دانشگاه آزاد اسلامی

واحد قزوین

دانشکده علوم تحقیقات- گروه کامپیوتر

پایان نامه برای دریافت درجه کارشناسی ارشد "M.A"

رشته کامپیوتر- گرایش نرم افزار

عنوان:

حل مساله تخصیص درجه دوم به روش الگوریتم جستجوی گرانشی ترکیبی

استاد راهنما:

جناب آقای دکتر بهروز معصومی

نگارش:

سید حامد ساعی

زمستان ۱۳۹۴


بسم الله الرحمن الرحيم

تقدیر و تشکر:

تشکر و سپاس بی پایان مخصوص خدایی است که بشر را آفریده و به او قدرت اندیشیدن داده و توانایی های بالقوه در وجود انسان قرار داده و او را امر به تلاش و کوشش نموده و راهنمایانی برای هدایت بشر فرستاده است.

پس از ارادت خاضعانه به درگاه خداوند بی همتا لازم است از استاد ارجمند جناب آقای دکتر بهروز معصومی به خاطر سعه صدر و رهنمودهای دلسوزانه که در تهیه ی این پایان نامه مرا مورد لطف خود قرار دادند و راهنماییهای لازم را نمودند تشکر و قدردانی نموده، موفقیت همگان را از درگاه احدیت خواهانم.

چکیده:

مساله تخصیص درجه دوم (QAP) به عنوان یکی از پیچیده ترین مسائل بهینه سازی شناخته می شود که به اختصاص تعدادی تسهیل به تعدادی مکان می پردازد. هدف، تخصیص هر تسهیل به یک مکان می باشد که در آن هزینه کل حداقل گردد. این مساله یک مساله NP-hard می باشد و به دست آوردن جواب بهینه برای مسائل سایز بزرگ به صورت دقیق امکان پذیر نیست. در این پایان نامه، روشی برای حل مساله تخصیص درجه دوم با استفاده از یک الگوریتم فرا ابتکاری به نام GSA ارائه شده است. فرض بر این است که فعالیت ها قطعی بوده و فعالیت احتمالی وجود ندارد. تعداد مکان ها با تعداد تسهیل ها نیز برابر بوده و هر تسهیل فقط و فقط به یک مکان می تواند تخصیص یابد. همچنین از یک روش ترکیبی با ترکیب الگوریتم جستجوی گرانشی با الگوریتم ژنتیک با نام GSA-GA برای حل مساله QAP مورد استفاده شده است. این الگوریتم بر روی تعدادی از مسائل نمونه QAP موجود در QAPLIB آزمایش شده و در اکثر موارد قادر به دست آوردن بهترین راه حل به دست آمده تا کنون یا راه حل های با خطای کمتر نسبت به الگوریتم های مشابه بوده است.

کلمات کلیدی: الگوریتم جستجوی گرانشی، مساله تخصیص درجه دوم، بهینه سازی ترکیبی، روش های فرا ابتکاری، الگوریتم ژنتیک.

فهرست مطالب

فهرست اشکال

فهرست جداول

فهرست نمودارها

فصل اول: کلیات پژوهش

۱-۱ مقدمه

در طول دهه های اخیر الگوریتم های الهام گرفته شده از طبیعت به طور وسیعی برای حل مسائل بهینه سازی مختلف استفاده شده است. در بین آنها الگوریتم های هوش جمعی و تکاملی[1] به طور گسترده ای به عنوان ابزاری برای جستجو و بهینه سازی مورد استفاده قرار گرفته است. محاسبه راه حل های بهینه برای اکثر مسائل بهینه سازی که در خیلی از زمینه های کاربردی و عملی مشاهده می گردند کاری دشوار و سخت است. روش های حل مسائل بهینه سازی مشتمل بر دو دسته روش های دقیق و روش های تقریبی می باشد. روش های دقیق راه حل های بهینه را به دست آورده و شرایط بهینگی را تضمین می کنند. روش های ابتکاری(تقریبی) راه حل های با کیفیت بالا را در زمان معقولی تولید می کنند، اما تضمینی برای یافتن راه حل بهینه سراسری ندارند. مساله تخصیص درجه دوم یکی از مسائل بهینه سازی ترکیبی است که در خصوص آن در بسیاری از شرکت ها جهت تخصیص تعدادی تسهیل به تعدادی مکان بررسی ها و پژوهش هایی انجام شده است. موضوعی که در این فرایند اهمیت ویژه ای دارد بحث هزینه های تخصیص می باشد که تلاش در این مساله بر حداقل سازی این گروه از هزینه ها می باشد. کاربرد مدل های مربوط به این مساله در مسائل دنیای واقعی نمود زیادی دارد. از جمله کاربرد های آن می توان به چیدمان ساختمان ها در بیمارستان ها، مدیریت انبارها و استراتژی های توزیع، حداقل سازی طول کل سیم در مدارهای الکترونیک، تخصیص برخی کارخانجات به برخی مکان ها، مساله سیم کشی پشت مدار، طراحی پنل های کنترلی و صفحه کلید ماشین نویس ها اشاره نمود. در این میان مسائل با ابعاد بزرگتر از ۲۰ معمولا از روش های دقیق قابل حل نیستند و یا در صورت حل شدن به لحاظ زمانی توجیه اقتصادی ندارند. لذا در این حالت ها از روش های فرا ابتکاری استفاده می گردد. این روش ها عمدتا در زمان کوتاهی به جواب دست می یابند اما دقت آنها زیاد نبوده و به جواب بهینه نمی رسند. در بین روش های فرا ابتکاری(مکاشفه ای) الگوریتم های مربوط به نیروی گرانش اخیرا مطرح شده و نتایج خوبی از آنها حاصل شده است. در این بین الگوریتم جستجوی گرانشی در حل بسیاری از مسائل نتایج خوبی را کسب نموده است، و این اولین بار است که از این الگوریتم برای حل مساله تخصیص استفاده می گردد.

[1] Swarm intellligence

۱-۲ بیان مساله

برنامه ریزی عدد صحیح[۲]، برنامه ریزی پویا[۳]، و روش های تئوری گراف[٤] (ترکیبی) نگرش های سنتی، برای حل مسائل بهینه سازی ترکیبی است. حل اکثر مسائل بهینه سازی ترکیبی مشکل است، زیرا این مسائل عمدتا مسائل با مقیاس بزرگ هستند و با تجزیه آنها به مسائل کوچک تر دشوار است.

مساله تخصیص درجه دوم[۵](QAP) یکی از مسائل بهینه سازی ترکیبی[٦] (CO) است که به بررسی تخصیص تعدادی تسهیل به تعدادی مکان می پردازد. موضوعی که در این مساله اهمیت ویژه ای دارد بحث هزینه های تخصیص می باشد که تلاش در این مساله برای حداقل سازی این هزینه هاست. حل اکثر مسائل بهینه سازی ترکیبی که در خیلی از زمینه های کاربردی و عملی مشاهده می گردند مشکل است زیرا این مسائل عمدتا مسائل با مقیاس بزرگ هستند و یا تجزیه آنها به مسائل کوچک تر دشوار است. روش های حل مسائل بهینه سازی ترکیبی مشتمل بر دو دسته روش های دقیق و روش های تقریبی می باشد. روش های دقیق راه حل بهینه را به دست می آورد و شرایط بهینگی را تضمین می کنند. روش های تقریبی (یا ابتکاری و فرا ابتکاری ها) راه حل های با کیفیت بالا را در زمان قابل قبولی ارائه می کند، برخلاف الگوریتم های بهینه سازی دقیق، فرا ابتکاری ها بهینه بودن جواب به دست آمده را تضمین نمی کنند. مسائل با ابعاد بزرگتر از ۲۰ معمولا با روش های دقیق قابل حل نیستند و یا در صورت حل شدن توجیح اقتصادی ندارند. لذا از روش های فرا ابتکاری استفاده می شود.

مجموعه $N = \{1,2,\dots,n\}$ و ماتریس های $D = d_{kl}$و $F = f_{ij}$ و $C = c_{ik}$ با اندازه $n \times n$داده شده اند. مساله تخصیص درجه دوم به صورت زیر بیان می شود:

$$\min_{p \in \pi_N} \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}\, d_{p(i)p(j)} + \sum_{i=1}^{n} c_{p(i)i} \qquad (1\text{-}1)$$

که در آن π_N مجموعه تمام جایگشت های N می باشددر واقع n! جایگشت امکان پذیر است. و راه حل بهینه، بهترین جایگشت است.

[2] Integer programming
[3] Dynamic programming
٤ Graph theory
[5] Quadratic Assignment Problem
[6] Combinatorial Optimization

ماتریس F ماتریس جریان[7] می باشد. برای مثال f_{ij} جریان بین تسهیلات i و j است و ماتریس D ماتریس فاصله[8] می باشد، برای مثال d_{kl} فاصله بین مکان k و مکان l می باشد. و ماتریس C، ماتریس هزینه[9] است برای مثال c_{ik} هزینه جایگیری تسهیل i در مکان k می باشد. هدف، یافتن یک تخصیص از تمام تسهیل ها به تمامی مکان ها است که در آن هزینه کل تخصیص حداقل شود.

بسیاری از مسائل در دنیای واقعی هستند که می توانند به عنوان مساله تخصیص درجه دوم فرموله شوند. نمونه ای از کاربردهای مساله QAP به شرح زیر است:مورد استفاده به عنوان یک مدل ریاضی مربوط به فعالیت های اقتصادی، مسائل زمان بندی، برای تعریف بهترین طراحی برای صفحه کلید ماشین تحریر و پنل های کنترلی، برنامه ریزی بیمارستان، برای باستان شناسی، در تجزیه و تحلیل واکنش های شیمیایی، واگذاری ساختمان ها در محیط دانشگاه، مساله مربوط به پارک جنگلی، و در نهایت کاربرد جدیدتر آن، مساله طراحی صفحه کلید در دستگاه های صفحه نمایش لمسی است و....

چندین مساله بهینه سازی ترکیبی دیگر که دارای پیچیدگی نمایی هستند از قبیل مساله فروشنده دوره گرد[10](TSP)، مساله بسته بندی سطلی[11](BPP)، مساله بزرگترین خوشه[12] (پیدا کردن بزرگترین زیرگراف کامل)(MCP) هم می توانند به صورت QAP مدل شوند.

مسااله تخصیص درجه دوم به عنوان یک مساله NP-hard شناخته شده است، از این رو یک مساله با پیچیدگی نمایی است. هیچ الگوریتم شناخته شده ای در زمان چند جمله ای برای حل نمونه های QAP با بیش از یک تعداد نسبتا کمی از ورودی وجود ندارد. حتی راه حل دقیق مسائل کوچک (۳۰ > N > ۲۰) از لحاظ محاسباتی غیر بدیهی به نظر می آیند. بدست آوردن راه حل بهینه برای اندازه متوسط از نمونه های QAP (به عنوان مثال ٤٠ > N > ۳۰) نیازمند یک محیط محاسباتی موازی قدرتمند است. از این رو، حل آنها با روش های بهینه سازی دقیق نظیر برنامه ریزی خطی یا شاخه و کران، به لحاظ زمانی، فقط در مورد اندازه های کوچک مقرون به صرفه است. برای حل مسائل با اندازه های بزرگ، به دلیل طولانی بودن راه حل های دقیق، یا در برخی موارد غیرممکن بودن به کارگیری آن ها، راه حل های ابتکاری[13] و فرا ابتکاری[14] مورد استفاده قرار می گیرد. الگوریتم

[7] Flow Matrix
[8] Distance Matrix
[9] Cost Matrix
[10] Traveling Salesman Problem
[11] Bin Packing Problem
[12] Maximum Clique Problem
[13] Heuristic algorithms
[14] Meta-heuristic algorithms

هایی نظیر بهینه سازی کلونی مورچگان[15](ACO)، بهینه سازی ازدحام ذرات[16](PSO)، الگوریتم ژنتیک[17](GA)، الگوریتم جستجوی ممنوع[18](TS) شبیه سازی تبریدی(الگوریتم انجماد تدریجی) [19](SA) و الگوریتم مهاجرت پرندگان[20](MBO) مهم ترین الگوریتم هایی هستندکه تا کنون برای حل مسائل مرتبط با تخصیص درجه دوم، بسیار مورد استفاده قرار گرفته اند.

الگوریتم های تقریبی معمولا بر روی نمونه های بنچمارک(معیار) که ویژگی های جالبی دارند مورد آزمایش قرار می گیرند در مورد QAP، تعداد زیادی از چنین نمونه های معیاری از طریق QAPLIB (یک کتابخانه برای تحقیق در موردQAP) در دسترس هستند. QAPLIB در حال حاضر شامل بیش از ۱۰۰ نمونه است که در تحقیقات پیشیین استفاده شده است و بخشی از آنها از کاربردهای واقعی مانند طرح بیمارستان، طراحی ماشین تحریر و ... برگرفته شده اند.

ما در این پژوهش سعی بر آن داریم که مساله تخصیص درجه دوم را با استفاده از یک روش فرا ابتکاری جدید به نام الگوریتم جستجوی گرانشی[21](GSA) در زمان قابل قبول با جواب بهینه تقریبی حل کنیم و در نهایت عملکرد آن را در حل نمونه های معیار در QAPLIB با نتایج سایر الگوریتم های ارائه شده مقایسه کنیم.

۱-۳ اهداف پژوهش

۱- حل مساله تخصیص درجه دوم با استفاده از یک روش جدید.

۲- بررسی امکان بهبود نتایج الگوریتم ارائه شده با استفاده از ترکیب آن با سایر الگوریتم های فراابتکاری مانند الگوریتم ژنتیک.

۳- تحلیل عملکرد الگوریتم ارائه شده و مقایسه نتایج به دست آمده از آن با الگوریتم های مشابه به کار رفته

۱-٤ فرضیات پژوهش

۱- در این پژوهشQAP های دقیق که در آنها مختصات مکان ها دقیق فرض شده است مدنظر هستند.

---

15 Ant Colony Optimization
16 Particle Swarm Optimization
17 Genetic algorithm
18 Tabu Search
19 Simulated annealling
20 Migrating Birds Optimization
21 Gravitational Search Algorithm

۲- همچنین فرض بر این است که انجام تمامی فعالیت ها قطعی بوده و فعالیت احتمالی نداریم.

۱-۵ ضرورت انجام پژوهش

با توجه به اهمیت کاهش هزینه ها در بنگاه های اقتصادی به لحاظ افزایش رقابت پذیری تخصیص ماشین ها به مکان ها با حداقل هزینه و به صورت بهینه اهمیت ویژه ای پیدا می کند. با ارائه یک الگوریتم فرا ابتکاری، می توان مسائل QAP با ابعاد بزرگ را به جای استفاده از روش های غیرخطی[۲۲] و با تبدیل روش های غیر خطی به خطی[۲۳] مانند توابع ماکس-مین[۲۴] با نرم افزار های مناسب مانند GAMS در زمان کوتاه تر حل کنیم. بنابراین ایده گرفتن از الگوی قانون گرانش جهت طراحی الگوریتم، برای حل مساله QAP یک ابتکار و نوآوری است که عملا سرعت رسیدن به جواب بهینه تقریبی را افزایش می دهد.

۱-٦ خلاصه و فصول بعدی پژوهش

در فصل دوم، مروری بر مبانی، و تحقیقات پیشین مرتبط صورت می گیرد، ابتدا مسائل بهینه سازی و روش های حل آن ها به طور کلی مطرح، و موارد مربوط به پیچیدگی مسائل تشریح می گردد. آنگاه تاریخچه پیدایش و روش های حل مساله تخصیص درجه دوم مطرح می گردد. در فصل سوم، مساله روش مبنا(الگوریتم جستجوی گرانشی) که تحقیق مبتنی بر آن تکامل می‌یابد توصیف شده و الگوریتم پیشنهادی برای حل مساله QAP ارائه می شود. در فصل چهارم نتایج محاسباتی حاصل از اجرای الگوریتم، ارائه شده است. و در نهایت در فصل پنجم به نتیجه گیری و ارائه پیشنهادهایی برای پژوهش های آینده می پردازیم.

[22] Nonlinear
[23] Linear
[24] Maxi-Min

فصل دوم :مروری بر مبانی و پیشینه تحقیق

۲-۱ مقدمه

محاسبه راه حل های بهینه برای اکثر مسائل بهینه سازی که در خیلی از زمینه های کاربردی و عملی مشاهده می گردند کاری دشوار و سخت است. در عمل معمولا به راه حل های خوب که از الگوریتم های ابتکاری یا فرا ابتکاری به دست می آید اکتفا می گردد. فرا ابتکاری ها مجموعه ای از فنون بهینه سازی تقریبی را که عمدتا در طول دو دهه گذشته شهرت پیدا کرده اند در بر می گیرند. روش های فرا ابتکاری راه حل های قابل قبول در زمان مقعول را برای مسائل NP-hard در زمینه های مهندسی و علوم پایه ارائه می کند. برخلاف الگوریتم های بهینه سازی دقیق، فرا ابتکاری ها، بهینه بودن جواب های بدست آمده را ضمانت نمی کنند.

کلمه هیورستیک از کلمه یونانی "هیورسکین"[۲۵] به معنای "هنر کشف قواعد جدید برای حل مسائل" گرفته شده است. پیشوند "متا" نیز از یک کلمه یونانی به معنای "متدولوژی سطح بالا" گرفته شده است. واژه متاهیورستیک اولین بار توسط گلاور[۲۶] در سال ۱۹۸٦ ارائه گردید. [۵]

۲-۲ مدل های بهینه سازی

یک مساله بهینه سازی ممکن است به صورت زوج مرتب (s,f) تعریف گردد که در آن s معرف مجموعه راه حل های امکان پذیر(با فضای پیکرده بندی و فضای جستجو) و $f : s$ تابع هدفی است که باید بهینه شود. تابع هدف به هر نقطه از فضای جستجو یک عدد حقیقی که نشان دهنده ارزش آن است اختصاص می دهد. هدف اصلی از حل یک مساله بهینه سازی یافتن راه حل بهینه سراسری[۲۷] است. یک راه حل $s^* \in S$ یک راه حل بهینه سراسری است اگر مقدار تابع هدف بهتری نسبت به تمامی راه حل های فضای جستجو داشته باشد. ممکن است

[25] Heuriskein
[26] F.Glover
[27] Global optimal solution

برای یک مساله چندین راه حل بهینه سراسری وجود داشته باشد و ممکن است در بعضی موارد به صورت یافتن تمامی راه حل های بهینه سراسری تعریف گردد.

یکی از مدل های رایج در برنامه ریزی ریاضی، برنامه ریزی خطی[۲۸] است که می تواند به صورت زیر فرموله گردد:

$$\begin{aligned} &min: \quad c.x \qquad (1\text{-}2)\\ &S.t. \quad A.x \geq b\\ &\qquad x \geq 0 \end{aligned}$$

که در آن x متغییر های پیوسته و یا گسسته تصمیم بوده و A وb ، c و ماتریس ضرایب(پارامترها) ثابت می باشند. هدف و محدویت های یک مساله برنامه ریزی خطی، همگی از نوع خطی می باشند. برای حل این مدل روش های دقیق یافتن جواب بهینه از قبیل روش سیمپلکس وجود دارد.

مدل برنامه ریزی غیری خطی[۲۹] مدل های برنامه ریزی ریاضی می باشند که در آن تابع هدف با تعدادی از محدودیت ها به شکل غیر خطی می باشند. بعضی از روش های مدل سازی وجود دارند که می تواند بعضی از روابط غیر خطی را به خطی تبدیل کرد. در این حالت متغییر ها و محدودیت های زیادی به مدل اضافه می گردد و ممکن است درجاتی از تقریب نیز استفاده شود. برای مدل های بهینه سازی غیر خطی ممکن است از برخی مدل های هیورستیک الهام گرفته شده از روش سیمپلکس مانند نلدر و مید استفاده شود.

با وجودی که بسیاری از الگوریتم های بهینه سازی برای مسائل پیوسته نسبت به مسائل گسسته توسعه پیدا گرده اند مسائل زیادی در دنیای واقعی وجود دارد که باید از متغییرهای گسسته برای مدل سازی آنها استفاده گردد.

زمانی که متغییر های تصمیم یک مدل هم متغییر های پیوسته و متغییرهای گسسته را در برداشته باشد مساله در گروه مسائل برنامه ریزی عدد صحیح ترکیبی[۳۰] قرار می گیرد. مدل های مذکور تعمیم یافته مدل های برنامه ریزی خطی و برنامه ریزی عدد صحیح[۳۱] می باشند. برای حل این دو مدل در اندازه های کوچک می توان از روش های شمارشی از قبیل شاخه و کران[۳۲] استفاده کرد. متاهیورستیک ها یکی از الگوریتم های مناسب برای حل این

[28] linear programming
[29] nonlinear programming models
[30] mixed integer programming problems
[31] integer programming
[32] Branch and bound

نوع مسائل در اندازه هایی که حل آنها با استفاده از روش های دقیق خیلی پیچیده به نظر می رسند، می باشند و جواب های خوبی برای آنها تولید می کنند.

شکل تعمیم یافته تر از مسائل برنامه ریزی عدد صحیح مسائل بهینه سازی ترکیبی است. این دسته از مسائل با متغییرهای تصمیم گسسته و فضای جستجوی محدود شناسایی می گردند. با این وجود تابع هدف با محدویت های آنها ممکن است شکل های مختلفی به خود بگیرند. [۵]

۲-۲-۱ پیچیدگی مسائل

پیچیدگی یک مساله برابر با پیچیدگی بهترین الگوریتم برای حل آن مساله تعریف می گردد. یک مساله ساده مسحوب می گردد اگر یک الگوریتم چند جمله ای زمانی برای حل آن وجود داشته باشد و یک مساله دشوار محسوب می گردد اگر هیچ الگوریتم چند جمله ای زمانی برای حل آن وجود نداشته باشد.

یکی از جنبه های مهم تئوری محاسباتی طبقه بندی مسائل به رده بندی پیچیدگی است. یک رده پیچیدگی مجموعه ای از مسائل را نشان می هد که با مقدار مشخصی از منابع محاسباتی قابل حل باشند. دو رده مهم از مسائل P و NP می باشند.

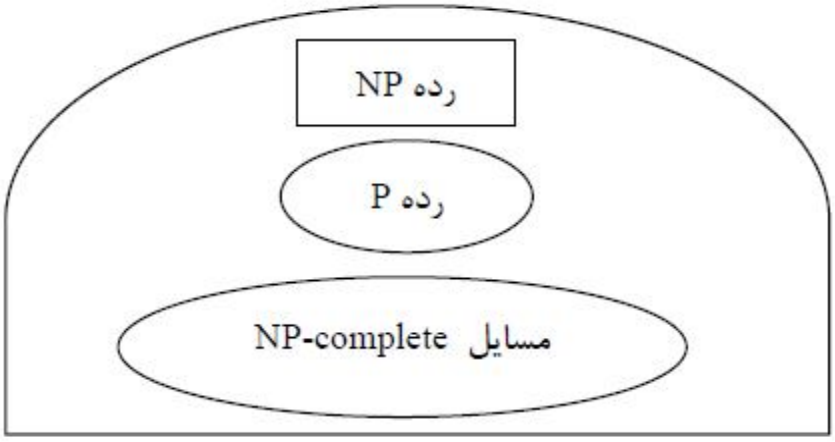
پیچیدگی رده P مجوعه ای از تمامی مسائل تصمیم گیری را نشان میدهد که توسط یک الگوریتم قطعی در زمان چند جمله ای قابل حل باشد. یک الگوریتم برای حل یک مساله A چند جمله ای محسوب می گردد اگر در بدترین پیچیدگی آن محدود به تابع چندجمله ای $f(n)$ باشد که n اندازه ورودی مساله است. مسائل در در رده P جزو مسائل ساده برای حل نیز محسوب می گردد. مانند مسائل حداقل درخت پوشاننده[۳۳]، مسائل کوتاهترین مسیر، شبکه با حداکثر جریان و مدل های پیوسته برنامه ریزی خطی مثال هایی از مسائل در رده P محسوب می گردند. یکی از الگوریتم های موثر برای حل مسائل برنامه ریزی خطی الگوریتم سیمپلکس می باشد که خود الگوریتم سیمپلکس دارای پیچیدگی نمایی است.

پیچیدگی رده NP مجموعه تمامی مسائل تصمیم گیری را نشان می دهد که توسط یک الگوریتم غیر قطعی(ترکیبی) در زمان چند جمله ای قابل حل باشند. واژه NP از nondeterministic polynomial-time گرفته شده است. رده NP اکثر مسائل بهینه سازی ترکیبی و همچنین تمامی مسائل موجود در رده P را در بر می

[33] Minimum spanning tree

گیرد. بنابراین $P \subseteq NP$. رده NP طبقه ای از مسائل تصمیم را در بر می گیرد که پاسخ مثبت به مساله آن در زمان کوتاهی محدود به زمان چند جمله ای قابل بررسی می باشد. نشان داده شده است که اغلب مسائل زمانبندی که به عنوان یک مساله تصمیم بررسی می شوند، متعلق به طبقه NP می باشند.

در داخل رده NP مسائلی از رده NP – complete وجود دارد. یک مساله تصمیم $A \in NP$، NP – complete محسوب می گردد اگر تمامی مسائل رده NP در زمان چندجمله ای قابل کاهش به مساله A باشند. شکل ۲-۱ رابطه بین P، NP، NP – complete را نمایش می دهد.



شکل ۲-۱ – رده های پیچیدگی مسائل [۵]

مسائل دارای پیچیدگی نمایی مسائل بهینه سازی می باشند که مسائل تصمیم مرتبط با آنها از نوع NP – complete باشند. اغلب مسائل بهینه سازی در دنیای واقعی از نوع مسائل دارای پیچیدگی زمانی می باشند و الگوریتم کارای اثبات شده ای برای آنها وجود ندارد. زمان حل بهینه آنها در رده زمان نمایی می باشد(به غیر از حالتی که P=NP باشد) متاهیورستیک ها گزینه مناسب و با اهمیتی برای حل این دسته از مسائل می باشند. در زیر دسته ای از مسائل دارای پیچیدگی نمایی آورده شده است:

۱. مسائل زمان بندی و تعیین توالی مانند زمان بندی جریان کارگاهی، زمان بندی تولید کارگاهی و مسائل زمان بندی کارگاهی باز.

۲. مسائل تخصیص و جانمایی از قبیل مساله تخصیص درجه دوم، مساله تخصیص تعمیم یافته[۳٤]، جانمایی و تسهیل ها، مساله P-median

۳. مسائل گروه بندی مانند خوشه بندی داده ها[۳۵]، افراز گراف[۳٦] و رنگ آمیزی گراف[۳۷].

٤. مسائل مسیریابی و پوشش دهی از قبیل مساله مسیریابی وسایل حمل و نقل[۳۸] (VRP) مسائل پوشش مجموعه[۳۹] و مسائل تور پوشاننده.

۵. مسائل برش، بسته بندی کوله پشتی و غیره. [۵]

۲-۲-۲ پیچیدگی محاسباتی

نتایج حاصل در این بخش بیانگر شواهدی است که اثبات می کند QAP جز مسائل NP-hard است. از نقطه نظر تئوریک نه تنها QAP نمی تواند به طور کارآمد حل شود بلکه حتی نمی تواند به طور تقریبی نیز به صورت کارآمد با برخی "ثابت های نسبت تخمینی"[٤۰] حل شود. بعلاوه یافتن بهینه موضعی حتی برای همسایگی های ساخت یافته ای مثل همسایگی 2-opt یک عمل بیهوده نیست. پیش از این در سال ۱۹۷٦ دو نتیجه بوسیله سحنی[٤۱] و گونزالز[٤۲] در رابطه با پیچیدگی حل و تقریب QAP بدست آمده بود. آنها نشان داده اند که در حالتی که پایش یک الگوریتم با تقریب ε چند جمله ای بر P=NP دلالت کند، QAP یک مساله با پیچیدگی نمایی است و حتی یافتن یک جواب با تقریب ε برای آن یک مساله NP-hard است. برکارد[٤۳]، چلا[٤٤]، پاردالوس[٤۵] و پیتسولیس[٤٦] در مقاله ۷۰ صفحه ای که در رساله بهینه سازی محاسباتی و توسط انتشارات آکادمی کلوور[٤۷] منتشر

[34] Generalized assignment problem
[35] Data clustering
[36] Graph partitioning
[37] Graph coloring
[38] Vehicle routing problem
[39] Set Covering Problems
[40] Constant approximating ratio
[41] Sahni
[42] Gonzalez
[43] Burkard
[44] çela
[45] Pardalos
[46] Pitsolis
[47] Kluwer Academic Publishers

شده است اثبات نموده اند که QAP یک مساه NP-hard می باشد و در رده مسائل دارای پیچیدگی نمایی قرار می گیرد. [۷]

لذا با توجه به موارد مطرح شده این مساله در ردیف مسائل با پیچیدگی نمایی است، لذا برای حل آن باید از یک الگوریتم فرا ابتکاری استفاده کنیم که در این پژوهش الگوریتم جستجوی گرانشی ترکیبی انتخاب گردیده است. در فصول بعدی به تفصیل به این الگوریتم خواهیم پرداخت.

۲-۳ روش های بهینه سازی

دو دسته کلی از روش های حل مسائل بهینه سازی شامل روش های دقیق و روش های تقریبی وجود دارد. روش های دقیق راه حل های بهینه را به دست آورده و شرایط بهینگی را تضمین می کنند. روش های تقریبی(ابتکاری) راه حل های با کیفیت بالا را در زمان معقولی تولید می کنند، اما تضمینی برای یافتن راه حل بهینه سراسری ندارند.

۲-۳-۱ روش های دقیق

روش های دقیق الگوریتم هایی هستند که جواب های بهینه را پیدا نموده و شرایط بهنیگی را تضمین می کنند. بعضی از این الگوریتم ها عبارتند از برنامه ریزی پویا و الگوریتم های خانواده شاخه و کران.

روش های دقیق برای ابعاد کوچک بعضی از مسائل بهینه سازی دشوار نیز قابل به کار گیری می باشند. اگرچه ابعاد مساله شاخص واحدی برای تشریح سختی یک مساله نیست. برای یک مساله ممکن است جبعادی از مساله قابل حل با روش های دقیق نباشند در حالی که ابعاد بزرگتر آن توسط روش های دقیق حل شده باشد. [۳]

۲-۳-۲ راه حل های ابتکاری

ابتکاری ها راه حل های خوب را برای مسائل با اندازه های بزرگ جستجو می کنند. آنها عملکرد قابل قبولی را همراه با هزینه قابل قبولی برای حوزه وسیعی از مسائل به همراه دارند. ابتکاری ها به دو خانواده تقیسم بندی می شوند: ابتکاری های خاص و فرا ابتکاری ها.

۲-۳-۲-۱ ابتکاری های خاص

ابتکاری های خاص برای حل یک مساله خاص طراحی شده اند. فرا ابتکاری ها، الگوریتم های عمومی می باشند که برای حل اغلب مسائل بهینه سازی قابل استفاده می باشند. آنها را می توان به عنوان متدولوژی های سطح بالای عمومی دید که به عنوان استراتژی راهنما برای طراحی ابتکاری های خاص و برای مسائل بهینه سازی خاص به کار روند.

۲-۳-۲-۲ فرا ابتکاری ها

برخلاف روش های دقیق، فرا ابتکاری ها برای مسائل با اندازه های بزرگ کاربرد داشته و راه حل راضی کننده ای در زمان معقولی ارائه می کنند. در این الگوریتم ها هیچ گونه ضمانتی برای یافتن جواب بهینه سراسری یا حدودی از آن وجود ندارد. فرا ابتکاری ها در طول بیست سال گذشته شهرت رو به افزونی داشته اند. کاربرد و استفاده از آنها در خیلی از مسائل کارایی و اثربخشی آنها را برای حل مسائل پیچیده و بزرگ نشان می دهد.[۵]

در طراحی یک الگوریتم فرا ابتکاری دو معیار متناقض شامل کاوش[۴۸] در فضای جستجو(گوناگونی و تنوع) و تبعیت[۴۹] از بهترین راه حل های پیدا شده(تشدید یا بهره برداری) باید در نظر گرفته شوند. در تشدید، عملیات کاوش بیشتر در محدوده امیدبخش (محدوده ای که در آن بهترین راه حل پیدا شده است) نسبت به کل فضا انجام می گردد. در گوناگونی منطقه های کاوش نشده مورد جستجو قرار می گیرد و هدف این است که اکثر قسمت های فضای جستجو مورد کاوش قرار گیرد. در صورتی که به کاوش اهمیت بیشتری داده شود الگوریتم به سمت رفتار تصادفی میل خواهد کرد و در صورتی که به رفتار تبعیت از راه حل های خوب توجه بیشتری شود، الگوریتم از حالت تصادفی فاصله گرفته و جستجو تنها در محدوده راه حل های خوب انجام می گردد.

معیارهای طبقه بندی زیادی ممکن است برای طبقه بندی فرا ابتکاری ها استفاده شود که در زیر به بعضی از آنها اشناره می شود:[۲۹]

[48] Exploration
[49] Exploitation

۱. **الهام گرفته شده از طبیعت در مقابل عدم الهام از طبیعت.** خیلی از فرا ابتکاری ها از فرایندهای طبیعی الهام گرفته شده اند، از قبیل الگوریتم های تکاملی[۵۰] و سیستم ایمنی مصنوعی[۵۱] از بیولوژی، کلونی مورچه ها[۵۲] و زنبوران عسل از هوش ذرات(گروهی) گونه های مختلف طبیعی از قبیل مورچه ها و زنبورها و انجماد تدریجی.

۲. **روش هایی که از حافظه استفاده می کنند در مقابل روش های بدون حافظه.** بعضی از الگوریتم های فرا ابتکاری از قبیل روش های جستجوی تطبیقی تصادفی حریصانه[۵۳] و انجماد تدریجی بدون حافظه می باشند. در حالی که دسته دیگر برای مثال جستجوی ممنوعه از یک حافظه که بعضی از اطلاعات را در طول جستجو ذخیره می کند، استفاده می کنند.

۳. **قطعی در مقابل احتمالی.** یک الگوریتم قطعی یک مساله بهینه سازی را از طریق تصمیم گیری قطعی حل می کند. (برای مثال، جستجوی محلی[۵۴] و جستجوی ممنوعه) در فرا ابتکاری های احتمالی بعضی از قواعد احتمالی در فرایند جستجو مورد استافده قرار می گیرد. (مثل الگوریتم انجماد تدریجی یا الگوریتم های تکاملی)

٤. **جستجوی مبتنی بر جمعیت در مقابل جستجوی مبتنی بر یک راه حل تکی.** الگوریتم های مبتنی بر راه حل تکی از قبیل انجماد تدریجی یک راه حل اولیه را مورد استفاده قرار داده و آن را در طول فرایند جستجو تغییر می دهند، در حالی که الگوریتم های مبتنی بر جمعیت از قبیل الگوریتم ژنتیک یک جمعیت از راه حل ها را مورد استفاده قرار داده و در طول فرایند جستجو آنها را تغییر می دهند.

۵. **تکراری در مثابل حریصانه.** در الگوریتم های تکراری الگوریتم با یک راه حل (یا جمعیتی را راه حل ها) کامل شروع شده و در هر تکرار راه حل را راه حل ها تغییر پیدا می کنند. در الگوریتم های حریصانه یک راه حل کامل در اختیار نبوده بالکه با یک راه حل خالی (ساخته نشده) شروع شده و در هر مرحله یک متغییر تصمیم از مساله یک قسمت از راه حل را می سازد. اغلب فرا ابتکاری ها، الگوریتم های تکراری می باشند.

٦. **مسائل بهینه سازی با توابع هدف با محدودیت های زمان بر**

۷. **مدل های غیر تحلیلی از مسائل بهینه سازی که با یک روش جامع قابل حل نیستند.**

۸. **مسائل طراحی**

---

[50] Evolutionary Algorithms
[51] Artificial Immune System
[52] Ants Colony
[53] Greedy Randomized Adaptive Search
[54] Local Search

۹. مسائل کنترل

هر فرا ابتکاری تکراری نیاز به ساختاری برای نمایش(کد گذاری) راه حل ها دارد. از طرفی معمولا الگوریتم های فرا ابتکاری محاسبات زیادی را به همراه داشته و الزاما باید با استفاده از رایانه حل گردند. به منظور برنامه نویسی الگوریتم لازم است تا راه حل مساله به صورت کدهای قابل فهم توسط رایانه بیان گردد. روش های مختلفی برای کد گذاری وجود دارد که در زیر تعدادی از آنها شرح داده می شود.

۲-٤ مساله تخصیص درجه دوم

۲-٤-۱ سابقه تاریخی

به مساله تخصیص تسهیل ها [۵۵] به مکان ها[۵۶]، اگر فاصله بین مکان ها، تقاضای جریان بین تسهیل ها و به طور کلی هزینه های تخصیص تسهیل به مکان مشخص باشد، به صورتی که هر تسهیل فقط به یک مکان و هر مکان فقط به یک تسهیل اختصاص یابد، توجه کنید. متون بین المللی مساله تخصیص درجه دوم را (QAP) به عنوان یک مساله برای یافتن حداقل هزینه تخصیص یافتن تسهیل ها به مکان ها تعریف می کند. کوپمن و بکمن[۵۷] در سال ۱۹۵۷ برای اولین بار QAP را به عنوان یک مدل ریاضی مربوط به فعالیت های اقتصادی معرفی کردند. از آن زمان تا کنون این مساله کاربردهای عملی متعددی یافته است. اشتانبرگ[۵۸] در سال ۱۹٦۱ مساله QAP را برای حداقل سازی تعداد اتصالات بین اجزا در سیم کشی پشت مدار به کار برد. هافلی[۵۹] آن را برای مسائل اقتصادی استفاده کرد. فرانسیس و وایت[٦۰] یک چارچوب تصمیم گیری برای تخصیص یک تسهیل جدید(مثل مکان پست های پلیس، سوپر مارکت ها، مدارس) به منظور خدمت رسانی به تعداد مشخصی از ارباب رجوع یا مشتریان را توسعه داد. جفوفیون و گراویز[٦۱] بر مساله برنامه ریزی زمانی متمرکز شد. [۸]

---

55 Facilities
56 Locations
57 Koopman and Beckman
58 Steinberg
59 Heffley
60 Francis and White
61 Geoffrion and Graves

پولاشتک[۶۲] و همکاران (۱۹۷۶) از QAP برای تعریف بهترین طراحی جهت صفحه کلید تایپیست ها و کنترل پنل ها استفاده نمودند. کراراپ و پروزان[۶۳] (۱۹۷۸) از آن برای باستان شناسی استفاده نمود. هوبرت[۶۴] (۱۹۸۷) در تحلیل آماری فورسبرگ[۶۵] و همکاران(۱۹۹٤) در تحلیل شیمیایی واکنش ها و براسکو و اشتال[۶۶] (۲۰۰۰) در تحلیل عددی از آن استفاده نمودند. با این حال مساله جانمایی تسهیل ها مشهورترین کاربرد QAP می باشد:دیکی و هاپکینز[۶۷] (۱۹۷۲) از QAP برای تخصیص زمین برای ساخت ساختمان های یک دانشگاه استفاده کردند. الشافی[۶۸] (۱۹۷۷) در برنامه ریزی یک بیمارستان و باس[۶۹] (۱۹۹۳) در یک مساله مربوط به پارک های جنگلی، بن جعفر[۷۰](۲۰۰۲) فرمولی را برای مساله طراحی جانمایی تسهیل به منظور حداقل سازی کار در جریان ارائه نمود. در کار او، وی نشان داد که جانمایی های بدست آمده با استفاده از یک فرمول مبتنی بر کار در جریان (WIP)می تواند بسیار متفاوت از کارهایی باشد که با استفاده از فرمول QAP قرار داده شده انجام شده باشد. برای مثال یک جانمایی بهینه QAP می تواند یک کار در حال انجام، نشدنی باشد. رابک و سیشمن[۷۱] (۲۰۰۳)، میرندا[۷۲] و همکاران (۲۰۰۵) و دومان و ایلحان[۷۳] (۲۰۰۵) برای استقرار اجزا الکترونیکی از آن استفاده نموده اند.

وس و زایتهوفر[۷٤] (۲۰۰٤) مساله بهینه سازی جانمایی حافظه را در پردازنده های سیگنالی بررسی کردند. پژوهش های بسیار زیادی در رابطه با این مقاله انجام شده که شرح آن در رفرنس [۱۰] آمده است. [۸]

سیریات مونن و اینچاکول و پیرایوت چارنستیکول[۷۵] (۲۰۰۷) الگوریتمی ارائه کردند که از برنامه ریزی پویا، تجزیه بندرس و روش های[۷۶] فرا ابتکاری را برای حل مسائل جانمایی تسهیل پویا[۷۷] استفاده کند. این مساله به عنوان یک مدل بسط داده شده مساله تخصیص درجه دوم که مساله تخصیص درجه دوم پویا[۷۸] نامیده می شود حل شده است. [۹] محمد سیف الله حسین و توماس اشتودسل[۷۹] در سال ۲۰۰۹ از روش جستجوی محلی

---

62 pollatsheck
63 Krarap and Pruzan
64 Hubert
65 Forsberg
66 Brusco and Stahl
67 Dicky and Hopkins
68 Elshafei
69 Bos
70 Benjaafar
71 Rabak and Sichman
72 Miranda
73 Duman and Ilhan
74 Wess and Zeitlhofer
75 Sirirat moenvanichakul and Peerayuth charnsethikul
76 Benders decomposition
77 Dynamic facility location
78 Dynamic quadratic assignment problem
79 Mohamed saifullah Hussin and Thomas Stutzle

تکراری سلسله مراتبی برای حل مساله QAP استفاده نمودند. [۱۱] هویی لی و داریو لاندا سیلوا[۸۰] از یک روش فرا ابتکاری جستجوی تطبیقی تصادفی حریصانه نخبه برای حل مسائل تخصیص درجه دوم چند هدفه استفاده نمودند. [۱۲] آرتور آلوز پسوا و سایرین[۸۱] در سال ۲۰۱۰ از ترکیب روش های تجزیه لاگرانژی و فرموله کردن مجدد خطی برای حل مساله تخصیص تعمیم یافته استفاده نمودند. [۱۳] رامکومار و سایرین[۸۲] (۲۰۰۹) برای حل مساله تخصیص درجه دوم در بحث طراحی جانمایی تسهیل ها از یک روش ابتکاری جستجوی محلوی سریع تکراری استفاده نمودند. آنها اقدام به اصلاح این روش با استفاده از ترکیب مجدد جدیدی از عملگر تقاطع کردند. [۱٤] در سال ۲۰۰۱ آرکین و همکاران[۸۳] تقریبی جهت حداکثر سلزی مساله تخصیص درجه دوم ارائه نمودند. [۱۵] زیوفتن یانگ و همکاران[۸٤] در سال ۲۰۰۸ مساله QAP را تحت مدل آدلمان- لیپتون –استیکر[۸۵] حل کردند. آنها یک الگوریتم DNA کارا برای حل QAP ارائه نمودند. [۱٦] هوی ژن ژانگ و همکاران[۸٦] (۲۰۰۱) به مساله کاهش فرمولی مساله QAP تحت فرمول مشهور بنامه ریزی خطی عدد صحیح آدامز و جانسون[۸۷] پرداختند. آنها نتایج کار خود را با حل ۳۰ مثال از QAPLB در ابعاد ۱۲ تا ۳۲ بیان نمودند. [۱۷] دمیرل و توکساری[۸۸] (۲۰۰٦) از الگوریتم کلونی مورچه برای حل مساله تخصیص درجه دوم استفاده کردند. [۱۸] لین ژونگ و بین ژنگ لی[۸۹] (۲۰۰٦) با استفاده از مدل های جدید و الگوریتم ژنتیک مساله تخصیص درجه دوم فازی را حل نمود. [۱۹] اوزبکیر و همکاران[۹۰] (۲۰۱۰)از الگوریتم جستجوی غذای زنبور عسل برای حل مساله QAP استفاده نمودند. [۲۰]

از زمان ارائه فرمول اولیه این مساله QAP در سرتاسر جهان مورد توجه پژوهشگران واقغ شد، نه تنها به خاطر عملی بودن آن و اهمیت تئوریک، بلکه به خاطر پیچیدگی آن. مساله تخصیص درجه دوم یک از سخت ترین مسائل بهینه سازی ترکیباتی می باشد. به طور کلی مثال های با ابعاد $n>30$ در زمان معقولی قابل حل نمی باشند. [۱۰]

لازم به توضیح است که در برخی متون این موضوع برای مسائل با بعد $n>20$ برای حل QAP عنوان شده است.

---

80 Hui Li and Dario Landa-Silva
81 Artur Alves Pessoa et al
82 Ramkumar et al
83 Arkin et al
84 Xiaofan Yang et al
85 Adleman- Lipton – Sticker model
86 Huizheng Zhang et al
87 Adams and Johnson
88 Demirel and Toksari
89 Linzhong Liu and Yinzheng Li
90 Ozbakir et al

سحنی و گونزالز (۱۹۷٦) نشان دادند که QAP یک مساله دارای پیچیدگی نمایی است و P=NP ممکن نیست تا یک الگوریتم تقریب زننده f برای ثابت f ارائه شود. چنین مسائلی تا زمانی معتبر هستند که جریان ها و فاصله ها به صورت ماتریس های ضرایب سیمتریک (متقارن)[۹۱] ظاهر شوند. بعلت پیچیدگی محاسبه بالا QAP به عنوان اولین کاربرد آزمایشی اصلی خود برای پروژه GRIBB[۹۲] انتخاب شد. این پروژه به دنبال ایجاد یک کتابخانه نرم افزاری برای حل مسائل جستجوی موازی ابعاد بزرگ با استفاده از تعداد زیادی کامپیوتر در سرتاسر جهان با دسترسی به اینترنت می باشد. نتایج اولیه آزمایش در مقاله Moe (۲۰۰۳) منتشر شده است.

چندین مساله بهینه سازی ترکیبی دارای پیچیدگی نمایی از قبیل مساله فروشنده دوره گرد، مساله بسته بندی سطلی و مساله بزرگترین خوشه می تواند به صورت QAP مدل شوند. جستجو برای بهینه سازی محلی در مثال های کلاسیک در دسترس در اینترنت ما را برای مقایسه عملکردهای تکنیکی مجاز می کند، تا زمانی که نقطه بهینه ناشناخته باشد یا زمانی که به کار گیری الگوریتم های دقیق در این مسائل میسر باشد، برکارد و همکاران(۱۹۹٦) و و چلا (۱۹۹۸).

کارشناسان مسائل بهینه سازی ترکیباتی تمایل دارند تا نسخه های قابل حلی از مسائل چند جمله ای خاص را که دارای پیچیدگی نمایی باشند را جستجو کرده و مکانیسم های تحقیق برای سنجش سختی مسائل طرح شده را پژوهش کنند. در رابطه با QAP کریستوفیدس و جرارد[۹۳] (۱۹۸۱) برخی مسائل خاص را بررسی نمودند. سیلا و بابا[۹٤] (۱۹۸۷)یک متدولوژی برای یک مساله تخصیص درجه دو منظم توسعه دادند. چن[۹۵] (۱۹۹۵)دیگر حالت های QAP را مطرح نمود که بوسیله چلا(۱۹۹۸) ادامه یافت و وی چندین مثال قابل حل چند جمله ای را ارائه نمود. هروالرن و ونگلیس[۹٦] (۱۹۸۵)سیگانسکی و همکاران[۹۷] (۱۹۹٤)، ماتور و روکایرول[۹۸] (۱۹۹٤) نشان دادند که مثال های QAP پالوبتسکی[۹۹] منحوط هستند.

91 Symmetric Coefficient Matrices
92 Great International Branch And Bound Search
93 Christifides and Gerrard
94 Sylla and Baba
95 Chen
96 Herroeleven and Vanglis
97 Cyganski et al
98 Roucairol and Mautor
99 Palubetski

۲-٤-۲ مدل عمومی مساله تخصیص درجه دوم

فرموله بندی های زیادی برای این مساله ارائه شده است. شامل فرمولاسیون هایی براساس برنامه ریزی بولی به دنبال برنامه ریزی خطی عدد صحیح، برنامه ریزی عدد صحیح مختلط به دنبال برنامه ریزی خطی، فرمولاسیون ها بر اساس جایگشت، فرمول اثر(رد)، برنامه نویسی نیمه قطی و فرمولاسیون بر اساس نمودار.

۲-٤-۲-۱ فرمول بندی براساس مدل کوپمنز و بکمان

با فرض وجود مجموعه $N = \{1,2,\dots,n\}$ و ماتریس های $D = d_{kl}$و $F = f_{ij}$ و $C = c_{ik}$ با اندازه $n \times n$، مساله تخصیص درجه دوم براساس مدل کوپمنز-بکمان به صورت زیر بیان می شود:

$$\min_{p \in \pi_N} \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}\, d_{p(i)p(j)} + \sum_{i=1}^{n} c_{p(i)i} \qquad (۲-۲)$$

که در آن π_N مجموعه تمام جایگشت های N می باشددر واقع n! جایگشت امکان پذیر است. و راه حل بهینه، بهترین جایگشت است.

ماتریس F ماتریس جریان می باشد. برای مثال f_{ij} جریان بین تسهیلات i و j است و ماتریس D ماتریس فاصله می باشد، برای مثال d_{kl} فاصله بین مکان k و مکان l می باشد. و ماتریس C، ماتریس هزینه است برای مثال c_{ik} هزینه جایگیری تسهیل i در مکان k می باشد. هدف، یافتن یک تخصیص از تمام تسهیل ها به تمامی مکان ها است که در آن هزینه کل تخصیص حداقل شود.

۲-٤-۲-۲ فرمول بندی برنامه ریزی خطی عدد صحیح

فرمول برنامه ریزی خطی عدد صحیح (IP) ابتدا توسط کوپمن و بکمن در سال ۱۹۵۷ پیشنهاد شد. کار بر روی این فرمول به وسیله یو و سارکر[100] [۲۵]و در نهایت فدجکی و دافا[101] [۲٦] ادامه یافت. این فرمول برای یک مساله QAP با اندازه n به شرح زیر است:

[100] Yu and Sarker
[101] Fedjki and Duffaa

$$min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{p=1}^{n} f_{ij} d_{kp}\, x_{ik} x_{jp} \qquad (3-2)$$

$$s.t. \quad \sum_{i=1}^{n} x_{ij} = 1 \ , \quad 1 \le j \le n \qquad (4-2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \ , \quad 1 \le i \le n \qquad (5-2)$$

$$x_{ij} \in \{0 , 1\} \ , \quad 1 \le i, j \le n \qquad (6-2)$$

که در آن f_{ij} جریان بین تسهیلات i و j است و d_{kp} فاصله بین مکان k و مکان p می باشد. [٤]

که اگر هزینه تخصیص تسهیلات به مکان ها را نیز در نظر بگیریم مدل عمومی QAP به صورت زیر می باشد:

$$min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{p=1}^{n} f_{ij} d_{kp}\, x_{ik} x_{jp} + \sum_{i=1}^{n}\sum_{k=1}^{n} C_{ik}\, x_{ik} \qquad (7-2)$$

$$s.t. \quad \sum_{i=1}^{n} x_{ij} = 1 \ , \quad 1 \le j \le n \qquad (8-2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \ , \quad 1 \le i \le n \qquad (9-2)$$

$$x_{ij} \in \{0 , 1\} \ , \quad 1 \le i, j \le n \qquad (10-2)$$

۲-٤-۳ روش های حل مساله تخصیص درجه دوم

۲-٤-۳-۱ روش های دقیق

در این روش ها یک الگوریتم برای یک مساله بهینه سازی ترکیبی جواب بهینه کلی برای مساله ارائه می کند. برخی الکوریتم های دقیق شامل شاخه و کران، صفحات برش دهنده و الگوریتم صفحات برشی چند وجهی می باشند. [٤]

۲-٤-۳-۱-۱ الگوریتم شاخه و کران

الگوریتم شاخه و کران به طور موفق برای بسیاری از مسائل بهینه سازی ترکیبی NP-hard استفاده شده است و نشان داده است که یکی از کاراترین الگوریتم های دقیق برای حل QAP می باشد. یکی از اجزاء اولیه الگوریتم های شاخه و کران، کران دار کردن، شاخه سازی، و قاعده انتخاب می باشد. اگرچه بسیاری از روش های کران دار کردن برای QAP توسعه یافته اند، کاراترین الگوریتم های شاخه و کران برای این مساله به کار گیری الگوریتم GLB[۱۰۲] است. [٤]

علت آن است که سایر کران ها که با GLB به نتایج بهتری دست پیدا می کند، با توجه به زمان محاسباتی مقرون به صرفه نمی باشند.

سه نوع استراتژی شاخه سازی اغلب در روش QAP استغاده می شوند:

۱. روش شاخه سازی تخصیص تکی[۱۰۳] ارائه شده به وسیله گیلمور و لاولر.
۲. روش شاخه سازی تخصیص زوجی[۱۰٤] ارائه شده به وسیله گاوت و پلیتر، لند، ناگنت و همکاران[۱۰۵].
۳. روش ساخه سازی براساس موقعیت یابی نسبی[۱۰٦] ارائه شده به وسیله میر چندانی و اوباتا[۱۰۷].

روش شاخه سازی تخصیص تکی که کاراترین روش می باشد یک تسهیل را به یک مکان در هر مرحله شاخه سازی تخصیص می دهد. بدین صورت که با ثابت نگه داشتن مکان یکی از تسهیل ها که هنوز تخصیص نیافته است. هر مساله به زیر مساله هایی تجزیه می شود. قاعده های متعددی برای تعیین زیر مساله های یک سطح

[102] Gilmore-Lawler Bound
[103] single assignment branching
[104] pair assignment branching
[105] Gavett and Plyter, Land and Nugent et al
[106] Relative positioning
[107] Mirchandani and Obata

جدید از درخت جستجو که به وسیله نویسندگان مختلف ارائه شده است وجود دارند که برای انتخاب جفت تسهیل-مکان مورد استفاده قرار می گیرد.

الگوریتم های تخصیص زوجی، زوجی از تسهیل ها به زوجی از مکان ها را در یک گام شاخه سازی اختصاص می دهند. در حالی که در الگوریتم موقعیت یابی نسبی سطوح درخت جستجو وابسته به تعداد تسهیل های تخصیص یافته به مکان ها نمی باشند. در اینجا تخصیص ثابت در هر زیر مساله با توجه به فواصل بین تسهیل ها تعیین می شود.

روکایرول قاعده شاخه سازی دیگری را توسعه داد که به هیچ یک از گروه های فوق تعلق ندارد قاعده موسوم به چند قسمتی[۱۰۸] یا شاخه سازی k قسمتی. در این روش، روش GLB به کار گرفته می شود و قاعده شاخه سازی براساس جواب ϕ (صفر) آخرین مساله تخصیص خطی حل شده، برای محاسبه کران پایین در گره کنونی درخت جستجو می باشد. $X_n^{(i)}$را زیر مجموعه X_n (مجموعه جایگشت های {1,2,…,n}) شامل آن جایگشت های Π که در آن $\Pi(i)=\phi(i)$. به طور مشابه $X_n^{(i)}$مجموعه جایگشت های $\Pi \in X_n$ که در آن $\Pi(i) \neq \phi(i)$. گره کنونی به اضافه $n+1$ گره جدید به صورت دیل با مجموعه جواب های ممکن داده شده است. [٤]

$$X_n^{(1)}, X_n^{(1)} \cap X_n^{(2)}, X_n^{(1)} \cap X_n^{(2)} \cap \ldots \cap X_n^{(n-1)} \cap X_n^{(n)}, X_n^{(1)} \cap X_n^{(2)} \cap \ldots \cap X_n^{(n)} \quad (11-2)$$

افرادی مثل هان[۱۰۹]، هال و گرانت[۱۱۰] و نیز پاردالوس، راماک ویشنام[۱۱۱] ریسنده[۱۱۲] و لی[۱۱۳] از روش های خاص خود برخی مسائل QAP را حل نمودند. [٤]

در سال های اخیر، روشهایی که تکنیک های شاخه و کران را با پیاده سازی های موازی ترکیب کرده اند به طور گسترده استفاده می شوند. و با استفاده از آنها، بهترین نتایج الگوریتم های دقیق برای QAP در حال دستیابی هستند. اما نکته مهم این است که موفقیت برای مسائل با اندازه بزرگتر نیز به پیشرفت های تکنولوژیکی در زمینه سخت افزار مرتبط است. [۳]

[108] polytomic
[109] Hahn
[110] Grant
[111] Ramakrishnam
[112] Resende
[113] Li

۲-٤-۳-۱-۲ الگوریتم صفحات برش سنتی[۱۱٤]

الگوریتم های صفحات برش سنتی برای QAP توسط افراد مختلفی توسعه داده شده است که بازارا و شرالی[۱۱۵] و مازولا و بالاس[۱۱٦] و کافسن و بروکس[۱۱۷] از جمله آنها هستند. این الگوریتم ها از فرمول برنامه ریزی خطی عدد صحیح (MILP) برای QAP که مناسب برای تجزیه از روش بندرس می باشد استفاده نموده اند. در روش بندرس فرمول MILP به یک مساله اصلی و یک زیر مساله تجزیه شده است که یک مساله برده[۱۱۸] نامیده می شود(در اینجا ما آن مساله را فرعی می نامیم)، که مساله اصلی شامل محدودیت ها و متغییر های تخصیص اصلی می باشد. [٤]

برای یک تخصیص ثابت مساله فرعی معمولا یک برنامه خطی با متغییرهای تخصیص اصلی و متغییرهای ثانویه و در زمان چند جمله ای برای مقادیر ثابت این متغییر های ثانویه فرموله می شود. الگوریتم به صورت زیر عمل می کند:

ابتدا یک روش ابتکاری برای ایجاد یه تخصیص اولیه به کار برده می شود، آنگاه مساله فرعی برای متغییرهای ثابت متغییرهای تخصیص حل می شود که به طور ضمنی به آن اشاره می کند، و مقادیر بهینه اولیه و ثانویه را محاسبه می کند. اگر جواب ثانویه مساله فرعی تمامی محدودیت های مساله اصلی را ارضا کند، ما یک جواب بهینه برای فرمول MILP اصلی QAP داریم. در غیر این صورت، حداقل یکی از محدودیت های مساله اصلی رعایت نشده است. در این حالت مساله اصلی با مقادیر ثابت برای متغییرهای ثانویه مساله فرعی حل می شود و جواب حاصل به عنوان ورودی مساله فرعی خواهد بود. آنگاه این روش آن قدر تکرار می شود تا جواب مساله فرعی تمامی محدودیت های مساله اصلی را ارضا کند. [٤]

واضح است که هر جواب مساله اصلی که با متغییرهای ثانویه ثابت مساله فرعی برای برخی مقادیر شدنی به دست می آید، یک کران پایین برای QAP مورد نظر می باشد. از طرف دیگر مقدار تابع هدف QAP برای هر مقدار شدنی متغییرهای تخصیص در یک کران بالایی قرار می گیرد. الگوریتم زمانی پایان می پذیرد که کران های بالا و پایین بر هم منطبق شوند. معمولا زمان لازم برای دست یابی به کران های بالا و پایین بسیار زیاد می باشد، و لذا این روش ها برای حل مسائل QAP کوچک مناسب می باشند. [٤]

---

[114] Traditional Cutting Planes
[115] Bazaraa and Sherali
[116] Mazzola and Balas
[117] Kaufmann and Broeckx
[118] Slave Problem

۲-٤-۳-۱-۳ صفحات برشی چند وجهی

به طور مشابه با روش های صفحات برش سنتی نیز صفحات برش چندوجهی یا الگوریتم های شاخه و برش[۱۱۹] از یک آزاد سازی LP یا MILP مساله بهینه سازی ترکیبی حل شده در خصوص مسالهQAP مورد نظر استفاده می کند. به علاوه روش های صفحه برش چند وجهی از یک رده معتبر یا فاصله های کوچک تعریف شده مشخص که، به وسیله تمامی جواب های شدنی مساله اصلی انجام شده اند استفاده می کنند. اگر جواب آزادسازی برای مساله اصلی شدنی بود، که کار انجام شده است، در غیر این صورت برخی از فاصله های معتبر ذکر شده فوق احتمالا نقض می شوند.

در این حالت یک برش انجام می شود، که یک یا تعداد بیشتری از نابرابری های(نامعادلات) ارضا نشده از طریق آزادسازی MILP یا LP به مساله ما اضافه می شود. در حالی که از هیچ یک از نامعادلات معتبر تخطی نشود.

این حالت اخیر مجددا حل شده و کل فرایند تکرار می گردد. اما برخی محدودیت های یکپارچه نیز شکسته خواهد شد. این الگوریتم، یک مرحله شاخه سازی با ثابت قرار دادن مقادیر صحیح (شدنی) برای متغییر های مربوطه اجرا می کند. مرحله شاخه سازی یک درخت جستجو ایجاد می کند، همانند آن که در الگوریتم های شاخه و کران وجود دارد. هر گره از این درخت به صورتی که قبلا گفته شد با انجام برش هایی و آنگاه شاخه سازی آن، در صورت نیاز، پردازش می شود. واضح است که اجزا مرتبط از الگوریتم های شاخه و کران همانند کران های بالایی، قاعده انتخاب و قاده شاخه سازی، در الگوریتم شاخه و برش نقش ایفا خواهد نمود. از این رو، چنین دیدگاهی اجزا الگوریتم صفحه برشی و روش های شاخه و کران را با هم ترکیب می کند. [٤]

۲-٤-۳-۲ روش های ابتکاری

اگرچه بهبود های قابل توجهی در توسعه الگوریتم های دقیق برای QAP انجام شده است، اما مسائل $n>20$ به خاطر نیاز به زمان بسیار بالا توسطه رایانه ها هنوز عملا حل نشده اند. این موضوع توسعه روش های ابتکاری را به عنوان الگوریتم هایی که جواب های با کیفیت خوب ایجاد می کنند در زمان منطقی ضروری می سازد. از جمله روش هایی که در این رابطه مطرح شده است به شرح ذیل می باشد:

[119] Branch and Cut algorithms

۲-٤-۳-۲-۱ روش های ساختاری[۱۲۰]

این روش ها به وسیله گیلمور معرفی شد. آنها رویکردهای تکراری هستند که معمولا با یک جایگشت خالی شروع می شوند و به طور تکراری یک جایگشت بخشی را برای یک جواب QAP با تخصیص برخی تسهیل ها که هنوز تخصیص نیافته اند به مکان های خالی تکمیل می کند. [٤][1]

۲-٤-۳-۲-۲ روش های شمارشی محدود[۱۲۱]

این روش ها بر مشاهده ای که اغلب روش های شمارشی، مثل الگوریتم های شاخه و کران، جواب های خوب را در مراحل اولیه جستجو می یابد تکیه دارد و آنگاه زمان زیادی را برای بهبود حاشیه ای آن جواب یا تایید بهینگی آن صرف می کند. این رفتار روش های شمارشی، راهی برای صرفه جویی در زمان برای حالتی که به دنبال یک جواب خوب هستیم می یابد. این راه تحمیل برخی محدودیت ها به فرایند شمارش است. این محدودیت باید یک محدودیت زمانی باشد و با یک محدودیت در تعداد تکرارهای الگوریتم که باید بسط داده شود. [٤]

استراتژی دیگری که هدف مشابهی را به کار می گیرد مدیریت کران پایین می باشد. در صورتی که هیچ بهبودی در جواب با اجرای تعداد زیادی از تکرارها حاصل نگردد این امر با افزایش کران پایین صورت می گیرد. و منجر به برش های عمقی در درخت جستجو می شود تا بتواند سرعت را افزایش دهد. واضح است، چنین رویکردی جواب بهینه را نیز برش می دهد و از این رو باید به دقت مورد استفاده قرار گیرد، که این امر ممکن است در رابطه با برخی روش های ابتکاری مشخص که جستجوهای بسیار دقیق را در فضای جستجو انجام می دهند صادق باشد. [٤]

۲-٤-۳-۲-۳ روش های بهبود[۱۲۲]

این روش ها به رده بالاتری از الگوریتم های جستجوی محلی تعلق دارند. یک روش جستجوی محلی با یک جواب شدنی اولیه شروع شده و به طور تکراری سعی می کند تا جواب کنونی را بهبود بخشد. این امر با جایگزینی جواب اخیر با یک جواب شدنی(بهتر) در همسایگی اش میسر می شود. این مرحله تکراری آن قدر

[120] construction methods
[121] limited enumeration methods
[122] Improvement methods

ادامه می یابد تا هیچ بهبودی در جواب حاصل نشود. روش های بهبود الگوریتم های جستجوی محلی هستند که فقط بهبود در جواب کنونی را در هر تکرار میسر می سازند.

اجزا اصلی روش های بهبود (به طور عام جستجوی محلی) همسایگی و ترتیبی که همسایگی انجام می شود می باشند. به طور متناوب، همسایگی های استفاده شده برای QAP همسایگی تعویض زوجی[۱۲۳] و همسایگی تعویض سه گانه دوره ای[۱۲٤] می باشد. در حالت مبادلات زوجی، بازنگری اجمالی بر کل همسایگی انجام می گردد، مثل آنکه محاسبه مقادیر تابع هدف برای همه همسایه های یک جایگشت داده شده به مقدر $O(n^3)$ نیاز دارد(اندازه همسایگی $\binom{n}{2}$ می باشد، و O(n) مرحله برای محاسبه تفاوت مقادیر تابع هدف یک جایگشت $\prod$و یک جایگشت $\prod'$ در همسایگی $\prod$می باشد). اگر همسایگی $\prod$قبل از آن باشد و $\prod'$ یک همسایه $\prod$ باشد، آنگاه همسایگی $\prod'$ در $O(n^2)$ می تواند به طور خلاصه مورد بررسی قرار گیرد.

روش بهبود، بیشترین نوع از روشهای ابتکاری است(روش بهبود غالبا در فرا ابتکاریها استفاده می شود) که مورد تحقیق قرار می گیرد. مشهورترین روش های بهبود روش های جستجوی محلی و جستجوی ممنوعه هستند. [۱] و [۲۵]

۲-٤-۳-۲-۳-۱ جستجوی محلی

به دنبال یک راه حل بهتر در همسایگی راه حل فعلی می گردد. جستجو زمانی پایان می یابد که هیچ راه حل بهتری در آن همسایگی وجود نداشته باشد. [۱]

۲-٤-۳-۲-۳-۲ جستجوی ممنوع

یک نوع روش جستجوی محلی است که به وسیله گلاور به عنوان روشی برای دستیابی به بهینه محلی ارائه شد. در رابطه با جستجوی ممنوع هدف اصلی به خاطر سپردن جواب هایی است که جهت حل الگوریتم به آنها رسیده ایم، تا جواب هایی که را که پیش از این به دست آورده ایم شناسایی کرده تا در جستجوی بعدی آنها را تکرار نکنیم. لذا، به خاطر سپاری و جستجوی محلی همسایگی جواب به دست آمده تا این مرحله جستجو را به پیش می برد.

[123] pair-exchange neighborhood
[124] cyclic triple-exchange neighborhood

اجزا اصلی برای جستجوی ممنوع، ساختار همسایگی، حرکات و لیست ممنوع، و معیار توقف می باشد.

یک حرکت، عملیاتی است که وقتی برای یک جواب معین $\prod$به کار می رود، یک همسایگی $\prod'$ برای آن ایجاد می کند. در مورد QAP همسایگی، همسایگی مبادلات زوجی است و حرکت ها معمولا پس و پیش است. یک لیست ممنوع، یک لیست از حرکت های ممنوع است، حرکت های ممنوع آنهایی هستند که مجاز نمی باشد تا در جواب های جاری به کار رود. حالت ممنوع حرکت ها در جستجو تغییر می کند و لیست ممنوع در طول دوره جستجو به روز رسانی می شود.

یک معیار توقف وضعیتی است که وقتی با یک حرکت ممنوع تکمیل می شود، حالت ممنوع را حذف می کند. [٤]

این روش مشابه با جستجوی محلی کار می کند. با این حال، جستجوی ممنوعه گاهی اوقات مطلوب تر است به علت اینکه این روش برای مقابله با مشکل به دام افتادن در بهینه محلی طراحی شده بود[3]. این روش به طور کلی رفتار جستجوی خوبی نشان می دهد، اما برای حل برخی موارد مشکل دارد. [7] اما با اصلاحاتی که روی آن صورت گرفت به عنوان یکی از قدرتمند ترین روش های ابتکاری برای سالهای زیادی باقی مانده است. [18]

۲-٤-۳-۳ روش های فرا ابتکاری

قبل از پایان دهه ۱۹۸۰، بیشتر روش های ابتکاری پیشنهاد شده برای مسائل بهینه سازی ترکیبی خاص بودند و به یک مساله داده اختصاص داده شده بودند. اما پس از آن دوره، این الگو تغییر کرد، تکنیک های کلی تر که به عنوان فرا ابتکاری ها شناخته می شوند ظهور کردند که می توانند برای حل طبقه های بزرگی از مسائل، چه به طور مستقیم و یا با تغییرات جزئی، مورد استفاده قرار گیرند. [۳] فرا ابتکاری ها بیشتر تکنیک های کلی به نظر می رسند که با ساختار مساله سازگار می شوند. تعدادی از این تکنیک ها براساس نوعی از شبیه سازی یک رفتار طبیعی مطالعه شده در حوزه دیگری از دانش هستند. اکثر الگوریتم فرا ابتکاری همچنین می توانند به عنوان روش جستجو همسایگی (یا محلی) نامیده شوند. در تعریف اصلی، فرا ابتکاری ها راه حلی هستند که تعامل بین روش های بهبود محلی و استراتژی های سطح الاتر را برای ایجاد یک فرآیندی که قادر به فرار از راه حل های بهینه محلی باشد،مدیریت می کنند و یک جستجو خوب در فضای راه حل انجام می دهند. این روش ها همچنین شامل هر روشی که از استراتژی ای برای غلبه بر گیر افتادن بهینگی محلی به کار می برند می شود. [۲٦] تا کنون بسیاری از الگوریتم های فرا ابتکاری توسط محققان ارائه شده اند. در ادامه از میان انبوه الگوریتم های فرا ابتکاری،

مهم ترین فرا ابتکاری های ارائه شده برای مساله QAP که نتایج قابل قبولی داشته اند را به اختصار بیان می کنیم. الگوریتم های فرا ابتکاری به دو دسته کلی تقسیم می شوند:

۱. فرا ابتکاری های الهام گرفته از طبیعت مثل: الگوریتم ژنتیک، الگوریتم شبیه سازی انجماد تدریجی،بهینه سازی کلونی مورچگان.
۲. فرا ابتکاری های مبتنی بر ملاحظات نظری و تجربی مثل:جستجوی ممنوعه،روش جستجو حریصانه تصادفی تطبیقی، جستجوی همسایگی متغیر.

۲-٤-۳-۳-۱ الگوریتم ژنتیک

این الگوریتم ها، نگرش های الهام گرفته شده از طبعیت برای مسائل ترکیبی می باشند. ایده اولیه از وفق دادن عملیات مکانیسم های تکاملی در فرایند انتخاب در طبیعت برای مائل بهینه سازی ترکیبی است. اولین بار الگوریتم ژنتیک در سال ۱۹۷۵ توسط هالند[۱۲۵] مطرح شد. [٤]

الگوریتم ژنتیک روش جستجوی احتمالاتی فراگیر است که از فرایند تکامل زیست شناختی پیروی می کند. الگوریتم ژنتیک بر جمعیت جواب های بالقوه عمل می کند و اصول تنازع بقا را در تولید تقریب های بهتر و بهتر جواب مساله به کار می گیرد. در هر نسل مجموعه ی جدیدی از تقریب ها با فرایند انتخاب بهترین عضو بر اساس میزان برازش آنها در دامنه مساله و تکثیر با عملگر های گرفته شده از ژنتیک طبیعی ساخته می شود. این فرایند در نهایت به تکامل جمعیتی از اعضا ختم می شود که نسبت به اعضای اولیه که در واقع والدین اصلی آنهاست با محیط سازگاری بهتری دارند.

آشنایی با مفاهیم زیست شناختی طبیعی

با توجه به استفاده از مفاهیم زیست شناختی در ساختار و رویه ی الگوریتم ژنتیک آشنایی بیشتر با این مفاهیم سودمند است.

[125] Holland

موجودات زنده از محموعه ای سلول تشکیل می شوند و هر سلول حاوی مجموعه ای مشابه شامل یک یا چند کروموزوم است. کروموزوم ها از رشته هایی به نام مولکول های DNA تشکیل شده اند که در واقع حامل نقشه ی کد تشکیل موجود زنده اند. هر کروموزوم را می توان به طور مفهومی به ژن هایی تقسیم کرد که هر کدام کد تشکیل یک پروتئین به خصوص را بر خود حمل می کنند. به طور خلاصه هر ژن را می توان تعینن کننده یک ویژگی به خصوصی از موجود زنده مثل رنگ چشمان دانست. صورت های مختلف این ویژگی مثل رنگ آبی، سبز و یا قهوه ای به اصطلاح آلل[۱۲۶] نامیده می شود. هر ژن در محل معینی از کروموزوم قرار می گیرد. موجودات زنده معمولا کروموزوم های متعددی در هر سلول دارند. مجموعه ی کروموزوم ها یا در واقع مواد ژنتیکی را ژنوم[۱۲۷] می گویند. عبارت ژنوتیپ[۱۲۸] معمولا به مجموعه ی معینی از ژن های یک ژنوم گفته می شود. هر دو موجودی که ژنوم های مشابه ای داشته باشند ژنوتیپ مشابه دارند. ژنوتیپ ها در نهایت پس از رشد موجود زنده باعث مشخصات فیزیکی و ذهنی مانند رنگ چشم، قد، اندازه ی مغز، و میزان هوش می شوند که به اصطلاح فنوتیپ[۱۲۹] نامیده می شوند.

موجوداتی که کروموزوم های آنها به صورت جفت است داپلوئید[۱۳۰] و آنهایی که جفت نیست هاپلوئید[۱۳۱] نامیده می شوند.

به طور معمول در طبیعت بیشتر گونه هایی که از طریق جفت گیری جنسی تکثیر می شوند مانند انسان ها ساختار کروموزومی دایپلوئید دارند. برای مثال انسان ۲۳ جفت کروموزوم در هر سلول به استثنای سلول های جنسی دارد. در هنگام آمیزش عملگر ژنتیکی تقاطع صورت می گیرد به طوری که ابتدا بین ژن ها های هر جفت کروموزوم تبادل صورت می گیرد تا کروموزوم های منفرد که اصطلاحا گامت[۱۳۲] نامیده می شود ایجاد می شود. آنگاه گامت های یک طرف با گامت های جفت خود دو به دو جفت می شوند تا مجددا مجموعه ای کامل از جفت های کروموزومی ایجاد کنند. برای مثال در انسان مجددا یک مجموعه معین ۲۳ جفتی کروموزوم در هر سلول نوازد به وجود آمده شکل خواهد گرفت. در تکثیر جنسی موجودات نوع هاپلوئید، تبادل ژن ها بین کروموزوم های منفرد یک موجود و کروموزوم های منفرد موجودی دیگر صورت می گیرد. ژن جنین تشکیل

126 Allel
127 Genom
128 Genotype
129 Fenotype
130 Diploid
131 Haploid
132 Gamet

شده خود ممکن است تحت تاثیر جهش قرار گیرد به طوری که بعضی از نوکلئوتیدهای[۱۳۳] مولکول های DNA هنگام انتقال از والیدن به جنین تغییر شکل دهند. [۲۳]

میزان سازگاری هر موجود با محیط نیز معمولا با میزان احتمال زنده ماندن تا تولید مثل(قدرت زیست) و یا به عنوان تابعی از تعداد نوزادانی که می تواند تولید کند(باروری) نشان داده می شود.

کروموزوم

رشته یا دنباله ای از بیت ها که به عنوان شکل کد شده یک جواب ممکن(مناسب یا نامناسب) از مساله مورد نظر می باشد، کروموزوم نام دارد. در حیقیت بیت های یک کروموزوم،نقش ژن ها را در طبیعت بازی می کنند. هر بیت متغییری گسسته است که از یک مجموعه Q یا آلل عضوی انتخاب می شود. چنانچه از کد گذاری باینری استفاده شود،هر بیت یک از دو مقدار ۰ یا ۱ را می پذیرد و بنابراین در این حالت Q=2 می باشد. [۲٤]

بررسی رشته های کروموزوم به طور جداگانه هیچ گونه اطلاعاتی را درباره ی مساله ای که درصدد حل آن هستیم به دست نمی دهد. تنها در صورت خارج کردن کروموزوم از کد و احتساب ارزش آن می توان نمایش آن را معنی دار کرد. [۲۳]

جمعیت[۱۳٤]

مجموعه ای از کروموزوم را جمعیت گویند. یکی از ویژگی های ژنتیک این است که به جای تمرکز بر روی یک نقطه از فضای جستجو یا یک کروموزوم،بر روی جمعیتی از کروموزوم ها تمرکز می کندبه این ترتیب در هر مرحله،الگوریتم دارای جمعیتی از کروموزوم ها بوده که خواص مورد نظر را بیشتر از جمعیت مرحله قبل دارا می باشد. هر جمعیت یا یک نسل از کروموزوم ها،دارای یک اندازه می باشد که به اندازه جمعیت معروف است. اندازه جمعیت معرف تعداد کروموزوم های موجود در جمعیت یا یک نسل است. اگر تعداد کروموزوم ها خیلی کم باشد،امکان شکل گیری عملیات جا به جایی توسط الگوریتم ژنتیک بسیار کم خواهد بود و تنها قسمت کمی از فضای جستجو مورد کاوش قرار خواهد گرفت. از طرف دیگر،اگر تعداد کروموزوم ها خیلی زیاد باشد،الگوریتم بسیار کند خواهد شد. براساس پژوهش ها،جمعیت های با اندازه حدود ۲۰ تا ۳۰ کروموزوم،مناسب تر می باشند.

133 Nucleotide
134 Population

البته گاهی اوقات جمعیت با اندازه ۵۰ تا ۱۰۰ بهترین جواب ها را داده اند. بعضی پژوهش ها نیز نشان می دهد که اندازه جمعیت باید براساس نوع مساله و کدینگ آن تعریف شود و افزایش بیشتر آن بی فایده خواهد بود و هرگز به حل سریع تر مساله کمک نمی کند[۲٤]

مقدار برازندگی[۱۳۵]

مناسب بودن یا نبودن جواب،با معیاری که از تابع هدف به دست می آید،سنجیده می شود. جواب هر چه مناسب تر باشد،به همان اندازه مقدار برازندگی بزرگتری دارد. برای آنکه شانس بقای چنین جوابی بیشتر شود،احتمال بقای آن،متناسب با مقدار برازندگی آن در نظر گرفته می شود. بنابراین، کروموزومی که برازنده تر است با احتمال بیشتری در تولید فرزندان شرکت می کند و دنباله های بیشتری از آن به وجود می آید. برای مثال،چنانچه هدف بیشینه کردن یک تابع باشد،مقدار برازندگی،یک تابع صعودی از تابع هدف در نظر گرفته می شود و اگر هدف یافتن مقدار کمینه یک تابع باشد،عدد برازندگی،یک تابع نزولی از آن قرار داده می شود. معمولا در مواردی که امکان دارد و مناسب باشد،تابع برازندگی را در فاصله {۰ و ۱} نرمال می کنند. [۲٤]

انتخاب

انتخاب،فرایند دو والد از جمعیت برای عمل تقاطع است. در این مرحله،باید در ارتباط با نحوه انتخاب والدین برای عمل تقاطع، نحوه تولید فرزندان و تعداد فرزندان تصمیم گیری گردد. هدف انتخاب این است که والدین شایسته تر انتخاب شده که منجر به تولید فرزندانی با برازندگی بالا گردد. کروموزوم هایی که از جمعیت اولیه برای تولید مثل انتخاب می گردند،والدین نام دارند. انتخاب،نحوه تعیین این والدین را مشخص می کند. براساس نظریه تکامل داروین،بهترین ها باید شانس بیشتری برای تولید مثل و بقا داشته باشند. [۲٤]

در مرحله انتخاب، ابتدا به کمک تابع برازش، به هر عضو جمعیت، ارزشی نمایانگر میزان سازگاری آن تعلق می گیرد. این ارزش در مرحله ی انتخاب میزان گرایش به اعضای سازگارتر را تعیین می کند. اعضایی که نسبت به دیگران سازگاری بالاتری دارند شانس بالاتری برای انتخاب خواهند داشت، در صورتی که اعضایی که سازگاری

[135] fitness value

کمتری دارند به همان نسبت شانس کمتری خواهند داشت. هنگامی که میزان سازگاری هر عضو تعیین شود اعضا را می توان با احتمالی متناسب با میزان سازگاری نسبی آنها برای تولید نسل بعدی انتخاب کرد.

انتخاب روشی است که به طور تصادفی کروموزوم هایی را از جمعیت،برای تولید مثل بیرون می آورد. کروموزوم با تابع برازندگی بالاتر،شانس بیشتری برای انتخاب دارد. فشار رویه انتخاب به معنای درجه توجه به والدین بهتر در رویه انتخاب است که توسط گلدربرگ و دب[۱۳۶] در سال ۱۹۱۹ تعریف گردید. فشار بالای رویه انتخاب منجر به تاکید بیشتر در انتخاب والدین با شایستگی بیشتر می گردد. [۲٤]

روش چرخه رولت[۱۳۷]

روش چرخ رولت،یکی از روش های سنتی انتخاب در الگوریتم ژنتیک است. در این حالت یک کروموزوم از مخزن تولید مثل براساس احتمالی متناسب با مقدار شایستگی اش انتخاب می گردد. در این مدل،سطح چرخ به بخش هایی تقسیم می شود که تعداد اعضای جمعیت و سطح هر بخش متناسب با مقدار برازندگی هر کروموزوم است. سپس چرخ به گردش در می آید تا در نقطه ای به تصادف متوقف گردد. این نقطه، کروموزوم انتخاب شده را مشخص می سازد. . این شیوه انتخاب سبب می شود که با گذشت زمان،تعداد کروموزوم های مطلوب در جمعیت افزایش یابد به طوری که میانگین مقدار برازندگی جمعیت،در مقایسه با جمعیت مرحله قبل،بیشتر می شود.

کروموزوم های جمعیت به شکل دنباله در آورده شده و سپس مجموع مقدار برازندگی هر کروموزوم با مقدار برازندگی تمام کروموزوم های قبل از آن محاسبه می گردد. سپس مقدار برازندگی نسبی تجمعی کروموزوم ها محاسبه می گردد. یک عدد تصادفی n بین صفر و یک ایجاد می شود. از بین کروموزوم ها،اولین کروموزومی که مقدار برازندگی تجمعی نسبی آن بیشتر از n باشد،انتخاب می گردد.

[۱۳۸]عملگر تقاطع

136 Goldberg and Deb
137 roulette wheel
138 Crossover operator

فرایندی است که دو والد را در نظر گرفته و براساس آن ها یک فرزند جدید تولید می کند. عملگر تقاطع روی حوضچه تولید مثل به این امید که فرزند بهتری تولید گردد،به کار گرفته می شود. این عملگر یک اپراتور ترکیب بندی است که در سه مرحله صورت می گیرد:

۱. اپراتور تولید مثل (رویه انتخاب) یک زوج از والد را از حوضچه تولید مثل انتخاب می کند.
۲. یک نقطه تقاطع به طور تصادفی در طول رشته انتخاب می شود.
۳. در نهایت،مقادیر رشته ها با توجه به نقطه تقاطع تعویض می گردند.

عملگر تقاطعی با یک احتمال از قبل تعیین شده(P_C) بر روی کروموزوم های والد عمل می کند. بدین معنی که با این احتمال عمل تقاطع انجام می گیرد. اگر هیچ تقاطعی صورت نگیرد،فرزندان دقیقا مشابه والدین خواهد بود. در صورتی که عمل تقاطع صورت گیرد،فرزندان از قسمت های مختلف کروموزوم های والد ساخته می شوند. اگر احتمال تقاطع یک باشد،تمامی فرزندان از طریق عمل تقاطع ایجاد می شوند. عملیات تقاطع با این هدف انجام می شود که کروموزوم های جدید در بردازنده قسمت های مناسب و خوب کروموزوم های قبلی خواهند بود و شاید این کروموزوم های جدید عملکرد بهتری داشته باشند. اما بهتر است همیشه بهترین کروموزوم های نسل قبلی بدون هیچ تغییری به نسل جدید منتقل شوند.

روش های مختلفی برای عملگر تقاطع وجود دارد و در زیر بعضی از روش های متداول آن شرح داده می شود:

عملگر تقاطعی تک نقطه ای:دو کروموزوم را به طور تصادفی از یک نقطه شکسته و بخش های شکسته شده دو کروموزوم را جا به جا می کند. بدین ترتیب،دو کروموزوم جدید به دست می آید. به کروموزوم های اولیه، کروموزوم های والد و به کروموزوم های حاصل شده از عمل جا به جایی، کروموزوم فرزند می گویند. شکل زیر یک مثال از عملگر تقاطع تک نقطه ای را نمایش می دهد. [۲٤]

PARENT 1	1 0 1 1 0 0 1 0
PARENT 2	1 0 1 0 1 1 1 1

CHILD 1	1 0 1 1 0 1 1 1
CHILD 1	1 0 1 0 1 0 1 0

شکل ۲-۲- نمونه ای از عملگر تقاطع تک نقطه ای[۲۴]

عملگر جهش[۱۳۹]

پس از اعمال تقاطع، کروموزوم های ایجاد شده تحت عملگر جهش قرار می گیرند. عملگر جهش در کروموزوم هایی که به صورت صفر و یک کد گذاری شده اند بیت های تشکیل دهنده ی کروموزوم را به صورت تصادفی و معمولا با احتمال ناچیزی مثلا ۰. ۰۰۱ تا ۰. ۰۱ تغییر می دهد.

در کروموزوم هایی که به صورت اعداد حقیقی کد گذاری شده اند جهش با تغییر متغییر ها و معمولا با احتمال بیشتری صورت می پذیرد. تجربه نشان داده است که میزان احتمال بیشتر در جهش متغییر های اعداد حقیقی در کارایی و سرعت همگرایی الگوریتم تاثیر بیشتری دارد.

روش های گوناگونی در تغییر بیت های صفر و یک یا متغییر های اعداد حقیقی اعمال می شود. [۲۳]

جایگزینی[۱۴۰]

مرحله جایگزنی در واقع مکمل مراحل انتخاب،تقاطع و جهش است. در این مرحله کروموزوم هایی که باید با نوزادان جدید تعویض شوند مشخص می شوند. در الگوریتم ژنتیک استاندارد از روش جایگزینی نسل به نسل

[139] mutation operator
[140] reproduction

استفاده می شود. در این شیوه ی جایگزینی،تمام جمعیت جاری با نوزادان جدید جایگزین می شود. در الگوریتم ژنتیک پایدار نوزاد خلق شده تنها در صورتی جایگزین والدین خود می شود که درجه ی سازگاری بهتری داشته باشد. بنابراین تنها بخشی از جمعیت جاری در هر نسل تعویض می شوند و کروموزوم می تواند در نسل های متعدد پایدار بماند. در الگوریتم ژنتیک تدریجی،نوزاد جدید یا به جای یکی از والدین خود قرار می گیرد، و یا جایگزین کروموزوم دیگری جانشین می شود،یا به جای بدترین کروموزوم قرار می گیرد،و یا جایگزین کروموزومی می شود که بیشترین شباهت را با آن دارد. در الگوریتم ژنتیک با رویکرد حفظ خبرگان تعدادی از کروموزوم ها با بالاترین میزان سازگاری از یک نسل به نسل دیگر عینا انتقال می یابند. این کروموزوم ها معمولا جهش نمی کنند و چنانچه در عمل تقاطع شرکت داده شوند نوزادانی که به وجود می آمورند باعث از بین رفتن آنها نمی شوند.

هنگامی که نوزادان جدیدی با استفاده از عملگرهای انتخاب،تقاطع،و جهش به وجود می آیند. می توان میزان سازگاری آنها را ابتدا تعیین کرد. اگر تعداد نوزادان تولید شده کمتر از تعداد جمعیت قبلی باشد به روش آنها را وارد جمعیت جدید کنیم. به همین روش اگر نخواهیم تمام نوزادان را وارد جمعیت جدید کنیم و یا این که تعداد نوزادان،بیشتر از تعداد جمعیت قدیم باشد باید طراحی را برای انتخاب نوزادان و استقرار آنها در جمعیت جدید اتخاذ کنیم. جایگزینی نوازادان در جمعیت جدید معمولا به دو شیوه ۱–فراگیر و ۲–محلی انجام می شود. [۲۳]

روش استاندارد GA حتی در موارد اندازه کوچک تا متوسط از QAP ضعیف عمل می کرد، بنابراین برای بهبود عملکرد روش ژنتیک در QAP، چند طرح ترکیبی پیشنهاد شده است. از جمله روش ترکیب GA با TS و روش های ترکیبی دیگر با ترکیب یک روش حریص با GA. به طور خاص، GA حریص به خوبی در موارد مقیاس بزرگ از موارد QAP عمل کرد. [۲] و [۳۰]

۲-٤-۳-۳-۲ روش جستجوی تطبیقی تصادفی حریصانه[۱٤۱]

این روش به وسیله فتو[۱٤۲] و ریسنده برای مسائل بهینه سازی NP-hard با موفقیت به کار گرفته شد و در این میان در مسائل QAP و BiQAP استفاده گردید.

GRASP ترکیبی از اجزا حریص با اجزا جستجوی تصادفی در یک ابتکار دو مرحله ای است که متشکل از مرحله ساخت و مرحله بهبود محلی است. در مرحله ساخت،جواب های خوب از فضای شدنی در دسترس

[141] Greedy Randomized Adaptive Search Procedure (GRASP)
[142] Feo

ساخته می شود. با در نظر گرفتن این که مرحله بهبود محلی همسایگی جواب ساخته شده در اولین مرحله به جستجوی بهبود های ممکن می پردازد. [٤]

مکانیسم ساخت جواب،یک جواب بهینه اولیه با استفاده از روش جستجوی تصادفی افزایشی،می سازد که تصادفی بودن آن،ما را قادر می سازد تا در فضاهای مختلف منقطه شدنی به جواب برسیم.

پارامترهای ورودی اندازه لیست کاندیدای محدود شده[۱۴۳]،حداکثر تعداد تکرارها و یک مقدار تصادفی می باشد. RCL مشتمل بر کاندیدهایی است که در آن نمونه برداری مربوط به ساخت یک جواب در اولین مرحله انجام خواهد شد. [٤] در مرحله ساخت یک جواب شدنی به طور تکراری ساخته می شود،یک جز در یک لحظه. در هر تکرار ساخت،انتخاب جز بعدی که باید اضافه گردد،پس از مرتب سازی تمامی اجزا در لیست کاندیدا به صورت افزایشی تعیین می گردد. این روش یک روش تطبیقی است زیرا منافع آن بستگی به هر جزئی دارد که در هر تکرار به روز می شود. این روش از ابتدای مرحله ساخت تا زمان ایجاد تغییرات در هر تکرار با انتخاب جز قبلی ادامه می یابد. بخش احتمالی این روش معمولا با انتخاب بهرین کاندیداها از لیست انجام می شود،اما نه لزوما با انتخاب بهرین گزینه کاندیدا. این روش انتخاب ما را مجاز می کند تا جواب های مختلفی را در هر تکرار به دست آوریم،اما لزوما قدرت رسیدن به راه حل مناسبی برای این روش را ندارد. [۲۲]

انجام یک جستجوی محلی جهت تلاش برای بهبود هر یک از جواب های مرحله ساخت سودمند می باشد. طبیعتا یک روش بهینه سازی محلی مثل 2-exchange به کار گرفته می شود. در حالی که چنین روش هایی نیازمند زمان نمایی از یک نقطه شروع دلخواه می باشند،به طور تجربی کارایی آنها نسبت به بهبود جواب اولیه به طور قابل توجهی بهبود می یابد. با استفاده از یک ساختار داده ای سفارشی و اجرای دقیق آن ساختار،می توان یک مرحله ساخت را اجرا کرد که در این مرحله جواب های اولیه خوبی تولید می شوند و براساس آن یک جستحوی معمولی کارا صورت می پذیرد. نتیجه اینکه اغلب بسیاری از جواب های حاصل از GRASP در زمان های مشابهی برای روش های بهینه سازی محلی جهت همگرا شدن از یک نقطه شروع تصادفی تکی تولید می شود. بهترین جواب های حاصل از GRASP عموما به طور قابل توجهی بهتر از جواب های حاصل از نقطه شروع تصادفی می باشد. [۲۲]

[143] restricted candidate list(RCL)

این تکنیک به وسیله چندین پژوهشگر برای QAP استفاده شد. این روش برای ۸۸ نمونه از مسائل QAP به کار برده شد، در تقریبا در هر مورد بهترین راه حل شناخته شده را به دست آورد،حتی برای چند نمونه راه حل ها را بهبود داد. [۳]

۲-٤-۳-۳-۳ سیستم های مورچه[۱٤٤]

مدل سازی رفتار حشرات اجتماعی مانند مورچه و زنبور و استفاده از این مدل ها برای جستجو و حل مسائل،از زمینه های مورد بحث در زندگی حشرات گروهی است. الگوریتم کلونی مورچه ها یک روش موفق بهینه سازی می باشد،که الگوریتم جستجو نشات گرفته از رفتار مورچه های واقعی در یافتن غذا می باشد.

سیستم های مورچه روش های ابتکاری را برای مسائل بهینه سازی ترکیبی توسعه داد که تلاش می کند تا از رفتار یک کلونی مورچه در جستجوی غذا تقلید کند. در ابتدا مورچگان در مجاورت لانه خود به طور تصادفی به دنبال غذا هستند. آنگاه مورچه به یک منبع غذایی دست یافته و مقداری از آن را برمیدارد و به لانه می آورد. در طول این سفر کوتاه مورچه اثری از خود در زمین بر جای می گذارد که ماده ای به نام فرومون نامیده می شود. اثر فرومون راهنمایی است برای جستجوی بعدی مورچه ها به سمت منبع غذایی به طور متناوب مراجعه می شود(به وسیله تعداد زیادی از مورچگان) اثر فرومونی قوی تر می گردد. در تلاش برای تقلید از رفتار مورچه الگوریتم هایی برای مسائل بهینه سازی ترکیبی به دست آمد. قیاس های ذیل می تواند در این الگوریتم استفاده شود.

الف) منطقه جستجو شده به وسیله مورچه شبیه جواب های شدنی می باشد.

ب) مقدار منابع غذایی شبیه مقدار تابع هدف می باشد.

ج) اثر فرومون شبیه یک جز از حافظه تطبیقی می باشد.

سیستم مورچه برای اولین بار توسط دوریگو و کلورنی[۱٤٥]،دوریگو و مانیتسو[۱٤٦] معرفی شد و تقریبا جواب های خوبی برای مسائل مشهور مثل مساله فروشنده دوره گرد و مساله تحصیص درجه دوم ارائه نمود. سپس دوریگو و دیکارو الگوریتم عمومی توده مورچه ها را به عنوان یک روش توده مورچه های فراکاوشی معرفی نمودند که ما را قادر به کاربرد برای دیگر مسائل مهندسی نمود، مشروط بر اینکه مساله به طور مناسبی فرموله شده و قابل

144 Ant Systems
145 Dorigo and Colorni
146 Manietzzo

تصویر شدن بر روی یک نمودار باشد. اخیرا دوریگو و همکاران کاربرد موفقی از الگوریتم مورچه را برای تعدادی از مسائل پایه بهینه سازی گزارش نموده اند. دیگر کاربرد های موفق الگوریتم مورچه در مسادل بهینه سازی طراحی و بهره برداری در مهندسی در شاخه های مختلف توسط عباسپور و همکاران، سیمسپون و همکاران و جلالی و همکاران گزارش شده است.

اگر چه پیاده سازی اولیه ACO برای QAP قابل رقابت با دیگر فرا ابتکاری ها نبودند، اما نتایج عددی نشان می دهد که به نظر می رسد ACO یکی از بهترین روش های موجود برای، موارد QAP ساخت یافته واقعی هستند. [۲] و [۳]

۲-٤-۳-۳-٤ الگوریتم انجماد تدریجی

این روش یک دیدگاه جستجوی محلی است که اینها هم برای مقابله با بهینه محلی استفاده می شوند، اما برخلاف جستجوی محلی اجازه حرکت به سمت مسیر جواب بدتر را با یک احتمال معین هم می دهد[7]. این روش از مکانیک های آماری سرچشمه گرفته است. و مبتنی بر مدل مونت کارلو برای شبیه سازی سطوح انرژی در سرد کردن مواد جامد است. این روش از سطح دما به عنوان یک استراتژی واضح برای هدایت جستجو استفاده می کند. در ابتدا، در حالت اولیه از اکتشاف، الگوریتم سخت گیر نیست و مستعد حرکت به سمت یک راه حل بدتر است. با این حال، با هر تکرار از الگوریتم با توجه به نیاز به یک راه حل بهتر در هر مرحله سختگیرانه تر می شود. [۳۰]

این روش از مقایسه بین مسائل بهینه سازی ترکیبی و مسائل آماری علم مکانیک بهره می گیرد. کرک پاتریک[۱٤۷] از انجمن IBM و گلات[۱٤۸] و واکچی و چنی[۱٤۹] جز اولین کسانی بودند که این قیاس را توانستند انجام دهند و نشان دهند که چگونه این الگوریتم، برای شبیه سازی رفتار یک سیستم ذرات فیزیکی، می تواند به عنوان یک روش ابتکاری برای مساله TSP به کار رود. مقایسه بین یک مساله بهینه سازی ترکیبی و یک سیستم فیزیکی با ذرات بسیار، به طور پایه ای بر دو واقعیت تکیه دارد:

۱. جواب های شدنی مساله بهنیه سازی ترکیبی بر حالت های یک سیستم فیزیکی بستگی دارد و
۲. مقادیر تابع هدف به انرژی وضعیت های سیستم فیزیکی وابسته است. [٤]

147 Kirkpatrick
148 Gellat
149 Vacchi and Çenny

مقایسه ساختار پیچیده پیکره بندی فضای جواب مسائل بهینه سازی NP-hard با بعضی از پدید های فیزیکی، منجر به ارائه پیشنهاد یک روش جدید تکراری (تکنیک انجماد تدریجی یا شبیه سازی تبرید) که توانایی خروج از بهینه محلی را داشت، گردید. همچنین اثر مشابهی در همان زمان توسط کرنی در سال ۱۹۸۵ منتشر گردید. آنها از الگوریتم انجماد تدریجی برای افراز بندی گراف ها استافده نمودند. انجماد تدریجی به دلیل سادگی و همچنین کارایی بالا در حل مسائل بهینه سازی ترکیبی، جایگاه ویژه ای در بین تکنیک های جستجو و هیوریستیک ها در دهه ۱۹۸۰ به دست آورد. این تکنیک بعد ها به حوزه مسائل بهینه سازی پیوسته نیز توسعه داده شد. مرور خوبی روی الگوریتم انجماد تدریجی توسط سومان و کومار انجام شده است. [٤]

۲-٤-۳-۳-۵ الگوریتم مهاجرت پرندگان[۱۵۰]

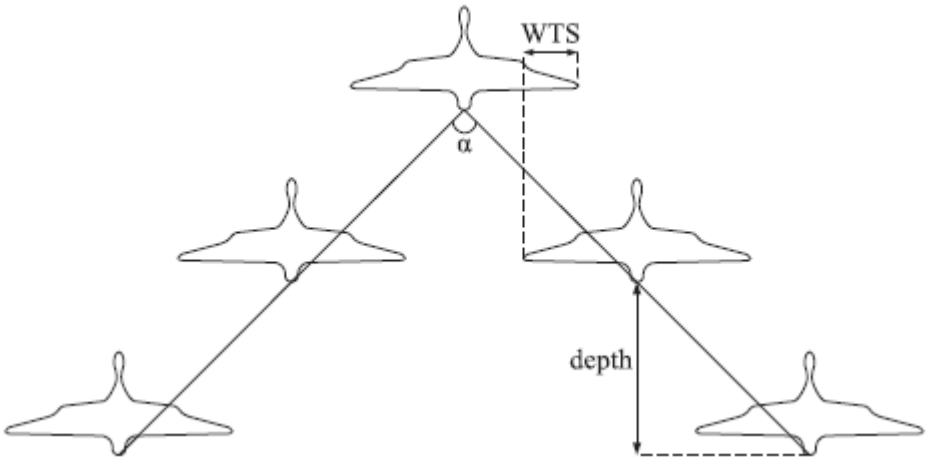
الگوریتم مهاجرت پرندگان از ساختار پرواز پرندگان مهاجر الهام گرفته شده است که یک ساختار کارا برای ذخیره سازی انرژی است. [۳۱]

الگوریتم MBO یک تکنیک جستجوی همسایگی است. این الگوریتم با تعدادی از راه حل اولیه متناظر با شکل پرندگان به صورت V شروع می شود. با راه حل اولیه (مربوط به پرنده رهبر) شروع می شود و و ادامه می یابد در امتداد خط ها به سمت دم، هر راه حل تلاش می کند تا راه حل همسایه خود را بهبود دهد. (برای پیاده سازی QAP، یک راه حل همسایه توسط جا به جایی دو به دوی هر دو محل به دست می آید.) اگر بهترین راه حل همسایه یک بهبود را به ارمغان آورد، راه حل فعلی با آن جایگزین می شود. همچنین یک مکانیزم مفید برای راه حل (پرنده) از راه حل هایی که در مقابل آنها وجود دارد. در اینجا این مکانیزم مفید را به عنوان به اشتراک گذاری بهترین همسایگان استفاده نشده با راه حل هایی که به دنبال می کنند تعریف می کنیم. در اینجا " استفاده نشده " به معنی یک راه حل همسایه است که به جای راه حل های فعلی استفاده نشده است. به عبارت دیگر، یک راه حل، تعدادی از همسایگان خود و تعدادی از بهترین راه حل های همسایگان قبلی را ارزیابی می کند و توسط بهترین آنها را جایگزین می شود. هنگامی که تمام راه حل های بهبود یافتند (یا تلاش برای بهبود انجام دادند) توسط راه حل های همسایه، این رویه به یک تعداد بار (تور) تکرار می شوند و پس از آن راه حل اول به عتوان آخرین راه حل می شود و یکی از راه حل دوم به عنوان اول در نظر گرفته می شود و حلقه از اول شروع می شود. این الگوریتم پس از یک تعداد تکرارها متوقف می شود. [۲۷]

ساختار V شکلی معروف ترین ساختاریست که پرندگان مهاجر برای پرواز در مسافت های طولانی استفاده می کنند. این ساختار به این دلیل به این نام شناخته می شود که شکل پرندگان به حرف " V " (شکل ۲-۷) شباهت

[150] Migrating Birds Optimization

دارد. در این ساختار، یک پرنده جمعیت پرندگان را رهبری می کند و دو خط ممتد از پرندگان دیگر وجود دارند که آن را دنبال می کنند .که در زیر جزییات آن قابل مشاهده است، که اعتقاد بر این است که این ساختار یک شکل بسیار کارآمد برای پرندگان مهاجر است.با این حال، این ساختار تنها ساختاری که جمعیت پرندگان استفاده می کنند نیست. ساختار های معمولی دیگر که مورد استفاده قرار می گیرید ساختار ستونی،ساختار قوسی شکل[۱۵۱]، ساختار J و ساختار پله[۱۵۲] هستند، که انواع دیگری از ساختار V شکلی هستند که در آن یکی از پایه های ساختار کوتاه تر شده و یا به طور کامل از بین رفته است.[۳۳]



شکل ۲-۳- ساختار Vشکلی[۲۷]

در زیر، در زیر نماد های استفاده شده و پس از آن شبه کدی از الگوریتم MBO ارائه داده می شود. فرض کنید:

N = تعداد راه حل های اولیه (پرنده)

K = تعدادی از راه حل های همسایه که در نظر گرفته می شود

X = تعدادی از راه حل های همسایه که با راه حل بعدی به اشتراک گذاشته می شود

M = تعداد تورها

K = حد تکرار

[151] bow-shaped
[152] echelon formations

شبه کد الگوریتم مهاجرت پرندگان:

Pseudocode of MBO:

```
1.  Generate n initial solutions in a random manner and place them on an hypothetical V formation arbitrarily
2.  i = 0
3.  while(i < K)
4.    for (j = 0; j < m; j++)
5.      Try to improve the leading solution by generating and evaluating k neighbors of it
6.      i = i + k
7.      for each solution s_r in the flock (except leader)
8.        Try to improve s_r by evaluating (k–x) neighbors of it and x unused best neighbors from the solution in the
    front
9.          i = i + (k – x)
10.     endfor
11.   endfor
12.   Move the leader solution to the end and forward one of the solutions following it to the leader position
13. endwhile
14. return the best solution in the flock
```

شکل ۲-۴ – شبه کد الگوریتم مهاجرت پرندگان[۲۷]

همانطور که تا الان مشخص است، الگوریتم MBO شباهت زیادی با داستان پرندگان مهاجر دارد. در ابتدا با راه حل ها به عنوان پرندگان تراز شده به شکل V رفتار می کند. تعداد همسایه تولید شده (K) را می توان به عنوان نیروی القایی مورد نیاز که به صورت معکوس با سرعت متناسب است تفسیر کرد. با مقدار K بزرگتر، ما فرض می کنیم که پرندگان در حال پرواز با سرعت کم پرواز می کنند همچنین ما می توانیم این تمثیل را داشته باشیم که سفر با سرعت پایین اطراف را با جزئیات بیشتر کشف می کند.

فلسفه MBO این است که، با تعدادی از راه حل های موازی شروع می شود، این به هدف کشف مناطق بیشتر در فضای شدنی راه حل می باشد. اکتشاف با مشاهده راه حل های همسایه آن امکان پذیر است. هر بار که یکی از راه حل (یکی که در موقعیت رهبر است) با جزئیات بیشتری پرداخته می شود هنگامی که یکی از راه حل ها نتواند خودش را به وسیله همسایگان خود بهبود بخشد و اگر راه حل پیش رو امیدوار کننده تر باشد، آن راه حل به وسیله یکی از همسایگان راه حل پیش رو جایگزین شده است. به این ترتیب در همسایگی راه حل با امید بیشتر با جزئیات بیشتری به کاوش و بررسی پرداخته خواهد شد. (توسط ترکیب نیروهای دو پرنده و یا دو راه حل). هنوز پس از چند تکرار این راه حل ها ممکن است به جهات مختلفی بروند تا زمانی که آنها بهبود را در امتداد مسیر خود پیدا کنند. با این حال، بعد از مدتی ممکن است ما انتظار داشته باشیم بسیاری از راه حل ها به یک یا چند منطقه که در آن ها بهینه محلی و یا حتی بهینه سراسری موجود هستند همگرا شوند. همگرایی با

مقادیر بزرگتر X می تواند سریعتر صورت گیرد اما در آن حالت ممکن است قبل از اینکه منطقه شدنی به طور کامل مورد بررسی شود پایان پذیرد و در نتیجه نتایج به دست آمده ممکن است خوب نباشد. [۲۷]

ویژگیهای MBO که آن را از دیگر رویکردهای فرا ابتکاری متمایز می کند تعداد راه حل هایی که به صورت موازی در حال اجرا هستند مکانیزم سود بین راه حل هاست. پردازش موازی به نحوی می تواند به عنوان ارث برده شده از الگوریتم ژنتیکی و جستجو پراکندگی در نظر گرفته شود. از سوی دیگر، اگر چه MBO به نظر می رسد که به برخی الگوریتم های هوش ازدحامی و به ویژه به ABC شباهت دارد. در حالی که در آن راه حل های بهتر، بیشتر مورد تحقیق قرار می گیرند، و همچنین مکانیسم سود معرفی شده در این الگوریتم MBO را کاملا منحصر به فرد می کند. [۲۷]

پس از مشاهده نتایج MBO که برای یک نمونه از QAP ناشی از داده های مونتاژ PCB واقعی بسیار موفق بود، توانایی آن را در به دست آوردن راه حل های بهینه دیگر مورد بررسی قرار دادند. برای این منظور آن را برای نمونه های QAPLIB که بیش از یک صد موارد QAP به همراه راه حل های بهینه و یا بهترین راه حل شناخته شده آنها(BKS) در آن موجود است مورد استفاده قرار دادند. [۲۷]
عملکرد این الگوریتم بر روی حل مساله تخصیص درجه دوم که از تخته مدار چاپی کارگاههای مونتاژ به وجود آمده اند تست شد. الگوریتم فرا ابتکاریMBO نسبت به بهترین عملکرد گزارش شده آن(انجماد تدریجی شبیه سازی) به طور متوسط حدود سه درصد عملکرد بهتری را نشان داده است. [۲۷]
الگوریتم MBO نیز برای یک مجموعه ای از مسائل معیار به دست آمده از QAPLIB اعمال شد که در آن در اغلب موارد قادر به پیدا کردن راه حل های شناخته شده بود. این نتایج نشان می دهد که MBO دارای پتانسیل برای تبدیل شدن به یکی رقابتی ترین الگوریتم های فرا ابتکاری را دارد. مکانیسم سود امکان کشف مناطق امیدوار کننده تر در فضای جستجو با جزئیات بیشتر را فراهم می سازد. این مکانیزم که منحصر به فرد برای MBO است می تواند به عنوان قدرت آن دیده شود. [۲۷]

۲-٦ مبانی الگوریتم جستجوی گرانشی

در دهه های گذشته، علاقه به الگوریتم های الهام گرفته از رفتار پدیده های طبیعی رو به رشد بوده است.
توسط بسیاری از محققان نشان داده شده است که این الگوریتم ها به خوبی برای حل مسائل محاسباتی پیچیده مثل بهینه سازی توابع هدف، الگو شناسی، کنترل اهداف، پردازش تصویر و مدل سازی فیلتر مناسب بوده است.

روش های فرا ابتکاری متعددی توسط تحقیقات تاکنون استفاده شده است. این الگوریتم ها به تدریج مورد تجزیه و تحلیل و یا طراحی توسط محققان در حوزه های مختلفی قرار گرفتند این الگوریتم ها مسائل بهینه سازی مختلف را حل کرده اند با این حال هیچ الگوریتم خاصی برای به دست آوردن بهترین جواب برای تمام مسائل بهینه سازی وجود ندارد. برخی از الگوریتم ها جواب بهتری برای بعضی از مسائل خاص می دهند نسبت به بقیه الگوریتم ها. بنابراین جستجو برای الکوریتم های بهینه سازی جدید یک مساله پایان ناپذیر است. [۳۷]
در این پژوهش از یک الگوریتم بهینه سازی براساس قانون جاذبه، به اسم الگوریتم جستجوی گرانشی (GSA) استفاده می شود. این الگوریتم بر اساس قانون نیوتن است: " هر ذره ای در عالم هر ذره دیگر را با نیروی که به طور مستقیم به حاصل ضرب جرم آن و به طور معکوس با مربع فاصله بین آنها متناسب است جذب میکند. با توجه به اینکه این الگوریتم مبتنی بر قانون جاذبه است در ادامه قانون جاذبه را بیان کرده و در فصل بعدی الگوریتم مورد نظر به تفضیل شرح داده می شود. [۳۷]

٦-۲-۱ قانون جاذبه

گرانش، تمایل اجرام به سرعت بخشیدن به سمت یکدیگر است. که آن یکی از چهار نیروی بنیادی در طبیعت است. [۲۳](بقیه نیرو های بنیادی: نیروی الکترومغناطیس، نیروی ضعیف هسته ای، و نیروی قوی هسته ای) هر ذره در جهان هر ذره دیگر را جذب می کند. جاذبه همه جا هست. این قابلیت جاذبه باعث می شود آن از همه نیروهای طبیعی دیگر متفاوت باشد.
روش نیروی گرانش نیوتن به عنوان "کنش در فاصله" نامیده میشود. این به این معنی است که اعمال گرانش بین ذرات جدا بدون هیچ واسطه و بدون هر گونه تاخیر است. در قانون گرانش نیوتن، هر ذره هر ذرات دیگر را با یک نیروی گرانشی جذب می کند. [۲۳] و [۲٤]. نیروی گرانشی بین دو ذره به طور مستقیم متناسب با حاصل ضرب جرم آن اجرام و به طور معکوس با مجذور فاصله بین آنها متناسب است[۲٤]:

$$F = G\frac{M_1 M_2}{R^2} \qquad (12-2)$$

که در آن F اندازه نیروی گرانشی است، G ثابت گرانش است، M_1و M_2به ترتیب جرم ذرات اول و دوم است، و R فاصله بین دو ذره است. قانون دوم نیوتن می گوید که هنگامی که یک نیروی، F، به یک ذره اعمال می شود، شتاب آن، a، تنها بستگی به نیرو و جرم آن، M دارد. [۲٤]

$$a = \frac{F}{M} \qquad (13-2)$$

بر اساس (۲-۱۵) و (۲-۱۶) یک نیروی گرانش جذبی میان تمام ذرات جهان وجود دارد که تاثیر ذرات بزرگتر و نزدیک تر بالاتر است.

افزایش فاصله بین دو ذره به معنای کاهش نیروی جاذبه بین آنها است که در شکل ۲-۵ نشان داده شده است.

در این شکل، F_{1j} نیروی است که بر M_1 از طرف M_j وارد می شود و F_1 نیروی کلی اعمال شده بر روی M_1 است که باعث شتاب a_1 می شود.

علاوه بر این، به دلیل اثر کاهشی جاذبه، اندازه واقعی " ثابت گرانش "بستگی به سن کنونی جهان دارد که معادله زیر کاهش ثابت گرانش، G، را متناسب با سن نشان می دهد. [۳۸]

$$G(t) = G(t_0) \times \left(\frac{t_0}{t}\right)^{\beta} , \quad \beta < 1 \qquad (14-2)$$

که در آن $G(t)$ مقدار ثابت گرانش در زمان t است. $G(t_0)$ مقدار ثابت گرانش در اولین کوانتوم-فاصله کیهانی در زمان t_0است[۳۸].

سه نوع از جرم در فیزیک نظری تعریف می شود:

۱- جرم گرانشی فعال(M_a): یک اندازه گیری از قدرت میدان گرانشی به یک شیء خاص است. میدان گرانشی یک شی با جرم گرانشی فعال کوچک از یک شی با جرم گرانشی فعال تر، ضعیف تر است.

۲- جرم گرانشی منفعل(M_p): یک اندازه گیری از قدرت تعامل یک شی با میدان گرانشی است، در همان میدان گرانشی. و یک شی با جرم گرانشی منفعل کوچکتر، یک نیروی کوچکتر از یک شی با جرم گرانشی منفعل بزرگتر است را تجربه می کند.

۳- جرم اینرسی(M_i):یک اندازه گیری است از مقاومت یک شی نسبت به تغییرات حرکت آن وقتی که یک نیرو اعمال می شود. یک شی با جرم اینرسی بزرگ حرکت خود را آهسته تر تغییر می دهد، و یک شی با جرم اینرسی کوچک آن را با سرعت بیشتری تغییر می دهد.

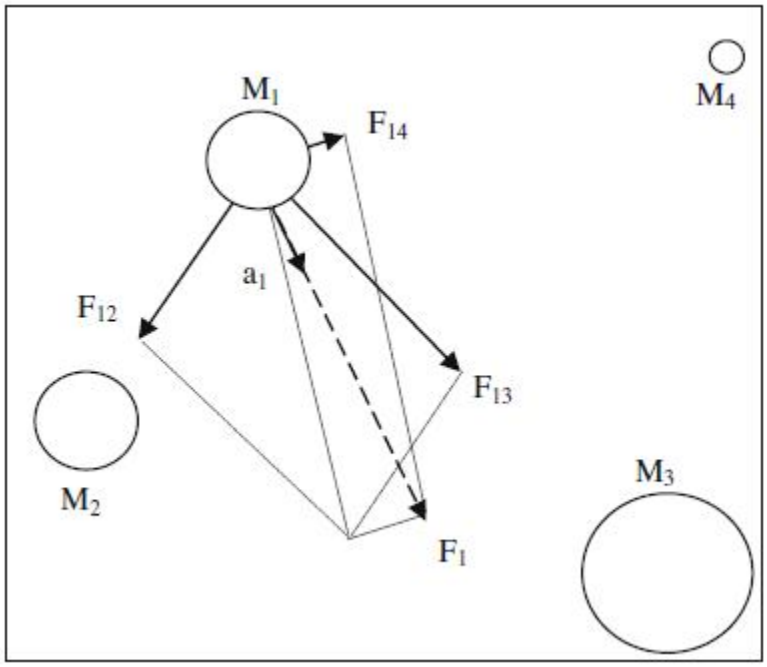
در اینجا با توجه به جنبه های ذکر شده در بالا، ما قوانین نیوتن را بازنویسی می کنیم. نیروی گرانشی، F_{ij}، که به جرم i از طرف جرم j اعمال می شود، با حاصلضرب جرم گرانشی فعال j و جرم گرانش منفعل i و معکوس فاصله مربع بین آنها متناسب است.

a_i با F_{ij} و با معکوس جرم اینرسی i متناسب است. به صورت دقیق تر معادله های ۲-۱۵ و ۲-۱۶ را دوباره می توان به این صورت بازنویسی کرد:

$$F_{ij} = G\frac{M_{aj} \times M_{pi}}{R^2} \qquad (15-2)$$

$$a_i = \frac{F_{ij}}{M_{ii}} \qquad (16-2)$$

که M_{aj} و M_{pi} به ترتیب نشان دهنده جرم گرانشی فعال ذره i و جرم گرانشی منفعل ذره j است و M_{ii} نشان دهنده جرم اینرسی ذره i است.



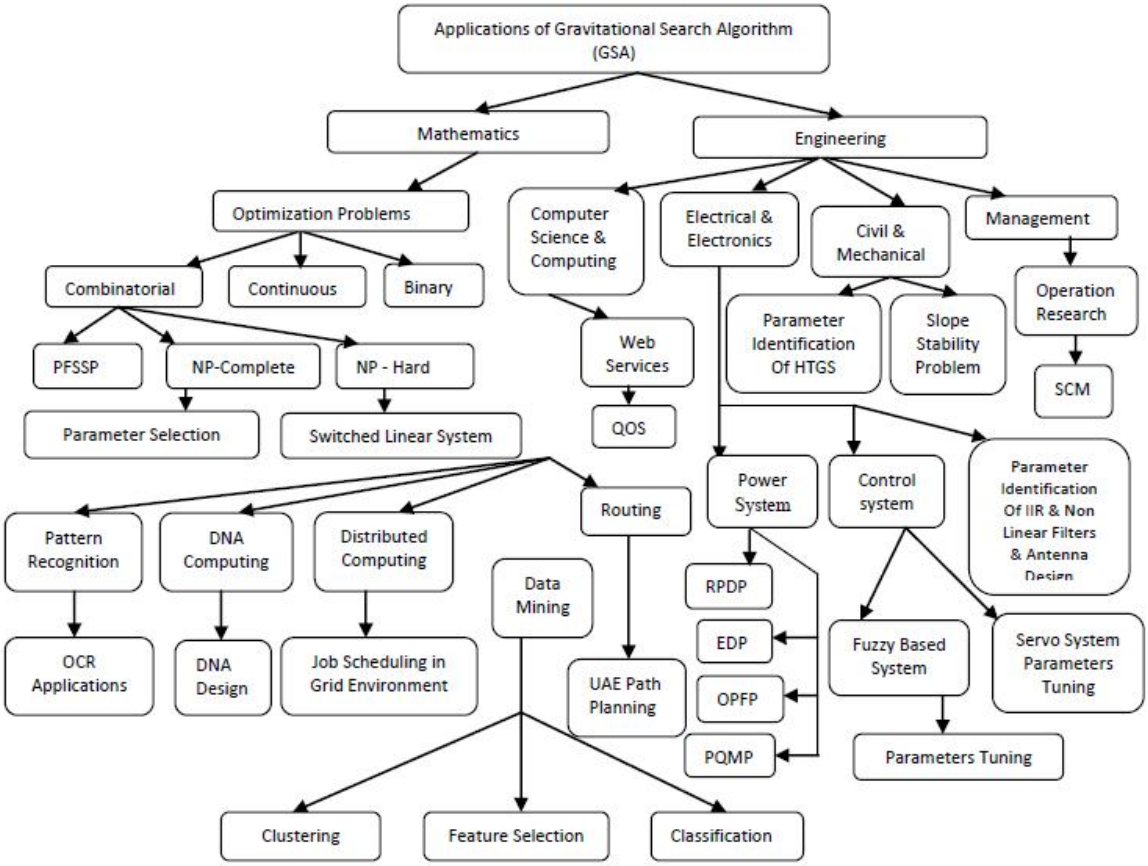
شکل ۲-۵- هر جسم تحت تاثیر نیروی گرانشی سایر اجسام شتابی می گیرد که با برآیند نیروهای وارد بر آن و عکس جرم اینرسی آن متناسب است. [۳۷]

اگر چه جرم اینرسی، جرم گرانشی منفعل و جرم گرانشی فعال مفاهیمی با یکدیگر متفاوت هستند، ولی هیچ آزمایشی تا به حال به روشنی تفاوتی بین آنها نشان نداده است. نظریه نسبیت عمومی بر این فرض استوار که جرم گرانشی اینرسی و جرم گرانشی منفعل معادل هم هستند. این به عنوان اصل هم ارزی ضعیف شناخته می شود. [۳۹]همچنین در نظریه نسبیت عمومی استاندارد هم ارزی جرم گرانشی اینرسی و جرم گرانشی فعال فرض می شود. این به عنوان اصل هم ارزی قوی نیز شناخته می شود. [۳۹]

۲-٦-۲ کاربرد الگوریتم جستجوی گرانشی

تا به حال، الگورینم جستجوی گرانشی بیشتر و بیشتر مورد توجه قرار گرفته شده است برای مسائل بهینه سازی چند هدفه، خوشه بندی داده، اختصاص متغییر ثابت جبرانی، تشخیص پارامترهای سیستم حاکم هیدرولیک، و غیره. [٤٠]

بسیاری از محققان الگوریتم جستجوی گرانشی را به علت آنکه آن فقط به دو پارامتر نیاز دارد و همچنین داشتن توانایی در پیدا کردن راه حل بهینه سراسری نزدیک و فراهم کردن نتایج بهتر نسبت به دیگر الگوریتم های الهام گرفته از طبیعت روی تعداد زیادی از مسائل به کار گرفته اند، تصویر ۲-٦ کاربرد الگوریتم جستجوی گرانشی در حوزه های مختلف را نشان می دهد. [۴۰]



شکل ۲-۶- کاربرد های الگوریتم GSA [۴۰]

در جدول ۲-۳ توسعه هایی که در زمینه کاری الگوریتم جستجوی گرانشی بوده و یا مسائلی که به GSA مرتبط شده اند آورده شده است. [۴۰]

Year	Author	Algorithm	Problem	Performance Metrics
2009	Rashedi et al (2009)	GSA	Optimization problems in continuous space	23 Non linear benchmark functions i.e. unimodal, multi model
2010	Rashedi et al (2010)	BGSA	Binary optimization Problems	23 Minimization and 2 maximization benchmark functions
2010	Hamid Reza Hassanzadeh et al (2010)	MOGSA	Multi Objective optimization problems	Spacing & gravitational distance criteria
2010	Xiangtao Li et al (2010)	SIGSA	Permutation flow shop scheduling problem	29 Problems of two classes of PFSSP such as car1, car2 to car8 etc.
2010	A. Chatterjeeet al (2010)	GSA	To improve Side lobe in concentric ring arrays(Antenna)	Fitness Value & Computational Time
2010	Seyedali Mirjalili et al (2010)	PSOGSA	To avoid slow convergence due to memory less feature of GSA	23 Non linear benchmark functions i.e. unimodal, multi model
2011	Jianhua Xiao and Zhen Cheng (2011)	GSA	DNA sequences optimization problem	Continuity, H- measure and Similarity
2011	S. Sarafrazi et al (2011)	Improved GSA	To increases the exploration and exploitation ability	23 Non linear benchmark functions
2011	M. Soleimanpour-moghadam et al (2011)	QGSA	Position and Velocity of an object in GSA	Unimodal & multimodal
2011	Saeid Saryazdi (2011)	GSA worked as heuristic search	Parameter estimation problem of IIR and Nonlinear rational filters	Unimodal function, multimodal function & complexity of problem
2011	Jianhua Xiao et al (2011)	GCSA	Partner selection problem with due date constraint	Cost criteria & convergence rate
2011	Radu-Emil Precup et al (2011)	GSA with modified deprecation equation	Tuning the fuzzy control systems	Performance indices
2011	Radu-Emil Precup et al (2011)	GSA with three modifications	To improve parametric sensitivity	Performance indices

2011	Serhat Duman et al (2011)	GSA	Economic Dispatch problem	Three test system
2011	J. P. Papa et al (2011)	OPF-GSA	Feature selection Task	Number of features selected
2011	M. Ghalambaz et al (2011)	HNNGSA	To solve wessinger's equation	……
2011	Chaoshun Li et al (2011)	IGSA	To identify the parameter for hydraulic turbine governing system	Frequency
2011	Abdolreza Hatamlou et al (2011)	GSA	Data clustering	Sum of intra cluster distance and Standard deviation
2011	Minghao Yin et al (2011)	IGSAKHM	Slow convergence problem of GSA in clustering	F- measure, run time, mean and standard deviation
2011	Abdolreza Hatamlou et al (2011)	GSA-HS	Clustering	Sum of intra cluster distance and center of corresponding cluster
2011	Soroor Sarafrazi et al (2011)	GSA-SVM	To increases classification accuracy in Binary Problems	Std. Dev., Mean, Max. & Min.
2012	Nihan Kazak et al (2012)	Modified GSA (MGSA).	To increases searching and convergence rate	3 Benchmark functions
				Such as unimodal, multimodal
2012	Mohadeseh Soleimanpour et al (2012)	Improved QGSA	Diversity loss problems	Unimodal & multimodal
2012	Lucian-Ovidiu Fedorovici et al (2012)	GSA with BP	OCR applications	Convergence and recognitions rates
2012	Nanji H. R et al (2012)	multi agent based GSA	Computational cost of original GSA	Six bench mark functions such as unimodal, multimodal
2012	Radu-Emil Precup et al (2012)	Adaptive GSA	To minimize the objective function	Sensitivity
2012	S. Duman et al (2012)	GSA	Reactive power dispatch problem	Minimization of system real power, improvement voltage profile & enhancement of voltage
2012	Serhat Duman et al (2012)	GSA	Optimal Power Flow problem	IEEE 30 Bus power system including six different cases
2012	Chaoshun Li et al (2012)	CGSA	Parameter identification problem of chaotic system	
2012	Abdolreza Hatamlou et al (2012)	GSA-KM	Clustering problem(For improve searching capabilities of GSA)	Number of steps and cost
2012	Chaoshun Li et al (2012)	GSAHCA	Identification of T-S fuzzy model	Model accuracy
2012	Hossein Askari et al (2012)	Intelligent GSA	Classification of data	Accuracy, minimum, maximum and average score of recognition
2012	Ahmad Asurl Ibrahim et al (2012)	QBGSA	Power quality monitor placement (PQMP) problem	Radial 69 bus distributed system and IEEE 118 bus system with Bolted three phase, Double line to ground and Single phase to ground
2012	Hamed Sadeghiet et al (2012)	GSA with sparse optimization	Identification of switch linear system and Parameter estimation	Statistical Robustness under excitation, noise and switching sequence
2012	Mohammad Khajehzadeh et al (2012)	MGSA	Slope Stability Analysis	Minimum factor of safety and reliability index
2012	Abbas Bahrololoum et al (2012)	prototype classifier based on GSA	Data classification	Average misclassification percentage error
2012	Li Pei et al (2012)	GSA with PSO & DE(IGSA)	Routing (Path planning for UAE)	Cost of threats & Evolution Curve
2013	Soroor Sarafrazi et al (2013)	GSA-SVM	Parameter Setting of SVM & Increased accuracy(Classification)	Accuracy (Mean & Standard Deviation)

جدول ۲-۱- لیست الگوریتم های ارائه شده به وسیله GSA [۴۰]

۲-۷ ترکیب سازی الگوریتم های فرا ابتکاری

فرصت های زیادی برای ترکیب الگوریتم های فرا ابتکاری ها با یکدیگر و (یا) با الگوریتم هایی از زمینه های دیگر وجود دارد. تعداد زیادی از نشریات در مورد منافع و موفقیت های بزرگ از این ترکیبات گزارش داده اند. در اینجا به مرور چندین روش ترکیبی محبوب خواهیم پرداخت و آنها را بر اساس ویژگی های مختلف طبقه بندی خواهیم کرد. به ویژه با توجه به ترکیبات سطح پایین فرا ابتکاری های مختلف، یک دیدگاه یکپارچه بر اساس یک قالب مشترک شرح داده شده است. این دیدگاه به ساخت شباهت ها و تفاوت های کلیدی مولفه های مختلف الگوریتم های فرا ابتکاری های موجود کمک میکند. سپس این اجزای کلیدی به عنوان یک جعبه ابزار برای ساخت، فرا ابتکاری ترکیبی موثر در نظر گرفته می شوند. [۷]

فرا ابتکاری ها ثابت کرده اند که برای حل تقریبی مسائل بهینه سازی در عمل دشوار بسیار مفید هستند.

یک دید کلی در این ناحیه از تحقیق را به عنوان مثال می توان در یافت، برای اطلاعات بیشتر همچنین [۲، ۳].

فرا ابتکاری برای اولین بار توسط گلوور معرفی شد.

امروز، الگوریتم های فرا ابتکاری به یک کلاس گسترده ای از مفاهیم الگوریتمک برای بهینه سازی و حل مساله اشاره می کنند، و مرزهای آنها تا حدودی فازی می باشند.

واب تعریف زیر را ارائه میدهد:

فرا ابتکاری یک فرایند تکراری است که عملیات های وابسته اکتشافی را در جهت تولید راه حل های با کیفیت بالا را هدایت و تغییر می دهد.

که ممکن است دستکاری یک راه حل مجزای کامل (یا ناقص) یا مجموعه ای از راه حل ها در هر تکرار باشند. تابع ابتکاری ممکن است روش های سطح بالا (یا پایین) باشند و یا یک جستجوی ساده محلی، و یا فقط یک روش ساخت.

مطابق با گفته گلاور:[٤١]

این روش ها در طول زمان شامل هر روشی برای حل مسائل از یک استراتژی برای غلبه بر به دام افتادن بهینه محلی در فضاهای جستجوی پیچیده استفاده کرده اند، به ویژه روش هایی که از یک یا چند ساختار همسایگی به عنوان ابزاری برای تعریف حرکت مجاز به انتقال از یک راه حل را به دیگری استفاده می کنند، و یا با ساخت یا نابود کردن راه حل در فرآیندهای سازنده و مخرب.

الگوریتم ژنتیک، الگوریتم انجماد تدریجی، بهینه سازی کلونی مورچگان و ... نمونه ای از الگوریتم های فرا ابتکاری هستند که هر کدام پس زمینه تاریخی و فلسفه خاص خود را دارند. [٤١]

به خصوص در سال گذشته تعداد زیادی از الگوریتم های گزارش شده است که صرفا از مفاهیم یک الگوریتم فرا ابتکاری سنتی استفاده نمی کنند، بلکه آنها ایده های مختلف الگوریتمیک، و حتی گاهی اوقات نیز خارج از شاخه الگوریتم های فرا ابتکاری سنتی را ترکیب می کنند. این روش ها معمولا به عنوان فرا ابتکاری های ترکیبی[۱۵۳] نامیده می شوند.

انگیزه چنین ترکیبانی از مفاهیم الگوریتمیک مختلف معمولا برای به دست آوردن سیستم اجرایی بهتری است که برای بهره برداری و تجمیع مزایای استراتژی های خالص تکی هر کدام، مورد استفاده قرار می گیرند.

افزایش بسیار تعداد کابرد های گزارش شده از فرا ابتکاری های ترکیبی و رویدادهای علمی اختصاص داده شده مانند مجموعه ای از کارگاه های آموزشی در مورد فرا ابتکاری های ترکیبی نشان از محبوبیت، موفقیت، و اهمیت این بخش خاص از تحقیقات است.

در واقع، امروز به نظر می رسد که انتخاب یک روش ترکیبی مناسب برای دستیابی به عملکرد بالا در حل مسائل دشوار تعیین کننده است.

چندین نشریه وجود دارد که طبقه بندی برای فرا ابتکاری های ترکیبی و یا زیر شاخه خاص را دارند. بخش زیر در ادامه برای ادغام مهمترین جنبه های این طبقه بندی ها و در برخی از قسمت ها گسترش این دیدگاه تلاش می کنیم

شکل ۲-۷ طبقات و خواص مختلف که برای دسته بندی ترکیبی فرا ابتکاری ها استفاده می شود نشان می دهد. بدینوسیله، ما جنبه های طبقه بندی توسط طالبی با نقطه نظرات کوتا و بلوم و همکاران ترکیب می کنیم. [٤١] ما با تشخیص اینکه چه چیز را ترکیب میکنیم شروع میکنیم، ما ممکن است حالت های زیر را بخواهیم ترکیب کنیم:

۱) استراتژی های مختلف فرا ابتکاری ها
۲) فرا ابتکاری ها با الگوریتم های خاص برای مساله مورد نظر ما، مانند شبیه سازی های خاص
۳) فرا ابتکاری با دیگر تکنیک های کلی تر برگرفته از زمینه هایی مانند تحقیق در عملیات (OR) و هوش مصنوعی (AI)

از نمونه های برجسته برای روش های بهینه سازی از زمینه های دیگر که با موفقیت با فرا ابتکاری ترکیب شده اند روش های دقیق مانند شاخه و کران، برنامه نویسی پویا و تکنیک های مختلف خاص برنامه نویسی خطی عدد صحیح در یک سمت، و روش محاسباتی نرم مانند شبکه های عصبی و منطق فازی در طرف دیگر هستند.

153 hybrid metaheuristics

علاوه بر این تمایز، طبقه بندی های فرا ابتکاری های ترکیبی براساس درجه تمایز سطح (و یا قدرت) وجود دارد که در آن الگوریتم های مختلف هم ترکیب می شوند:

۱) ترکیب سطح بالا در اصل هویت فردی از الگوریتم های اصلی را حفظ می کند و همکاری با یک رابط، نسبتا به خوبی تعریف شده است ؛ هیچ رابطه قوی مستقیمی از فعالیت های داخلی الگوریتم ها وجود ندارد.

۲) در مقابل، الگوریتم ها در ترکیب سطح پایین به شدت به یکدیگر وابسته هستند – اجزاء منحصر به فرد و یا توابع الگوریتم ها بین هم رد و بدل می شوند.

ویژگی دیگری که به وسیله آن ممکن است سیستم های ترکیبی را تقسیم بندی کنیم ترتیب اجرا[۱۵۴] است.

در مدل دسته ای[۱۵۵]، یک الگوریتم اکیدا پس از الگوریتم دیگری اجرا می شود، و اطلاعات فقط در یک جهت انتقال داده می شود. پردازش هوشمند از داده های ورودی و یا پس پردازی نتایج حاصل از یک الگوریتم دیگر در این دسته قرار می گیرند. مثال دیگر، مسائل چند سطحی است که با در نظر گرفتن یک سطح پس از دیگری توسط الگوریتم های بهینه سازی اختصاص یافته حل می شوند.

در مقابل، ما باید مدل های لایه ای[۱۵۶] و موازی[۱۵۷] را داریم که در آنها الگوریتم ممکن است با روش های پیچیده تری تعامل داشته باشد. فرا ابتکاری های موازی امروزه یک حوزه پژوهشی بزرگ و مهم برای خود هستند. با بررسی خصوصیات کلی الگوریتم های موازی، ما می توانیم معماری را متمایز کنیم:SIMD): دستورات واحد، در مقابل جریان داده چندگانه، MIMD: دستورات چند گانه، جریان های داده چند گانه، تقسیم بندی موازی (ریز در مقابل درشت)، سخت افزار (ناهمگن در مقابل همگن)، استراتژی حافظه (حافظه مشترک در مقابل حافظه توزیع شده)

وظیفه و استراتژی تخصیص داده (دینامیک در مقابل پویا) و این که آیا وظایف مختلف هماهنگ شده و همزمان هستند و یا از روش ناهمزمان اجرا می شوند.

نوع دیگر طبقه بندی فرا ابتکاری های ترکیبی تقسیم بندی براساس استراتژی های کنترل خودشان هستند. با توجه به [۹ و ۱۳]، ترکیبات یکپارچه (اجباری) و مشترک (تعاونی) وجود دارد.

در روش های یکپارچه، یک الگوریتم به عنوان یک تابع در نظر گرفته شده و به عنوان جزء از الگوریتم دیگر جاسازی می شود. این رویکرد بسیار محبوب است.

---

154 order of execution
155 batch
156 interleaved
157 parallel

برای مثال، در الگوریتم ممتیک[۱۵۸]، انواع مختلف از جستجوی محلی در یک الگوریتم تکاملی برای بهبود محلی راه حل های به دست آمده از عملگر های متنوع تعبیه شده است.

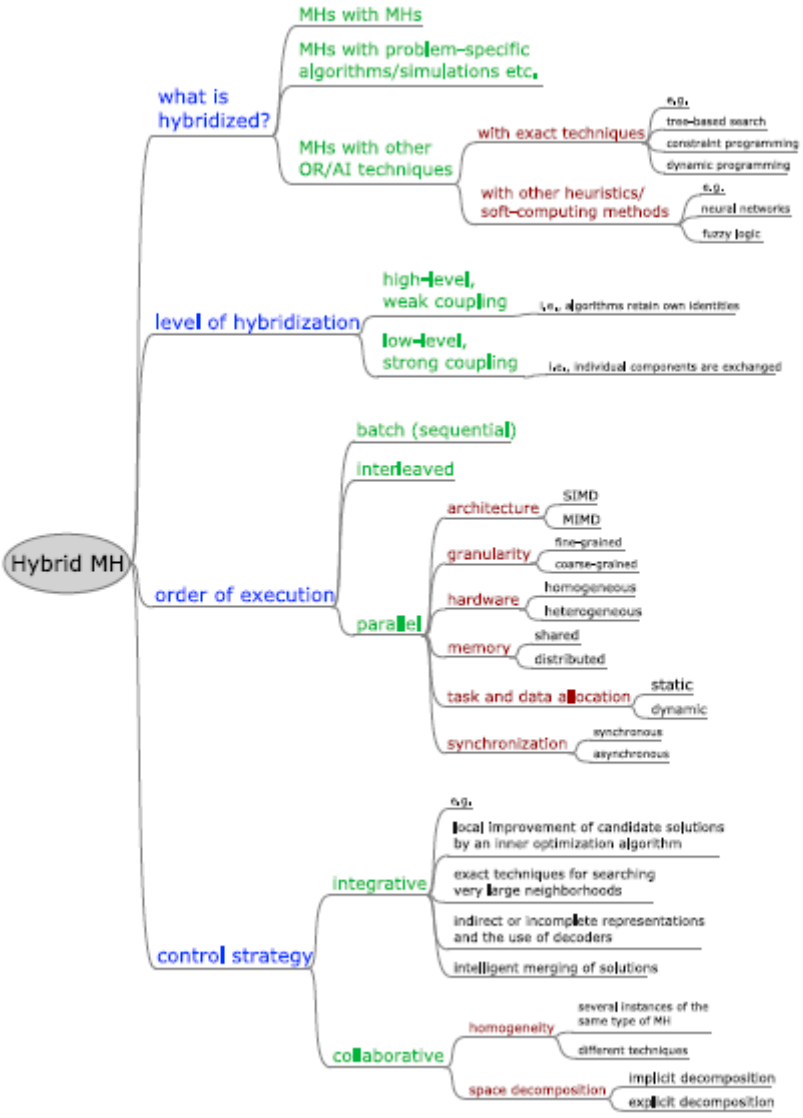
جستجوی مقیاس بسیار بزرگ همسایه[۱۵۹] (VLSN) مثال دیگری از آن است.

ادغام راه حل: در روش های مبتنی بر جمعیت مانند الگوریتم های تکاملی و یا جستجو پراکنده، یک اپراتور تغییر سنتی، یک نوترکیب است. آن یک راه حل جدید را با ترکیب ویژگی های دو (یا بیشتر) راه حل های والد مشتق میک ند. به خصوص در الگوریتم های ژنتیک کلاسیک، این عملگر براساس تصمیم گیری های تصادفی خالص است و در نتیجه بدون بهره برداری از هرگونه دانش خاص مساله کار می کند. گاهی اوقات، این روش توسط الگوریتم های قوی تر جایگزین می شوند مانند مسیر=پیوند داده و یا با استفاده از روش دقیق براساس شاخه و کران یا برنامه ریزی خطی عدد صحیح است که شناسایی یک بهترین ترکیب از ویژگی های والد است.

در ترکیب مشترک، الگوریتم ها تبادل اطلاعات می کنند، اما بخشی از یکدیگر نیستند. به عنوان مثال، مدل جزیره محبوب برای موازی سازی الگوریتم های تکاملی در این دسته قرار می گیرد. [۷]

[158] memetic

[159] Very large scale neighborhood search



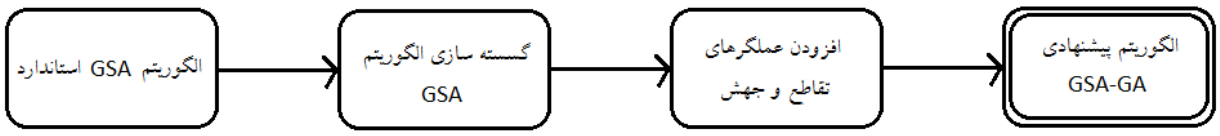
شکل ۲-۷- طبقه بندی فرا ابتکاری های ترکیبی [۷]

در این پژوهش نیز الگوریتم پیشنهادی، ترکیبی از الگوریتم های چستجوی گرانشی و الگوزیتم ژنتیک می باشد تا از نقاط قوت هر دو الگوریتم استفاده کند که در فصل بعدی آن را بررسی خواهیم کرد.

فصل سوم :روش پیشنهادی

۳-۱ مقدمه

پس از توضیح مقدمات در این فصل روش پیشنهادی برای حل مساله QAP ارائه می گردد. شمای کلی الگوریتم پیشنهادی را در شکل ۳-۱ مشاهده می کنید.در ابتدا الگوریتم جستجوی گرانشی (که مبنای این روش پیشنهادی می باشد) بیان می گردد و سپس با توجه به اینکه این الگوریتم ذاتا برای مسائل پیوسته طراحی شده با تکنیک به کار رفته، آن را برای مسائل گسسته (که QAP نمونه ای از آن هاست) قابل استفاده می کنیم.همچنین به جهت بهبود کارایی الگوریتم GSA از دو عملگر تقاطع و جهش مربوط به الگوریتم ژنتیک استفاده می کنیم تا الگوریتم پیشنهادی ما کامل گردد.



شکل ۳-۱- شمای کلی الگوریتم پیشنهادی

۳-۱ الگوریتم جستجوی گرانشی

در طول دهه های اخیر الگوریتم های الهام گرفته شده از طبیعت به طور وسیعی برای حل مسائل بهینه سازی مختلف استفاده شده است. در بین آنها الگوریتم های هوش جمعی و تکاملی به طور گسترده ای به عنوان ابزاری برای جستجو و بهینه سازی مورد استفاده قرار گرفته است. در دامنه هایی از مسائل مختلف، هوش جمعی یک روش هوش مصنوعی است که در رابطه با رفتار جمعی در سیستم های غیر متمرکز می باشد. الگوریتم های بهینه سازی کلونی مورچه، بهینه سازی ازدحام ذرات، الگوریتم زنبور و کلونی زنبور مصنوعی[۱۶۰] برخی از معروف ترین آنها هستند. این الگوریتم های در مسائل دنیای واقعی می تواند استفاده شوند. [۲۸]

[160] Artificial Bee Colony

الگوریتم های هوش جمعی می تواند گزینه ای مناسب برای حل مسائل بهینه یابی در محیط های پویا باشند، در این پژوهش از یک الگوریتم هوش جمعی مبتنی بر قانون جاذبه و بازخورد اجرام استفاده شده که در ادامه آن را معرفی می کنیم. [٤٢]

در الگوریتم جستجوی گرانشی، عامل ها به صورت اشیاء در نظر گرفته می شوند و عملکردشان با جرم آنها اندازه گیری می شود. تمام این اشیاء یکدیگر را جذب می کنند توسط نیروی جاذبه، و این نیرو باعث یک حرکت سراسری از تمام اشیاء به سمت اشیاء با جرم سنگین تر می شود. از این رو، اجسام با استفاده از یک نوع ارتباط مستقیم، از طریق نیروی گرانشی باهم تعامل و همکاری دارند. اجرام سنگین تر (که متناظر با راه حل های خوب هستند) آهسته تر از آنهایی که سبک تر هستند حرکت می کنند، که این مرحله بهره برداری از الگوریتم را تضمین می کند.

در GSA، هر جرم (عامل) دارای چهار مشخصات: موقعیت، جرم اینرسی، جرم گرانشی فعال، و جرم گرانش منفعل است. موقعیت هر جرم مربوط به یک راه حل از مساله است، و جرم گرانشی و اینرسیایی آن با استفاده از یک تابع ارزیابی تعیین می شود.

به عبارت دیگر، هر جرم یک راه حل را نشان می دهد، و الگوریتم با تنظیم گرانشی درست و اجرام اینرسی جهت یابی می شود. به مرور زمان، انتظار می رود که اجرام توسط سنگین ترین جرم جذب شوند. این جرم یک راه حل بهینه در فضای جستجو ارائه می دهد.

GSA می تواند به عنوان یک سیستم ایزوله اجرام در نظر گرفته شود. آن مانند یک دنیای مصنوعی کوچکی از اجرام هست که از قوانین نیوتنی گرانش و حرکت پیروی می کند. دقیق تر، اجرام از قوانین زیر پیروی می کند:

قانون جاذبه:هر ذره بقیه ذرات را جذب میکند و نیروی گرانش بین دو ذره به طور مستقیم متناسب با حاصلضرب جرم آنها و با معکوس فاصله بین آنها متناسب است. ما در اینجا از R به جای R^2، استفاده میکنیم چرا که با توجه به نتایج آزمایش ها، R نتایج بهتری نسبت به R^2 در تمام موارد تجربی را فراهم می کند. [۳۷]

قانون حرکت:سرعت کنونی هر جرم برابر است با مجموع ضریبی از سرعت قبلی آن و تغییرات در سرعت (شتاب) آن می باشد. تغییر در سرعت یا شتاب هر جرم برابر است با نیروی وارد بر روی سیستم تقسیم بر جرم اینرسی.

الگوریتم جستجوی گرانشی در زمره الگوریتم های مبتنی بر هوش جمعی طبقه بندی می شود. که یک فضای جستجوی چند بعدی را برای پیدا کردن مقدار اکسترمم تابع هدف مورد جستجو قرار می دهد.

در این الگوریتم از مشابهت حرکت سینماتیک و کلاسیک اجرام در میدان گرانشی الهام گرفته شده است.

الگوریتم جستجوی گرانشی در واقع با استفاده از طرح قوانین گرانش و حرکت در یک سیستم مصنوعی با زمان گسسته ساخته می شود که در آن عامل های جستجو کننده مجموعه ای از اجرام هستند که هر جرم در واقع محل و وضعیت سایر اجرام را درک می کند و به طریقی می تواند با توجه به تبادل نیرو بین اجرام دیگر،به تبادل اطلاعات پرداخته و سیستمی مصنوعی را ایجاد کند. محیط سیستم همان محدوده تعریف مسأله می باشد. طبق قانون گرانش هر جرم محل و وضعیت سایر اجرام را از طریق نیروی جاذبه گرانشی درک میکند. بنابراین میتوان از این نیرو به عنوان ابزاری برای تبادل اطلاعات استفاده کرد. از بهینه یاب طراحی شده، می توان برای حل هر مسأله بهینه سازی که در آن هر جواب مسأله به صورت یک موقعیت در فضا قابل تعریف است و میزان شباهت آن با سایر جوابهای مسأله به صورت یک فاصله قابل بیان باشد، استفاده نمود. اندازه اجرام با توجه به تابع هدف تعیین می شود. در قدم اول فضای سیستم مشخص میشود. محیط شامل یک دستگاه مختصات چند بعدی در فضای تعریف مسأله است. هر نقطه از فضا، یک جواب مسأله است. عاملهای جستجو کنده، مجموعه ای از اجرام میباشند. هر جرم چهار مشخصه دارد:

الف)موقعیت جرم، ب)جرم گرانشی فعال، ج)جرم گرانشی غیر فعال و د)جرم اینرسی.

اجرام فوق برگرفته از مفاهیم جرم گرانشی فعال و جرم اینرسی در فیزیک میباشند. اجرام گرانشی و اینرسی با الهام از مفاهیم فیزیک نیوتنی تعریف شدهاند و با توجه به برازندگی عامل ها تعیین می شوند. جرم گرانشی فعال، معیاری از میزان شدت نیروی گرانشی حول یک جسم است. جرم گرانشی غیر فعال نشان دهنده قدرت اثر متقابل در میدان گرانشی است. جرم اینرسی نیز معیاری از مقاومت شیء در قابل تغییر موقعیت مکانی و حرکت است. پس از تشکیل سیستم، قوانین حاکم بر آن مشخص می شوند. فرض می کنیم تنها قانون گرانش و قوانین حرکت حاکمند. صورت کلی این قوانین تقریبا شبیه قوانین طبیعت است و به صورت زیر تعریف شده اند.

بردار موقعیت اولیه شامل مجموعه از N جرم است که موقعیت هر جرم نقطه ای است که می تواند شامل d بعد باشد. موقعیت i امین عامل به صورت زیر تعریف می شود:

$$X_i = (X_i^1, \dots X_i^d, \dots X_i^n) \qquad (1\text{-}3)$$

که در آن X_i^d موقعیت عامل i ام در بعد d ام می باشد.

در این سیستم در زمان t و در هر بعد d نیرویی به اندازه $F_{ij}^d(t)$ از سوی جرم j به جرم i وارد می شود. مقدار این نیرو از رابطه زیر به دست می آید:

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi}(t)\times M_{aj}(t)}{R_{ij}(t)+\varepsilon}(x_j^d(t) - x_i^d(t)) \qquad (2\text{-}3)$$

که در رابطه بالا $M_{aj}(t)$ مقدار جرم گرانشی موثر مربوط به عامل j و $M_{pi}(t)$ جرم گرانشی منفعل مربوط به عامل i می باشد. و $G(t)$ مربوط به ثابت گرانش در زمان t و $R_{ij}(t)$ فاصله اقلیدسی بین عامل های i و j می باشد. ε نیز یک مقدار ثابت کوچک می باشد.

فاصله اقلیدسی به صورت زیر تعریف می شود:

$$R_{ij}(t) = ||X_i(\mathrm{t}), X_j(\mathrm{t})||_2 \qquad (3-3)$$

برای حفظ تصادفی بودن الگوریتم، نیروی وارد بر ذره i ام در بعد d ام از طرف هر ذره را در یک عدد تصادفی با توزیع یکنواخت در بازه [۰. ۱] ضرب کرده و کل نیروی وارد بر ذره i ام در بعد d ام حاصل جمع نیروهای وارده از طرف همه ی ذره ها به جز ذره i ام در گرفته می شود.

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j\, F_{ij}^d(t) \qquad (4-3)$$

با توجه به قانون دوم نیوتن چنانچه به ذره ای نیرو وارد شود، متناسب با آن نیرو شتاب می گیرد. بنابراین براساس قانون حرکت شتاب ذره i ام در زمان t و در جهت بعد d ام به این صورت به دست می آید:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \qquad (5-3)$$

که در آن $M_{ii}(t)$ جرم لختی مربوط به ذره i ام است.

سرعت بعدی یک ذره ضریبی از سرعت فعلی و شتاب فعلی ذره است. بنابراین موقعیت و سرعت ذره از روابطه زیر به دست می آید:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \qquad (6-3)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \qquad (7-3)$$

که در این بخش نیز $rand_i$ یک متغییر تصادفی یکنواخت در بازه [۰. ۱] برای این در نظر گرفته شده است که خاصیت تصادفی جستجو از بین نرود.

ثابت گرانش G در ابتدا مقداردهی می شود و سپس با به منظور دقت جستجو کاهش می یابد. به عبارت دیگر G تابی از مقدار اولیه(G_0) و زمان (t) می باشد:

$$G = (G_0, t) \qquad (8-3)$$

جرم های گرانشی و لختی با توجه به مقادیر تابع برازندگی محاسبه می شوندجرم سنگین تر به معنی ذره موثر تر می باشد. با فرض برابر بودن اجرام گرانشی و لختی مقدار اجرام با استفاده از نگاشت ارزیابی محاسبه می شوند:

$$M_{ai}(t) = M_{pi}(t) = M_{ii}(t) = M_i(t) \qquad (9-3)$$

$$m_i(t) = \frac{Value_i(t) - worst(t)}{best(t) - worst(t)} \qquad (10-3)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \qquad (11-3)$$

که در روابط بالا $Value_i(t)$ مقدار برازندگی ارزیابی برای ذره i ام در زمان t است. همچنین$best(t)$ و $worst(t)$ که به ترتیب نشان دهنده برازندگی قوی ترین و ضعیف ترین ذره های جمعیت در زمان t هستند که با استفاده از روابط زیر به دست می آیند:

$$best(t) = min_{j\in\{1,\dots,N\}} fit_j(t) \qquad (12-3)$$

$$worst(t) = max_{j\in\{1,\dots,N\}} fit_j(t) \qquad (13-3)$$

باید در نظر داشته برای مسائل حداکثرسازی روبط بالا به صورت زیر تغییر می کنند:

$$best(t) = max_{j\in\{1,\dots,N\}} fit_j(t) \qquad (14-3)$$

$$worst(t) = min_{j\in\{1,\dots,N\}} fit_j(t) \qquad (15-3)$$

عامل توقف در GSA همان G(t) یا ثابت گرانش است که در آن G_0 وα مقادیر ثابتی در نظر گرفته شده اند و T و t به ترتیب بیانگر کل تکرار ها و تکرار فعلی است.

$$G(t) = G_0 . e^{-\alpha \frac{t}{T}} \qquad (16-3)$$

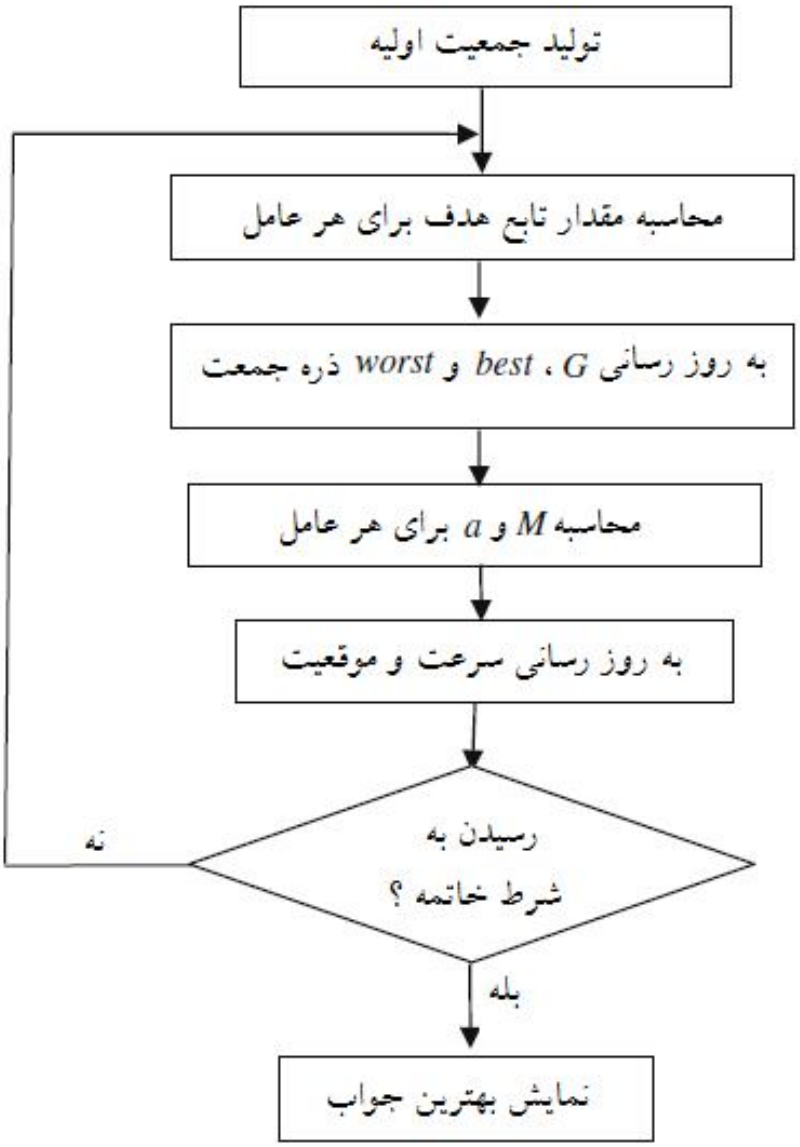
در ابتدای تشکیل سیستم، هر جسم به صورت تصادفی در یک نقطه از فضا قرار میگیرد که جوابی از مسأله است. در هر لحظه از زمان، اجرام ارزیابی شده سپس، تغییر مکان هر جرم محاسبه می شود. پارامترهای سیستم

شامل محاسبه اجرام گرانشی، جرم لختی و ثابت گرانش نیوتن هستند که در هر مرحله بروز رسانی می شوند. شرط توقف می تواند پس از طی مدت زمان مشخصی تعیین شود.

الگوریتم جستجوی گرانشی باید به گونه ای هدایت شود که موقعیت اجرام با گذشت زمان بهبود پیدا کند. استراتژی به کار گرفته شده برای این منظور مبنی بر تنظیم جرم عامل ها است. طبق روابط ارائه شده، تمام عامل ها متناسب با جرمشان روی هم تاثیر می گذارند و اثر گذاری آنها روی عامل های همسایه بیشتر است. تاثیر عامل های با جرم سنگین تر بیشتر است و شعاع تاثیر گذاری بزرگتری دارند. بنابراین به اجرامی که تابع برازش بهتری دارند، جرم گرانشی بهتری نسبت داده می شود. در نتیجه هر جرم به اندازه شایستگی خود، سایر اجرام را به سمت خود دعوت می کند. بنابراین با گذشت زمان اجرام به سمت موقعیت های مناسب اتر می روند.
از آنجا که این الگوریتم حافظه دار نیست، به همین دلیل هیچ یک از موقعیت های دیده شده قبلی را به خاطر نمی سپارد. با وجود این دیده می شود که عملکردی به خوبی الگوریتم های حافظه دار دارد.
یکی از راه های ایجاد یک تعادل خوب بین اکتشاف و بهره برداری کاهش تعداد عامل ها با گذشت زمان در رابطه ۹ است. از این رو، ما تنها مجموعه ای از عامل ها با جرم بزرگتر را برای اعمال نیروی خود به دیگر اجرام پیشنهاد می کنیم. با این حال، باید مراقب استفاده از این سیاست بود به دلیل آن که ممکن است قدرت اکتشاف را کاهش و قابلیت بهره برداری افزایش دهد.
یادآوری می کنیم که به منظور جلوگیری از به دام افتادن در بهینه محلی الگوریتم باید در آغاز از اکتشاف استفاده کند. به مرور تکرار، تاثیر اکتشاف باید کمرنگ شود و تاثیر بهره برداری پررنگ تر شود. برای بهبود عملکرد الگوریتم جستجوی گرانشی با کنترل اکتشاف و بهره برداری تنها تعداد Kbest عامل اجرام دیگر را جذب می کند. Kbest یک تابع از زمان است، با مقدار اولیه K_0 آغاز می شود و با گذشت زمان کاهش می یابد. در این روش، در آغاز، تمام عامل ها نیرو اعمال می کنند، و با گذشت زمان، Kbest به صورت خطی کاهش می یابد و در پایان فقط نیروی اعمال شده از سوی یک عامل به عامل های دیگر وجود دارد خواهد شد. [۳۷]
رابطه ۹ بصورت زیر اصلاح می شود:

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i}^{N} rand_j \, F_{ij}^d(t) \qquad (17-3)$$

که در آن Kbest مجموعه ای از K عامل اول با بهترین مقدار تابع ارزیابی و بزرگترین اجرام هستند.قاعده کلی الگوریتم GSA در شکل زیر نشان داده شده است:



شکل ۳-۲- قاعده کلی و عمومی الگوریتم GSA [۳۷]

برای اینکه که چگونگی کارآمد بودن الگوریتم پیشنهاد شده را دید برخی از نکات اشاره شده است:

– از آنجا که هر عامل می تواند عملکرد دیگران را مشاهده کند، نیروی گرانشی یک ابزار انتقال اطلاعات است.

– با توجه به نیروهایی که از طرف عامل های همسایه خود بر روی یک عامل اعمال می شود، می تواند فضای اطراف خود را ببیند.

– اجرام سنگین دارای شعاع بزرگ از جاذبه های موثر هستند و از این رو به شدت جاذبه زیادی دارند. بنابراین، عامل های با عملکرد بالاتر یک جرم گرانشی بیشتری دارند. در نتیجه، عوامل تمایل به حرکت به سمت بهترین عامل دارند.

– جرم لختی در برابر حرکت است و حرکت اجرام را کند می کند. از این رو، عامل های با جرم لختی سنگین به آرامی حرکت می کنند از این رو فضا جستجو بیشتر در در سطح محلی است. بنابراین، می توان آن را به عنوان یک نرخ یادگیری انطباقی در نظر گرفت.

– ثابت گرانش دقت و صحت جستجو را تنظیم می کند، بنابراین مقدار آن با گذشت زمان کاهش می یابد(مشابه دما در الگوریتم انجماد تدریجی)

– GSA یک الگوریتم بدون–حافظه است. با این حال،آن مانند الگوریتم های با حافظه به صورت موثر کار می کند. نتایج تجربی نشان از نرخ همگرایی خوب برای GSA نشان می دهد.

– در اینجا، ما فرض کنیم که جرم گرانشی و جرم لختی یکسان هستند. با این حال، برای برخی از کاربرد ها، مقادیر متفاوت برای آنها می تواند استفاده شود. یک جرم لختی بزرگتر یک حرکت آهسته برای عامل ها در فضای جستجو را فراهم می کند و از این رو جستجو دقیق خواهد بود. در مقابل، یک جرم گرانشی بزرگتر باعث جذب بیشتر عامل ها می شود. این اجازه همگرایی سریع تر را می دهد. [۳۷]

1. Search space identification, t=0;
2. Random initialization, $Xi(t)$;
For $i=1, ..., N$
3. Fitness evaluation of objects;
4. Update the parameters of *G, best, worst and M;*
For $i=1, ..., N$
5. Calculation of the force on each object;
6. Calculation of the acceleration and the velocity of each object;
7. Update the position of the agents by (7-3) to yield $Xi(t+1)$;
t=t+1;
8. Repeat steps 3 to 7 until the stop criteria is reached;
9. end

شکل ۳-۳- شبه کد الگوریتم جستجوی گرانشی

۳-۳ الگوریتم جستجوی گرانشی برای مسائل گسسته

نسخه اصلی توسعه داده شده ی الگوریتم جستجوی گرانشی برای جستجو راه حل در یک فضای پیوسته طراحی شده است. به منظور استفاده از آن برای مساله تخصیص درجه دوم، که در واقع یک مساله در فضای حالت گسسته است از تکنیک کلیدهای تصادفی[۱۶۱] استفاده می کنیم.

تکنیک کلید های تصادفی را می توان برای تبدیل یک موقعیت در یک فضای خالی پیوسته به فضای گسسته مورد استفاده قرار داد. [۴۳]

یک بردار در فضای کلید تصادفی متشکل از اعداد واقعی است. مطابق با تکنیک کلید تصادفی، یک جرم به وسیله اعداد حقیقی که می توانند یک عملیات جایگشت متشکل از اعداد گسسته را شبیه سازی ارائه می شود.

در روش کلید تصادفی، ما به هر موقعیت یک جرم یک عدد تصادفی یکنواخت در بازه [۰، ۱) اختصاص می دهیم. برای رمزگشایی موقعیت از فضای کلید تصادفی به فضای راه حل واقعی، ما گره ها را به ترتیب صعودی مقدار هر بعدشان مشاهده می کنیم. یک نمونه به صورت زیر نشان داده شده است:

کلید تصادفی:	0.42	0.06	0.38	0.48	0.81
رمزگشایی شده:	3	1	2	4	5

شکل ۳-۴- مثالی از تکنیک کلید تصادفی [۴۳]

گره هایی که باید در مراحل اولیه تور باشند مقدار بُعدشان تمایل دارد که به ۰ نزدیک تر باشد و آنهایی که در ادامه می آیند مقدار بعدشان به ۱ نزدیک است.

با انجام این کار،موقعیت یک شی در الگوریتم جستجوی گرانشی $X_i = (X_i^1, \dots X_i^d, \dots X_i^n)$ را می توان به یک راه حل معتبر در فضای جستجو مساله تخصیص درجه دوم تبدیل کرد، [۴۳] یعنی:

$$V = (v_1, \dots, v_k, \dots, v_N) \qquad (18\text{-}3)$$

نتایج الگوریتم GSA بر روی مسائل منتخب QAP

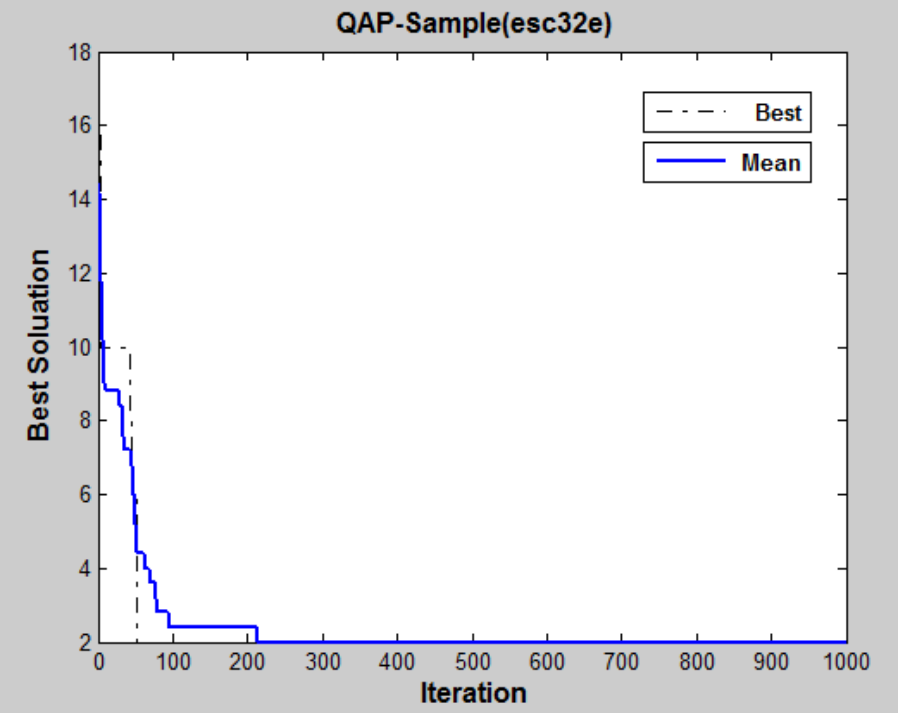
در این قسمت نتایج اجرای الگوریتم جستجوی گرانشی بر روی ۱۰ مساله منتخب از مسائل موجود در مرجع QAPLIB آورده شده است.(الگوریتم مورد نظر ۱۰ بار بر روی هر مساله انجام شده است.)

[161] Random-key Encoding

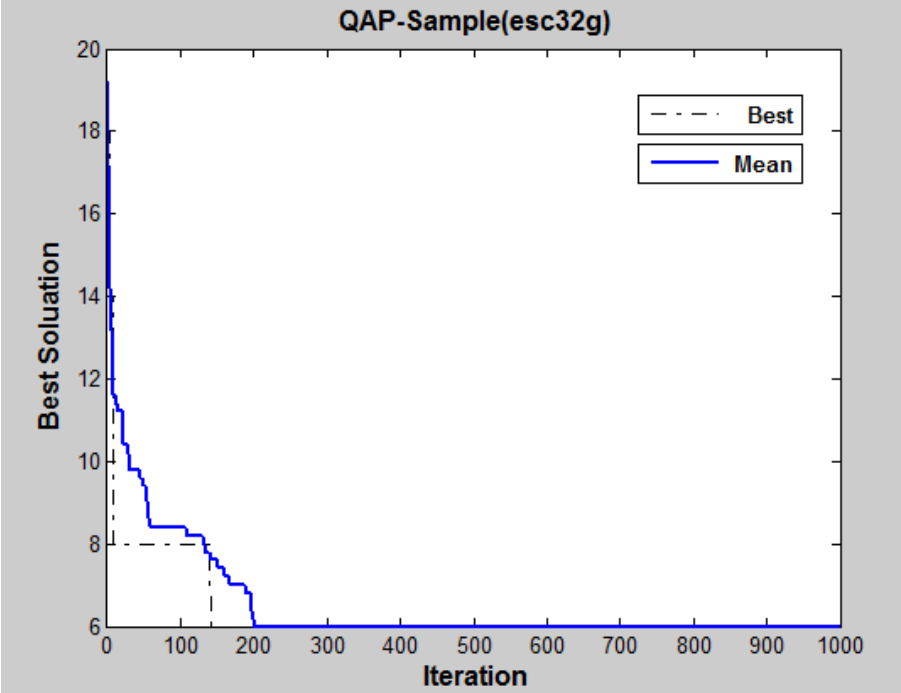
نام مساله	ابعاد مساله	بهترین جواب به دست آمده تاکنون	بهترین جواب GSA	میانگین جواب GSA	درصد خطای بهترین	درصد خطای میانگین
Esc32e	32	2	2	2	0	0
Esc32g	32	6	6	6	0	0
Esc32h	32	438	472	490	7.76	11.87
Esc64a	64	116	128	137	10.3	18.1
Tai64c	64	1855928	1877754	1945100	1.17	4.80
Lipa40b	40	476581	581484	585960	22.01	22.95
Sko49	49	23386	24322	24546	4.00	4.96
Wil50	50	48816	50192	50622	2.81	3.69
Lipa70a	70	169755	172258	172400	1.47	1.55
Lipa80a	80	253195	256636	256900	1.35	1.46

جدول ۳-۱- نتایج اجرای الگوریتم GSA بر روی ده مساله منتخب QAP

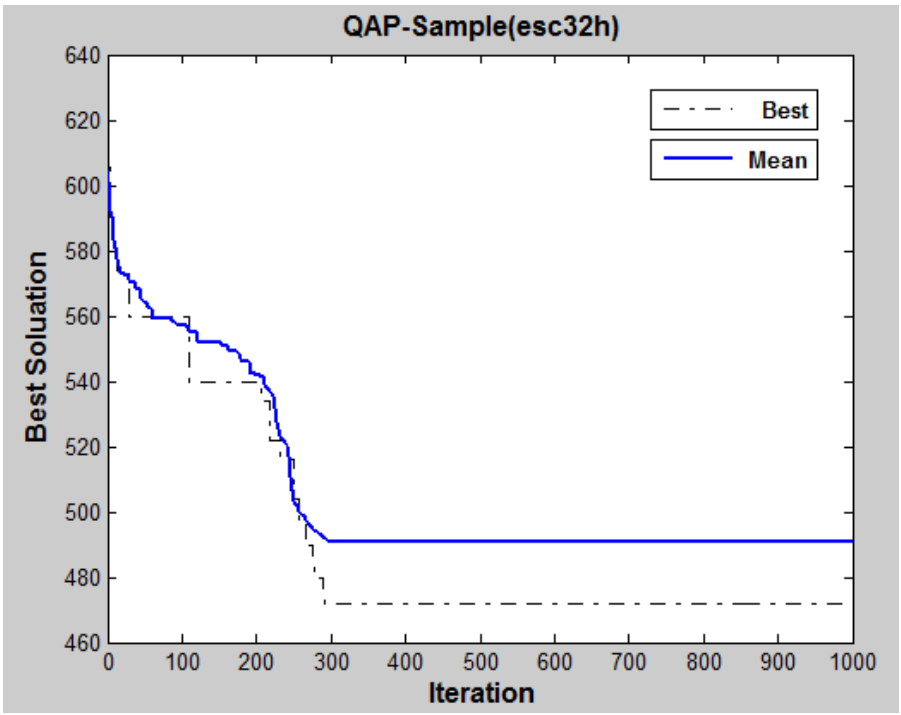
نمودار های زیر مربوط به سرعت همگرایی الگوریتم GSA در حل مسائل موجود در جدول ۳-۱ می باشند. که خطوط آبی مربوط به بهترین جواب و خطوط خط تیره مربوط به میانگین بهترین جواب می باشند.



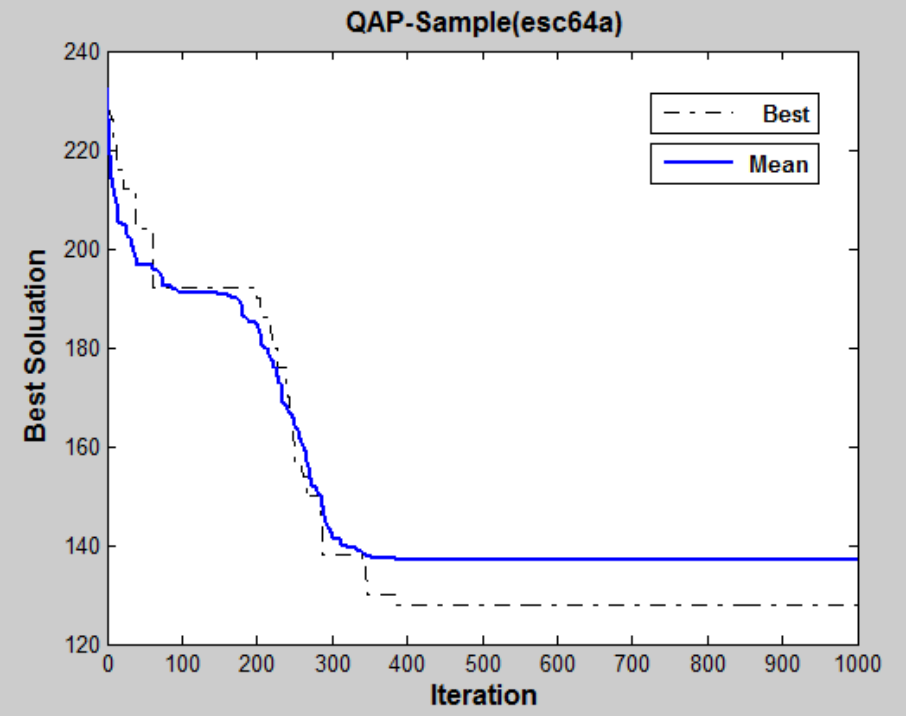
نمودار ۳-۱- نمودار سرعت همگرایی الگوریتم GSA در حل مساله esc32e



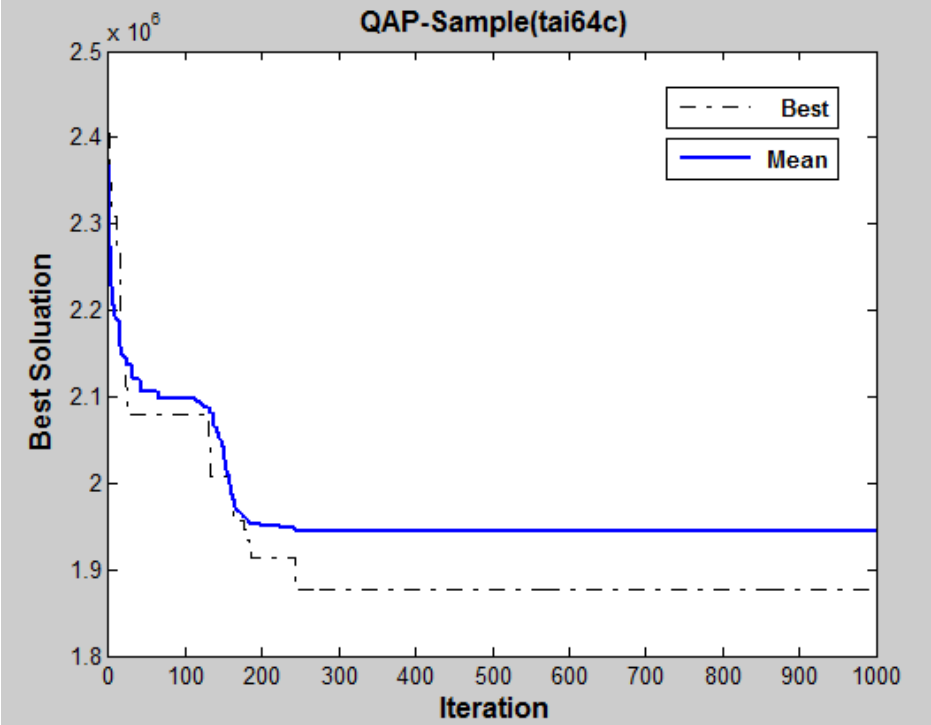
نمودار ۳-۲- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **esc32g**



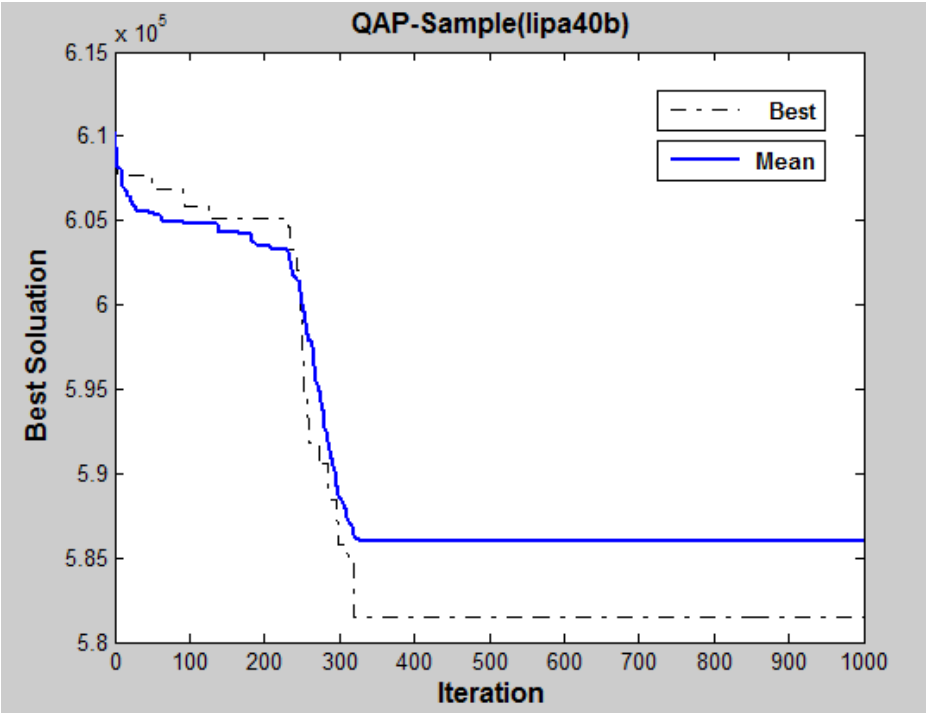
نمودار ۳-۳- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **esc32h**



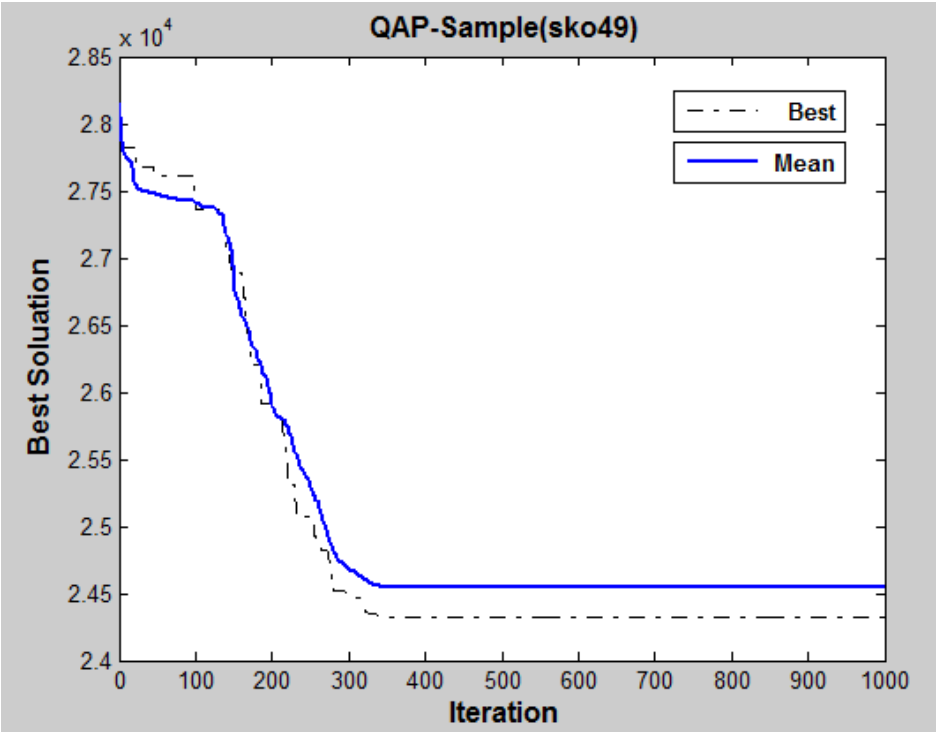
نمودار ۳-۴- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **esc64a**



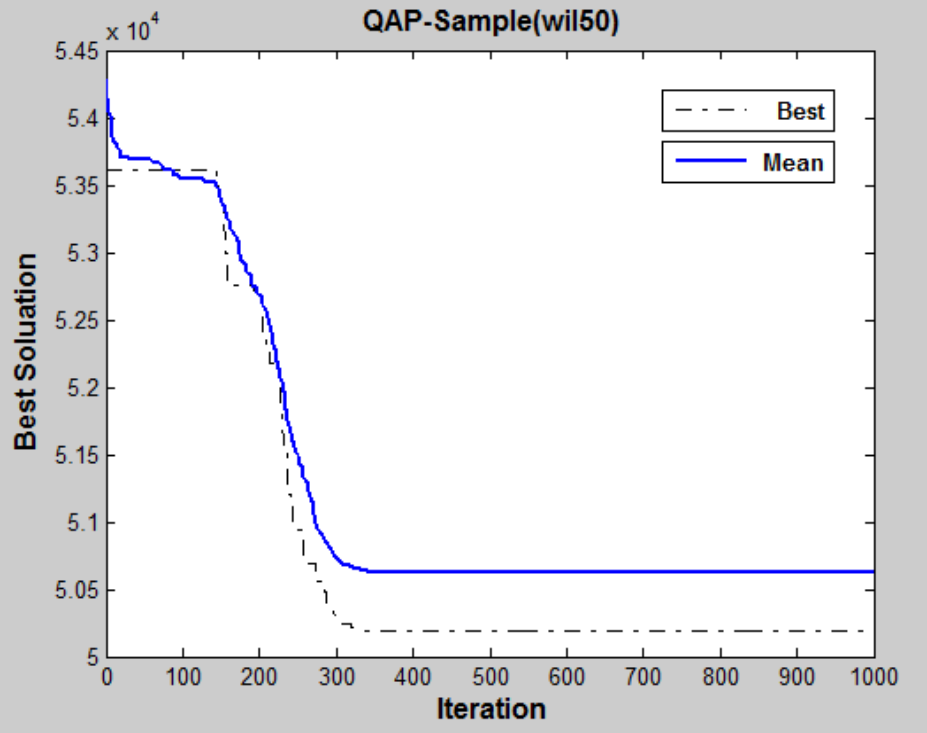
نمودار ۳-۵- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **tai64c**



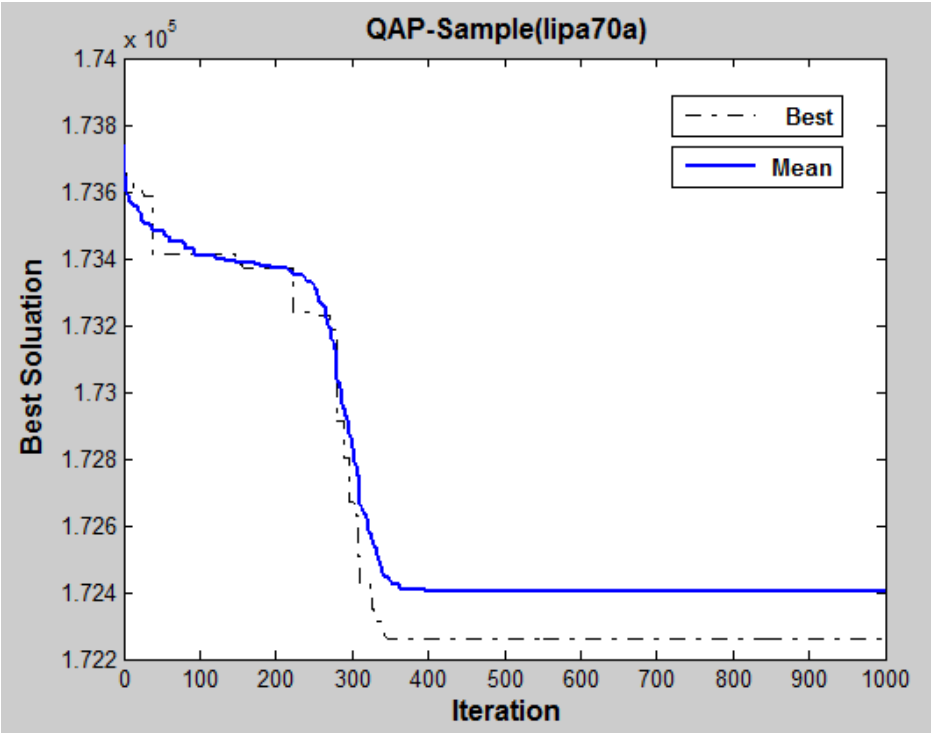
نمودار ۳-۶- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **lipa40b**



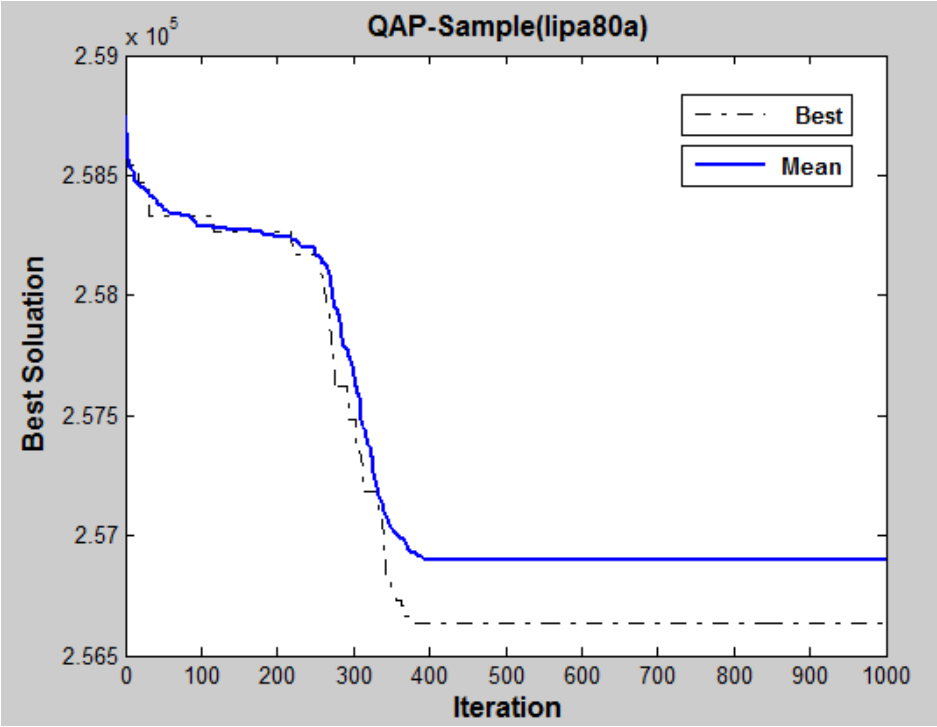
نمودار ۳-۷- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **sko49**



نمودار ۳-۸- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **wil50**



نمودار ۳-۹- نمودار سرعت همگرایی الگوریتم **GSA** در حل مساله **lipa70a**



نمودار ۳-۱۰- نمودار سرعت همگرایی الگوریتم GSA در حل مساله lipa80a

با توجه به نمودار های فوق و همچنین نتایج موجود در جدول می توان دریافت که علی رغم اینکه الگوریتم جستجوی گرانشی از سرعت همگرایی مناسبی برخوردار است اما در بعضی مسائل الگوریتم دچار همگرایی زودرس می شود و در عبور از نقاط بهینه محلی ناتوان عمل کرده و به عبارت دیگر در بهینه محلی گیر می کند که همین امر موجب ناتوانی در یافتن راه حل های بهتر و افزایش خطا می شود. از این رو با توجه به این که الگوریتم جستجوی گرانشی، توانایی جستجو محلی ضعیفتری نسبت به توانایی جستجو سراسری دارد به جهت رسیدن به راه حل بهتر، از ترکیب آن با الگوریتم ژنتیک استفاده می کنیم تا با استفاده از عملگر جهش و تقاطع جستجوی محلی آن را بهبود ببخشیم.

۳-٤ الگوریتم پیشنهادی

در GSA، عملکرد عامل ها به وسیله جرم آنها در نظر گرفته می شود. همه عوامل یکدیگر را توسط نیروی گرانش جذب می کنند، در حالی که این نیرو باعث یک حرکت سراسری از همه عوامل به سمت عامل های سنگین تر می شود. اجرام سنگین منجر به راه حل های خوب برای مساله می شود. به عبارت دیگر، هر جرم

نشان دهنده یک راه حل است، و الگوریتم به وسیله تنظیم مناسب جرم گرانشی و جرم لختی هدایت می شود. با گذشت زمان، اجرام توسط سنگین ترین جرم که آن نشان دهنده یک راه حل بهینه در فضای جستجو هست جذب می شوند. دستیابی به GSA به دو هدف متضاد که اکتشاف و بهره برداری هستند بستگی دارد. اکتشاف توانایی گسترش بررسی سراسری در فضای جستجو است، در حالی که بهره برداری، توانایی پیدا کردن یک نقطه بهینه در اطراف یک راه حل خوب است. در تکرار های اولیه، الگوریتم باید از اکتشاف برای جلوگیری از به دام افتادن در بهینه محلی استفاده کند. در حالی که مراحل جستجو ادامه می یابد، اکتشاف رفته رفته تاثیر کمتر و بهره برداری تاثیر بیشتری پیدا می کنند که باعث پیدا کردن راه حل بهتری خواهد شد.

با این حال، برخی از نقاط ضعف الگوریتم جستجوی گرانشی در مراحل جستجو برای بهترین راه حل وجود دارند.

ضعف اول کنترل تعادل بین اکتشاف و بهره برداری است که در آن اکتشاف بیشتر، در همگرایی زودرس تاثیر می گذارد درحالی که بهره برداری نرخ همگرایی را تحت تاثیر قرار می دهد. ضعف دوم این است که بهترین عامل حتی اگر در بهترین موقعیت باشد هنوز در حال بررسی فضای سراسری است.[٤٥]

اگر چه ثابت شده است GSA یک الگوریتم بهینه موثر است، ولی پس از همگرایی، GSA توانایی اکتشاف خود را از دست می دهد و غیر فعال می شود.

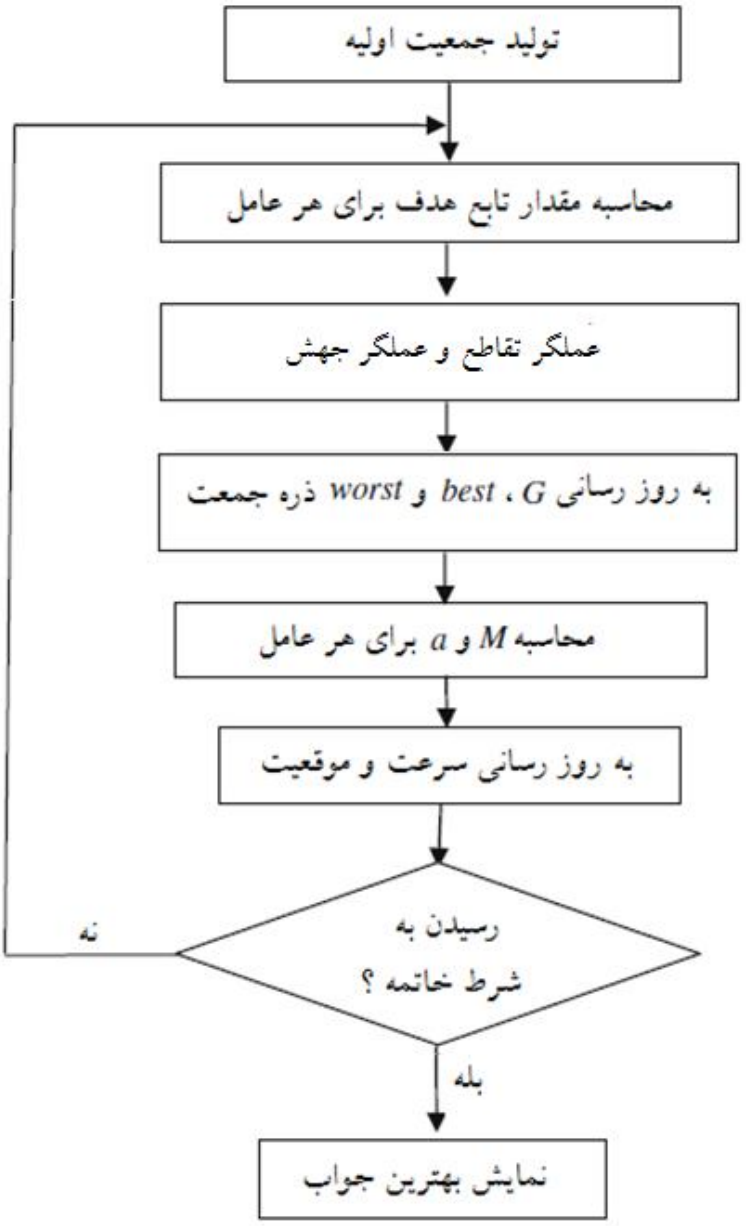
در مقابل، الگوریتم ژنتیک قادر به پیدا کردن راه حل های جدید با عملگرهای تقاطع و جهش هنگام مواجهه با همگرایی زودرس است، هر چند GA در پیدا کردن یک راه حل دقیق مشکلاتی دارد. از این رو، برای مقابله با همگرایی زودرس GSA، الگوریتم پیشنهادی GSA-GA، که ترکیبی از قوانین به روز رسانی سرعت و موقعیت GSA استاندارد با ایده های تقاطع و جهش در GA است، ارائه شده است. GA-GSA، GA را برای تولید پرش برای اجتناب از مشکل گیر افتادن در بهینه محلی در GSA مورد استفاده قرار داده است. به عبارت دیگر ادغام بهینه سازی سراسری GA و همگرایی سریع GSA توسط همکاری و یکپارچه سازی عملگر های تقاطع و جهش از GA و فرمول سرعت- جابجایی از GSA برای حل مسائل بهینه سازی بهتر و موثر تر است.

GA-GSA دارای سه مرحله اصلی است. اول از همه، تولید یک جمعیت P'از جمعیت اصلی P با توجه به عملیات های ادغام و جهش در GA و سپس تولید نسل بعدی جمعیت P با توجه به معادلات (۳-۱ تا ۳-۱۷) از GSA و سرانجام بهترین راه حل پس از ملاقات معیار بهینه سازی برگردانده می شود.

با توجه به تجزیه و تحلیل بالا، ترکیب با GA به توانایی جستجوی GSA شتاب می بخشد با ترکیب کردن بهینه سازی سراسری از GA با جستجوی محلی سریع GSA. سرعت همگرایی منطقه محلی در نتیجه افزایش می یابد و GSA دیگر تمایل به گیر کردن در بهینه محلی ندارد. هم زمان، دقت جستجو بهبود داده می شود. [٤٤]

از آنجا که نیروی گرانشی کمک می کند تا تمام اجرام به سرعت به همگرایی برسند،در نتیجه GSA به طور بالقوه مستعد انتخاب بهینه محلی است. با توجه به تجزیه و تحلیل بالا، رفتار GSA جستجو از یک راه حل بهینه را ارتقا می دهد، در حالی که GA در رسیدن به یک ناحیه سراسری بهتر است.

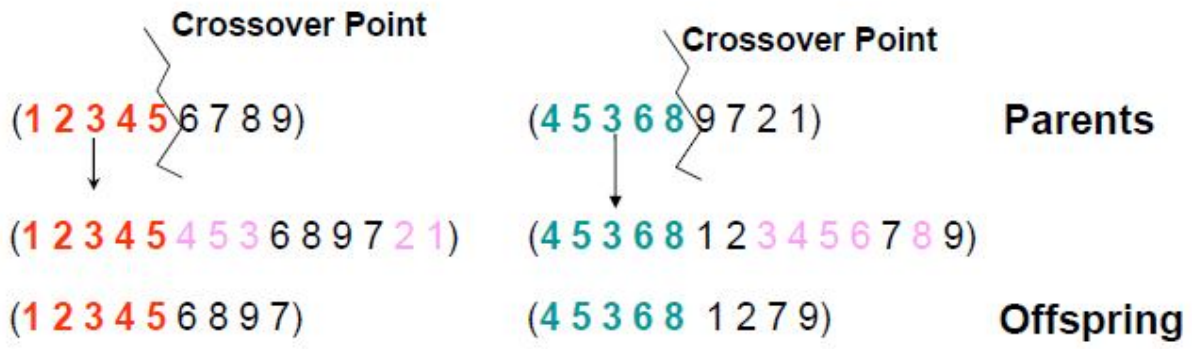
بنابراین، الگوریتم پیشنهادی GA را برای تولید پرش برای اجتناب از مشکل گیر افتادن در بهینه محلی در GSA مورد استفاده قرار داده است.



شکل ۳-۵- قاعده کلی الگوریتم GSA-GA

عملگر تقاطع

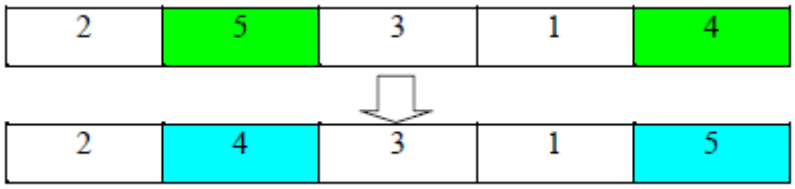
برای عملگر تقاطع از تقاطع تک نقطه ای استفاده شده است. در این روش دو والد براساس چرخ رولت انتخاب می شوند و به صورت تصادفی از یک نقطه شکسته می شوند. بخش اول از کروموزوم والد به کروموزوم فرزند منتقل می شوند. برای تکمیل بخش دوم کروموزوم هر فرزند، کروموزوم والد دیگر را از ابتدا اسکن نموده و هر عددی که در کروموزوم فرزند موجود نیست، به آن منتقل می شود.



شکل ۳-۶- مثال عملگر تقاطع تک نقطه ای

عملگر جهش

عملگر جهش نیز در هر تکرار بر روی عامل ها اعمال می شود. در عملگر جهش دو نقطه به صورت تصادفی انتخاب می شوند و مقدار آن ها با هم جا به جا می شود.



شکل ۳-۷- مثال عملگر جهش

شبه کد الگوریتم پیشنهادی در شکل زیر آمده است.

1. Search space identification, t=0;
2. Random initialization, $Xi(t)$;
For $i=1, ..., N$
3. Fitness evaluation of objects;
4. Perform GA crossover and mutation operations to generate new population
5. Update the parameters of *G, best, worst and M;*
For $i=1, ..., N$
6. Calculation of the force on each object;
7. Calculation of the acceleration and the velocity of each object;
8. Update the position of the agents by (7-3) to yield $Xi(t+1)$;
t=t+1;
9. Repeat steps 3 to 7 until the stop criteria is reached;
10. end

شکل ۳-۸- شبه کد الگوریتم پیشنهادی GSA-GA

در ادامه الگوریتم پیشنهادی GSA-GA را برای مسائل مختلف QAP به کار گرفته و نتایج آن را بررسی می کنیم که این نتایج در فصل بعدی آمده است.

فصل چهارم:تجزیه و تحلیل داده ها

با توجه به اینکه این پژوهش به مدل عمومی مساله تخصیص درجه دوم پرداخته است و مقصود از آن این بوده است تا برای اولین بار رفتار الگوریتم جستجوی گرانشی برای حل مساله تخصیص درجه مورد بررسی قرار گیرد، تا نسبت به کارایی آن اطمینان حاصل شده و آن گاه بتوان این مساله را در پژوهش های آینده در موارد خاص دنیای واقعی حل نمود، لذا مجبور بودیم تا مسائلی را انتخاب کنیم که شناخته شده بوده و برای آزمایش سایر الگوریتم ها از آن استفاده شده باشد. بنابراین به معتبرترین مرجع مساله تخصیص درجه دوم که توسط پیتر هان، برکارد، چلا و رندل و کاریش استادان ریاضی که به طور تخصصی در زمینه مساله تخصیص درجه دوم فعالیت می کنند، مراجعه کردیم. این مرجع سایت معتبر QAPLIB می باشد که در آن مسائل مختلفی از مساله تخصیص درجه دوم در ابعاد مختلف توسط افرادی چون برکارد، الشافی، اشتاین برگ، و ... طرح و با استفاده از روش های دقیق، ابتکاری و یا فرا ابتکاری آنها را حل نموده اند.

به منظور دستیابی به نتایج مورد نظر مساله را با آزمایش های مختلف مورد بررسی و تحلیل قرار دادیم.

مشخصات رایانه:

Processor: Intel(R) Core(TM)2 Duo CPU T9550 @ 2.66GHz
RAM: 2.99GB
Windows: 7 Professional 32bit
Coding Environment: Matlab R2013a (8.1.0.604)

٤-١ آزمایش اول:بررسی دقت الگوریتم با توجه به ابعاد مساله

در این آزمایش به بررسی دقت الگوریتم پیشنهادی با توجه به ابعاد مساله می پردازیم. به این صورت که مسائلی با ابعاد ۳۰ و ٦۰ و ۷۰ و ۸۰ و ۹۰ از گروه مسائل lipa*a و همینطور مسائل با ابعاد ٤۲ تا ۸۱ بُعدی از گروه مسائل *sko را مورد آزمایش قرار می دهیم.

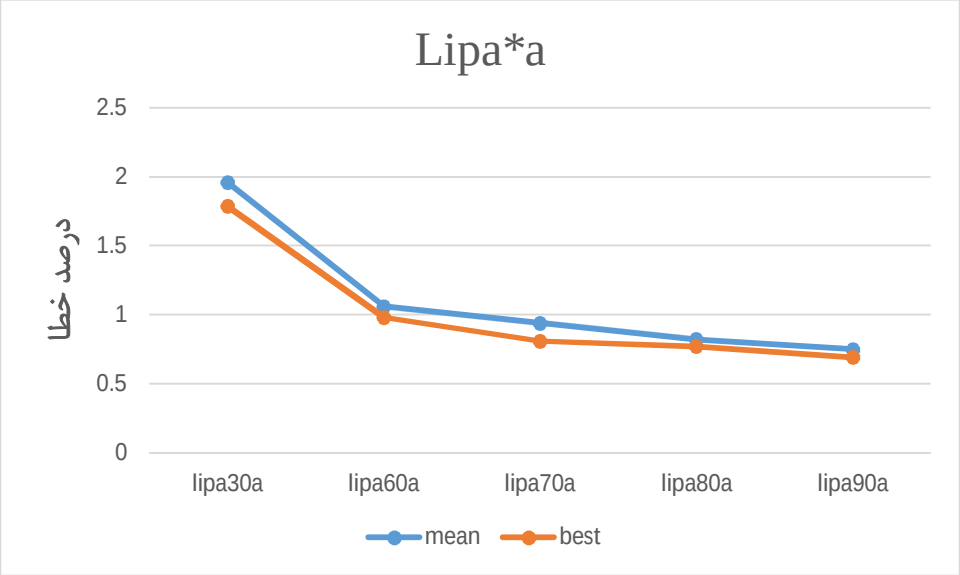
در این آزمایش تعداد عامل ها برابر ۵۰ فرض شده و تعداد تکرار برابر با ۱۰۰۰ می باشد.

شایان ذکر است که برای هر مساله، مساله مورد نظر ۱۰ بار اجرا شده و بهترین جواب در بین آنها در نظر گرفته شده است. و همینطور ذکر این نکته ضروری است که منظور از دقت الگوریتم اختلاف بین بهترین جواب به دست آمده تا کنون(BKS) [۱٦۲] و بهترین جواب به دست آمده از الگوریتم پیشنهادی است.

[162] Best known solution

نام مساله	ابعاد مساله	بهترین جواب به دست آمده تاکنون	بهترین جواب GSA-GA	میانگین جواب GSA-GA	درصد خطای بهترین	درصد خطای میانگین
Lipa30a	30	13178	13414	13437	1.79	1.96
Lipa60a	60	107218	108277	108360	0.98	1.06
Lipa70a	70	169755	171133	171360	0.81	0.94
Lipa80a	80	253195	255166	255290	0.77	0.82
Lipa90a	90	360630	363141	363350	0.69	0.75

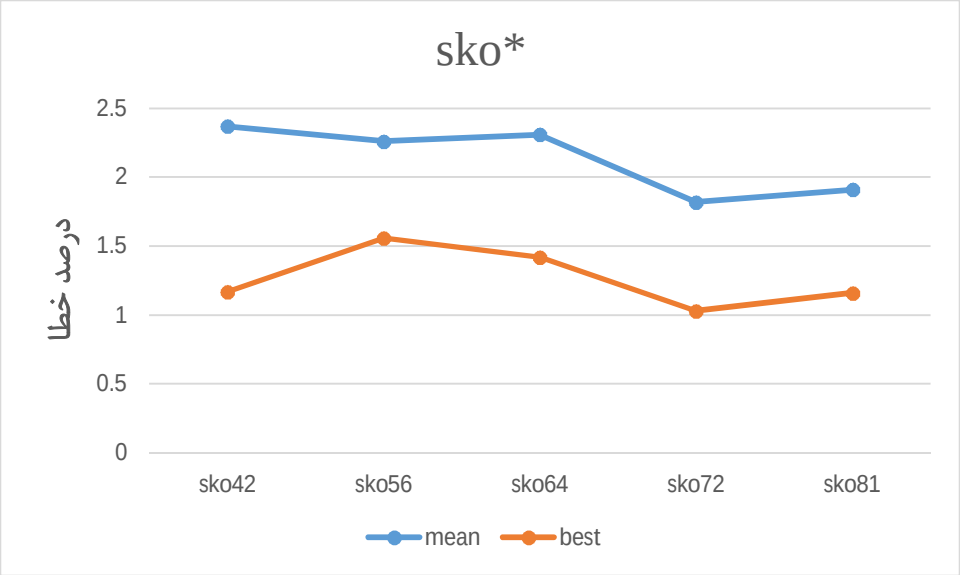
جدول ۴-۱- نتایج حاصل از اجرای الگوریتم **GSA-GA** بر روی برخی مسائل گروه **lipa*a** با ابعاد مختلف



نمودار ۴-۱- نمودار بررسی دقت الگوریتم در حل مسائل **lipa*a** با توجه به ابعاد مساله

نام مساله	ابعاد مساله	بهترین جواب به دست آمده تاکنون	بهترین جواب GSA-GA	میانگین جواب GSA-GA	درصد خطای بهترین	درصد خطای میانگین
Sko42	42	15812	15998	16188	1.17	2.37
Sko56	56	34458	34998	35239	1.56	2.26
Sk064	64	48498	49190	49623	1.42	2.31
Sko72	72	66256	66944	67465	1.03	1.82
Sko81	81	90998	92062	92737	1.16	1.91

جدول ۴-۲- نتایج حاصل از اجرای الگوریتم **GSA-GA** بر روی برخی مسائل گروه **sko*** با ابعاد مختلف



نمودار ۴-۲- نمودار بررسی دقت الگوریتم در حل مسائل *sko با توجه به ابعاد مساله

با توجه به نمودار ٤-١ مشخص است که در مسائل گروه lipa*a با افزایش ابعاد مساله، فاصله بین جواب بهینه به دست آمد و جواب بهینه اعلام شده در مرجع QAPLIB که به اصطلاح به آن گپ[۱۶۳] می گویند، کاهش یافته در واقع الگوریتم ارائه شده با افزایش ابعاد مساله با دقت بالاتری کار می کند، اما با مشاهده نمودار ٤-٢ مشخص می شود که این یک قاعده کلی نبوده است. در این نمودار مشخص شده است که با افزایش ابعاد مساله تغییر درصد خطا از الگوی خاصی پیروی نمی کند. بنابراین در حالت کلی می توان گفت که دقت الگوریتم ارائه شده برای ابعاد مختلف یک نوع مساله به ساختار مساله ای که برای آن به کار رفته است بستگی دارد. به عبارت دیگر الگوریتم ارائه شده بیشتر از آن که به ابعاد مساله بستگی داشته باشد به ساختار مساله مورد بررسی بستگی دارد.

٤-٢ آزمایش دوم:بررسی نقش پارامترهای ورودی مساله در دقت الگوریتم

٤-٢-١ آزمایش دوم – قسمت اول:بررسی نقش پارمتر N (تعداد اجرام یا عامل ها)

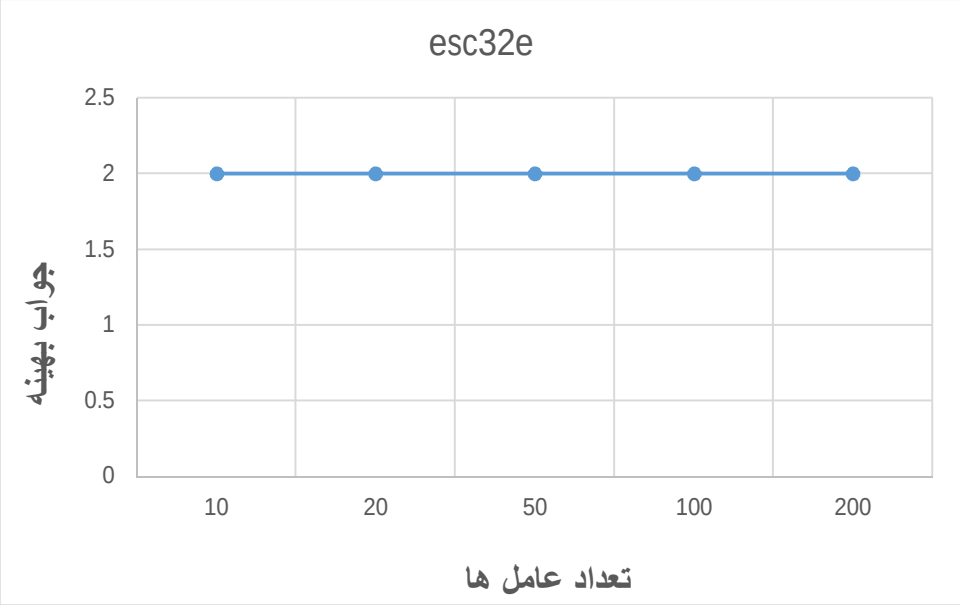
برای بررسی نقش پارامتر N ما به ازای مقادیر ۱۰، ۲۰، ۵۰، ۱۰۰، ۲۰۰الگوریتم ارائه شده را بر روی تعدادی از مسائل مرجع به کار بردیم که نتایج آن را در جدول زیر مشاهده می کنید.

تعداد تکرار برای هر مساله ۱۰۰۰ تکرار فرض شده و برای هر مساله بهترین جواب بعد از اجرای ۱۰ بار آن لحاظ گردیده است.

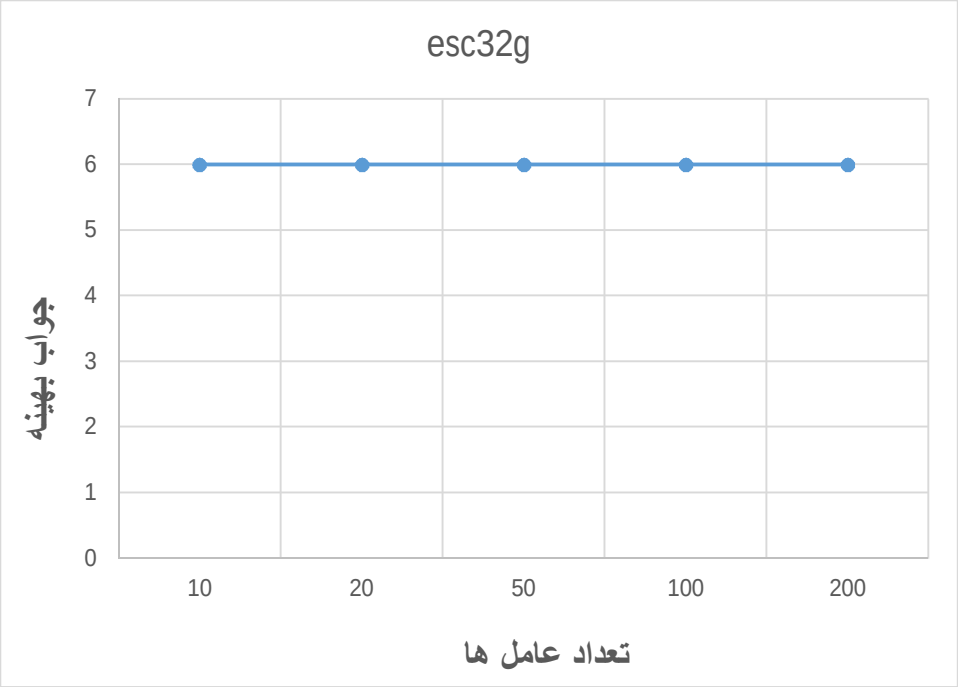
[163] Gap

نام مساله	ابعاد مساله	بهترین جواب به دست آمده تاکنون	N=10	N=20	N=50	N=100	N=200
Esc32e	32	2	2	2	2	2	2
Esc32g	32	6	6	6	6	6	6
Esc32h	32	438	448	442	440	442	440
Esc64a	64	116	120	118	116	116	116
Tai64c	64	1855928	1866152	1863678	1855928	1856393	1855928
Lipa40b	40	476581	565765	561499	476581	476581	476581
Sko49	49	23386	24206	24024	23636	23620	23656
Wil50	50	48816	499722	49780	49106	49098	49050
Lipa70a	70	169755	171400	171412	171312	171320	171268
Lipa80a	80	253195	255824	255215	255068	255060	255046

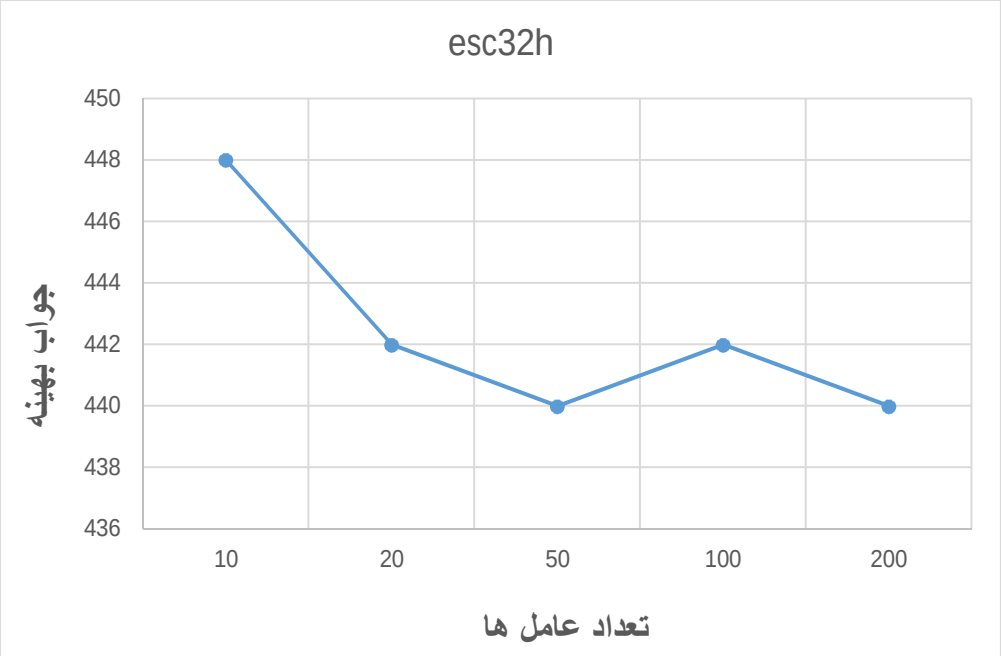
جدول ۴-۳- نتایج حاصل از اجرای الگوریتم **GSA-GA** بر روی مسائل منتخب جهت بررسی تاثیر تعداد عامل ها



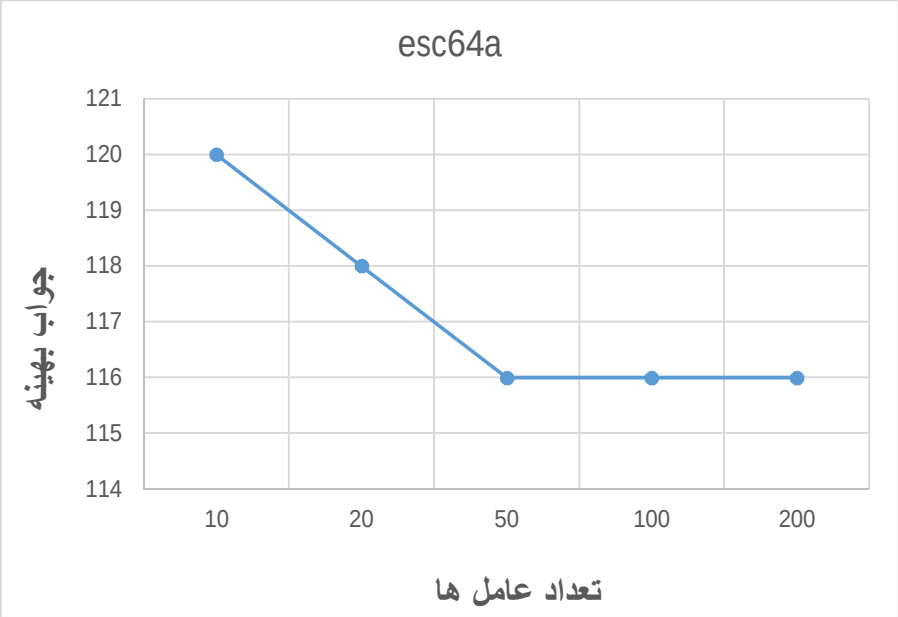
نمودار ۴-۳- نمودار تاثیر تعداد عامل ها در حل مساله **esc32e**



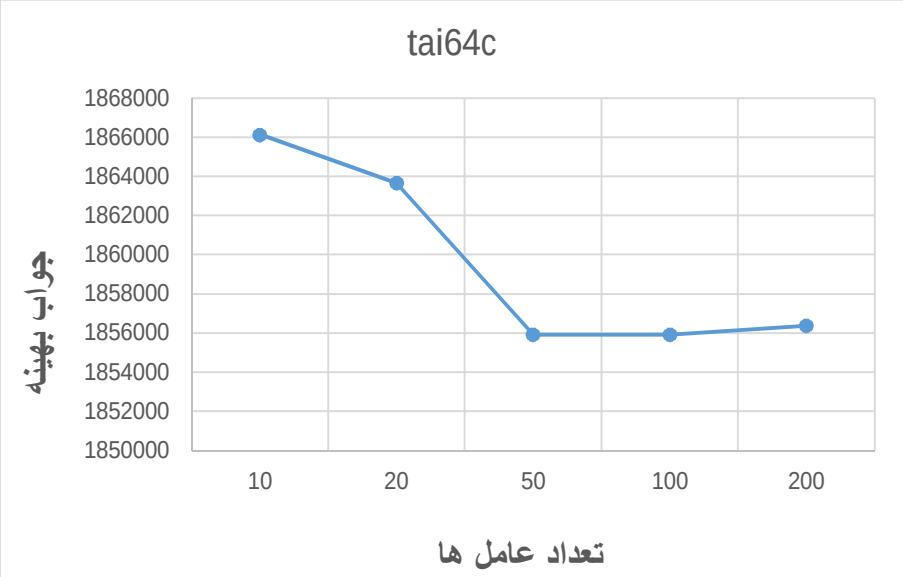
نمودار ۴-۴- نمودار تاثیر تعداد عامل ها در حل مساله **esc32g**



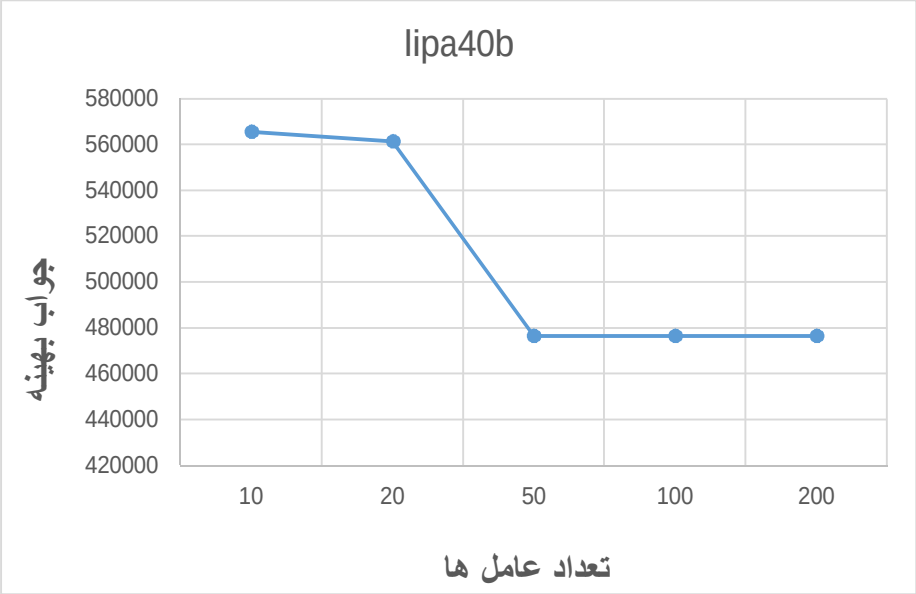
نمودار ۴-۵- نمودار تاثیر تعداد عامل ها در حل مساله **esc32h**



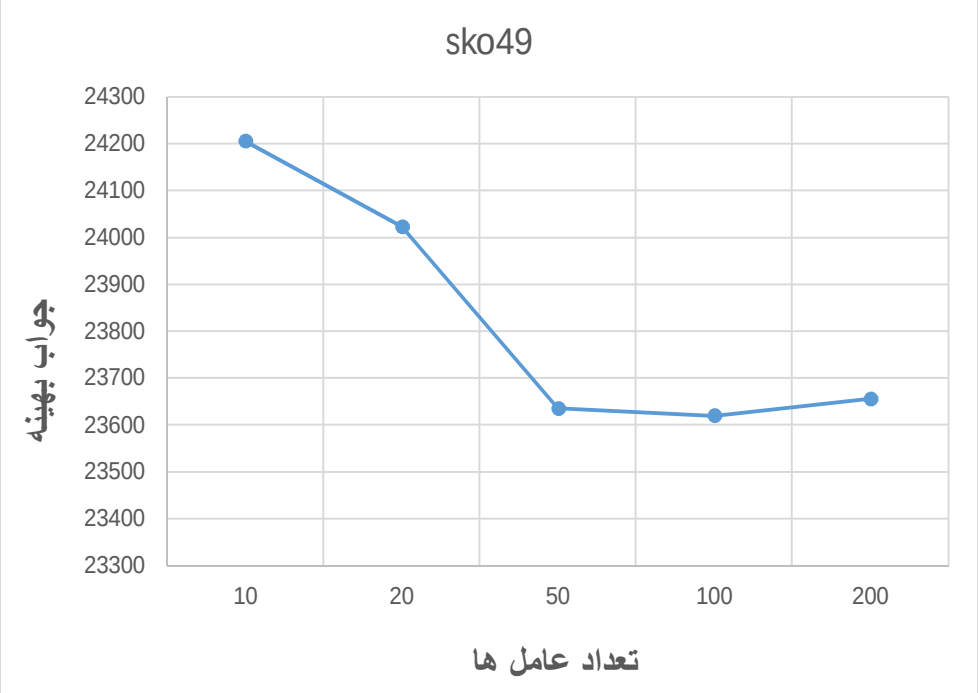
نمودار ۴-۶- نمودار تاثیر تعداد عامل ها در حل مساله **esc64a**



نمودار ۴-۷- نمودار تاثیر تعداد عامل ها در حل مساله **tai64c**

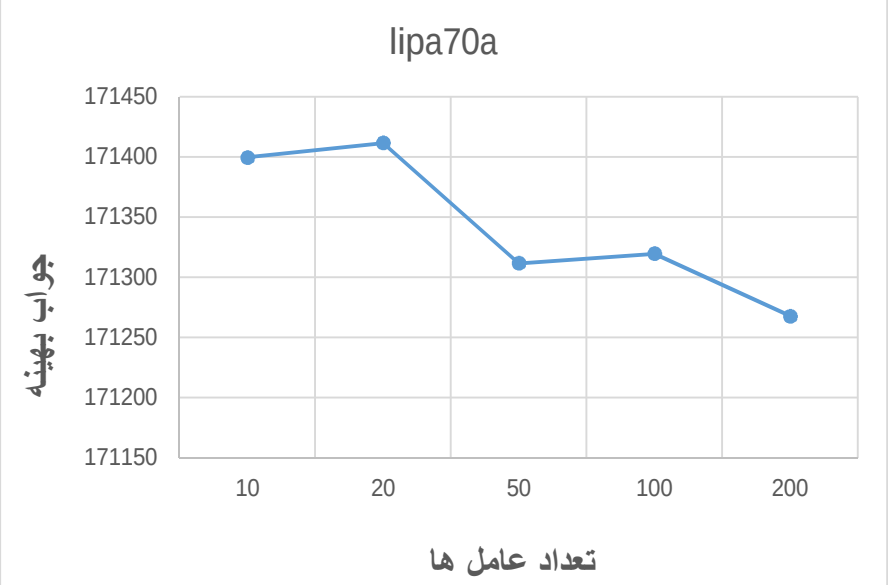


نمودار ۴-۸- نمودار تاثیر تعداد عامل ها در حل مساله **lipa40b**

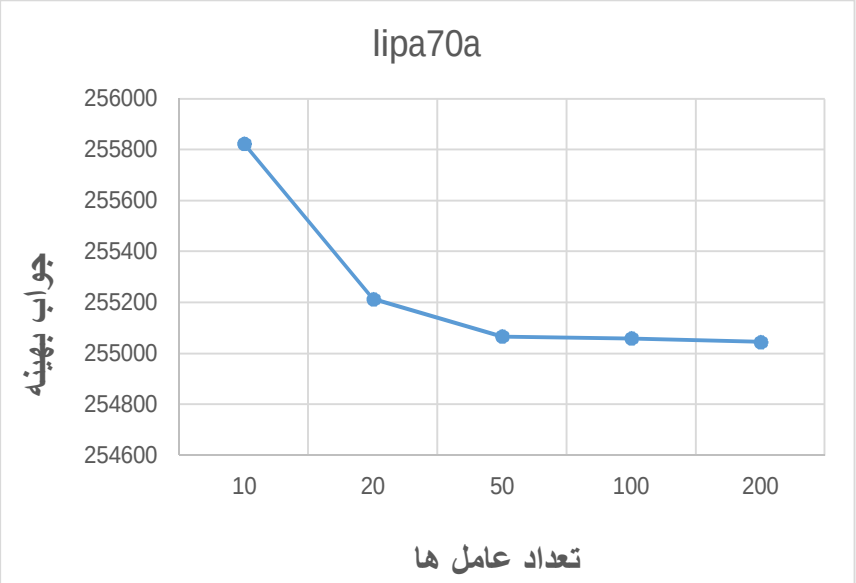


نمودار ۴-۹- نمودار تاثیر تعداد عامل ها در حل مساله **sko49**

نمودار ۴-۱۰- نمودار تاثیر تعداد عامل ها در حل مساله **wil50**



نمودار ۴-۱۱- نمودار تاثیر تعداد عامل ها در حل مساله **lipa70a**



نمودار ۴-۱۲- نمودار تاثیر تعداد عامل ها در حل مساله lipa80a

همانطور که از اشکال ٤-۳ تا ٤-۱۲ مشخص است با افزایش تعداد عامل ها، عملکرد الگوریتم بهتر شده است و درصد خطای جواب به دست آمده نسبت به بهترین جواب ها کاهش می یابد. اما با بررسی دقیق تر می توان دریافت که برای N های کوچکتر از 50 الگوریتم ارائه شده تقریبا غیر قابل پیش بینی بوده و نتایج آن وابستگی زیادی به جستجوی تصادفی دارد و همچنین می توان مشاهده کرد که در اکثر مسائل شیب تغییرات برای N=50 زیاد بوده و پس از آن با شیب ملایمی بهبود صورت می گیرد. در واقع برای N های بزرگتر از 50 اگر چه بهبود صورت می گیرد اما این بهبود با توجه به افزایش مقدار عامل ها چندان چشمگیر نیست.

از این رو با توجه به اینکه در اکثر مسائل برای N=50 الگوریتم به حالت پایدار رسیده و پس از آن بهبود در جواب ها جزیی می باشد و همچنین با توجه به اینکه تعداد عامل ها تاثیر مستقیمی در سرعت و زمان اجرای الگوریتم ارائه شده دارند بنابراین ما همین مقدار N=50 را به عنوان مقدار بهینه برای آن در نظر می گیریم که در زمان قابل قبولی بهینگی مورد نظر را تا حدودی تضمین می کند. از این رو در باقی آزمایشات از همین مقدار برای بررسی کیفیت الگوریتم ارائه شده در حل مسائل منتخب در مقایسه با سایر الگوریتم ها خواهیم پرداخت.

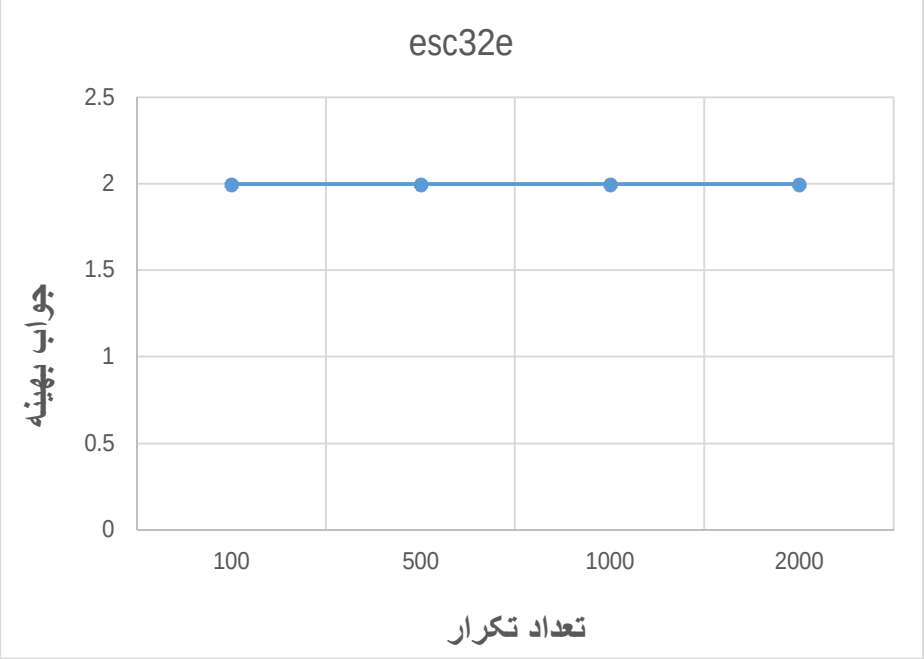
٤-۲-۲ آزمایش دوم –قسمت دوم:بررسی نقش تعداد تکرار ها در دقت الگوریتم

برای بررسی نقش پارامتر تکرار(iteration) ما به ازای مقادیر ۱۰۰، ۵۰۰، ۱۰۰۰، ۲۰۰۰ الگوریتم ارائه شده را بر روی همان مسائل مرجع به کار بردیم که نتایج آن را در جدول زیر مشاهده می کنید.

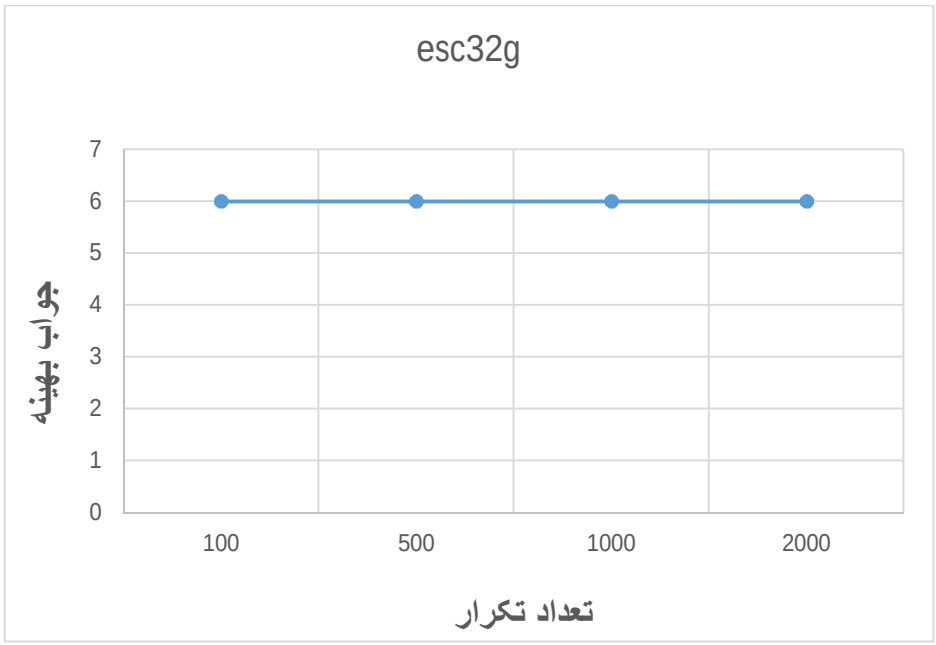
تعداد عامل ها در این قسمت با توجه به آزمایش قبلی برابر با N=50 در نظر گرفته شده است.

نام مساله	ابعاد مساله	بهترین جواب به دست آمده تاکنون	۱۰۰ تکرار	۵۰۰ تکرار	۱۰۰۰ تکرار	۲۰۰۰ تکرار
Esc32e	32	2	2	2	2	2
Esc32g	32	6	6	6	6	6
Esc32h	32	438	464	442	440	438
Esc64a	64	116	168	156	116	116
Tai64c	64	1855928	1867144	1864238	1855928	1855928
Lipa40b	40	476581	580412	565382	476581	478652
Sko49	49	23386	25150	23870	23636	23532
Wil50	50	48816	50926	49568	49106	49047
Lipa70a	70	169755	172367	171484	171312	171250
Lipa80a	80	253195	255824	255215	255068	255060

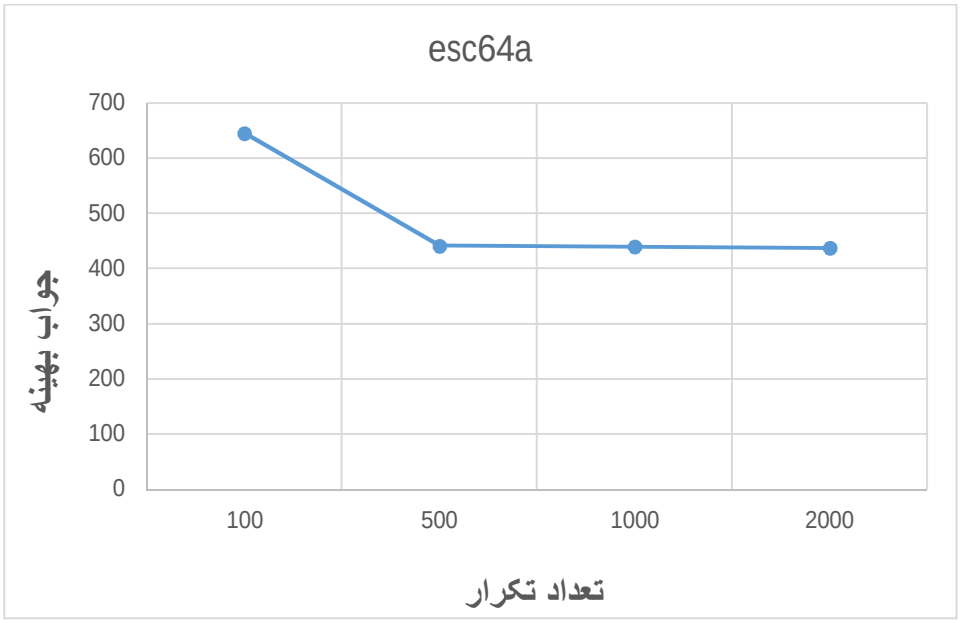
جدول ۴-۴- نتایج حاصل از اجرای الگوریتم GSA-GA بر روی مسائل منتخب جهت بررسی تاثیر تعداد تکرار الگوریتم



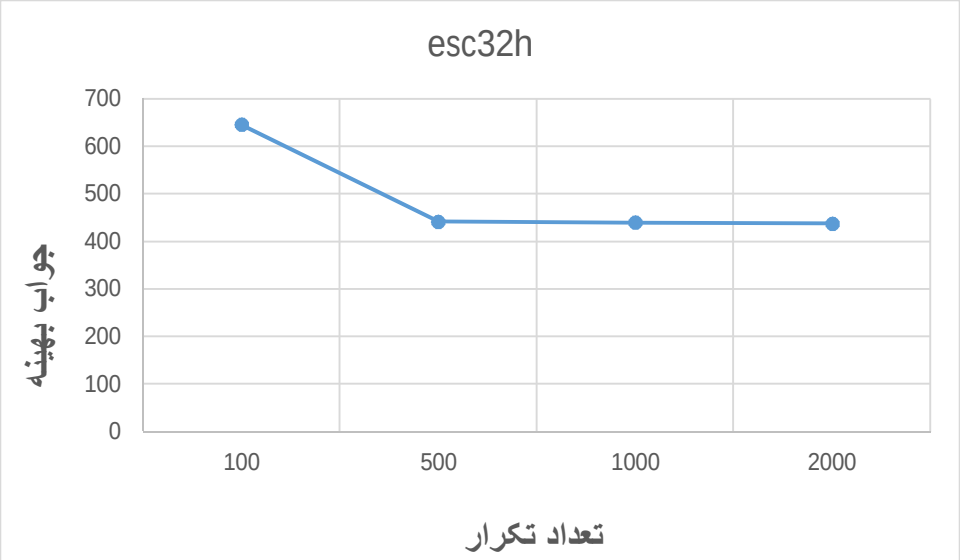
نمودار ۴-۱۳- نمودار تاثیر تعداد تکرار در حل مساله esc32e



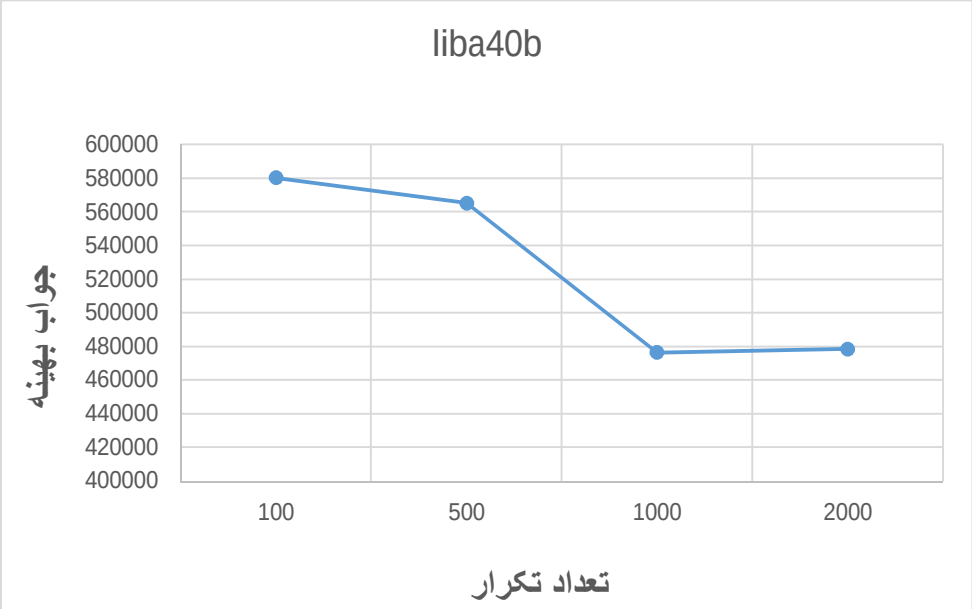
نمودار ۴-۱۴- نمودار تاثیر تعداد تکرار در حل مساله **esc32g**



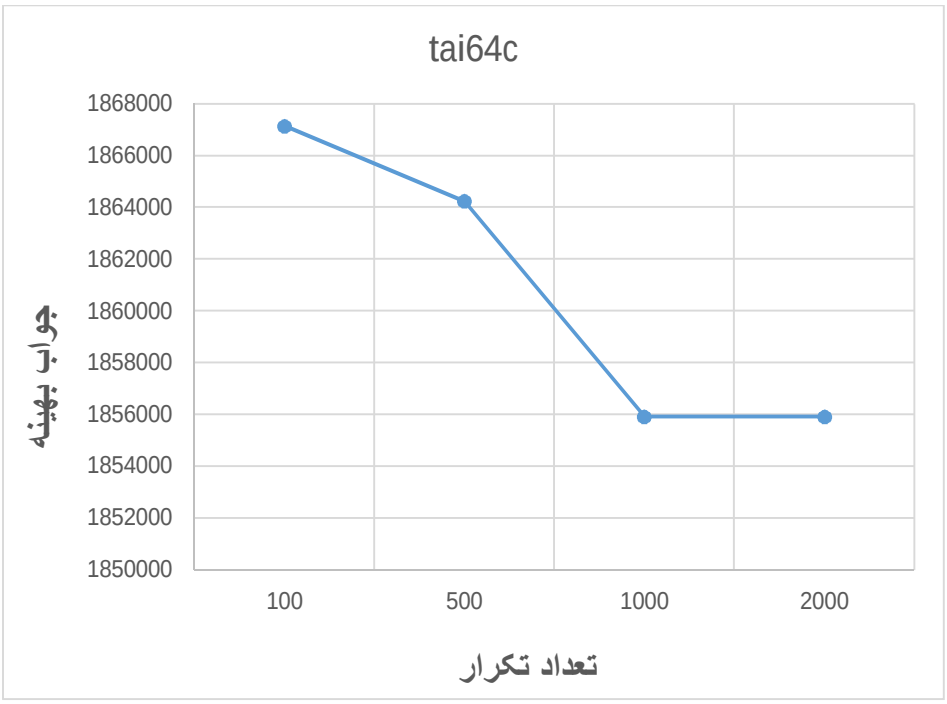
نمودار ۴-۱۵- نمودار تاثیر تعداد تکرار در حل مساله **esc64a**



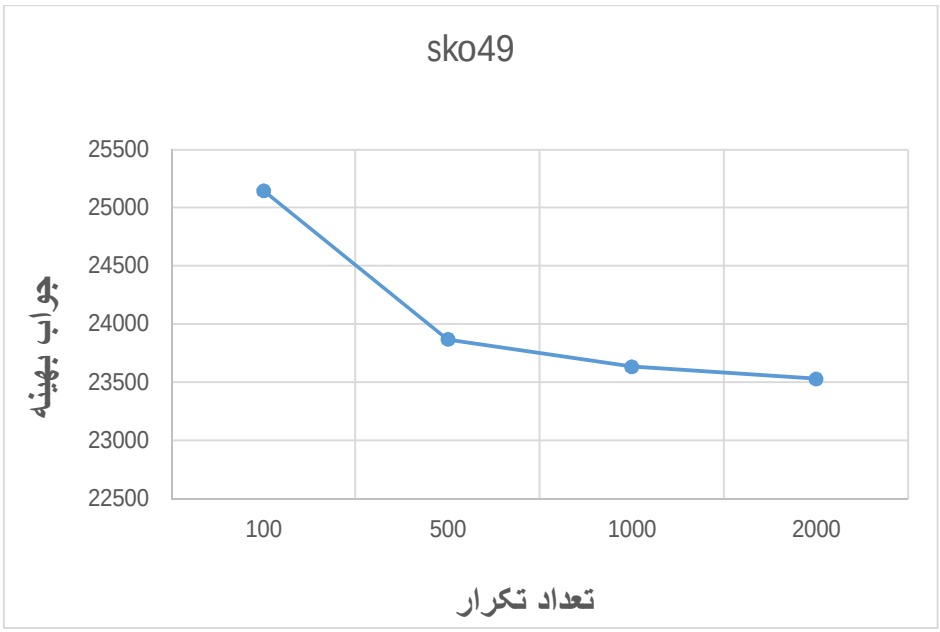
نمودار ۴-۱۶- نمودار تاثیر تعداد تکرار در حل مساله **esc32h**



نمودار ۴-۱۷- نمودار تاثیر تعداد تکرار در حل مساله **lipa40b**

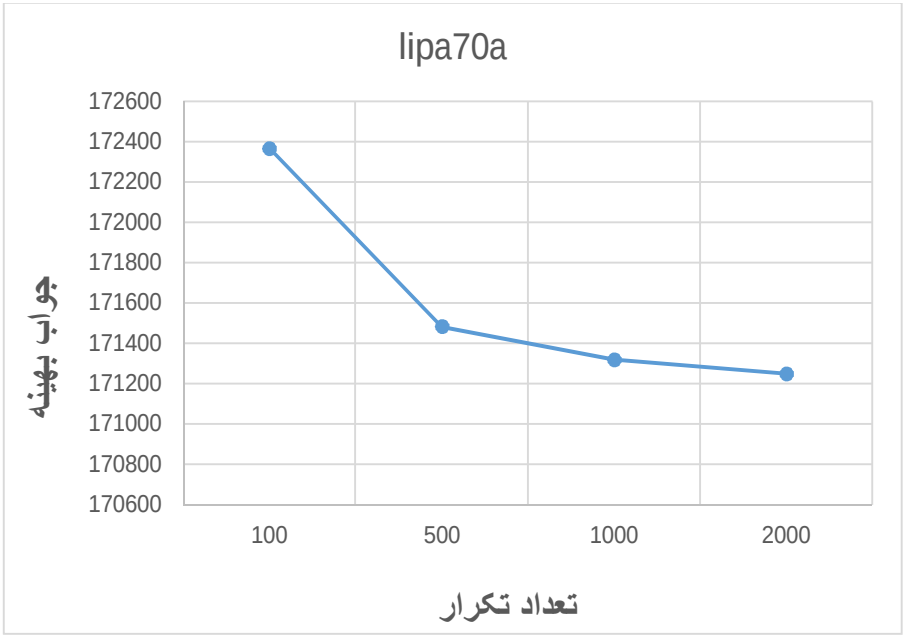


نمودار ۴-۱۸- نمودار تاثیر تعداد تکرار در حل مساله **tai64c**

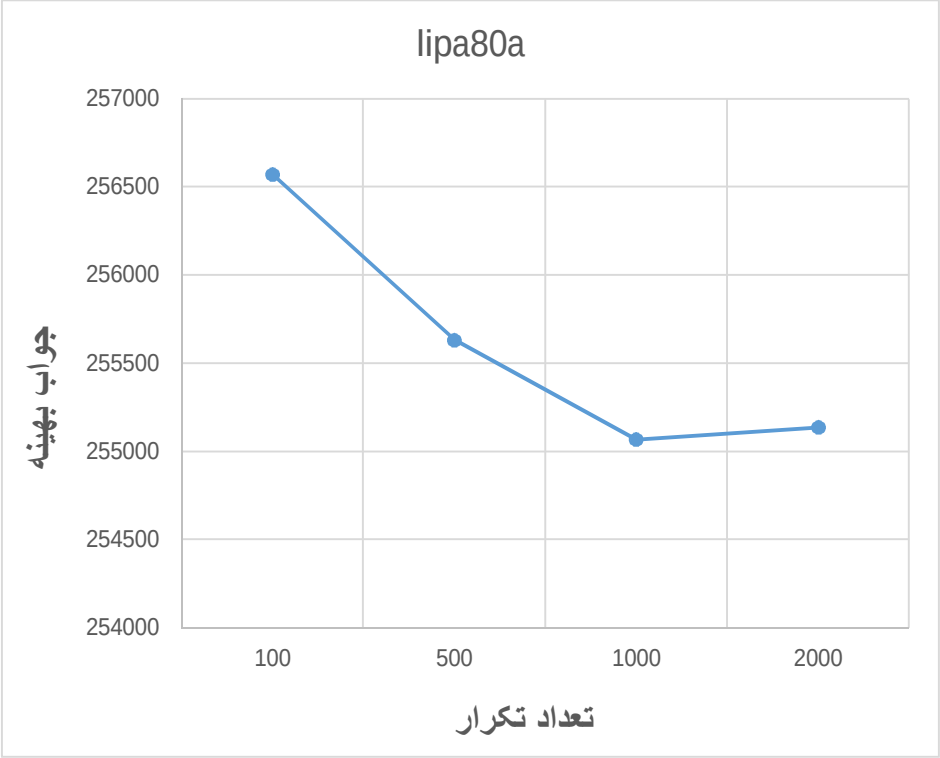


نمودار ۴-۱۹- نمودار تاثیر تعداد تکرار در حل مساله **sko49**

نمودار ۴-۲۰- نمودار تاثیر تعداد تکرار در حل مساله **wil50**



نمودار ۴-۲۱- نمودار تاثیر تعداد تکرار در حل مساله **lipa70a**



نمودار ۴-۲۲- نمودار تاثیر تعداد تکرار در حل مساله lipa80a

با توجه به نمودار های ٤-۱۳ تا ٤-۲۲ مشخص می شود که با افزایش تعداد تکرار های الگوریتم همان گونه که انتظار می رفت مقدار جواب بهینه کاهش می یابد، این کاهش در ابتدا با شیب بیشتری صورت می گیرد و در ادامه از شدت آن کاسته می شود.

همان گونه که مشخص است میزان فاصله حاصل از حل مساله در تکرار های ۱۰۰۰ و ۲۰۰۰ اختلاف محسوسی نداشته و نتیجه تقریبا مشابهی دارند. البته ذکر این نکته ضروری است که به طور منطقی تکرارهای بسیار بالا ممکن است منجر به بهبود جواب شوند، اما موضوع اصلی در این حالت زمان بسیار زیادی است که صرف خواهد شد و عملا توجیه اقتصادی از لحاظ زمانی برای حل مساله با تکرار های بسیار زیاد را از بین می برد.

در اکثر مسائل بررسی شده الگوریتم ارائه شده در ۱۰۰۰ تکرار تقریبا به حالت پایدار رسیده و اختلاف بین ۱۰۰۰ تکرار و ۲۰۰۰ تکرار (اگرچه بهبود صورت می گیرد) چندان قابل توجه نیست و از لحاظ زمانی توجیه اقتصادی ندارد، بنابراین ما مقدار ۱۰۰۰ تکرار را به عنوان مقدار بهینه برای حل مسائل مورد نظر انتخاب می کنیم.

٤-٣ ارزیابی عملکرد الگوریتم ارائه شده در مقایسه با پرکاربردترین الگوریتم های به کار رفته برای QAP

در این قسمت نتایج حاصل از الگوریتم ارائه شده را بر روی مسائل برگزیده از QAPLIB با نتایج الگوریتم های برتر SS و GA [۳۵] و PSO [۳٦] و DE [٤٦]و H14 و MBO [۲۷] که جز کارامدترین الگوریتم ها برای حل مسائل QAP می باشند مقایسه می کنیم تا عملکرد الگوریتم مورد نظر را در حل مسائل ببنیم.

n	K	M	X
51	3	10	1

جدول ۴-۵- پارامتر های الگوریتم MBO

N	Iteration
50	1000

جدول ۴-۶- پارامتر های الگوریتم GSA-GA

برای هر دو الگوریتم بهترین مقدار به دست آمده در ۱۰ بار اجرای الگوریتم بر روی مسائل ثبت گردیده است.

روابط مورد نیاز:

BKS = بهترین جواب به دست آمده تا کنون

درصد خطای جواب i ام:

$$gap_i = \frac{Solution_i - BKS}{BKS} \times 100 \qquad (1-4)$$

درصد خطای میانگین:

$$gap_{avg} = \frac{1}{n}\sum_{i=1}^{n} gap_i \qquad (2-4)$$

درصد انحراف معیار خطاها:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(gap_i - gap_{avg})^2} \qquad (3-4)$$

با توجه به اینکه برای هر مساله ۱۰ بار الگوریتم مورد نظر اجرا شده است بنابراین مقدار n=10 در نظر گرفته شده و بهترین جواب و میانگین جواب های این تکرار ها به همراه درصد خطای هر کدام و انحراف معیار خطاها در جدول زیر آمده است.

نام مساله	ابعاد مساله	BKS	بهترین جواب GSA	میانگین جواب GSA	درصد خطای بهترین	درصد خطای میانگین	σ
Esc32e	32	2	2	2	0	0	0
Esc32g	32	6	6	6	0	0	0
Esc32h	32	438	440	451	0.45	2.96	2.21
Esc64a	64	116	116	117.8	0	1.55	1.48
Tai64c	64	1855928	1855928	1860600	0	0.25	0.20
Lipa40b	40	476581	476581	546790	0	14.73	6.80
Sko49	49	23386	23638	23825	1.07	1.87	0.53
Wil50	50	48816	48984	49397	0.34	1.19	0.44
Lipa70a	70	169755	171133	171360	0.81	0.94	0.05
Lipa80a	80	253195	255170	255400	0.78	0.87	0.04

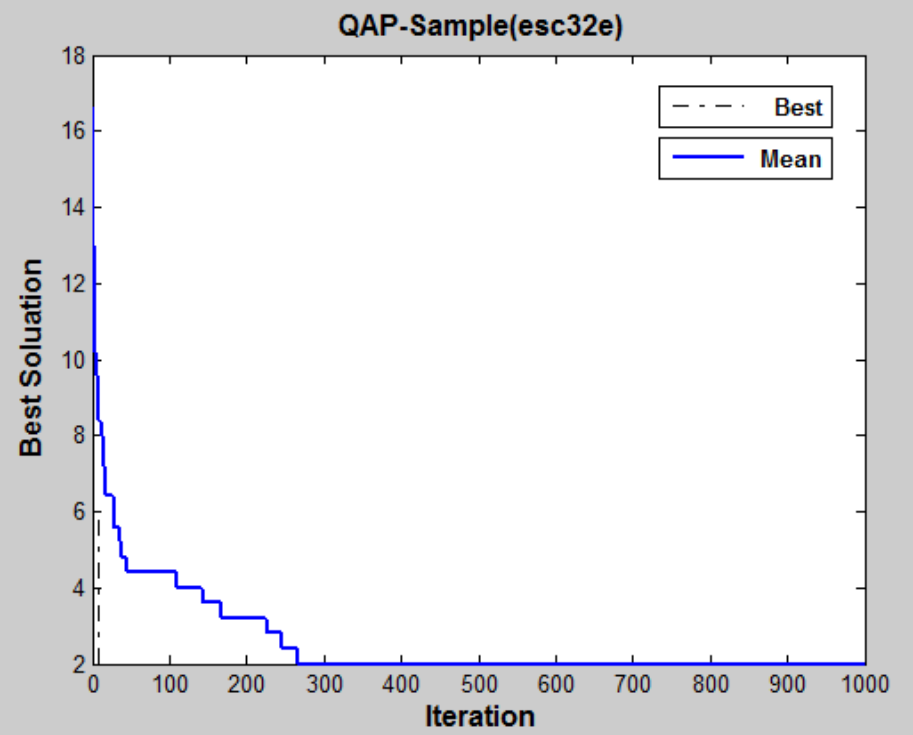
جدول ۴-۷- نتایج اجرای الگوریتم GSA-GA بر روی ده مساله منتخب QAP

نام مساله	ابعاد مساله	BKS	DE(%) [38]	PSO(%) [39]	SS(%)	GA(%)	H14(%)	MBO(%)	GSA-GA(%)
Esc32e	32	2	0.00	0.00	0.00	0.00	0.00	0.00	**0.00**
Esc32g	32	6	0.00	0.00	0.00	0.00	0.00	0.00	**0.00**
Esc32h	32	438	15.07	5.02	4.57	0.91	5.94	0.00	0.45
Esc64a	64	116	24.14	8.62	12.07	5.17	31.03	0.00	**0.00**
Tai64c	64	1855928	0.19	0.32	4.89	1.78	5.23	0.00	**0.00**
Lipa40b	40	476581	24.73	22.46	21.32	21.74	21.15	5.29	**0.00**
Sko49	49	23386	12.50	7.92	8.04	5.94	8.41	1.27	**1.07**
Wil50	50	48816	7.58	4.17	3.42	3.48	4.15	0.57	**0.34**
Lipa70a	70	169755	1.76	1.41	1.49	1.38	1.44	0.81	**0.81**
Lipa80a	80	253195	1.59	1.28	1.29	1.32	1.32	0.74	0.78
میانگین خطا			8.75	5.12	5.70	4.17	7.86	0.86	**0.34**

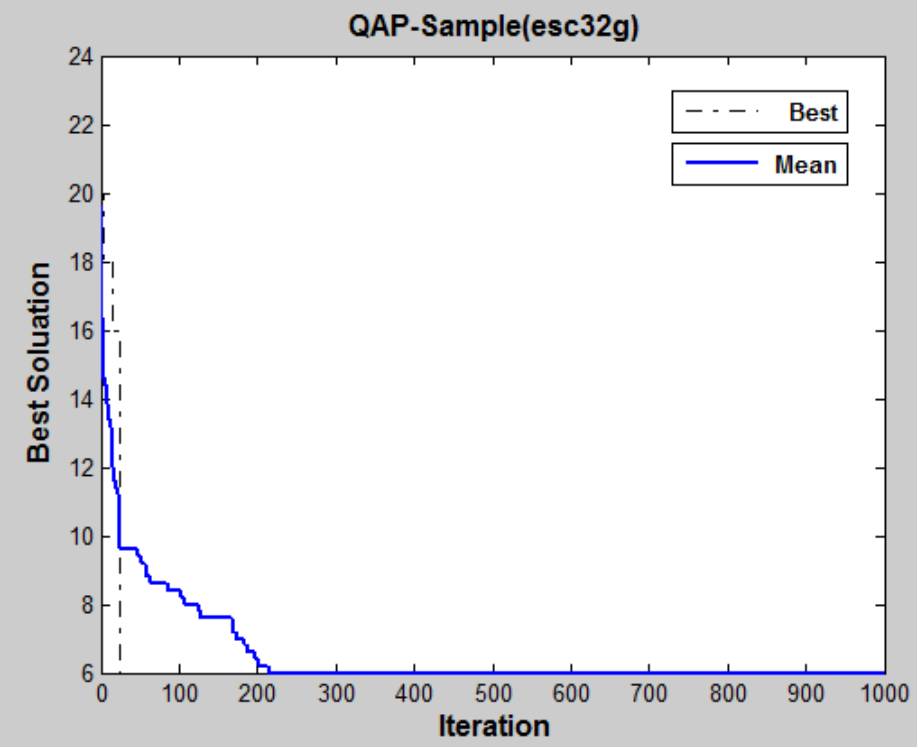
جدول ۴-۸- مقایسه نتایج اجرای الگوریتم GSA-GA با الگوریتم های مشابه

همانطور که در جدول مشخص است الگوریتم ارائه شده در اکثر موارد عملکردی مشابه با بهترین الگوریتم (الگوریتم مهاجرت پرندگان) داشته و حتی در برخی از نمونه ها عملکرد بهتری از الگوریتم مذکور داشته است. و همچنین مشاهده می شود که در مجموع ۱۰ مورد بررسی شده به صورت میانگین الگوریتم ترکیبی ما خطای کمتری از بقیه الگوریتم ها داشته است. شایان ذکر است که برای مقایسه، نتایج الگوریتم با ۱۰۰۰ تکرار در نظر گرفته شده بود، در صورتی که به جای آن از نتایج با ۲۰۰۰ تکرار استفاده می کردیم برتری الگوریتم ترکیبی ما مشهودتر بود.

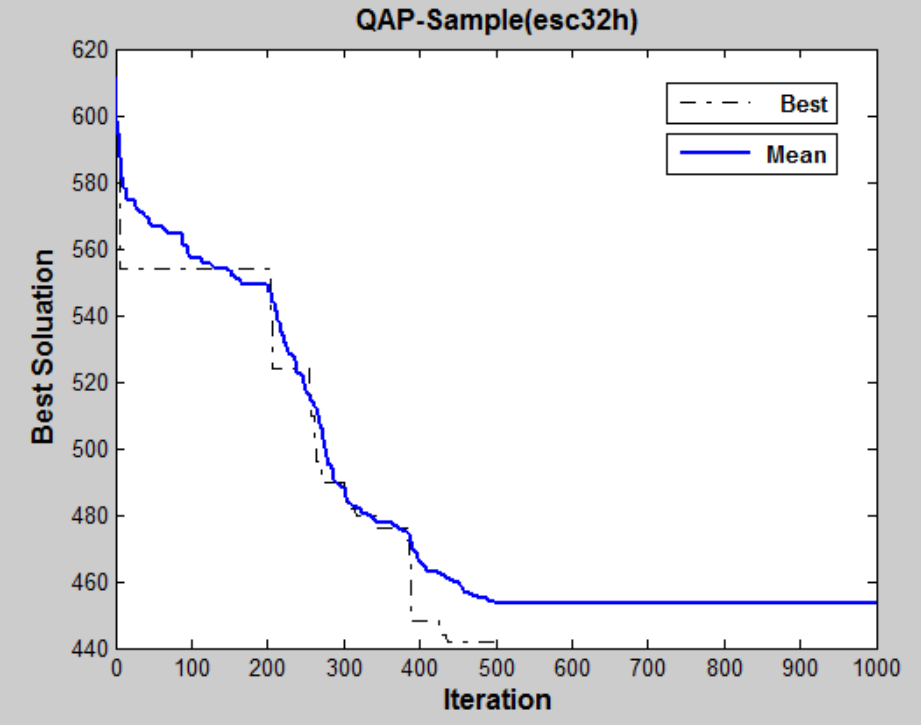
نمودار سرعت همگرایی برای مسائل جدول ٤-٧ به صورت زیر می باشد.



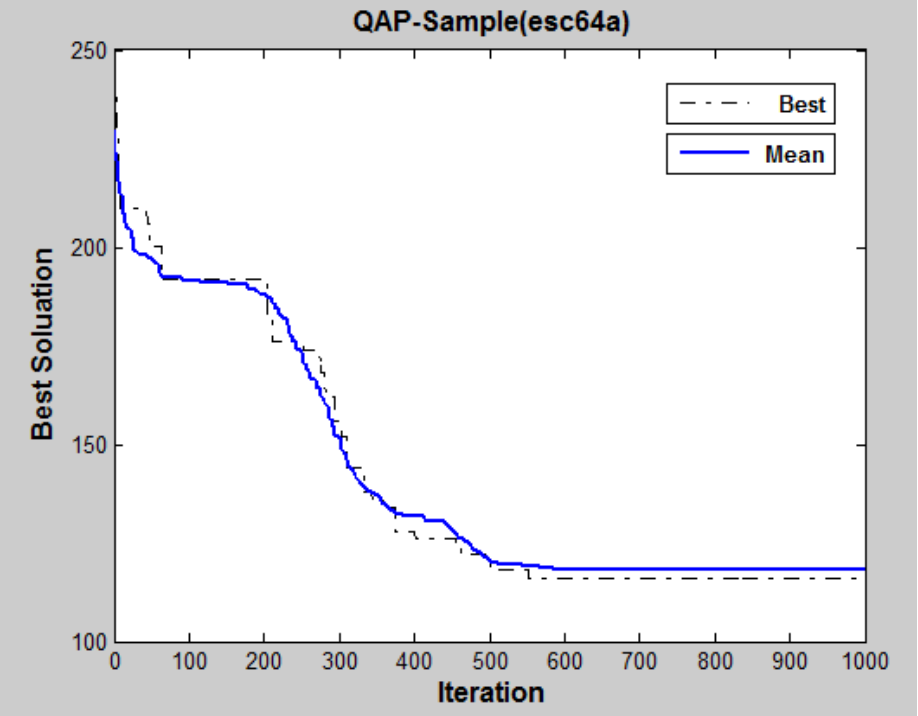
نمودار ۴-۲۳- نمودار سرعت همگرایی الگوریتم GSA-GA در حل مساله esc32e



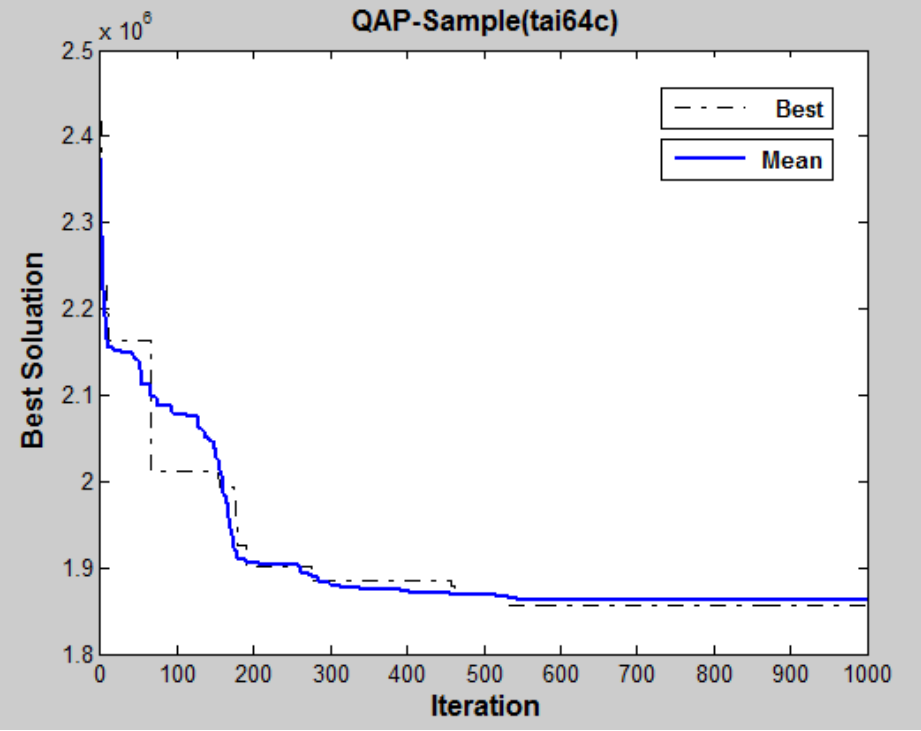
نمودار ۴-۲۴- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **esc32g**



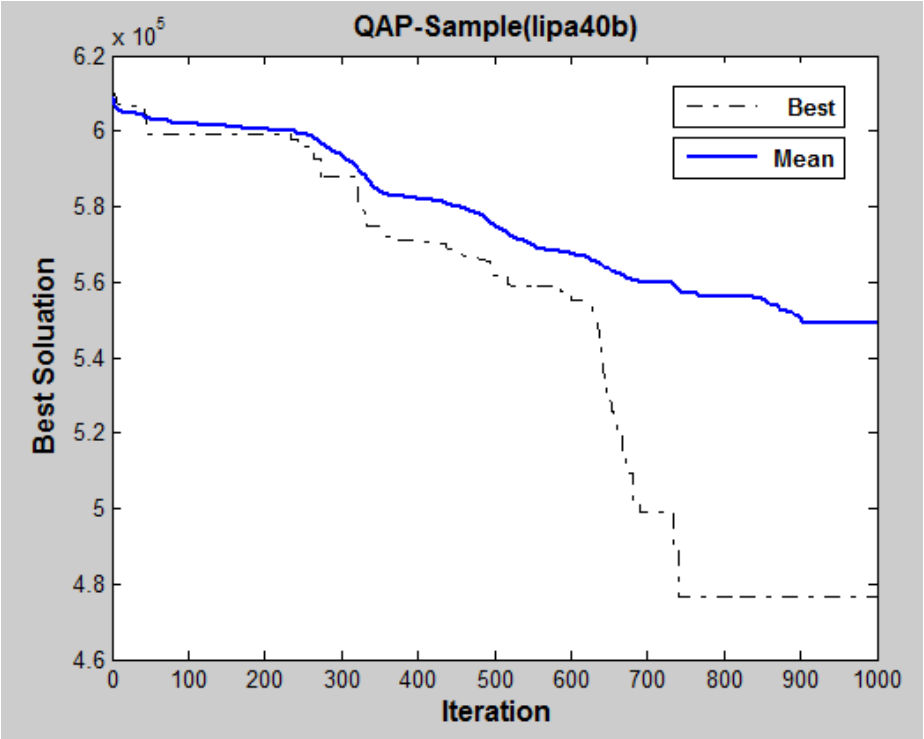
نمودار ۴-۲۵- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **esc32h**



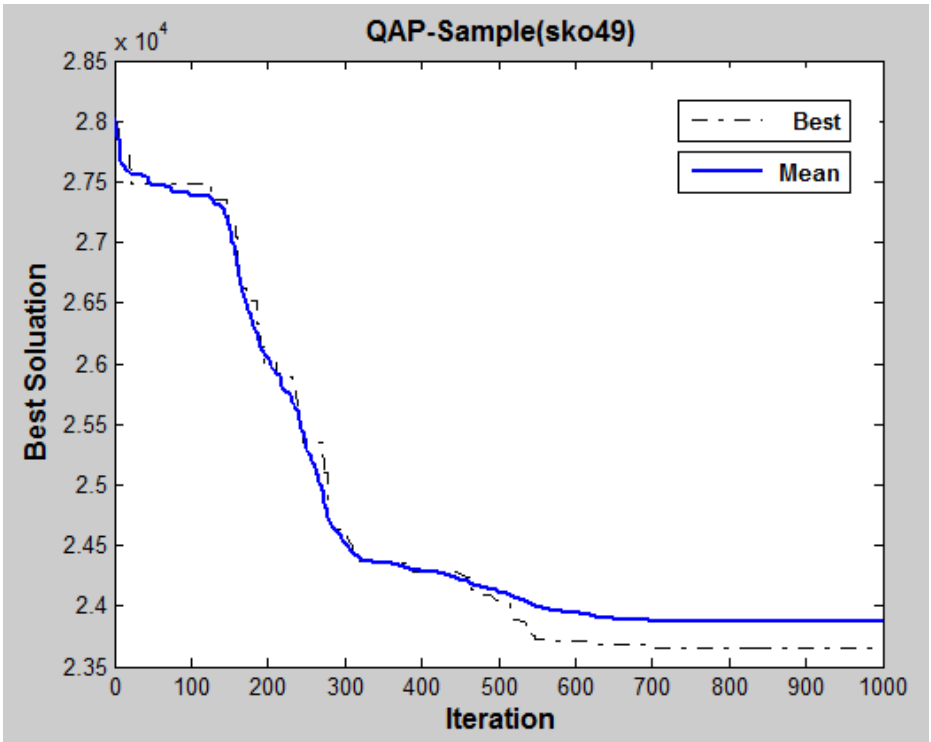
نمودار ۴-۲۶- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **esc64a**



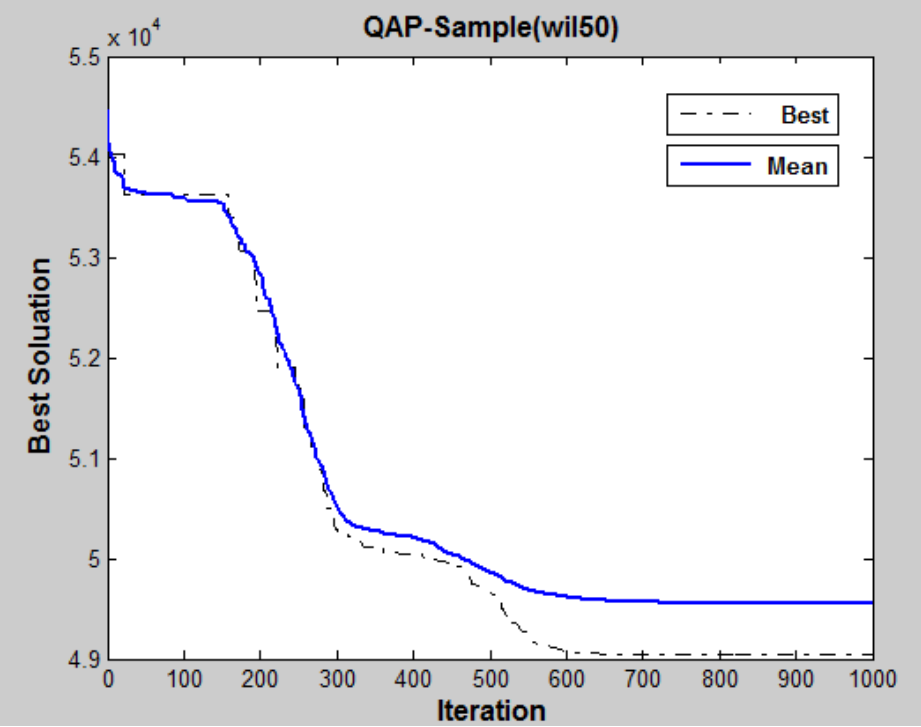
نمودار ۴-۲۷- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **tai64c**



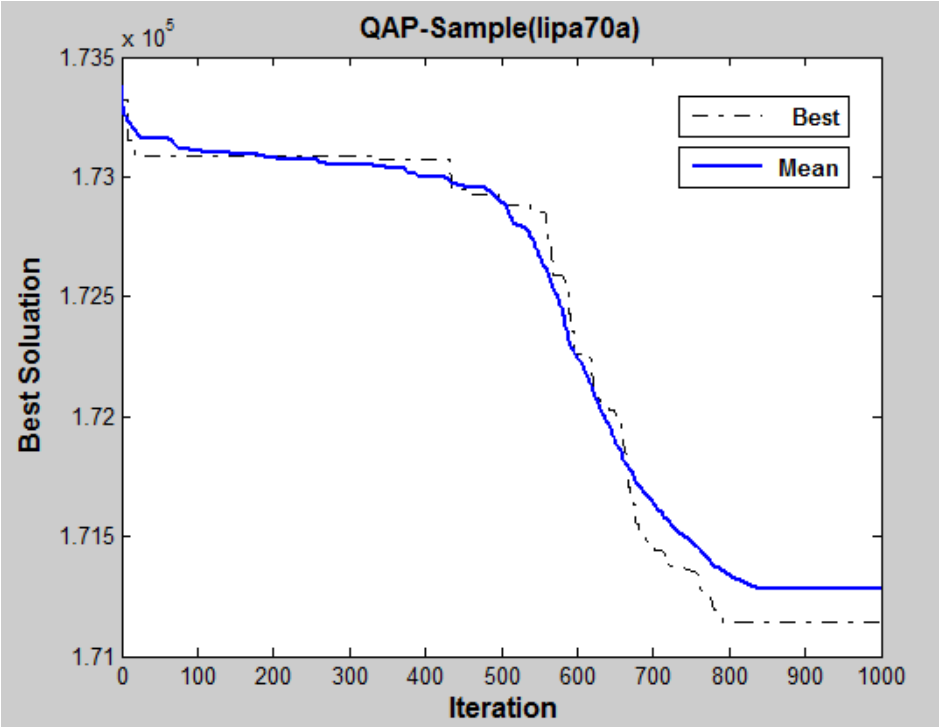
نمودار ۴-۲۸- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **lipa40b**



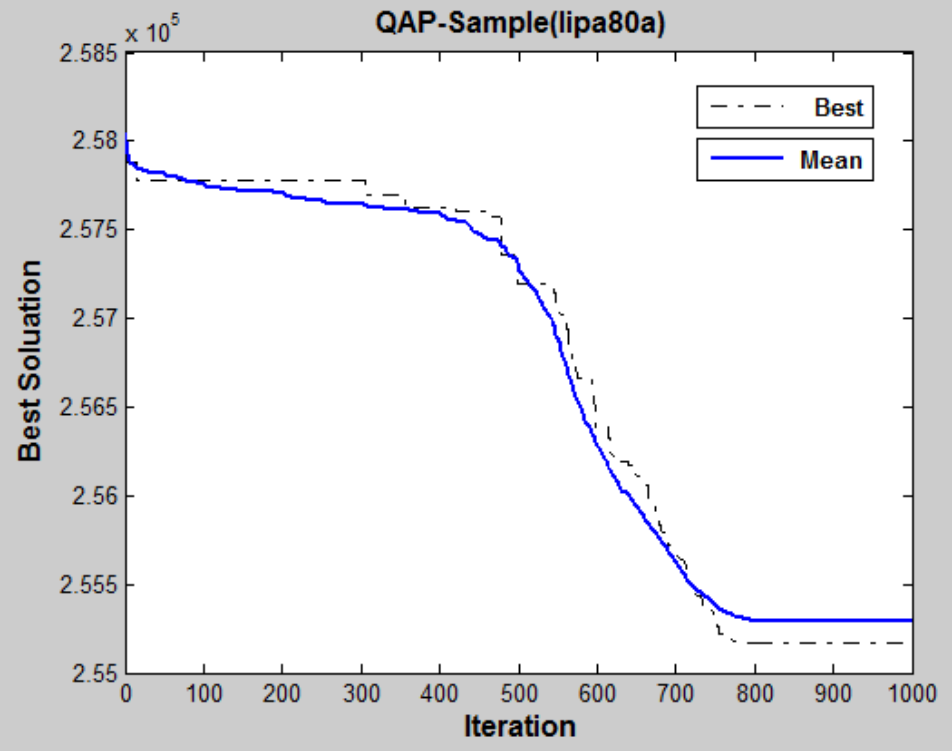
نمودار ۴-۲۹- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **sko49**



نمودار ۴-۳۰- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **wil50**



نمودار ۴-۳۱- نمودار سرعت همگرایی الگوریتم **GSA-GA** در حل مساله **lipa70a**



نمودار ۴-۳۲- نمودار سرعت همگرایی الگوریتم GSA-GA در حل مساله lipa80a

ملاحظه می شود که علاوه بر سرعت همگرایی مطلوب، مشکل گیر افتادن در بهینه محلی که در الگوریتم GSA استاندارد بود در الگوریتم پیشنهادی تا حدود زیادی رفع شده و خطای الگوریتم در حل مسائل مورد نظر بسیار پایین تر آمده است.

٤-٤ ارزیابی عملکرد الگوریتم ارائه شده در مقایسه با جدیدترین الگوریتم های به کار رفته برای QAP

پس از مقایسه نتایج الگوریتم پیشنهادی با این چند الگوریتم مطرح و کارآمد در حل مسائل QAP و مشاهده قدرت آن در حل مسائل منتخب، در آزمایش بعدی نتایج این الگوریتم را روی مسائل دیگری در مقایسه با الگوریتم BAT [۳۲] که جز جدیدترین الگوریتم های به کار رفته تا به امروز برای حل مسائل QAP می باشد، مورد ارزیابی قرار می دهیم.

نتایج این مقایسه در جدول زیر آمده است:

نام مساله	BKS	میانگین جواب BAT	میانگین جواب GSA-GA	بهترین جواب BAT	بهترین جواب GSA-GA	انحراف جواب BAT	انحراف جواب GSA-GA	درصد خطا جواب BAT	درصد خطا جواب GSA-GA
Bur26a	5426670	5530825	5446500	5519108	5435688	104155	8962	1.90	**0.36**
Bur26b	3817852	3898402	3832900	3895507	3817852	80550	7274	2.10	**0.39**
Bur26c	5426795	5523970	5434400	5520289	5426801	97175	5783	1.79	0.14
Bur26d	3821225	3887216	3831400	3879814	3821667	65991	1040	1.72	**0.26**
Bur26e	5386879	5500460	5402400	5491088	5387539	113581	16179	2.10	**0.28**
Bur26f	3782044	3850692	3799100	3830614	3782680	68648	12869	1.80	**0.45**
Bur26g	10117172	10328699	10145000	10320120	10118177	211527	28666	2.09	**0.27**
Bur26h	7098658	7267393	7126400	7251046	7098905	168735	27162	2.37	**0.39**
Esc32e	2	2	2	2	2	0	0	0	**0**
Es32g	6	6	6	6	6	0	0	0	**0**
Esc16a	68	68	68.8	68	68	0	1.03	0	1.17
Esc16b	292	292	292	292	292	0	0	0	**0**
Esc16c	160	160	161.4	160	160	0	1.64	0	0.87
Esc16d	16	16	16	16	16	0	0	0	**0**
Esc16e	28	28	28.6	28	28	0	1.26	0	2.1
Esc16f	0	0	0	0	0	0	0	0	**0**
Lipa20a	3683	3824	3779	3821	3756	141	14.23	3.8	**2.6**
Lipa30a	13178	13618	13437	13614	13414	440	104.2	3.3	**1.96**

جدول ۴-۹- مقایسه نتایج اجرای الگوریتم GSA-GA با الگوریتم BAT

همانطور که ملاحظه می شود در مجموع ۱۹ مورد ارزیابی، به جز ۳ مورد در باقی موارد الگوریتم GSA-GA درصد خطای میانگین برابر یا بهتری نسبت به الگوریتم BAT داشته است.در صورتی ملاک ارزیابی را درصد خطای بهترین جواب در نظر می گرفتیم در تمامی موارد این برتری قابل رویت بود.

٤-٥ بررسی عملکرد الگوریتم بر روی نمونه مسائل دیگر

نام مساله	ابعاد مساله	BKS	بهترین جواب GSA-GA	میانگین جواب GSA-GA	درصد خطای بهترین	درصد خطای میانگین	σ
Had12	12	1652	1652	1654	0	0.12	0.22
Had14	14	2724	2724	2726	0	0.07	0.07
Had16	16	3720	3720	3723	0	0.08	0.16
Had18	18	5358	5358	5396	0	0.70	0.39
Had20	20	6922	6922	6953	0	0.44	0.25

جدول ۴-۱۰- نتایج حاصل از اجرای الگوریتم GSA-GA بر روی برخی مسائل گروه *had

نام مساله	ابعاد مساله	BKS	بهترین جواب GSA-GA	میانگین جواب GSA	درصد خطای بهترین	درصد خطای میانگین	σ
Nug12	12	578	578	583	0	0.86	1.18
Nug15	15	1150	1150	1194	0	3.82	2.31
Nug16b	16	1240	1240	1273	0	2.66	1.64
Nug20	20	2570	2570	2649	0	3.07	1.24
Nug25	25	3744	3754	3818	0.26	1.97	1.15
Nug30	30	6124	6146	6279	0.35	2.53	0.85

جدول ۴-۱۱- نتایج حاصل از اجرای الگوریتم GSA-GA بر روی برخی مسائل گروه *nug

الگوریتم پیشنهادی خود را بر روی دو گروه دیگر از مسائل QAP که در آزمایشات قبلی مورد آزمایش قرار نگرفته بودند امتحان کرده و نتایج آن را در جداول ٤-٨ و ٤-٩ مشاهده می کنید. همانطور که قابل مشاهده است الگوریتم ارائه شده عملکرد قابل قبولی داشته و در اکثر موارد توانسته بهترین مقدار به دست آمده تا کنون را تکرار کند و در باقی موارد هم خطای بهترین جواب الگوریتم پیشنهادی کمتر از ۰,۵ درصد می باشد که نشان از قدرت بالای این الگوریتم در حل نمونه های مختلف مسائل QAP دارد.

فصل پنجم:نتیجه گیری و پیشنهادات

۵-۱ مقدمه

در فصل یک این پژوهش کلیاتی از موضوع پایان نامه ارائه گردید. در فصل دوم مروری بر ادبیات موضوع مساله تخصیص درجه دوم به تفصیل مورد بررسی قرار گرفت و روش های حل دقیق، ابتکاری و فرا ابتکاری برای آن معرفی گردید. در فصل سوم الگوریتم جستجوی گرانشی و مبانی آن مطرح شد و همچنین روش استفاده از این الگوریتم برای مسائل گسسته بیان شد و در نهایت برای رفع مشکل همگرایی این الگوریتم روشی ترکیبی ارائه گردید تا از قدرت بهرمنده ای الگوریتم ژنتیک استفاده شود. و در فصل چهارم عملکرد این الگوریتم پیشنهادی در حل مسائل مختلف مساله تخصیص درجه دوم بر روی نمونه های بسیاری مورد ارزیابی قرار گرفت و نتایج محاسباتی آن که با استفاده از نرم افزار متلب کد نویسی شده بود با نتایج الگوریتم های دیگر مورد مقایسه قرار گرفت. در نهایت در این فصل به نتیجه گیری و پیشنهاد برای انجام پژوهش های آینده می پردازیم.

۵-۲ نتیجه گیری

مساله تخصیص درجه دوم یکی از مسائل بهینه سازی ترکیبی است که به بررسی تخصیص تعدادی تسهیل به تعدادی مکان میپردازد. موضوعی که در این مساله اهمیت ویژه ای دارد بحث هزینه های تخصیص می باشد که تلاش در این مساله برای حداقل سازی این هزینه هاست. حل اکثر مسائل بهینه سازی ترکیبی مشکل است زیرا این مسائل عمدتا مسائل با مقیاس بزرگ هستند و یا تجزیه آنها به مسائل کوچک تر دشوار است،از این رو عمدتا برای حل آنها از روش های فرا ابتکاری استفاده می شود.

در این پژوهش به منظور حل مساله QAP الگوریتم جستجوی گرانشی ترکیبی پیشنهاد شده است که در آن از عملگرهای جهش و تقاطع الگوریتم ژنتیک جهت جلوگیری از به دام افتادن الگوریتم GSA در بهینه محلی

استفاده شده است. همان گونه که از نتایج حاصل از محاسبات بر می آید عملکرد الگوریتم پیشنهادی به عوامل مختلفی همچون پارامترهای الگوریتم مانند تعداد اجرام(عامل ها)، تعداد دفعات تکرار الگوریتم، و همچنین به عواملی همچون ماهیت مساله مورد نظر و ابعاد آن بستگی دارد. در آزمایشات اول تاثیر ماهیت مساله و ابعاد مساله را با اجرای الگوریتم مورد نظر برای مسائل گروه Lipa*a و sko* مورد بررسی قرار دادیم،که مشاهده شد درصد خطای جواب به دست آمده با افزایش ابعاد مسائل در گروه Lipa*a کاهش یافت، در حالی که برای گروه sko* این عمل صورت نگرفت که نشان می دهد عملکرد الگوریتم مورد نظر علاوه بر ابعاد مساله به ماهیت مساله نیز بستگی دارد. مسائل lipa*a از ماتریس جریانی برخوردار است که شباهت به یک ماتریس واحد دارد (اما ماتریس واحد نیست) ولی ماتریس جریان در مسائل sko* یک ماتریس بالا مثلثی است که شباهتی هم به ماتریس واحد ندارد. در آزمایشات بعدی تاثیر پارامتر N (تعداد عامل ها) و تعداد تکرار های الگوریتم ارائه شده بر عملکرد آن مورد ارزیابی قرار گرفت. که مشاهده شد در اکثر موارد با افزایش تعداد عامل ها و همینطور افزایش تعداد تکرار های الگوریتم مورد نظر همان گونه که انتظار می رفت عملکرد الگوریتم بهتر شده و خطای جواب به دست آمده کاهش می یابد. این بهبود در هر دو حالت تا یک محدوده ای شدت بیشتری داشته و پس از آن الگوریتم تقریبا به یک ثباتی می رسد که افزایش تعداد عامل ها و تعداد تکرارهای الگوریتم اگرچه باعث بهبود جزیی می شود اما این بهبود قابل توجه نبوده است.

پس از تعیین پارامترهای مناسب، آزمایش های متعددی بر روی مسائل منتخب در QAPLIB صورت گرفت که نتایج این آزمایش ها نشان داد در مقایسه با سایر الگوریتم های فرا ابتکاری مثل MBO و BAT از دقت بالاتر (خطای کمتری) برخوردار است.

در مجموع الگوریتم پیشنهادی برای حل مساله تخصیص الگوریتم عملکرد قابل قبولی از خود نشان داد.

۵-۳ پیشنهاد برای پژوهش های آتی

در الگوریتم GSA پارامتر های K و α فاکتور های مهمی در تنظیم دو خاصیت بهره برداری و کاوش می باشند (مقدار ثابت گرانش G به پارامتر α وابسته است) که تنظیم مناسب این پارامترها می تواند عملکرد آن را بهبود بخشید لذا توصیه می شود در پژوهش های آتی این پارامتر ها را به وسیله کنترلرهای منطق فازی کنترل نمود تا به این وسیله عملکرد الگوریتم GSA را بهبود داده و سرعت همگرایی آن را افزایش داد.

مراجع

[1]. Commander. Clayton W,"A Survey of the Quadratic Assignment Problem, with Applications" , Morehead Electronic Journal of Applicable Mathematics,University of Florida, Gainesville, FL 32611, (2005).

[2]. Kammerdiner.Alla, Gevezes.Theodoros, Pasiliao.Eduardo, Pitsoulis.Leonidas, Pardalos.Panos M., "Quadratic Assignment Problem", Springer Science+Business Media New York,(2013).

[3]. Loiola.Eliane Maria , de Abreu.Nair Maria Maia, Boaventura-Netto.Paulo Oswaldo , Hahn.Peter , Querido.Tania , "A survey for the quadratic assignment problem" , European Journal of Operational Research ,(2007), pp.657-690

[4]. Rainer E. Burkard, Eranda Çela, Panos M. Pardalos, Leonidas S. Pitsoulis, "Handbook of Combinatorial Optimization", Springer(1999), Vol 1.

[5]. فتاحی – پرویز، الگوریتم های فرا ابتکاری، ۱۳۸۸، انتشارات دانشگاه بو علی سینا،چاپ اول.

[6]. البرزی – محمود، الگوریتم ژنتیک، ۱۳۸۸، موسسه انتشارات دانشگاه صنعتی شریف،چاپ اول.

[7]. R. Raidl Gunther,"A Unified View on Hybrid Metaheuristics", Springer-Verlag Berlin Heidelberg (2006), pp.1-12.

[8]. Loiola E.M, Maia de Abreu N.M, Boaventura-Netto P.O, Hahn P. and Querido T,"A survey for the quadratic assignment problem",European Journal of Operational Research(2007), Vol.176, pp.657-690.

[9]. Muenvanichakul.Sirirat,Charnsethikul.Peerayuth, "The Approximated Dynamic Programming Approach to the Dynamic Quadratic Assignment Problem", Thammasat Int. J. Sc. Tech(2007), Vol.12, No.2.

[10]. Wu-Ji.Li,MacGregor.Smith,"An algorithm for quadratic assignment problems", European Journal of Operational Research (1995), Vol.81, pp.205-216.

[11]. Hussin.M.S,Stutzle.Thomas, "Hierarchical Iterated Local Search for the Quadratic Assignment Problem",Springer(2009), pp.115-129.

[12]. Li Hui, Landa-Silva Dario, "An Elitist GRASP Metaheuristic for the Multi-objective Quadratic Assignment Problem",Springer-Verlag Berlin Heidelberg(2009), pp. 481-494.

[13]. Alves Pessoa .Artur, M. Hahn. Peter, Guignard .Monique, Zhu .Yi-Rong, "Algorithms for the generalized quadratic assignment problem combining Lagrangean decomposition and the Reformulation-Linearization Technique", European Journal of Operational Research (2010), Vol.206, pp.54-63.

[14]. Ramkumar .A.S, Ponnambalam. S.G, Jawahar .N, "A new iterated fast local search heuristic for solving QAP formulation in facility layout design", Robotics and Computer-Integrated Manufacturing(2009), Vol.25, pp. 620- 629.

[15]. Arkin.Esther M,Hassin.Refael, Sviridenko.Maxim,"Approximating the maximum quadratic assignment problem", Information Processing Letters(2001), Vol.77, pp.13-16.

[16]. Yang.Xiaofan, Lu.Qing, Li.Chuasndong, Liao .Xiaofeng,"Biological computation of the solution to the quadratic assignment problem", Applied Mathematics and Computation (2008), Vol.200, pp.369-377.

[17]. Zhang.Huizhen, Beltran-Royo. Cesar, Constantino. Miguel,"Effective formulation reductions for the quadratic assignment problem", Computers & Operations Research (2010), Vol.37, pp. 2007-2016.

[18]. Demirel.Nihan Çetin, Toksar.M. Duran, "Optimization of the quadratic assignment problem using an ant colony algorithm", Applied Mathematics and Computation (2006), Vol.183, pp. 427-435.

[19]. Liu .Linzhon, Li.Yinzhen,"The fuzzy quadratic assignment problem with penalty: New models and genetic algorithm",Applied Mathematics and Computation(2006), Vol.174, pp.1229-1244.

[20]. Özbakir. Lale,Baykasoğlu.Adil,Tapkan.Pinar, "Bees algorithm for generalized assignment problem", Applied Mathematics and Computation (2010), Vol.215, pp.3782-3795.

[21]. Moe.R, "GRIBB – Branch and Bound methods on the Internet", Parallel Processing and Applied Mathematics (2003), pp.1020-1027.

[22]. Li.Yong,M. Pardalos. Panos, Resende .Mauricio G.C, "A Greedy Randomized Adaptive Search Procedure for the Quadratic Assignment Problem", DIMACS Series in Discrete Mathematics and Theoretical Computer Science,(1993).

[23]. B. Schutz, "Gravity from the Ground Up", Cambridge University Press,(2003).

[24]. D. Holliday, R. Resnick, J. Walker, "Fundamentals of physics", John Wiley and Sons,(1993).

[25]. H. Zaied.bdel Nasser, Shawky.Laila Abd El-Fatah ,"A Survey of the Quadratic Assignment Problem", International Journal of Computer Applications,(2014), No.6.

[26] Bashiri.Mahdi, Karimi.Hossein," Effective heuristics and meta-heuristics for the quadratic assignment problem with tuned parameters and analytical comparisons" , Journal of Industrial Engineering International,(2012).

[27]. Duman.Ekrem, Uysal.Mitat, Alkaya.Ali Fuat , "Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem", Information Sciences 217(2012),pp.65-77.

[28]. Bernardino.E.M., Bernardino.A.M, Sánchez-Pérez J.M., Gómez-Pulido J.A, Vega-Rodriguez M.A, "Hybrid Honey Bee Mating Optimization Algorithm to assign terminals to concentrators", Applied Sciences in Biomedical and Communication Technologies,IEEE(2010),pp.1-7.

[29]. Birattari.M,Paquete.L,Strutzl.T,Varrentrapp.K,"Classification of Metaheuristics and Design of Experiments for the Analysis of Components",Tech. Rep. AIDA-01-05, (2001).

[30]. Gamal.Abd El-Nasser A. Said, Abeer.M.Mahmoud, El-Sayed.M.El-Horbaty,"A Comparative Study of Meta-heuristic Algorithms for Solving Quadratic Assignment Problem", International Journal of Advanced Computer Science and Applications,(2014).

[31]. P.B.S. Lissaman, C.A. Shollenberer, "Formation flight of birds", Science 168 (1970),pp.1003-1005.

[32]. Shukla.Apurv, "A modified Bat Algorithm for the Quadratic Assignment Problem",Evolutionary Computation(CEC), IEEE Congress on(2015),pp.486 – 490.

[33]. P. Seiler, A. Pant, J.K. Hedrick, "A systems interpretation for observations of bird V-formations", Journal of Theoretical Biology 221 (2003),pp.279-287.

[34]. J.M.V. Rayner,"A new approach to animal flight mechanics", Journal of Experimental Biology 80 (1979),pp.17-54.

[35]. B. M. Kıyıcıgˇı, E. Duman, A.F. Alkaya, "Finding best performing solution algorithm for the QAP", Proceedings of IMS2010, Sarajevo, Bosnia Herzegovina, (2010), pp. 510-522.

[36]. Q.-K. Pan, M.F. Tasgetiren, Y.-C. Liang,"A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem", Computers & Operations Research 35 (9) (2008),pp.2807-2839.

[37].Rashedi.Esmat,Nezamabadi-pour.Hossein,Saryazd.Saeid,"GSA:A Gravitational Search Algorithm" , Information Sciences 179 (2009),pp.2232-2248.

[38]. Mansouri.R, Nasseri.F,Khorrami.M,"Effective time variation of G in a model universe with variable space dimension", Physics Letters 259 (1999),pp.194-200.

[39]. Kenyon.I.R,"General Relativity", Oxford University Press,(1990).

[40]. Yugal.kumar,Sahoo.G,"A Review on Gravitational Search Algorithm and its Applications to Data Clustering & Classification",I.J. Intelligent Systems and Applications,(2014),pp.79-93.

[41]. Glover.F, Kochenberger.G.A, "Handbook of Metaheuristics", Kluwer (2003).

[42]. Rashedi.E,"Gravitational Search Algorithm", M.Sc. Thesis, Shahid Bahonar University of Kerman, Kerman, Iran, (2007) (in Farsi).

[43]. Chen Huiqin, Li Sheng and Tang Zheng,"Hybrid Gravitational Search Algorithm with Random-key Encoding Scheme Combined with Simulated Annealing", IJCSNS International Journal of Computer Science and Network S 208 ecurity,(2011),vol.11,No.6.

[44]. Aizhu.Zhang, Genyun.Sun, Zhenjie Wang, Yanjuan Yaoy,"a hybird algorithem and gravitational serach algorithem for global optimazation",Neural Network World1/15,(2015),pp.53-73.

[45]. Genyun.Sun,Aizhu.Zhang,"Hybrid Genetic Algorithm and Gravitational Search Algorithm for Image Segmentation Using Multilevel thresholding", Springer-Verlag Berlin Heidelberg(2013),pp. 707-714.

[46]. Q.-K. Pan, M.F. Tasgetiren, Y.-C. Liang,"A discrete differential evolution algorithm for the permutation flowshop scheduling problem", Computers & Industrial Engineering 55 (2008),pp.795-816.

Abstract:

Quadratic Assignment Problem (QAP) has been considered as one of the most complicated optimization problems. The problem is to assign all facilities to different locations with the goal of minimizing the sum of the distances multiplied by the corresponding flows (total cost). it is NP-Hard and the optimal solutions are not available for large-scale problems. In this research, we presented a metaheuristic algorithm called Gravitational Search Algorithm(GSA) for solve quadratic assignment problem. It is assumed that tasks are deterministic and facilities and location number are equal. It should be noted that any facility must be assign to only one location. As well as we use from a hybrid method that is combined GSA with Genetic algorithm(GA) called GSA-GA for solve this problem, and proposed algorithm tested on a number of benchmark problems obtained from the QAPLIB and in most cases it was able to obtain the best known solutions or it was less error than the similar algorithms.

Keywords: Gravitational Search Algorithm, Quadratic assignment problem, Hybrid Optimization, Metaheuristic methods, Genetic algorithm

Islamic Azad University

Qazvin Branch

Faculty of Science and Research – Department of Computer

M.S. Theses Dissertation on

Computer Course – Software

Subject:

Solving Quadratic Assignment Problem Using Hybrid Gravitational Search Algorithm

Supervisor:

Dr. Behrouz Masoumi

By:

Seyed Hamed Saei

Season:

Winter 2015