فصلنامه‌ی علمی و پژوهشی شریف
تابستان ۱۳۸۵، شماره‌ی سی و چهارم، ص. ۴۳-۵۰

حل مسائل بهینه‌سازی توزیع با استفاده از شبکه‌های عصبی

عبدالحمید مدرس (استادیار)
دانشکده‌ی مدیریت و اقتصاد، دانشگاه صنعتی شریف

در این نوشتار چند روش جدید بر مبنای رویکرد شبکه‌های عصبی خودسازنده[۱] برای حل مسائل بهینه‌سازی ارائه می‌شود. این روش‌ها به‌ویژه برای دو مسئله‌ی مهم در برنامه‌ریزی توزیع ــ مسئله‌ی فروشنده‌ی دوره‌گرد[۲] (TSP) و مسئله‌ی مسیریابی[۳] (VRP) ــ توسعه یافته‌اند. عملکرد روش‌های ارائه شده با به‌کارگیری مسائل استاندارد موجود در ادبیات مورد ارزیابی قرار گرفته‌اند. نتایج این آزمایشات نشان می‌دهد که روش‌های ارائه شده از نظر سرعت و کیفیت نسبت به روش‌های مشابه کارایی مناسب‌تری دارند.

مقدمه

با گسترش روزافزون فناوری اطلاعات و امکان برقراری ارتباط اطلاعاتی بین بخش‌های مختلف یک سازمان و نیز سازمان‌های مرتبط، از نظر فعالیت‌های تجاری فراهم شده است. این ارتباط امکان تصمیم‌گیری را از طریق به‌کارگیری سطح وسیعی از داده‌های درونی و بیرونی سازمان فراهم ساخته است. بسیاری از تصمیمات و برنامه‌ریزی‌های سازمان‌ها در عملیات بازاریابی، مالی، نیروی انسانی با استفاده از مدل‌های بهینه‌سازی صورت می‌گیرد. به‌عبارت دیگر روند پیشرفت فناوری اطلاعات و توسعه‌ی ارتباط درون سازمانی و بین سازمانی نیاز به استفاده از مدل‌های بهینه‌سازی را برای استفاده منطقی از داده‌ها و اطلاعات فراهم شده گسترش داده است. همچنین توسعه‌ی اینترنت و شبکه‌های رایانه‌یی امکان استفاده‌ی همزمان از داده‌های مورد نیاز را در گستره‌ی وسیعی از سازمان‌ها ایجاد کرده است. این مطلب متضمن بزرگ شدن اندازه‌ی مسائل بهینه‌سازی که در عمل وجود دارند، خواهد بود. بدیهی است در این شرایط لزوم به‌کارگیری روش‌های کارآمدی که بتوانند با سرعت بالا مسائل بسیار بزرگ را با کیفیت قابل قبول حل کنند بیش از بیش احساس می‌شود.

اخیراً روش‌های بهینه‌سازی که بر پایه‌ی رویکرد هوش مصنوعی توسعه یافته‌اند، موفقیت‌های چشم‌گیری در حل مؤثر و کارای مسائل بهینه‌سازی به‌دست آورده‌اند. روش‌هایی چون الگوریتم ژنتیک[۴] (GA)، جستجوی ممنوع[۵] (TS) و گرم وسرد کردن شبیه‌سازی شده[۶] (SA)، قابلیت‌های خود را در حل مسائل بسیار بزرگ عملی به خوبی نشان داده‌اند. امتیازات ویژه‌ی موجود در شبکه‌های عصبی امکان کاربرد آنها را در حوزه‌ی وسیعی از تحقیقات فراهم ساخته است. از جمله‌ی این امتیازات می‌توان به امکان یادگیری و بهبود عملکرد براساس داده‌های ورودی اشاره کرد. همچنین امکان انجام محاسبات به‌صورت موازی در شبکه‌های عصبی امتیاز دیگری است که با توجه به گسترش سخت‌افزارهای موازی، امکان حل مسائل بسیار بزرگ را توسط این رویکرد ممکن می‌سازد. این دو ویژگی، شبکه‌های عصبی را به‌عنوان گزینه‌یی جذاب برای حل مسائل عملی بهینه‌سازی مطرح می‌سازد.[۱]

در این نوشتار دو مسئله‌ی فروشنده‌ی دوره‌گرد (TSP) و(VRP) به‌عنوان مسائل پایه در برنامه‌ریزی توزیع برای به‌کار بردن روش‌های پیشنهادی انتخاب شده‌اند. در مسئله‌ی TSP فروشنده‌یی مایل است از شهر مبدأ شروع کند و به تمامی شهرهای مورد نظر خود، هرکدام فقط یکبار، سر زده و نهایتاً به شهر مبدأ مراجعت کند. در شکل مسئله‌ی VRP، مجموعه‌یی از وسایل نقلیه برای انتقال کالا از یک مرکز توزیع به تعدادی نقاط تقاضای معلوم در دسترس‌اند. هدف مسئله کمینه‌سازی هزینه‌ی انتقال مواد به نقاط مورد تقاضاست. البته در عمل این مسئله، در مقایسه با فرم ساده‌ی فوق، دارای پیچیدگی بسیار زیادی است.[۲] تاکنون روش‌های زیادی برای حل مسئله‌ی VRP, TSP ارائه شده است. طبیعت NP-Hard این مسائل امکان به‌کارگیری روش‌های دقیق برای حل این مسائل را سلب می‌کند و برای مسائل عملی و بزرگ باید به روش‌های ابتکاری توجه داشت. با توجه به حجم هزینه‌های صرف شده در بخش توزیع و با عنایت به این نکته که حل این مسائل به کاهش درصد قابل قبولی از هزینه‌های توزیع می‌انجامد، حل این مسئله از نظر کاربردی دارای اهمیت بالایی است.

بخش دوم این نوشتار، مروری است بر رویکردهای مختلف به‌کارگیری شبکه‌های عصبی برای حل مسائل بهینه‌سازی و نیز ضمن آن شبکه‌های عصبی انطباقی[۷] تشریح می‌شود. در فصل سوم، روشی جدید برای حل مسائل بهینه‌سازی ارائه شده که جزئیات آن برای حل دو مسئله‌ی TSPو VRP تشریح می‌شود. در بخش چهارم روش ارائه شده با استفاده از مسائل استاندارد موجود در ادبیات، در مقایسه با روش‌های

مشابه، مورد ارزیابی قرار می‌گیرد. جمع‌بندی مطالب نیز در بخش پنجم ارائه خواهد شد.

شبکه‌های عصبی در بهینه‌سازی

ایده‌ی به‌کارگیری شبکه‌های عصبی برای حل مسائل بهینه‌سازی توسط هاپفیلد وتانک ارائه شد.[۳] آنها با استفاده از یک شبکه‌ی کاملاً به‌هم پیوسته به ارائه‌ی تعریف تابع انرژی برای این شبکه مبادرت کردند. می‌توان نشان داد که در فرایند یادگیری، مقدار این تابع انرژی کاهش می‌یابد تا در نقطه‌یی به یک کمینه‌ی محلی می‌رسد. بنابراین با تصویر کردن مسائل بهینه‌سازی توسط این شبکه می‌توان از آن برای حل مسائل بهینه‌سازی استفاده کرد. این امر با قرار دادن تابع هدف و فرمی جبرانی ازمحدودیت‌ها در تابع انرژی امکان‌پذیر خواهد بود. آنها از این الگوریتم برای حل مسئله TSP استفاده کردند و به جواب نسبتاً قابل قبولی دست یافتند. اصلاحات زیادی که در تحقیقات بعدی ارائه شد،[۴] کارایی این روش را بالا برد و این رویکرد را به‌عنوان رویکرد جذابی برای حل مسائل بهینه‌سازی مطرح ساخت. رویکرد هاپفیلد و تانک دارای چندین ضعف عمده بود. پیچیدگی شبکه‌ی پیشنهادی که نیازمند حجم بالای محاسبات است، امکان به‌کارگیری شبکه را برای حل مسائل بزرگ سلب می‌کند. از طرفی با توجه به اینکه شبکه سازوکاری برای خروج از کمینه‌ی محلی ندارد، به‌شدت به مقادیر اولیه حساس بوده و نیز متضمن ارائه‌ی جواب با کیفیت بالا نیست. همچنین با توجه به اینکه سازوکار ارضای محدودیت‌ها توسط حضور آنها در تابع هدف با ضریب بزرگ است، الگوریتم هیچگونه تضمینی برای ارائه‌ی جواب امکان‌پذیر ندارد.

اخیراً شبکه‌های عصبی که یادگیری از داده‌های ورودی را امکان‌پذیر می‌سازند، به شدت به‌عنوان روش‌های جذاب برای حل مسائل بهینه‌سازی مورد توجه محققین قرار گرفته‌اند.[۱] روش‌های توسعه یافته در این حوزه را می‌توان به دو دسته‌ی Elastic Net (EN) و شبکه‌های عصبی خودسازنده (SOFM)تقسیم کرد. هر دو روش از الگوی مشابهی برای یادگیری برخوردارند و تفاوت آنها در به‌روزآوری گره‌هاست. در حالی که EN در به‌روز آوردن وزن کمان‌های مرتبط به گره‌ها تمامی آنها را مد نظر قرار می‌دهد، SOFM گره‌ها را از طریق یک فرایند رقابتی انتخاب می‌کند. در قسمت‌های بعدی به تشریح عملکرد این دو روش برای حل مسئله‌ی TSP خواهیم پرداخت.

روش Elastic Net (EN)

این روش را می‌توان از نظر هندسی به الگوریتمی برای تغییر شکل یک حلقه‌ی لاستیکی برای قرار دادن آن بر روی نقاط موردنظر با کم‌ترین طول تعبیر کرد. در صورتی که شهرهای موردنظر در مسئله‌ی TSP را در یک صفحه قرار دهیم، می‌توان این روش را به لحاظ هندسی به‌عنوان الگوریتمی برای جایگزینی حلقه‌یی لاستیکی با کم‌ترین طول بر روی شهرهای مسئله تعبیر کرد.[۵]

در صورتی‌که تعداد شهرهای مسئله N باشد، این الگوریتم ابتدا تعداد M (M= ۲٫۵N)گره را در مرکز مختصات شهرهای مسئله ایجاد می‌کند و در یک فرایند تکراری شروع به تغییر محل گره‌ها و نزدیک کردن مختصات آنها به مختصات واقعی شهرهای مسئله می‌پردازد. این روش تا آنجا که گره‌ها به اندازه‌ی قابل قبول به شهرها نزدیک شده باشند ادامه می‌یابد. به‌لحاظ هندسی می‌توان گفت در ابتدا یک حلقه لاستیکی کوچک در مرکز شهرها قرار داده می‌شود و به تدریج این حلقه به سمت مختصات شهرهای واقعی مسئله اتساع می‌یابد. نکته‌ی مهم در این روش، سازوکار نزدیک شدن گره‌ها به شهرهاست. در فرایند نزدیک شدن گره‌ها به شهرها، هر گره تحت تأثیر دو نیرو قرار دارد. نیروی اول گره را به سمت نزدیک‌ترین شهر به خود جذب می‌کند و نیروی دوم در جهت جذب گره به سمت گره‌های مجاور خود است. در واقع نیروی دوم به گونه‌یی عمل می‌کند که حلقه کم‌ترین طول را داشته باشد. در حقیقت این مطلب همان کمینه‌سازی تابع هدف مسئله‌ی بهینه‌سازی است.

اگر X_i مختصات شهر i و Y_i مختصات گره j باشد، میزان حرکت گره j از رابطه‌ی ۱ تعیین می‌شود:

$$\Delta Y_i = \alpha \sum w_{ij}(X_i - Y_j) + \beta K(Y_{j+1} - 2Y_j + Y_{j-1}) \quad (1)$$

در این رابطه α و β پارامترهای ثابت‌اند و K پارامتری است که به‌تدریج مقدار آن کاهش می‌یابد. اگر D_{xiyi} فاصله‌ی شهر iاز گره j باشد مقدار W_{ij} از رابطه‌ی ۲ تعیین می‌شود.

$$w_{ij} = \frac{\Phi(d_{X_iY_j}, K)}{\sum_k \Phi(d_{X_iY_K}, K)} \quad (2)$$

در رابطه‌ی ۲، Φ (d,K) تابعی گوسی به‌صورت $exp(-d^2/2K^2)$ است. می‌توان نشان داد که این الگوریتم در حقیقت در یک فرایند تکراری در جهت کمینه‌سازی تابع انرژی زیر عمل می‌کند.

$$E = -\alpha K \sum_i \ln \sum_j \Phi(d_{SiYj}, K) + \sum_j |Y_{j+l} - Y_j|^2 \quad (3)$$

آشکار شدن تمامی گره‌ها در هر مرحله مستلزم عملیات سنگینی است که منجر به کاهش سرعت حل مسئله می‌شود. البته این امر پایایی قابل قبولی به الگوریتم خواهد داد. این الگوریتم در تحقیقات بعدی تا حد زیادی بهبود یافت،[۶-۸] اما جواب‌های ارائه شده توسط این رویکرد، نسبت به رویکرد SOFM از کیفیت پایین‌تری برخوردارند.

روش SOFM

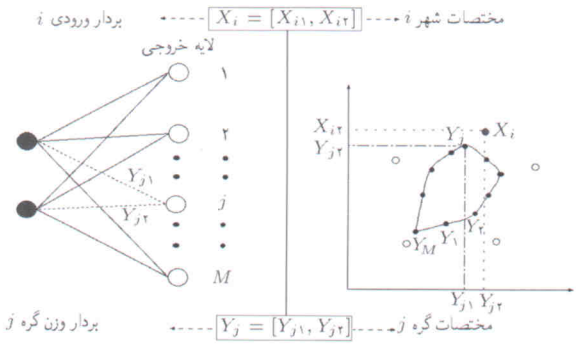
الگوهای خودسازنده نمونه‌یی از شبکه‌های عصبی رقابتی‌اند که از آنها به‌منظور خوشه‌بندی و دسته‌بندی داده‌ها استفاده می‌شود. این دسته از شبکه‌های عصبی از مجموعه شبکه‌هایی هستند که از رژیم یادگیری بدون نظارت[۸] استفاده می‌کنند. این شبکه‌ها بر اساس ساختار مغز پستانداران، که در آن هر بخش به انجام وظیفه‌ی ویژه‌یی تخصیص داده شده و هر دسته از عصب‌ها تنها نسبت به سیگنال خاصی حساسیت نشان می‌دهند، طراحی شده‌اند.[۹و۱۰]

یک شبکه‌ی SOFM ساده از یک لایه گره‌های ورودی و یک لایه گره‌های خروجی تشکیل می‌شود، و هر گره ورودی از طریق یک کمان به تمامی گره‌های خروجی مرتبط است. شبکه با مقادیر تصادفی اولیه برای وزن هر یک از کمان‌ها شروع به یادگیری می‌کند. به ازاء هر داده‌ی ورودی، گره خروجی براساس معیار تعیین شده برای شبکه انتخاب شده و وزن کمان‌های آن گره به همراه وزن چند گره مجاور آن به گونه‌یی که به داده ورودی نزدیک شوند، آشکار می‌شود. به‌روز نمودن وزن گره‌های مجاورِ گره برنده در هر تکرار، نکته‌ی کلیدیِ این الگوریتم است. این عمل باعث نگه‌داری گره‌های مجاور در کنار یکدیگر می‌شود، و تا زمان پایداری شبکه، یعنی زمانی‌که داده‌های ورودی به تغییر چندانی در وضعیت شبکه منجر نشود، ادامه خواهد یافت.

به‌منظور به‌کارگیری این روش برای مسئله‌ی TSP [۱۱-۱۶] می‌توان از یک شبکه‌ی دو لایه‌یی که شامل دو گره ورودی و M گره خروجی مطابق شکل ۱ است، استفاده کرد.

فرایند تکامل شبکه را می‌توان به‌لحاظ هندسی به کشیده شدن یک حلقه‌ی فرضی به سمت مختصات شهرها تصور کرد. مختصات شهرها به‌طور تصادفی به شبکه وارد می‌شود؛ فاصله‌ی بین داده‌ی ورودی و وزن هر گره به‌عنوان معیار ارزیابیِ انتخاب گره برنده در نظر گرفته می‌شود. به‌عبارت دیگر، گره برنده (J) آن است که دارای حداقل فاصله با شهر i که در این مرحله به شبکه وارد می‌شود باشد.

$$net_j = |X_i - Y_j| \qquad J = Argmin_i\{net_j\} \qquad (۴)$$



شکل ۱. شبکه‌ی SOFM برای TSP و تعبیر هندسی آن.

X_i مختصات شهر i و Y_i مختصات گره j است و |.| اپراتور فاصله است. همچنین مختصات گره برنده به همراه چند گره مجاور آن با استفاده از رابطه‌ی ۵ تعیین می‌شود.

$$Y_j^{*\ new} = Y_j^{*\ old} + \mu f(.)(X_i - Y_j^{*\ old}) \qquad (۵)$$

در اینجا μ یک پارامتر با مقداری بین صفر ویک، و f(.) تابع همسایگی است که تعیین‌کننده‌ی میزان تأثیر مختصات شهر ورودی بر گره‌های مجاورِ گره برنده است. این تابع که در یادگیری با کیفیت بالای شبکه نقش اساسی دارد، باید دارای دو ویژگی اساسی باشد: اولاً این تابع باید میزان تأثیر داده‌ی ورودی به گره‌های مجاور را با دور شدن گره از گره برنده کاهش دهد، ثانیاً مقدار این تأثیر در فرایند تکامل شبکه باید کاهش یابد، به‌گونه‌یی که میزان تأثیر داده‌ی ورودی به‌گره‌های مجاور در مراحل اولیه زیاد و به‌تدریج کاهش یابد. پیداست که سازوکار تأثیرگذاری بر گره‌های مجاور باعث ایجاد نیروی کشش بین گره‌ها شده و تنها همین نیروست که به کمینه‌سازی مسیر نهاییِ تعیین شده توسط شبکه کمک می‌کند. بنابراین کیفیت جواب مسئله تا حد زیادی به چگونگی تعیین تابع f(.) بستگی دارد.

روشی برای حل مسائل توزیع

در این بخش با استفاده از رویکرد SOFM روش‌هایی برای حل مسائل مهم بهینه‌سازی در برنامه‌ریزی توزیع ارائه می‌شود. ابتدا روشی برای مسئله‌ی TSP که در حقیقت مسئله‌ی پایه برای مسائل برنامه‌ریزی توزیع و مسیریابی است، ارائه می‌شود. با توجه به فراوانی تعداد پارامترهای الگوریتم‌های موجود، در روش ارائه شده تعداد این پارامترها کاهش یافته است. همچنین در این نوشتار روش جدیدی برای انتخاب گره برنده به‌منظور دست‌یابی به کیفیت بهتر و سرعت بالاتر ارائه شده است. در قسمت بعد این روش برای حل مسئله VRP توسعه می‌یابد.

الگوریتم برای TSP

چنان که ذکر شد، فرایند حل مسئله‌ی TSP شامل دو بخش عمده‌ی نحوه‌ی انتخاب گره برنده، و چگونگی میل کردن گره‌ها به سمت شهرهاست. از آنجا که تعیین تعداد گره‌ها دقیقاً به اندازه‌ی تعداد شهرها، فرایند جداسازی گره‌ها را بسیار ظریف کرده و الگوریتم را نسبت به مقادیر اولیه و پارامترها حساس می‌کند، در روش پیشنهادی تعداد گره‌ها دو برابر تعداد شهرها انتخاب شده است (M=۲N). لازم به توضیح است که آزمایشات نشان داد که انتخاب M=۱٫۵N چندان مؤثر نیست، و نیز انتخاب M=۲٫۵N زمان اجرای الگوریتم را به‌شدت افزایش می‌دهد. همچنین به‌منظور ارائه فرصت یکسان به گره‌ها برای برنده شدن، برای

هر گره شاخصی بازدارنده در نظر گرفته شده است که در صورت برنده‌شدن گره، این شاخص مقدار پیدا می‌کند و به دیگر گره‌ها فرصت بیشتری برای برنده شدن می‌دهد. یادآور می‌شود که تنها ترکیب افزایش تعداد گره‌ها و به‌کارگیری شاخص بازدارنده می‌تواند به بالا بردن کیفیت عملکرد الگوریتم منجر شود. در حالی که در الگوریتم‌های موجود، در شروع الگوریتم، شهرها به‌صورت تصادفی مرتب شده و این ترتیب در تمامی مراحل حفظ می‌شود. در روش پیشنهادی در هر تکرار ترتیب شهرها به‌طور تصادفی تغییر می‌کند. این امر منجر به پایایی الگوریتم می‌شود و از وابستگی آن به مقادیر اولیه می‌کاهد.

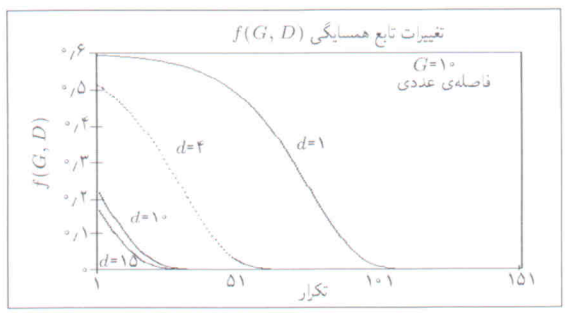
تابع همسایگی که تعیین‌کننده‌ی میزان تأثیر شهر ارائه شده به شبکه به گره‌های مجاور گره برنده است، عامل مهم دیگری در کیفیت عملکرد الگوریتم است. به‌طور کلی این تابع تعیین‌کننده‌ی میزان تغییر مکان گره‌های مجاورِ گره برنده به سمت مختصات ارائه شده به شبکه است. طبیعی است میزان تأثیر برای برنده زیاد است و با زیاد شدن فاصله‌ی گره از گره برنده میزان تأثیر کاهش می‌یابد، به‌طوری که بعضی از گره‌های دور هیچ‌گونه تأثیری نمی‌پذیرند.

در شکل ۲ میزان تأثیر تابع همسایگی برای گره‌ها در تکراری‌های مختلف الگوریتم نشان داده شده است.

چنان که شکل نشان می‌دهد با دور شدن فاصله‌ی گره از گره برنده میزان تأثیر به شدت کاهش می‌یابد، به‌طوری که برای گره‌های دارای فاصله‌ی عددی بیش از ۱۵ با گره برنده تقریباً هیچگونه تأثیری وجود ندارد. با توجه به این نکته، در الگوریتم پیشنهادی تنها موقعیت ۴۰ درصد از گره‌ها تعیین می‌شود. این راهکار باعث افزایش قابل توجه سرعت الگوریتم بدون تأثیر در کیفیت جواب‌ها می‌شود. به‌طور خلاصه می‌توان الگوریتم پیشنهادی را به شکل زیر بیان کرد.

قدم۰: N را برابر تعداد شهرها و تعداد گره‌ها، M را برابر 2N انتخاب کنید.

قدم۱: شهرها را به‌طور تصادفی مرتب کرده و آنها را از ۱ تا N شماره‌گذاری کنید. شاخص بازدارنده تمام شهرها را صفر کنید.



شکل ۲. تغییرات تابع همسایگی در تکرارهای مختلف و برای مقادیر مختلف فاصله عددی.

قدم ۲: از میان گره‌هایی که شاخص بازدارنده‌ی آنها صفر است، گرهی که دارای نزدیک‌ترین فاصله به شهر ورودی است انتخاب کنید.

$$for\{j|INHIBIT[j]=۰\},\quad J=Argmin_j|X_i-Y_j| \tag{۶}$$

سپس شاخص بازدارنده‌ی شهر برنده را به ۱ تغییر دهید.

$$INHIBIT[J]=۱.$$

قدم ۳: گره J و گره‌های مجاور آن را با استفاده از تابع همسایگی F(G,d) که در آن G پارامتر و d فاصله‌ی عددی از گره J است، تعیین کنید.

$$d=min\{||j-J||, M-||j-J||\}, \tag{۷}$$

$$F(d,G)=\begin{cases} e^{(-d^۲/G^۲)} & if\ d<H \\ ۰ & Otherwise \end{cases}$$

||.|| بیانگر قدر مطلق و H=۰/۲M.

قدم ۴: مقدار i را یک عدد افزایش دهید. اگر $i \le N+۱$ به قدم ۲ بروید، و در غیر این‌صورت $G=(۱-\alpha)^*G$ و i=۱ و به قدم بعدی بروید.

قدم ۵: اگر محل گره‌ها در حد قابل قبول به شهرها نزدیک شده است متوقف شوید، وگرنه به قدم ۱ بروید.

قابل ذکر است که فرایند جداسازی گره‌ها در مراحل اولیه بسیار ظریف است و تأثیر زیادی در کیفیت عملکرد الگوریتم دارد. در روش پیشنهادی ترکیب افزایش تعداد گره‌ها به دوبرابر تعداد شهرها و تعریف شاخص بازدارنده برای گره‌های برنده به‌نحو مناسبی این فرایند را هدایت می‌کند.

الگوریتم برای مسئله‌ی VRP

مسئله‌ی مسیریابی با توجه به محدودیت ظرفیت برای وسایل نقلیه، در مقایسه با TSP بسیار پیچیده‌تر می‌شود. تاکنون تحقیقات چندانی در زمینه‌ی به‌کارگیری SOFM برای مسئله‌ی مسیریابی انجام نگرفته است و در این رابطه تنها می‌توان به کارهای Ghaziri [۱۷] و Vakhutinsky & Golden [۱۸] اشاره کرد.

با فرض در دسترس بودن K وسیله‌ی نقلیه، به‌منظور ارائه‌ی سرویس به N نقطه با تقاضای معین می‌توان روش بکارگرفته شده برای TSP را برای این مسئله توسعه داد. در این مسئله به جای شکل دادن یک مسیر بسته در مرکز مختصات شهرها باید K مسیر بسته حول مرکز توزیع

مسئله شکل داده و فرایند یادگیری شبکه را متناسب با محدودیت‌ها و ویژگی‌های این مسئله تغییر داد. چنان که پیش‌تر ذکر شد یکی از نقاط ضعف SOFM عدم توانایی کافی در جداسازی گره‌ها است. با توجه به پیچیدگی مسئله‌ی VRP که در آن علاوه بر وجود چندین مسیر، محدودیت ظرفیت وسایل نقلیه نیز وجود دارد، باید فرایند جداسازی گره‌ها را به‌نحو مناسبی سامان بخشید.

در SOFM تنها نیرویی که گره‌ها را درمجاورت یکدیگر نگه می‌دارد، تعیین گره‌های مجاور گره برنده است. به نظر می‌رسد تقویت این نیرو نمی‌تواند مشکل جداسازی مناسب گره‌ها را به‌نحو مناسبی حل کند. بدین منظور در این تحقیق تابع جدیدی برای تعیین گره‌ها معرفی شده است. این تابع که ایده‌ی آن از روش EN اقتباس شده است، دربردارنده‌ی نتایج مطلوبی در فرایند جداسازی گره‌ها برای مسئله‌ی مسیریابی است.

پیشنهاد می‌شود که وزن گره‌ها توسط تابع ۹ تعیین شود.

$$Y_j^r = \mu F(g,d)(X_i - Y_j^r) + \lambda(Y_{j+1}^r - 2Y_j^r + Y_{j-1}^r) \quad (8)$$

در این تابع، Y_j^r بیانگر وزن گره j در مسیر r است. همچنین محدودیت ظرفیت وسایل نقلیه را می‌توان با سازوکار ارضاء محدودیت به الگوریتم افزود. برای تحقق این مطلب در این تحقیق به سازوکار انتخاب گره برنده توجه شده است. قانون انتخاب گره برنده با افزودن یک عبارت که دربردارنده‌ی میزان بار تخصیص یافته به‌وسیله‌ی نقلیه است، اصلاح می‌شود. این عبارت جدید را «جریمه‌ی مسیر» می‌نامیم. «جریمه‌ی مسیر» باعث محرومیت وسایل نقلیه‌ی دارای بیش از حد مجاز ظرفیت از رقابت شده و بدین ترتیب امکان برنده شدن برای سایر وسایل نقلیه را فراهم می‌سازد. این سازوکار قادر است که بارها را به‌طور یکنواخت بین وسایل نقلیه تقسیم کند.

گرچه در این تحقیق ظرفیت وسایل نقلیه یکسان فرض شده است، ولی این ایده به‌راحتی قابل توسعه به مسئله‌ی مسیریابی با وسایل نقلیه با ظرفیت‌های متفاوت است. بنابراین در روش پیشنهادی، گره برنده از طریق رابطه‌ی تعیین می‌شود.

$$J = Argmin_j\{|X_i - Y_j^r| + v.B_r\} \quad (9)$$

که در آن

$$B_r = \sum(q_l^r/Q^2) \quad (10)$$

در معادله‌ی ۱۱ q_l^r بیانگر تقاضای مکان l که به مسیر r تخصیص یافته است، Q ظرفیت وسیله نقلیه و v پارامتر تنظیم است. عبارت اول رابطه‌ی ۱۰ بیانگر نزدیکی گره j با شهر ارائه شده به شبکه، و عبارت دوم همان «جریمه‌ی مسیر» است که به‌منظور قراردادن محدودیت ظرفیت در فرایند یادگیری شبکه طراحی شده است. لازم به ذکر است که پارامتر v نقش تعیین‌کننده‌یی در عملکرد الگوریتم دارد. این پارامتر در حقیقت تعادل لازم بین امکان‌پذیری و بهینه‌سازی در مسئله را ایجاد می‌کند. از آنجا که با انجام تکرارهای متوالی فاصله‌ی گره‌ها از شهرها کاهش می‌یابد، لازم است به فرم مناسبی با گذشت زمان مقدار v نیز کاهش یابد. بدین منظور براساس ایده‌ی به‌کار گرفته شده در SA مقدار این پارامتر در هر تکرار براساس رابطه‌ی زیر تعیین می‌شود.

$$v = v/(1 + \delta.v) \quad (11)$$

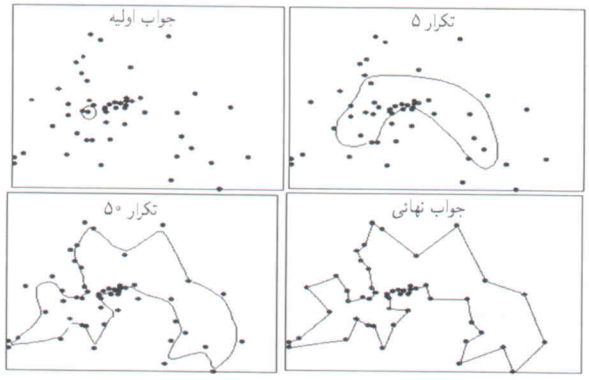
ارزیابی روش‌های پیشنهادی

به‌منظور نشان دادن کارایی روش‌های پیشنهادی با استفاده از داده‌های استاندارد، این روش در مقایسه با روش‌های موجود مورد ارزیابی قرار گرفته است. مسائل مورد استفاده از TSPLIB گرفته شده است.

نتایج آزمایشات برای TSP

شکل ۳ چگونگی شکل‌گیری جواب‌ها را برای یکی از مسائل انتخاب شده در روش پیشنهادی ارائه می‌کند. در این شکل وضعیت جواب‌ها در شروع، پس از ۵ و ۵۰ تکرار و همچنین جواب نهایی ارائه شده است. چنان که شکل نشان می‌دهد، در آغاز یک حلقه‌ی کوچک شکل گرفته و در جریان یادگیری الگوریتم این حلقه به سمت شهرها کشیده می‌شود تا بالاخره کاملاً به کلیه‌ی شهرها برسد. این شکل که فرایند شکل‌گیری جواب برای مسئله‌ی Berlin۵۲ را به تصویر کشیده است، پیچیدگی فرایند انتخاب گره و جداسازی گره‌ها را برای یک مسئله‌ی واقعی نشان می‌دهد.

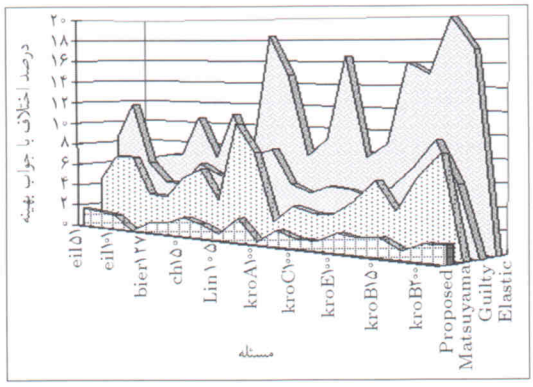
به‌منظور مقایسه‌ی توانایی روش پیشنهادی با شبیه‌سازی چندین الگوریتم موجود،[۱و۱۹] جواب‌های ارائه شده توسط الگوریتم پیشنهادی



شکل ۳. مراحل شکل‌گیری جواب‌ها برای مسئله‌ی Berlin۵۲.

جدول ۱. نتایج آزمایشات روش پیشنهادی برای TSP.

مسئله	روش پیشنهادی		روش Matsuyama		روش Guilty Net		روش EN		جواب بهینه
	جواب	زمان	جواب	زمان	جواب	زمان	جواب	زمان	
eil۵۱	۴۳۳	۹	۴۴۵	۴۷	۵۶۲	۱۱	۴۴۰	۱۲۲	۴۲۶
eil۷۶	۵۴۶	۲۱	۵۷۴	۷۹	۶۰۱	۲۴	۵۷۲	۳۶۵	۵۳۷
eil۱۰۱	۶۳۷	۳۸	۶۷۱	۱۴۶	۶۶۷	۱۲۶	۶۷۱	۶۲۹	۵۶۴
berlin۵۲	۷۵۴۲	۲۱	۵۶۹۲	۴۲	۷۸۷۳	۳۶	۸۰۶۱	۳۹۴	۷۵۴۲
bier۱۲۷	۱۱۹۲۷۲	۸۹	۱۲۲۰۶۲	۵۹۵۱	۱۲۳۳۶۰	۱۶۶	۱۳۰۷۳۷	۵۲۴۸	۱۱۸۲۸۲
ch۱۳۰	۶۱۷۲	۷۲	۶۴۲۲	۲۰۱	۶۴۹۶	۱۳۱	۶۵۲۱	۴۷۳	۶۱۱۰
ch۱۵۰	۶۶۴۳	۱۰۳	۶۹۲۲	۲۷۵	۶۸۸۳	۱۶۸	۷۲۴۸	۷۴۰	۶۵۲۸
rd۱۰۰	۸۰۱۱	۸۳	۸۱۷۱	۲۰۶	۸۲۹۶	۹۳	۸۴۲۳	۷۶۶	۷۹۱۰
lin۱۰۵	۱۴۴۶۸	۵۹	۱۵۸۷۳	۳۳۹۲	۱۵۴۳۸	۱۳۷	۱۷۰۲۵	۸۴۳	۱۴۳۷۹
lin۳۱۸	۴۲۰۲۹	۶۱۵۳	۴۸۲۵۲	۱۶۶۸	۴۵۴۲۸	۱۲۶۳۳	۴۵۳۶۹	۷۱۵	۴۲۹۳۹
kroA۱۰۰	۲۱۳۴۷	۵۳	۲۱۶۶۸	۳۰۸۴	۲۲۳۴۷	۷۰	۲۲۸۷۱	۷۴۹	۲۱۲۸۲
kroB۱۰۰	۲۲۴۵۷	۵۵	۲۲۸۸	۳۰۹۵	۷۱	۲۳۰۹۵	۲۴۱۴۳	۸۱۸	۲۲۱۴۱



شکل ۴. مقایسه‌ی جواب‌های ارائه شده توسط روش پیشنهادی در مقایسه با روش‌های انتخابی.

با این روش‌ها مورد مقایسه قرار گرفته است. لازم به ذکر است در این آزمایش، برای ایجاد شرایط یکسانِ مقایسه، تمامی الگوریتم‌ها پیاده‌سازی و بر روی یک ماشین اجرا شده‌اند. نتایج این آزمایشات در جدول ۱ خلاصه شده است. به‌منظور راحتی مقایسه در شکل ۴، درصد اختلاف جواب‌ها با جواب بهینه، برای روش‌های انتخاب شده، نشان داده شده است.

چنان‌که ملاحظه می‌شود، الگوریتم پیشنهادی، نسبت به الگوریتم‌های مشابه، دارای سرعت بسیار بالایی است، که این امر ناشی از تغییرات مناسب پیشنهاد شده در الگوریتم است. لازم به ذکر است که این آزمایشات نشان می‌دهد که تصادفی کردن ترتیب شهرها به‌طور متوسط به حداقل دو درصد بهبود در جواب‌ها می‌انجامد. برای الگوریتم پیشنهادی اختلاف با جواب بهینه حداکثر ۲٫۱۷٪ و به‌طور متوسط ۱٫۲۲٪ است. این مقادیر برای روش Matsuyama [۲۰]، ۵٫۱۳٪ و برای روش Gulity Net ۶٫۲۶٪ است. روش EN ، نسبت به سایر الگوریتم‌ها، از کارایی کم‌تری برخوردار است. این نتایج تأییدکننده‌ی توانایی بالای روش پیشنهادی برای حل مسئله TSP است.

نتایج برای مسئله VRP

کارایی روش پیشنهادی برای VRP با به‌کارگیری ۱۰ مسئله با اندازه‌های ۲۰ تا ۲۰۰ مورد ارزیابی قرار گرفته است. مقادیر اولیه vg ، براساس ویژگی‌های مسئله انتخاب شده‌اند. مقدار g براساس اندازه‌ی مسئله و v براساس متوسط فاصله‌ی شهرها از مرکز توزیع تعیین شده‌اند. جواب

جدول ۲. نتایج آزمایشات روش پیشنهادی برای VRP.

مسئله	اندازه	تعداد مسیر	Vakhutinsky & golden	جواب روش پیشنهادی	بهترین جواب
eil۲۲	۲۱	۴	۶۵۹	۳۹۰	۳۷۵
eill۲۳	۲۲	۵	۹۵۰	۵۸۵	۵۶۹
eil۳۰	۲۹	۳	۸۵۵	۵۵۷	۵۳۴
۳۲	۴	۸۹۴	۸۸۹	۸۳۵	eil۳۳
۵۰	۵	۵۶۰	۵۳۷	۵۲۱	C۱
۷۵	۱۰	N/A	۸۷۶	۸۳۸	C۲
۱۰۰	۸	N/A	۸۶۳	۸۲۹	C۳
۱۵۰	۱۲	N/A	۱۰۸۲	۱۰۴۴	C۴
۱۹۹	۱۷	N/A	۱۳۳۴	۱۳۸۶	C۵
۱۲۰	۷	N/A	۱۰۶۶	۱۰۴۲	C۱۱

آزمایشات در جدول ۲ ارائه شده است. در این جدول روش پیشنهادی با روش Vakhutinsky وGolden [۱۸] که در آن از شبکه‌های عصبی برای حل مسئله VRP استفاده شده است، مقایسه شده است. روش پیشنهادی در کلیه‌ی مسائل بهتر از روش فوق عمل می‌کند. همچنین میزان اختلاف از جواب بهینه، دربدترین وضعیت ۶٫۵٪ و به‌طور متوسط ۴٪ است. این مقایسه توانایی روش پیشنهادی برای حل مسئله VRP را تصویر می‌کند.

نتیجه‌گیری

در این مطالعه بر پایه‌ی رویکرد SOFM دو روش برای حل مسائل مهم توزیع ارائه شد. آزمایشات، کارایی و سرعت روش‌های پیشنهادی را در مقایسه با روش‌های موجود ارائه کردند. لازم به ذکر است که با ارائه‌ی اصلاحات مناسب می‌توان روش پیشنهادی را برای سایر مسائل بهینه‌سازی توسعه داد. همچنین با به‌کارگیری سخت‌افزارهای موازی می‌توان سرعت حل مسئله را بالا برده و امکان به‌کارگیری این روش را برای مسائل بسیار بزرگ فراهم کرد.

روش‌های ابتکاری برای حل مسئله‌ی بهینه‌سازی را می‌توان به سه دسته‌ی سازنده، بهبوددهنده و ترکیبی تقسیم کرد. چنان‌که محققین اشاره داشته‌اند،[۲۱-۲۳] روش‌های ترکیبی که از ترکیب دو روش ایجاد شده‌اند، توانایی بالاتری برای حل مسائل خواهند داشت. Tsaiو XU [۴] این ایده را برای روش هاپفیلد[۳] به‌کار برده و با به‌کارگیری یک روش بهبوددهنده در پایان حل مسئله توسط روش هاپفیلد جواب‌های بسیار مناسبی به دست آورده‌اند.

پانوشت

1. self-organing neural networks
2. Teraveling Salesman Problem (TSP)
3. Vehicle Routing Problem (VRP)
4. Genetic Algorithms(GA)
5. Tabu Search (TS)
6. simulated annealing
7. adaptive neural network
8. unsupervised

منابع

1. John, H. Anders, K. and Richard, G.P. "The theory of neural computation", Addison-Wesley (1991).
2. Boding, L. Golden, B. Assad, A. and Ball M. "Routing and scheduling of vehicles and crews", *Computer & Operations Research,* (10), pp. 63-211 (1983).
3. Hopfield, J.J. and Tank, D.W. "Neural computation of decisions in optimization problems", *Biological Cybernetics*, (52), pp. 141-152 (1985).
4. Xu, X. and Tsai, W.T. "Effective neural algorithms for the traveling salesman problem", *Neural Network*, (4), pp. 193-2 (1991).
5. Durbin, R. and Willshaw, D. "An analogue approach to the traveling salesman problem using an elastic net method", *Nature*,(326) pp. 689-691 (1987).
6. Boeres, M.C.S. and De Carvalho, L.A.V. "A faster elastic net algorithm for the traveling salesman problem, In Proceedings of the International Joint Conference on Neural Networks, Baltimore, MD, pp. 215-220 (1992).
7. Burr, D.J. "An improved elastic net method for the traveling salesman problem", in Proceedings of the IEEE International Conference on Neural Networks, San Diego, CA, pp. I-69-76 (1988).
8. Martin, W.S. "Parameter sensitivity of the elastic net approach to the traveling salesman problem", *Neural Computation*, (3), pp.363-374 (1991).
9. Hertz, J. Korgh, A. and Palmer, R.G. "The theory of neural computation", Addison-Wesley (1991).
10. Kohonen, T. "Self-organization and associative memory", Springer-Verlag (1984).
11. Angeniol, B. De La Croix Vaubois, G. and Le-Texier, J. "Self-organizing feature maps and the travelling salesman problem", *Neural Networks*, **1** (4), pp. 289-293 (1988).
12. Burke, L.I. "Neural methods for the traveling salesman problem: insights from operations research", *Neural Networks*, (7), pp. 681-690 (1994).
13. Desieno, D. "Adding a conscience mechanism to competitive learning", in Proceedings of the IEEE International Conference on Neural Networks, San Diego, CA, pp. I-177-124 (1988).

14. Favata, F. and Walker, R. "A study of the application of the kohonen-type neural network to the traveling salesman problem", *Biological Cybernetics*, (64), pp.463-468 (1991).

15. Jean-Yves, P. "The traveling salesman problem: a neural network perspective", *ORSA Journal on Computing*, **5** (4), pp. 328-348 (1993).

16. Torki, A. Somhom, S. and Enkawa, T. "A Survey of Adaptive Neural Network Models for Combinatorial Problems"', Technical Report 96-8, Department of Industrial Engineering and Management, Tokyo institute of technology (1996).

17. Ghaziri, H.E. "Solving routing problems by a self-organizing map", in *Artificial Neural Network* by: Kohonen, T., Makisara, K., Simula, O.,and Kangas, J., Elsevier Science Publisher. Networks, San Diego, CA, I-177-124 (1991).

18. Vakhutinsky, A.I. Golden, B.L. "Solving vehicle routing problems using elastic net", *IEEE International Conference on Neural Network*, pp. 4535-4540 (1994).

19. Burke, L.I. and Damany, P. "The guilty net for the traveling salesman problem", *Computers and Operations. Research*, **19** (3/4), pp. 255-265 (1992).

20. Matsuyama, Y. "Self-organization neural networks and various euclidean traveling salesman problems", *Systems and Computers in Japan*, **23** (2), pp. 101-112 (1992).

21. Gerhard, R. "The traveling salesman: computational solutions for TSP applications", Lecture Notes in Computer Science, Springer-Verlag (1991).

22. Golden, B.L. and Stewart, W.R. "Empirical analysis of heuristics. In: *The Travelling Salesman Problem*, Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. and Shmoys, D.B., John Wiely & Sons (1985).

23. Lawler, E.L. Lenstra, J.K. Kan Rinnooy, A.H.G. and Shmoys, D.B. "The traveling salesman problem: a guided tour of combinatorial optimization", John Wiley, Chichester (1985).