
بسم الله الرحمن الرحیم

دانشکده برق و رباتیک

رشته الکترونیک، گرایش الکترونیک سیستم

پایان‌نامه کارشناسی ارشد

جایابی و مسیریابی قطعات برد مدار چاپی با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات

نگارنده: میترا رفعت‌پور

استاد راهنما

دکتر علی سلیمانی ایوری

تیرماه ۱۳۹۶

این مجموعه را با سپاس بی کران تقدیم می کنم به

حضور ارزشمند پدر و مادر عزیزتر از جانم که با صبر و شکیبایی همواره پشتیبانم بوده اند و هر لحظه دعای خیرشان بدرقه راهم بوده است. از آن ها بخاطر تلاش ها و حمایت بی دریغ شان نهایت تشکر را دارم.

برادر عزیز و مهربانم که همیشه کنارم بوده است و حمایت او کمک زیادی به من کرده است. دستش را به گرمی می فشارم و از او تشکر می کنم.

سپاس گزاری

منت خدای راعزوجل که طاعتش موجب قربت است وبه شکر اندرش مزید نعمت. سپاس وستایش مخصوص آفریدگاری است که خویشتن رابه ماشناساند ودرهای علم رابرما گشود و عمری وفرصتی عطافرمود تابدان، بنده ی ضعیف خویش رادر طریق علم ومعرفت بیازماید.

پس از ارادت خاضعانه به درگاه خداوند بی همتا از زحمات استاد ارجمند جناب آقای دکتر علی سلیمانی ایوری، دانشیار دانشکده مهندسی برق و رباتیک که راهنمایی این پایان نامه رابر عهده داشتند، کمال تشکر رادارم.

از کلیه اساتید دانشکده مهندسی برق و رباتیک به خاطر تمامی زحماتشان در طول دوسال تحصیلم در دانشگاه صنعتی شاهرود، کمال تشکر رادارم.

ازپدرومادرعزیزم که همواره متحمل زحماتم بودند و دوستان بامحبتم که مرادر رسیدن به این مهم یاری کردند، بسیار سپاسگزارم.

تعهد نامه

اینجانب **میترا رفعت‌پور** دانشجوی کارشناسی ارشد رشته **الکترونیک** دانشکده **برق و رباتیک** دانشگاه شاهرود، نویسنده پایان‌نامه با عنوان **جایابی و مسیریابی قطعات برد مدار چاپی با استفاده از الگوریتم بهینه‌سازی ازدحام ذرات** ، تحت راهنمایی **علی سلیمانی ایوری** متعهد می‌شوم:

- تحقیقات در این پایان‌نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است.
- در استفاده از نتایج پژوهش‌های دیگر پژوهش‌گران، به مرجع مورد استفاده استناد شده است.
- مطالب این پایان‌نامه، تا کنون توسط خود، یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ‌جا ارایه نشده است.
- حقوق معنوی این اثر، به دانشگاه صنعتی شاهرود تعلق دارد، و مقالات مستخرج با نام‘‘ دانشگاه صنعتی شاهرود ‘‘ یا ‘‘ Shahrood University of Technology ‘‘ به چاپ خواهد رسید.
- حقوق معنوی تمام افرادی که در به‌دست آوردن نتایج اصلی پایان‌نامه تاثیرگذار بوده‌اند، در مقالات مستخرج از پایان‌نامه رعایت می‌گردد.
- در تمام مراحل انجام این پایان‌نامه، در مواردی که از موجود زنده (یا بافت‌های آنها) استفاده شده است، ضوابط و اصول اخلاقی رعایت شده است.
- در تمام مراحل انجام این پایان‌نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته (یا استفاده شده است)، اصل رازداری و اصول اخلاق انسانی رعایت شده است.

میترا رفعت‌پور
تیرماه ۱۳۹۶

مالکیت نتایج و حق نشر

- تمام حقوق معنوی این اثر و محصولات آن (مقالات مستخرج، کتاب، برنامه‌های رایانه‌ای، نرم‌افزارها و تجهیزات ساخته شده) متعلق به دانشگاه صنعتی شاهرود می‌باشد. این مطلب باید به نحو مقتضی، در تولیدات علمی مربوطه ذکر شود.
- استفاده از اطلاعات و نتایج موجود در این پایان‌نامه بدون ذکر منبع مجاز نمی‌باشد.

چکیده

جایابی قطعات و چیدمان موضوعی است که از سال‌ها پیش موردتوجه قرار گرفته است. این مسئله بر روی اینکه چگونه اشیا در یک فضای محدود منطقی تحت قیود مشخص قرار بگیرند مطالعه می‌کند. از نظر تئوری مسائل چیدمان از نوع NP-Complete هستند. با این نوع مسائل در بسیاری از رشته‌های مهندسی مواجه می‌شویم و از اهمیت زیادی برخوردار هستند و در زمینه‌های کاربردی مختلف مثل طراحی چیدمان ماژول‌های فضاپیما، تجهیزات ماشینی و کارخانه‌ای، سیستم‌های حفاری دریایی، حمل و نقل، وسایل نقلیه و ربات موردنیاز هستند. این مسائل معمولاً به طور مستقیم برخی شاخص‌های انجام طراحی، از جمله قابلیت اطمینان و اقتصاد را تحت تاثیر قرار می‌دهند. از جمله کاربردهای این مسائل نیز می‌توان در جایابی و مسیریابی قطعات روی بردهای مداری بهره برد. امروزه استفاده از مدارهای چاپی در صنعت به سرعت رو به پیشرفت است و ابعاد این بردها روز به روز کوچک‌تر می شود. در نتیجه نیاز به نرم افزارهایی برای طراحی خودکار و بهینه‌ی این بردها وجود دارد.

در این پایان‌نامه با استفاده از الگوریتم بهینه سازی ازدحام ذرات که این الگوریتم از هوش جمعی بهره می‌برد، اهدافی همچون کوچک کردن فضای برد از طریق کاهش سیم‌بندی‌ها و نزدیک کردن قطعات به هم مدنظر قرار گرفته است. برخی محدودیت‌های اضافی از جمله قیدهایی برای تعادل، اتصال و حالت‌های مجاور هم یا متوالی در نظر گرفته می‌شود. در نظر گرفتن همه‌ی قیدها و محدودیت‌های مدنظر از جمله چالش‌های پیش روی این مسئله است، چرا که با زیاد شدن قیدها احتمال دور شدن از جواب بهینه زیاد می‌شود و نیز اجرای الگوریتم به زمان بیشتری نیاز دارد. یافتن ارتباطی منطقی بین اهداف و پیاده‌سازی ریاضی توابع مربوط به آن‌ها بسیار پیچیده است. نتایج نشان می‌دهد این الگوریتم با در نظر گرفتن هدف‌ها و قیدهای درست، در حل مسئله‌ی جایابی به خوبی عمل می‌کند. در اینجا آزمایش‌هایی از قبیل ترکیب این الگوریتم با عملگر جهش برای بهبود نتایج و همچنین ترکیب با الگوریتم ژنتیک نیز صورت گرفته است. همچنین مقایسه‌ای بین عملکرد الگوریتم موردنظر با یک الگوریتم ترکیبی HGAPSO صورت گرفته است.

کلمات کلیدی: جایابی قطعات[۱]، طراحی چیدمان[۲]، هوش جمعی، الگوریتم‌های تکاملی، الگوریتم بهینه سازی ازدحام ذرات[۳]، برد مدار چاپی[۴]، نرم‌افزار آلتیوم

[۱] Component Placement
[۲] Layout Design
[۳] Particle Swarm Optimization
[۴] Printed Circuit Board

فهرست مطالب

فهرست تصاویر

فهرست جداول

فصل ۱

مقدمه و پیشینه‌ی تحقیق

۱.۱ مقدمه

برد مدار چاپی سطح مکانیکی نگه‌دارنده‌ی قطعات الکترونیکی است که مکان قرارگیری قطعات و مسیرهای ارتباطی بین آن‌ها از طریق پوشش رسانا (معمولاً مس) روی صفحه‌ی نارسانا (معمولاً فیبر) مشخص شده است. دلیل تولید و توسعه پیدا کردن مدارهای چاپی داشتن یک فضای مناسب برای نصب قطعات بوده است. اولین مدارهای چاپی در ابتدای قرن بیستم ساخته شده‌اند. همزمان با پیشرفت تکنولوژی و استفاده از تراشه‌ها و قطعات مختلف در بسته‌بندی‌های مختلف، تکنولوژی ساخت مدارهای چاپی نیز تحول زیادی پیدا کرده است. امروزه اهمیت مدارهای چاپی از آن جهت بسیار زیاد است که تقریباً در تمامی وسایل الکتریکی و الکترونیکی، از بردهای مدار چاپی برای برقراری ارتباط بین ادوات الکتریکی استفاده می‌شود. استفاده از بردهای مدار چاپی سبب افزایش سرعت و دقت و آسانی در مونتاژ، کاهش نویز، جلوگیری از اشتباهات انسانی و مزایای متعدد دیگر می‌گردد.

یکی از چالش‌هایی که مهندسان و طراحان با آن رو به رو هستند، تولید مدارهای چاپی باکیفیت و با قابلیتهای بالا است. برای طراحی مدارهای چاپی، علاوه بر به دانش و تخصص فنی که در زمینه‌ی مربوطه نیاز است، هنر و خلاقیت مهندسی نیز در آن دخیل است. نرم‌افزارهای مختلفی توسعه پیدا کرده‌اند که برای سرعت بخشیدن به فرایند طراحی با امکانات حرفه‌ای مورداستفاده قرار می‌گیرند. برخی از مهم‌ترین این نرم‌افزارها عبارتند از:

- Altium Designer DXP

- Eagle
- Cadence Orcad PCB Designer
- Diptrack
- Ads

با وجود در دسترس بودن نرم‌افزارهای مختلف برای طراحی مدارهای چاپی که قابلیت مسیریابی خودکار[۱] مدار را نیز دارند، اکثر طراحان مدار ترجیح می‌دهند از این قابلیت استفاده نکنند و به خاطر معایبی که مسیریاب‌های خودکار دارند، همچنان جایابی قطعات به طور دستی انجام می‌شود. جایابی دستی معایب و مزایایی دارد. از جمله مزایای آن این است که بنا بر سلیقه و نیاز، می‌توان طراحی مدار را با رعایت استانداردها و قوانین مشخص طراحی به شکل دلخواه انجام داد، اما این همواره کاری زمان‌بر است و تنها افراد متخصص و باتجربه می‌توانند طراحی را انجام دهند. در نتیجه نیاز به بهبود نرم‌افزارها و دستگاه‌های مسیریابی و جایابی قطعات مشهود است. در سال‌های اخیر سعی در بهبود این روش‌ها صورت گرفته است. در این پایان‌نامه قصد بر این است که روشی برای جایابی مناسب قطعات با رعایت اصولی که مدنظرمان است و در بخش اهداف پایان‌نامه شرح داده شده است ارائه شود.

۲.۱ بیان مسئله

با توجه به کاربرد زیاد و نیاز روزافزون به بردهای مدارچاپی در تکنولوژی و صنعت ساخت مادربردها و دیگر بردهای الکترونیکی مثل بردهای گوشی موبایل، مسئله‌ی ساخت بردهایی با ابعاد کوچکتر و درعین حال سیم‌بندی مناسب قابل‌توجه است. کاری که در این پایان‌نامه انجام می‌گیرد بهینه سازی و کاهش ابعاد یک برد مداری از طریق جایابی و مسیریابی مناسب قطعات است.

هدف از انجام این پایان‌نامه این است که با استفاده از الگوریتم بهینه سازی ازدحام ذرات که یکی از روش‌های بهینه سازی تکاملی است، سعی در رسیدن به حداقل ابعاد یک برد با تعداد قطعات مشخص و محدود داریم. برای این‌کار باید تابع هدف مناسب برای حل چنین مسئله‌ای را پیدا کرد. علاوه بر ابعاد، مسائل دیگری همچون حداقل شدن سیم‌بندی‌ها و روی هم نیفتادن قطعات نیز برای ما اهمیت دارد. تابع هدفی که به دست می‌آوریم باید بتواند همه‌ی این نیازها را برآورده کند.

هر مسئله‌ی بهینه سازی علاوه بر اهداف و هزینه‌ها شامل چند محدودیت نیز هست. محدودیت‌ها، در واقع قیدهایی هستند که باعث می‌شوند فضای موردبحث از محدوده‌های خاصی که خواسته شده است خارج نگردد، و در طول اجرای هر الگوریتم باید به آن‌ها توجه کرد. به عنوان مثال بحث حذف تداخل که در بالا نام برده شد، از جمله‌ی این قیدها می‌باشد. همچنین قرار گرفتن بعضی قطعات در جاهای مشخص و نیز ابعاد برد که از حدی بیشتر نشود جزو قیدهایی هستند که باید رعایت شود.

[۱] Autoroute

۳.۱ مفهوم بهینه سازی

بهینه‌سازی به دریافت ورودی‌ها و خصوصیات یک دستگاه، فرایند ریاضی و یا آزمایش تجربی است، به نحوی که بهترین خروجی یا نتیجه به دست آید. ورودی‌ها، متغیرهای فرایند یا تابع مورد بررسی هستند که با نام‌های تابع هدف[۱]، تابع هزینه[۲] و یا تابع برازندگی[۳] نامیده می‌شود. خروجی نیز به صورت هزینه، سود و یا برازندگی تعریف می‌شود. غالب مسائل بهینه‌سازی به صورت کمینه‌سازی مقدار یک تابع هزینه در نظر گرفته شده‌اند. به راحتی می‌توان نشان داد که هر نوع مسئله‌ی بهینه‌سازی را می‌توان در قالب یک مسئله‌ی کمینه‌سازی تعریف نمود.

۱.۳.۱ مراحل حل یک مسئله‌ی بهینه سازی

قدم اصلی برای حل یک مسئله‌ی بهینه سازی این است که متغیرهای بهینه سازی و همچنین اهداف بهینه سازی تعریف شود. فرضاً مسئله‌ی ما، تنظیم آنتن تلویزیون به گونه‌ای است که تصویر دریافتی بیشترین کیفیت را داشته باشد. در بعضی موارد روش‌های سعی و خطا برای یافتن جواب بهینه استفاده می‌شود. اما در بسیاری از موارد حل مسئله‌ی بهینه‌سازی از طریق سعی و خطا ممکن نیست، یکی از دلایلی که می‌تواند باعث این امر شود بزرگ بودن فضای جستجوی مسئله است. یکی از مشکلات دیگر این است که نمی‌توان هر بار سیستم را اجرا کرده و نتیجه‌ی آن را مشاهده کرد. به عنوان مثال فرض کنید که برای قرار گرفتن یک ماهواره در مدار زمین، باید بهینه‌ترین سرعت و زاویه‌ی پرتاب و بهینه‌ترین مسیر حرکت تعیین شوند. نمی‌توان برای تعیین جواب مسئله هر بار ماهواره ارسال کرد.

روشی که می‌توان به جای سعی و خطا جایگزین حل مسائل بهینه سازی کرد این است که مسئله‌ی بهینه سازی را به صورت ریاضی مدل کنیم. این عمل، مهمترین اصل در حل مسائل بهینه سازی در حوزه‌ی مهندسی و علوم است. در مثال فوق، هدف اصلی این است که به بیشترین کیفیت تصویر دریافتی از تلویزیون دست پیدا کنیم.

تعیین متغیرهای بهینه سازی

پس از اینکه هدف موردنظر به صورت کیفی بیان شد، باید پارامترهایی که می‌توانند برای رسیدن به هدف مورداستفاده قرار گرفته و مفید واقع شوند را شناسایی کرد. در این مورد، متغیر قابل‌تغییر زاویه‌ی آنتن تلویزیون می‌باشد. همه‌ی پارامترهایی که در روند حل مسئله تأثیر دارند، در نظر گرفته و پس از فهرست کردن همه‌ی عامل‌ها، به گام بعدی می‌رویم. به عنوان مثال در کنار زاویه‌ی آنتن، ممکن است که ارتفاع آن نیز در کیفیت تصویر دریافتی تأثیر داشته باشد. موارد دیگری همچون طول سیم آنتن و کیفیت آن و موارد مشابه دیگر می‌توانند بسته به تعریف ما، به عنوان متغیرهای بهینه سازی در مسئله به کار گرفته شوند. می‌توان رابطه‌ی مفروض زیر را به عنوان تابع هدف مسئله در نظر گرفت:

$$Z = f(x_1, x_2)$$

---

[۱] Objective Function
[۲] Cost Function
[۳] Fitness Function

که در آن x_1 اندازه‌ی زاویه‌ی آنتن و x_2 اندازه‌ی ارتفاع آنتن است.

ارتباط میان هدف بهینه سازی و متغیرهای بهینه سازی به صورت ریاضی

در مرحله‌ی قبل مسئله‌ی بهینه سازی به صورت کیفی بیان شد. بخش اصلی این است که مسئله‌ی بهینه سازی به صورت ریاضی تعریف شود. در این مرحله باید ارتباط میان متغیرها و هدف بهینه سازی را بیان کرد. معمولاً این بخش نیاز به داشتن اطلاعات و دانش زیادی از مسئله‌ی بهینه سازی در حوزه‌ی تخصصی مربوط به مسئله دارد و هر فردی خارج از حوزه‌ی مرتبط با مسئله، نمی‌تواند به بیان تابع هزینه بپردازد. در مورد مثال ذکر شده، برای بیان ارتباط میان کیفیت تصویر دریافتی و زاویه و ارتفاع آنتن، نیاز به داشتن به اطلاعات تخصصی در حوزه‌ی میدان‌ها و امواج و نیز تصویر است. فرض می‌کنیم با کسب دانش لازم و تحقیق و بررسی به رابطه‌ی زیر برسیم:

$$Z = 100 - x_1^2 - (x_2 - 5)^2$$

در اینجا فرآیند تعریف مسئله‌ی بهینه سازی به پایان می‌رسد. سپس برای حل مسئله می‌توان از یک الگوریتم کمک گرفت. در جاهایی که مسئله فقط دارای یک نقطه‌ی حداقل (یا حداکثر) است و تابع هدف نیز نسبت به متغیرها مشتق‌پذیر است، بهترین پاسخ روش‌های مبتنی بر گرادیان (مشتق) می‌باشند و به سادگی بهترین جواب را می‌دهند. مثلاً جواب مسئله‌ی ذکر شده به صورت زیر به دست می‌آید:

$$x_1 = 0 \quad , \quad x_2 = 5$$

اما در مورد مثال زیر:

$$IQ = -50 - x_1^2 + x_2^2 + 10 \times (\cos(2\pi x_1) + (\cos(2\pi x_2))$$

برای حل چنین مسائل بهینه سازی که پیچیدگی بیشتری دارند و حل آن‌ها با استفاده از معادلات ریاضی پیچیده و زمان‌بر است، می‌توان از روش‌های دیگر بهینه سازی همچون بهینه سازی تکاملی استفاده کرد.

برای پیاده‌سازی عملی کار، می‌توان از کدهای مختلف مربوط به الگوریتم‌های بهینه سازی تکاملی که در مراجع گوناگون وجود دارد استفاده کرد. همه‌ی این کدها به این صورت عمل می‌کنند که تابع هزینه‌ی مسئله را گرفته و پس از جستجو با روش‌های مختلف، جواب مسئله را می‌دهند. در عمل، در موارد بسیاری، تابع هزینه به صورت یک رابطه‌ی ریاضی مشخص موجود نیست. بلکه برای یافتن مقدار آن، باید یک پروسه‌ی فیزیکی، شیمیایی و یا کنترلی را طی کرد تا رابطه‌ی مربوط به هزینه به دست آید.

نکته‌ی قابل توجه این است که اغلب تابع هزینه علاوه بر متغیرهای بهینه سازی پارامترهای دیگری دارد که از اهمیت بالایی برخوردارند. باید تمام این پارامترهای اضافی را داخل کد تابع هزینه قرار دهیم و تابع هزینه فقط شامل متغیرهای بهینه سازی باشد، به گونه‌ای که تابع‌مان، فقط متغیرهای بهینه سازی را به عنوان ورودی دریافت کند. به عنوان مثال در مورد مثال ذکر شده در بالا، جنس آنتن پارامتری است که مدل شده است، ولی جزو متغیرهای بهینه سازی نیست و نیز روی تابع هزینه تأثیر دارد.

۴.۱ چیدمان

مسائل چیدمان[۱][۱، ۲] ، بر روی اینکه چگونه اشیا در یک فضای محدود منطقی تحت قیود مشخص قرار بگیرند مطالعه می‌کند. (برای مثال، افزایش نسبت استفاده از فضا، بدون تداخل). مسائل چیدمان از نظر تئوری از نوع مسائل NP-Complete هستند که در سال‌های اخیر بیشتر و بیشتر موردتوجه قرار گرفته‌اند و در زمینه‌های کاربردی مختلف مثل طراحی چیدمان ماژول‌های فضاپیما، تجهیزات ماشینی و کارخانه‌ای، سیستم‌های حفاری دریایی، حمل و نقل، خودرو و ربات موردنیاز هستند. در مسائل پیچیده، برخی محدودیت‌های اضافی از جمله قیدهایی برای تعادل، اتصال و حالت‌های مجاور هم یا متوالی باید در نظر گرفته شود. با این نوع مسائل در بسیاری از رشته‌های مهندسی مواجه می‌شویم و از اهمیت زیادی برخوردار هستند. این مسائل معمولاً به طور مستقیم برخی شاخص‌های انجام طراحی، از جمله قابلیت اطمینان و اقتصاد را تحت‌تأثیر قرار می‌دهند. از آنجا که این نوع مسائل وابسته به NP-Complete می‌باشند، حل کردن آن‌ها به طور رضایت‌بخش کاملاً دشوار است.

مراجع مربوطه [۱، ۳، ۴]، به طور خلاصه به روش‌های رایج برای حل مسائل چیدمان پرداخته‌اند، از جمله برنامه‌نویسی ریاضی و روش‌های مقیاسی، الگوریتم‌های اکتشافی، نظریه‌ی گراف، سیستم‌های خبره و الگوریتم‌های مبتنی بر هوش جمعی و قوانین طبیعی. با توجه به روند الگوریتم و کیفیت راه‌حل، نشان داده شده است که الگوریتم‌های جامع قوی مبتنی بر هوش جمعی و قوانین طبیعی برتری دارند. به خصوص برای حل مسائل پیچیده طراحی در مقیاس بزرگ و یا متوسط، در مقایسه با دیگر روش‌های سنتی مناسب هستند [۵].

۵.۱ پیشینه‌ی تحقیق

در سال‌های اخیر الگوریتم بهینه سازی ازدحام ذرات به طور گسترده و به شکل‌های مختلف در زمینه‌های گوناگون مورد استفاده قرار گرفته است. در مقالات اغلب این الگوریتم به صورت ترکیبی با الگوریتم‌های دیگر از جمله الگوریتم ژنتیک مورد استفاده قرار می‌گیرد، دلیل این کار این است که الگوریتم‌های تکاملی مختلف می‌توانند نقص‌های یکدیگر را تا حدودی جبران کنند.

الگوریتم‌های مبتنی بر هوش مصنوعی برای حل مسائل چیدمان نسبتاً موثر هستند، اما دیده شده است که دو نقص اصلی وجود دارد: همگرایی زودرس و نرخ همگرایی آهسته. در [۶] برای غلبه بر آنها یک الگوریتم PSO-GA[۲] ترکیبی بهبودیافته‌ی جدید بر اساس الگوریتم ژنتیک موازی پیشنهاد شده است. در این الگوریتم مقداردهی اولیه‌ی بی‌نظمی، روش ترکیبی و تکامل چندزیرجمعیتی براساس جهش و تقاطع تطبیقی بهبودیافته به تصویب رسیده است. ترتیب قرارگیری مبتنی بر انتخاب با فشار در روش پیشنهادی، می‌تواند در مراحل اولیه از زودرس بودن الگوریتم جلوگیری کند و باعث همگرایی سریع‌تر در مراحل آخر شود. و مهمتر از آن، مطابق ویژگی‌های کلاس‌های مختلف زیرمجموعه‌های جمعیت، حالت‌های مختلفی از عامل‌های به روز رسانی الگوریتم بهینه سازی ازدحام ذرات [۳] معرفی

[۱] Layout Problems
[۲] Particle Swarm Optimization-Genetic Algorithm
[۳] Particle Swarm Optimization

شده‌اند. این روش‌ها باعث استفاده‌ی کامل از خاصیت همگرایی سریع بهینه‌سازی ازدحام ذرات شده است. در مقاله‌ی فوق با یک مثال نشان داده است که استفاده از HPSO-GA مؤثر و عملی است. اجرای الگوریتم فوق به این صورت است که حالت‌های مختلف عملگر به روز رسانی الگوریتم PSO، در زیرگروه‌های جمعیتی متفاوتی معرفی شده است. در فصل پنجم این پایان‌نامه مقایسه‌ای بین الگوریتم پیشنهادی ما و الگوریتم به کار رفته در مقاله‌ی فوق صورت می‌گیرد.

مقاله‌ی [۷] شامل یک الگوریتم جدید قسمت‌بندی برای مدارهای دوقسمت‌بندی[۱]، مورداستفاده برای کاهش تعداد ارتباطات بین عناصر مدارهای VLSI می‌باشد. در این مقاله یک الگوریتم ترکیبی PSO و SA[۲] برای مسئله‌ی دوقسمت‌بندی پیشنهاد شده است. برای حل چنین مسئله‌ای الگوریتم بهینه‌سازی ازدحام ذرات که جستجوی محلی (با تجربه شخصی) و جستجوی سراسری (با تجربه همسایگی) را ترکیب می‌کند، می‌تواند بهره‌وری جستجوی بالایی را در اختیار بگذارد. الگوریتم SA احتمال جلوگیری از به دام افتادن در بهینه‌ی محلی را بالا می‌برد و فرآیند جستجو می‌تواند با برنامه cooling schedule کنترل شود. نتایج تجربی نشان می‌دهد که الگوریتم PSO-SA ترکیبی توسعه‌یافته، می‌تواند به طور مداوم، مقادیر تناسب[۳] بهتری تولید کند و زمان موردنیاز کمتر از دیگر الگوریتم‌های بهینه سازی است.

مقاله‌ی [۸] یک روش کارآمد برای اندازه و بهینه‌سازی چیدمان سازه‌ی چوب‌بست (خرپا) ساختمان ارائه داده‌اند. برای انجام این‌کار، ترکیب یک الگوریتم PSO گسسته‌ی بهبودیافته[۴] و روش حرکت مجانب[۵] پیشنهاد شده است. در ترکیب IDPSO و MMA، مختصات‌های گرهی تعریف شده برای چیدمان سازه با MMA بهینه می‌شود و پس از آن نتایج MMA برای بهینه کردن نواحی (سطح) مقطع در IDPSO استفاده شده است. نتایج این تحقیق نشان می‌دهد ترکیبی از IDPSO و MMA می‌تواند به طور مؤثر به نرخ همگرایی سرعت بخشد و به سرعت به چیدمان بهینه برسد.

در مقاله‌ی [۹] نیز یک نسخه‌ی جدید بهبودیافته از PSO ارائه شده است که با قبلی متفاوت است. یک تغییر مهم با اضافه کردن توابع احتمالاتی در PSO رخ داده است و الگوریتم جدید PPSO نام دارد. از آنجا که تغییر سرعت ذرات در PSO درواقع موتور جستجوی آن را تشکیل می‌دهد، باید دو مرحله از فرایند بهینه سازی را ارائه دهد که عبارتند از: مرحله‌ی اکتشاف و مرحله‌ی بهره‌برداری. با این حال این هدف به دلیل نبودن فرمول سرعت ذرات متعادل در PSO، غیر قابل حصول است. ویژگی اصلی ارائه شده در مقاله‌ی فوق، معرفی یک طرح احتمالی برای به روز رسانی سرعت هر ذره است. بسط رابطه‌ی PPSO به ما اجازه می‌دهد بهترین دنباله از مراحل اکتشاف و بهره‌برداری، شامل فرایند جستجوی بهینه را پیدا کنیم.

اعتبار رویکرد فوق با حل سه مسئله‌ی بهینه‌سازی اندازه کلاسیک در سازه‌ی چوب‌بست فضایی و سه‌بعدی نشان داده شده است. به طور کلی پیکربندی فضایی، به پیدا کردن مکان‌ها و ابعاد ممکن برای مجموعه‌ای از اشیای مرتبط با هم تلقی می‌شود که همه نیازهای طراحی و همچنین به حداکثر رساندن کیفیت طراحی به لحاظ ترجیحات طراحی را مدنظر قرار می‌دهد.

مقاله‌ی [۱۰] که مربوط به بهینه سازی طراحی چیدمان در مبحث معماری است، به توصیف

---

[۱] bipartitioning
[۲] Simulated Annealing
[۳] Fitness
[۴] IDPSO
[۵] MMA

پیکربندی فضایی پرداخته است. پیکربندی فضایی به پیداکردن ابعاد و مکان‌های ممکن برای مجموعه‌ای از اشیای مرتبط باهم گفته می‌شود که همه‌ی نیازهای طراحی و به حداکثر رساندن کیفیت طراحی چیدمان به لحاظ ترجیحات طراحی را در نظر می‌گیرد. پیکربندی فضایی، مربوط به همه مسائل طراحی فیزیکی است. بنابراین حوزه‌ی قابل توجهی از پرسش‌ها را در بر می‌گیرد. تحقیق در زمینه‌ی اتوماسیون پیکربندی فضایی شامل بسته‌بندی قطعات [۱۱]، برنامه‌ریزی مسیر راه [۱۲]، چیدمان مراحل مختلف و امکانات، طراحی VLSI [۱۳]، و چیدمان مربوط به معماری [۱۴] می‌شود.

چیدمان مربوط به معماری منحصراً مبحث جالبی است، به این دلیل که علاوه بر اهداف معمول مهندسی، مثل هزینه و عملکرد، طراحی معماری به طور ویژه با خاصیت کاربردی بودن و زیبایی یک چیدمان نیز مرتبط می‌باشد که در نتیجه توصیف آن بصورت رسمی(یا صریح یا با رعایت قوانین) سخت‌تر است. همچنین در مورد چیدمان ساختمان (اتاق‌ها و دیوارها) اغلب ابعاد از پیش تعریف شده نیست، بنابراین هر قطعه از چیدمان قابل اندازه‌بندی مجدد است. گزارش شده است که تلاش برای خودکارسازی فرایند طراحی چیدمان بیش از ۵۰ سال پیش آغاز شده است. محققان از چندین تکنیک ارائه‌ی مسئله و جستجوی راه‌حل برای توصیف و حل مسئله استفاده نموده‌اند.

یک رویکرد برای تخصیص فضایی، تعریف فضای در دسترس به عنوان یک مجموعه از شبکه‌ی مربعی و استفاده از الگوریتم برای تخصیص هر مربع به یک اتاق یا فعالیت خاص می‌باشد. این مسئله ذاتا گسسته و چندوجهی است. به دلیل پیچیدگی ترکیبی، عموماً نمی‌تواند برای مسائل چیدمان با اندازه منطقی حل شود. چندین روش ابتکاری به صورت جامع برای پیدا کردن راه‌حل‌ها بدون جستجوی فضای طراحی ایجاد شده‌اند. لیژت و میچل[۱] [۱۵] از یک روش جایابی سازنده که دنبال‌کننده‌ی یک روش بهبود تکراری است استفاده کردند. در این روش، فضای اختصاص داده شده به اتاق‌ها بر مبنای بهترین حرکت (تغییر مکان) طراحی احتمالی در هر مرحله است. محققان دیگر نیز از الگوریتم‌های اتفاقی برای جستجو استفاده کرده‌اند[۱۶، ۱۷، ۱۸].

رویکرد دیگر در فضای طراحی چیدمان ساختمان، تجزیه‌ی مسئله به دو بخش است: توپولوژی(جانمایی و مکان شناسی) و هندسه. جانمایی اشاره به روابط منطقی بین اجزای طرح دارد. هندسه نیز اشاره به موقعیت و اندازه‌ی هر یک از مولفه‌ها در طرح دارد. تصمیمات توپولوژیکی محدودیت‌هایی برای فضای طراحی هندسی تعریف می‌کنند. طراح پس از آن قادر است احتمالات توپولوژیکی ممکن را بررسی کند و آن‌هایی را که می‌خواهد از نظر هندسی جستجو کند.

مقاله‌ی [۱۰] یک مدل ریاضی برای تصمیمات هندسی در مسئله‌ی چیدمان ارائه می‌دهد که راه‌حل کارآمدی با روش‌های ترکیبی محلی–سراسری و مبتنی بر گرادیان را ارائه می‌هد. فرایند بهینه سازی هندسی، راه‌حل‌های سریعی درمورد مسائل خیلی پیچیده ارائه می‌دهد که همچنین یک فرایند طراحی متقابل درست توصیف شده در پایان مقاله را در اختیار قرار می‌دهد.

در [۱۹] مدلی ارائه می‌کند که چیدمان معماری برای یک واحد آپارتمان دارای شکل منظم برای فضاهای مختلف از جمله اتاق خواب، حمام، آشپزخانه، بالکن، اتاق نشیمن، و اتاق ناهارخوری تولید می‌کند. با استفاده از محدودیت‌هایی در دو سطح توپولوژیکی (مجاورت، تراکم، و محدودیت‌های نمای باز و بسته) و ابعادی (محدودیت نسبت طول به عرض)، الگوریتم ژنتیک برای تولید آرایش (ترتیب)

[۱] Liggett and Mitchell

توپولوژیکی محیط‌ها در چیدمان و بیشتر اگر نیاز باشد، امکان‌پذیری از نظر ابعادی تحلیل و تجزیه شده است.

یکی دیگر از مسائلی که در زمینه‌ی بهینه سازی مورد بحث قرار گرفته است، مسئله‌ی برنامه‌ی زمانی کلاس‌ها برای مدارس و دانشگاه‌ها می‌باشد. مقالات متعددی در این زمینه وجود دارد و در آن‌ها انواع الگوریتم‌های بهینه سازی به کار گرفته شده است. [۲۰] با استفاده از الگوریتم PSO استاندارد و همچنین یک الگوریتم ازدحام ذرات ترکیبی، به توصیف این روش پرداخته است.

۶.۱ ساختار پایان نامه

در فصل اول، ابتدا مقدمه‌ای از موضوع موردنظر به همراه اهداف پایان‌نامه و چالش‌های پیش رو بیان شد. سپس به توصیف مفهوم کلی بهینه سازی با ذکر یک مثال پرداخته شد. پس از آن مفهوم چیدمان و جایابی مورد بحث قرار گرفت. سپس به بیان پیشینه‌ای از تحقیق مورد بررسی که توسط افراد دیگر و در سال‌های اخیر انجام شده است، پرداخته شد. در این مقالات اغلب از الگوریتم‌های ترکیبی و پیوندی استفاده شده است.

در فصل دوم، ابتدا به توصیف الگوریتم‌های تکاملی و سپس توضیح مختصری از هوش جمعی پرداخته می‌شود. سپس الگوریتم بهینه سازی ازدحام ذرات به طور مفصل شرح داده شده، و انواع آن از جمله انواع الگوریتم‌های بهینه سازی چندهدفه و الگوریتم PSO چندهدفه مورد بررسی قرار می‌گیرد و با توصیف معایب و مزایای استفاده از این الگوریتم، مقایسه‌ای بین آن و دیگر الگوریتم‌های تکاملی از جمله الگوریتم ژنتیک نیز صورت می‌گیرد.

فصل سوم به مدارهای چاپی اختصاص دارد. پس از بیان مقدمه‌ای درباره‌ی کاربرد این نوع مدارها، تاریخچه‌ای از ساخت آن‌ها آورده شده است. سپس به معرفی نرم‌افزار آلتیوم[۱] که از معروف‌ترین نرم‌افزارهای طراحی مدار چاپی است و قابلیت‌های آن پرداخته می‌شود. پس از آن درباره‌ی چگونگی طراحی، ساخت و مونتاژ قطعات روی بردهای مدار چاپی بحث شده، و سپس تعریف جایابی قطعات و نکاتی در مورد جایابی قطعات توضیح داده می‌شود.

در فصل چهارم نیز پس از مقدمه‌ای کوتاه در مورد کلیات مسئله، اهداف موردنظر تعیین می‌گردد و گزارشی از شرح کامل آزمایش‌های انجام گرفته در طول این پایان‌نامه داده شده، و الگوریتم پیشنهادی شامل پارامترها و متغیرهای مربوط به الگوریتم و توابع هدف در نظر گرفته شده، ارائه شده است.

در فصل پنجم، به منظور ارزیابی دقیق‌تر الگوریتم پیشنهاد شده و نتیجه‌گیری، مقایسه‌ای بین الگوریتم پیشنهادی این پایان‌نامه با یک الگوریتم ترکیبی بهینه سازی ازدحام ذرات و الگوریتم ژنتیک، صورت گرفته است.

[۱] Altium Designer

فصل ۲

الگوریتم‌های تکاملی و الگوریتم بهینه سازی ازدحام ذرات

۱.۲ مقدمه

در این فصل ابتدا به توصیف الگوریتم‌های تکاملی و مفهوم هوش جمعی پرداخته شده است. سپس الگوریتم بهینه سازی ازدحام ذرات که الگوریتم مورداستفاده در این پایان‌نامه است به طور کامل تشریح می‌گردد و مزیت‌ها و اشکالات آن بررسی می‌گردد. مباحثی همچون بهینه سازی چندهدفه و الگوریتم های MOPSO نیز به طور مفصل شرح داده شده است.

۲.۲ الگوریتم‌های تکاملی

الگوریتم‌های تکاملی[۲] نوعی جستجوی تصادفی بر مبنای جمعیت می‌باشند که اولین و مهمترین نقطه‌ی قوت آن‌ها این است که ذاتاً موازیند، و می‌توانند فضای جستجو را در جهات مختلف بررسی نمایند. بررسی موازی زیرفضاها باعث می‌شود که جستجوی فضا به نواحی که متوسط آماری تابع هدف در آن‌ها زیاد است و امکان وجود نقطه‌ی بهینه‌ی مطلق در آن‌ها بیشتر می‌باشد، سوق پیدا کند. چون در این روش‌ها بر خلاف روش‌های تک‌مسیری، فضای جواب به طور همه‌جانبه جستجو می‌شود، امکان کمتری برای گرفتار شدن در یک نقطه‌ی بهینه‌ی محلی وجود خواهد داشت. الگوریتم‌های تکاملی این توانایی

[۲] Evolutionary Algorithmes

را دارند که به طور مؤثر کل فضای جستجو را مورد بررسی قرار دهند. همچنین این الگوریتم‌ها به پیوستگی تابع و مشتق آن وابسه نیستند و در مسیر جستجو تنها به تعیین مقدار تابع هدف در هر نقطه نیاز است. الگوریتم‌های تکاملی تنها نیاز به اطلاعاتی در مورد کیفیت راه‌حل‌های ایجاد شده به وسیله‌ی هر مجموعه از متغیرها دارند، در صورتی که در روش‌های دقیق برای حل مسئله به شناخت کاملی از ساختمان مسئله و متغیرها نیاز است.

الگوریتم‌های تکاملی در شاخه‌ی هوش مصنوعی قرار می‌گیرند و شامل الگوریتم‌هایی جهت جستجو هستند که در آن‌ها عمل جستجو از چندین نقطه در فضای جواب آغاز می‌شود.

الگوریتم‌های تکاملی با دیگر روش‌های بهینه‌سازی و روش‌های قدیمی جستجوی راه‌حل بهینه، تفاوت اساسی دارند. از جمله این تفاوت‌ها عبارت است از:

- الگوریتم‌های تکاملی در یک زمان جمعیتی از نقاط را بررسی می‌نمایند و فقط یک نقطه را جستجو نمی‌کنند.
- در حل مسائل با الگوریتم‌های تکاملی تنها تعریف تابع هدف مربوطه در جهت‌های جستجو موردنیاز است و نیاز به اطلاعات در زمینه‌های دیگر مسئله ندارند. همچنین محدودیتی برای تعریف تابع هدف وجود ندارد.
- الگوریتم‌های تکاملی از قوانین در حال تغییر احتمالی استفاده می‌کنند و از موارد مشخصی بهره نمی‌برند.

۱.۲.۲ انواع الگوریتم‌های تکاملی

انواع الگوریتم‌های تکاملی به شرح زیر است:

- الگوریتم ژنتیک
- الگوریتم بهینه سازی ازدحام ذرات
- الگوریتم کلونی زنبور عسل
- روش بهینه‌سازی گروه مورچه‌ها
- راهبرد تکاملی
- الگوریتم رقابت استعماری

۲.۲.۲ روش کار الگوریتم‌های تکاملی

الگوریتم‌های تکاملی از مکانیزم‌های ساده برای حل مسئله استفاده می‌کنند و پس از تعداد مشخصی از تکرارها به بهترین راه‌حل ممکن دست پیدا می‌کنند. این الگوریتم‌ها معمولاً از یک جمعیت که شامل راه‌حل‌های تصادفی است شروع، و در طی هر مرحله تکرار سعی در بهتر کردن مجموعه راه‌حل‌ها دارند.

۳.۲.۲ حوزه‌های کاربردی

الگوریتم‌های تکاملی در حوزه‌های مختلف کاربرد دارند، از جمله:

- هوش مصنوعی
- دانش‌های کاربردی: برق، مکانیک، صنایع، شیمی، زیست‌شناسی و غیره
- سنتز و آزمون‌های سخت‌افزاری
- طراحی و بهینه‌سازی فیلترهای دیجیتال و آنالوگ
- استفاده در سیستم‌های چند پردازنده‌ای
- کنترل ربات‌ها
- جانمایی سلول‌های لاجیکی

۳.۲ هوش جمعی

هوش جمعی[۱] یا هوش گروهی، به نوعی رفتار دسته‌جمعی بین اعضای مختلف در سیستم‌های خودسازمانده از نوع طبیعی یا مصنوعی گفته می‌شود. این مفهوم در حوزه‌ی هوش مصنوعی بکار می‌رود و برای اولین بار توسط جراردو بنی و جینگ وانگ در سال ۱۹۸۹ و در رابطه با ربات‌ها مطرح شد [۲۱]. سیستم هوش جمعی متشکل از جمعیتی از عاملین ساده است که در مقیاس محلی با هم و با محیط اطراف تعامل می‌کنند. مکانیزم ربات‌ها اغلب از طبیعت، به خصوص سیستم‌های بیولوژیکی الهام می‌گیرند. عاملین این سیستم‌ها از قوانین ساده‌ای تبعیت می‌کنند و با وجودی که ساختار کنترلی متمرکزی وجود ندارد که رفتار آن‌ها را کنترل کند، ولی تعامل افراد منجر به ظهور نوعی رفتار مبتنی بر هوش می‌شود که در مقیاس انفرادی برای هر یک از آن‌ها ناشناخته است. نمونه‌ی آن هوش جمعی کلونی مورچه‌ها، ایجاد دسته در پرندگان، ایجاد رمه در حیوانات بزرگ، رشد باکتری‌ها، ایجاد مدرسه در ماهی و هوش میکروبی می‌باشد. یکی از معروف‌ترین الگوریتم‌هایی که از هوش جمعی بهره می‌برد، الگوریتم بهینه‌سازی ازدحام ذرات می‌باشد.

۴.۲ الگوریتم بهینه سازی ازدحام ذرات

الگوریتم بهینه سازی ازدحام ذرات[۲] اولین بار در سال ۱۹۹۵ توسط جیمز کندی[۳] و راسل ابرهارت[۴] به عنوان یک روش جستجو با پاسخ غیرقطعی برای بهینه سازی توابع مطرح شد[۲۲]. الگوریتم PSO یک الگوریتم جستجوی اجتماعی است و از روی رفتار اجتماعی دسته‌های پرندگان مدل شده است[۲۳] . در

[۱] Swarm intelligence
[۲] Particle Swarm Optimization
[۳] James Kennedy
[۴] Russell Eberhart

PSO، ذرات در فضای جستجو پخش می‌شوند. پس از مقداردهی اولیه به ذرات، تغییر مکان آن‌ها در فضای جستجو تحت تأثیر تجربه و دانش خودشان و همسایگانشان است. درنتیجه موقعیت ذرات دیگر روی چگونگی جستجوی یک ذره نیز تأثیرگذار است. با مدل‌سازی این رفتار اجتماعی، نتیجه‌ی حاصل روش جستجویی است که ذرات به سمت بهترین نواحی میل می‌کنند. ذرات از دانش یکدیگر استفاده می‌کنند و بر مبنای دانش بدست آمده به سمت بهترین همسایگان خود می‌روند. اساس کار PSO به این صورت است که در هر لحظه، هر ذره مکان خود را در فضای جستجو با توجه به بهترین مکانی که تاکنون در آن قرار گرفته است و بهترین مکانی که در کل همسایگی‌اش وجود دارد، تنظیم می‌کند.

به طور مفهومی، دسته‌ای از پرندگان و یا ماهی‌ها در یک محیط به طور تصادفی دنبال غذا می‌گردند، طوریکه تنها یک تکه غذا در محیط موردنظر موجود است. هیچ کدام از آن‌ها محل غذا را نمی‌داند. یک روش خوب می‌تواند پیروی از پرنده‌ای باشد که فاصله‌ی کمتری تا غذا دارد. این روش جانمایه‌ی الگوریتم است. هر راه‌حل که به آن یک ذره[۱] گفته می‌شود، در الگوریتم PSO معادل یک پرنده یا ماهی در حرکت جمعی پرندگان یا ماهی‌ها می‌باشد. هر ذره یک مقدار شایستگی[۲] دارد که توسط یک تابع شایستگی[۳] یا تابع هدف محاسبه می‌شود.

هر ذره یک سرعت معین دارد که هدایت حرکت ذره را بر عهده دارد. هر ذره‌ای با دنبال کردن ذرات بهینه در حالت کنونی، به حرکت خود در فضای مسئله ادامه می‌دهد. اساس کار PSO به این گونه است که مجموعه‌ای از ذرات به طور تصادفی به وجود می‌آیند و با به روز کردن نسل‌ها در هر مرحله به سمت راه‌حل بهینه حرکت می‌نمایند. در هر گام، هر ذره با استفاده از دو بهترین مقدار به روز می‌شود. اولین مورد، بهترین موقعیتی است که تاکنون ذره موفق شده است به آن برسد. این موقعیت با نام Pbest شناخته می‌شود. بهترین مقدار دیگری که توسط الگوریتم مورد استفاده قرار می‌گیرد، بهترین موقعیتی است که تاکنون توسط کل جمعیت ذرات بدست آمده است. این موقعیت با Gbest شناخته می‌شود.

بعد از پیدا کردن این دو بهترین مقادیر، سرعت و مکان هر ذره با استفاده از معادله‌ی (۱.۲) و (۲.۲) به روز می‌شود. به طور کلی اگر $x_i(t)$ نشان‌دهنده‌ی موقعیت ذره p_i در فضای جستجو در لحظه t باشد، موقعیت p_i با افزودن سرعت $v_i(t)$ به موقعیت فعلی به صورت زیر تغییر می‌نماید:

(۱.۲)
$$v_i(t) = v_i(t-1) + c_1 \times rand_1 \times (x_{pbest} - x_i(t-1)) + c_2 \times rand_2 \times (x_{gbest} - x_i(t-1))$$

$$x_i(t) = x_i(t_1) + v_i(t) \qquad (۲.۲)$$

که در آن $v_i(t)$ بردار سرعت در گام tام، c_1 و c_2 مقادیر ثابت مثبت (ضرایب شتاب) و $rand_1$ و $rand_2$ اعدادی تصادفی هستند که به صورت نرمال در بازه [۰،۱] تولید می‌شوند. پارامترهای x_{pbest} و x_{gbest} به ترتیب نشان‌دهنده‌ی موقعیت بهترین تجربه‌ی شخصی و جمعی می‌باشند.

برای سرعت ذره مقادیر حداقل و حداکثری در نظر می‌گیریم به طوریکه $vi \in [-v_i(max), +v_i(max)]$. اگر $v_i(max)$ خیلی بزرگ باشد ذرات ممکن است در راه‌حل‌های خوب قبلی پرش کنند، درحالی که اگر خیلی کوچک باشد، الگوریتم ممکن است در حالت مطلوب محلی گیر افتد.

---

[۱] Particle
[۲] Fitness
[۳] Fitness Function

به منظور ایجاد قابلیت بهتر جستجو، پارامتری به نام وزن اینرسی[1] به شکل معادله‌ی (۳.۲) و به صورت ضریبی در پارامتر سرعت به الگوریتم اضافه می‌گردد:

(۳.۲)

$$v_i(t) = w \times v_i(t-1) + c_1 \times rand_1 \times (x_{pbest} - x_i(t-1)) + c_2 \times rand_2 \times (x_{gbest} - x_i(t-1))$$

همانند سایر الگوریتم‌هایی که از هوش جمعی بهره می‌برد، الگوریتم PSO از مجموعه‌ای از پاسخ‌های ممکن استفاده می‌نماید که این پاسخ‌ها تا زمان پیدا شدن یک پاسخ بهینه و یا مهیا شدن شرایط پایان الگوریتم، به حرکت خود ادامه می‌دهند.

به منظور تنظیم پارامترهای شناختی و جمعی و هماهنگ‌سازی آن با مقدار وزن اینرسی روش‌های مختلفی ارائه شده است. از جمله:

$$\vec{v}_i(t) = \chi(\vec{v}_i(t-1) + \rho_1(\vec{x}_{pbest} - \vec{x}_i(t)) + \rho_2(\vec{x}_{pbest} - \vec{x}_i(t))) \quad (۴.۲)$$

$$\chi = \frac{2k}{| 2 - \rho - \sqrt{\rho(\rho-4)} |} \quad (۵.۲)$$

$$\rho = \rho_1 + \rho_2 = r_1 c_1 + r_2 c_2$$

که در روابط فوق می‌باید $\rho \geq 4$ بوده و $k \in [0,1]$ می‌باشد.

همچنین برای روش فوق موارد زیر نیز ارائه شده اند:

$$\chi = 1 - \frac{1}{\rho} + \frac{\sqrt{| \rho^2 - 4\rho |}}{2} \quad (۶.۲)$$

$$\rho = \rho_1 + \rho_2 > 4.0$$

همچنین رویکرد زیر نیز قابل ارائه می‌باشند:

$$c_1(t) = (c_{1,min} - c_{1,max})\frac{t}{T} + c_{1,max} \quad (۷.۲)$$

$$c_2(t) = (c_{2,min} - c_{2,max})\frac{t}{T} + c_{2,max}$$

که در روابط فوق داریم:

$$c_{1,max} = c_{2,max} = 2.5$$

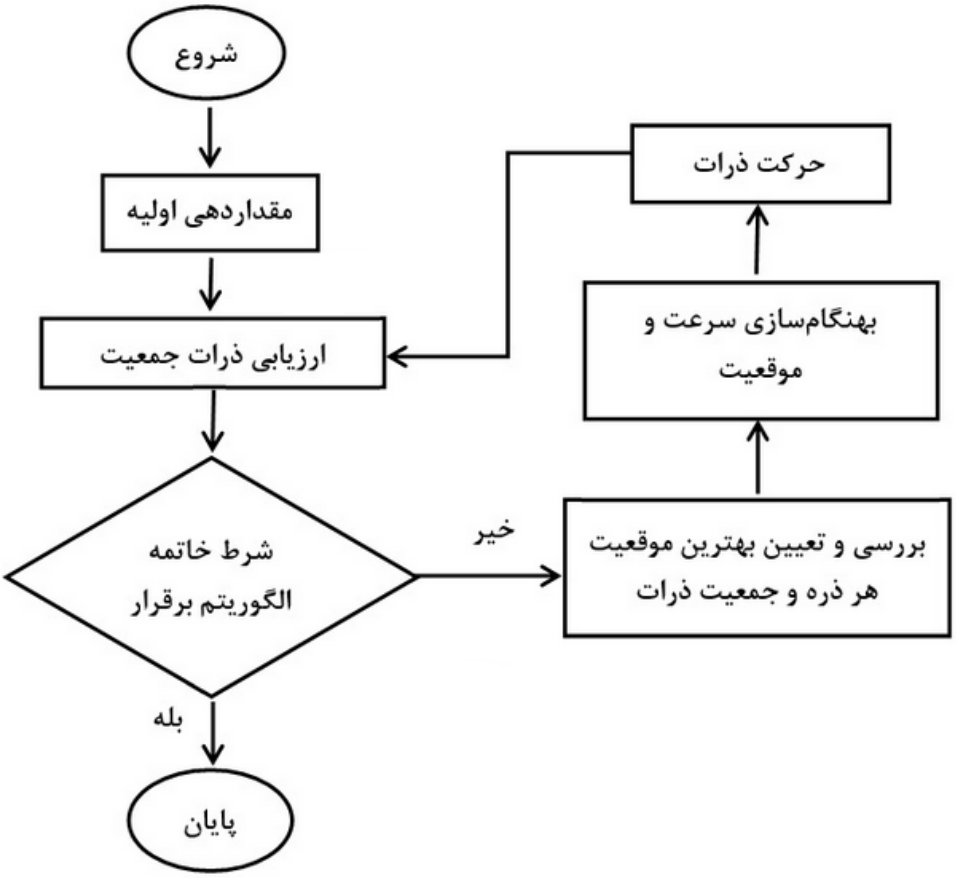
$$c_{1,\min} = c_{2,\min} = 0.5$$

در مورد حل مشکل مقادیر سرعت ایجاد شده در خارج از باند سرعت مجاز، به این ترتیب عمل می‌شود که هر کدام از اجزای بردار وزن که محدودیت حداکثر سرعت مجاز را رعایت نکنند، با جایگزینی از مقادیر باند تولید متغیر x اصلاح می‌شوند.

[1] Inertia Weight

۱.۴.۲ فلوچارت الگوریتم بهینه سازی ازدحام ذرات

در ابتدا، ذرات به صورت تصادفی در سرتاسر فضای جستجو مقداردهی می‌شوند که این موقعیت‌های اولیه به عنوان بهترین تجربه‌ی شخصی[۱] ذرات نیز شناخته می‌شوند. در گام بعد بهترین ذره از میان ذرات موجود انتخاب شده و به نام بهترین پاسخ[۲] شناخته می‌شود. سپس گروه ذرات در فضای جستجو حرکت می‌نمایند تا زمانی که شرایط پایان محقق شود. این حرکت شامل اعمال معادله‌ی سرعت به گروه ذرات می‌باشد که موقعیت هر ذره براساس آن تغییر می‌کند. فلوچارت الگوریتم بهینه سازی ازدحام ذرات در شکل (۱.۲) قابل‌مشاهده است.



شکل ۱.۲: فلوچارت الگوریتم بهینه سازی ازدحام ذرات

۲.۴.۲ شبه کد الگوریتم بهینه سازی ازدحام ذرات

```
For each particle
Initialize particle
End For
Do
For each particle
```

[۱] Personal Best
[۲] Global Best

```
Calculate fitness value of the particle fp
/*updating particle's best fitness value so far)*/
If fp is better than pBest
set current value as the new pBest
End For
/*updating population's best fitness value so far)*/
Set gBest to the best fitness value of all particles
For each particle
Calculate particle velocity according equation (1)
Update particle position according equation (2)
End For While maximum iterations OR
minimum error criteria is not attained
```

۵.۲ وزن اینرسی

روش‌های گوناگونی به منظور کنترل پارامتر وزن اینرسی معرفی شده است. از جمله روش‌های انتخاب تصادفی، کاهش خطی و نیز کنترل فازی به منظور انتخاب این مقدار معرفی شده‌اند. در الگوریتم پایه PSO از روش زیر به منظور تولید این وزن استفاده می‌شود که در آن t شماره گام فعلی و T تعداد کل گام‌های الگوریتم می‌باشد.

$$w = 1 - 0.5(\frac{1-t}{1-T}) \qquad (۸.۲)$$

به منظور تولید وزن‌های با کاهش خطی از روش زیر نیز می‌توان استفاده نمود که در آن از یک وزن اولیه‌ی بزرگ (حدودا ۰.۹) به یک وزن کوچک (حدودا ۰.۴) خواهیم رسید:

$$w(t) = (w(1) - w(T))\frac{T-t}{T} + w(T) \qquad (۹.۲)$$

که در آن T حداکثر تعداد دفعات اجرای الگوریتم است، $w(1)$وزن اینرسی اولیه، $w(T)$ وزن اینرسی نهایی و $w(t)$ وزن اینرسی در گام t است. قابل ذکر است که $w(1) > w(T)$ خواهد بود.

در روش کاهش غیرخطی نیز از یک وزن اولیه‌ی بزرگ به یک وزن کوچک خواهیم رسید. روش غیرخطی به زمان جستجوی کمتری نسبت به روش خطی منجر خواهد شد و زمان بیشتری را بر روی بازیابی پاسخ‌های مناسب خواهد گذاشت.

این روش برای فضای جستجوی نرم‌تر مناسب‌تر به نظر می‌رسد. رابطه‌ی آن با مقدار $w(1) = 0.9$ عبارت است از:

$$w(t+1) = \frac{(w(t) - 0.4)(T-t)}{T + 0.4} \qquad (۱۰.۲)$$

روش دیگر استفاده از رابطه‌ی زیر می‌باشد که در آن $\alpha = 0.975$ و t گام زمانی است که در آن مقدار وزن اینرسی عوض شده است:

$$w(t+1) = \alpha w(t\prime) \tag{۱۱.۲}$$

وزن اینرسی تنها زمانی عوض می‌شود که تغییر قابل توجهی در مقدار تطابق اجتماع ذرات ایجاد نشود. معمولاً ۲۰ درصد از ذرات جمعیت به صورت تصادفی انتخاب شده و مورد استفاده قرار می‌گیرد. مقدار بزرگ اولیه تضمین‌کننده‌ی جستجوی فضای مسئله است.

روش دیگر استفاده از تولید وزن‌های تصادفی است که به صورت زیر است:

$$w = w_0 + rand(1 - w_0) \tag{۱۲.۲}$$

به منظور تولید وزن‌های تصادفی از روش زیر می‌توان استفاده نمود:

$$w \sim N(0.72, \sigma)$$

در این روش در هر تکرار مقدار وزن اینرسی به صورت تصادفی از تابعی با توزیع گوسی انتخاب می‌شود. به این نکته باید توجه کرد که مقدار σ به نحوی انتخاب می‌شود که سبب تولید وزن‌های بزرگتر از ۱ نشود. یک رویکرد دیگر هم استفاده از رابطه‌ی زیر است:

$$w = c_1 r_1 + c_2 r_2 \tag{۱۳.۲}$$

۶.۲ توپولوژی‌ها

الگوریتم‌های بهینه سازی گروه ذرات به دو صورت مورد استفاده قرار می‌گیرند:

نسخه‌ی سراسری[۱]

نسخه‌ی محلی[۲]

نسخه‌ی سراسری سرراست ترین و متداول ترین نسخه است. در نسخه‌ی سراسری، سرعت هر ذره مطابق با بهترین کارایی خود و بهترین کارایی بدست آمده توسط تمامی ذرات تنظیم می‌شود.

در نسخه‌ی محلی، سرعت هر ذره مطابق با بهترین کارایی خود و بهترین کارایی بدست آمده توسط ذرات همسایه منطبق می‌شود. همسایه‌های هر ذره، عموماً نزدیکترین ذرات از نظر مکانی به آن ذره تعریف می‌شوند. نسخه‌ی محلی می‌تواند به صورت نسخه‌ی سراسری در نظر گرفته شود، به شرط آن که همسایه‌های هر ذره، تمامی ذرات در نظر گرفته شود.

نسخه‌ی سراسری همگرایی سریعی دارد اما امکان افتادن در مینیموم محلی وجود دارد. نسخه‌ی محلی، شانس بیشتری برای پیدا نمودن راه‌حل‌های بهتر اما به صورت کندتری دارد.

---

[۱] Global Version

[۲] Local Version

۷.۲ معرفی چند مدل مختلف الگوریتم بهینه سازی ازدحام ذرات

۱.۷.۲ الگوریتم بهینه سازی ازدحام ذرات پرورشی (breeding)

این الگوریتم، یک الگوریتم ازدحام ذرات ترکیبی است که قوانین به روزرسانی سرعت و موقعیت معمول را با ایده‌ی پرورش (و یا تولید مثل) و زیرگروه‌های جمعیتی[۱] ترکیب می‌کند. الگوریتم‌های PSO با استراتژی پرورشی، پتانسیل رسیدن به همگرایی سریع‌تر و پیدا کردن راه‌حل بهتر را دارند[۲۴].

ایده‌ی اصلی معرفی الگوریتم ازدحام ذرات پرورشی استفاده از عملگر جهش مطابق الگوریتم ژنتیک در رویه‌ی بهینه سازی می‌باشد. برای این منظور با فرض اینکه دو ذره‌ی a و b از جمعیت برای انجام عمل جهش انتخاب شوند به طریق زیر عمل می‌شود:

$$x_a(t+1) = p_i \times x_a(t) + (1-p_i) \times x_b(t) \qquad (۱۴.۲)$$

$$x_b(t+1) = p_i \times x_b(t) + (1-p_i) \times x_a(t)$$

$$v_a(t+1) = \frac{v_a(t)+v_b(t)}{\mid v_a(t)+v_b(t) \mid} \mid v_a(t) \mid \qquad (۱۵.۲)$$

$$v_b(t+1) = \frac{v_a(t)+v_b(t)}{\mid v_a(t)+v_b(t) \mid} \mid v_b(t) \mid$$

که در روابط فوق x_a و x_b در واقع موقعیت فرزندان برای هر بعد می‌باشد که از طریق تقاطع ریاضی در موقعیت هر کدام از والدین بدست می‌آید. p_i یک مقدار رندم توزیع یکنواخت بین ۰ و ۱ است. بردارهای سرعت فرزندان، به صورت مجموع بردارهای سرعت نرمالیزه شده به طول اولیه بردار سرعت هر والد محاسبه شده‌اند.

۲.۷.۲ الگوریتم ازدحام ذرات گسسته (دودویی) (Binary PSO)

بسیاری از مسائل بهینه سازی ماهیتی به شکل گسسته دارند، همچنین در بسیاری از کاربردها، حل مسائل با ماهیت پیوسته نیز در فضای گسسته انجام می‌شود. بنابراین نیاز به الگوریتم PSO باینری احساس می‌شود. مدل باینری الگوریتم بهینه سازی ازدحام ذرات در سال ۱۹۹۷ توسط مبدعان PSO، ارائه شد[۲۵]. در این مدل موقعیت هر ذره با دو مقدار صفر و یک در هر بُعد مشخص می‌شود و v_{id} احتمال یک بودن x_{id} را بیان می‌کند.

الگوریتم اجتماع ذرات باینری، با تعریف دیگری از مفهوم سرعت در آن، درواقع شاخه‌ای از الگوریتم فضای پیوسته است. در نسخه‌ی باینری، موقعیت هر ذره در هر بعد به دو مقدار صفر و یک محدود می‌شود. یعنی هر ذره، در یک فضا محدود به صفر و یک حرکت می‌کند. در این حالت فضای جستجو، مکعب بزرگی است که ذرات از یک رأس آن به رأس دیگر جابجا می‌شوند. در الگوریتم نسخه‌ی باینری، مفهوم سرعت به مفهوم احتمال، تغییر یافته و v_{id} احتمال یک بودن x_{id} را بیان می‌کند. بدین معنی

[۱] subpopulation

که مقدار v_{id} به یک مقدار بین $[0,1]$ نگاشت شده که این مقدار بیانگر احتمال یک بودن x_{id} است. برای پیاده‌سازی رویکرد باینری‌سازی الگوریتم، ابتدا سرعت ذره در هر بعد با استفاده از رابطه‌ی (۱.۲) محاسبه می‌شود. سپس، این مقدار با استفاده از تابع محدود کننده‌ی سیگموید (۱۶.۲) به مقداری بین صفر و یک نگاشت می‌شود و در نهایت موقعیت ذره iاُم در بعد dاُم با رابطه‌ی (۲.۲) به روز می‌شود.

$$S(v_{id}) = sigmoid(v_{id}) = \frac{1}{1+e^{-v_{id}}} \tag{۱۶.۲}$$

$$if \quad rand < S(v_{id(t+1)}) \quad then \quad x_{id}(t+1) = 1$$

$$else \quad x_{id}(t+1) = 0$$

که rand همانند قبل یک عدد تصادفی در بازه‌ی $[0,1]$ است. همانطور که مقدار v_{id} در الگوریتم پیوسته به یک بازه‌ی متقارن محدود می‌شود، در الگوریتم باینری نیز v_{id} به یک مقدار بیشینه v_{max} محدود می‌شود ($| v_{id} | < v_{max}$). بطور متداول مقدار v_{max} برابر ۶ در نظر گرفته می‌شود. اگرچه با این مقدار، خروجی تابع احتمال بکار رفته به بازه $[0/0025, 0/9975]$ محدود می‌شود، اما این محدودیت باعث همگرایی بهتر الگوریتم می‌شود. لازم به ذکر است که الگوریتم باینری نیز مشابه نوع حقیقی با استفاده از مدل‌های کلی و محلی قابل‌پیاده‌سازی است.

۳.۷.۲ الگوریتم AWPSO

الگوریتم AWPSO[۱] اولین بار توسط Mahfouf به منظور بهبود قابلیت الگوریتم PSO در بهینه سازی چندهدفه مطرح شد[۲۶]. در الگوریتم AWPSO معادله‌ی اصلاح سرعت به صورت زیر می‌باشد:

$$\vec{v}_i(t) = w\vec{v}_i(t-1) + c_1 r_1(\vec{P}_{bi} - \vec{x}_i(t-1)) + c_2 r_2(\vec{P}_g - \vec{x}_i(t-1)) \tag{۱۷.۲}$$

عبارت دوم در این رابطه در اصطلاح تحت عنوان پارامتر شتاب شناخته می‌شود که اندازه‌ی آن وابسته است به فاصله‌ی موقعیت ذره تا بهترین تجربه‌ی شخصی خود و نیز فاصله تا موقعیت بهترین تجربه‌ی عمومی. از همین رو رابطه‌ی فوق به صورت زیر اصلاح می‌شود:

$$\vec{v}_i(t) = w\vec{v}_i(t-1) + \alpha[r_1(\vec{P}_{bi} - \vec{x}_i(t-1)) + r_2(\vec{P}_g - \vec{x}_i(t-1))] \tag{۱۸.۲}$$

که در آن مقدار α فاکتور شتاب می‌باشد که از رابطه‌ی زیر محاسبه می‌شود:

$$\alpha = \alpha_0 + \frac{t}{T} \tag{۱۹.۲}$$

که در آن t شماره‌ی گام فعلی الگوریتم بوده و T بیشترین تعداد گام‌های اجرای الگوریتم می‌باشد و α_0 مقدار اولیه‌ی فاکتور شتاب می‌باشد که عددی حقیقی است که از بازه‌ی $[0.5, 1]$ انتخاب می‌شود.

همانطور که در رابطه‌ی فوق مشاهده می‌شود، فاکتور شتاب با افزایش شمار گام‌های الگوریتم زیاد شده است که همین امر سبب می‌شود که قابلیت جستجوی عمومی الگوریتم در گام‌های پایانی اجرا

[۱] Adaptive Weighted Particle Swarm Optimization

بیشتر شود. این مسئله به الگوریتم کمک می کند تا از افتادن در تله‌ی بهینه‌های محلی فرار کند، که این امر خصوصاً خود را در مسائل چندمدله نمایش می‌دهد.

نکته‌ی دیگر حذف پارامترهای شخصی و عمومی c_1 و c_2 می‌باشد. پارامترهایی که از آن‌ها به عنوان پارامتر اعتماد یاد می‌شود و نشان دهنده‌ی این است که رویه‌ی جستجو تا چه حد به تجربیات شخصی و یا جمعی اعتماد کند. با حذف این مقادیر، تعادل فوق به صورت تصادفی در طول اجرای الگوریتم تغییر می‌کند. به منظور انتخاب وزن اینرسی در این روش از رویکرد زیر استفاده می‌شود:

$$w = w_0 + r(1 - w_0) \qquad (۲۰.۲)$$

که در آن w_0 عددی در بازه‌ی $[0, 1]$ و r عددی تصادفی در بازه‌ی $[0, 1]$ می‌باشد که به صورت نرمال تولید می‌شود. بازه‌ی پیشنهادی برای مقدار w_0 عددی در بازه‌ی $[0, 0.4]$ می‌باشد، که سبب می‌شود وزن اینرسی به صورت تصادفی در بازه‌ی ۰.۴ تا ۱ تغییر کند.

۸.۲ بهینه سازی چندهدفه‌ی تکاملی

در این بخش رویکردهای مختلفی که برای حل یک مسئله‌ی بهینه سازی می‌توان از آن‌ها استفاده کرد بررسی می‌شود. مزیتی که مبحث بهینه سازی چندهدفه یا همان MO[۱] دارد این است که می‌شود برای آن یک سری بلوک‌های سازنده[۲] تعریف کرد که بعد از تلفیق آن‌ها می‌توان روش‌های جدید چندهدفه را ابداع کرد. اگر بخواهیم دقیق مشخص کنیم، اصولا ۳ نوع روش بهینه سازی تکاملی تا به حال معرفی شده است. چون بعضی روش‌ها آنقدر شبیه‌اند که فقط تفاوتشان در اپراتورهای جستجویی است که در این الگوریتم‌ها وجود دارد و عملاً چیزی که مربوط به بهینه سازی چندهدفه است، ۳ روش است. در ادامه به الگوریتم‌های شاخص هر کدام از روش‌ها اشاره خواهد شد.

با طرح مثال می‌توان به توصیف این مبحث پرداخت. فرضاً تابع هدف به صورت تابع هزینه باشد. درمورد خرید کالایی می‌خواهیم جوانب مختلف را در نظر بگیریم تا بهترین کالا را خریداری کنیم. می‌خواهیم min f(x) را پیدا کنیم، بطوریکه $x \in X$ (X فضای جستجو باشد). تفاوت چند هدفه با تک‌هدفه این است که اگر x عضو اسکالر باشد ($f : X \longrightarrow R$)، تابع تک‌هدفه است و اگر x عضو بردار باشد ($f : X \longrightarrow R^m$)، یعنی m تا مولفه داشته باشد، تابع چندهدفه می‌باشد.

بهینه سازی چند هدفه را به عنوان بهینه سازی برداری[۳] نیز می‌شناسند (تابع هدف به صورت بردار تعریف شده است) و مقالات بسیاری در این زمینه چاپ شده است. ویژگی مسائلی که بهینه سازی می‌شوند این است که در آن‌ها عواملی وجود دارد که بعضی را می‌خواهیم حداقل یا حداکثر کنیم. (البته با قرینه کردن یک تابع هدف، می‌شود مسئله را از حداقل‌سازی به حداکثرسازی تبدیل کرد.)

می‌توان ادعا کرد در مورد مسائلی که بطور روزمره با آن‌ها مواجه هستیم سه ویژگی مهم مطرح است:

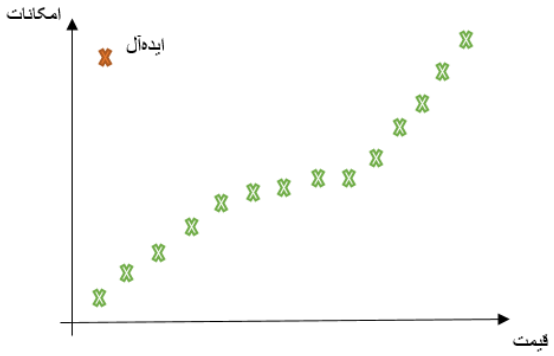
۱ــ مسائل بهینه سازی مقید کسر قابل توجهی از این نوع مسائل هستند.

[۱] Multi Objective
[۲] Building Block
[۳] Vector Optimization

۲ـ تعداد مسائل بهینه سازی استاتیک خیلی کم است. اغلب مسائل بهینه سازی دینامیک هستند که به اسم کنترل بهینه بحث می‌شود.

۳ـ مسائل بهینه سازی تک‌هدفه در اقلیت هستند. به دلیل اینکه مسائل بهینه سازی با کاربرد روزمره به خاطر در نظر گرفتن فاکتورهای موردنیاز مختلف و امکانات گوناگون، چندهدفه هستند. در نتیجه موازنه‌ای بین چند عامل مختلف داریم که با هم مصالحه دارند. یعنی نمی‌توان همه را با هم بهتر کرد و وقتی یک عامل را بهبود دهیم وضعیت عامل دیگر به هم می‌ریزد.

در مورد خرید یک وسیله دو عامل مهم وجود دارد. کمینه کردن هزینه و بیشینه کردن امکانات. فرض می‌کنیم نمودار امکانات بر حسب قیمتی که موجود است به شکل زیر باشد: نقطه‌ی ایده‌آل نیز



شکل ۲.۲: نمودار امکانات بر حسب قیمت

مشخص است. اگر یک کمپانی محصولی که تولید می‌کند زیر خط باشد، مشکل دارد و کمپانی مهندسی و معقول اقتصادی، کمپانی است که روی این سطح حرکت کند. به این سطح اصلاحاً جبهه[۱] گفته می‌شود. یک جبهه (به افتخار Pareto اقتصاددان ایتالیایی، آن را Pareto front می‌نامند)، مجموعه‌ای از پاسخ‌ها است که قطعاً پاسخی بهتر از آن‌ها وجود ندارد و نسبت به یکدیگر نیز هیچ ارجحیتی ندارند. اگر در یک مسئله‌ی بهینه سازی چندهدفه پاسخ‌های از این نوع را بررسی کنیم و به هم وصل کنیم، مجموعه ای از پاسخ‌ها به نام پاسخ‌های پرتو[۲] بدست می‌آید که به دنبال آن هستیم.

برخلاف بهینه سازی تک‌هدفه، در اینجا هدف پیدا کردن یک جواب نیست بلکه مجموعه‌ای از جواب‌ها وجود دارد. اما اینکه کدام پاسخ انتخاب شود، به میزانی که می‌توان هزینه کرد، باید نقطه‌ی کیفیت را پیدا و انتخاب کرد.

پس یکی از روش‌های حل مسائل بهینه سازی مقید تبدیل آن‌ها به MO می‌باشد. پس از بدست آوردن Pareto front، بهترین پاسخ موجه، جواب مسئله‌ی بهینه سازی مقید است. روش epsilon-constraint یکی از روش‌های حل مسائل MO است. این روش هوشمند نیست و اساسش این است که می‌توان از پیچیدگی فضای جستجو، یعنی از قیدها کم کرد و به پیچیدگی مسئله از نظر تعداد توابع هدف اضافه کرد.

---

[۱] front
[۲] Pareto Solutions

۱.۸.۲ تفاوت مسائل تک‌هدفه و چندهدفه

تفاوت اصلی بین مسائل تک‌هدفه و چندهدفه، اسکالر نبودن چندهدفه است. اشکالی که پیش می‌آید این است که فضای برداری و دوبعدی مرتب‌پذیر نیست و نمی‌توان برای آن مقدار بیشتر و کمتر تعریف کرد. چند رویکرد کلی برای حل مسائل MO وجود دارد. روش‌های حل این نوع مسائل به چند دسته تقسیم می‌شود:

۱ـ مسئله را همانطور که هست به صورت چندهدفه حل کنیم.

۲ـ مسئله را به یک مسئله‌ی بهینه سازی تک‌هدفه تبدیل کنیم. (تک‌هدفه‌سازی یا تجزیه[۱])

۲.۸.۲ روش‌های تجزیه

۱ـ روش مجموع وزن‌دار که این روش ضعف‌های زیادی دارد.

۲ـ روش epsilon-constraint

۳ـ روش‌های مبتنی بر الگوی چبی‌شف (فاصله از نقطه‌ی آرمانی)

در روش سوم، هرچه پاسخ‌ها را به پاسخ ایده‌آل نزدیک کنیم بهتر است. باید فاصله از نقطه‌ی آرمانی را کمینه کرد (فاصله یک کمیت اسکالر است، به همین دلیل است که می‌گوییم تک‌هدفه شده است). پس یک مسئله‌ی تک‌هدفه داریم که اگر آن را چندین بار حل کنیم راه‌حلی برای مسئله‌ی چندهدفه می‌باشد.

روش‌های مبتنی بر الگوی چبی‌شف روش‌های هوشمند و تکاملی هستند. مهمترین این روش‌ها عبارتند از:

۱ـ روش NSGA-II[۲۷].

۲ـ روش PESA2[۲]. الگوریتم MOPSO که آقای کوهلو (یکی از افراد بسیار شاخص در زمینه‌ی بهینه سازی چندهدفه) معرفی کردند، در واقع همین PESA می‌باشد با این تفاوت که PESA به صورت کلاسیک از اپراتورهای الگوریتم ژنتیک استفاده می‌کند و در MOPSO اپراتورهای PSO جایگزین شده‌اند.

۳ـ روش MOEA/D

۳.۸.۲ روش NSGA-II

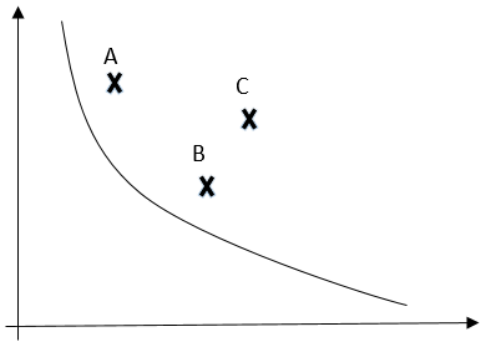
در NSGA با بخش NS[۳] کار داریم. این بخش NS را با هر الگوریتم بهینه سازی می‌توان ترکیب کرد و مسائل چندهدفه را حل کرد. کاری که این بخش انجام می‌دهد این است که فضای چندهدفه را که مرتب‌پذیر نیست، «با تعریف معیارهایی» به یک فضای ترتیب‌پذیر تبدیل می‌کند. در این روش دو معیار وجود دارد که به دنبال آن هستیم، معیار کیفیت و معیار نظم. فرض می‌کنیم دو هدف[۴] داریم که هر دو باید کمینه شوند. درنتیجه فضای هدف یک فضای دوبعدی است. بین نظم و کیفیت، کیفیت عامل

[۱] Decomposition
[۲] Pareto Englobe Sorting Algorithm
[۳] Nondominated Sorting
[۴] objective

مهمتری است. کار همه‌ی این الگوریتم‌ها این است که باعث تقویت پاسخ‌های بهتر شوند و بین دو پاسخ با امتیاز یکسان (کیفیت برابر) پاسخی که نظم بهتری دارد را انتخاب کنند.

۴.۸.۲ مفهوم غلبگی

اگر در فضای هدف دو ذره‌ی A و B نسبت به منحنی pareto موقعیت یکسانی از نظر فاصله به صورت شکل (۳.۲) داشته باشند، گفته می‌شود این دو ذره یکدیگر را مغلوب[۱] نمی‌کنند. رابطه‌ی (۲۱.۲) برای مفهوم غلبگی بین نقاط مختلف وجود دارد:



شکل ۳.۲: موقعیت سه ذره نسبت به منحنی پَرِتو

$$A \quad dom \quad B \quad \leftrightarrow \quad \forall j : f_j(A) \leq f_j(B), \quad \exists j_0 : f_{j0}(A) < f_{j0}(B) \qquad (۲۱.۲)$$

A بر B غلبه می‌کند اگر و فقط اگر به ازای همه‌ی نقاط j که $f_j(A) \leqslant f_j(B)$، یک j_0 وجود داشته باشد که $f_{j0}(A) < f_{j0}(B)$. مفهوم این عبارت این است که A بر B غلبه می‌کند اگر و فقط اگر اولاً A از هیچ نظر بدتر از B نباشد و دوماً حداقل از یک نظر A از B بهتر باشد.

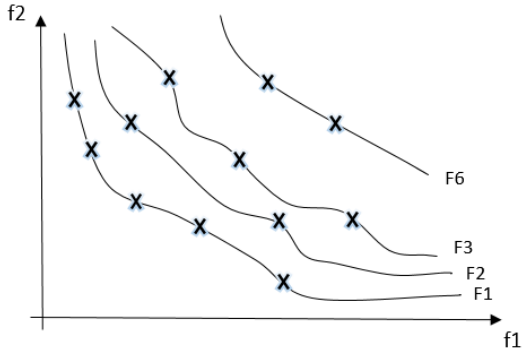
فاز اول فاز کیفیت نام دارد. فرض می‌کنیم این نقاطی که در شکل (۴.۲) می‌بینیم، جواب‌هایی هستند که جمعیت فعلی را در فضای هدف نشان می‌دهند (این جواب ها براساس تقاطع و جهش بدست آمده‌اند). می‌خواهیم این نقاط را رتبه‌بندی (ranking) کنیم. کاری که باید انجام شود این است که همه‌ی پاسخ‌ها دو به دو مقایسه شوند، پاسخ‌هایی که در کل فضای هدف، بهتر از آن‌ها وجود ندارد را پیدا کند و پاسخ‌هایی که مغلوب شوند، یک واحد به مقدار مغلوب‌شوندگی (یا یک واحد به دفعات مغلوب شدن) آن‌ها یعنی به n_q افزوده شود و آن‌ها نیز به مجموعه‌ی مغلوبین آن فرد یا پاسخی که غلبه دارد ینی S_p برده شوند. توصیف ریاضی این عمل در رابطه‌ی (۲۲.۲) مشاهده می‌شود. پاسخ‌هایی که درصد مغلوب شدنشان صفر است (به این پاسخ‌ها، پاسخ‌های مغلوب‌نشدنی[۲] گفته می‌شود)، تقریبی از Pareto front هستند.

$$p \quad dom \quad q \quad \rightarrow \quad n_q++ \qquad (۲۲.۲)$$

[۱] Dominate
[۲] Nondominated Solutions

$$q \longrightarrow S_p$$

در مرحله‌ی بعد پاسخ‌های درجه اول که اصلاً مغلوب نشده‌اند کنار گذاشته می‌شوند و گروه دیگری از



شکل ۴.۲: جمعیت فعلی در فضای هدف

پاسخ‌ها که با فرض نبودن پاسخ‌های درجه اول، مغلوب نمی‌شوند، با یک واحد کم کردن از درجه‌ای که آن افراد pareto front به این‌ها غلبه کرده بودند، بدست می‌آیند. نقاط و ذرات بدست‌آمده رتبه‌ی دوم نامغلوب [۱] را دارند. سپس این‌ها را حذف و رتبه‌ی سوم بدست می‌آید. این یک پروسه‌ی ساده‌ی iterative است. به این روش مرتب‌سازی نامغلوب و یا مغلوب‌نشدنی [۲] گفته می‌شود که مرتب‌سازی را انجام می‌دهد.

اگر تعداد پاسخ‌های موردنیاز به نحوی باشد که نیاز باشد تعدادی از پاسخ‌های درون یک جبهه را انتخاب کنند، باید معیاری برای انتخاب بهترین افراد آن جبهه وجود داشته باشد. درنتیجه در فاز دوم، عامل ثانویه‌ای به نام فاصله‌ی ازدحامی [۳] معرفی می‌شود که بین اعضای یک جبهه تعریف می‌شود و فاصله‌ی یک پاسخ از نزدیکترین پاسخ‌های هم‌سطح را نشان می‌دهد (بین همه‌ی f_1ها یک جور تعریف می‌شود). فاصله‌ی ازدحامی برای پاسخ‌هایی که در منطقه‌ی خلوتتری هستند بیشتر است. در واقع پاسخ‌هایی مهمترند که نقاط بکرتری را پیدا می‌کنند.

فرض می‌کنیم پاسخ‌هایی به صورت شکل (۵.۲) داریم. پاسخ i به همراه همسایه‌هایش کسری از این بازه را پوشش داده‌اند که در رابطه‌ی (۲۳.۲) به صورت نرمال شده این درصد را حساب می‌کنیم.

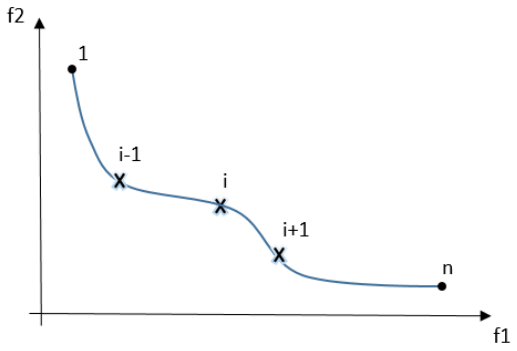
$$d_{ij} = \frac{| f_j^{i+1} - f_j^{i-1} |}{| f_j^1 - f_j^n |} \qquad (۲۳.۲)$$

$$d_i = d_{i1} - d_{i2} \qquad (۲۴.۲)$$

رابطه‌ی (۲۳.۲) فاصله‌ی ازدحامی مربوط به عضو یا پاسخ i در راستای بعد اول (در فضای تابع هدف) را نشان می‌دهد و رابطه‌ی (۲۴.۲) فاصله‌ی ازدحامی مربوط به عضو i را نشان می‌دهد. هر چه عدد بزرگتری برای d بدست آید نشان می‌دهد که i در ناحیه‌ی خلوتتری قرار دارد. (این یعنی متعادل کننده‌ی نظم)

در NSGA-II در هنگام مقایسه‌ی دو پاسخ، پاسخی بهتر است که

[۱] nondominated
[۲] Nondominated Sorting
[۳] Crawdind Distance



شکل ۵.۲: موقعیت ذره‌ی i و همسایگی‌های آن

۱ــ رتبه‌ی بهتری داشته باشد. (رتبه‌ی کمتر)
۲ــ فاصله‌ی ازدحامی آن بیشتر باشد.
اگر به این شکل مرتب کنیم، قطعاً هرگاه پاسخی را حذف کنیم، پاسخ بدتر را حذف کرده‌ایم.

۹.۲ الگوریتم MOPSO

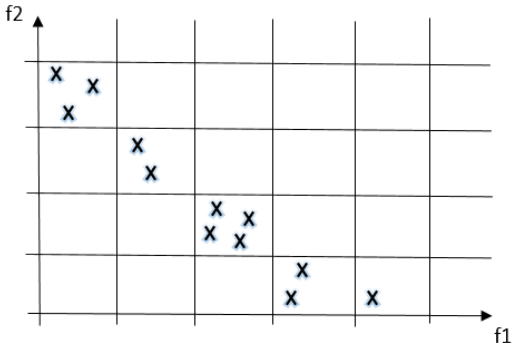
الگوریتم MOPSO شکل‌های مختلفی دارد. در اینجا به شکلی که در [۲۸] آورده شده است پرداخته می‌شود. البته اپراتورهایی که اشاره می‌کنیم مربوط به الگوریتم PESA-II است. تنها تفاوتش در نحوه‌ی انتخاب Global Best (در اینجا به اسم leader) می‌باشد.

در مسائل MO بهترین مکان (یا همان Global Best که در الگوریتم PSO به شکل g(t) در رابطه سرعت وجود دارد)، قابل تعریف نیست. چون فضا ترتیب پذیر نیست بلکه پاسخ‌های پرتو[۱] داریم. در نتیجه برای اینکه بتوانیم به روز رسانی را انجام دهیم، یک ایده این است که به صورت تصادفی یکی از پاسخ‌هایی که کسی نمی‌تواند آن را مغلوب کند انتخاب کنیم. در واقع این یعنی به صورت تصادفی یکی از ذرات جمعیت عضو f_1 را به عنوان لیدر برداریم. هر عضوی لیدر خودش را دارد. نکته‌ی دیگر این است که احتمال انتخاب شدن برای همه‌ی اعضا یکسان نیست. در اینجا به جای g(t) عضوی داریم به نام مخزن که $rep_{h,i}$ مقدار آن برای عضو iام می‌باشد. مخزنی برای پاسخ‌هایی که کسی آن‌ها را مغلوب نکرده است.(یعنی f_1)

تفاوت این الگوریتم با NSGA-II این است که NSGA-II از همان جمعیت به عنوان آرشیو خود استفاده می‌کند اما این الگوریتم مانند الگوریتم‌های چندهدفه‌ی دیگر یک مخزن جداگانه دارد که به عنوان آرشیوی از راه‌حل‌ها می‌باشد و یکی از اعضای مخزن به عنوان لیدر برگزیده می‌شود. برای انتخاب لیدر، در ابتدا باید فضای هدف را به شکل زیر درجه‌بندی کنیم. چون می‌خواهیم یکی از پاسخ‌های نامغلوب را به عنوان لیدر انتخاب کنیم، در اینجا به جای انتخاب مبتنی بر فردی، انتخاب مبتنی بر ناحیه را انجام می‌دهیم. طبق رابطه‌ی تناسب زیر، خانه‌ای در اولویت انتخاب است که جمعیت کمتری دارد.

$$p_i \propto \frac{1}{n_i}$$

[۱] pareto



شکل ۶.۲: درجه‌بندی فضای هدف و انتخاب بهترین گرید

که در آن p_i احتمال انتخاب شدن خانه‌ی iام و n_i کل جمعیت می‌باشد. این تناسب باید ساده گردد. به این ترتیب یک خانه انتخاب می‌شود و در نتیجه x_{t+1} پیدا می‌شود. پس از این مرحله، به روزرسانی انجام می‌شود.

$$p(t+1) = \begin{cases} p(t) & p(t) dom x(t+1) \\ x(t+1) & x(t+1) dom p(t) \\ x(t+1) \text{ و } p(t) \text{ انتخاب تصادفی بین} & \text{otherwise} \end{cases} \quad (۲۵.۲)$$

۱.۹.۲ به روز رسانی مخزن

در لحظه t یک مخزن (rep) داریم. یک مجموعه‌ی جمعیت جدید (pop) هم به دست می‌آوریم. بخش نامغلوب pop را با rep تلفیق می‌کنیم. درنتیجه یک مخزن جدید در لحظه $t+1$ به دست می‌آید ($rep(t+1)$). بعد از این، اعضایی که مغلوب می‌شوند را حذف می‌کنیم. rep به تعداد محدود و مشخصی می‌تواند عضو داشته باشد، اگر اعضای اضافه در rep وجود داشت، باید بعضی پاسخ‌ها را حذف کنیم. برای حذف پاسخ‌های اضافه، اول یک گرید انتخاب می‌کنیم (ناحیه ای که اعضای بیشتری دارد، با احتمال $q_i \propto n_i^{\beta}$). سپس این گرید را حذف، و یک مخزن جدید داریم. به این ترتیب پاسخ‌های اضافه حذف می‌شوند. پس با مکانیزم ساده‌ی الگوریتم PSO و تلفیق آن با مکانیزم گرید، الگوریتم MOPSO را می‌توان حل کرد.

۱۰.۲ جمع‌بندی

در این فصل به الگوریتم‌های فراابتکاری و انواع آن‌ها پرداخته شد و الگوریتم بهینه‌سازی ازدحام ذرات که در این پایان‌نامه استفاده شده است به همراه مزایا و ویژگی‌های آن مفصلاً شرح داده شده است. پس

از فصل سوم که به مدارهای چاپی اختصاص دارد، در فصل چهار الگوریتم پیشنهادی مورد بحث قرار می‌گیرد.

فصل ۳

آشنایی با مدارهای چاپی و نرم افزار طراحی آن

۱.۳ مقدمه

مدار چاپی به منظور سرعت در ساخت بردهای الکترونیکی ساخته شده است. یک مدار چاپی از یک برد نارسانا که به وسیله‌ی مس پوشانده شده است، تشکیل شده است که این مس‌ها در واقع سیم‌هایی است که ارتباط قطعات الکترونیکی مدار را به هم ممکن می‌سازد. مدار چاپی دارای مزایای زیر است: ۱ـ افزایش سرعت تولید ۲ـ حداقل شدن احتمال خطای ساخت ۳ـ مدار کمتر نویز می‌گیرد و بهتر عمل می‌کند. ۴ـ کم شدن حجم مدار ۵ـ کمک به زیبایی مدار

برد مدار چاپی مجموعه‌ای از مدارهای الکتریکی است و می‌تواند یک طرفه (یک لایه مس)، دو طرفه (دو لایه مس) و یا چندلایه باشد، به گونه‌ای که قطعات الکترونیکی مانند قطعات پسیو، مدارهای مجتمع و دیگر قطعات بر روی آن مونتاژ شده و جهت استفاده در تجهیزات الکترونیکی به کار می‌رود. ماده‌ی خام تشکیل دهنده‌ی این بردها از مواد مختلفی مانند فایبر، راجرز، تفلون، فلکسی‌بل و غیره ساخته شده است و با ضخامت‌های ۰.۲ تا ۳.۲ میلی‌متر عرضه می‌گردند. استاندارد جهانی تولید بردهای مدار چاپی بر اساس استاندارد UL و IPC بوده و جهت طراحی این بردها عموماً از نرم‌افزار Protel استفاده می‌گردد.

۲.۳ تاریخچه‌ی پیدایش و ساخت مدار چاپی

امروزه بردهای مدار چاپی[۱] آنقدر به طور گسترده بکار گرفته می‌شوند که هرگونه ساخت و مونتاژ تجهیزات برقی و الکترونیکی بدون آن‌ها تقریباً غیرممکن و غیرقابل‌تصور است، اما تاریخ توسعه و تکامل بردهای مدار چاپی خیلی کهن نیست و استفاده‌ی تجاری از آن‌ها به دهه ۱۹۵۰ میلادی باز می‌گردد، اگرچه اندیشه‌ی آن از حدود پنجاه سال پیش‌تر شکل گرفته بود. پیش از آنکه این اندیشه شکل بگیرد، لازم بوده است که برخی پیشرفت‌ها و فن‌آوری‌ها که می‌توانند زمینه‌ساز آن باشند تکامل یافته و به دنیای دانش و تکنولوژی شناسانده شده باشند. بردهای مدار چاپی در واقع تکامل‌یافته‌ی سیستم‌های اتصال الکتریکی است که در سال ۱۸۵۰ توسعه یافته بود.

در آن سال‌ها برای ساخت تجهیزات برقی بدون استفاده از بردهای مدار چاپی روش اتصال مستقیم قطعات به یکدیگر را به کار می‌گرفتند. در این روش همواره معایبی وجود داشت، از جمله اینکه هر سیستم مونتاژ شده با دیگر سیستم‌ها متفاوت بود (نبود تکرارپذیری در تولید)، کار مونتاژ نیازمند حضور افراد توانا و باتجربه بود، تعمیر مشکل بود و برای پیداکردن قطعه‌ی معیوب، سیم‌ها و اتصالات نیز در منطقه‌ی معیوب باید به دقت و همزمان کنترل می‌شد، نقشه‌ای معتبر که محل قطعات را نشان دهد وجود نداشت و اشکالات عمده دیگر.

اولین قدم برای حل این قبیل مسائل انتقال قطعات مونتاژ شده به درون یک چارچوب و نمایش ترسیمی مکان دقیق هر قطعه بود. برای سال‌های متمادی و حتی در طول سال‌های جنگ جهانی دوم و پس از آن روشی که تکنیک اتصال نقطه به نقطه خوانده می‌شد به طور گسترده‌ای به خدمت گرفته شد که دارای برخی مزایای اساسی بردهای مدار چاپی بود.

اما اختراع آلبرت هانسون[۲] در سال ۱۹۰۳ به وضوح توصیفی از مدارهای دوطرفه‌ی متالیزه بود. این مخترع تشخیص داده بود که تراکم مداری خیلی مهم است و از این رو مدارهای خود را با هادی‌هایی در دو طرف طراحی کرده بود. او همچنین می‌دانست که ارتباط بین لایه‌ای چقدر می‌تواند حیاتی باشد و لذا سوراخ‌های دستیابی بین لایه‌ای و عبور از سطح بالا به پائین هادی‌ها به صورت انتخابی را اضافه کرده بود. آلبرت هانسون همچنین بیان داشت که هادی‌ها می‌توانند در محل خود از طریق آبکاری الکتریکی و یا بکارگیری پودر فلزی در یک حامل مناسب (مرکب‌هادی) تشکیل شوند. این که در نخستین ثبت اختراع مدار چاپی چنین نظریه‌هایی که می‌تواند مدرن در نظر گرفته شود آمده است، جالب توجه می‌باشد.

نظریه‌های متعددی در طول دهه‌های بعدی همچنان که دانش الکترونیک گسترش می‌یافت با سرعت هیجان برانگیزی پدیدار شد. رادیو خیلی زود به مهم‌ترین پیش‌برنده‌ی مدار چاپی بدل گشت. بی‌سیم توجه جهانی را به خود جلب کرده بود. پیشتازان الکترونیک برای تولید انبوه مدارهای چاپی بازار بزرگی را می‌توانستند پیش رو ببینند و مخترعین با قدرت برای پاسخ‌گویی به این نیاز فعال بودند.

در سال ۱۹۱۳، آرتور بِری[۳] اختراعی به ثبت رساند و طی آن مدعی روش ساخت مدار با تکنیک فلزبری بود. بِری به عنوان نخستین کسی که مدارهای فلزبری شده را توصیف می‌کند ظاهر گردید.

---

[۱] Printed Circuit Boards
[۲] Albert Hanson
[۳] Arthur Berry

چارلز دوکاس[۱] در سال ۱۹۵۲ روشی را برای ایجاد طرح‌های هادی آبکاری شده بر روی یک بستر غیرهادی پیشنهاد کرد و اختراع خود را در آمریکا به ثبت رساند [۲۹]. وی برای ایجاد این طرح هادی از استنسیل و چاپ با یک مرکب هادی استفاده کرده بود و پس از برداشتن استنسیل، خطوط را تا ضخامت مورد نظر با آبکاری الکتریکی تقویت می‌کرد. این طرح‌ها به الگوهای مداری امروزی شباهت قابل‌ملاحظه‌ای داشته‌اند. با این روش نام برد مدار چاپی[۲] متولد شد. این روش به طرز چشمگیری ساخت وسایل برقی را آسان می‌نمود. فرایند ساده‌ای که توسط کارگر غیرماهر نیز انجام‌شدنی بود.

سهم عمده برای توسعه‌ی فناوری نوین ساخت بردهای مدار چاپی توسط پُل اِسلر[۳] (۱۹۰۷ – ۱۹۹۵) ادا گردید. امروزه او را به عنوان پدر مدار چاپی می‌شناسند. وی یک اتریشی ساکن انگلستان بود و آنچه را که احتمالاً قدیمی‌ترین برد مدار چاپی است، به صورتی که ما می‌شناسیم به عنوان بخشی از یک رادیو ساخت. در شکل (۱.۳) تصویری از نخستین برد مدار چاپی ساخته شده توسط این فرد نشان داده شده است. این رادیو اولین وسیله‌ای است که با استفاده از یک برد مدار چاپی کار کرده است و فناوری آن توسط اِسلر اختراع شده است. در روش وی پوششی مقاوم به شکل طرح مداری بر روی سطح مسی چاپ می‌شد و بخش‌های نپوشیده با مس‌بری حذف می‌گردید. در سال ۱۹۴۷ معلوم شد که تعداد زیادی

شکل ۱.۳: نخستین برد مدار چاپی

از بردهای مدار چاپی مورد استفاده صنایع نظامی در ساخت ساز و برگ ارتش در ایالات متحده‌ی آمریکا ساخته می‌شده است. درواقع جنگ جهانی دوم و نیازهای ارتش آمریکا به تجهیزات نوین از سویی این فناوری جدید را به پیش رانده و از سوی دیگر مانع از تجاری‌سازی آن شده بود. علاقه‌مندی به ساخت و استفاده از بردهای مدار چاپی پس از جنگ به سرعت رشد کرد و فرایندهای متعددی در اوایل سال ۱۹۵۰ به صنعت الکترونیک معرفی گردید. به ویژه اینکه در همین زمان ترانزیستور به بازار عرضه شد و کوچک بودن آن گرایش به بردهای مدار چاپی را تشدید نمود. درمقایسه با لوله‌های خلاء که بسیار بزرگ بودند و با نصب نقطه‌به‌نقطه و روش‌های مرسوم مونتاژ همخوانی داشتند. این یعنی توسعه‌ی ترانزیستور بهره‌برداری از مدارهای چاپی را با حذف یکی از پرحجم‌ترین قطعات، لوله‌های خلاء، آسان‌سازی کرد. توسعه‌ی بیشتر منجر به تولید مدارهای مجتمع در دهه ۱۹۶۰ گردید.

در روزهای ابتدایی ساخت صنعتی رادیو و تا پایان جنگ جهانی دوم گیرنده‌های رادیویی شامل قطعاتی مانند مقاومت‌ها، خازن‌ها، سیم‌پیچ‌ها و لوله‌های الکترونیکی بودند که بوسیله سیم‌هایی که با عایق‌های رنگی مشخص می‌گردید، به یکدیگر متصل می‌شدند. هر رنگ به مثابه یک کد، مشخص

[۱] Charles Ducas
[۲] Printed Circuit Board
[۳] Dr. Pual Eisler

کننده‌ی اتصال مداری ویژه‌ای بود که مورد قبول جهانی نیز قرار گرفته بود. حتی به منظور تسهیل تولید و ردیابی خطاها، سیم‌ها را به اندازه‌های یکسان بریده و به هم می‌بستند تا به شکل یک تسمه در می‌آمد و در سرعت بخشیدن به عملیات مونتاژ مؤثر بود.

در طول دهه‌ی ۱۹۵۰ و اوایل دهه‌ی ۱۹۶۰ میلادی، صفحات مدار چاپی از انواع رزین‌ها در مخلوط با انواع متفاوتی از مواد ساخته می‌شد، اما بردهای مدار چاپی همچنان یک طرفه مانده بودند. توسعه‌ی صنعت الکترونیک پس از جنگ جهانی دوم و تقاضای بالا برای محصولات مصرفی مانند رادیو و تلویزیون و به طور همزمان استفاده از صنعت الکترونیک در کاربردهای نظامی، موجب سخت‌تر شدن مقررات و الزاماتی برای قابلیت اطمینان بیشتر بردهای مدار چاپی و نیز افزایش تراکم مداری و پیچیدگی آن‌ها شد و با رسیدن به بالاترین تراکم مداری نیاز به جایگزینی با بردهای دو رو بیش از پیش احساس گردید. آبکاری به عنوان روشی برای ایجاد اتصالات میانی دو طرف برد مدار چاپی در طول این دهه به آهستگی رشد کرد.

متالیزه کردن سطوح نارسانا از مدت‌ها پیش شناخته شده بود. اما این روش‌ها به دلیل ناپایداری محلول‌ها و تجزیه آن‌ها در کمتر از چند ساعت و نیز قیمت بالا به ندرت برای تولید انبوه توصیه می‌شدند. در فاصله‌ی سال‌های ۱۹۵۵ – ۱۹۵۳ شرکت موتورولا، فرآیند آبکاری مس را برای ایجاد اتصال میان دو طرف برد مدار چاپی معرفی کرد که برای تولید انبوه مناسب‌تر بود.

۳.۳ معرفی نرم‌افزار آلتیوم دیزاینر و قابلیت‌های آن

آلتیوم یک نرم‌افزار طراحی برد مدار چاپی است. یک بسته‌ی نرم‌افزاری طراحی خودکار الکترونیکی برای طراحی PCB، FPGA و نیز نرم‌افزار طراحی جایابی، و کتابخانه‌ی مربوطه و انجام مدیریت خودکار. این بسته‌ی کاربردی به‌وسیله‌ی شرکت آلتیوم در استرالیا توسعه یافته و عرضه شده است. شرکت آلتیوم در سال ۱۹۸۵ توسط فردی به نام نیک مارتین تأسیس شد. یکی از نرم‌افزارهای مشهور این شرکت در آن زمان پروتِل[۱] بود. این نرم افزار در سال ۲۰۰۱ به آلتیوم تغییر یافت. آلتیوم دیزاینر یکی از بهترین نرم‌افزارهای طراحی مدارهای چاپی است. این نرم‌افزار توانایی طراحی چندلایه‌ای PCB در محیط دوبعدی و همچنین سه‌بعدی را دارد. در آلتیوم می‌توان یک شماتیک طراحی کرده و به راحتی آن را به PCB تبدیل کرد.

۴.۳ طراحی، ساخت و مونتاژ برد مداری

برای طراحی و ساخت یک برد مداری که بتواند نیازهای موردنظر یک پروژه را برآورده کند، چگونگی طراحی موضوع مهمی است. به این دلیل که چگونگی طراحی برد از جمله عواملی است که بر هزینه‌ی تولید برد تأثیر می‌گذارد.

---

[۱] Protel

۱.۴.۳ مونتاژ برد مدار چاپی

برای تولید مدار چاپی، مجموعه‌ای از اسناد مربوط به طراحی موردنیاز است[۳۰]. مدار الکترونیکی به عنوان یک طرح گرفته شده، ساخته شده از کتابخانه‌هایی شامل نمادهای قطعات که در کنار هم قرار داده شده و سیم‌کشی شده‌اند. پس از طراحی شماتیک مدار، طرح به ویرایشگر PCB منتقل می‌شود، جایی که از هر قطعه به صورت یک الگو و یا به اصطلاح رد پا نمونه‌گیری شده و مدار با خطوط ارتباطی نقطه به نقطه سیم‌کشی می‌شود. برای PCB نهایی شکلی همراه با لایه‌های فیزیکی که برد را تشکیل می‌دهد، تعریف می‌شود.

قوانین طراحی، الزامات طرح مثل عرض مسیر و فضاهای خالی بین قطعات و سیم‌ها را مشخص می‌کند. اجزا و قطعات درون فضای برد قرار می‌گیرند و خطوط ارتباطی بوسیله‌ی مسیریابی بصورت دستی و خودکار جایگزین می‌شوند. وقتی طراحی کامل شود، فایل‌های خروجی با فرمت استاندارد تولید می‌شود که می‌تواند برای ساخت برد، پیکربندی دستگاه مونتاژ و غیره استفاده شود.

فرایند مونتاژ برد مدار چاپی می‌تواند از فناوری جای‌گذاری سطحی[۱] یا فناوری درون حفره یا از هر دوی آن‌ها استفاده کند. وقتی قطعات روی برد مداری جای‌گذاری شدند، برد برای تست و در نهایت برای مونتاژ با محصول آماده شده است. ولی تضمین نمی‌شود که مونتاژ برد ۱۰۰ درصد بی‌نقص باشد. قطعاً مشکلاتی وجود خواهد داشت و باید این مشکلات برطرف شوند.

۵.۳ جایابی قطعات

کار جایابی قرار دادن همه‌ی قطعات روی برد با در نظر گرفتن محدودیت‌های فرایند تولید است، به طوری که عملی و منطقی نیز باشد[۳۱]. بدیهی است جایابی تأثیر به سزایی در مسیریابی بین قطعات دارد. از نکات قابل‌توجه در جایابی این است که قطعات مجاز به هم‌پوشانی نیستند و باید به برخی حداقل فاصله‌ها پایبند باشند. هدف اصلی در جایابی اجازه‌ی مسیریابی خوب و عملی است. در اصل بهتر است که جایابی و مسیریابی در یک مرحله انجام شود اما با توجه به مشکلات موجود، به طور معمول هر دو قسمت به صورت متوالی انجام می‌شود[۳۲]. چون پیدا کردن یک جایابی فیزیکی عملی به طور دستی معمولاً آسان است، هدف اصلی در جایابی اجازه‌ی مسیریابی با کیفیت بالا است. با این حال، تعریف «کیفیت بالا» در یک شکل ریاضی دقیق، تا حدودی سخت است. در عوض، اهداف جایگزین تعریف می‌شود. یک هدف گسترده در عمل این است که کل طول سیم تمام اتصالات را به حداقل برسانیم. متأسفانه طول دقیق سیم هر نِت مشخص نیست، تا زمانی‌که نت‌ها دقیق مسیریابی شوند. چون حتی محاسبه‌ی حداقل طول هر نِت واحد یک مسئله‌ی NP hard است.

۱.۵.۳ روش‌های جایابی

به دلیل اینکه محدودیت‌های فیزیکی در جایابی قطعات کم است، هر روشی از جمله آزمایش‌های مکرر جایابی‌های تصادفی، می‌تواند برای تولید یک جایابی عملی استفاده شود[۳۳]. در نتیجه عملاً هر طرح

[۱] SMT

اکتشافی شناخته شده از جمله توسعه‌ی خوشه[۳۴]، دانش بر مبنای سیستم‌ها[۳۵]، الگوریتم‌های جستجوی محلی تصادفی مانند SA[۱][۳۶]، و الگوریتم‌های ژنتیک[۳۷]، و نیز ترکیب این رویکردها[۳۸] برای محاسبه‌ی جایابی‌ها استفاده شده است.

۶.۳ جمع‌بندی

در فصل سوم به مدارهای چاپی، روش‌های ساخت و مونتاژ آن‌ها پرداخته شد. برای طراحی آن‌ها نرم‌افزارهای مختلفی ارائه شدند، که بهترین آن‌ها نرم‌افزار آلتیوم شرح داده شد. هرچند با وجود این نرم‌افزارها هنوز هم بر مسیریابی دستی به دلیل دقت بیشتر آن تأکید می‌شود، اما تلاش برای خودکارسازی آن وجود دارد. در فصل بعد الگوریتم پیشنهادی پایان‌نامه برای خودکارسازی طراحی و مسیریابی ارائه می‌گردد.

[۱] Simulated Annealing

فصل ۴

الگوریتم پیشنهادی

۱.۴ مقدمه

همان‌طور که در فصل‌های قبل شرح داده شد، هدف از انجام این پایان‌نامه یافتن راه‌حلی برای کوچک کردن ابعاد بردهای مدار چاپی و به حداقل رساندن طول سیم مورداستفاده در طراحی برد، با در نظر گرفتن محدودیت‌ها و قیدهای مشخصی می‌باشد. اولین و مهم‌ترین محدودیت موردنظر این است که قطعات نباید تقاطع داشته باشند و قیدهایی برای از بین بردن روی هم افتادگی قطعات موردنیاز است. قیدهای دیگر مسئله‌ی ما شامل جاگیری مناسب قطعه روی برد نسبت به قطعات دیگر، کنار هم قرار گرفتن قطعات مرتبط با هم، مکان مناسب برای قرار گرفتن قطعات قدرت، قطعات RF، کانکتورها و دیگر قطعات است.

جایابی قطعات[۲] یک فرایند ساخت الکترونیکی است که با ایجاد سیم‌کشی الکتریکی و به هم وصل کردن (باهم اتصال دادن) مدارات در PCBها، اجزا و قطعات الکتریکی را به طور دقیق بر روی PCB قرار می‌دهد. مسئله‌ی جایابی در صنعت به مسائل بسته‌بندی در فضای دوبعدی اطلاق می‌گردد که بسیاری از صنایع از جمله صنایع تولید کفش، البسه، صنایع ساخت اتومبیل، هواپیما و کشتی را شامل می‌شود. مسئله‌ی جایابی چنین تعریف می‌گردد که موقعیت و جهت اجزایی که باید در صفحه‌ای مورد نظر چیده شوند به شکلی باشد که اولاً قطعات بر روی هم چیده نشوند و ثانیاً هدف نهایی که همان کاهش مقدار اتلافی صفحه (کاهش فضای برد و یا به عبارتی میزان فیبر مصرفی کمتر برای صفحه است) است، فراهم آید. این مسئله از این جهت در صنعت بسیار مهم ارزیابی می‌شود که کوچکترین

[۲] Component Placement

صرفه‌جویی در یک کار صنعتی و با تولید انبوه که با استفاده از کاهش دورریز صفحه مهیا شده باشد، می‌تواند سالیانه تا میلیون‌ها دلار صرفه‌جویی به همراه داشته باشد. بدست آوردن یک جواب بهینه برای یک مسئله‌ی جایابی وقتی که تعداد قطعات کم و هندسه آن‌ها (شکل آن‌ها) ساده می‌باشد، کاری آسان است. ولی وقتی تعداد قطعات زیاد باشد و یا هندسه‌ی قطعات پیچیده باشد، دستیابی به جواب بهینه کار آسانی نیست. روش‌های مختلفی در طول سال‌های اخیر برای جایابی ارائه شده‌اند. از جمله روش تقریب مستطیلی، روش‌های احتمالی که الگوریتم ژنتیک جزو این روش‌ها می‌باشد و روش استفاده از سیستم‌های خبره و هوشمند[۱] که امروزه بر بسیاری از مسائل اعمال می‌گردد. روش بهینه سازی ازدحام ذرات یک الگوریتم بهینه سازی هوشمند مبتنی بر هوش جمعی است که در این پایان‌نامه برای انجام جایابی به کار گرفته شده است.

بهینه سازی ازدحام ذرات، شباهت‌های زیادی با تکنیک‌های محاسبات تکاملی مانند الگوریتم GA دارد. این سیستم با یک جمعیت از راه‌حل‌های تصادفی مقداردهی اولیه می‌شود و حالت مطلوب را با به روز رسانی نسل‌ها جستجو می‌کند. بر خلاف GA، PSO هیچ عملگری مثل تقاطع و یا جهش ندارد. در PSO راه‌حل‌های بالقوه، که ذرات[۲] نامیده می‌شوند، در سرتاسر فضای مسئله با دنبال کردن ذرات بهینه‌ی فعلی پرش می‌کنند [۳۹]. بررسی‌های انجام شده نشان می‌دهد الگوریتم بهینه سازی ازدحام ذرات به دلیل مزایای آن نسبت به دیگر الگوریتم‌های بهینه سازی که به سادگی، توابع کمتر، همگرایی سریع، و نتایج بهتر اشاره می‌کنیم، یکی از بهترین الگوریتم‌های موجود برای حل مسائل مکان‌یابی می‌باشد.

۲.۴ خلاصه‌ی فصل

بخش اول این فصل، به توصیف الگوریتم PSO، تعیین پارامترهای آن و پیاده‌سازی آن‌ها اختصاص یافته است. مرحله‌ی بعد تعیین تابع هدف برنامه، با توجه به اهداف موردنظر و محدودیت‌های موجود است. پس از استفاده از روش‌های مختلف، تابع هدف نهایی تعیین می‌شود. همچنین در اینجا از چند مدار مختلف طراحی شده با نرم‌افزار آلتیوم به عنوان دیتابیس استفاده گردیده است. درنهایت خروجی‌ها شامل شکل قرارگیری قطعات و خروجی تابع هدف نمایش داده شده و تابع هدف مربوط به بهترین خروجی به عنوان تابع هدف نهایی انتخاب می‌شود.

۳.۴ مراحل انجام پروژه

۱.۳.۴ پیاده‌سازی الگوریتم بهینه سازی ازدحام ذرات

برای پیاده‌سازی الگوریتم بهینه سازی ازدحام ذرات ابتدا به بیان مسئله پرداخته می‌شود. در این بخش ابتدا تابع هدف مسئله فراخوانی می‌شود و تعداد متغیرهای تصمیم‌گیری که قرار است به عنوان ذرات در نظر گرفته شوند و بهینه سازی شوند تعیین می‌گردد. همچنین باند بالا و پایین که متغیرها می‌توانند

[۱] Expert and Intelligent Systems
[۲] particles

در آن فضا جابه جا شوند تعیین می‌شود. چیزی که در ابتدا در نظر گرفته شده است، یک برد (یا فضای مربعی) با ابعاد فرضی 10×10 (با واحد فرضی میلی‌متر) است. این همان فضای جستجوی الگوریتم خواهد بود. ۱۰ قطعه دایره‌ای شکل با شعاع r شبیه‌سازی می‌شود به طوریکه مرکز این دایره‌ها همان نقاط فضای جستجوی ما هستند. درنتیجه تعداد متغیرهای مورد بررسی در الگوریتم ۲۰ عدد می‌باشد که به تعداد مجموع مختصات طولی و عرضی مرکز دایره‌ها از مبدأ مختصات است. با شرایطی که در الگوریتم در نظر گرفته شده است، مرکز این دایره‌ها باید درون فضای مورد نظر قرار گیرند. همچنین باید همه بخش‌های موردنیاز برای رسیدن به خروجی بهینه در الگوریتم در نظر گرفته شود. خروجی که برای سنجش عملکرد الگوریتم نمایش داده می‌شود، مقدار تابع هزینه[۱] است.

۲.۳.۴ تعیین پارامترهای الگوریتم

در این قسمت، پارامترهای الگوریتم PSO تعیین می‌شوند. این پارامترها شامل حداکثر تعداد تکرار الگوریتم، اندازه‌ی جمعیت (اندازه توده) و نیز انتخاب وزن اینرسی می‌شود. در اینجا در طی آزمایش‌ها حداکثر تعداد تکرار اجرای الگوریتم به طور معمول ۱۰۰۰ و تعداد اعضای جمعیت ۲۰۰ در نظر گفته شده است. برای اعضای جمعیت، مقدار کمتری هم می‌توان در نظر گرفت. برای وزن اینرسی، ضریب یادگیری شخصی (c_1) و ضریب یادگیری سراسری (c_2) نیز روابط معینی به کار برده شده است. همچنین برای سرعت ذرات نیز مقدار حداقل و حداکثری در نظر گرفته شده است تا ذرات از فضای جستجو خارج نگردند.

۳.۳.۴ مقداردهی اولیه‌ی الگوریتم

قسمت سوم مقداردهی اولیه‌ی الگوریتم است. موقعیت ذرات به صورت تصادفی مقداردهی اولیه می‌شوند و مقدار آن‌ها باید در بازه‌ی فضای در نظر گرفته شده باشد. سرعت اولیه‌ی ذرات نیز صفر می‌باشد. در هر تکرار مقدار تابع هزینه با توجه به موقعیت ذرات محاسبه می‌گردد و سپس مقدار بهینه‌ی شخصی و بهینه‌ی سراسری به روز رسانی می‌گردند تا مقدار جدید بهینه‌ی سراسری بدست آید. انتهای این حلقه بهترین تجربه‌ی سراسری[۲] به دست خواهد آمد.

۴.۳.۴ مرحله‌ی بهبود الگوریتم

مرحله‌ی بعد مرحله‌ی بهبود الگوریتم PSO است، که حلقه‌ی اصلی الگوریتم می‌باشد. در این قسمت سرعت و موقعیت ذرات به روز رسانی می‌گردند. در اینجا نیز محدودیت‌هایی برای سرعت و موقعیت در نظر گرفته می‌شود، تا از حدود معینی از فضای جستجو تجاوز نکنند. سپس مقدار تابع تناسب به روز رسانی می‌شود. پس از آن مقدار Gbest و Pbest به روز می‌شوند. این فرایند تا رسیدن مقدار تابع تناسب به کمترین مقدار ادامه پیدا می‌کند تا زمانی که به حد مطلوب برسد و همچنین تعداد تکرار به حدی که خواسته شده است برسد. در نهایت شکل نهایی دایره‌ها در خروجی نمایش داده می‌شود.

[۱] Cost Function
[۲] Global Best

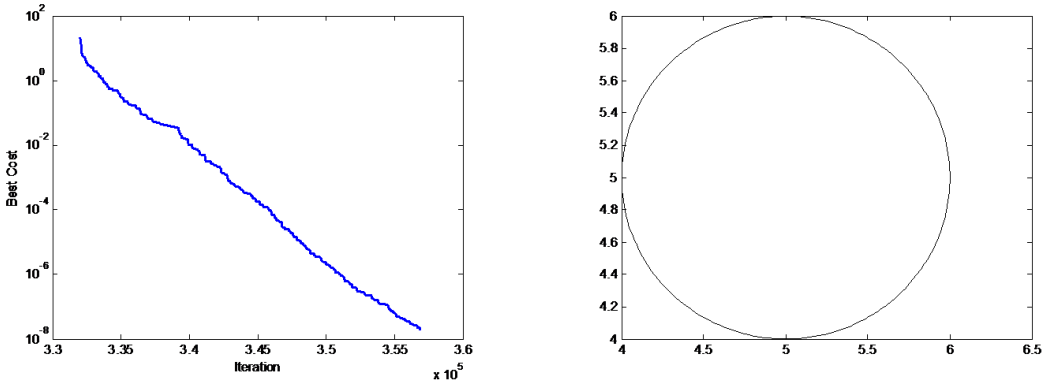
۵.۳.۴ تعیین تابع هدف

همانطور که در فصل دو شرح داده شد، برای حل هر مسئله‌ی بهینه سازی، باید یک تابع شایستگی[۱] طراحی و ابداع شود. تابع شایستگی باید از خصوصیات خاصی برخوردار باشد. این تابع باید به ازای هر مجموعه‌ای از مقادیر ورودی که توسط ذرات ارائه می‌شود پاسخ مطلوبی را برگرداند. تابع شایستگی همان تابع هدف است و باید به گونه‌ای طراحی شود که هر چه جواب ارائه شده به جواب بهینه نزدیک‌تر باشد، عدد شایستگی بزرگ‌تری برگرداند که نشان‌دهنده‌ی شایستگی بیشتر آن ذره است. در مسائل بهینه سازی، تابع هدف از اهمیت بالایی برخوردار است، چرا که کیفیت خروجی تولید شده و به طور مشخص در اینجا کیفیت جایابی و مسیریابی صحیح تراشه، به آن بستگی دارد.

در اینجا در اولین گام هدف موردنظر این است که دایره‌های یاد شده نسبت به مرکز برد کمترین فاصله‌ی ممکن را داشته باشند. در نتیجه تابع هدف، کمینه کردن فاصله‌ی مرکز دایره‌ها تا مرکز برد در نظر گرفته شده است که طبق رابطه‌ی (۱.۴) به دست می‌آید. در این رابطه x و y مختصات نقاط مرکز دایره‌ها هستند.

$$d = x^2 + y^2 \qquad (۱.۴)$$

پس از ۳۰۰ تکرار و در زمان ۶ ثانیه، مقدار خروجی تابع هدف صفر شده (عددی نزدیک به صفر) و خروجی نیز به صورت شکل (۱.۴) قابل‌مشاهده است. تمام دایره‌ها در مرکز برد بر روی هم قرار گرفته‌اند.



(آ) خروجی حاصل از اعمال کمینه کردن فاصله مرکز دایره‌ها از مرکز

(ب) مقدار Best Cost برای این مرحله

شکل ۱.۴: خروجی مرحله‌ی اول

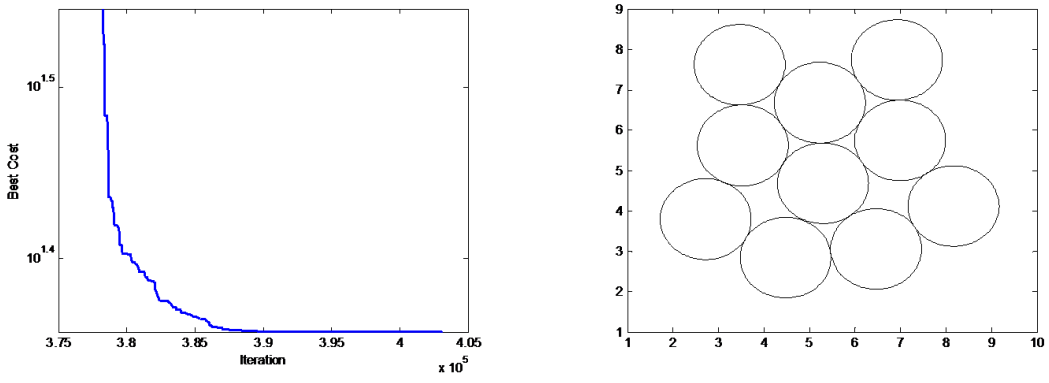
۶.۳.۴ پیاده‌سازی قید از بین بردن تداخل در اشیا

نکته‌ای که در این نوع مسائل حائز اهمیت است، ایجاد شرطی برای روی هم نیفتادن قطعات است. این موضوع یکی از قیدهایی است که با آن روبه‌رو هستیم.

[۱] Fitness Function

دستوراتی که به تابع هدف داده می‌شود که در چارچوب آن‌ها خروجی مطلوب را بدهد، در واقع نوعی قید یا محدودیت هستند که بازدهی الگوریتم را پایین می‌آورند و هرچه تعداد شرط‌ها و محدودیت‌های موجود بیشتر باشد، نزدیک کردن خروجی به حالت مطلوب دشوارتر خواهد شد. کاری که باید انجام شود این است که تابع هدف و شرط‌ها طوری تعیین گردند که خروجی موردنظر به خروجی مطلوب هر چه نزدیک‌تر باشد. تابعی که تحت عنوان تابع هزینه در الگوریتم‌های بهینه سازی و از جمله الگوریتم بهینه سازی ازدحام ذرات تعریف می‌شود، در واقع تابعی است که مقدار آن می‌بایست به صفر نزدیک باشد. اگر مسئله چند هدفه باشد، برای هر هدف یک تابع جدا در نظر گرفته می‌شود که مجموعه‌ی آن‌ها باید به صفر میل کند و مجموع آن‌ها با ارزش‌گذاری و دادن ضریب به هرکدام، تابع هدف کلی را تشکیل می‌دهد.

محدودیت‌ها[۱] تحت عنوان تابع جریمه[۲] به توابع هزینه اعمال می‌شوند. اکنون برای از بین بردن تداخل بین دایره‌های یاد شده، جریمه‌ای به این شکل در نظر گرفته می‌شود که اگر دایره‌ها روی‌هم‌افتادگی[۳] داشته باشند، به میزان روی هم افتادگی‌شان، به فاصله‌ی بین مراکز اضافه گردد به نحوی که در تکرار بعدی این تداخل از بین برود. با اعمال این دستورات به الگوریتم خود، نتیجه‌ای که حاصل می‌شود این است که پس از ۵۰۰ تکرار و در زمان ۱۲ ثانیه از شروع الگوریتم (اعضای جمعیت ۵۰ انتخاب شده است)، این دایره‌های یک شکل بدون داشتن روی هم افتادگی در کمترین فاصله نسبت به مرکز برد (فضای مربعی در نظر گرفته شده) و در کنار هم قرار می‌گیرند. خروجی حاصل از اعمال شرط نداشتن تداخل، در شکل (۲.۴) دیده می‌شود.



(آ) خروجی حاصل از اعمال دستور حذف تداخل در اشیای دایره‌ای

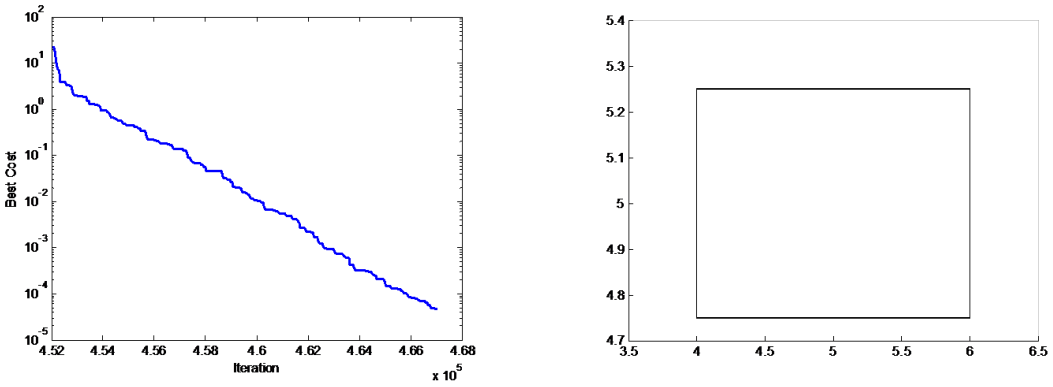
(ب) مقدار Best Cost

شکل ۲.۴: خروجی مرحله‌ی دوم

[۱] Constraints
[۲] Penalty
[۳] Overlap

۷.۳.۴ در نظر گرفتن اشیای ورودی به صورت اشکال مستطیلی و چالش‌های موجود در این زمینه

هدف ما در وحله‌ی اول این است که ابعاد هر برد دلخواه، حداقل مقدار را داشته باشد. پس در حالت ایده‌آل باید کاری کنیم که اشکال به طور منظم در یک فضای مستطیلی و در کنار هم قرار گیرند. معمولاً ابزارها و قطعات مدارهای مجتمع به شکل مربع و مستطیل هستند، در نتیجه به جای اشیای دایره‌ای، مستطیل‌هایی با طول L و عرض W در نظر گرفته می‌شود. این کار باعث می‌شود که فضای مورد آزمایش به شکل یک برد واقعی نزدیک‌تر باشد. در ابتدا طول و عرض همه مستطیل‌ها ثابت و با مقدار فرضی $L = 2$ و $W = 0.5$ در نظر گرفته می‌شود. ابتدا آزمایش قرار گرفتن مرکز همه مستطیل‌ها در مرکز برد و بدون حذف تداخل انجام می‌شود. نتایج حاصل از اجرای الگوریتم PSO در شکل (۳.۴) قابل‌مشاهده است. این نتایج پس از ۳۰۰ تکرار و زمان ۴ ثانیه به دست آمده است.



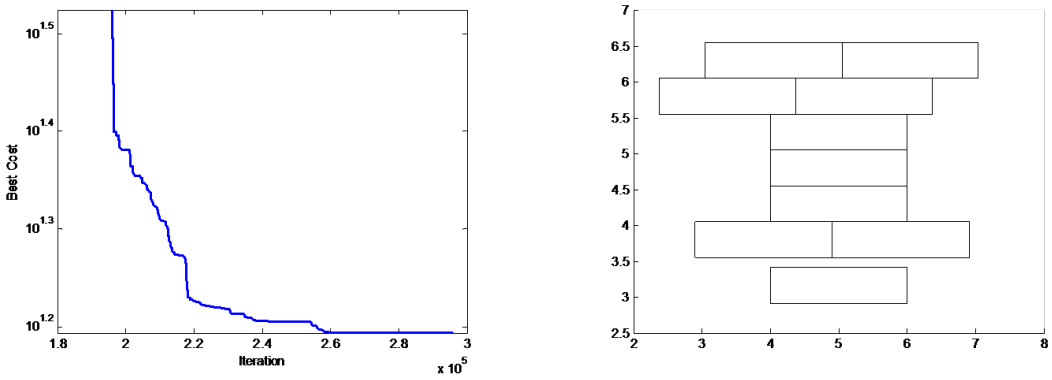
(آ) خروجی حاصل از اعمال کمینه کردن فاصله‌ی مرکز مربع‌ها از مرکز

(ب) مقدار Best Cost

شکل ۳.۴: خروجی مرحله‌ی سوم

گذاشتن شرطی برای حذف تداخل بین اشیای مستطیلی به مراتب پیچیده‌تر است. برای بدست آوردن میزان تداخل بین اشیا، تابعی پیاده‌سازی می‌گردد که تابع Overlap نام‌گذاری می‌شود و در تابع هزینه برای حذف این مقدار که در آنجا خطا[۱] نامیده می‌شود، قیدی قرار داده می‌شود. به این صورت که اگر خطا مقداری مخالف صفر داشته باشد، این مقدار خطا به عنوان یکی از هدف[۲]ها خواهد بود که باید کمینه گردد. نتیجه‌ی حاصل از اعمال این قید پس از ۲۰۰۰ تکرار و در زمان ۵۶ ثانیه در شکل (۴.۴) قابل‌مشاهده است. نتایج حاصل از اجرای الگوریتم PSO نشان می‌دهد که این الگوریتم در حالتی که یک یا دو قید به آن اعمال می‌شود به خوبی عمل می‌کند و مقدار Best Cost در آخرین تکرار برابر با ۱۵ است.

[۱] Violation
[۲] Fitness Function



(آ) خروجی حاصل از اعمال دستور حذف تداخل در اشیای مستطیلی

(ب) مقدار Best Cost

شکل ۴.۴: خروجی مرحله‌ی چهارم

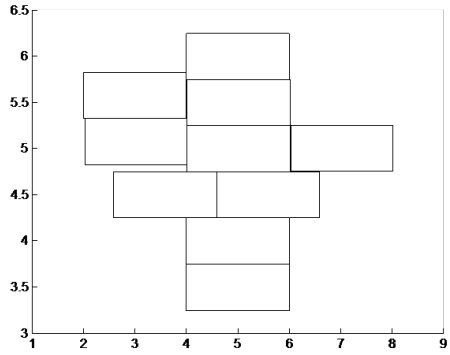
۸.۳.۴ قیدهایی برای کوچک کردن ابعاد برد مدار چاپی

مسئله‌ی دیگری که حائز اهمیت است، قرار گرفتن اشیا به طور منظم و در کمترین فضای ممکن است. پیدا کردن راه‌حل بهینه و عملی نیازمند استفاده از روش‌های مختلف و سعی و خطا و مشاهده‌ی نتایج آن‌ها و مقایسه بین آن‌ها می‌باشد. از جمله قیدهایی که برای قرارگیری منظم اشیا در یک فضای مستطیلی محدود در نظر گرفته شده است، کمینه کردن فاصله‌ی نقاط مرکز اشیا نسبت به خط $y_c = 5$ است که طبق رابطه‌ی (۲.۴) به دست می‌آید. در این رابطه Y نقاط مرکز اشیا می‌باشد.

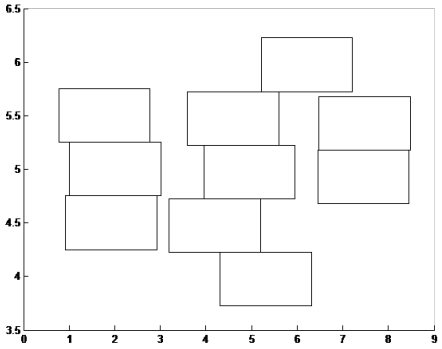
$$d = \sum \mid (Y - y_c) \mid \qquad (۲.۴)$$

پس از اعمال این هدف به برنامه و چند بار اجرا کردن آن، نتایج در یکی از دفعات اجرای برنامه در قسمت (آ) شکل (۵.۴) قابل‌مشاهده است. همچنین برای رسیدن به نتایج بهتر دو خط $x = 5$ و $y = 5$ نیز به عنوان هدف در نظر گرفته می‌شود (رابطه‌ی (۳.۴)). ولی این کار شکل خروجی مطلوب را نمی‌دهد. نتیجه‌ی حاصل در قسمت (ب) شکل (۵.۴) قابل‌مشاهده است. مشکلی که وجود دارد این است که با در نظر گرفتن چنین تابع هدفی نمی‌توان به طور بهینه از همه‌ی فضای خالی برد استفاده کرد.

$$d = \sum \mid (Y - y_c) \mid + \mid (X - x_c) \mid \qquad (۳.۴)$$



(ب) خروجی حاصل از اعمال هدف کم کردن فاصله تا $y = y_c = 5$ و $x = x_c = 5$



(آ) خروجی حاصل از اعمال هدف کم کردن فاصله تا $y = y_c = 5$

شکل ۵.۴: خروجی مرحله‌ی پنجم

۹.۳.۴ پیاده‌سازی قید کاهش سیم‌بندی

تا به اینجا هدف حداقل کردن فاصله‌ی مرکز اشیا نسبت به خط $y = 5$ در نظر گرفته می‌شود. سپس قیدهای دیگری در تابع هدف قرار داده می‌شود. قید مهمی که جزو اهداف مسئله محسوب می‌گردد، تعداد سیم‌های موجود بین قطعات مجتمع به کار رفته می‌باشد. ابتدا به صورت فرضی یک ماتریس با مقادیر دلخواه به عنوان ماتریس وزن در نظر گرفته می‌شود (البته مقادیر این ماتریس مربوط به یک مدار واقعی USBaspمی‌باشد). درایه‌های ماتریس وزن (w) نشان دهنده‌ی ارزش بین هر دو شیء می‌باشد. هدف این است که هر دو شیئی که وزن بین آن‌ها بیشتر است، کمترین فاصله را نسبت به هم داشته باشند. ماتریس زیر، ماتریس وزن فرضی است که از آن استفاده شده است.

$$w = \begin{bmatrix} 0 & 4 & 0 & 0 & 4 & 4 & 2 & 0 & 1 & 1 \\ 4 & 0 & 2 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

پس از استفاده از روش‌های مختلف برای پیدا کردن بهترین شکل تابع هدف برای بهینه کردن این ارتباط‌ها، با مطالعه مقالات مربوطه، طبق روش استفاده شده در [۶] عمل می‌شود. اگر فاصله‌ی مرکز

اشیا به صورت زیر بدست آید:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{۴.۴}$$

معادله‌ی (۵.۴) را به عنوان هدف در نظر می‌گیریم.

$$C = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} w_{ij} \tag{۵.۴}$$

در معادله‌ی فوق ماتریس w (یا ماتریس وزن) شامل تعداد ارتباطات بین n نقطه است و d فاصله‌ی هر دو قطعه از یکدیگر است که با فرمول (۴.۴) محاسبه می‌گردد.

۱۰.۳.۴ ارزش‌گذاری و تعیین ضریب برای هدف‌ها و قیدها

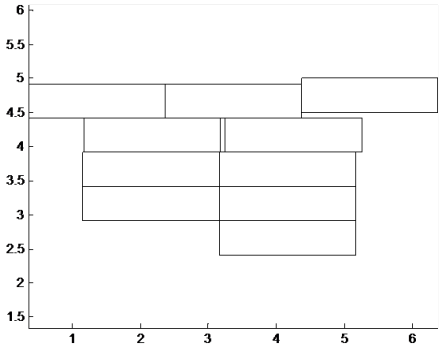
ارزش‌گذاری بین شرط‌های اعمال شده به مسئله نیز نکته‌ی قابل توجهی است. در اینجا چند هدف متفاوت وجود دارد که در تناقض با هم هستند، اینکه اشیا با هم تداخل نداشته باشند و همزمان فاصله‌ی آن‌ها برای رسیدن به حداقل سیم‌بندی کم شود در تناقض با هم هستند. زیرا در یکی از این شرط‌ها الگوریتم سعی می‌کند قطعات را از هم دور کند و در دیگری سعی در نزدیک کردن آن‌ها به یکدیگر دارد به نحوی که اگر شرط حذف تداخل حذف شود، دایره‌ها کاملاً روی هم قرار می‌گیرند و مرکز آن‌ها در یک نقطه‌ی $(x_1، y_1)$ قرار می‌گیرد. در استفاده از الگوریتم‌های بهینه سازی چندهدفه، یکی از کارهایی که باید انجام شود ارزش‌گذاری برای هریک از هدف‌ها است. هر یک از هدف‌ها یک ضریب دارند که نشانگر ارزش آن هدف است و الگوریتم با توجه به آن معادله را حل می‌کند.

در اینجا ضریب هر دو هدف ۱ قرار داده شده است. نتایج حاصل از اجرای الگوریتم با داشتن دو هدف فوق پس از ۲۰۰۰ تکرار و زمان ۱۲۰ ثانیه در شکل (۶.۴) قابل‌مشاهده است. مقدار Best Cost پس از ۱۰۰۰ تکرار برابر با ۵۷ می‌باشد. در شکل‌ها مشاهده می‌شود که مقدار هدف مربوط به کم کردن فاصله (Connection) کاهش چشمگیری داشته است. همچنین فاصله از خط $y = 5$ نیز کم شده است.

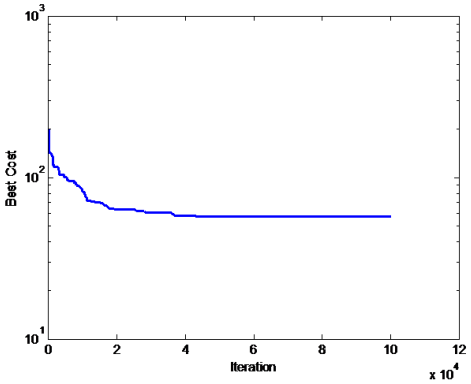
۱۱.۳.۴ استفاده از مدار فرضی طراحی شده در نرم افزار آلتیوم به عنوان داده‌ی ورودی در متلب

در این قسمت از مدارهای pcb که در نرم افزار آلتیوم طراحی شده است به عنوان پایگاه داده استفاده می‌شود. ابتدا از یک برد فرضی استفاده می‌شود که شامل ۱۵ قطعه است و مدل شماتیک و برد pcb آن در نرم افزار آلتیوم، در شکل (۷.۴) دیده می‌شود. در اینجا ابعاد قطعات یکسان فرض شده‌اند.

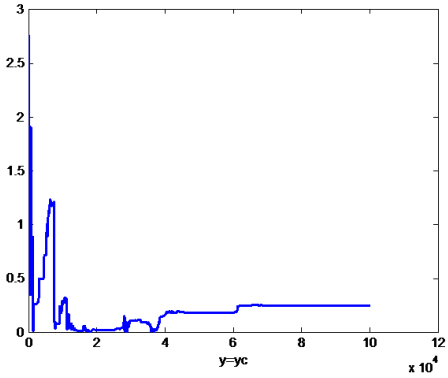
در نرم افزار آلتیوم، مشخصات قطعات از جمله علامت اختصاری هر قطعه، کتابخانه‌ای که قطعه از آن‌جا انتخاب شده است و تعداد پین‌های آن را می‌توان استخراج کرد و مورد استفاده قرار داد.



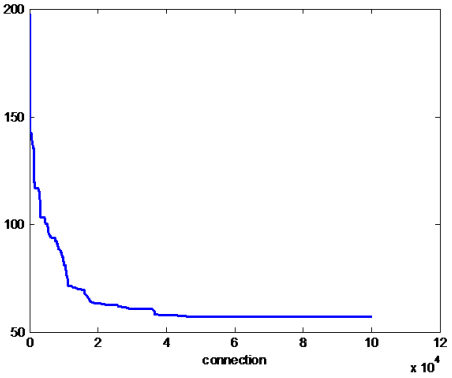
(آ) شکل خروجی



(ب) مقدار Best Cost برای این مرحله

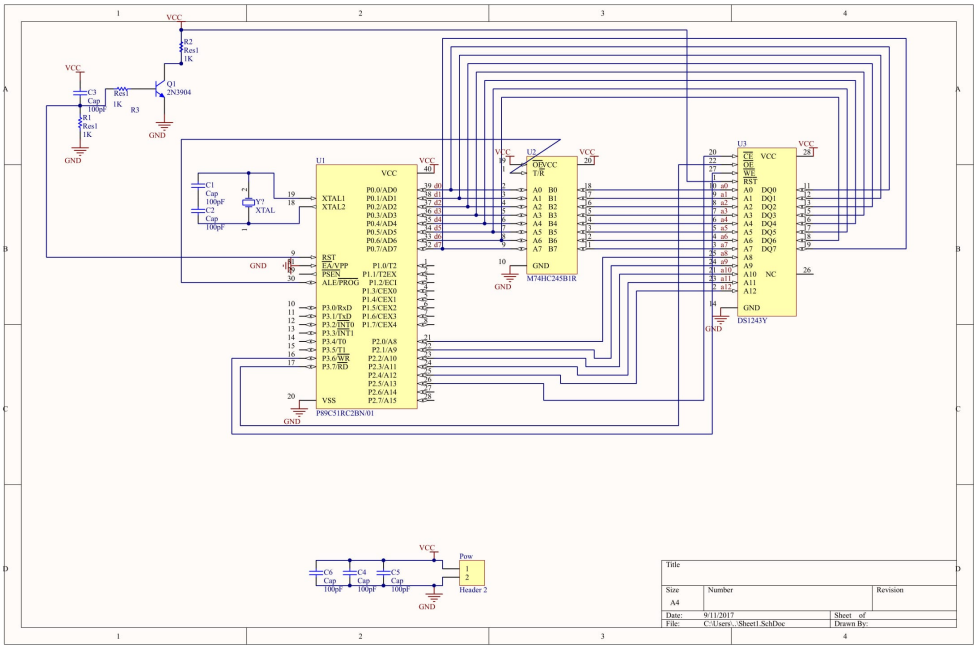


(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$

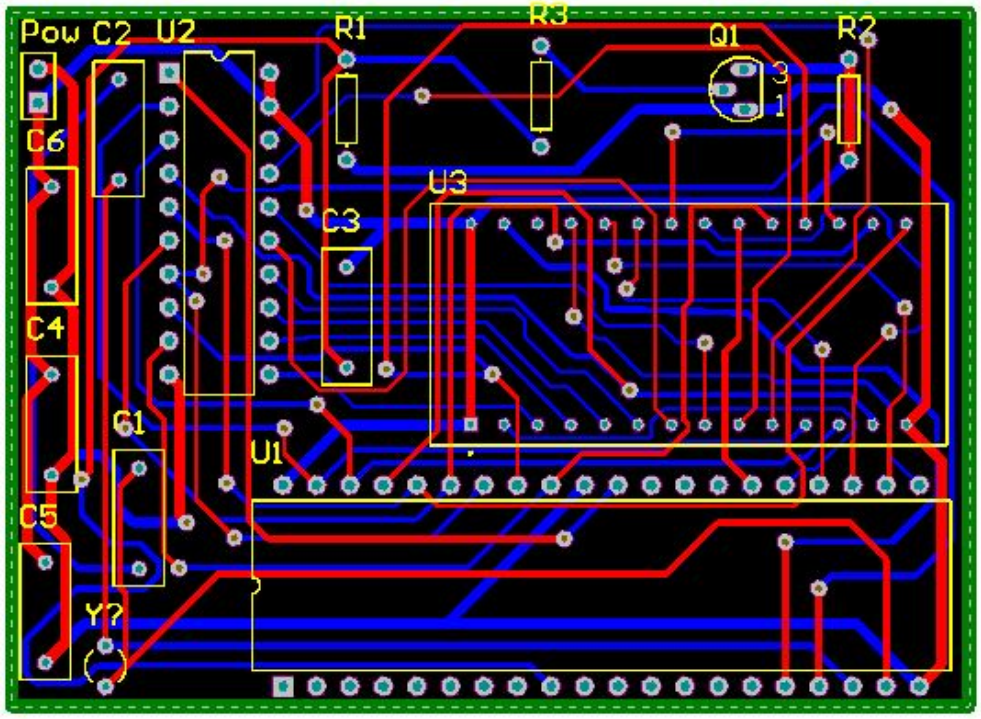


(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات

شکل ۶.۴: خروجی مرحله‌ی ششم



(آ) طرح شماتیک مدار فرضی



(ب) طرح PCB مدار فرضی

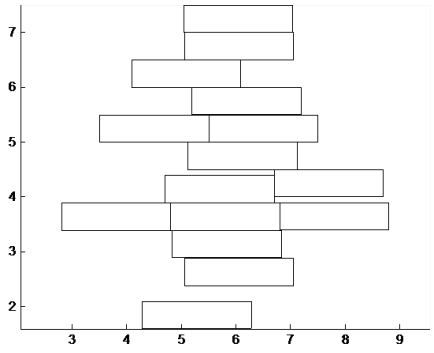
شکل ۷.۴: طرح مدار فرضی

همچنین در قسمت PCB مدار، می‌توان ابعاد قطعات را بدست آورد. در هنگام استفاده از بردهای طراحی شده با نرم‌افزار آلتیوم به عنوان دیتابیس، باید مشخصات آن را نیز داشته باشیم. اطلاعات موردنظر در نرم افزار آلتیوم درون فایلی با فرمت Net. قابل‌دستیابی است و این اطلاعات با نوشتن برنامه‌ی مربوطه، در نرم‌افزار متلب و در داخل الگوریتم PSO بارگذاری می‌گردد، به نحوی که خود الگوریتم اطلاعات را از روی فایل ذخیره شده با فرمت mat. می‌خواند. به این ترتیب می‌توان نام قطعات مربوط به هر برد pcb که در آلتیوم طراحی شده است را به عنوان داده‌ی ورودی در اختیار الگوریتم قرار داد تا جایابی صحیح را در مورد آن انجام دهد، و در شکل خروجی مکان قطعات با نام آن‌ها قابل نمایش است. همچنین می‌توان برنامه‌ای پیاده‌سازی کرد که ماتریس وزن مربوط به مدار را در متلب فراخوانی کند و از آن برای بهینه کردن ارتباط‌ها استفاده کرد. در ابتدا همان‌طور که گفته شد برد فرضیی که در شکل (۷.۴) مشاهده می‌شود مورد آزمایش قرار می‌گیرد.

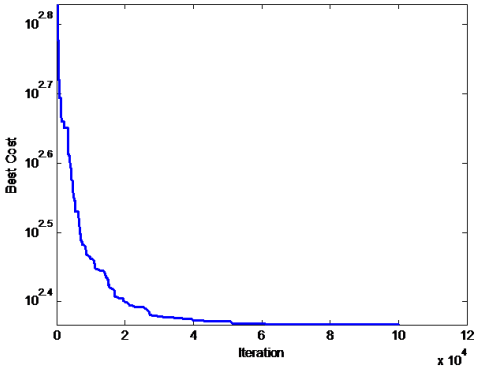
ماتریس وزن برای این مدار به صورت زیر به دست آمده است و فعلاً ابعاد همه‌ی قطعات یکسان در نظر گرفته شده است.

$$w = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 1 & 1 & 0 & 2 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 1 & 0 & 2 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 1 & 1 & 0 & 2 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 1 & 0 & 2 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11 & 18 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 18 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

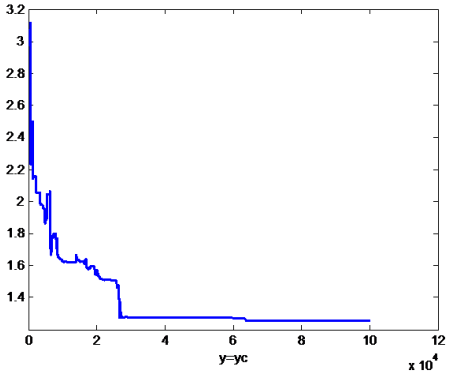
خروجی حاصل در حالتی که ماتریس وزن به صورت بالا باشد و دو هدف کمینه شدن فاصله نسبت به $y = y_c = 5$ و کم شدن فاصله سیم‌بندی‌ها و همچنین اعمال قید حذف تداخل در نظر گرفته شود، پس از ۲۰۰۰ تکرار و با جمعیت ۵۰ در شکل (۸.۴) مشاهده می‌شود. در اینجا نیز ضریب همه‌ی اهداف و قید ۱ در نظر گرفته شده است. آخرین مقدار Best Cost در اینجا برابر با ۲۳۰ و زمان طی شده ۲۰۰ ثانیه می‌باشد.



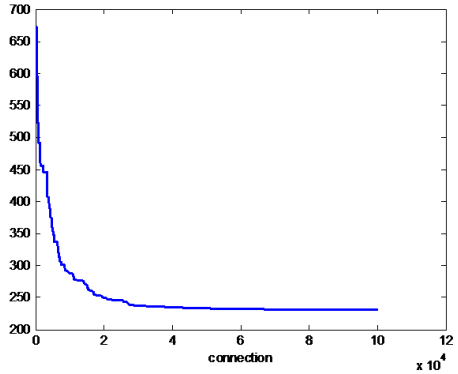
(آ) شکل خروجی



(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$



(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات

شکل ۸.۴: خروجی مرحله‌ی هفتم

۱۲.۳.۴ تعیین قید حذف تداخل برای اشیای مستطیلی

دستوری که در این بخش برای حذف تداخل داده شده است به این صورت است که پس از محاسبه‌ی مقدار تداخل (با استفاده از تابع نوشته شده با نام overlap)، اگر مقدار خطا[۱] مخالف صفر بود، مقدار تابع هدف کلی (تابع هزینه یا z) با یک ضریب λ از مقدار تابع هدفِ (D) و مقدار خطا (violation) تغییر کند که رابطه‌ی آن در زیر قابل‌مشاهده است. پس از آن باید مقداری که از این رابطه بدست آمد را به عنوان هزینه (z) در نظر گرفت. در غیر اینصورت، اگر مقدار خطا برابر با صفر بود $z = D$ شود. در اینجا مقدار $\lambda = 100$ در نظر گرفته می‌شود.

$$z = \frac{D^2}{(violation + \epsilon) \times \lambda} \tag{۶.۴}$$

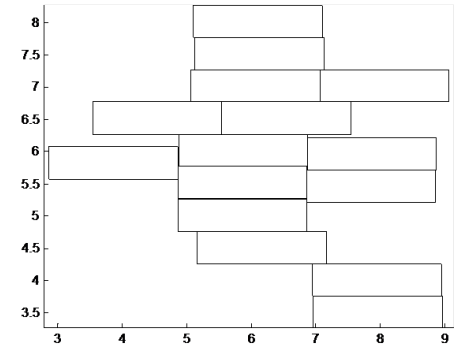
۱۳.۳.۴ تعیین هدفی برای کوچک کردن مساحت برد مدار چاپی

یکی از قیدهایی که در بخش تعریف مسئله‌ی الگوریتم برای کوچک کردن ابعاد و فضای اشغال شده توسط تراشه در نظر گرفته شده است این است که ابعاد برد به صورت دستی مقداردهی نشود بلکه به صورت هوشمند مقداردهی انجام شود، به این صورت که ابعاد برد ضریبی از مساحت کل اشیا باشد. به این ترتیب حداقل اندازه‌ی مشخصی برای مساحت کل فضای جستجو در دسترس است. اما می‌توان برای بهبود بیشتر این مساحت، یک شرط دیگر به تابع هدف اضافه کرد. با داشتن مقدار مساحت کل قطعات، می‌توان مقدار مساحت اشغال شده توسط اشیا را به آن میل داد. اگر مقدار مساحت کل قطعات با S و مقدار مساحت اشغال شده توسط اشیا به صورت یک مستطیل با area نشان داده شود، میل دادن مقدار area به S در تابع هزینه، می‌تواند به بهبود و کوچک کردن ابعاد برد کمک کند. در این مرحله این هدف با ضریب مناسب در تابع هزینه در نظر گرفته می‌شود. نتیجه‌ی حاصل در شکل (۹.۴) مشاهده می‌شود. زمان سپری شده در اجرای این مرحله ۲۱۸ ثانیه و آخرین مقدار Best Cost ۲۳۲ به دست آمده است.

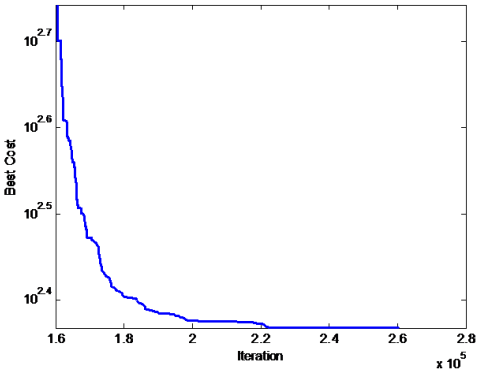
همین آزمایش در صورتی که ضریب مربوط به کاهش ابعاد ۱۰۰ در نظر گفته می‌شود، به صورت شکل (۱۰.۴) تغییر می‌کند. مشاهده می‌شود ابعاد کاهش زیادی پیدا کرده است و تا حدود زیادی بهینه شده است. (زمان طی شده ۲۲۰ ثانیه، و آخرین مقدار به دست آمده برای Best Cost، ۳۸۲ می‌باشد.)

مشکلی که وجود دارد این است که برنامه‌ی فوق بعضی مواقع هنگامی که اجرا می‌شود تداخل را از بین نمی‌برد. همچنین اگر در تابع تداخل زمانی که خطا مخالف صفر است $z = inf$ قرار دهیم همانند آنچه که در مراحل اول انجام داده شده است، متلب در اکثر مواقع خطا می‌گیرد. در مقایسه‌هایی که انجام شده است، مواقعی که خطا هم گرفته نشود و برنامه اجرا شود، برنامه نتیجه‌ی مطلوب را نمی‌دهد.

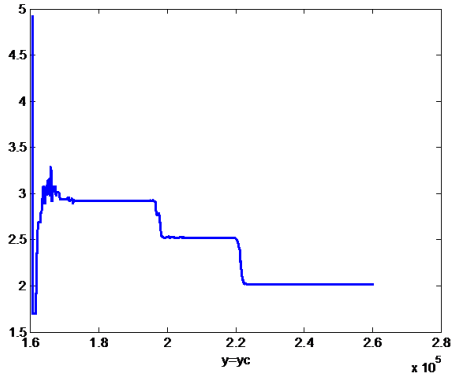
[۱] violation



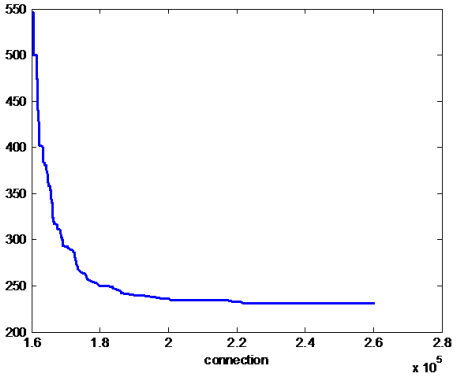
(آ) شکل خروجی



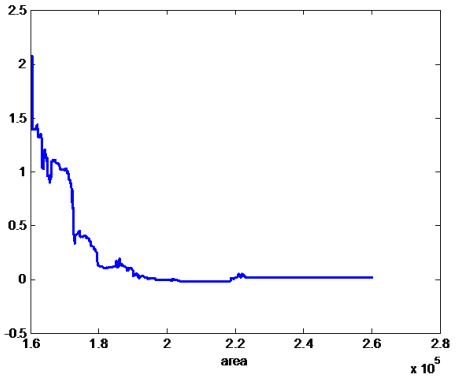
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$

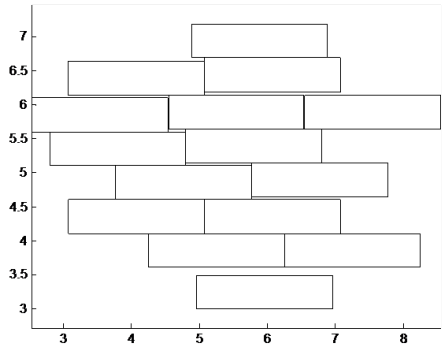


(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات

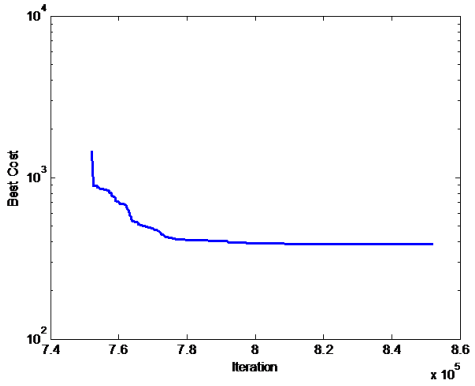


(ه) نمودار نتیجه‌ی کاهش مساحت برد

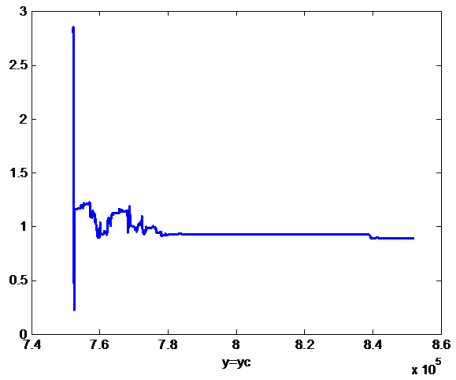
شکل ۹.۴: خروجی مرحله‌ی هشتم



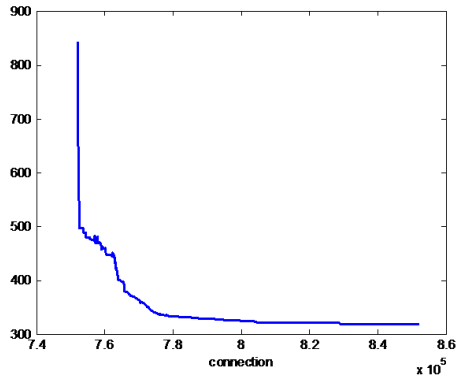
(آ) شکل خروجی



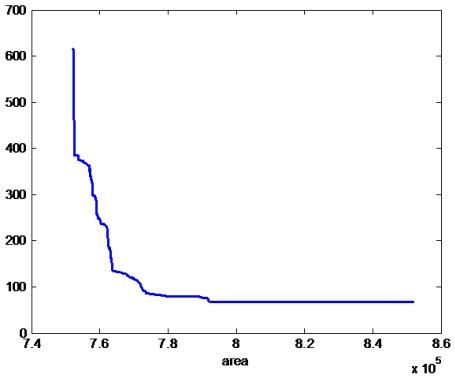
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$



(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



(ه) نمودار نتیجه‌ی کاهش مساحت برد

شکل ۴.۱۰: خروجی مرحله‌ی نهم

۴.۳.۴ رفع اشکالات ایجاد شده با استفاده از تغییر دستور حذف تداخل

برای حذف تداخل بین قطعات مستطیلی، می‌توان به جای استفاده از تابعی که خودمان نوشته‌ایم، از دستور آماده‌ی متلب به نام $rectint$ که میزان تداخل بین قطعات مستطیلی را نشان می‌دهد استفاده کرد. در این حالت دیگر تداخل به عنوان قید و محدودیت قرار داده نمی‌شود، بلکه به عنوان یکی از

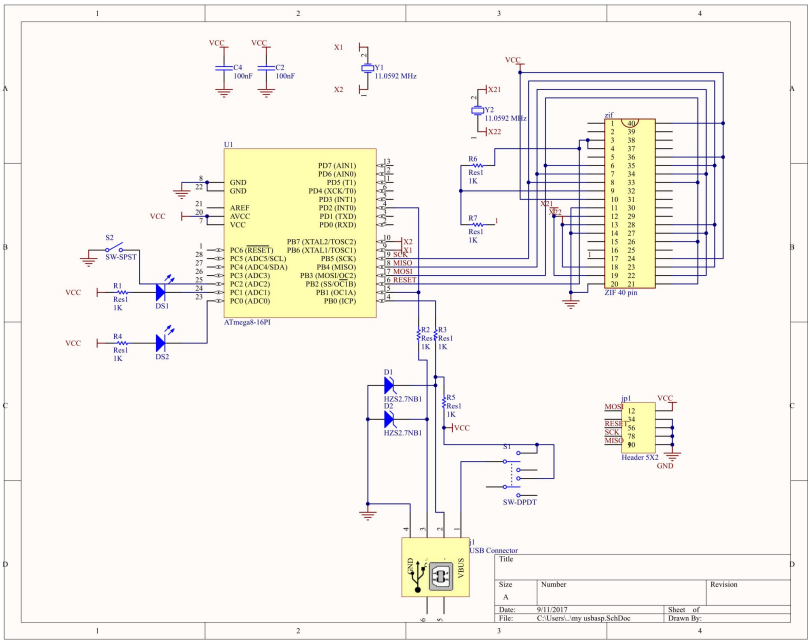
هدف‌ها در تابع هزینه قرار می‌گیرد.

۱۵.۳.۴ طراحی یک مدار واقعی به عنوان ورودی

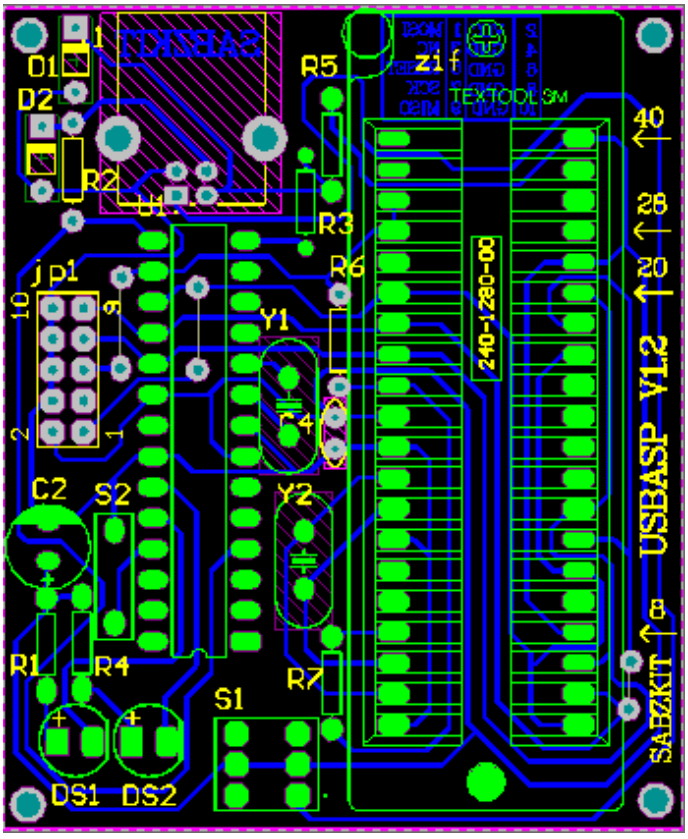
در این قسمت شماتیک یک برد مداری متعلق به یک پروگرامر in-circuit USB برای کنترلر Atmel AVR، که متشکل از ۲۱ قطعه از جمله میکروکنترلر ATMega8 و تعدادی از قطعات پسیو کوپل شده است، در نرم افزار آلتیوم طراحی و شبیه سازی شده، و سپس تبدیل به سند PCB آن نیز صورت می‌گیرد. طرح شماتیک و سند PCB مربوط به این مدار در شکل (۱۱.۴) قرار داده شده است.

همانطور که گفته شد، مشخصات هر کدام از قطعات شامل نام آن‌ها و ارتباط بین قطعات با یکدیگر، در صورت نیاز می‌تواند مورد استفاده‌ی الگوریتم قرار بگیرد. فایلی به نام component.mat در متلب ذخیره شده است که توسط الگوریتم بارگذاری می‌شود و ماتریس وزن با نام matcomp نمایش داده می‌شود که از آن در تابع هدف استفاده می‌شود. در اینجا ماتریس وزن نمایش داده شده است:

$$w = \begin{bmatrix}
0 & 2 & 1 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 2 & 0 & 0 & 2 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 2 & 0 & 0 & 2 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 6 & 0 & 0 & 6 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 6 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$



(آ) طرح شماتیک مدار *usbasp*



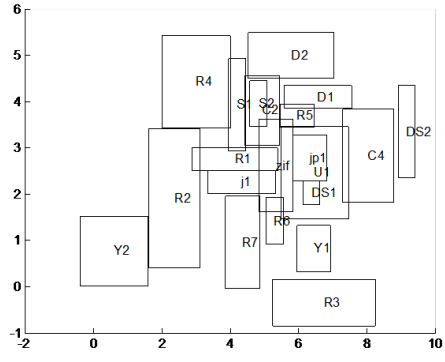
(ب) طرح *pcb* مدار *usbasp*

شکل ۱۱.۴: مدار *usbasp*

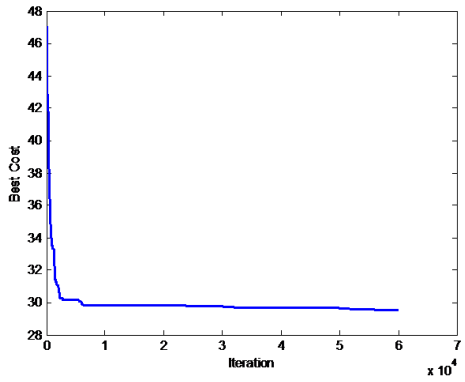
۱۶.۳.۴ اعمال هر سه قید هدف به مسئله

در این قسمت تابع هدفی شامل سه بخش مختلف حذف تداخل، حداقل فاصله تا خط $Y = Y_C = 2.5$ و کاهش سیم‌بندی در نظر گرفته می‌شود. ضریبی که برای هرکدام در نظر گرفته شده است به ترتیب ۱، ۰.۱ و ۰.۱ می باشد. نتیجه‌ی حاصل از اجرای الگوریتم پس از ۲۰۰۰ تکرار و انتخاب تعداد اعضای جمعیت ۳۰ برای ذرات، در شکل (۱۲.۴) قابل‌مشاهده است. آخرین مقدار Best Cost ۲۹ است که پس از گذشت زمان ۵۸۰ ثانیه به دست آمده است. (فعلاً مقادیر طول و عرض قطعات فرضی در نظر گرفته شده‌اند و مقادیر واقعی نیستند.) مساحت اشغال شده حدود ۵۰ است و مساحت کل قطعات ۳۸.۵ است.

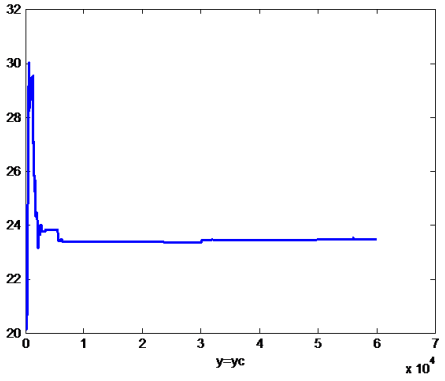
مشاهده می‌شود که با وجود این شرط‌ها تداخل به شکل مطلوب از بین نرفته است و اگر ضریب مربوط به حذف تداخل مقدار بیشتری داده شود تا بهتر بتواند تداخل را از بین ببرد، شرایط دیگر خواسته شده مانند هدف کاهش سیم‌بندی با مشکل مواجه می‌شود. همچنین حتی با زیاد کردن مقدار مساحت در نظر گرفته شده برای برد (مقدار ماکزیموم high طول برد در الگوریتم از ۵ به ۸ تغییر داده شد)، باز هم مشکل تداخل وجود دارد. در نتیجه باید شرط‌های تابع و نیز الگوریتم تغییرات دیگری پیدا کنند. نتیجه‌ی این آزمایش وقتی ضرایب هدف‌ها به ترتیب ۱، ۲۰ و صفر است نیز به صورت شکل (۱۳.۴) است.



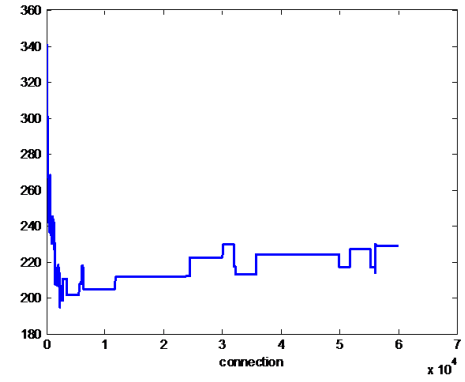
(آ) شکل خروجی



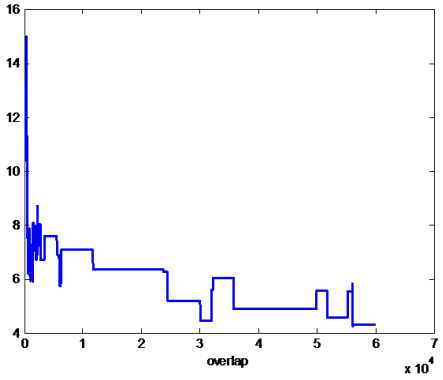
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$

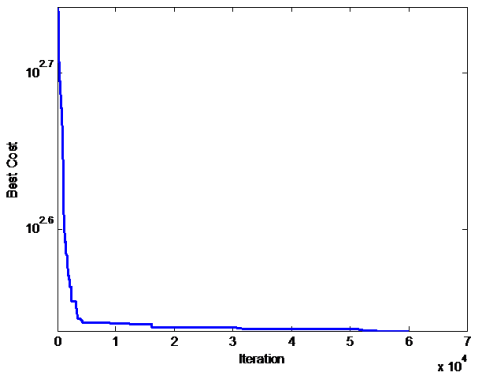


(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات

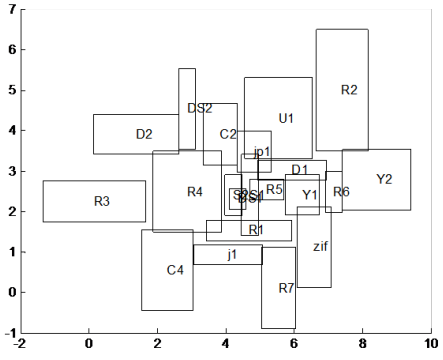


(ه) نمودار نتیجه‌ی حذف تداخل

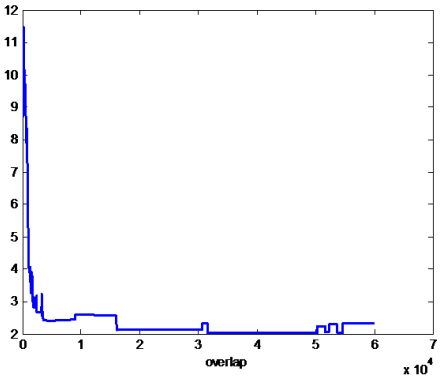
شکل ۱۲.۴: خروجی مرحله‌ی دهم



(ب) مقدار Best Cost



(آ) شکل خروجی



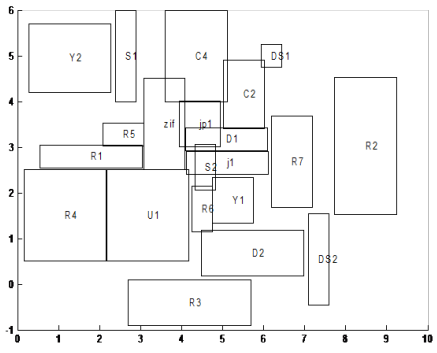
(ج) نمودار نتیجه‌ی حذف تداخل

شکل ۱۳.۴: خروجی مرحله یازدهم

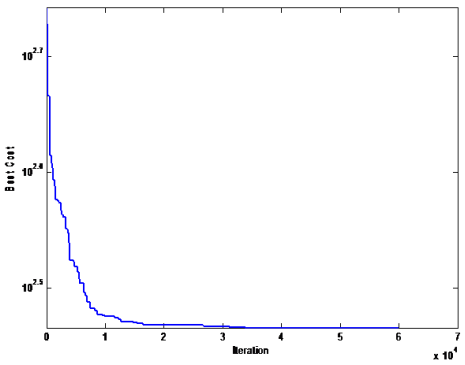
یکی از مشکلاتی که وجود دارد این است که الگوریتم ذرات را خوب جا به جا نمی‌کند. در واقع یکی از مشکلات الگوریتم PSO این است که ذرات سعی می‌کنند در جای خود بهترین موقعیت را داشته باشند و

جابه‌جایی زیادی برای رسیدن به موقعیت بهتر انجام نمی‌دهند. برای رفع این مشکل در الگوریتم PSO عملگر جهش[۱] را با الگوریتم ترکیب می‌کنیم. ابتدا درباره‌ی عملکرد الگوریتم PSO بعد از اضافه کردن جهش در پروسه‌ی آزمایش بحث می‌شود. مزیت اصلی پیاده‌سازی جهش، افزایش تغییرپذیری جمعیت می‌باشد. جهش می‌تواند مشکل به دام افتادن PSO را کاهش دهد و نتایج جستجوی بهتری را نتیجه دهد. از سوی دیگر اشکال آن سخت بودن تنظیم پارامترهای جهش برای هر مسئله‌ی خاص است. به عبارتی اگر پارامترها under-mutated باشند، به دام افتادن می‌تواند اتفاق بیفتد و درواقع جهش را بی‌فایده می‌سازد [۴۰].

در اینجا آزمایش اخیر (با ضرایب ۲۰ و ۱ و ۰) با در نظر گرفتن عملگر جهش در الگوریتم انجام می‌شود و نتیجه را خواهیم دید. تعداد اعضای جمعیت ۳۰ انتخاب شده است. نتایج حاصل از این آزمایش نشان می‌دهد که با ترکیب الگوریتم PSO با اپراتور جهش، خروجی بهتری حاصل می‌گردد. مشاهده می‌شود که تداخل پس از ۱۵۰۰ تکرار حذف می‌شود. نتیجه پس از ۲۰۰۰ تکرار در شکل (۱۴.۴) قابل‌مشاهده است.

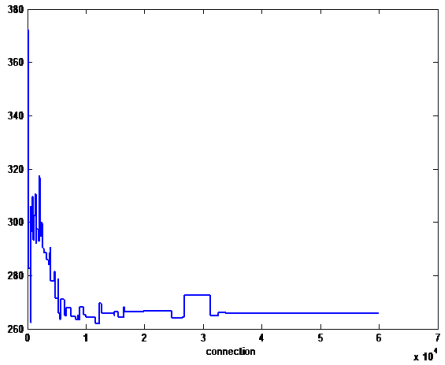


(آ) شکل خروجی

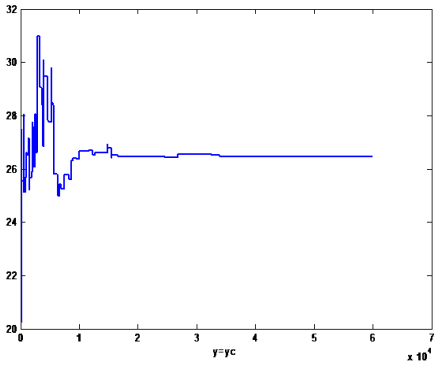


(ب) مقدار Best Cost

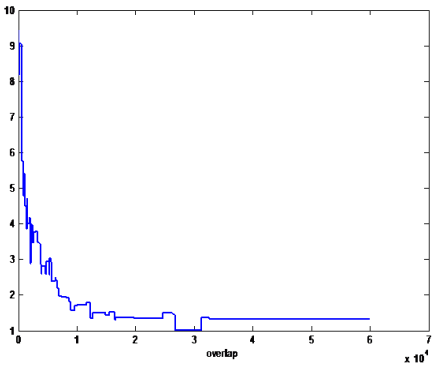
[۱] mutation



(د) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



(ج) نمودار نتیجه‌ی کم کردن فاصله از خط $y = y_c$



(ه) نمودار نتیجه‌ی حذف تداخل

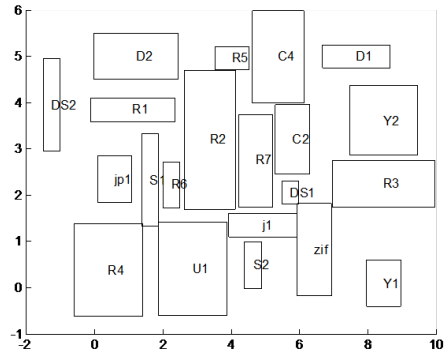
شکل ۱۴.۴: خروجی مرحله دوازدهم

۱۷.۳.۴ ایجاد تغییراتی در تابع هدف و استفاده از دستور rectint در متلب برای حذف تداخل

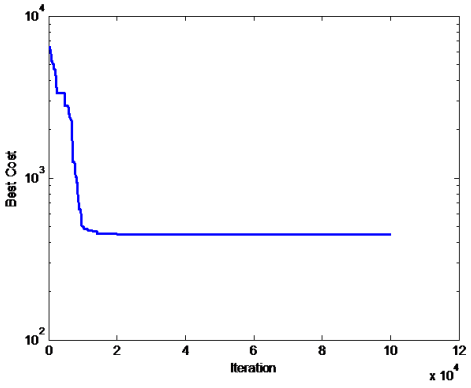
در آزمایش بعدی، تغییراتی در تابع هدف صورت می‌گیرد که نتایج آن بهبود می‌یابد. به این صورت که تابع هدف علاوه بر اینکه شامل دستور حذف تداخل و بهینه کردن ارتباط قطعات (کاهش سیم‌بندی) می‌باشد، شرط مربوط به کمینه کردن فاصله تا خط $y = y_c$ حذف شده و هدف کوچک کردن ابعاد برد مدار چاپی جایگزین آن می‌شود. برای این کار همانطور که قبلاً به آن پرداخته شد، دستوری اعمال می‌شود که ابعاد برد به اندازه‌ی مساحت کل قطعات شود تا فضای خالی بین قطعات را حذف کند.

همچنین می‌توان تابع حذف تداخل را به عنوان تابع هدف مورد استفاده قرار داد. به این صورت که مقدار تداخل بین قطعات را با دستور rectint به دست آورده و دستوری به این شکل نوشته می‌شود که هرجا خطای تداخل وجود داشت مقدار آن با ضریب a با مقدار z (تابع هزینه) جمع بسته شود. با این کار تداخل از بین می‌رود. در این حالت ضریب قید تداخل نمی‌تواند یک باشد. در این آزمایش ابتدا a تا ۲۰۰۰ بالا برده شد و در طول آزمایش‌های مختلف مقدار آن کم شد تا زمانیکه در مقدار $a = 800$ تداخل را از بین می‌برد و مقادیر کوچکتر از آن تداخل حذف نمی‌شود. در نتیجه $a = 800$ انتخاب

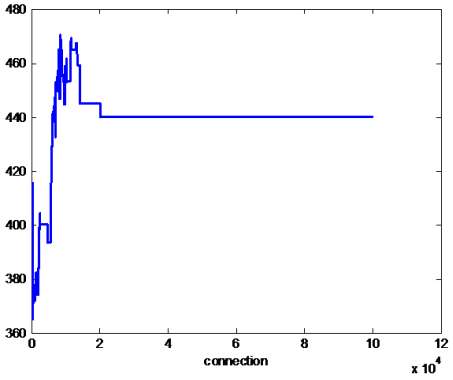
می‌شود. همچنین در اینجا عملگر جهش نیز با ضریب mu=0.1 استفاده شده است. نتیجه‌ی حاصل از این آزمایش در شکل (۱۵.۴) قابل‌مشاهده است. (تعداد جمعیت ذرات ۵۰)



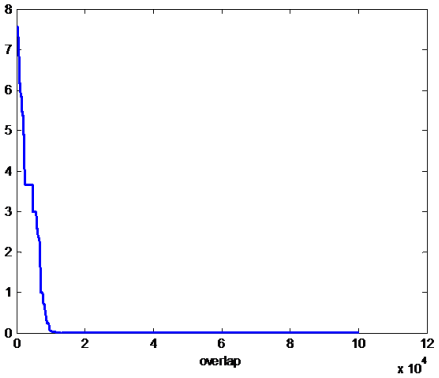
(آ) شکل خروجی



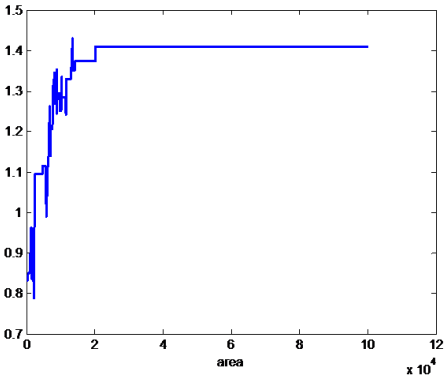
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



(د) نمودار نتیجه‌ی حذف تداخل



(ه) نمودار نتیجه‌ی کاهش مساحت برد

شکل ۱۵.۴: خروجی مرحله‌ی سیزدهم

در این آزمایش تابع هدف مربوط به area نرمالیزه شده است ولی هدف مربوط به connection و overlap مقادیر واقعی آن‌ها می‌باشد.

۱۸.۳.۴ امتحان کردن روش هوشمندسازی ضرایب توابع هدف

هوشمند کردن ضرایب توابع هدف به این معنی است که ضرایب خودشان به نسبت تأثیرگذاری که دارند تغییر کنند و نیازی به مقدار دستی دادن به آن‌ها وجود داشته باشد. برای اینکار ابتدا همه‌ی توابع هدف نرمالیزه می‌شوند تا مقداری که در خروجی این هدف‌ها به دست می‌آید باهم تناسب داشته باشند و تأثیر یکسانی در الگوریتم بگذارند. به عنوان مثال مقادیر همه‌ی آن‌ها زیر ۱ باشد. سپس برای هوشمند کردن ضرایب روش‌هایی به کار برده شدند ولی نتیجه‌ی مطلوب حاصل نگردید. در نتیجه فعلاً به صورت دستی مقداردهی می‌شوند.

درنهایت از این پس برای هر کدام از هدف‌ها یک ضریب تعیین می‌گردد تا با تغییر این ضرایب بهترین موقعیت با بهترین نتایج مربوط به همه‌ی تابع هدف‌ها مشخص شود. ضریب a_1 مربوط به هدف بهینه کردن ارتباط سیم‌بندی، ضریب a_2 مربوط به هدف بهینه کردن مساحت و ضریب a_3 مربوط به قید حذف تداخل است. به طور کلی اگر ضرایب هدف‌ها با a_1 و a_2 و a_3 نشان داده شوند، رابطه‌ی (۷.۴) به عنوان تابع هدف کلی در نظر گرفته می‌شود:

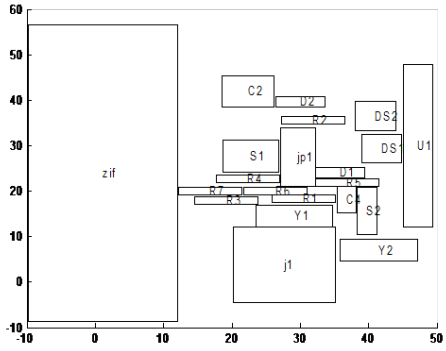
$$z = a_1 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} w_{ij} + a_2 S + a_3 E \qquad (۷.۴)$$

که در رابطه‌ی فوق S مقدار مربوط به هدف مساحت و E مقدار خطای تداخل است. علاوه بر نرمالیزه کردن رابطه‌ی کاهش مساحت، در اینجا قید کاهش سیم‌بندی نیز نرمالیزه شده است.

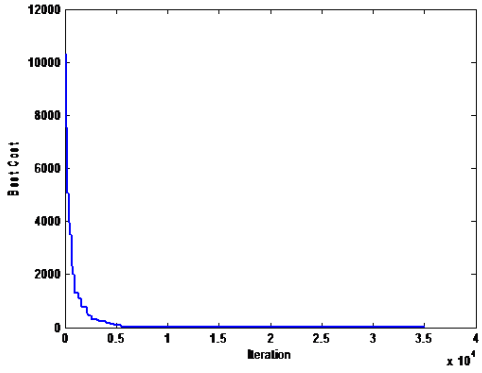
۱۹.۳.۴ استخراج مقادیر واقعی قطعات از نرم افزار آلتیوم و به‌کارگیری آن‌ها درالگوریتم

همچنین در این مرحله تغییراتی در مسئله ایجاد می‌شود. ابعاد واقعی قطعات از نرم افزار آلتیوم دیزاینر استخراج شده و در اختیار مسئله قرار داده می‌شود. یکی از بهترین خروجی‌ها در حالتی که ضرایب به ترتیب ۱ و ۱.۶ و ۰.۶ انتخاب شده است در شکل (۱۶.۴) مشاهده می‌شود. در این آزمایش از جهش با ضریب $mu = 0.1$ نیز استفاده شده است.

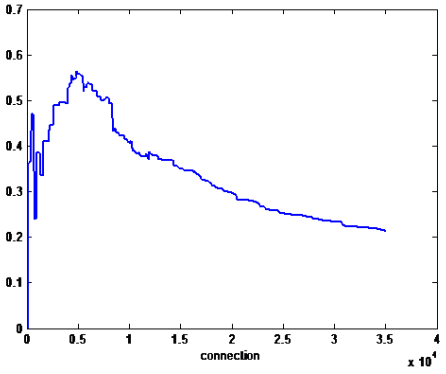
همانطور که در نمودارهای خروجی مشاهده می‌شود الگوریتم PSO توانسته است در مدت زمان کوتاهی، در اولین تکرارها تداخل بین قطعات را از بین ببرد و پس از حذف تداخل، دو شرط دیگر یعنی سیم‌بندی و ابعاد برد کاهش چشمگیری پیدا می‌کند، تا اینکه آخرین شکل خروجی به صورت قسمت (آ) شکل (۱۶.۴) به دست می‌آید.



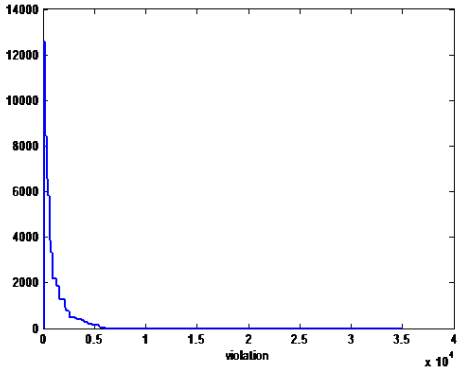
(آ) شکل خروجی



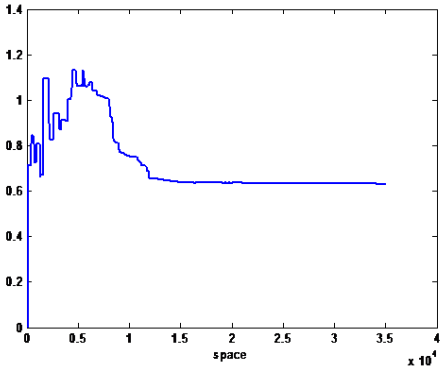
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



(د) نمودار نتیجه‌ی حذف تداخل



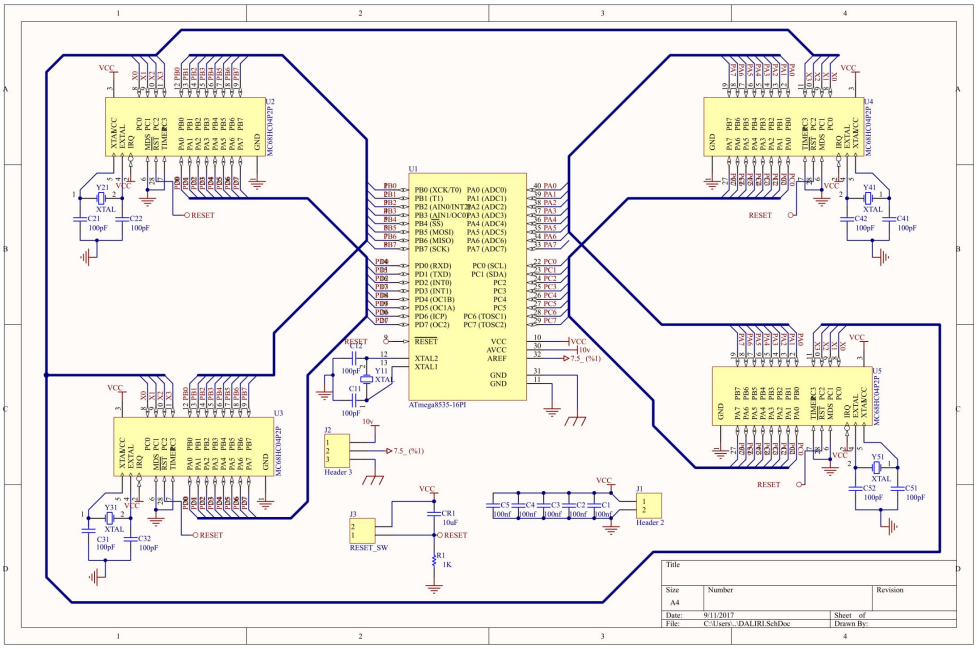
(ه) نمودار نتیجه‌ی کاهش مساحت برد

شکل ۱۶.۴: خروجی مرحله‌ی چهاردهم

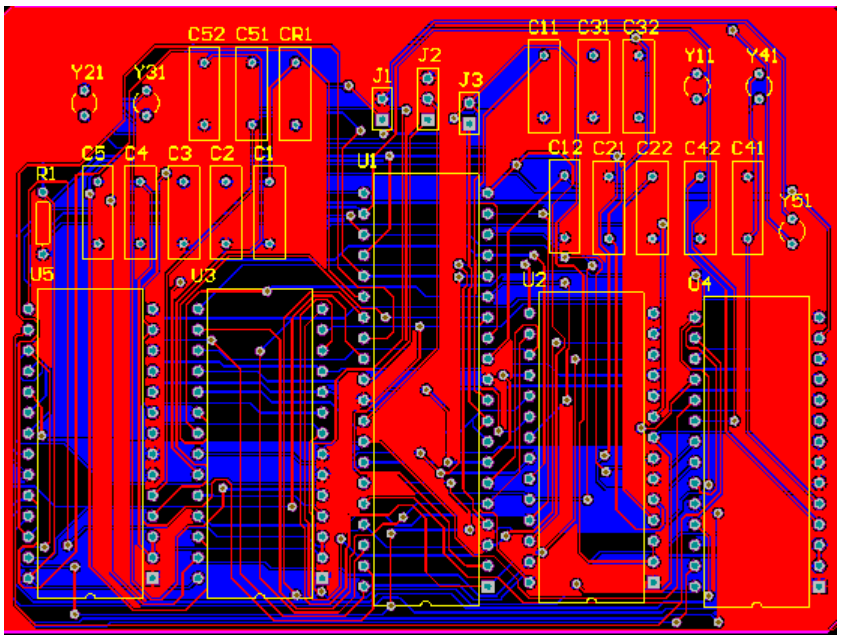
۴.۴ انجام آزمایش فوق بر روی یک برد مداری دیگر به عنوان داده‌ی ورودی

در این قسمت از یک تراشه‌ی دیگر به عنوان ورودی استفاده می‌شود تا بتوان بهتر نتایج را مورد بررسی قرار داد و اطمینان حاصل کرد که الگوریتم در مورد همه‌ی ورودی‌ها پاسخ مطلوبی می‌دهد. طرح

شماتیک و سند PCB مربوط به مدار این برد مداری در شکل (۱۷.۴) مشاهده می‌شود. این برد از ۳۰ قطعه تشکیل شده است و ماتریس وزن آن در ادامه آورده شده است.



(آ) طرح شماتیک

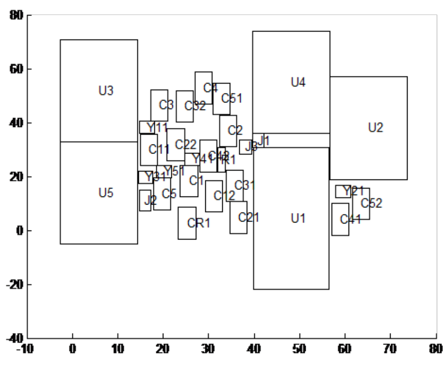


(ب) طرح PCB

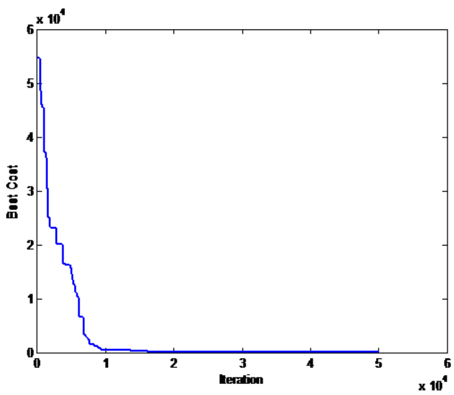
شکل ۱۷.۴: طرح مدار دوم

$$w = \begin{bmatrix}
0&2&2&2&2&1&1&1&1&1&1&1&1&1&1&1&2&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&2&2&2&1&1&1&1&1&1&1&1&1&1&1&2&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&2&2&1&1&1&1&1&1&1&1&1&1&1&2&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&2&1&1&1&1&1&1&1&1&1&1&1&2&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&2&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&0&1&1&0&1&2&1&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&1&1&0&1&2&1&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&0&1&1&0&1&1&2&1&1&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&0&1&1&0&1&1&2&1&1&1&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&0&1&1&0&1&1&1&2&1&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&1&1&0&1&1&1&2&1&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&1&1&0&1&1&1&1&2&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&0&1&1&1&1&2&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&1&1&2&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&1&1&2&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&2&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&3&1&1&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&2&2&2&2&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&19&19&19&19&2&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&23&7&7&0&2&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&7&7&0&0&2&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&23&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

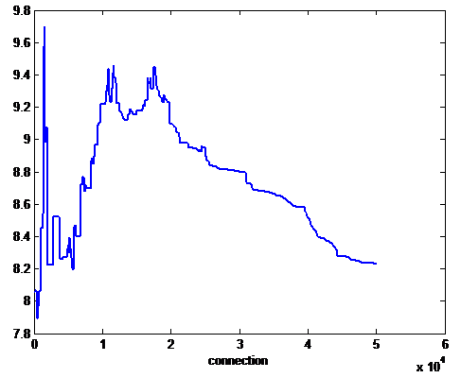
در اینجا نیز همانند آخرین مرحله‌ی بخش قبل، سه قید حذف تداخل، کاهش سیم‌بندی و کاهش مساحت برد به این مدار اعمال می‌شود. همچنین جهش با ضریب $mu = 0.1$ نیز به الگوریتم اعمال شده است. پس از ۱۰۰۰ تکرار، خروجی به شکل (۱۸.۴) قابل‌مشاهده است.



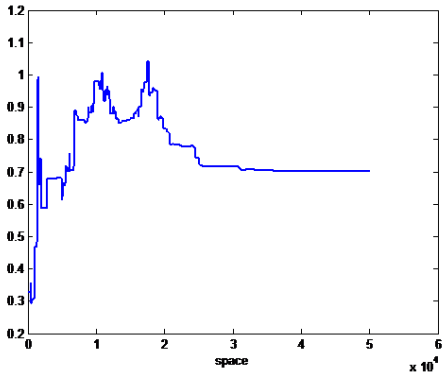
(آ) شکل خروجی



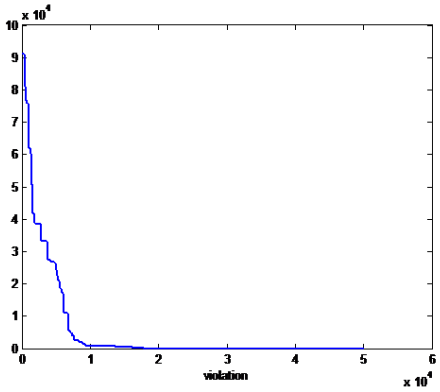
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



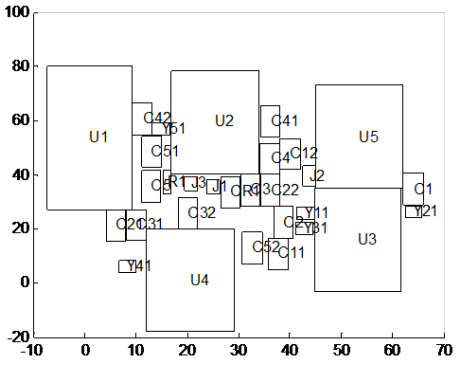
(د) نمودار نتیجه‌ی کاهش مساحت برد



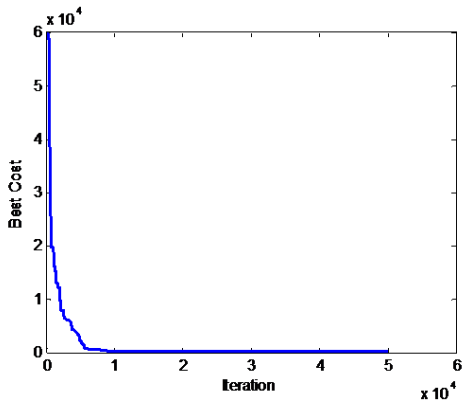
(ه) نمودار نتیجه‌ی حذف تداخل

شکل ۱۸.۴: خروجی مرحله‌ی پانزدهم

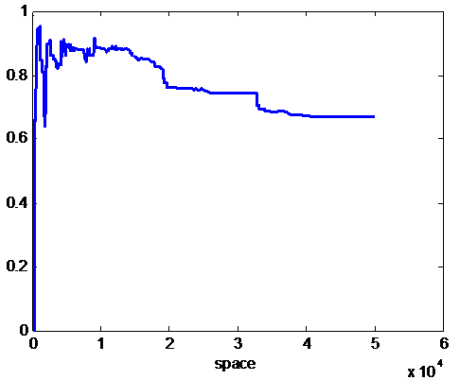
اگر بخواهیم همین آزمایش را با شرایط مشابه و بدون استفاده از جهش انجام دهیم نتیجه به صورت شکل (۱۹.۴) خواهد بود:



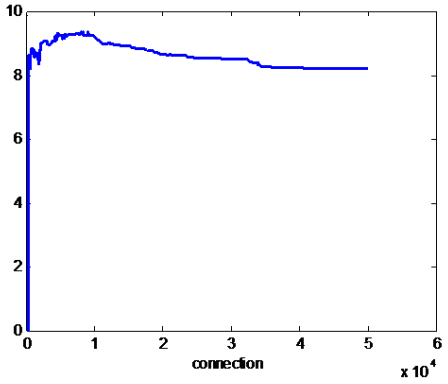
(آ) شکل خروجی



(ب) مقدار Best Cost



(د) نمودار نتیجه‌ی کاهش مساحت برد



(ج) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات

شکل ۱۹.۴: خروجی مرحله‌ی شانزدهم

وقتی از عملگر جهش استفاده می‌شود تنها تفاوتی که در مقایسه با حالت بدون جهش دارد، طوری که نتایج نشان می‌دهد در مرحله‌ی کاهش سیم‌بندی است. در حالت بدون جهش، از یک تکرار به بعد دیگر جابه‌جایی برای کاهش سیم‌بندی انجام نشده است ولی وقتی از جهش استفاده شده است در هر تکرار این جابه‌جایی انجام می‌شود. نتیجه شاید در تکرارهای پایین محسوس نباشد ولی هرچه تعداد تکرار بیشتر باشد این تغییر بیشتر مشخص می‌شود. در شرط مربوط به مساحت نیز هردو الگوریتم از نظر مقدار مشابه هم عمل می‌کنند و تفاوتشان در زمان رسیدن به کوچکترین مساحت است.

۵.۴ جمع‌بندی و نتیجه‌گیری

در این فصل در ابتدا الگوریتم PSO موردنظر پیاده‌سازی شد. توابع هدف و قیدهایی که مدنظر بودند نیز به الگوریتم اعمال شده و در هر مرحله خروجی‌ها را مشاهده کرده و با هم مقایسه شدند. نتیجه‌ای که حاصل می‌شود حاکی از این است که الگوریتم PSO در حل مسئله‌ی جایابی عملکرد بسیار خوبی دارد. با در نظر گرفتن ضرایب توابع هدف و قیدها به طور دقیق و پس از آزمایش‌های زیاد، این الگوریتم می‌تواند همه‌ی این قیدها را ارزیابی و کنترل کرده و سپس پاسخ مطلوبی در راستای رسیدن به تمام اهداف نتیجه بدهد. تداخل را از بین برده، ابعاد برد را کاهش داده و قطعات را طوری قرار داده است که سیم‌بندی تا حد امکان کاهش پیدا کنند و قطعاتی که ارتباط بیشتری دارند نزدیک‌تر به هم قرار بگیرند.

فصل ۵

ارزیابی الگوریتم پیشنهادی

در این بخش برای ارزیابی الگوریتم پیشنهادی ارائه شده، مقایسه‌ای بین الگوریتم مورداستفاده در این پایان‌نامه با الگوریتم‌های دیگری که در سال‌های اخیر در زمینه‌ی جایابی ارائه شده‌اند انجام می‌شود. در [۶] یک الگوریتم ترکیبی ارائه شده است. با اعمال شرایط یکسان، می‌توان نتیجه را مورد مقایسه و بررسی دقیق قرار داد. در مقاله‌ی فوق، ۱۵ دایره با شعاع‌های مختلف را در نظر گرفته است که میزان ارتباط بین هر دو دایره توسط ماتریس وزن که در صفحه‌ی بعد آورده شده است مشخص می‌شود.

مقاله‌ی فوق با استفاده از الگوریتم HGAPSO[۲] و روش خاصی که در تعریف این الگوریتم استفاده شده است، نتایجی را به دست آورده و آن‌ها را با الگوریتم PGA[۳] مقایسه کرده است. این نتایج شامل مساحت اشغال شده توسط اشیا بر حسب میلی‌متر مربع، مجموع وزن کل بین اشیا و زمان اجرای الگوریتم‌ها می‌باشد.

[۲] Hybrid Genetic Algorithm-Particle Swarm Optimization

[۳] Parallel Genetic Algorithm

$$w = \begin{bmatrix} 0 & 0 & 98 & 98 & 98 & 0 & 81 & 0 & 92 & 93 & 45 & 61 & 99 & 84 & 27 \\ 0 & 0 & 34 & 0 & 0 & 0 & 93 & 44 & 0 & 0 & 33 & 60 & 0 & 0 & 56 \\ 0 & 34 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 85 & 0 & 65 & 39 & 0 & 50 \\ 98 & 0 & 0 & 0 & 91 & 50 & 5 & 24 & 73 & 0 & 4 & 0 & 0 & 31 & 23 \\ 98 & 0 & 0 & 91 & 0 & 37 & 0 & 0 & 78 & 95 & 0 & 0 & 73 & 32 & 0 \\ 0 & 0 & 0 & 50 & 37 & 0 & 0 & 35 & 0 & 31 & 0 & 0 & 0 & 48 & 0 \\ 81 & 93 & 0 & 5 & 0 & 0 & 0 & 94 & 33 & 34 & 26 & 61 & 0 & 87 & 87 \\ 0 & 44 & 0 & 24 & 16 & 35 & 94 & 0 & 91 & 0 & 0 & 0 & 59 & 39 & 0 \\ 92 & 0 & 0 & 73 & 78 & 0 & 33 & 91 & 0 & 0 & 30 & 0 & 0 & 0 & 0 \\ 93 & 0 & 85 & 0 & 95 & 31 & 34 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 45 & 33 & 0 & 4 & 0 & 0 & 26 & 0 & 30 & 0 & 0 & 0 & 21 & 35 & 2 \\ 61 & 60 & 65 & 0 & 0 & 0 & 61 & 0 & 0 & 0 & 0 & 0 & 56 & 0 & 43 \\ 99 & 0 & 39 & 0 & 73 & 0 & 0 & 59 & 0 & 0 & 21 & 56 & 0 & 1 & 0 \\ 84 & 0 & 0 & 31 & 32 & 48 & 87 & 39 & 0 & 0 & 35 & 0 & 1 & 0 & 0 \\ 27 & 56 & 50 & 23 & 0 & 0 & 87 & 0 & 0 & 0 & 2 & 43 & 0 & 0 & 0 \end{bmatrix}$$

۱.۵ تعیین تابع تناسب

همان‌طور که گفتیم هدف اول مدنظر ما، چینش قطعات در مکان مناسب به منظور کوچک کردن ابعاد برد و هدف دوم به حداقل رساندن طول سیم مورد استفاده است. برای قسمت اول تابع تناسبی که می‌توان در نظر گرفت این است که مساحت کل فضای برد را تا جایی کوچک کنیم که به مجموع مساحت قطعات نزدیک شود. برای قسمت دوم نیز می‌توان از رابطه‌ی (۱.۵) به عنوان تابع تناسب استفاده کرد که در زیر مشاهده می‌شود.

$$C = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} w_{ij} \qquad (۱.۵)$$

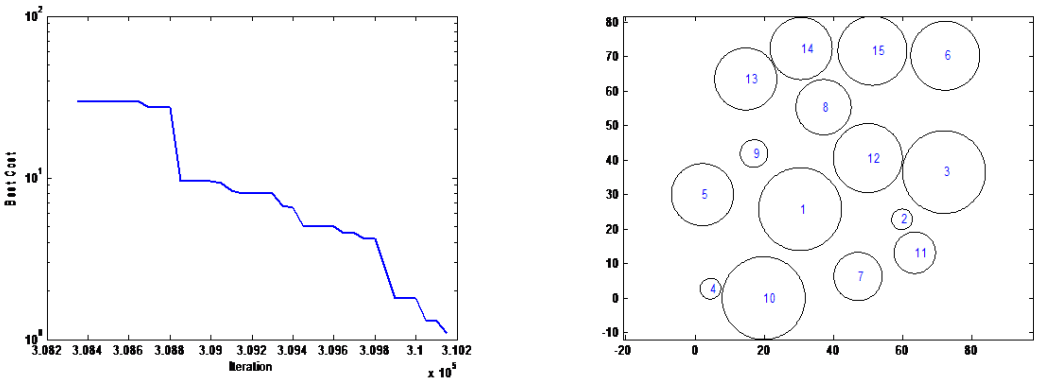
که در این رابطه d فاصله‌ی بین دو قطعه i و j است. w نیز ماتریس وزن است که در هر مسئله‌ی مهندسی متفاوت است. در مسائل مربوط به طراحی چیدمان بردهای مدار چاپی، این ماتریس تعداد سیم‌های ارتباطی بین هر دو قطعه‌ی مجتمع را به ما می‌دهد. حال با استفاده از تابع تناسب در نظر گرفته شده، می‌توان الگوریتم‌ها را مورد مقایسه‌ی دقیق قرار داد. در اینجا مقایسه در زمینه‌های مختلف همچون زمان اجرای الگوریتم‌ها، مقدار مساحت نهایی اشغال شده توسط دایره‌ها و اینکه در تکرار چندم بهترین خروجی دریافت شده است و به بهترین همگرایی دست پیدا می‌کند، و درنهایت در مقدار C که نشانگر مجموع ارتباطات بین قطعات است، صورت می‌گیرد.

۲.۵ نتایج حاصل از بررسی الگوریتم PSO استاندارد

ابتدا نتایج حاصل از الگوریتم PSO بدون ترکیب شدن با الگوریتم دیگری در زیر بررسی می‌شود. استفاده از الگوریتم بهینه سازی ازدحام ذرات نشان می‌دهد که در اکثر مواقع در حل مسائل جایابی، می‌تواند برای اجرای هر کدام از اهداف به تنهایی خوب عمل کند و در زمان کوتاه و با مقدار تناسب نزدیک به صفر همگرا شود. در شکل‌های زیر نتیجه‌ی حاصل از اجرای الگوریتم برای هر کدام از اهداف موردنظر به تنهایی مشاهده می‌شود. دقت شود که فضای مورد بررسی شامل ماتریس وزن، تعداد و شعاع دایره‌ها با فضای مقاله‌ی نام برده شده یکسان در نظر گرفته شده است.

۱.۲.۵ بررسی شرط حذف تداخل

در ابتدا فقط شرط حذف تداخل اعمال شده است که بعد از ۴۰ تکرار مقدار تداخل به صفر رسیده است. در شکل (۱.۵) خروجی‌ها پس از ۱۰۰ تکرار و تعداد اعضای جمعیت ذرات ۵۰، نشان داده شده است. زمان طی شده ۴۷ ثانیه است.

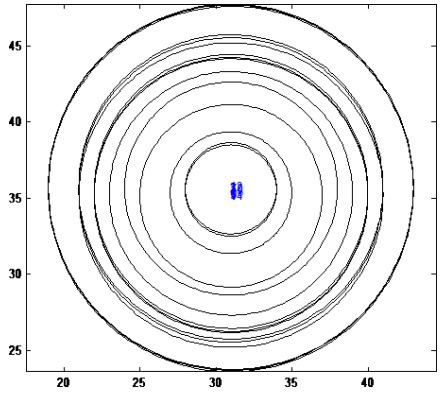


(آ) شکل قرارگیری دایره‌ها (ب) مقدار Best Cost

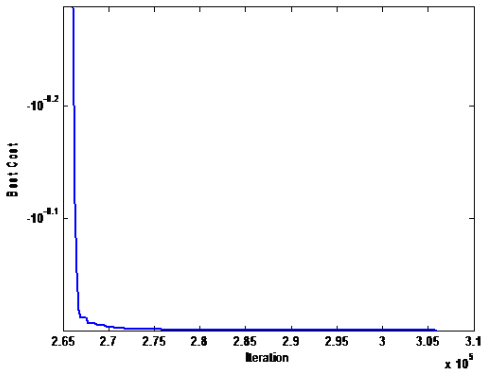
شکل ۱.۵: شکل خروجی حاصل از قید حذف تداخل

۲.۲.۵ اعمال شرط کوچک کردن مساحت برد

در شکل (۲.۵) نیز شرط کوچک کردن مساحت برد اعمال شده است. مشاهده می‌کنیم که بعد از ۲۰۰ تکرار و تعداد اعضای جمعیت ذرات ۲۰۰، اشیا درون بزرگترین دایره قرار گرفته‌اند تا فضای کمتری را اشغال کنند. زمان سپری شده ۱۰۶ ثانیه است.



(آ) شکل قرارگیری دایره‌ها

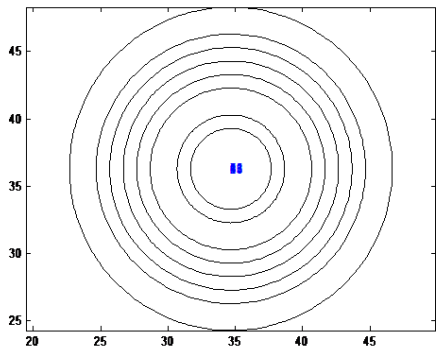


(ب) مقدار Best Cost

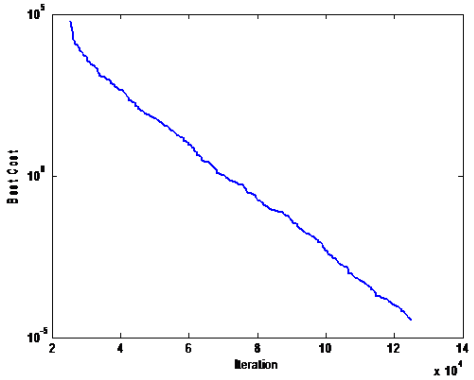
شکل ۲.۵: شکل خروجی حاصل از هدف کوچک کردن ابعاد تراشه

۳.۲.۵ اعمال شرط کاهش سیم‌بندی

در شکل (۳.۵) نیز تنها شرط کم کردن اتصالات و حداقل کردن طول سیم مورد استفاده اعمال شده است. در این حالت مقدار بست کاست به تدریج به صفر میل می‌کند و بعد از ۵۰۰ تکرار و تعداد اعضای جمعیت ذرات ۲۰۰، مقدار بست کاست صفر شده و الگوریتم به خروجی مطلوب دست پیدا کرد. مشاهده می‌شود که الگوریتم فاصله‌ی مرکز قطعات را به صفر رسانده است و به همین دلیل مرکز همه‌ی قطعات در یک نقطه قرار گرفته‌اند. زمان سپری شده برای این آزمایش ۳۶ ثانیه است.



(آ) شکل قرارگیری دایره‌ها



(ب) مقدار Best Cost

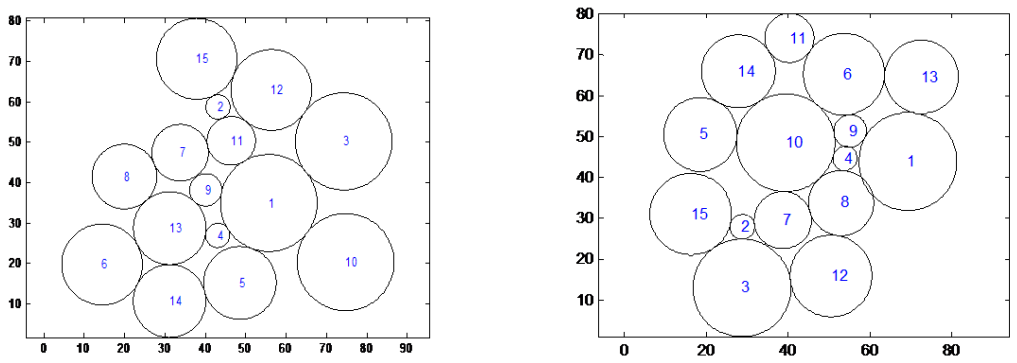
شکل ۳.۵: شکل خروجی حاصل از هدف کاهش سیم‌بندی

۳.۵ بررسی نتیجه‌ی شرط‌های قبل با اضافه شدن جهش

سپس همین قیدها به مسئله داده می‌شود با این تفاوت که در این جا عملگر جهش[1] را که از اجزای الگوریتم ژنتیک است، برای بهبود وضعیت جایابی با الگوریتم ترکیب کرده و به کار گرفته می‌شود. نتایج نشان می‌دهد که می‌توان به طور قطع نتیجه گرفت که وجود عملگر جهش با ضریب $mu = 0.2$ باعث بهبود شرایط در این مرحله شده است.

۴.۵ اعمال قیدها به صورت دو به دو به مسئله و مقایسه‌ی نتایج در دو حالت بدون جهش و با جهش

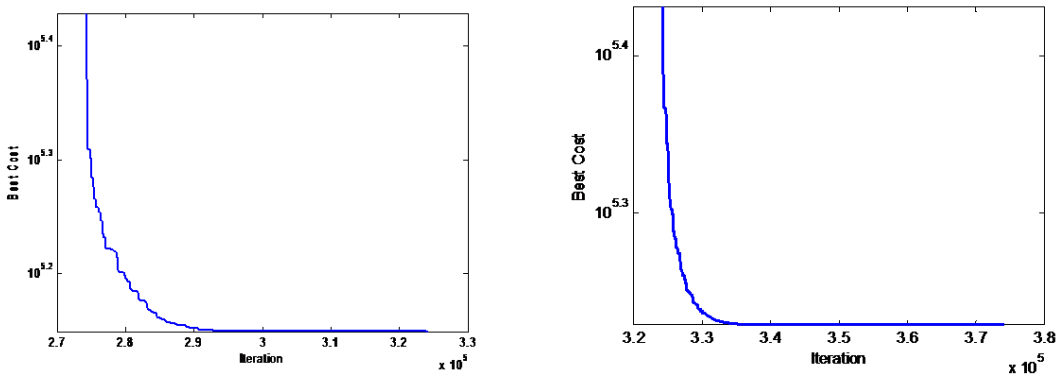
در مرحله‌ی بعد قیدها دو به دو به مسئله داده می‌شود و نتیجه‌ی حاصل از الگوریتم PSO نمایش داده می‌شود. در مرحله‌ی اول قید حذف تداخل با هدف بهینه کردن ارتباط‌ها داده شده و نتیجه را مشاهده می‌کنیم. ضریب این دو هدف به ترتیب ۱۰۰۰ و ۱ در نظر گرفته می‌شود. نتایج نشان می‌دهد که بدون استفاده از عملگر جهش در الگوریتم PSOخروجی مطلوب به دست نمی‌آید. در شکل (۴.۵) نتایج حاصل برای دو حالت بدون عملگر جهش و با عملگر جهش مشاهده می‌شود. مقایسه‌ی این دو نشان می‌دهد که الگوریتم PSO معمولی پس از گذشت چندین تکرار به علت گیر افتادن در بهینه‌ی محلی نمی‌تواند نتایج را بهبود دهد و مثلاً در قسمت حذف تداخل ناتوان شده است و با وجود ضریب نسبتاً بزرگی که به قید مربوط به آن اعمال شده است، نتوانسته به طور کامل تداخل را حذف کند. همچنین شرط مربوط به ارتباطات نیز با وجود عملگر جهش، کاهش ارتباطات به طور چشمگیری رخ داده است. نتایج زیر پس از ۱۰۰۰ تکرار و تعداد جمعیت ذرات ۵۰ به دست آمده است.



(آ) بدون کمک عملگر جهش (ب) با استفاده از جهش

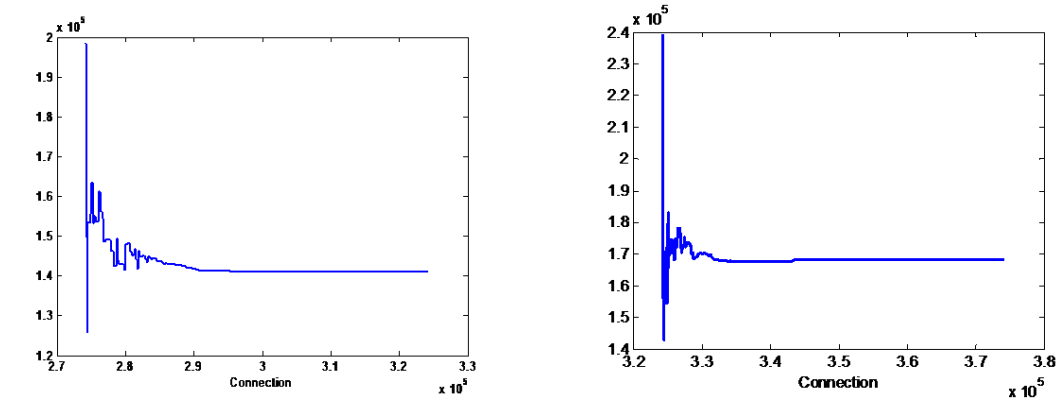
شکل ۴.۵: شکل خروجی حاصل از دو هدف حذف تداخل و کاهش سیم‌بندی

[1] Mutation Operator



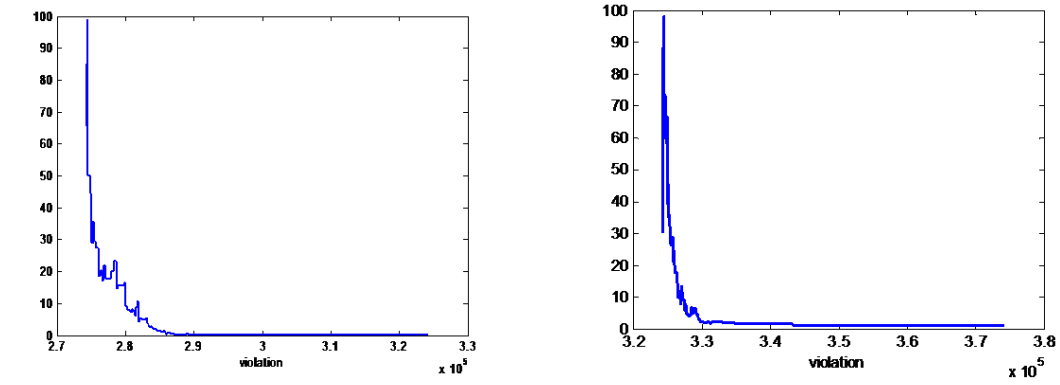
(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۵.۵: مقدار Best Cost حاصل از دو هدف حذف تداخل و کاهش سیم‌بندی



(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۶.۵: نمودار نتیجه‌ی کاهش سیم‌بندی با اجرای الگوریتم با دو هدف حذف تداخل و کاهش سیم‌بندی

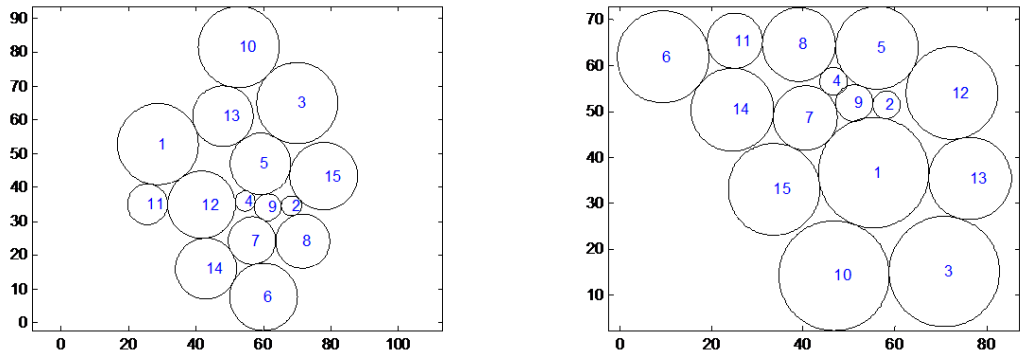


(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۷.۵: نمودار نتیجه‌ی حذف تداخل با اجرای الگوریتم با دو هدف حذف تداخل و کاهش سیم‌بندی

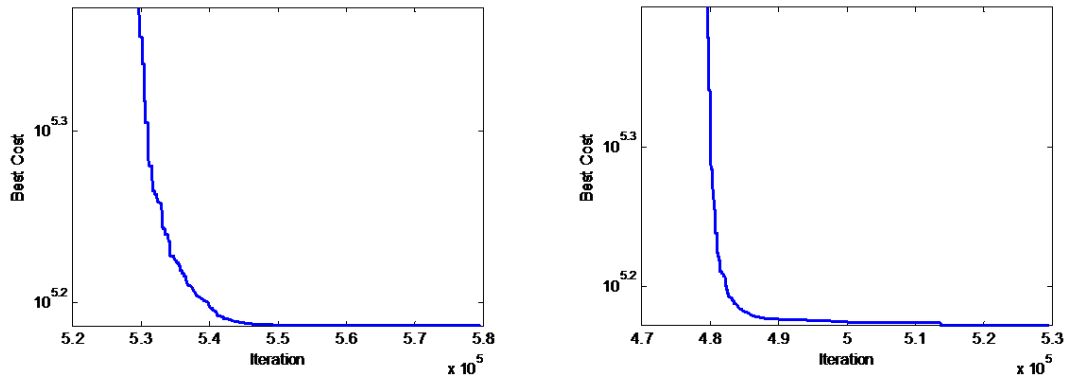
۵.۵ اعمال سه قید به مسئله و مقایسه‌ی نتایج در دو حالت بدون جهش و با جهش

حال پس از اصلاح دستورات مربوط به محاسبه‌ی مساحت دربرگیرنده‌ی قطعات، سه شرطی که داریم را به همین برنامه اعمال کرده و نتایج مربوط به الگوریتم ترکیب شده با جهش و بدون جهش در شکل‌ها نمایش داده می‌شود. زمان سپری شده در انجام آزمایش برای الگوریتم PSO بدون استفاده از جهش ۵۵۶ ثانیه و با استفاده از جهش، ۴۵۰ ثانیه است. نتیجه‌گیری‌ها نشان می‌دهد الگوریتم PSO ترکیبی با جهش بهتر عمل می‌کند. از این رو از این پس در آزمایشات بعدی از الگوریتم پیوندیافته PSO با جهش استفاده می‌شود.



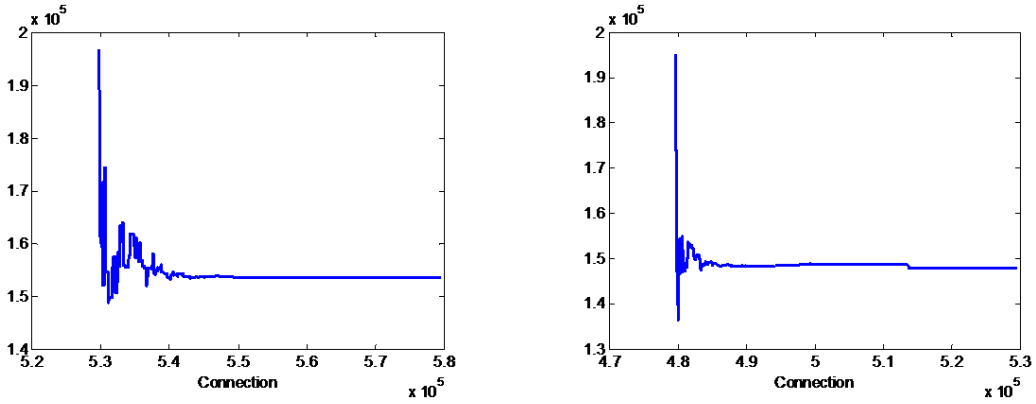
(آ) شکل خروجی بدون کمک عملگر جهش (ب) شکل خروجی با استفاده از جهش

شکل ۸.۵: شکل خروجی حاصل از سه هدف



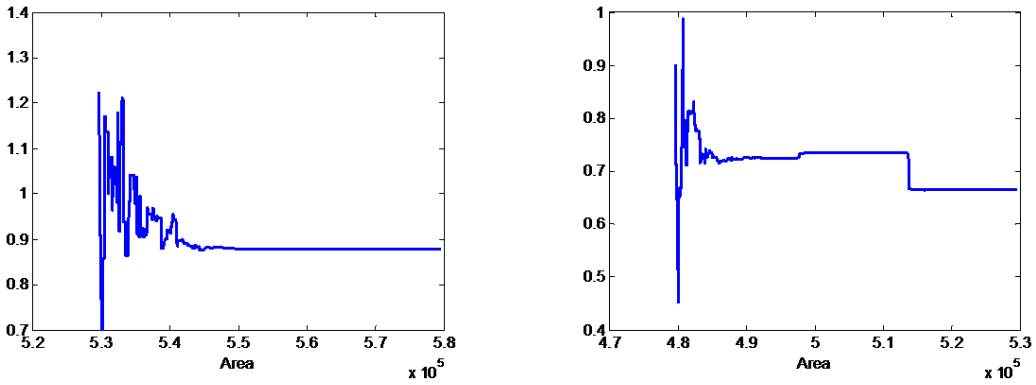
(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۹.۵: نمودار مقدار Best Cost حاصل از اجرای الگوریتم با هر سه هدف



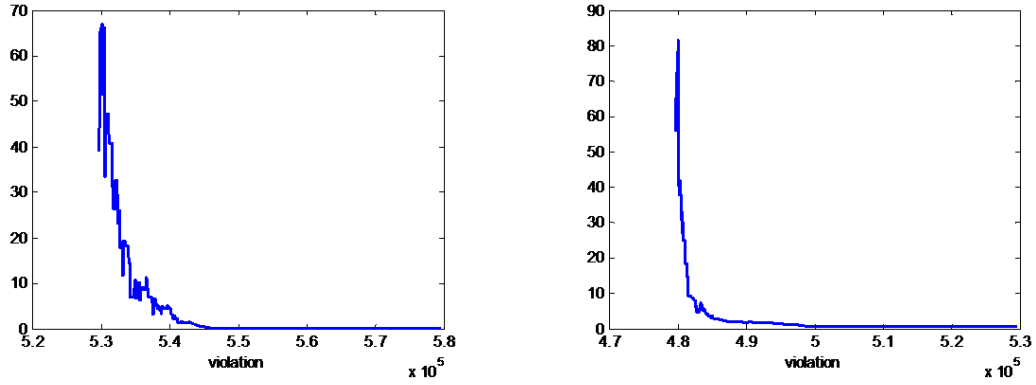
(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۱۰.۵: نمودار نتیجه‌ی کاهش سیم‌بندی حاصل از اجرای الگوریتم با سه هدف



(آ) بدون استفاده از جهش (ب) با استفاده از جهش

شکل ۱۱.۵: نمودار نتیجه‌ی کوچک کردن مساحت حاصل از اجرای الگوریتم با سه هدف



(آ) بدون استفاده از جهش (ب) با استفاده از جهش

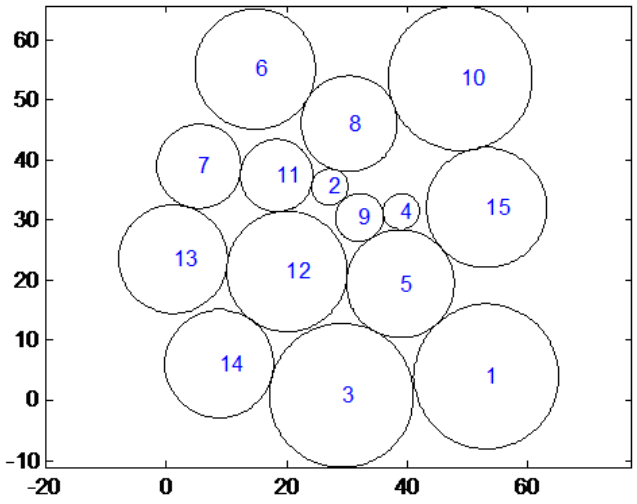
شکل ۱۲.۵: نمودار نتیجه‌ی حذف تداخل حاصل از اجرای الگوریتم با سه هدف

۱.۵.۵ تعیین ضرایب مناسب برای هدف‌ها

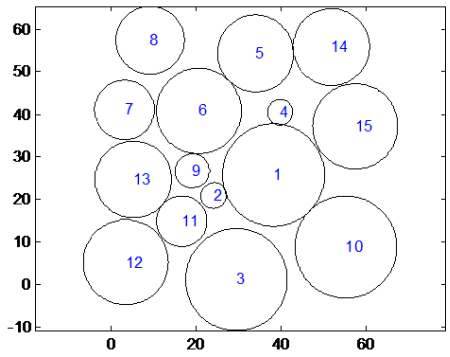
تا به اینجا نتایج مربوط به حذف تداخل قابل‌قبول هستند، اما نتایج مربوط به دو هدف دیگر به علت اینکه ضریب ارزش آن‌ها یک‌هزارم حذف تداخل در نظر گرفته شده است، نتیجه‌ی مطلوبی نیستند و مساحت و فاصله‌ی سیم‌بندی‌ها باید کاهش پیدا کند. در نتیجه تغییرات دیگری در شکل تابع هزینه صورت می‌گیرد. ضریب a_1 باید حذف شود. اولین کاری ک انجام می‌شود این است که برای هرکدام از هدف‌ها ضرایب a_1، a_2 و a_3 در نظر گرفته شود.

پس از به کار گیری روش‌های مختلف برای انتخاب ضرایب مناسب، با نرمالیزه کردن تابع هدف ارتباط‌ها (در اینجا با عدد ۷۰۰۰۰ نرمالیزه می‌شود)، رابطه‌ی بین ضرایب به ثبات می‌رسد و حال با در نظر گرفتن ضرایب به ترتیب ۲.۸ و ۱ و ۰.۲ مشاهده می‌شود که تداخل نیز با اینکه ارزش کمتری به آن اختصاص یافته است به طور کامل از بین می‌رود و نتایج دیگر هم تا حدودی قابل‌قبول هستند. این در حالی است که اگر هدف ارتباط نرمالیزه نشود، تداخل از بین نمی‌رود.

در اینجا دو حالت مختلف مربوط به جهش آزمایش شده است. یکبار تابع مربوط به جهش با ضریب جهش $mu = 0.2$ استفاده شده است و بار دیگر مقدار $pm = 0.5$ در نظر گرفته شده است و نتایج این دو حالت باهم مقایسه می‌شوند. این نتایج پس از ۲۰۰۰ تکرار و با تعداد اعضای جمعیت ذرات $npop = 200$ به دست آمده است. مساحت کل در حالت اول ۵۶۰۰ و زمان سپری شده اجرای الگوریتم ۷۹۵ ثانیه است. در حالی‌که در حالت دوم در شرایط یکسان، مساحت کل ۵۸۰۰ و زمان سپری شده ۸۷۰ ثانیه است.

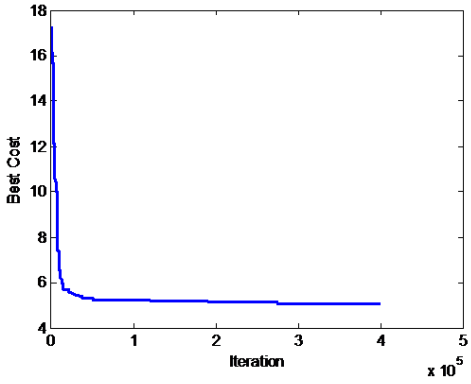


(آ) با عملگر جهش با $mu = 0.2$

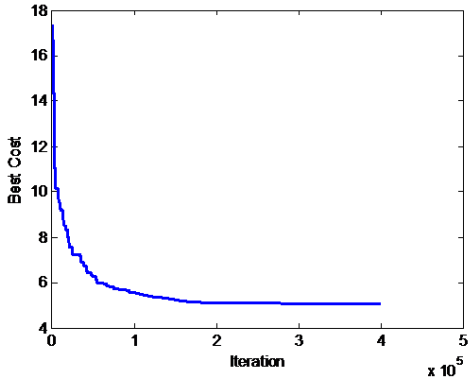


(ب) با عملگر جهش با $pm = 0.5$

شکل ۱۳.۵: شکل خروجی حاصل از سه هدف

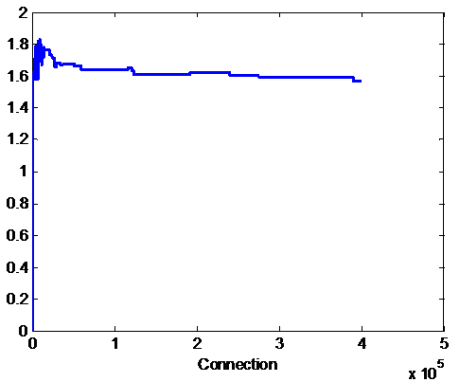


(ب) با عملگر جهش با $pm = 0.5$

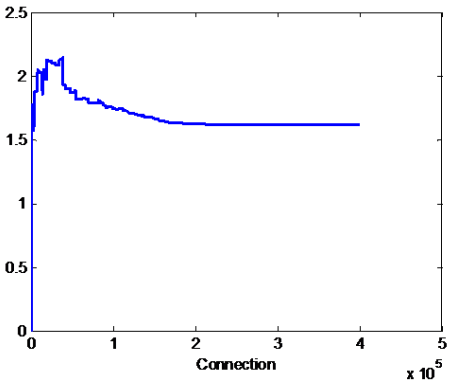


(آ) با عملگر جهش با $mu = 0.2$

شکل ۱۴.۵: نمودار مقدار Best Cost حاصل از اجرای الگوریتم با سه هدف

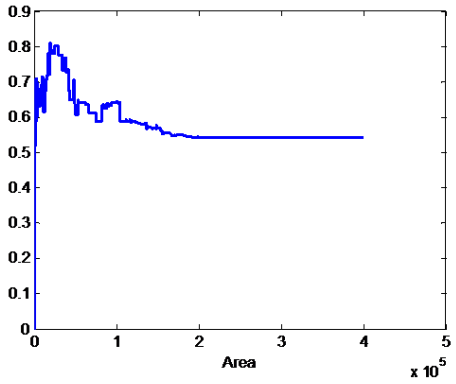


(ب) با عملگر جهش با $pm = 0.5$

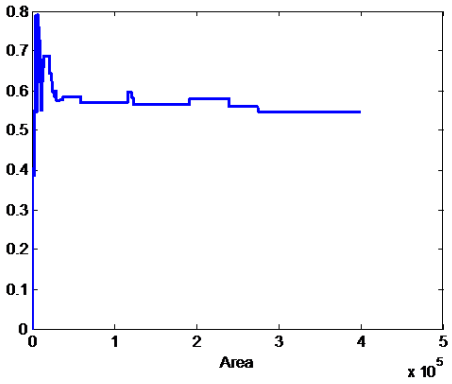


(آ) با عملگر جهش با $mu = 0.2$

شکل ۱۵.۵: نمودار نتیجه‌ی کاهش سیم‌بندی حاصل از اجرای الگوریتم با سه هدف

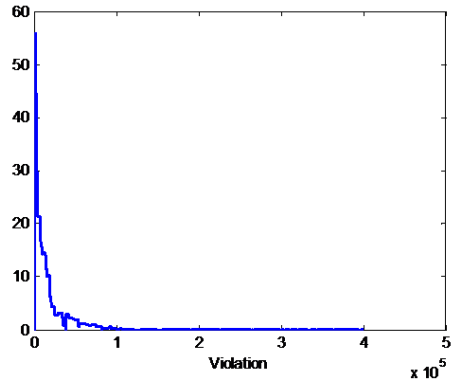


(آ) با عملگر جهش با $mu = 0.2$

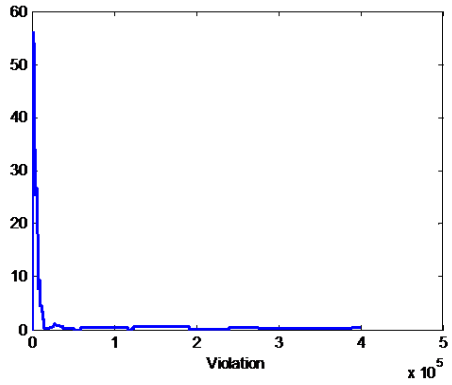


(ب) با عملگر جهش با $pm = 0.5$

شکل ۱۶.۵: نمودار نتیجه‌ی کوچک کردن مساحت حاصل از اجرای الگوریتم با سه هدف



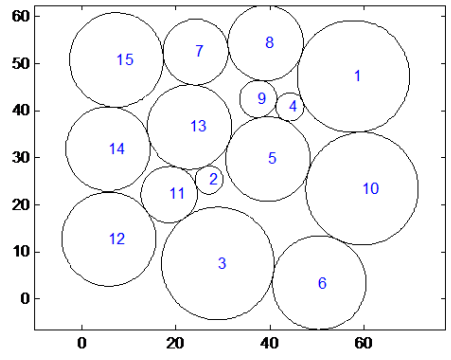
(آ) با عملگر جهش با $mu = 0.2$



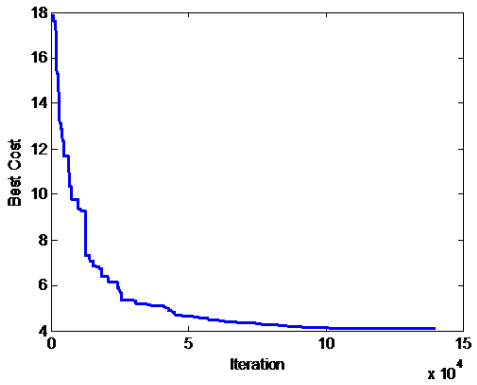
(ب) با عملگر جهش با $pm = 0.5$

شکل ۱۷.۵: نمودار نتیجه‌ی حذف تداخل حاصل از اجرای الگوریتم با سه هدف

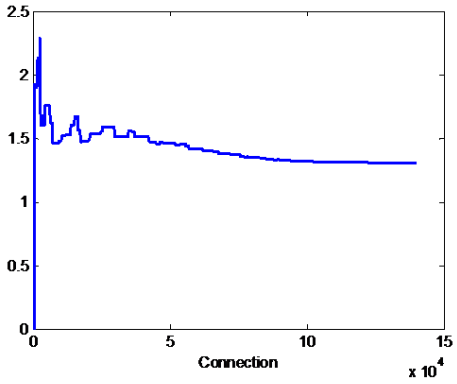
آزمایش فوق این‌بار با $mu = 0.3$ انجام می‌شود. نتیجه به صورت شکل (۱۸.۵) است.



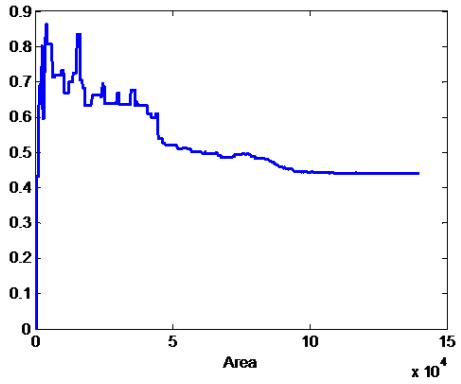
(آ) شکل خروجی



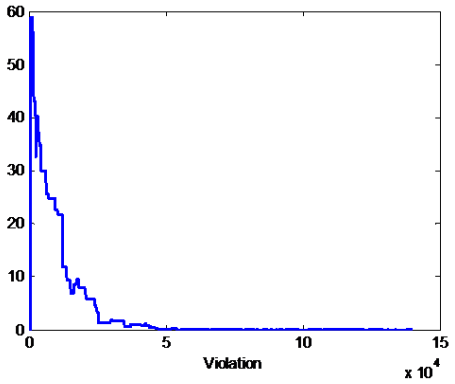
(ب) مقدار Best Cost



(ج) نمودار نتیجه‌ی کاهش سیم‌بندی ارتباطات



(د) نمودار نتیجه‌ی کاهش مساحت برد



(ه) نمودار نتیجه‌ی حذف تداخل

شکل ۱۸.۵: خروجی حاصل از اجرای الگوریتم با جهش با ضریب $mu = 0.3$

۶.۵ مقایسه‌ی نتایج دو الگوریتم

در جدول زیر نتایج حاصل از الگوریتم پیشنهادی این پایان‌نامه که یک الگوریتم PSO جهش‌یافته است و الگوریتم HGAPSO با هم مقایسه می‌شود. (یادآوری می‌شود در جدول زیر، S مساحت کل برد را نشان می‌دهد و واحد آن میلی‌متر مربع می‌باشد. قسمت connection نیز نشان‌دهنده تعداد کل ارتباطات سیمی است که بین قطعات وجود دارد. t نیز زمان اجرای الگوریتم را نشان می‌دهد که این عامل به سرعت پردازش سیستم رایانه‌ای که استفاده شده است بستگی دارد.)

جدول ۱.۵: جدول مقایسه‌ی نتایج

الگوریتم‌ها	S	Connection	t(s)
PSO	۵۲۴۶	۸۰۲۳۰	۵۵۰
HGAPSO	۵۴۰۰	۸۶۹۶۲	۵۱۲

مقایسه‌ی نتایج دو الگوریتم حاکی از این است که الگوریتم PSO ترکیبی با جهش مورد استفاده عملکرد بهتری داشته است. میزان سیم‌بندی و مساحت کل قطعات کاهش محسوسی داشته است.

۷.۵ پیشنهاد برای کار در این زمینه

کارهای دیگری که می‌تواند انجام شود همانطور که به طور مختصر توضیح داده شد، بحث و بررسی در مورد مکان دقیق بعضی از قطعات و یا قطعاتی که باید دقیقاً کنار یکدیگر قرار بگیرند می‌باشد. بعضی قطعات جای مشخص و ثابتی روی برد دارند، می‌توان همه‌ی این قیدها را به عنوان پارامترهای الگوریتم در نظر گرفت و با تعیین ضرایب مناسب برای هدف‌ها خروجی مناسب و مطلوبی را دریافت کرد.

مراجع

[1] Cagan, J., Shimada, K., & Yin, S. (2002). A survey of computational approaches to three-dimensional layout problems. Computer-Aided Design, 34(8), 597-611.

[2] Jankovits, I., Luo, C., Anjos, M. F., & Vannelli, A. (2011). A convex optimisation framework for the unequal-areas facility layout problem. European Journal of Operational Research, 214(2), 199-215.

[3] Qian, A., & Teng, H. (2002). Algorithm of complex layout design problems. Zhongguo Jixie Gongcheng/China Mechanical Engineering, 13(8), 696-699.

[4] Yang, J., & Yang, J. (2011). Intelligence optimization algorithms: a survey. International Journal of Advancements in Computing Technology, 3(4), 144-152.

[5] Sun, Y., Zhang, L., & Gu, X. (2012). A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. Neurocomputing, 98, 76-89.

[6] Zhao, F., Li, G., Du, J., Guo, C., Li, T., Fu, X., & Zhang, R. (2014). A New Hybrid PSO-based Genetic Algorithm and its Application to Lay-out Problems. Open Automation and Control Systems Journal, 6, 813-822.

[7] Arora, S. (2011). Hybrid Algorithm PSO and SA in Achieving Partitioning Optimization for VLSI Applications. International Journal of P2P Network Trends and Technology (IJPTT), 1(2), 1-3.

[8] Shojaee, S., Arjomand, M., & Khatibinia, M. (2013). A hybrid algorithm for sizing and layout optimization of truss structures combining discrete PSO and convex approximation. International Journal of Optimization in Civil Engineering, 3, 57-83.

[9] Kaveh, A., & Nasrollahi, A. (2014). A new probabilistic particle swarm optimization algorithm for size optimization of spatial truss structures. Int. J. Civ. Eng, 12, 1-13.

[10] Michalek, J., Choudhary, R., & Papalambros, P. (2002). Architectural layout design optimization. Engineering optimization, 34(5), 461-484.

[11] Yin, S., & Cagan, J. (2000). An extended pattern search algorithm for three-dimensional component layout. Journal of Mechanical Design, 122(1), 102-108.

[12] Ito, T. (1999). A genetic algorithm approach to piping route path planning. Journal of Intelligent Manufacturing, 10(1), 103-114.

[13] Koide, T., & Wakabayashi, S. I. (1999). A timing-driven floorplanning algorithm with the Elmore delay model for building block layout. Integration, the VLSI Journal, 27(1), 57-76.

[14] Medjdoub, B., & Yannou, B. (2000). Separating topology and geometry in space planning. Computer-aided design, 32(1), 39-61.

[15] Liggett, R. S., & Mitchell, W. J. (1981). Optimal space planning in practice. Computer-Aided Design, 13(5), 277-288.

[16] Sharpe, R., Marksjö, B. S., Mitchell, J. R., & Crawford, J. R. (1985). An interactive model for the layout of buildings. Applied mathematical modelling, 9(3), 207-214.

[17] Jo, J. H., & Gero, J. S. (1998). Space layout planning using an evolutionary approach. Artificial Intelligence in Engineering, 12(3), 149-162.

[18] Jagielski, R., & Gero, J. S. (1997). A genetic programming approach to the space layout planning problem. In CAAD futures, 875, 875-884. Dordrecht: Kluwer.

[19] Thakur, M. K., Kumari, M., & Das, M. (2010). Architectural layout planning using genetic algorithms. In Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, 4, 5-1. IEEE.

[20] Tassopoulos, I. X., & Beligiannis, G. N. (2012). Solving effectively the school timetabling problem using particle swarm optimization. Expert Systems with Applications, 39(5), 6029-6040.

[21] Beni, G., & Wang, J. (1993). Swarm intelligence in cellular robotic systems. In Robots and Biological Systems: Towards a New Bionics? (pp. 703-712). Springer, Berlin, Heidelberg.

[22] J. Kennedy, and R. C. Eberhart, ”Particle swarm optimization,” Proceedings of IEEE International Conference on Neural Networks, 4, 1942-1948, 1995.

[23] Parsopoulos, K. E., & Vrahatis, M. N. (2002). Recent approaches to global optimization problems through particle swarm optimization. Natural computing, 1(2-3), 235-306.

[24] Løvbjerg, M., Rasmussen, T. K., & Krink, T. (2001, July). Hybrid particle swarm optimiser with breeding and subpopulations. In Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation (pp. 469-476). Morgan Kaufmann Publishers Inc.

[25] Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, 5, 4104-4108. IEEE.

[26] Mahfouf, M., Chen, M. Y., & Linkens, D. (2004). Adaptive weighted particle swarm optimisation for multi-objective optimal design of alloy steels. In Parallel problem solving from nature-ppsn viii (762-771). Springer Berlin/Heidelberg.

[27] Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000, September). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In International Conference on Parallel Problem Solving From Nature (849-858). Springer, Berlin, Heidelberg.

[28] Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. IEEE Transactions on evolutionary computation, 8(3), 256-279.

[29] Charles, D. (1925). U.S. Patent No. 1,563,731. Washington, DC: U.S. Patent and Trademark Office.

[30] Sabunin, A. E. (2009). Altium Designer. New solutions in the design of electronic devices. Moscow: Solon-Press, 432.

[31] Abboud, N., Grötschel, M., & Koch, T. (2008). Mathematical methods for physical layout of printed circuit boards: an overview. OR Spectrum, 30(3), 453.

[32] Kumar, R., Leong, W. K., & Heath, R. J. (2006). An Integer Programming Approach to Placement and Routing in Circuit Layout. université du kentucky.

[33] Magnuson, W. G. (1977). A comparison of constructive placement algorithms. In Region Six Conference Record, 1977. IEEE 1977 (28-32). IEEE.

[34] Areibi, S., & Yang, Z. (2004). Effective memetic algorithms for VLSI design= genetic algorithms+ local search+ multi-level clustering. Evolutionary Computation, 12(3), 327-353.

[35] Pannérec, T. (2003). Knowledge-based automatic components placement for single-layer PCB layout. In Knowledge-Based Intelligent Information and Engineering Systems (pp. 669-675). Springer Berlin/Heidelberg.

[36] Sechen, C. (1988). Introduction. In VLSI Placement and Global Routing Using Simulated Annealing (1-30). Springer US.

[37] Sait, S. M., Faheemuddin, M., Minhas, M. R., & Sanaullah, S. (2005, June). Multiobjective VLSI cell placement using distributed genetic algorithm. In Proceedings of the 7th annual conference on Genetic and evolutionary computation (1585-1586). ACM.

[38] Zhang, L., Raut, R., Jiang, Y., Kleine, U., & Kim, Y. (2005). A hybrid evolutionary analogue module placement algorithm for integrated circuit layout designs. International Journal of Circuit Theory and Applications, 33(6), 487-501.

[39] Lazinica, A. (Ed.). (2009). Particle swarm optimization. Kirchengasse: InTech.

[40] Ratanavilisagul, C., & Kruatrachue, A. B. (2014). A modified particle swarm optimization with mutation and reposition. Int J Innov Comput Inform Control, 10(6), 2127-2142.

[۴۱] م. رستمی شهربابکی، و ح. نظام آبادی پور، ''انتخاب ویژگی در طبقه بندی معنایی تصاویر با استفاده از الگوریتم PSO،'' یازدهمین کنفرانس سالانه انجمن مهندسی کامپیوتر ایران، صفحات ۲۷۵-۲۶۹، ۱۳۸۴.

Aabstract

Component placement and layout, is a topic that has been taken into consideration since many years ago. It studies how are objects placed in a logical boundary space under a specific constraints. Theoretically, the layout problems are of the type NP-complete. we are faced with this type of problems in many engineering fields. They are needed in a variety of application fields such as the layout design of spacecraft modules, plant equipments, platforms of marine drilling systems, shipping, vehicle and robots. These problems often affect some design indicators, including reliability and economy, directly. One of the applications of this field is placement and routing on the printed circuit boards. Today, the use of printed circuits in the industry is arising quickly and the dimensions of these boards are getting smaller every day. As a result, it needs to softwares for automated and optimal design of these boards.

In this thesis, using Particle Swarm Optimization Algorithm which that uses social intelligence, objectives such as minimizing the space of board by reducing the wiring and closing of the components are considered. Some additional constraints such as constriants for balance, connectivity, and adjacent components are considered. Consideration of all the constraints is one of the challenges facing this problem. Because with increasing constraints, the probability of getting away from the optimal solution increases and the implementation of the algorithm takes more time. Finding a logical connection between objectives and implementing functions mathematically, is very complicated. The results show that this algorithm works well in solving this problem by considering correct objectives and constraints. Here, experiments such as combining this algorithm with the genetic algorithm are carried out. There is also a comparison between operation of the algorithm with combined algorithm HGAPSO.

keywords: Component placement, Layout design, Social intelligence, Evolutionary algorithms, Particle swarm optimization algorithm, Printed circuit board, Altium designer software.

Shahrood University of Technology

Faculty Of Electrical and Robotics Engineering

MSc Thesis in:

Component Placement And Routing of Printed Circuit Boards Using Particle Swarm Optimization Algorithm

By: Mitra Rafatpour

Supervisor

Dr. Ali Soleimani

July 2017