲» در جدول شماره ۴ آمده است.

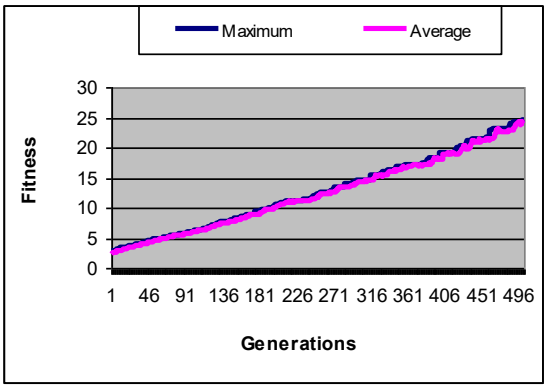


شکل ۶ منحنی برازش- نسل مربوط به اجرای الگوریتم ژنتیک عادی با افزایش نرخ جهش و ادغام برای جمعیتی با اندازه ۵۰۰ (آزمون ۲)

۳-۱-۴ اجرای الگوریتم ژنتیک عادی با کپی کردن مستقیم کروموزوم‌های بهتر در نسل بعد (آزمون ۳)

- مقدار دهی اولیه: کاملاً تصادفی
- نرخ ادغام: ۶۰٪
- نرخ جهش: متغیر ۵٪

همان طور که در شکل ۷ مشخص است به علت کپی کردن مستقیم کروموزوم‌های بهتر در نسل بعد، نمودار صعودی است.



شکل ۷ منحنی برازش- نسل مربوط به اجرای الگوریتم ژنتیک عادی با کپی کردن مستقیم کروموزومهای بهتر در نسل بعد برای جمعیتی با اندازه ۵۰۰ (آزمون ۳)

نتایج جدول بهینه نهایی حاصل از اجرای این الگوریتم با عنوان «آزمون شماره ۳» در جدول ۴ آمده است.

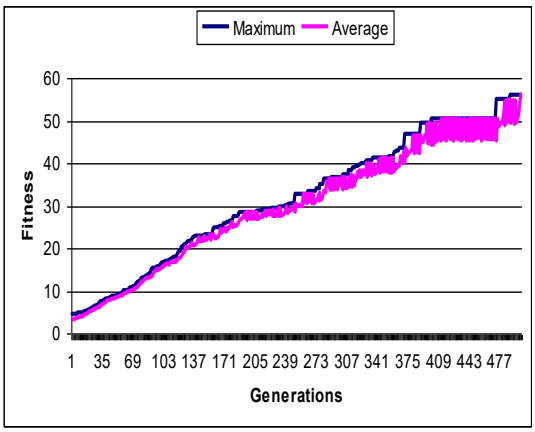
همان طور که از نتایج آزمون‌های ۱ تا ۳ مشخص است، مشکل اصلی و مهم عدم پیشرفت برازش به دلیل نقض محدودیت سخت تکه تکه شدن ساعات یک درس می باشد(محدودیتی که جریمه آن نیز بالا است). بنابراین بهتر است در هنگام مقدار دهی اولیه، عدم تکه تکه شدن ساعات یک درس در نظر گرفته شود.

۴-۱-۴ اجرای الگوریتم ژنتیک عادی با رعایت محدودیت سخت عدم مجزا شدن ساعات یک درس (آزمون ۴)

در این اجرا با وجود اینکه حالت تصادفی اولیه[15] با اجراهای قبلی یکسان است ولی به دلیل اینکه یک محدودیت سخت در مرحله مقدار دهی اولیه رعایت می شود، جداول اولیه مقداری با جداول اولیه در مرحله قبل متفاوت است:

شرایط این اجرا به صورت زیر است:

- مقدار دهی اولیه: شبه تصادفی، با در نظر گرفتن عدم مجزا شدن ساعات درس
- نرخ ادغام: ۶۰٪
- نرخ جهش: متغیر ۵٪



شکل ۸ منحنی برازش-نسل مربوط به اجرای الگوریتم ژنتیک عادی با رعایت محدودیت سخت عدم مجزا شدن ساعات یک درس برای جمعیتی با اندازه ۵۰۰ (آزمون ۴)

نتایج جدول بهینه نهایی حاصل از اجرای این الگوریتم با عنوان «آزمون شماره ۴» به همراه جدول اولیه آن در جدول شماره ۴ آمده است، و نیز شکل ۸ بردار برازش/نسل آزمون را نشان می دهد.

با وجود این که برازش کلی در مرحله اول با برازش در مراحل اولیه در سه اجرای قبل تقریبا یکسان است، ولی در پایان چون نیازی برای تلاش الگوریتم به منظور از بین بردن محدودیت سخت عدم تکه تکه شدن ساعات درس وجود ندارد، کروموزوم‌ها خیلی راحت‌تر بهینه می‌شوند و برازش نهایی بسیار بالاتر از اجراهای قبلی است.

باید دقت شود با وجود این که در مرحله مقدار دهی اولیه نقض عدم تکه تکه شدن ساعات یک درس برابر صفر است ولی به هر حال درطول اجرای الگوریتم این شرط به میزان کمی نقض شده است که علت آن تأثیر عملگرهای جهش و ادغام روی کروموزوم‌ها و جابه‌جا شدن آنها می‌باشد.

از این مرحله به بعد و با توجه به نتایج حاصله این جمع‌بندی حاصل می‌شود که بهتر است به منظور بالا بردن مقدار برازش در نسل‌های اول و در نتیجه سبک‌تر شدن کار الگوریتم در مراحل بعدی، بعضی از محدودیت‌های سخت در مرحله مقدار دهی اولیه رعایت شوند. همان‌گونه که قبلاً

گفته شد عدم تفکیک ساعات درس، تفکیک دروس نظری و آزمایشگاه‌ها و توجه به گنجایش کلاس یا آزمایشگاه، محدودیت‌های سختی هستند که بین مقداردهی اولیه و تشکیل جامعه ابتدایی لحاظ شده‌اند.

۴-۲ نتایج حاصل از اجرای الگوریتم جدید پیشنهادی

در جدول شماره ۴ نتایج مربوط به اجرای الگوریتم پیشنهادی برای دروس ارائه شده در نیمسال اول تحصیلی با عنوان آزمون شماره ۵ همراه با جدول اولیه آن ارائه شده است. مشخصاً نتایج حاصل از این آزمون به مراتب بهتر از استفاده از الگوریتم ژنتیک استاندارد است. به عنوان مثال در آزمون ۴ (شکل ۸) بعد از حدود ۴۵۰ تکرار نسل برازش به زیر ۶۰ محدود است، لیکن در آزمون ۵ (شکل ۱۰) بعد از حدود ۴۰۰ مرتبه تکرار نسل برازش به حدود ۴۵۰ رسیده است.

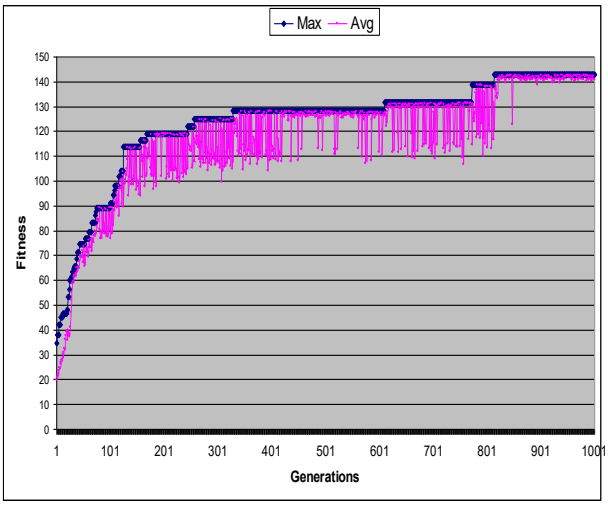
۴-۲-۱ نمودارهای برازش بیشینه– نسل نهایی

پس از اجرای موفقیت آمیز آزمون ۵، به منظور ارزیابی تأثیر اندازه جمعیت بر کیفیت اجرای الگوریتم، این آزمون مجدداً برای جمعیتی با اندازه ۵۰ و سپس ۱۰۰ انجام شد که نمودارهای مربوط به هر سه آزمون در شکل‌های ۹، ۱۰و۱۱ قابل مشاهده است.

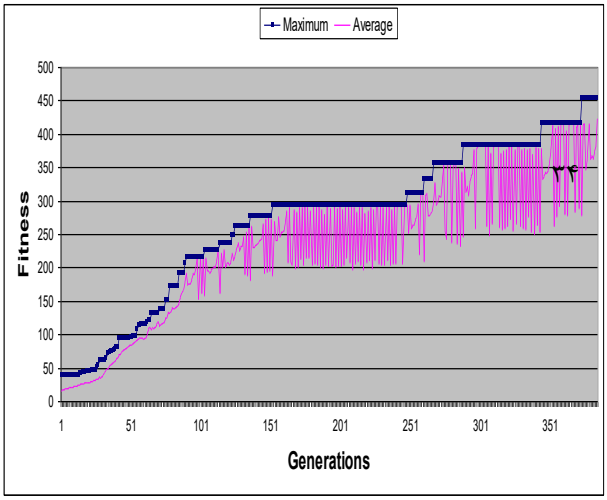
نمودار «برازش بیشینه – نسل» (شکل‌های ۹-۱۱) همیشه نموداری صعودی است. از آنجا که در روش پیشنهادی نیمه بهتر جمعیت مستقیماً در نسل بعد کپی می شود، هیچ گاه بر اثر گذشت نسل‌ها بیشینه جمعیت جدید کمتر از بیشینه جمعیت در نسل‌های قبلی نخواهد شد ممکن است این بیشینه به مدت چندین نسل ثابت بماند، مگر زمانی که بر اثر واقع شدن جهش و ادغام بر روی جمعیت، کروموزومی تولید شود که برازش آن بیشتر از بیشینه فعلی باشد. هرچه اندازه جمعیت کمتر باشد، توقف بیشتری روی هر برازش بیشینه خواهیم داشت، یعنی نسل‌های بیشتری می‌گذرد و برازش بیشینه ثابت می‌ماند، چون احتمال ساختن جدولی که برازش آن از برازش بیشینه کنونی بیشتر باشد با سایز جمعیت نسبت مستقیم دارد. البته در مورد میانگین برازش جامعه، نوسان طبیعتاً قابل مشاهده است. وقتی اندازه جمعیت کم است هر چقدر که در موقع نزدیک شدن بیشینه و میانگین برازش اعضای جمعیت به یکدیگر (همگرا شدن جمعیت به یک جدول خاص) نرخ جهش زیاد شود، بازهم چون تنوع جداول زیاد نیست، ترکیب آنها با یکدیگر و عمل جهش روی آنها هنوز شانس زیادی برای به دست آمدن یک مقدار برازش بالاتر از مقدار بیشینه کنونی ایجاد نمی‌کند. در حقیقت به نوعی در یک چرخه قرار می‌گیریم که مرتباً نرخ جهش را افزایش می دهد، جداول جدید (ولی با برازشی کمتر از برازش بیشینه کنونی) تولید می‌کند، و در نتیجه باعث کاهش میانگین و جلوگیری از همگرا شدن جمعیت می‌شود. سپس دوباره با ترکیب این جداول، میانگین افزایش می‌یابد و به بیشینه نزدیک می‌شود. به همین ترتیب این چرخه ممکن است ساعت‌ها برای هر افزایش کوچک در بیشینه جمعیت ادامه پیدا کند، و جدولی با برازش بالا حاصل نشود.

شکل ۹ منحنی مربوط به اجرای الگوریتم برای جمعیتی با اندازه ۱۰۰ را نشان می دهد. در این شکل محور افقی، تعداد تکرار نسل و محور عمودی، مقدار تابع برازش است. خطوط نازک میانگین و خطوط ضخیم ماکزیمم برازش را نمایش می دهند. کاملاً مشهود است که چگونه الگوریتم در تلاش برای به دست آوردن مقادیر برازش بالاتر، مرتباً باعث کاهش و افزایش میانگین برازش جمعیت می‌شود تا بالاخره بعد از مدت زمان قابل ملاحظه‌ای بتواند برازشی بالاتر از بیشینه کنونی خود تولید کند. همان طور که در شکل مشاهده می‌کنید در نسل‌های اولیه که بیشینه برازش کم است، شیب منحنی زیاد است؛ چون احتمال تولید جداول با برازش بالا هنوز قابل ملاحظه است. هر چه نسل‌های بیشتری می‌گذرد و برازش بیشینه بیشتر می‌شود، احتمال تولید جداولی با برازش بالاتر از بیشینه کنونی کمتر شده و نسل‌های بیشتری طول می‌کشد تا چنین جدولی تولید شود. در نتیجه شیب منحنی کم می‌شود و نواحی تخت در منحنی بیشتر ظاهر می‌گردد.

به طور مثال همان گونه که در شکل قابل مشاهده است، از نسل ۳۲۹ تا نسل ۶۱۰، یعنی به مدت ۲۸۱ نسل، برازش بیشینه روی ۱۲۸ ثابت بوده است. پس از گذشت ۱۰۰۰ نسل، بیشینه برازش به مقدار ۱۴۲ رسیده است، که تا بیشینه مطلوب فاصله زیادی **دارد**، هر چند بسیار بالاتر از برازش‌های حاصل از الگوریتم‌های ژنتیک استاندارد است.



شکل ۹ منحنی مربوط به اجرای الگوریتم پیشنهادی برای جمعیتی با اندازه ۱۰۰ (آزمون ۵)

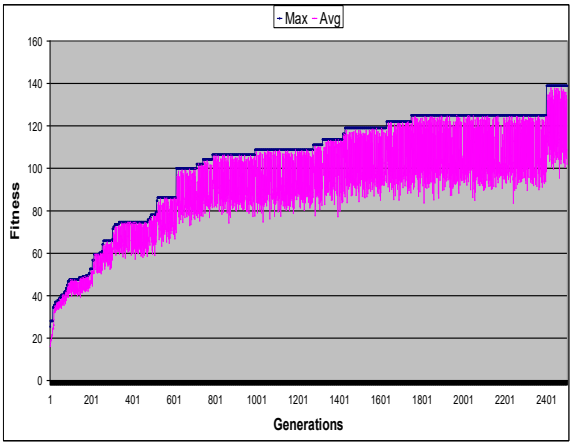


شکل ۱۰ منحنی برازش- نسل مربوط به اجرای الگوریتم پیشنهادی برای جمعیتی با اندازه ۵۰۰ (آزمون ۵)

یک نمونه دیگر از نتایج اجرای الگوریتم با اندازه جمعیت ۵۰۰ و تعداد نسل ۴۰۰ در شکل ۱۰ نشان داده شده است. در حین اجرا، پس از گذشت تعداد ۱۵۴ نسل، به مقدار برازش بیشینه ۶۲۵ رسیدیم، که در مقایسه با اجرای قبلی، با اندازه جمعیت ۱۰۰ تایی، که پس از گذشت ۱۰۰۰ نسل برازش بیشینه به ۱۴۲ رسیده بود، قابل ملاحظه می‌باشد. همان طور که در نمودار مشهود است، نرخ تغییرات برازش بیشینه نسبت به حالت قبل بسیار بیشتر است. تا قبل از نسل ۱۵۴ نمودار برازش بیشینه دارای شیب تندی به سمت بالا است، ولی پس از رسیدن به برازش ۶۲۵، اتفاقی تقریباً مشابه با حالت قبل می‌افتد.

در حقیقت صرف نظر از تعداد نسل‌ها به هر حال به این مرحله که میانگین و بیشینه بسیار به هم نزدیک شده، و نرخ تغییرات برازش بیشینه کاهش می‌یابد، خواهیم رسید. با این تفاوت که وقتی اندازه جمعیت بزرگ باشد نسل های بیشتری باید سپری شود تا این اتفاق رخ دهد و شانس رسیدن به برازش‌های بالاتر نیز بیشتر است.

همان طور که پیشتر ذکر شد، یکی از ابتکاراتی که در این الگوریتم به کار برده شده این است که به وسیله در نظر گرفتن دو نرخ جهش متفاوت که هر یک در شرایط خاص خود اعمال می شود، احتمال همگرا شدن جمعیت به یک برازش واحد و در نتیجه به دام افتادن در یک بیشینه محلی بسیار ناچیز و در سایز جمعیت‌های بالا نزدیک به صفر است. یعنی گرچه ممکن است نسل‌های زیادی بگذرد و بیشینه تغییری نکند، اگر زمان کافی به تعداد دفعات تکرار نسل الگوریتم داده شود، به احتمال زیاد الگوریتم با افزایش خودکار نرخ جهش از این بیشینه محلی نجات خواهد یافت. شکل ۱۱ که مربوط به اجرای الگوریتم با اندازه جمعیت ۵۰ می‌باشد، نکات ذکر شده را به خوبی نشان می‌دهد. به طور مثال از نسل ۱۷۴۶ تا نسل ۲۳۹۹، یعنی به مدت ۶۵۳ نسل، برازش بیشینه ۱۲۵ می باشد، ولی نهایتاً الگوریتم از دام این بیشینه محلی رها یافته و به برازش۱۳۹ تغییر مقدار می‌دهد.



شکل ۱۱ منحنی مربوط به اجرای الگوریتم پیشنهادی برای جمعیتی با اندازه۵۰ (آزمون ۵)

ابتکار دیگری که برای افزایش کارایی تابع جهش در به وجود آوردن جداول متفاوت در هنگام میل کردن برازش‌های اعضای جمعیت به سمت عددی خاص انجام شد، این بود که تابع جهش در هر بار اعمال شدن، تعداد متفاوتی از بیت‌ها را جابه‌جا می‌کند. این کار احتمال ایجاد جداول جدید، و در نتیجه احتمال ایجاد جدولی با مقدار برازش بیشتر از برازش بیشینه کنونی را افزایش می‌دهد. در جدول ۴، سطرهای ابتدایی، امتیاز دو جدول از جداول اولیه قبل از بهینه‌سازی توسط الگوریتم ژنتیک را نشان می‌دهند. همان‌گونه که مشاهده می‌شود، امتیاز این جدول‌ها بسیار پایین و شروط سخت و نرم نقض، شده متعدد می‌باشند. در سطرهای بعدی نتایج آزمون الگوریتم ژنتیکی استاندارد قابل مشاهده است. دیده می‌شود که میانگین و ماکزیمم امتیاز (برازش) جداول در این آزمون‌ها بسیار نزدیک به هم است که این همر نشانگر همگرایی و یکنواختی جامعه پس از چندین تکرار نسل می‌باشد. همچنین، هر چند شرایط سخت تداخل ساعات برای دانشجویان و اساتید هر دو بهبود یافته است، اما تکه تکه شدن ساعات دروس و شرط اندازه کلاس از بین شرایط سخت و تقریباً عموم شرایط نرم لحاظ نشده‌اند که این خود باعث کاهش امتیاز کلی جداول تولیدی در این آزمون‌ها شده است. در عوض نتایج آزمون ۵ حاصل از به کارگیری الگوریتم ژنتیک پیشنهادی برتری محسوسی را نشان می‌دهد. کلیه شرایط سخت لحاظ شده‌اند و تنها سه مورد نقض شرایط نرم به چشم می‌خورد.

بدین دلیل امتیاز جدول به ۶۲۵/۰۱ رسیده است که تقریبا ۱۱ برابر بهتر از الگوریتم‌های ژنتیک استاندارد می‌باشد. جدول بهینه تولیدی از نظر جمیع شرایط یک جدول زمان‌بندی مطلوب و خوش ساخت محسوب می‌شود.

۵– نتیجه گیری

در این مقاله خصوصیات الگوریتم ژنتیک به عنوان ابزاری مناسب در بهینه سازی پاسخ‌های یک مسأله CSP مورد بررسی قرار گرفت. سپس مسأله طراحی جداول زمان‌بندی برای دروس دانشگاهی به عنوان یک مسأله کاربردی CSP معرفی و یک راه حل ژنتیکی استاندارد برای آن ارائه شد. به کمک افزایش نرخ جهش و ادغام و در مقابل انتقال نیمه بهتر جمعیت در نسل بعد، شبه تصادفی بودن مرحله مقدار دهی اولیه، و دیگر تغییراتی که در الگوریتم ژنتیک معمولی اعمال شد از همگرا شدن الگوریتم جلوگیری شده و نتایج مطلوبی حاصل گشته است. همچنین مشخص گردید که اجتناب از مقداردهی اولیه کاملاً تصادفی و سعی در گریز از مینیمم‌های محلی با دستکاری در پارامتر جهش می‌تواند در بهبود بازدهی الگوریتم ژنتیکی در این بهینه‌سازی خاص مؤثر باشد. *خصوصاً که در جستجوهای غیر سراسری مانند الگوریتم ژنتیک، همگرایی پیش از موعد موجب یکنواختی جامعه و به دام افتادن در یک اپتیموم محلی می‌شود.* نتایج حاصله نشان می‌دهد الگوریتم ژنتیک پیشنهادی در طراحی جداول زمان‌بندی خوش ساخت برای یک دانشکده مفروض به خوبی عمل می‌کند و جداول تولیدی ضمن عدم نقض شرایط سخت و فقط با نادیده گرفتن تعدادی شرط نرم، می‌توانند به برازش چندین برابر جداول ساخته شده توسط الگوریتم‌های ژنتیک عادی دست یابند. نتایج حاصل از تحقیق بر کارایی روش‌های بومی سازی شده مبتنی بر یک الگوریتم تکاملی استاندارد و معمول تأکید دارند. *آزمون‌های مشابهی روی جدول زمان‌بندی امتحانات یک دانشکده نیز صورت گرفته است که نتایج آنها نیز تأیید کننده نتایج آزمون‌ها و استدلال‌های ارائه شده در این مقاله می‌باشد.*

پی نوشت

[1] Constraints Satisfaction problem
[2] Graph Coloring
[3] Heuristic
[4] Memetic
[5] John Holland
[6] Fitness
[7] Ben Paechter, R.C.Rankin
[8] Edmund Burke,David Elliman,Rupert Weare
[9] Wilhelm Erben,Jurgen Keppler
[10] Michael W.Carter
[11] PMX: Partially Mapped Crossover
[۱] Initialization
[۱] NP-Hard (Nondeterministic Polynomial)
[۱] Crossover
[۱] Seed

مراجع

[1] Carter M., *A Comprehensive Course Timetabling and Student Scheduling System at the University of Waterloo*, Lecture Notes in Computer Science, Vol. 2079, **2001**, pp. 64-82.

[2] Burke E., Elliman D., Wearer R., *A Genetic Algorithm based University Timetabling System*, Proceedings of the 2nd East-West International Conference on Computer Technologies in Education, **1994**, pp. 35-40.

[3] Russell S., Norvig P., *Artificial Intelligence: A Modern Approach*, 3rd Ed., Prentice Hall, **2009**.

[۴] علیرضا مهدی، مقدمه‌ای بر الگوریتم های ژنتیک و کاربردهای آن، ناقوس اندیشه، **۱۳۸۵**.

[5] Goldberg D., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Welsy, **1998**.

[6] Whitely D., *A Genetic Algorithm Tutorial*, Journal of Statistics and Computing Vol. 4, **1994**, pp. 65-85.

[7] Rossi-Doria O., Paechter B, *A Memetic Algorithm for University Course Timetabling*, Proceedings of the CO2004 Conference, Lancaster, UK, **2004**, p. 65.

[8] Erben W., Keppler J., *A Genetic Algorithm Solving a Weekly Course-timetabling Problem*, Proceedings of The First International Conference on The Practice and Theory of Automated Timetabling, Edinburgh, UK, **1995**, pp. 198-211.

[9] Abramson D., Abela J., *A Parallel Genetic Algorithm for Solving the School Timetabling Problem*, Proceedings of the 15th Australian Computer Science Conference, Hobart, Australia, **1992**, pp.101.

[10] Abdullah S., Burke E., McCollum B., *An Investigation of Variable Neighbourhood Search for University Course Timetabling*, Proceedings of the 2nd Multidisciplinary Conference on Scheduling: Theory and Applications, **2005**, pp. 413-427.

[11] Vorac J., Vondrak I., Vlcek K., *School Timetabling Using Genetic Algorithm*, Technical Report, VSB-Technical University of Ostrava, Czech Republic, **2002**.

[12] Cooper T., Kingston J., *The Complexity of Timetable Construction Problems*, Lecture Notes in Computer Science, Vol. 1153, **1996**, pp. 281-295.

[13] Beligiannisa G., Moschopoulosa C., Kaperonisa G., Likothanassisa D., *Applying Evolutionary Computation To The School Timetabling Problem: The Greek Case*, Journal of Computers & Operations Research, Vol. 35, **2008**, pp. 1265–1280.

[14] Pillay N., Banzhaf W., *An Informed Genetic Algorithm For The Examination Timetabling Problem*, Journal of Applied Soft Computing, Vol. 10, **2010**, pp. 457–467.

[15] Wilke P., Ostler J., *Benchmarking Curriculum-Based Course Timetabling: Formulations, Data Formats, Instances, Validation, and Results*, Proceedings of the 7th International Conference for the Practice and Theory of Automated Timetabling (PATAT'2008), Montreal, Canada, **2008**.

[۱۶] نیپولیتان ریچارد و نعیمی‌پور کیومرث، *طراحی الگوریتم ها با شبه کدهای C++*، جهاد دانشگاهی مشهد، **۱۳۸۱**، صفحه ۳۶۱.

جدول۳ خروجی برنامه برای روز شنبه به ازای ۳ کلاس، ۳ آزمایشگاه و ۱ سایت کامپیوتر

عدد قبل از علامت # کد استاد و عدد بعد از G گروهی است که آن درس برای آن گروه ارائه شده است

	۸-۹	۹-۱۰	۱۰-۱۱	۱۱-۱۲	۱۳-۱۴	۱۴-۱۵	۱۵-۱۶	۱۶-۱۷	۱۷-۱۸	۱۸-۱۹
	طراحی لگوریم ۶۱۰-۱ #۱۰ G2 --- شبکه ۲ ۶۱۰-۲ #۶ G4	شبکه ۲ ۶۱۰-۲ #۶ G4 --- زبانهای برنامه سازی، ۶۱۰-۳ #۱۶ G2 --- آز ریزپردازنده ۱ Lab_mic #۳ G3	رمزنگاری ۶۱۰-۲ #۵ G4 --- معماری ۶۱۰-۳ #۲۰ G2 --- آز ریزپردازنده ۱ Lab_mic #۳ G3	مهندسی اینترنت ۶۱۰-۲ #۷ G4 --- معماری ۶۱۰-۳ #۲۰ G2 --- آز ریزپردازنده ۱ Lab_mic #۳ G3	مهندسی نرم افزار ۲ ۶۱۰-۱ #۱۲ G3 --- کامپایلر ۶۱۰-۲ #۸ G4 --- مدارهای واسط ۶۱۰-۳ #۴ G4	کامپایلر ۶۱۰-۲ #۸ G4 --- ریزپردازنده ۱ ۶۱۰-۳ #۳ G3	مدارهایVLSI ۶۱۰-1 #۲ G4 --- ریاضی مهندسی ۶۱۰-۲ #۹ G3 --- مبانی کامپیوتر ۶۱۰-۳ #۲۱ G1	ریاضی مهندسی ۶۱۰-۲ #۹ G3 --- مبانی کامپیوتر ۶۱۰-۳ #۲۱ G1 --- آز الکترونیک Lab_dig #۱۸ G4	سیستم عامل ۶۱۰-۲ #۱۰ G3 --- پیشرفته ۶۱۰-۳ #۲۰ G1 --- آز الکترونیک Lab_dig #۱۸ G4	سیستم عامل ۶۱۰-۲ #۱۰ G3 --- پیشرفته ۶۱۰-۳ #۲۰ G1 --- آز الکترونیک Lab_dig #۱۸ G4
شنبه										

جدول۴ نتایج حاصل از آزمون ها

آزمون	نقض شرایط سخت							نقض شرایط نرم					امتیاز میانگین	امتیاز ماکزیمم
	تداخل ساعات اساتید	تداخل ساعات دانشجویان	تکه تکه شدن ساعات درس	در نظر نگرفتن ساعت ناهار و نماز	مکان نامناسب برای یک درس	در نظر نگرفتن امکانات لازم برای دروس آزمایشگاه	عدم توجه به اندازه کلاس	سه ساعت متوالی یک درس سه واحدی	سه ساعت درس سه واحدی در یک روز	ساعات درس سه واحدی در دو روز متوالی	فواصل خالی بین ساعات درسی دانشجویان	نادیده گرفتن ترجیحات اساتید		
جدول اولیه ۱	۲۰	۵۸	۱۲۰	۰	۲۵۲	۰	۲۰	۰	۲	۱۳	۵۹	۰	-	۲/۵۵
جدول اولیه ۲	۴	۶۲	۰	۱۴	۲۵۲	۰	۴۸	۰	۱۲	۸	۴۵	۰	-	۳/۱۴
آزمون ۱	۰	۲	۸۱	۰	۴	۰	۰	۰	۲	۴	۱۸	۰	۱۳/۲۳	۱۳/۲۶
آزمون ۲	۰	۴	۶۶	۰	۰	۰	۲	۰	۲	۶	۱۴	۰	۱۶/۲۰	۱۶/۲۹
آزمون ۳	۰	۲	۴۲	۲	۰	۰	۱	۰	۲	۴	۹	۰	۲۳/۹۵	۲۴/۵۱
آزمون ۴	۰	۴	۱۲	۴	۰	۰	۴۸	۰	۰	۲	۲	۰	۵۶/۰۸	۵۶/۱۸
آزمون۵ (اندازه جمعیت ۵۰۰)	۰	۰	۰	۰	۰	۰	۰	۰	۰	۲	۱	۰	۶۲۴/۷۲	۶۲۵/۰۱