یک شبکه عصبی فازی ژنتیکی جدید برای حل مسأله فروشنده دوره‌گرد

مهدی سعادتمند طرزجان
دانشکده برق، دانشگاه صنعتی خواجه نصیر الدین طوسی، تهران، ایران
m_saadatmand_tarzjan@yahoo.com

چکیده در این مقاله پاسخ‌های شبکه عصبی CNN-TSP به مسأله فروشنده دوره‌گرد، به کمک یک سیستم فازی ژنتیکی بهبود یافته است. این سیستم بصورت یک تصمیم‌گیرنده فازی در بخش سازنده شبکه عمل کرده و جایگزین تصمیم‌گیرنده رقابتی شده است. پایگاه قواعد این سیستم توسط یک الگوریتم ژنتیکی، با هدف انتخاب مناسب شهرها در بخش سازنده، بهینه گردیده است. نه تنها شبکه پیشنهادی دارای دقتی بیشتر از CNN-TSP است بلکه با افزایش تعداد شهرها، دقت آن افزایش نیز می‌یابد. بعلاوه، زمان همگرایی شبکه پیشنهادی برای توزیع‌هایی با بیش از تقریبا ۲۱۰ شهر از همتای خود کمتر است که این اختلاف، با افزایش تعداد شهرها بیشتر هم می‌شود.

واژه‌های کلیدی *مسأله فروشنده دوره‌گرد، شبکه عصبی CNN-TSP، سیستم‌های فازی ژنتیکی*

۱) مقدمه

مسایل بهینه‌سازی ترکیبی، از قبیل مسأله فروشنده دوره‌گرد (TSP)، از خانواده مسایل NP-Complete هستند [۱]. مدت زمان لازم برای حل این نوع مسائل با افزایش تعداد پارامترها بصورت نمایی افزایش می‌یابد و این درحالی است که با پیشرفت تکنولوژی، نیاز به حل سریعتر و مناسبتر این مسایل بسرعت رو به افزایش است. تعیین مسیر بهینه حرکت مته برای سوراخ کردن صفحه‌های PCB [۲]، تعیین مسیر بهینه انتقال داده در شبکه‌های کامپیوتری [۳] و پردازش تصویر و تشخیص الگو [۴] از جمله زمینه‌هایی هستند که حل TSP برایشان بسیار راه گشاست.

TSP، همانطور که از نامش بر می‌آید، عبارت است از یافتن کوتاهترین مسیر بسته ممکن بین N شهر. روشهایی که تاکنون برای حل این مسأله ارائه شده‌اند، معمولا بر پایه جستجوگرها و آشکارسازهای ایستا یا احتمالاتی هستند که از آنجمله می‌توان به الگوریتمهای کلاسیک جستجوی محلی، بازپخت تطبیقی (SA) [۵]، شبکه‌های عصبی مصنوعی، الگوریتمهای ژنتیکی (GA) [۶]، و سیستم کولونی مورچه‌ها (ACO) [۷]، اشاره کرد. هر یک از این روشها دارای نقاط ضعف و قوت خاص خود است. به عنوان مثال، از روشهای تکاملی مانند GA معمولا به دلیل حجم زیاد محاسبات بیشتر در کاربردهای خارج خط استفاده می‌شود ولی درعوض توانایی بالایی در بهینه‌سازی مسائل غیرخطی و پیچیده ازخود نشان داده‌اند. همچنین با اینکه SA و ACO نسبت به الگوریتمهای تکاملی سریعتر هستند اما معمولا از دقت کمتری برخوردارند [۸،۷]. ما قبلا در [۹] و [۱۰] شبکه جدیدی معرفی کرده، ساختار و خواص آن را بررسی نموده‌ایم. ما شبکه پیشنهادی را با CNN-TSP می‌نامیم. در این مقاله ضمن معرفی اجمالی آن، با تغییر ساختار و جایگزینی یک تصمیم‌گیرنده فازی ژنتیکی به جای تصمیم‌گیرنده رقابتی آن عملکردش را بهبود بخشیده‌ایم.

بخش بعدی به معرفی شبکه عصبی CNN-TSP و الگوریتم آموزش آن اختصاص یافته است. در بخش سوم نحوه

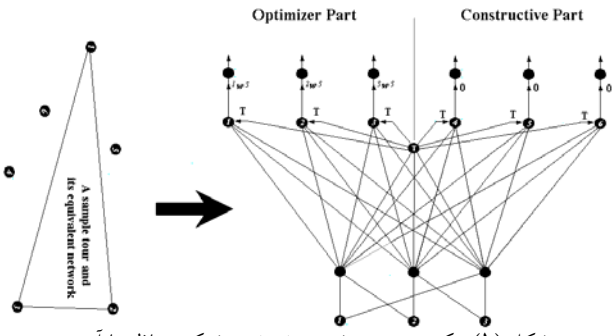
بهینه‌سازی CNN-TSP، با استفاده از سیستمهای فازی شرح داده شده است. بخش چهارم الگوریتم ژنتیکی که برای بهینه‌سازی پایگاه قواعد سیستم فازی مورد استفاده قرار گرفته را شرح می‌دهد. و در نهایت در بخشهای پنجم و ششم به ترتیب شبیه‌سازیها و نتیجه‌گیری ذکر گردیده است.

۲) شبکهٔ عصبی CNN-TSP

شبکه عصبی پیشنهادی دارای دو بخش و هر بخش شامل چهار لایه می‌باشد. خروجی نرونهای هر لایه به شکل بردار است. شکل (۱) این شبکه را برای یک مسیر سه شهری از بین شش شهر نشان می‌دهد. در شکل، ورودی‌ها، خروجی‌ها و مقادیر آستانهٔ نرونها نیز نشان داده شده‌اند. شود.

در شبکه CNN-TSP، نرون‌های لایهٔ اول (Path layer) بیانگر مسیر ارائه شده توسط شبکه در هر مرحله هستند. وزنهای این نرونها به سمت مختصات شهرها میل نمی‌کند، بلکه به شهرها اجازهٔ داده می‌شود تا در آنها جابجا شوند. در لایهٔ دوم (Link layer) هر نرون یک کمان از مسیر را مشخص می‌کند. لایه‌های سوم (Link competitive layer) و چهارم شبکه (Path competitive layer) به دو قسمت تقسیم می‌شوند. نرونهایی که بر روی مسیر قرار گرفته‌اند در بخش اول جای می‌گیرند و بقیه نرونها در بخش دوم قرار داده می‌شوند. وظیفه بخش اول بهینه‌سازی مسیر موجود و وظیفه بخش دوم توسعه مسیر است. در این دو لایه متناظر با هر شهر یک نرون قرار دارد. بعلاوه در لایه سوم یک نرون آستانه نیز وجود دارد (که در شکل (۱) با T نشان داده شده است).

همانطور که گفتیم، CNN-TSP یک شبکه دو قسمتی است. بر همین اساس الگوریتم آموزش آن نیز الگوریتمی دو فازه است. در فاز اول با اضافه شدن شهرهای جدید به مسیر، شبکه توسعه می‌یابد. این مرحله برعهده بخش سازنده شبکه است. در فاز دوم با جابجایی شهرهای موجود در مسیر، مسیر فعلی بهبود می‌یابد که این وظیفه بر عهده فاز بهینه‌ساز شبکه است. فرآیند آموزش چنان طراحی گردیده است که شبکه در هر محله، حتی قبل از پایدارشدن، مسیری معتبر ارائه دهد (مسیری که از تمام شهرهای مسیر تنها یکبار عبور کرده باشد).



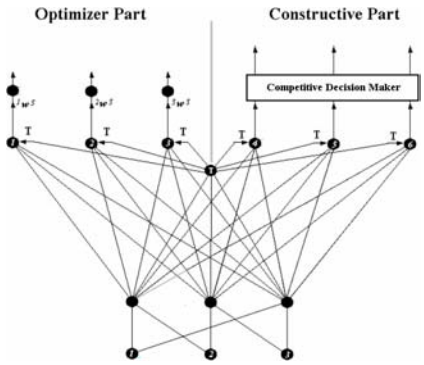
شکل (۱): یک مسیر سه شهری نمونه و شبکه متناظر با آن.

برای مطالعه بیشتر درباره شبکه CNN-TSP و نحوه آموزش آن به [۹] و [۱۰] مراجعه کنید.

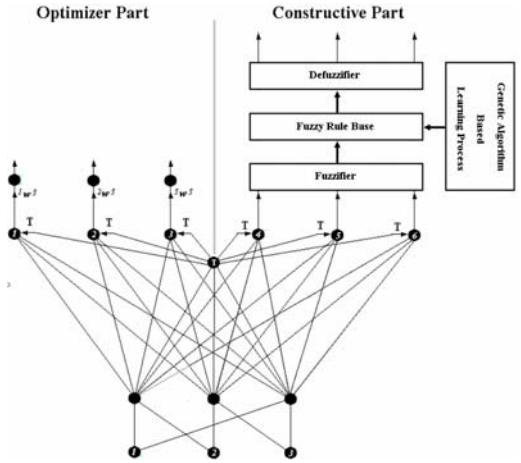
۳) بهینه‌سازی CNN-TSP با استفاده از منطق فازی

مطابق شبیه‌سازیهای که در [۹] ارائه داده‌ایم، CNN-TSP در مقایسه با دیگر شبکه‌های عصبی چون کوهونن پاسخهای قابل قبولی ارائه می‌دهد؛ به طوری که طول پاسخهای ارائه شده توسط آن، به طور متوسط (برای مسیرهای ۵۰ شهری)، تقریبا ۲/۵٪ کوتاهتر از پاسخهای کوهونن است. این در حالی است که زمان همگرایی CNN-TSP بسیار کمتر از آن و در حدود 1/20 است. اما با وجود این تمهیدات هنوز هم شبکه در مینیممهای محلی خود به دام می‌افتد.

با اینکه عملکرد شبکه برای کاربردهای بلادرنگ بسیار مناسب و قابل قبول است، اما چنانچه هدف ما یافتن پاسخهای بهینه‌تری باشد، آنگاه باید برای شبکه تمهیداتی اندیشید. آنچه که بر پاسخهای شبکه CNN-TSP به طور مستقیم اثر می‌گذارد، این است که در هر مرحله از فاز سازنده، کدام شهر در مسیر قرار گیرد. بدیهی است که چنانچه این ترتیب به بهترین حالت ممکن باشد آنگاه شبکه قادر به یافتن بهترین پاسخ نیز خواهد بود. یک روش برای یافتن آرایش مذکور، استفاده از الگوریتمهای ژنتیکی است، اما از این روش نمی‌توان در کاربردهای بلادرنگ استفاده کرد. ما برای این منظور از منطق فازی سود جسته‌ایم. علیرغم تلاش فراوان ما برای یافتن منابع و مراجعی علمی، در زمینه کاربرد فازی در شبکه‌های عصبی به منظور حل مسأله فروشنده دوره‌گرد، تنها دو مقاله علمی بدست آمد. هر دوی مقاله مذکور به مباحثی درباره تنظیم پارامترهای شبکه عصبی هاپفیلد همچون شکل توابع



شکل (۲): شبکه عصبی پیشنهادی از دیدگاهی دیگر (توضیحات لازم در متن مقاله ذکر شده است).

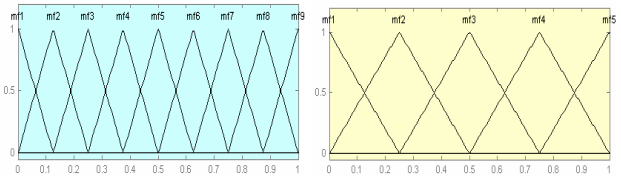


شکل (۳): تلفیق شبکه عصبی پیشنهادی و تصمیم‌گیرنده‌ای فازی که توسط یک الگوریتم ژنتیکی بهینه گردیده است.

عضویت [۱۱] و یا دینامیک آن [۱۲] می‌پرداختند که هیچ کدام برای CNN-TSP قابل استفاده نبود.

برای بیان چگونگی استفاده از منطق فازی در بهینه‌سازی پاسخهای شبکه پیشنهادی، قبل از هر چیز باید از دیدگاهی نو به شبکه نگریست. همانطور که قبلا شرح دادیم، نرونهایی که بر روی مسیر قرار ندارند در لایهای سوم و چهارم، بخش سازنده شبکه را تشکیل می‌دهند. در این دو لایه، طی یک فرآیند رقابتی –در هر مرحله– تعیین می‌شود که کدام نرون باید در کجای مسیر قرار بگیرد. اینکه کدام نرون باید در مسیر قرار گیرد، در لایه چهارم تعیین می‌شود. عاملی که در این انتخاب، نقش اصلی دارد همان افزایش طولی است که نرون با قرار گرفتن در مسیر ایجاد می‌کند. از این دیدگاه می‌توان به لایه چهارم شبکه (در بخش سازنده) به عنوان یک تصمیم گیرنده رقابتی نگریست. این موضوع در شکل (۲) نشان داده شده است. بدیهی است که تصمیم‌گیری تنها بر اساس افزایش طول نمی‌تواند به نتایج بهینه‌ای منجر شود. ما در اینجا قصد داریم به جای این تصمیم‌گیرنده از همتای فازی‌اش استفاده کنیم. در فرآیند تصمیم‌گیری برای انتخاب شهرهای جدید، نکته مهم این است که چه اطلاعات دیگری می‌تواند در بهبود تصمیم‌گیریها مؤثر باشد. مشاهدات تجربی نشان داد که طول کمان برنده برای هر شهر در لایه سوم می‌تواند نقش بسزایی در تصمیم‌گیری مناسب داشته باشد. بنابراین ورودی‌های سیستم فازی برای هر شهر عبارتند از افزایش طول ناشی از افزودن آن به مسیر که توسط نرون‌های لایه سوم محاسبه می‌شود و طول کمانِ برنده، در نرون متناظر با شهر در لایه سوم، که توسط نرون آستانه محاسبه می‌شود (برای مشاهده روابط ریاضی نرون‌ها به [۱۰] مراجعه کنید).

اکنون مسأله این است که سیستم فازی مورد نظرمان چگونه باید طراحی شود. اصولا یکی از معایب عمده سیستمهای فازی این است که این سیستمها قادر به یادگیری نیستند اما نیازمند به پایگاه دانشی هستند که باید بر اساس دانسته‌های فردی خبره ایجاد گردد [۱۳]. ما برای این منظور از الگوریتمهای ژنتیکی سود جسته‌ایم؛ به این ترتیب که پس از انتخاب توابع عضویت ورودی و خروجی از یک الگوریتم ژنتیکی برای طراحی و بهینه‌سازی پایگاه قواعد استفاده شده است. در شکل (۳) نحوه پیاده‌سازی سیستم مذکور و ارتباط بخشهای مختلف آن با یکدیگر نشان داده شده است. همانطور که می‌بینید خروجی نرونهای لایه سوم پس از نرمال شدن و انتقال به بازه *[1,0]* وارد فازی کننده شده، از موتور استنتاج فازی گذشته و در نهایت حاصل استنتاج فازی نتایج در خروجی غیر فازی می‌شود. در هر مرحله از بین شهرهای خارج مسیر، شهری در مسیر قرار می‌گیرد که بیشترین خروجی فازی را دارد. دو ورودی سیستم فازی مطابق شکل (a-۴) دارای ۴ تابع عضویت و خروجی آن مطابق شکل (b-۴) دارای ۹ تابع است. توابع عضویت جهت سادگی در محاسبات به صورت مثلثی انتخاب گردیده‌اند. شبکه عصبی–فازی پیشنهادیمان را FNN-TSP (Fuzzy Neural Net to solve TSP (FNN-TSP)) می‌نامیم. در بخش بعد جزئیات الگوریتم ژنتیکی مذکور شرح داده شده است.



(b) (a)

شکل (۴): (a) توابع عضویت ورودیها و (b) توابع عضویت خروجی سیستم فازی.

۴) بهینه‌سازی پایگاه قواعد با استفاده الگوریتم ژنتیکی

الگوریتمهای ژنتیکی خانواده‌ای از مدلهای محاسباتی هستند که از تکامل موجودات (مسأله انتخاب طبیعی در طبیعت) الهام گرفته شده‌اند [۱۴]. انتخاب طبیعی یک فرایند زیستی است که بر اساس آن در یک جامعهٔ رقابتی، احتمال برنده شدن افراد قویتر بیشتر است.

همانطور که می‌دانید هر الگوریتم ژنتیکی از بخشهای رمزنگاری، انتخاب والدین، برش و جهش تشکیل شده است [۱۴] [۱۵]. هر کروموزوم باید دربرگیرنده مجموعه قواعد سیستم فازی باشد تا شبکه بر اساس آن، در هر مرحله شهری مناسب را انتخاب و در مسیر قرار دهد. به‌طورکلی تنظیم جدول قواعد با الگوریتمهای ژنتیکی به سه روش صورت می‌گیرد: روش میشیگان، روش پیتزبرگ و روش آموزش قواعد با تکرار. در روش میشیگان که در سال ۱۹۷۸ ارائه شد، هر قاعده یک کروموزوم در نظر گرفته می‌شود و جمعیت از تعدادی قاعده تشکیل می‌شود. در روش پیتزبرگ مجموعه قواعد به عنوان یک کروموزوم در نظر گرفته می‌شود. در روش سوم نیز کروموزومها شامل دسته قواعدی می‌باشند که هر قاعده با روش تکرار و اجرای زیاد الگوریتم ژنتیکی وارد کروموزوم می‌شود [۱۶]. در این مقاله از روش پیتزبرگ استفاده شده است. با توجه به اینکه هر ورودی دارای ۵ تابع عضویت است، پس مناسب است که هر کروموزوم شامل بیش از ۲۵ (۵×۵) قانون باشد که ما عدد ۲۷ را انتخاب نمودیم. قانونها ژنهای کروموزومها را تشکیل می‌دهند. هر قانون را می‌توان به شکل کلی ذیل در نظر گرفت:

اگر $^{i}v^{3}$ *عضو* $mf(j)$ *و (یا)* $^{i}p^{3}$ *عضو* $mf(i)$ *باشد، آنگاه خروجی عضو* $mf(k)$ *شود و ارزش این قانون* w *است.*

که mf ها توابع عضویت مربوط به ورودی‌ها و خروجی را مشخص می‌کنند (به شکل (۴) مراجعه کنید) و w ضریبی است بین صفر و یک که ارزش و اهمیت یک قانون را در برابر سایر قوانین تعیین می‌کند. اثر این ضریب در محاسبات، در معادله (۱) نشان داده شده است (سایر توضیحات در ادامه ذکر شده است). بنابراین هر ژن شامل پنج بخش مجزا به صورت « i j k w c» است که به ترتیب تابع عضویتهای ورودی اول (افزایش طول)، ورودی دوم (طول کمان برنده)، خروجی (ارزش شهر)، وزن قانون و نحوه ترکیب دو بخش مقدم قانون را تعیین می‌کند. i و j به ترتیب اعدادی صحیح از صفر تا پنج می‌توانند باشند که عدد صفر نشان دهنده آن است که ورودی مربوطه در آن قانون نقشی ندارد. بدیهی است که i و j نمی‌توانند بطور همزمان صفر باشند. k عددی صحیح از یک تا نه است. w همانطور که گفته شد عددی حقیقی بین صفر و یک است و در نهایت c می‌تواند یک یا دو باشد که یک به معنی ترکیب دو بخش اولیه قانون با یک t-norm و دو به معنی ترکیب با یک s-norm است. به این ترتیب هر کروموزوم عبارت است از آرایه‌ای با ۲۵ سطر و ۵ ستون که هر سطر بیانگر یک قانون یا به عبارت دیگر یک ژن کروموزوم است. به منظور کمینه کردن حجم محاسبات از سیستم فازی ممدانی با موتور استنتاج ضرب، فازی‌ساز منفرد، غیر فازی‌ساز میانگین مراکز، t-norm ضرب و s-norm جمع جبری استفاده گردید. در رابطه ذیل معادله کلی سیستم فازی نشان داده شده است [۱۷] که در آن w^l وزن، $\bar{y}^l$ مرکز تابع عضویت خروجی و mf_i^l تابع عضویت ورودی i، در قانون l هستند. همانطور که می‌بینید در حقیقت وظیفه ضریب w^l تنظیم مرکز تابع عضویت خروجی برای هر قانون است.

$$y=\sum_{l=1}^{25} w^l \times \bar{y}^l \times \left(\prod_{i=1}^{2} mf_i^l(x_i)\right) \Big/ \sum_{l=1}^{25}\left(\prod_{i=1}^{2} mf_i^l(x_i)\right) \qquad (۱)$$

عملگر برش مورد استفاده بشرح ذیل است. ابتدا عددی صحیح بین ۱ تا ۲۵ انتخاب می‌شود. سپس از هر پدر به همان تعداد قانون انتخاب شده و قانونهای هر ولی با دیگری دو بدو عوض می‌شوند. قبل از تعویض قوانین با یکدیگر، بر روی عنصر چهارم از قانونها (که همان وزن آنها است) عملگر برش یک نقطه‌ای حقیقی اعمال می‌شود [۱۴].

از عملگر جهش به منظور فرار از مینیممهای محلی استفاده می‌شود. به این ترتیب که قبل از شروع فرآیند تکامل عددی به عنوان احتمال جهش در نظر گرفته می‌شود. سپس در هنگام تولید نسل جدید، پس از اعمال عملگر برش، برای هر ژن کروموزومهای فرزند، عددی تصادفی تولید می‌شود. چنانچه

این عدد کوچکتر از احتمال جهش بود، بجای ژن مذکور، ژنی که بصورت تصادفی بوجود آمده است قرار می‌گیرد. ما مقدار ۰/۰۱ را برای احتمال جهش انتخاب نمودیم. والدین نیز در هر تولید نسل از روش "مسابقه (Tournament)" [۱۸] انتخاب می‌شوند.

در الگوریتم ژنتیکی پیشنهادی مشابه با Genitor [۱۹]، در هر تولید نسل، از میان جمعیت موجود تنها دو کروموزوم به عنوان ولی انتخاب می‌شوند و بنابراین تنها دو کروموزوم فرزند تولید می‌گردد. جمعیت جدید با جایگذاری یکی از فرزندان به جای بدترین کروموزوم جمعیت قبلی و فرزند دیگر بجای بدترین کروموزوم جمعیت مسابقه (که قبلا برای انتخاب والدین ایجاد شده است) بوجود می‌آید.

برای تعیین کیفیت پایگاه قوانین ارائه شده توسط هر کروموزوم، از ۲۰ توزیع ۱۰۰ شهری استفاده می‌گردد. روش کار چنین است که پاسخ شبکه عصبی با پایگاه قواعدی که کروموزوم مورد نظرمان ارائه می‌دهد، برای هر توزیع بدست می‌آید سپس از مقایسه پاسخهای بدست آمده با پاسخهای CNN-TSP ارزش کروموزوم طبق معادله ذیل محاسبه می‌گردد.

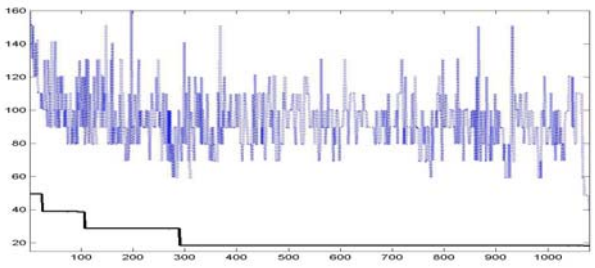
$$e=\frac{1}{20}\sum_{i=1}^{20}|FNN_i-CNN_i|+10\times\sum_{i=1}^{20}U(FNN_i-CNN_i) \qquad (۲)$$

که FNN_i و CNN_i به ترتیب پاسخ شبکه فازی و CNN-TSP به توزیع iام و $U(.)$ تابع پله واحد است. همانطور که می‌بینید تابع هزینه دارای دو بخش است. وظیفه بخش اول که ارزش کمتری دارد، کمینه کردن میانگین قدرمطلق خطاها و وظیفه جمله دوم جریمه کردن کروموزوم به ازای هر پاسخ ضعیفتر از CNN-TSP است. بدیهی است که تعداد ۲۰ توزیع برای بررسی ارزش کروموزومها بسیار کم است. ما برای پرهیز از طولانی شدن بیش از حد فرآیند تکامل مجبور به این انتخاب شده‌ایم. اما با این تعداد کم نمونه آموزشی، مشکل بزرگ دیگری رخ می‌نماید و آن کاهش قدرت فازی در تعمیم تصمیم‌گیریهایش برای توزیعهایی متفاوت با توزیعهای آموزشی‌اش است. به‌طوری‌که طی شبیه‌سازیهای انجام شده گاه خطای متوسط پایگاه داده به توزیعهای آموزشی به ۱/۸٪- می‌رسید، در حالی خطای همان پایگاه به سایر توزیعها در حین تست بیش از ۱٪ بود. با مطالعه تغییرات خطای کروموزومها در حین تکامل دریافتیم که معمولا کروموزومها پایگاه قواعد را چنان تنظیم می‌کنند که به تعدادی از توزیعهای آموزشی پاسخهای بسیار خوبی ارائه می‌دهند در حالی که پاسخ آنها به بقیه همچنان ضعیف باقی می‌ماند. وظیفه جمله دوم در معادله (۲) مبارزه با این عملکرد نامناسب الگوریتم تکاملی است. این جمله سبب می‌شود تا کروموزومها تا آنجا که ممکن است پاسخهایی مناسب به همه توزیعهای آموزشی ارائه دهند و سپس در مرحله بعد به بهینه‌سازی پاسخها بپردازند.

مسأله دیگر از بین رفتن پراکندگی جمعیت الگوریتم ژنتیکی قبل از رسیدن به پاسخ بهینه بود. درتوضیح این پدیده باید گفت که در فرآیند تکامل بسیار اتفاق می‌افتد که کروموزومهایی بسیار مشابه و یا حتی تکراری تولید می‌شوند. حال چنانچه به طور متناوب این اتفاق بیافتد و کروموزومهای حاصل در جمعیت قرار گیرند، بسرعت پراکندگی جمعیت از بین خواهد رفت. از طرف دیگر به علت محدود بودن منابعی نظیر حافظه و زمان محاسباتی، نمی‌توانیم اندازه جمعیت را بیش از حد معینی انتخاب کنیم. بنابراین به مکانیزمی برای جلوگیری از کاهش پراکندگی جمعیت موجود نیازمندیم. برای رفع این اشکال، تنها کروموزومهایی را وارد جمعیت می‌کنیم که پاسخ تابع هزینه به آنها با پاسخ دیگر کروموزومها متفاوت باشد. مشاهدات تجربی نشان داد که معمولا در فرآیند تصمیم‌گیری نرون برنده یکی از ۲۰ شهری است که کمترین v^3 را دارا هستند. بنابراین جهت کاهش حجم محاسبات و همچنین افزایش کارایی الگوریتم ژنتیکی تنها این نرونها جهت تصمیم‌گیری وارد سیستم فازی شدند.

۵) شبیه‌سازی

ما الگوریتم ژنتیکی پیشنهادیمان را شبیه‌سازی نمودیم. اندازه جمعیت الگوریتم ژنتیکی ۵۰ و اندازه جمعیت مسابقه ۷ انتخاب گردید. در شکل (۵) منحنی تغییرات مقدار تابع هزینه



شکل (۵): منحنیهای تغییرات مقدار تابع هزینه الگوریتم ژنتیکی برای بهترین (پایین) و بدترین (بالا) کروموزوم جمعیت در هر تولید نسل.

برای بهترین و بدترین کروموزوم جمعیت نشان داده شده است. همانطورکه در شکل می‌بینید FNN-TSP به ازای بهترین پایگاه قواعد، تنها به دو توزیع از توزیعهای آموزشی پاسخی ضعیفتر از CNN-TSP ارائه داده است. خطای متوسط پایگاه بهینه به نمونه‌های آموزشی ۱/۷۰٪- است که مقدار قابل قبولی است.

برای بررسی عملکرد FNN-TSP، پاسخهای آنرا برای ۱۰۰ توزیع ۵۰ شهری، ۱۰۰ توزیع ۶۰ شهری، ... و ۱۰۰ توزیع ۲۵۰ شهری با CNN-TSP مقایسه نمودیم. از آنجا که طول مسیرها، وابسته به توزیع شهرهاست، لازم است که قبل از میانگین‌گیری، آنها را نرمال کرد. در هر مرحله برای هر توزیع دو مسیر بدست می‌آید، یکی توسط FNN-TSP و دیگری CNN-TSP که یک مجموعه دوتایی را تشکیل می‌دهند. برای نرمال کردن طول این مسیرها، در هر مجموعه از بین دو مسیر بدست آمده، کوتاهترین مسیر را انتخاب کرده و طول هر دو مسیر را بر طول آن تقسیم کرده در صد ضرب می‌کنیم. در پایان از طول نرمال شده همه مسیرهای تولید شده توسط شبکه‌ها (هر شبکه به تنهایی) میانگین می‌گیریم. بدیهی است که اعداد حاصل معیاری هستند از طول مسیرهای تولید شده توسط هر یک از شبکه‌ها نسبت به دیگری. میانگین‌گیری برای مسیرهای ۵۰ شهری، ۶۰ شهری، ... و ۲۵۰ شهری به طور جداگانه انجام شدکه نتایج در شکل (۶-a) نشان داده شده است. همانطور که در این شکل می‌بینید، FNN-TSP در تمام موارد پاسخهای بهتری داده است که این نشان دهنده کارایی سیستم فازی طراحی شده است. بعلاوه بوضوح مشخص است که با افزایش تعداد شهرها بر دقت FNN-TSP در مقایسه CNN-TSP افزوده می‌شود. به‌طوری‌که بهبود ۰/۶۱٪- برای توزیعهای ۵۰ شهری، در توزیعهای ۲۵۰ شهری به حدود ۰/۸۴٪- می‌رسد. اما این تنها بخشی از چیزی است که بدست آورده‌ایم. ما علاوه بر طول متوسط مسیرها، میانگین زمان همگرایی و تعداد تکرارهای شبکه در فاز بهینه‌سازی را نیز برای هر توزیع محاسبه کرده‌ایم که نتایج به ترتیب در شکل‌های (۶-b) و (۶-c) نشان داده شده است. همانطور که در این شکل‌ها می‌بینید، زمان متوسط همگرایی FNN-TSP تا توزیع ۲۱۰ شهری بیش از CNN-TSP است اما از این نقطه به بعد این زمان کمتر از زمان همگرایی رقیب خود می‌شود. دلیل این امر را بوضوح در شکل (۶-c) می‌توان یافت. همانطور که می‌بینید از این نقطه به بعد تعداد تکرارهای FNN-TSP در فاز بهینه‌سازی به نحو قابل توجهی کمتر از CNN-TSP می‌شود. این پدیده ناشی از کیفیت انتخابهای تصمیم‌گیرنده فازی نسبت به تصمیم‌گیرنده رقابتی است که سبب کاهش حجم محاسبات آموزش شبکه، در فاز بهینه‌سازی شده است. به عبارت دیگر برای توزیعهای با بیش از ۲۱۰ شهر نه تنها پاسخهای FNN-TSP از CNN-TSP بهتر است بلکه زمان همگرایی آن نیز کوتاهتر است.

۶) نتیجه‌گیری

در این مقاله عملکرد شبکه CNN-TSP با استفاده از یک سیستم فازی ژنتیکی بهبود یافته است. سیستم فازی جایگزین تصمیم‌گیرنده رقابتی، در بخش سازنده شبکه شده است. قوانین سیستم فازی مذکور توسط یک الگوریتم ژنتیکی بهینه گردیده است. سیستم فازی نه تنها دقت شبکه را افزایش داده بلکه توانسته با انتخاب مناسب شهرها در فاز سازنده، زمان همگرایی شبکه را برای توزیعهایی که بیش از ۲۱۰ شهر دارند، کاهش دهد. بعلاوه دقت و سرعت شبکه FNN-TSP نسبت به CNN-TSP با افزایش تعداد شهرها افزایش می‌یابد که این مزیتی با اهمیت است.

قدردانی

نویسنده مقاله از دکتر اکبرزاده و دکتر تشنه‌لب بخاطر توجه و راهنمایی‌هایشان تشکر می‌نماید.

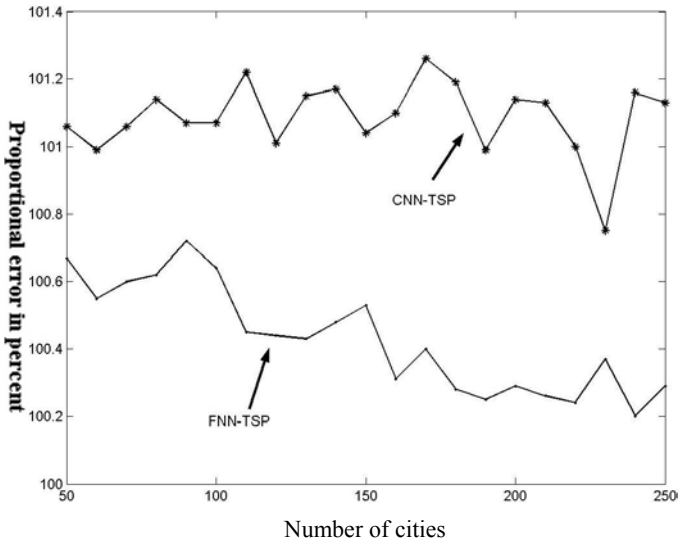
مراجع

[1] P. Crescenzi, V. Kann, “A compendium of NP optimization problems,” Oct. 1995. http://www.zvne.fer.hr/~zmija/resources/science_resources/nn_for_optimization/index.html

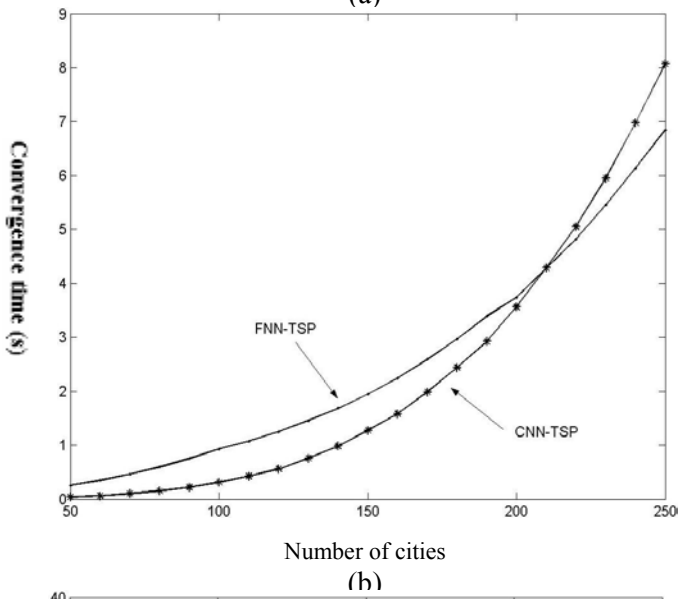
[2] K. Fujimura, S. Fujiwaki, O.-C. Kwaw, H. Tokutaka, “Optimization of electronic chip-mounting machine using SOM-TSP method with 5 dimensional data,” International Conferences on Info-tech and Info-net (ICII), 4: 26-31, 2001.

[3] M. K. Mehmet Ali, F. Kamoun, “Neural networks for shortest path computation and routing in computer networks,” IEEE Trans. Neural Networks, 4(6), Nov. 1993.

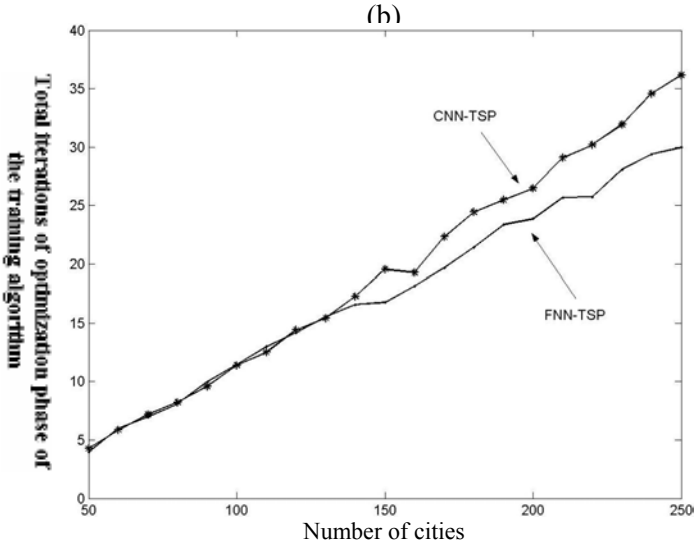
[4] D. Banaszak, G.A. Dale, A.N. Watkins, J.D. Jordan, “An optical technique for detecting fatigue cracks in



(a)



(b)



(c)

شکل (۶): (a) منحنیهای خطای نسبی، (b) زمان همگرایی و (c) تعداد تکرارهای مرحله بهینه‌سازی تا رسیدن به همگرایی برای شبکه‌های عصبی CNN-TSP و FNN-TSP در مقایسه بایکدیگر. محور افقی نشان دهنده تعداد شهرهای توزیع تحت بررسی است.

aerospace structures," 18th International Congress on Instrumentation in Aerospace Simulation Facilities (ICIASF), pp. 27/1-27/7, 1999.

[5] F. Tian, L. Wang, "Chaotic simulated annealing with augmented Lagrange for solving combinatorial optimization problems," 26th Annual Conference of the IEEE Industrial Electronics Society (IECON), 4: 2722 -2725, 2000.

[6] L. Jiao, L. Wang, "A novel genetic algorithm based on immunity," IEEE Trans. on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 30, No. 5, Sep. 2000.

[7] T. Stützle, M. Dorigo, "ACO algorithms for the traveling salesman problem," In K. Miettinen, M. Makela, P. Neittaanmaki, J. Periaux, editors, Evolutionary Algorithms in Engineering and Computer Science, Wiley, 1999.

[8] S. Boettcher, A. Percus, "Research Note: Nature's way of optimizing," Artificial Intelligence, 119:275–286, 2000.

[9] M. Saadatmand-T., M.-R. Akbarzadeh-T., M. Khademi, "A novel hybrid neural network for the traveling salesman problem (TSP)," 9th Iranian Conference on Electrical Engineering (ICEE2001), May 2001.

[10] M. Saadatmand-T., M.-R. Akbarzadeh-T., "A novel constructive hybrid neural network for the symmetric shortest path problem with specified city number," 5th Iranian Conference on Intelligent Systems (CIS2003), 2003.

[11] T. Ueda, K. Takahashi, I. Sasase, S. Mori, "Hopfield-type neural networks with fuzzy sets to gather the convergence speed", International Joint Conference on Neural Networks (IJCNN), 4: 624-629, 1992.

[12] W.J. Wolfe, "Summary of the fuzzy Hopfield-Tank TSP model", International Joint Conference on Neural Networks (IJCNN), 6: 4228-4231, 1999.

[13] O. Cordòn, F. Herrera, F. Hoffmann, L. Magdalena, "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases", Advances in Fuzzy Systems-Application and Theory Vol. 19, World Scientific Publishing, Singapore, 2001.

[14] D. Whitley, "A genetic algorithm tutorial", Statistics and Computer, 4: 65-85, 1994.

[15] D. E. Goldberg, "Genetic and evolutionary algorithms in the real world", IlliGAL Report No. 99013, March 1999.

[۱۶] م. محمدی تاکامی، ع. فاتحی، "طراحی و تنظیم بهینه قواعد کنترل‌کننده فازی یا الگوریتم ژنتیکی و پیاده‌سازی آن بر روی سیستم توپ و صفحه"، یازدهمین کنفرانس مهندسی برق ایران، ۱۳۸۲.

[۱۷] م. تشنه‌لب، ترجمه، "سیستمهای فازی و کنترل فازی"، انتشارات دانشگاه صنعتی خواجه نصیر الدین طوسی، ۱۳۷۸.

[18] D. E. Goldberg, "A note on boltzman tournament selection for genetic algorithms and population oriented simulated annealing", TCGA 90003, Engineering Mechanics, Alabama University.

[19] D. Whitley, "The GENITOR algorithm and selective pressure", Proc. 3th International Conference on Genetic Algorithms, ed. Morgan-Kaufmann: 116-121, 1989.