الگوریتم تعمیم‌یافته برای مدل مکان‌یابی ۲-میانه ناخوشایند روی کاکتوس‌ها

لیلامدبر *، بهروز علیزاده

گروه ریاضی کاربردی، دانشکده علوم پایه مهندسی، دانشگاه صنعتی سهند، تبریز.
*دانشجوی کارشناسی ارشد.
modabber۱۳۹۲@gmail.com

چکیده

در این مقاله هدف این است که یک الگوریتم تعمیم‌یافته جدید بر مبنای ایده کلی کانگ (L. Kang, 2014) برای مدل مکان‌یابی ۲-میانه ناخوشایند روی کاکتوس‌گراف‌ها ارائه دهیم. این الگوریتم نسبت به روش ارائه شده توسط کانگ ساده بوده و بطور قابل ملاحظه دارای عملیات محاسباتی کمتری می‌باشد.

کلمات کلیدی نظریه مکان‌یابی؛ مدل میانه ناخوشایند؛ بهینه‌سازی ترکیبیاتی؛ کاکتوس‌گراف

۱ مقدمه و بیان مسأله

مسائل مکان‌یابی از جمله مسائل بهینه‌سازی ترکیبیاتی می‌باشند که به دلیل داشتن کاربردهای فراوان در تئوری و عمل همواره مورد توجه محققان بوده‌اند. این مسائل عمدتاً به دو دسته خوشایند و ناخوشایند تقسیم می‌شوند. در مسائل ناخوشایند هدف پیدا کردن بهترین مکان برای سرویس‌دهنده‌هایی است که علیرغم ارائه خدمات دارای مضراتی هستند و باید تا حد ممکن به دور از مشتریان مکان‌یابی شوند. از جمله کاربردهای این نوع مسائل می‌توان به مکان‌یابی بهینه محل دفن زباله‌های شهری اشاره کرد. فرض کنید $G=(V,E)$ یک گراف همبند با مجموعه رأسی V و مجموعه یالی E باشد به طوریکه برای هر رأس v، یک وزن نامنفی $w(v)$ و برای هر یال e یک طول مثبت $l(e)$ تخصیص داده شده است. برای هر دو نقطه x و y در G کوتاهترین مسیر از x به y را با $P[x,y]$ و طول آن را با $d(x,y)$ نشان می‌دهیم. با فرض آنکه رأس‌های گراف G مکان مشتریان موجود باشد، آنگاه در یک مدل مکان‌یابی$p-$ میانه ناخوشایند(ماکسین) روی گراف G هدف این است که تعداد p نقطه $x_1^*,...,x_p^*$ روی G پیدا کنیم بطوریکه یک جواب بهین برای مدل بهینه‌سازی زیر باشد:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \max_{j=1,2,\dots,p} w_i d(v_i,x_j) \qquad (\mathrm{LOC_P}) \\ \text{s.t.} \quad & x_1,...,x_p \in G. \end{aligned}$$

درسال ۱۹۸۴، تینق یک الگوریتم با زمان اجرای خطی برای مسأله $\mathrm{LOC_1}$ روی شبکه‌های درختی ارائه داد[۵]. بورکارد و همکاران[۱] ثابت کردند که مدل $\mathrm{LOC_p}$ در زمان خطی روی درخت‌ها می‌تواند حل گردد. رویکردهای جواب خطی برای مسأله $\mathrm{LOC_p}$ روی گراف‌های بلوکی و بازه‌ای در[۳، ۱] پیشنهاد گردید. اخیرا در سال ۲۰۱۴، کانگ و همکاران[۲] مدل $\mathrm{LOC_2}$ را روی کاکتوس‌گراف‌ها مطرح کرده و نشان دادند که این مسأله در زمان $O(n^2)$ قابل حل است. در این مقاله هدف ما این است که یک الگوریتم تعمیم‌یافته جدید با محاسبات کمتر، بر مبنای ایده کلی کانگ برای مدل $\mathrm{LOC_2}$ روی کاکتوس‌گراف‌ها ارائه دهیم.

۲ رویکرد جواب روی کاکتوس گراف‌ها

در این مقاله فرض کنید گراف G داده شده یک کاکتوس باشد. رأس w رأس مجرد مسیر $P[u,v]$ روی G است، هرگاه روی هیچ دوری قرار نداشته باشد، یا روی دو دوری که شامل برخی از یال‌های مسیر $P[u,v]$ هستند قرار گیرد. اگر m_{uv} نقطه وسط مسیر داده شده $P[u,v]$، روی دوری مانند C قرارگرفته و رأس مجرد روی G نباشد، آنگاه نقطه وسط دیگری مانند m'_{uv} روی C وجود دارد که آن را نقطه وسط دوم این مسیر گوییم. فرض کنید $P[a,b]$ یک مسیر قطری با نقطه وسط m_{ab} روی G بوده و m_{ab} یک رأس مجرد $P[a,b]$ نباشد. در این صورت m_{ab} روی

دوری مانند C قرار می‌گیرد. همچنین فرض کنید C دارای تعداد γ رأس باشد که در جهت عقربه‌های ساعت شماره‌گذاری شده‌اند، و برای هر رأس $v_i \in C$، $v_{i+\frac{\gamma}{2}}$ نشان‌دهنده رأسی از C باشد بطوریکه $d(v_i, v_{i+\frac{\gamma}{2}}) = \frac{L(C)}{2}$. با حذف یال‌های $E(C)$ از G، γ زیرکاکتوس $G_1, ..., G_\gamma$ حاصل می‌شود به طوریکه برای هر $\gamma ,... ,1 = i$، داریم $v_i \in V(G_i)$. برای هر $i = 1, ..., \gamma$، فرض کنید t_i رأسی از G_i باشد که بیشترین فاصله را با v_i دارد، این رأس بعنوان کران G_i نامیده شده و در زمان خطی قابل محاسبه است. برای هر زیرگراف G_i و رأس v_i تعریف کنید:

$$W(G_i) = \sum_{v \in G_i} w(v).$$

می‌توان $W(G_i)$ را در زمان خطی محاسبه کرد. [۲].

لم ۱. *(کانگ و همکاران -[۲]). فرض کنید $X = \{u, v\}$ یک جواب بهین برای مدل LOC_2 ، روی کاکتوس G باشد. نقطه وسط مسیر $P[u, v]$ روی همان دوری قرار می‌گیرد که نقطه وسط مسیر قطری $P[a, b]$ روی آن قرار گرفته است.*

حال از لم ۱ ، نتیجه می‌شود برای حل مدل LOC_2 روی کاکتوس G کافی است ابتدا مجموعه همه زوج‌کران‌هایی را پیدا کنیم که نقطه وسط آن‌ها روی دور C قرار می‌گیرند. سپس مقدار تابع هدف مدل LOC_2 را بازای تمامی این زوج‌کران‌های کاندید، محاسبه کنیم. یک زوج‌کران با بزرگترین مقدار هدف متناظر بعنوان یک جواب بهین مدل LOC_2 در نظر گرفته می‌شود.

۱.۲ پیدا کردن زوج‌کران‌های کاندید

در این بخش الگوریتمی برای پیدا کردن زوج‌کران‌های کاندید ارائه می‌دهیم که نسبت به الگوریتم ارائه شده در [۲]، بسیار ساده بوده و دارای عملیات محاسباتی کمتری می‌باشد. به ازای $i = 1, ..., \gamma$ تعریف کنید $S(i) = \{d(t_i, t_j) : j = i+1, ..., i+\frac{\gamma}{2}\}$.
با در دست داشتن مجموعه $S(i)$ الگوریتم پیشنهادی ما برای محاسبه زوج‌کران‌های کاندید به صورت زیر خلاصه می‌شود:

الگوریتم ۲ پیدا کردن زوج‌کران‌های کاندید.

۱.بازای $i = 1, ..., \gamma$ عملیات زیر را انجام دهید:
۲.	قرار دهید $D_i = \emptyset$
۳.	به ازای هر $j = i+1, ..., i+\frac{\gamma}{2}$ انجام دهید:
۴.	اگر $d(v_i, t_i) \leqslant \frac{d(t_i,t_j)}{2}, d(v_j, t_j) \leqslant \frac{d(t_i,t_j)}{2}$ ، آنگاه
۵.	قرار دهید $D_i = D_i \bigcup \{(t_i, t_j)\}$
۶. مجموعه $\bigcup_{i=1}^{\gamma} D_i$ زوج‌کران‌های کاندید را مشخص می‌کند.

قضیه ۱. *مجموعه زوج‌کران‌های کاندید می‌تواند در زمان $O(n^2)$ توسط الگوریتم ۱ محاسبه گردد.*

محل دقیق نقطه وسط زوج‌کران‌های کاندید روی دور C به راحتی در زمان $O(n^2)$ قابل تشخیص است.

۲.۲ محاسبه مقدار تابع هدف به ازای تمام زوج‌کران‌های کاندید

با توجه به اینکه تعداد $O(n^2)$ زوج‌کران کاندید وجود دارد. لذا محاسبه مستقیم مقدار تابع هدف بازای این زوج‌کران‌ها زمان $O(n^3)$ را می‌طلبد. در این بخش یک روش مؤثر پیشنهاد می‌کنیم که می‌تواند تمامی مقادیر هدف مورد نظر را در زمان $O(n^2)$ محاسبه کند. روش ما نسبت به روش ارائه شده در [۲] بسیار ساده بوده و دارای عملیات محاسباتی کمتری است. حال زوج کران دلخواه $\{t_r, t_s\}$ را در نظر گرفته و تعریف کنید:

$$V_{\geq}(C) = \{v_i \in V(C) | d(v_i, t_r) \geq d(v_i, t_s)\},$$
$$V_{<}(C) = \{v_i \in V(C) | d(v_i, t_r) < d(v_i, t_s)\},$$
$$F'(t_r, t_s) = \sum_{v_i \in V_{\geq}(C)} W(G_i) d(v_i, t_r) + \sum_{v_i \in V_{<}(C)} W(G_i) d(v_i, t_s).$$

با بازنویسی تابع هدف مدل LOC۲، نشان داده داده‌ایم که برای حل LOC۲، روی کاکتوس‌گراف G کافی است بیشینه مقدار تابع F' را به ازای تمام زوج‌کران‌های کاندید پیدا کنیم. تعریف کنید:

$$F_{rs} = \sum_{v_i \in V_{\geq}(C)} W(G_i) d(v_i, t_r)$$
$$F_{sr} = \sum_{v_i \in V_{<}(C)} W(G_i) d(v_i, t_s)$$

در این صورت می‌توان نوشت $F'(t_r, t_s) = F_{rs} + F_{sr}$.
برای هر $i = 1, \ldots, \gamma$، فرض کنید $\{(t_i, t_{i_0}), \ldots, (t_i, t_{i_{k_i}})\}$ مجموعه زوج‌کران‌های کاندیدی باشد که $i_j \leqslant i + \frac{\gamma}{2}$ و این زوج‌کران‌ها بر حسب نقطه وسط اولشان با شروع از $v_{i+\frac{\gamma}{2}}$ در خلاف جهت عقربه‌های ساعت مرتب شده‌اند. همچنین فرض کنید نقطه وسط دوم زوج‌های فوق با شروع از $v_{i+\frac{\gamma}{2}}$ و در جهت عقربه‌های ساعت با نمادهای $m'_{i_0}, \ldots, m'_{ik_i}$ نشان داده شوند. برای هر $i = 1, \ldots, \gamma$، دنباله‌های $\{A_k\}_0^{k_i}$ و $\{B_k\}_0^{k_i}$ را به صورت زیر در زمان $O(n)$ بسازید:

$$A_0 = 0, \qquad B_0 = \sum_{v_z \in P[m_{ii_0} \quad m'_{i_0}]} W(G_z) d(v_z, t_i)$$
$$B_k = B_{k-1} + \sum_{v_z \in P[m_{ii_{k-1}} \quad m_{ii_k}]} W(G_z) d(v_z, t_i) \qquad k \geq 1$$
$$A_k = A_{k-1} + \sum_{v_z \in P[m'_{ik-1} \quad m'_{ik}]} W(G_z) d(v_z, t_i) \qquad k \geq 1$$

برای هر $k = 0, \ldots, k_i$، با فرض اینکه $m'_{ik} = m'_{ii_j}$، قرار دهید $B'_j = A_k$. بنابراین بازای هر $j = 0, \ldots, k_i$، خواهیم داشت:

$$F_{ii_j} = B_j + B'_j.$$

با تکرار روند فوق برای هر زوج‌کران کاندید (t_i, t_{i_j}) با $i_j > i + \frac{\gamma}{2}$، مقدار F_{ii_j} برای هر زوج (t_i, t_{ii_j}) در زمان $O(n)$ محاسبه می‌شود.
با بررسی هر کران t_i از C می‌توان مقادیر F_{ij} و F_{ji} را برای هر زوج‌کران کاندید (t_i, t_j) در زمان $O(n^2)$ بدست آورده و با جمع آنها مقدار هدف $F'(t_i, t_j)$ را محاسبه کرد. بنابراین با روش‌های ارائه شده در این مقاله می‌توان نتیجه گرفت:

قضیه ۲. مدل LOC۲ روی کاکتوس‌گراف‌ها می‌تواند توسط روش ارائه شده در این مقاله در زمان $O(n^2)$ حل گردد.

مراجع

[1] *R. E. Burkard, J. Fathali and H. T. Kakhki* p-maxian problem on tree, *Operations Research Letters* ***35*** *(2007), 331–335.*

[2] *Y. K. Cheng and L. Y. Kang,* The p-maxian problem on interval graphs, *Discrete Applied Mathematics* ***158*** *(2010), 1986–1993.*

[3] *L. Kang, C. Bai, E. Shan and K. Nguyen,* The 2-maxian problem on cactus graphs, *Discrete Optimization* ***13*** *(2014), 16–22.*

[4] *L. Y. Kang and Y. K. Cheng,* The p-maxian problem on block graphs, *Combinatorial Optimization* ***20*** *(2008), 131–141.*

[5] *S. S. Ting,* A linear-time algorithm for maximum facility location on tree networks, *Transportation Science* ***18*** *(1984), 76–84.*