The 1st National Conference on
Geospatial Information Technology

K.N.Toosi University of Technology
Faculty of Geomatics Engineering

19 - 20 January 2016

اولین کنفرانس مهندسی فناوری اطلاعات مکانی

دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی نقشه‌برداری
۲۹ و ۳۰ دی ماه ۱۳۹۴

استفاده از سیستم تصویر برای حل مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی

رضا مرتضوی[۱]

۱- دانشگاه دامغان، دانشکده فنی و مهندسی، گروه مهندسی کامپیوتر، استادیار

چکیده :

یکی از مسائل بهینه‌سازی مهم در حوزه‌ی تحلیل شبکه، مسئله‌ی فروشنده‌ی دوره‌گرد است. حل این مسئله به جهت کاربردهای فراوان از اهمیت بالایی برخوردار است و به‌طور خاص در فناوری‌های اطلاعات مکانی به‌عنوان یک مسئله‌ی مهم تحلیل شبکه مطرح می‌شود. متأسفانه علی‌رغم سادگی ظاهری آن، اثبات شده که حل کلی این مسئله از درجه‌ی NP-سخت است. به همین جهت روش‌های ابتکاری زیادی برای کاربردهای عملی پیشنهاد شده است. در این مقاله از روش سیستم تصویر برای تحویل نسخه‌ای از مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی به نسخه‌ی ساده‌تر استفاده شده است. همچنین یکی از روش‌های ابتکاری مرتبط بهبود داده شده است. نتایج تجربی بر روی مجموعه داده‌های واقعی در مقایسه با روش‌های مشابه، مؤید برتری روش پیشنهادی به لحاظ کیفیت پاسخ و زمان دستیابی به آن است.

واژه‌های کلیدی : : مسئله‌ی فروشنده‌ی دوره‌گرد، تحلیل شبکه، سیستم تصویر، فناوری‌های اطلاعات مکانی.

نویسنده مکاتبه کننده : رضا مرتضوی

آدرس پستی : گروه مهندسی کامپیوتر – دانشکده فنی و مهندسی – دانشگاه دامغان – دامغان

تلفن : +98 233 522 0081-6 (320)

آدرس پست الکترونیک : r_mortazavi@du.ac.ir

۱- مقدمه

مسئله‌ی فروشنده‌ی دوره‌گرد (TSP[1]) یکی از مسائل معروف ترکیبیات است. هدف این مسئله در یک بیان تمثیلی بدین‌صورت است که یک فروشنده باید تعداد مشخصی شهر را با شروع از یکی از آن‌ها ملاقات کند و به نقطه‌ی شروع برگردد، به‌طوری‌که هر شهر دقیقاً یک‌بار مشاهده شود و طول مسیر طی شده کمینه باشد.

این مسئله اولین بار در ۱۹۳۰ به‌صورت ریاضی بیان شده است [1] و در حوزه‌های مختلف مانند مسیریابی، شبکه‌های کامپیوتری، طراحی مدارات الکترونیکی، کاربرد دارد. علی‌رغم قدمت و کاربردهای زیاد TSP، این مسئله در حالت کلی در زمان چندجمله‌ای برحسب تعداد شهرها، قابل‌حل نیست و در دسته‌ی مسائل NP-سخت تقسیم‌بندی می‌شود؛ بنابراین با توجه به اهمیت حل آن در کاربردهای عملی، روش‌های ابتکاری مختلفی برای آن پیشنهاد شده است که سعی می‌کنند در زمان قابل قبولی جواب نزدیک به بهینه را بیابند.

با استفاده از روش‌های بهینه‌سازی مبتنی بر تقسیم و حل با شروع از جواب‌های نزدیک به بهینه می‌توان راحت‌تر به جواب‌های بهینه رسید [1]. یکی از معروف‌ترین ابزارها در این بخش Concorde[2] است که برای دستیابی به حل بهینه از روش شاخه و برش استفاده می‌کند. این ابزار می‌تواند مسائلی به بزرگی ۸۵۹۰۰ شهر را در مدت ۱۳۶ پردازنده-سال[3] حل کند [2] و برای نسخه‌هایی با چند میلیون شهر نیز جواب آن در نزدیکی جواب بهینه است [3]؛ بنابراین دستیابی به جواب اولیه‌ی نزدیک به بهینه در زمان معقول، از اهمیت بالایی در حل این مسئله برخوردار است.

یکی از روش‌های موفق برای حل مسئله‌ی فروشنده‌ی دوره‌گرد با عنوان الگوریتم ابتکاری پس‌انداز[4] شناخته می‌شود [4]، اما نسخه‌ی اولیه از این الگوریتم برای مجموعه داده‌های بزرگ قابل استفاده نیست و با محدودیت حافظه و زمان اجرا مواجه است. بهبودهایی از این الگوریتم برای حل این مشکلات پیشنهاد شده است که مبتنی بر استفاده از ساختمان داده‌ای با نام kd-tree [5] است [6]، ولی این بهبودها برای نمونه‌هایی از مسئله با معیار فاصله‌ی اقلیدسی ارائه شده‌اند و نمی‌توانند برای نسخه‌ای از مسئله با معیار فاصله‌ی جغرافیایی استفاده شوند.

در این مقاله برای حل مشکل فاصله‌ی غیر اقلیدسی برای نمونه‌های با معیار فاصله‌ی جغرافیایی، از روش سیستم تصویر[5] استفاده شده است تا فضای مسئله را از جغرافیایی به اقلیدسی (در صفحه) تقریب بزند. همچنین بهبودی در یک الگوریتم ابتکاری ارائه شده است که به بهبود جواب نهایی منجر می‌شود. نهایتاً این روش بر روی نمونه‌هایی از مسئله با اندازه‌های متفاوت ارزیابی شده است و نتایج به لحاظ کیفیت جواب نهایی و زمان اجرا ارائه شده است.

ادامه‌ی این مقاله به‌صورت زیر بخش‌بندی شده است: در بخش ۲ مفاهیم پایه بیان می‌شود. در بخش ۳ روش پیشنهادی ارائه شده است و در ادامه در بخش ۴ نتایج تجربی حاصل از ارزیابی روش ارائه می‌شود. بخش ۵ نیز به نتیجه‌گیری و پیشنهادهایی برای ادامه‌ی کار اختصاص دارد.

[1] Travelling Salesman Problem
[2] http://www.math.uwaterloo.ca/tsp/concorde/index.html
[3] CPU-year
[4] Savings Heuristic
[5] Map Projection

۲- مفاهیم پایه

در این بخش ابتدا مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی به‌صورت رسمی بیان می‌شود. سپس روش‌های ابتکاری حل TSP به اجمال مرور می‌شود. در ادامه روش ابتکاری پس‌انداز که در این مقاله از آن برای حل TSP استفاده کرده‌ایم، توضیح داده می‌شود. در پایان از برخی روش‌های سیستم تصویر مورداستفاده در این مقاله نام برده می‌شود.

۲-۱- معرفی مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی

فرض کنید شبکه‌ای از مکان‌های جغرافیایی (شهرها) به‌صورت گراف کامل $G = (V, E)$ در اختیار است که $V = \{v_1, v_2, \dots, v_n\}$ نشان‌دهنده‌ی شهرها[۶] و $E = V \times V$ یال بین آن‌هاست. وزن هر یال بین دو گره v_i و v_j به صورت $c(v_i, v_j)$ نشان داده می‌شود. هدف مسئله‌ی فروشنده‌ی دوره‌گرد کمینه کردن دوری روی این گراف است که از همه‌ی گره‌ها دقیقاً یک‌بار می‌گذرد (و به مکان شروع برمی‌گردد). به‌صورت رسمی‌تر، هدف یافتن ترتیبی از شهرهاست که تابع هدف رابطه‌ی (۱) کمینه شود.

$$\min \left[\sum_{i=2}^{n} c(v_{i-1}, v_i) + c(v_n, v_1)\right] \qquad (۱)$$

نسخه‌ای از مسئله که در این مقاله به آن پرداخته شده است فواصل بین شهرها را به‌صورت متقارن تعریف می‌کند، یعنی $c(v_i, v_j) = c(v_j, v_i)$. برای نمونه‌هایی از TSP با معیار فاصله‌ی جغرافیایی و در گستره‌ای به وسعت کره‌ی زمین در مجموعه داده‌ی استاندارد TSPLib[۷] معیار فاصله‌ی جغرافیایی بین دو شهر v_i و v_j یعنی $c(v_i, v_j)$ بر حسب متر بر روی کره‌ای به شعاع $R = 6378388$ متر به کمک تابع GEOM(i,j) محاسبه می‌شود[۸]. فرض کنید طول و عرض جغرافیایی شهر v_i در آرایه‌های متناظر به ترتیب به‌صورت $long[i]$ و $lat[i]$ و بر حسب *رادیان* ذخیره شده‌اند. محاسبه‌ی فاصله در الگوریتم ۱ نشان داده شده است.

```
GEOM(i,j) {
R = 6378388
q1 = cos(lat[j])*sin(long[i] – long[j])
q3 = sin((long[i] – long[j]) / 2.0)
q4 = cos((long[i] – long[j]) / 2.0)
q2 = sin(lat[i] + lat[j])*q3*q3 - sin(lat[i] – lat[j])*q4*q4
q5 = cos(lat[i] – lat[j])*q4*q4 - cos(lat[i] + lat[j])*q3 * q3
return (int)(R * atan(sqrt(q1 * q1 + q2 * q2), q5) + 1.0)
}
```

الگوریتم ۱- محاسبه‌ی فاصله‌ی جغرافیایی بین دو شهر v_i و v_j بر حسب متر

---

[۶] از این پس لفظ شهر یا گره به جای یکدیگر استفاده می‌شوند.

[7] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95

[۸] یعنی برای سادگی زمین به صورت یک کره مدل شده است. گرچه این موضوع به طور مستقیم تاثیری بر روش مورد استفاده در این مقاله ندارد.

در الگوریتم ۱، توابع atan، sqrt و int به ترتیب نشان‌دهنده‌ی معکوس تانژانت، جذر و جزء صحیح هستند.

۲-۲- روش‌های ابتکاری حل TSP

روش‌های ابتکاری برای حل TSP شامل روش‌های ساختنی[۹] و روش‌های بهبود تدریجی[۱۰] می‌شوند. در روش‌های ساختنی، بر اساس یک قاعده‌ی تعریف‌شده دور ساخته می‌شود و بعد از ساختن (بخشی از) آن، تلاشی برای بهبود آن انجام نمی‌شود و یال‌ها بدون تغییر می‌مانند. در روش‌های بهبود تدریجی، با شروع از یک جواب اولیه (مثلاً با استفاده از روش‌های ساختنی) به‌دفعات با تغییر یال‌ها سعی می‌شود دور نهایی بهبود یابد. به‌طورکلی روش‌های ساختنی خیلی سریع‌تر از روش‌های بهبود تدریجی هستند ولی معمولاً کیفیت جواب پایین‌تری نسبت به روش‌های بهبود تکراری دارند [7].

روش‌های ساختنی شامل ابتکارات مبتنی بر نزدیک‌ترین همسایه (NN[۱۱])، ابتکارهای درج یال [2]، ابتکارهای مبتنی بر درخت‌های پوشا[۱۲]، ابتکار حریصانه (GH[۱۳]) [8] و ابتکار پس‌انداز[۱۴] می‌شود. در روش NN، با شروع از یک گره، نزدیک‌ترین همسایه‌ی بعدی از گره جاری که هنوز مشاهده نشده به انتهای دور کنونی اضافه می‌شود. در روش GH، همواره کوتاه‌ترین یال به دور ناکامل جاری افزوده می‌شوند، به‌شرط این‌که این افزودن باعث ایجاد یک *زیردور*[۱۵] نشود و درجه‌ی هر دو گره این کوتاه‌ترین یال در دور ناکامل[۱۶] کمتر از ۲ باشد [8]. بهبودهایی از این روش با استفاده از صف اولویت[۱۷] پیشنهاد شده است [9]. روش برووکای سریع (QB[۱۸]) توسعه‌ای از روش GH و مبتنی بر درخت پوشاست که به‌جای استفاده از صف‌های اولویت، از مرتب‌سازی داده‌ها استفاده می‌کند [9].

روش‌های بهبود تدریجی شامل روش 2-opt [10] و روش‌های تکاملی مانند شبیه‌سازی ذوب فلزات[۱۹] [11] و الگوریتم ژنتیک [12] می‌شود. در روش 2-opt به‌عنوان یک روش جستجوی محلی، جای گره‌های انتهایی دو یال از دور داده‌شده به‌صورت ضربدری تغییر می‌کند تا طول کلی دور کاهش یابد. توسعه‌هایی نیز از این روش برای در نظر گرفتن تعداد بیشتری یال ارائه شده است [13]. در روش تکاملی ذوب فلزات تغییرات محدودی در دور داده‌شده پیشنهاد داده می‌شود. درصورتی‌که این تغییر منجر به بهبود شود، تغییر پذیرفته می‌شود و درصورتی‌که منجر به بهبود نشود، با احتمالی این تغییر پذیرفته می‌شود. این احتمال در طول اجرای الگوریتم به‌مرور کاهش می‌یابد. در روش تکاملی مبتنی بر الگوریتم ژنتیک برای حل TSP چندین جواب کاندید در قالب یک نسل در نظر گرفته می‌شود و علاوه بر تغییرات مختصر بر روی اعضای یک نسل برای بهبود آن‌ها، از ترکیب جواب‌های خوب نیز برای تولید یک جواب جدید استفاده می‌شود. به طور کلی استفاده از روش‌های تکاملی بر روی نمونه‌های بزرگ TSP به علت سرعت پایین آنها، به عنوان اولویت اول مطرح نمی‌شود.

---

[9] Construction
[10] Improving
[11] Nearest neighbor
[12] Spanning trees
[13] Greedy Heuristic
[۱۴] این روش در بخش ۲-۳ با جزئیات بیشتری بیان شده است.
[15] subtour
[16] Partial tour
[17] Priority queue
[18] Quick Boruvka
[19] Simulated annealing

۲-۳- روش ابتکاری پس‌انداز و بهبودهای آن برای حل TSP با معیار فاصله‌ی اقلیدسی

یکی از روش‌های ابتکاری موفق برای حل TSP ابتکار پس‌انداز است [4]. این روش در اصل برای حل مسائلی در مسیریابی پیشنهاد شده است و دارای نسخه‌های متنوعی هست. الگوریتم (۲) نسخه‌ی اولیه این روش را توضیح می‌دهد.

1. **function** $tour$ = savings(V) {
2. $hub \leftarrow$ index of the most centeral city in V
3. $V_h \leftarrow \{1, \dots, n\} - hub$
4. **foreach** $i, j \in V_h,\ i < j$ **do**
5. $S[i,j] \leftarrow D(hub, i) + D(hub, j) - D(i,j)$
6. **while** $|V_h| > 2$ **do**
7. add (v_i, v_j) with the best $S[i,j]$ to $partial\ tour$ provided that no subtour is created
8. **if** degree of v_i is 2, remove i from V_h
9. **if** degree of v_j is 2, remove j from V_h
10. Stitch remaining two cities in V_h through the hub node
11. **return** $tour$
12. }

الگوریتم ۲- روش ابتکاری پس‌انداز برای حل TSP– نسخه‌ی اولیه

در الگوریتم ۲، در خط ۲ مرکزی‌ترین شهر از مجموعه‌ی V با عنوان گره قطب انتخاب می‌شود و اندیس آن در متغیر hub ذخیره می‌شود. اندیس همه‌ی شهرها به‌جز hub در خط ۳ در V_h ذخیره می‌شود. در خطوط ۴ و ۵، برای همه‌ی جفت شهرها در V_h مقداری به‌عنوان پس‌انداز محاسبه و در متغیر متناظر یعنی $S[i,j]$ ذخیره می‌شود. در خط ۵، تابع $D(i,j)$ نشان‌دهنده‌ی فاصله‌ی بین v_i و v_j است. در حلقه‌ی اصلی این روش، یعنی خطوط ۶ تا ۹، تا زمانی که تعداد عناصر V_h بیشتر از دو است، یال متصل‌کننده‌ی شهرهای i–ام و j–ام یعنی (v_i, v_j) که مقدار پس‌انداز مربوط به آن از همه بیشتر است، درصورتی‌که به ایجاد دوری منجر نشود، به مجموعه‌ی یال‌های دور ناکامل[20] اضافه می‌شود (خط ۷). در خط ۸ و ۹، اندیس مربوط به شهرهای v_i و v_j درصورتی‌که به‌طور کامل از دو طرف در $tour$ متصل شده باشند، از V_h حذف می‌شود. در پایان و در خط ۱۰، گره قطب با اندیس hub به دو گره باقیمانده که اندیس‌هایشان در V_h قرار دارد، متصل شده و دور را تکمیل می‌کند. دور تکمیل شده در خط ۱۱ برگشت داده می‌شود.

پیاده‌سازی این الگوریتم اولیه آسان است، ولی مرتبه‌ی زمان اجرای آن $O(n^2 \log n)$ است[21] و همچنین فضای موردنیاز برای ذخیره‌کردن مقادیر پس‌انداز از مرتبه‌ی $O(n^2)$ است. این بدان معناست که اجرای این الگوریتم برای مقادیر بزرگ n میسر نیست. بهبودهایی برای رفع این مشکلات پیشنهاد شده است. Golden و همکاران نشان داده‌اند که کافی است فقط برخی جفت گره های خاص را در ضمن ساخت دور در نظر بگیریم و بر این اساس یک روال

[20] Partial tour

[21] یادآوری می‌شود که n تعداد شهرها در مسئله‌ی موردبحث است.

مرتب‌سازی پس‌اندازها را با استفاده از ساختمان داده‌ی کپه[22] پیشنهاد کرده‌اند [14]. به‌علاوه Paessens روشی با عنوان «*یافتن تدریجی بیشترین مقدار پس‌انداز*[23]» ارائه کرده‌اند. در این روش ابتدا همه‌ی رکوردها بر اساس فاصله تا گره قطب مرتب می‌شوند و تنها آن دسته از جفت گره‌ها در محاسبه‌ی پس‌انداز وارد می‌شوند که فاصله‌ی هر دو گره آن‌ها تا گره قطب حداقل به‌اندازه‌ی نصف بیشترین پس‌انداز موجود باشد، زیرا در فضای اقلیدسی حداکثر مقدار پس‌انداز برای یک جفت گره همواره نابیشتر از ۲ برابر حداقل فاصله‌های آن گره‌ها تا گره قطب است. به‌بیان‌دیگر برای دو جفت گره با اندیس i و j همواره نامساوی $S[i,j] \le \min\{D(i,hub),D(j,hub)\}$ برقرار است [6]؛ بنابراین نیازی نیست که همه‌ی جفت گره‌ها را در ضمن ساخت دور در نظر گرفت. به‌علاوه مرتضوی و همکار روشی برای بهبود این ابتکار ارائه کرده‌اند که از ساختمان داده‌ی kd-tree به همراه مفهومی به نام گره فعال استفاده می‌کند [15]. در این روش برای محاسبه‌ی پس‌انداز به‌صورت تدریجی تنها نزدیک‌ترین همسایه‌ی فعال یک گره (یعنی همسایه‌ای با درجه‌ی کمتر از ۲ در دور ناکامل، برای جفت شدن مورد بررسی قرار می‌گیرد، زیرا اصولاً احتمال جفت شدن دو گره نزدیک به هم در دور با طول کمینه بیشتر است. تمامی این بهبودها با فرض این قابل‌استفاده هستند که فاصله‌ی بین گره‌ها در فضای اقلیدسی محاسبه می‌شود.

۲-۴- روش‌های سیستم تصویر

به‌طورکلی به روش قاعده‌مندی که برای تبدیل نقاطی بر روی سطح یک کره یا بیضی بر حسب طول و عرض جغرافیایی به نقاطی بر روی کاغذ استفاده می‌شود، سیستم تصویر می‌گویند [16]. همواره چنین تبدیلی همراه با نوعی اعوجاج[24] در مساحت، شکل، جهت، فاصله یا مقیاس خواهد بود. همچنین نحوه‌ی مدل کردن زمین (به‌صورت کره، بیضی یا ...) نیز در ویژگی‌های نقشه‌ی نهایی تأثیر خواهد داشت. در ادامه‌ی این مقاله و با توجه به نحوه‌ی محاسبه‌ی فاصله در مجموعه داده‌های موردنظر در مسئله‌ی TSP (بخش ۲-۱)، به دنبال روش‌های سیستم تصویر هم‌فاصله[25] و با فرض کروی بودن زمین هستیم.

در این مقاله از روش‌های سیستم تصویر هم‌فاصله Azimuthal equidistant، Equidistant Conic و Cassini استفاده شده است. در روش Azimuthal equidistant که توسط ابوریحان بیرونی در سال ۱۰۰۰ میلادی ارائه شده است، از یک نقطه مرکزی استفاده می‌شود که فواصل تا این نقطه حفظ می‌شوند. روش Equidistant Conic در سال ۱۰۰ میلادی پیشنهاد شده است و فواصل را بر روی نصف‌النهارها (همان فاصله‌ی مدارات استاندارد عرض‌های جغرافیایی)، حفظ می‌کند. روش Cassini که در سال ۱۷۴۵ توسط César-François Cassini de Thury ارائه شده است از نوع سیستم تصویر استوانه‌ای است و مقیاس‌ها را در طول نصف‌النهار و خطوط موازی با آن حفظ می‌کند.

۳- روش پیشنهادی

ایده‌ی اصلی این مقاله برای حل TSP با معیار فاصله‌ی جغرافیایی بر پایه‌ی تبدیل فضا هست. بدین منظور از روش‌های ترسیم نقشه که به‌نوعی فاصله‌ها را حفظ می‌کنند، برای تبدیل مختصات جغرافیایی شهرها به مختصات کارتزین در دو بعد استفاده می‌شود. برای حل نسخه‌ی دوبعدی، نسخه‌ی بهبودیافته‌ای از روش ابتکاری پس‌انداز استفاده می‌شود که می‌تواند با سرعت و کیفیت بالا نسخه‌هایی تا چند میلیون شهر از مسئله‌ی TSP را حل کند. در پایان می‌توان جواب نمونه‌های دوبعدی را به نسخه‌ی اولیه مسئله در فضای اصلی اعمال کرد تا طول مسیر جواب پیشنهادی به دست آید.

[22] heap
[23] iterative determination of the maximum saving value
[24] distortion
[25] equidistant

همان‌طور که در بخش ۲-۳ بیان شد، نسخه‌ی اولیه‌ی روش ابتکاری پس‌انداز برای حل TSP دارای مشکلاتی هست. به همین دلیل در این مقاله برای بهبود زمان و حافظه‌ی مصرفی این الگوریتم، بهبودهایی در آن داده شده است. فرض کنید $knn(i,k) \in \{1,\dots,n\}$ نشان‌دهنده‌ی اندیس k-امین نزدیک‌ترین همسایه‌ی قابل اتصال[۲۶] گره v_i در V (به‌جز خود v_i) باشد. این تغییرات علاوه بر موارد اشاره‌شده در بخش ۲-۳، شامل موارد زیر می‌شود:

(۱) در ابتدا یک فاز پیش‌پردازش بر روی گره‌ها به ترتیب عکس فاصله‌ی آنها تا گره قطب انجام می‌شود. در این فاز، تمامی نزدیک‌ترین شهرها به یکدیگر متصل می‌شوند، یعنی اگر $knn(i,1)=j$ و $knn(j,1)=i$ در این صورت یال (v_i,v_j) قبل از شروع ابتکار پس‌انداز به یکدیگر متصل می شوند.

(۲) در مورد گره‌های v_i و v_j که $j>i$، چنانچه $knn(i,1)=j$ و $knn(j,2)=i$ باشد و در مرحله‌ی (۱) گره j به گره دیگری متصل نشده باشد، در این صورت یال (v_i,v_j) به‌عنوان جزئی از دور نهایی به یکدیگر متصل می‌شوند.

درواقع این مراحل به‌نوعی تضمین می‌کنند که بهترین انتخاب برای گره v_i انجام شده است، زیرا طبیعتاً اتصال به نزدیک‌ترین همسایه‌ی v_i به‌عنوان یک تصمیم محلی بهترین انتخاب حریصانه[۲۷] برای v_i محسوب می‌شود. ادامه‌ی اجرا به‌صورت الگوریتم (۳) در تابع savings2 آمده است.

```
function tour=savings2(V) {
hub ← index of the most centeral city in V
V ← sort(V, hub)
KD ← kd-tree on V
H ← empty Max Heap
V_h ← {2, …, n}
idx ← n
while |V_h| > 2 do
while H is empty or D(idx, hub) ≥ H.top/2 and idx > 1 do
NN ← knn(idx, 1)
S ← D(hub, idx) + D(hub, NN) − D(idx, NN)
H.push(S)
idx ← idx − 1
(i, j) ← H.pop
if i ∉ V_h continue
if j ∈ V_h and adding(i, j) to partial tour does not create a subtour
add(i, j) to partial tour
if degree of v_j is 2, remove j from V_h
if degree of v_i is 2, remove i from V_h, and continue
NN ← knn(i, 1)
S ← D(hub, i) + D(hub, NN) − D(i, NN)
H.push(S)
Stitch remaining two cities in V_h through the hub node
return tour
}
```

الگوریتم ۳- نسخه‌ی بهبودیافته ابتکار پس‌انداز برای حل TSP

[۲۶] یعنی درجه‌ی گره همسایه کمتر از ۲ است.

[27] greedy

در خط ۱ مختصات گره‌ها به تابع تحویل می‌شود. در خط ۲ مرکزی‌ترین گره به نام قطب انتخاب می‌شود. در خط ۳، همه‌ی گره‌ها در V بر حسب فاصله از گره قطب به‌صورت صعودی مرتب می‌شوند به‌نحوی‌که گره قطب با اندیس ۱ در V شناخته می‌شود. در خطوط ۴ و ۵ به ترتیب ساختمان داده‌های kd-tree و کپه ساخته می‌شوند. در خط ۶ اندیس همه‌ی گره‌ها به‌جز گره قطب (با اندیس ۱) در V_h ذخیره می‌شود. در خط ۷ اشاره‌گر گره جاری برای محاسبه‌ی پس‌انداز به آخرین گره در V با اندیس n تنظیم می‌شود. حلقه‌ی اصلی برنامه در خطوط ۸ تا ۲۲ به اضافه‌کردن یال‌ها تا زمانی که اندازه‌ی V_h بیشتر از ۲ هست، می‌پردازد. ابتدا در یک حلقه‌ی داخلی تا زمانی که امکان بهبود بهترین پس‌انداز باشد (خط ۹)، اطلاعات پس‌اندازها در کپه به‌روز می‌شود. اندیس نزدیک‌ترین همسایه‌ی قابل اتصال به گره جاری با اندیس idx در متغیر NN ذخیره می‌شود (خط ۱۰) و مقدار پس‌انداز مربوط به یال (v_{idx}, v_{NN}) در S ذخیره می‌شود (خط ۱۱). این اطلاعات در کپه ذخیره می‌شوند (خط ۱۲) و مقدار idx برای اشاره به گره بعدی به‌روز می‌شود (خط ۱۳). در خط ۱۴ بهترین یال کاندید برای اضافه شدن به دور ناکامل در (i, j) ذخیره می‌شود و اطلاعات آن از بالای کپه حذف می‌شود. اگر v_i کاملاً متصل شده باشد و اندیس آن در V_h نباشد، ادامه‌ی حلقه نادیده گرفته می‌شود و مورد بعدی بررسی می‌شود (خط ۱۵). درصورتی‌که v_j به‌طور کامل متصل نشده باشد و افزودن یال (i, j) باعث ایجاد دور نشود، این یال به دور ناکامل اضافه می‌شود (خط ۱۷). در خط ۱۸ و ۱۹ اطلاعات V_h در صورت لزوم به‌روز می‌شود. در خطوط ۲۰ تا ۲۲ نیز اطلاعات مربوط به همسایگی گره v_i محاسبه و کپه به‌روز می‌شود. پس از پایان حلقه‌ی اصلی، خط ۲۳ وظیفه‌ی کامل کردن دور ناکامل را با افزودن یال‌هایی بین گره قطب و گره‌های با اندیس ذکرشده در V_h به عهده دارد. نهایتاً خط ۲۴ دور کامل شده را برمی‌گرداند.

۴- نتایج تجربی

در این بخش نتایج اجرای نسخه‌ی بهبودیافته‌ی ابتکار پس‌انداز بر روی نمونه‌های TSP با معیار فاصله‌ی جغرافیایی ارائه می‌شود. این ۱۰ نمونه از ۱۰۰۰۰ تا ۱۰۰۰۰۰ موقعیت جغرافیایی شهرهای کره‌ی زمین را در بر دارند و از ابتدای نمونه‌ی TSP جغرافیایی با عنوان world[28] با ۱۹۰۴۷۱۱ شهر انتخاب شده‌اند. اطلاعات این شهرها در قالب طول و عرض جغرافیایی ارائه شده است. کلیه‌ی محاسبات بر روی یک کامپیوتر شخصی با سیستم‌عامل ویندوز ۱۰ و پردازنده‌ی Core i7 - 1.60 GHz و ۸ گیگابایت حافظه‌ی اصلی انجام شده است. برای سیستم تصویر از نرم‌افزار ArcMap 10.3.1 و جعبه‌ابزار سیستم تصویر نرم‌افزار MATLAB استفاده شده است. در این مورد از مقادیر پیش‌فرض پارامترهای این نرم‌افزارها برای سیستم تصویر استفاده شده است.

نتایج ارائه‌شده در این بخش مربوط به سه نوع سیستم تصویری معرفی‌شده در بخش ۲-۴ یعنی Azimuthal، Cassini و Conic است. همچنین نتایج مربوط به طول دور در اجرای ابتکار پس‌انداز بهبودیافته با نتایج مربوط به نمونه‌ی اصلی (بدون سیستم تصویر) مبتنی بر روش‌های GH، NN و QB که در بخش ۲-۲ توضیح داده شد، در جدول ۱ مقایسه شده‌اند. پیاده‌سازی این روش‌ها در برنامه‌ای با عنوان linkern[29] ارائه شده است. زمان اجرای روش‌های فوق (بدون در نظر گرفتن زمان کار با دیسک) نیز در جدول ۲ ارائه شده است. ازآنجاکه زمان اجرای سیستم تصویر به ازای هر نمونه با اندازه‌ی ثابت، برای هر سه روش فوق بسیار کم و تقریباً یکسان بوده، تنها یک زمان اعمال سیستم تصویر ($t_{projection}$) همراه با زمان اجرای ابتکار پس‌انداز بهبودیافته ($t_{savings2}$) در جدول ۲ ارائه شده است.

[28] http://www.math.uwaterloo.ca/tsp/world/world.tsp.gz
[29] http://www.math.uwaterloo.ca/tsp/concorde/downloads/codes/cygwin/linkern.exe.gz

جدول ۱- طول دور روش savings2 مبتنی بر سیستم تصویر در مقایسه با سایر روش‌های ابتکاری بدون استفاده از سیستم تصویر (همه‌ی فاصله‌ها بر حسب متر هستند.)

اندازه (۱۰۰۰)	مبتنی بر سیستم تصویر			بدون سیستم تصویر		
	Azimutal	Cassini	Conic	L_{GH}	L_{NN}	L_{QB}
۱۰	۱۵۸۴۰۱۵۲۴	۱۵۴۶۰۳۶۳۵	۱۵۳۰۵۳۵۳۲	۱۶۷۷۹۸۹۳۹	۱۶۰۶۰۲۰۵۰	۱۳۲۲۶۱۹۷۸
۲۰	۲۴۷۵۹۴۶۲۱	۲۵۱۵۶۴۴۴۵	۲۴۷۵۱۱۴۶۶	۲۵۸۰۸۸۵۹۲	۲۴۹۷۷۹۸۸۵	۲۱۵۶۰۰۲۹۶
۳۰	۳۴۹۱۷۱۴۲۸	۳۵۲۴۳۶۲۰۳	۳۴۶۴۹۲۰۲۳	۳۶۶۵۱۵۹۵۲	۳۷۰۲۲۸۴۱۶	۳۱۷۷۳۴۳۹۸
۴۰	۴۱۶۸۱۵۵۰۴	۴۱۹۳۵۴۰۸۹	۴۱۴۸۷۶۷۷۲	۴۶۰۳۶۶۲۹۳	۴۴۷۷۶۴۵۶۳	۳۷۸۵۷۹۳۶۳
۵۰	۴۸۰۹۲۷۴۰۷	۴۸۱۵۰۵۶۲۱	۴۷۶۳۲۹۵۹۶	۵۴۷۰۰۹۸۰۳	۵۱۷۱۶۴۹۷۵	۴۳۹۹۷۳۴۶۵
۶۰	۵۴۳۸۱۵۹۹۷	۵۴۵۴۳۹۸۰۸	۵۴۰۹۰۴۶۵۲	۶۱۶۴۲۰۸۲۰	۵۹۳۷۴۵۹۰۵	۴۹۱۵۴۴۱۷۱
۷۰	۶۰۵۰۴۹۰۸۴	۶۰۶۲۹۷۵۶۳	۶۰۶۳۶۷۹۳۵	۶۷۵۰۲۱۸۸۰	۶۷۵۲۹۵۷۲۴	۵۵۶۶۵۵۹۶۷
۸۰	۶۷۳۷۷۶۷۸۸	۶۷۴۲۷۹۹۳۸	۶۷۴۹۰۱۱۰۸	۷۳۰۴۰۶۸۸۷	۷۴۵۳۱۵۱۹۰	۶۱۵۸۷۰۴۸۴
۹۰	۷۳۴۷۶۰۳۵۳	۷۳۴۷۲۳۸۳۳	۷۳۷۶۴۹۳۴۳	۸۱۴۴۱۵۳۲۱	۸۰۶۹۰۱۲۱۷	۶۸۳۲۲۴۱۷۷
۱۰۰	۷۹۳۳۲۲۸۲۰	۷۸۹۳۴۷۲۹۸	۷۹۱۷۸۳۹۳۷	۸۸۶۳۷۲۴۱۸	۸۷۲۹۹۳۱۴۰	۷۳۱۰۲۱۹۶۷

جدول ۲- زمان اجرای روش مبتنی بر سیستم تصویر در مقایسه با سایر روش‌های ابتکاری بدون استفاده از سیستم تصویر (همه‌ی زمان‌ها بر حسب ثانیه هستند.)

اندازه (۱۰۰۰)	مبتنی بر سیستم تصویر		بدون سیستم تصویر		
	$t_{projection}$	$t_{savings2}$	t_{GH}	t_{NN}	t_{QB}
۱۰	۰٫۰۰۱	۰٫۱۴	۰٫۲۷	۳۳٫۸۹	۰٫۷۳
۲۰	۰٫۰۰۱	۰٫۱۸	۰٫۸۷	۱۲۸٫۷۶	۲٫۹۳
۳۰	۰٫۰۰۱	۰٫۲۳	۱٫۸۵	۲۸۴٫۲۴	۶٫۲۱
۴۰	۰٫۰۰۲	۰٫۲۵	۳٫۲۱	۳۹۷٫۷۵	۱۰٫۶۸
۵۰	۰٫۰۰۲	۰٫۳۳	۴٫۹۱	۵۹۲٫۲۵	۱۶٫۳۹
۶۰	۰٫۰۰۲	۰٫۳۷	۶٫۷۶	۹۲۸٫۷۹	۲۳٫۴۳
۷۰	۰٫۰۰۲	۰٫۴۱	۹٫۰۴	۱۲۵۷٫۴۵	۳۰٫۹۴
۸۰	۰٫۰۰۲	۰٫۴۳	۱۱٫۶۱	۱۶۴۷٫۵۶	۴۰٫۹۳
۹۰	۰٫۰۰۳	۰٫۵۲	۱۴٫۹۵	۱۶۴۹٫۰۶	۴۹٫۵۲
۱۰۰	۰٫۰۰۳	۰٫۵۸	۱۸٫۲۷	۲۳۶۱٫۲۱	۶۰٫۶۷

نتایج جدول ۱ نشان می‌دهد که همواره بهترین جواب‌های روش مبتنی بر سیستم تصویر (روش مبتنی بر سیستم تصویر Azimutal)، از بهترین جواب سایر روش‌ها بهتر بوده است. این بهبود به‌طور میانگین بیش از ۹٪ بوده است. برای مثال طول دور نهایی روش مبتنی بر سیستم تصویر Azimutal به ازای $n = ۱۰۰۰۰$ نسبت به بهترین ابتکار بدون سیستم تصویر (یعنی QB) بهبود ۱۳٪ را نشان می‌دهد. به‌علاوه در بین ابتکارهای QB، NN و GH همواره کوتاه‌ترین دورها توسط روش QB به‌دست‌آمده‌اند که با توجه به زمان اجرای ثبت‌شده برای این روش در جدول ۲ در مقایسه با روش‌های NN و GH، آن را قابل‌قبول نشان می‌دهد. درهرصورت برتری مطلق به لحاظ زمان اجرا نیز با روش‌های

مبتنی بر سیستم تصویر است. درواقع این زمان اجرا (شامل سیستم تصویر و اعمال روش ابتکاری پس‌انداز بهبودیافته) همواره کمتر از ۱ ثانیه بوده که برتری مطلق روش پیشنهادی را نشان می‌دهد.

۵- نتایج و پیشنهادها

در این مقاله روشی برای حل مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی بر روی کره زمین ارائه شد. این روش در ابتدا داده‌های مکانی را با استفاده از سیستم تصویر از فضای جغرافیایی به فضای اقلیدسی منتقل می‌کند و سپس نسخه‌ی بهبودیافته‌ای از روش ابتکاری پس‌انداز را بر روی این نمونه‌ی جدید اعمال می‌کند. نتایج تجربی نشان‌دهنده‌ی برتری روش پیشنهادی به لحاظ طول دور جواب و نیز زمان اجرای الگوریتم در مقایسه با روش‌های موجود است. استفاده از مقیاس‌گذاری چندبعدی (MDS[۳۰]) برای تبدیل فضای مسئله‌ی فروشنده‌ی دوره‌گرد با معیار فاصله‌ی جغرافیایی یکی از روش‌های توسعه‌ی ایده‌ی این مقاله است که در کارهای آتی به آن توجه خواهد شد.

مراجع

[1] E. L. Lawler, The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization. John Wiley & Sons, 1985.

[2] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, The Traveling Salesman Problem: A Computational Study: A Computational Study. Princeton university press, 2011.

[3] C. Rego, D. Gamboa, F. Glover, and C. Osterman, “Traveling salesman problem heuristics: leading methods, implementations and latest advances,” European Journal of Operational Research, vol. 211, no. 3, pp. 427–441, 2011.

[4] G. Clarke and J. Wright, “Scheduling of vehicles from a central depot to a number of delivery points,” Operations research, vol. 12, no. 4, pp. 568–581, 1964.

[5] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” Communications of the ACM, vol. 18, no. 9, pp. 509–517, 1975.

[6] H. Paessens, “The savings algorithm for the vehicle routing problem,” European Journal of Operational Research, vol. 34, no. 3, pp. 336–344, 1988.

[7] H. Okano, S. Misono, and K. Iwano, “New TSP construction heuristics and their relationships to the 2-Opt,” Journal of Heuristics, vol. 5, no. 1, pp. 71–88, 1999.

[8] J. L. Bentley, “Experiments on traveling salesman heuristics,” in Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms, 1990, pp. 91–99.

[9] D. S. Johnson and L. A. McGeoch, “Experimental analysis of heuristics for the STSP,” in The traveling salesman problem and its variations, Springer, 2007, pp. 369–443.

[10] G. A. Croes, “A method for solving traveling-salesman problems,” Operations research, vol. 6, no. 6, pp. 791–812, 1958.

[11] S. Rees and R. Ball, “Criteria for an optimum simulated annealing schedule for problems of the travelling salesman type,” Journal of Physics A: Mathematical and General, vol. 20, no. 5, p. 1239, 1987.

[30] Multidimensional Scaling

[12] J. Knox and F. Glover, Comparative Testing of Traveling Salesman Heuristics Derived from Tabu Search, Genetic Algorithms, and Simulated Annealing. 1989.

[13] K. Helsgaun, “General k-opt submoves for the Lin–Kernighan TSP heuristic,” Mathematical Programming Computation, vol. 1, no. 2–3, pp. 119–163, 2009.

[14] B. L. Golden, T. L. Magnanti, and H. Q. Nguyen, “Implementing vehicle routing algorithms,” Networks, vol. 7, no. 2, pp. 113–148, 1977.

[15] R. Mortazavi and S. Jalili, “Fast data-oriented microaggregation algorithm for large numerical datasets,” Knowledge-Based Systems, vol. 67, pp. 195–205, 2014.

[16] J. Snyder, “Album of Map Projections, United States Geological Survey Professional Paper,” United States Government Printing Office, vol. 1453, 1989.

Using map projection for solving geographic instances of Travelling Salesman Problem

Reza Mortazavi[*]

1- School of Engineering, Damghan University, Damghan, Iran

Abstract

One of the most important optimization problems in Network Analysis is Travelling Salesman Problem. This is due to its practical usages, especially in geospatial information technology. It is proved that the problem is NP-Hard, but many heuristic methods are presented for practical applications. In this paper, the map projection method is utilized to transform a geographic instance of travelling salesman problem into a simpler Euclidean one, such that traditional heuristics can be applied more successfully. The results confirm the superiority of the proposed method with respect to both the tour length and running time of the method in comparison with similar techniques.

Keywords: Travelling Salesman Problem, Network Analysis, Map Projection, Geospatial Information Technology.

[*] Contact Information: Dr. Reza Mortazavi, Computer Engineering Department, School of Engineering, Damghan University, Damghan, Iran, EMAIL: r_mortazavi@du.ac.ir, TEL: +98 233 522 0081-6 (320)