

رائه يك الگوریتم برای یافتن کوتاه‌ترین مسیر در شبکه‌های حلقوی

اصغر عینی *

امیر صالحی پور **

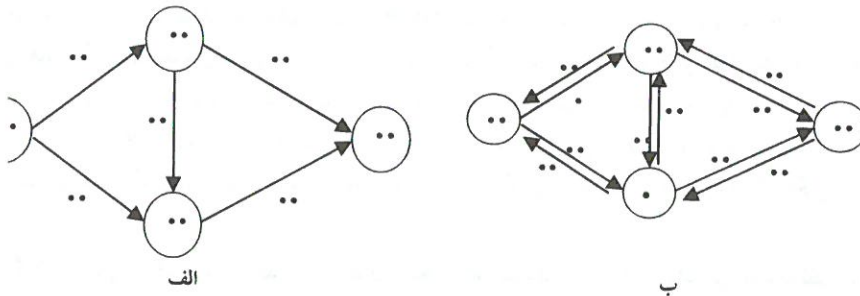
کیده

برای یافتن کوتاه‌ترین مسیر بین هر دو گره در شبکه‌های دارای حلقه که در آن حداقل یک حلقه بود دارد الگوریتم فلوید - وارشال (Floyd-Warshall) به‌عنوان پرکاربردترین الگوریتم مطرح است. در مقاله الگوریتم جدیدی با عنوان الگوریتم مستطیلی توسعه داده می‌شود که به‌طور قابل ملاحظه‌ای حجم اسباب موردنیاز را نسبت به الگوریتم فلوید - وارشال کاهش می‌دهد. علاوه بر این، روش ارائه شده از ساده‌تر و قابل فهم‌تر از الگوریتم فلوید - وارشال است که این خود می‌تواند به‌عنوان یک مزیت گگ در حوزه آموزشی محسوب شود. نحوه به‌کارگیری الگوریتم جدید در قالب مثال کوچکی بررسی شود.

گان کلیدی: الگوریتم فلوید - وارشال، الگوریتم مستطیلی، روش آبخاری تجدیدنظر شده، که‌های کوتاه‌ترین مسیر دارای حلقه

مقدمه

مسئله کوتاه‌ترین مسیر، یکی از مسائل معروف و پرکاربرد در تحقیق در عملیات است. در این مسئله، هدف یافتن مسیر بین دو گره از یک گراف است به طوری که مجموع وزن‌های (هزینه، زمان، فاصله و دیگر معیارها) بردارهای اتصال‌دهند حداقل شود [۴، ۶ و ۸]. این مسئله دارای کاربردهای متنوعی در دنیای واقعی است یکی از این کاربردها را می‌توان در یافتن کوتاه‌ترین مسیر در شبکه راه‌ها نام برد (شبکه کوتاه‌ترین مسیر). شکل ۱- الف چنین شبکه‌ای (گرافی) که دارای ۴ گره و ۵ کمان (جهت حرکت بین هر دو گره) است را نشان می‌دهد. گره‌های ۱ و ۴ به ترتیب مبدا و مقصد حرکت هستند. با توجه به این شکل کوتاه‌ترین مسیر باید از گره ۳ عبور کرده و دارای هزینه ۵ باشد.



شکل ۱. دو شبکه ساده، الف. بدون حلقه. ب. دارای حلقه

در یک تقسیم‌بندی، شبکه‌های کوتاه‌ترین مسیر به شبکه‌های بدون حلقه (سیکل) و شبکه‌های دارای حلقه تقسیم می‌شوند. شکل ۱- الف شبکه‌های کوتاه‌ترین مسیر بدون حلقه را نشان می‌دهد. شبکه‌های کوتاه‌ترین مسیر دارای حلقه نوع جامع کامل تری از شبکه‌های کوتاه‌ترین مسیر هستند (شکل ۱- ب). در این نوع شبکه حداقل یک حلقه وجود دارد. این نوع شبکه‌ها فاقد گره شروع و پایان مشخص هستند و لذا هر گره می‌تواند گره شروع یا گره پایان باشد [۲]. در این شبکه‌ها معمولاً

مسیرها از هر دو گره دو طرفه می‌باشند و به همین دلیل آنها را شبکه‌های دارای حلقه می‌نامند. برای حل شبکه‌های کوتاه‌ترین مسیر دارای حلقه (یافتن کوتاه‌ترین مسیر) الگوریتم‌های متفاوتی وجود دارد که الگوریتم فلویید - وارشال (الگوریتم آبشاری تجدیدنظرشده^۱) یکی از مهم‌ترین و پر استفاده‌ترین این الگوریتم‌ها است. برای مطالعه بیشتر در خصوص الگوریتم‌های موجود و مقایسه آنها در حل مسئله کوتاه‌ترین مسیر (با حلقه و بدون حلقه) می‌توان به چرکاسکی و همکاران (۱۹۹۶) و گالو و پالوتینو (۱۹۸۸) مراجعه نمود [۵ و ۹].

علاوه بر الگوریتم‌های حل بهینه ارائه شده برای مسئله کوتاه‌ترین مسیر، کاربردهای عملی مسئله بعضاً منجر به توسعه الگوریتم‌های ابتکاری برای مسئله شده‌است. در این راستا می‌توان به فوا و همکاران (۲۰۰۶) رجوع نمود که در آن بررسی و مطالعه جامعی از الگوریتم‌های ابتکاری برای مسئله کوتاه‌ترین مسیر آورده شده‌است [۱۱]. منظور مطالعه در ارتباط با رویکردهای بهینه به کار گرفته شده به ویژه در کاربردهای عملی می‌توان به آلتین و همکاران (۲۰۰۹) و چرکاسکی و همکاران (۱۹۹۶) رجوع نمود [۱۲ و ۵]. کلاندر و پست (۲۰۰۶) عملکرد و سرعت ۷ الگوریتم حل مسئله کوتاه‌ترین مسیر در شبکه‌های حمل و نقل واقعی (دنیای واقعی) را مقایسه نموده‌اند [۱۳].

همان‌طور که در ابتدا نیز شرح داده شد مسئله کوتاه‌ترین مسیر دارای کاربردهای متنوعی در دنیای واقعی است. گاماچه و همکاران و رویست و همکاران به دو کاربرد بسیار جالب از مسئله در حوزه معادن زیرزمینی [۱۴] و مسیریابی هواپیما [۱۵] اشاره نموده‌اند. در کاربرد اول مسئله مسیریابی و زمان‌بندی ماشین‌های معدن به طوری که در سریع‌ترین زمان بتوانند کارهای محوله را انجام دهند، به وسیله مسئله کوتاه‌ترین مسیر مدل‌سازی شده‌است. کاربرد دوم اشاره به برنامه‌ریزی ماموریت‌ها به هنگام حملات نظامی دارد. تابع هدف، حداقل‌سازی خطرات حملات موشکی و محدودیت‌ها، مصرف سوخت و زمان پرواز می‌باشند.

در این مقاله، نویسندگان شبکه‌های کوتاه‌ترین مسیر دارای حلقه را مورد توجه

قرار داده و با ارائه الگوریتم جدیدی با عنوان "الگوریتم مستطیلی"، حجم محاسبات الگوریتم فلوید - وارشال را به طور قابل توجهی کاهش داده‌اند. علاوه بر این نوآوری مهم، درک و فهم الگوریتم جدید نیز بسیار آسان‌تر از الگوریتم فلوید- وارشال است. ساختار ارائه مقاله مطابق زیر است. در بخش دوم الگوریتم فلوید - وارشال تشریح می‌گردد. نوآوری این مقاله در بخش سوم مورد بررسی قرار می‌گیرد و الگوریتم مستطیلی به طور کامل شرح داده می‌شود. در بخش چهارم با ارائه مثالی عملکرد الگوریتم مستطیلی در حل شبکه‌های کوتاه‌ترین مسیر دارای حلقه بررسی می‌شود. همان‌طور که خواهیم دید حجم و زمان انجام محاسبات در الگوریتم مستطیلی به طور قابل ملاحظه‌ای نسبت به الگوریتم فلوید- وارشال کاهش یافته‌است. مقاله با نتیجه‌گیری و بیان تحقیقات آتی خاتمه می‌یابد.

الگوریتم فلوید - وارشال (روش آبشاری تجدیدنظرشده)

شبکه $N(V, A)$ با مجموعه گره‌های $V = \{1, 2, \dots, n\}$ ($|V| = n$) و مجموعه کمان‌های $A = \{(i, k) : i, k \in V, i \neq k\}$ را در نظر بگیرید. اگر این شبکه دارای حلقه باشد الگوریتم فلوید - وارشال بهترین و کامل‌ترین روش در بین الگوریتم‌های موجود برای به دست آوردن کوتاه‌ترین فاصله و مسیر مربوط به آن بین هر دو گره دلخواه i و k می‌باشد [۷ و ۱۰]. این الگوریتم براساس یک فرآیند ۴ مرحله‌ای است که در آن دو ماتریس D_j و R_j برای $j = 0, \dots, n$ محاسبه می‌شوند. این دو ماتریس به ترتیب مقادیر کوتاه‌ترین هزینه و کوتاه‌ترین مسیر (مبداء و مقصد) را در خود ذخیره می‌کنند. بنابراین، این الگوریتم کوتاه‌ترین هزینه و مسیر بین هر دو گره دلخواه در یک شبکه دارای حلقه را پس از $n+1$ مرحله نتیجه می‌دهد. باید در نظر داشت اگرچه محاسبات الگوریتم ساده است ولی به دلیل تعداد بسیار بالای این محاسبات، زمان محاسباتی الگوریتم به طور قابل توجهی بالا می‌باشد [۱ و ۲]. الگوریتم ۱ الگوریتم فلوید - وارشال را تشریح می‌نماید.

الگوریتم ۱. الگوریتم فلویید - وارشال برای حل مسئله کوتاه‌ترین مسیر دارای حلقه

گام ۱. اگر n برابر تعداد گره‌های شبکه باشد، دو ماتریس مربع $n \times n$ ، D_j و R_j به‌ازای $j = 0, \dots, n$ که در آن j را مرحله نامند، تعریف کنید.

گام ۲. به‌ازای $j = 0$ عناصر ماتریس‌های D_0 و R_0 را به‌ترتیب زیر محاسبه نمایند.
عناصر ماتریس D_0

d_{ik} اگر بردار مستقیم از گره i به گره k وجود داشته باشد
 ∞ اگر بردار مستقیم از گره i به گره k وجود نداشته باشد
صفر اگر $i = k$
عناصر ماتریس R_0

k اگر بردار مستقیم از گره i به گره k وجود داشته باشد
- اگر بردار مستقیم از گره i به گره k وجود نداشته باشد
- اگر $i = k$

گام ۳. برای $j = 1, \dots, n$ ماتریس‌های D_j و R_j را مطابق زیر محاسبه کنید.
عناصر ماتریس D_j

d_{ik} اگر $k = j$ و $i = j$ ، $i = k$
 $\min(d_{ik}, d_{ij} + d_{jk})$ در غیر اینصورت
عناصر ماتریس R_j
 k اگر $k = j$ و $i = j$ ، $i = k$
 k اگر $d_{ik} \leq d_{ij} + d_{jk}$
 j اگر $d_{ik} > d_{ij} + d_{jk}$

گام ۴. گام ۳ را تا جایی انجام دهید که ماتریس‌های D_n و R_n حاصل شوند.

نکات بیشتر در خصوص حجم محاسبات الگوریتم فلویید - وارشال در بخش ۴ آورده شده‌است. در اینجا از آوردن جزئیات بیشتر در خصوص این الگوریتم صرف نظر می‌شود. جهت مطالعات بیشتر می‌توان به منابع [۵ و ۹] مراجعه نمود.

الگوریتم مستطیلی

نوآوری اصلی این مقاله، در قالب طراحی و ارائه یک الگوریتم بهینه جدید برای حل شبکه‌های کوتاه‌ترین مسیر دارای حلقه (یافتن کوتاه‌ترین مسیر)، در این بخش

آورده شده است. علاوه بر اینکه الگوریتم پیش رو حجم و زمان لازم برای محاسبات را به طور چشمگیری کاهش می دهد، به کارگیری آن نیز بسیار آسان تر از الگوریتم فلویید - وارشال است. از طرفی دارای این مزیت است که آموزش آن بسیار آسان تر از الگوریتم فلویید - وارشال می باشد. توجه داشته باشید که الگوریتم مستطیلی بر پایه الگوریتم فلویید - وارشال طراحی شده است، با این تفاوت مهم که الگوریتم بر پایه ترسیم مجموعه ای از مستطیل ها جهت انجام محاسبات می باشد (به همین دلیل نام الگوریتم مستطیلی برای آن انتخاب شده است). الگوریتم ۲ مراحل الگوریتم مستطیلی را شرح می دهد.

الگوریتم ۲. الگوریتم مستطیلی برای مسئله کوتاه ترین مسیر دارای حلقه

گام ۱. اگر n برابر تعداد گره های شبکه باشد، دو ماتریس مربع $n \times n$ ، D_j و R_j به ازای $n, \dots, 0 = j$ که در آن j را مرحله نامند، تعریف کنید.

گام ۲. ماتریس های D_0 و R_0 را همانند گام ۲ الگوریتم فلویید - وارشال محاسبه نمایید ($j = 0$).

گام ۳. برای $j = 1, \dots, n$ ماتریس های D_j را مطابق زیر محاسبه کنید.

الف. اگر هر سطر / ستون در ماتریس D_j شامل ∞ باشد، سطر / ستون مجاور آن تغییر نخواهد کرد. بنابراین مقادیر این درآیه ها دقیقاً همانند ماتریس D_{j-1} ($j \geq 1$) خواهند بود (رویه بهبود عملکرد ۱).

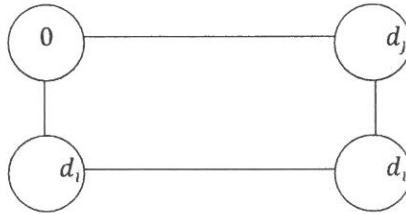
ب. اگر با به کارگیری بند الف. هنوز درآیه ای مجهول وجود داشت، توسط ترسیم مجموعه ای از مستطیل ها آنرا محاسبه کنید (مطابق شکل ۲). در حقیقت می توان گفت که برای محاسبه هر آرایه d_{ik} از ماتریس D_j یک مستطیل مورد نیاز خواهد بود.

ج. ماتریس R_j را مطابق مقابل محاسبه کنید. اگر درآیه ای در ماتریس D_j تغییر نکند قطعاً ماتریس R_j نیز تغییر نخواهد کرد و اگر درآیه ای در ماتریس D_j تغییر کند درآیه متناظر در ماتریس R_j با j جایگزین می شود. این مهم را با مقایسه $\min(d_{ik}, d_{ij} + d_{jk})$ در ماتریس D_j با $d_{ij} + d_{jk} < d_{ik}$ در ماتریس R_j می توان دریافت. به عبارت بهتر ماتریس R_j به ماتریس D_j وابسته است. (رویه بهبود عملکرد ۲). توجه داشته باشید که در اینجا ماتریس R_j براساس ماتریس D_j محاسبه خواهد شد.

گام ۴. گام ۳ را تا جایی انجام دهید که ماتریس های D_n و R_n حاصل شوند.

رویه ترسیم مستطیل

در هر مرحله $j = 1, \dots, n$ برای هر درآیه d_{ik} (سطر i و ستون k ، و به جزء صفرهای قطر اصلی)، یک مستطیل با استفاده از صفر قطر اصلی مرحله j ترسیم می شود. همان طور که شکل ۲ نشان می دهد این صفر گوشه بالای چپ مستطیل را تشکیل می دهد.



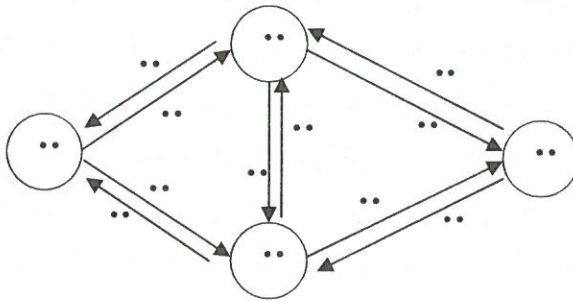
شکل ۲. نحوه ایجاد مستطیل در الگوریتم مستطیلی. در این شکل صفر، همان صفر قطر اصلی مرحله i می‌باشد.

همانند الگوریتم فلویید - وارشال، در الگوریتم مستطیلی نیز رابطه $d_{ik} = \min(d_{ik}, d_{ij} + d_{jk})$ برقرار است. توجه داشته‌باشید که برای محاسبه درآیه d_{ik} از ترسیم مستطیل کمک گرفته می‌شود. لذا نه تنها به کارگیری آن آسان‌تر از الگوریتم فلویید - وارشال است بلکه فهم و آموزش آن نیز راحت‌تر است. با توجه به اینکه در مرحله i نمی‌توان با سطر و ستون این مرحله مستطیل ایجاد نمود (این گره به‌صورت زیر واسطه قرار می‌گیرد)، لذا سطر و ستون مربوطه همانند الگوریتم فلویید - وارشال تکرار خواهد شد.

رویه‌های بهبود عملکرد ۱ و ۲ که در بندهای ۳. الف و ۳. ب تشریح شدند به‌طور قابل توجهی حجم محاسبات را کاهش می‌دهند. لذا الگوریتم مستطیلی بسیار سریع‌تر از الگوریتم فلویید - وارشال عمل می‌نماید. این در حالی است که الگوریتم مستطیلی به‌خودی خود دارای منطقی ساده‌تر بوده که این امر باعث به کارگیری آسان‌تر الگوریتم و بهبود سرعت آن می‌شود. باید توجه داشت که با توجه به این مهم که این الگوریتم همانند الگوریتم فلویید - وارشال یک الگوریتم بهینه برای حل مسئله کوتاه‌ترین مسیر در شبکه‌های دارای حلقه می‌باشد و منجر به جواب بهینه می‌شود، لذا ضرورتی به مقایسه جواب‌های این الگوریتم با دیگر الگوریتم‌های موجود همانند الگوریتم فلویید - وارشال نمی‌باشد. بنابراین تنها به بررسی زمان محاسباتی این الگوریتم در مقابل الگوریتم فلویید - وارشال اکتفا می‌شود. این بررسی به تفصیل در بخش ۴ آورده شده‌است. در ادامه با آوردن مثالی کوچک به کارگیری الگوریتم مستطیلی تشریح می‌شود.

حل مثال نمونه

شبکه کوتاه‌ترین مسیر دارای دارای حلقه که در شکل ۳ نشان داده شده‌است را در نظر بگیرید. چون $n = 4$ است، لذا دو ماتریس D_j و R_j به ابعاد 4×4 را به تعداد ۵ مرحله تکرار خواهیم نمود. به منظور نشان دادن بهتر توانایی الگوریتم مستطیلی و تعداد کم محاسبات مورد نیاز این الگوریتم در مقایسه با الگوریتم فلوید-وارشال، در ابتدا مسئله را با الگوریتم فلوید-وارشال حل می‌نماییم.



شکل ۳. شبکه مربوط به مثال

الگوریتم فلوید - وارشال

این الگوریتم بدین ترتیب محاسبه می‌شود: به دنبال گام ۲ الگوریتم فلوید-وارشال ابتدا باید ماتریس‌های D_0 و R_0 را محاسبه نماییم:

$$D_0 = \begin{bmatrix} 0 & 3 & 5 & \infty \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ \infty & 6 & 3 & 0 \end{bmatrix}, R_0 = \begin{bmatrix} - & 2 & 3 & - \\ 1 & - & 3 & 4 \\ 1 & 2 & - & 4 \\ - & 2 & 3 & - \end{bmatrix}$$

برای محاسبه ماتریس‌های D_1 و R_1 ، با توجه به این که $j = 1$ است مقادیر سطر و ستون ۱ تکرار خواهد شد. باید توجه کرد که قطر اصلی در کلیه مراحل ثابت خواهد ماند. با پی‌موندن گام ۳ الگوریتم فلوید-وارشال خواهیم داشت:

$$\begin{aligned} d_{23} &= \min(d_{23}, d_{21} + d_{13}) = \min(7, 4 + 5) = 7 \\ d_{24} &= \min(d_{24}, d_{21} + d_{14}) = \min(4, 4 + \infty) = 4 \\ d_{32} &= \min(d_{32}, d_{31} + d_{12}) = \min(3, 6 + 3) = 3 \\ d_{34} &= \min(d_{34}, d_{31} + d_{14}) = \min(5, 6 + \infty) = 5 \\ d_{42} &= \min(d_{42}, d_{41} + d_{14}) = \min(6, \infty + \infty) = 6 \\ d_{43} &= \min(d_{43}, d_{41} + d_{13}) = \min(3, \infty + 5) = 3 \end{aligned}$$

بنابراین ماتریس D_1 مطابق زیر به دست خواهد آمد:

$$D_1 = \begin{bmatrix} 0 & 3 & 5 & \infty \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ \infty & 6 & 3 & 0 \end{bmatrix}$$

با توجه به گام ۳ الگوریتم فلویید - وارشال با در دست داشتن D_1 می‌توان R_1 را محاسبه نمود:

$$\begin{aligned} r_{12} &= k = 2 \\ r_{13} &= k = 3 \\ r_{21} &= k = 1 \\ r_{23} &= k = 3 \\ r_{24} &= k = 4 \\ r_{31} &= k = 1 \\ r_{32} &= k = 2 \\ r_{34} &= k = 4 \\ r_{42} &= k = 2 \\ r_{43} &= k = 3 \end{aligned}$$

بنابراین ماتریس R_1 مطابق زیر به دست خواهد آمد:

$$R_1 = \begin{bmatrix} -2 & 3 & - \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ -2 & 3 & - \end{bmatrix}$$

کلیه محاسبات فوق به تعداد ۸ مرحله دیگر (۴ مرحله برای هر ماتریس) باید تکرار شوند. در اینجا از نشان دادن محاسبات ماتریس‌های D_2 و D_3 ، R_2 و R_3 و D_4 و R_4 صرف نظر می‌کنیم و تنها مقادیر این ماتریس‌ها ارائه می‌شوند:

$$D_2 = \begin{bmatrix} 0 & 3 & 5 & 7 \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ 10 & 6 & 3 & 0 \end{bmatrix}, R_2 = \begin{bmatrix} -2 & 3 & 2 \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ 2 & 2 & 3 & - \end{bmatrix}$$

$$D_3 = \begin{bmatrix} 0 & 3 & 5 & 7 \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ 9 & 6 & 3 & 0 \end{bmatrix}, R_3 = \begin{bmatrix} -2 & 3 & 2 \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ 3 & 2 & 3 & - \end{bmatrix}$$

$$D_4 = \begin{bmatrix} 0 & 3 & 5 & 7 \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ 9 & 6 & 3 & 0 \end{bmatrix}, R_4 = \begin{bmatrix} -2 & 3 & 2 \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ 3 & 2 & 3 & - \end{bmatrix}$$

حال با در دست داشتن دو ماتریس D_4 و R_4 می‌توان فاصله و مسیر را از هر گره دلخواه i به هر گره دلخواه k به دست آورد. به‌عنوان مثال کوتاه‌ترین فاصله بین دو

گره ۱ و ۴ عبارتست از ۹ ($d_{41} = 9$) و مسیر متناظر (کوتاه‌ترین مسیر) عبارت است از عبور از طریق گره ۳ ($r_{31} = 1$ و $r_{41} = 1$).

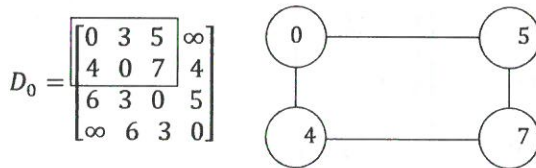
الگوریتم مستطیلی

با توجه به الگوریتم مستطیلی (الگوریتم ۲)، ابتدا باید ماتریس‌های D_0 و R_0 محاسبه شوند:

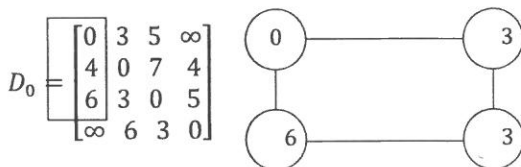
$$D_0 = \begin{bmatrix} 0 & 3 & 5 & \infty \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ \infty & 6 & 3 & 0 \end{bmatrix}, R_0 = \begin{bmatrix} -2 & 3 & - \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ -2 & 3 & - \end{bmatrix}$$

با توجه به قدم سوم الگوریتم مستطیلی، محاسبه دیگر ماتریس‌ها (در اینجا D_1 و R_1, D_2, R_2, D_3, R_3 و D_4, R_4) براساس رویه‌های بهبود عملکرد ۱ و ۲، و ترسیم مجموعه‌ای از مستطیل‌ها انجام می‌گیرد. در این بخش این الگوریتم برای محاسبه دو ماتریس D_1 و R_1 به‌طور کامل شرح داده می‌شود و برای دیگر ماتریس‌های تنها به ذکر مقادیر نهایی اکتفا می‌شود.

استفاده از رویه بهبود عملکرد ۱ باعث می‌شود تنها محاسبه درایه‌های d_{23} و d_{32} برای دستیابی به ماتریس D_1 کافی باشد (در حالی که در الگوریتم فلوید - وارشال باید تمامی درآیه‌ها جز قطر اصلی محاسبه شوند). چون $z = 1$ است صفر روی سطر و ستون ۱ معیار ترسیم مستطیل می‌باشد. برای محاسبه درایه d_{23} مستطیلی رسم می‌شود (با استفاده از ماتریس D_0) که یک راس آن درایه d_{23} و راس مقابل آن ۰ سطر و ستون ۱ (درایه d_{11}) باشد (شکل ۴).



شکل ۴: مستطیل ترسیم شده برای محاسبه d_{23}



شکل ۵. مستطیل ترسیم شده برای محاسبه d_{32}

با توجه به شکل ۴ $d_{23} = \min(7, 4 + 5) = 7$ و به طور مشابه $d_{32} = \min(3, 6 +$
 $= (3) (شکل ۵)).$ در نتیجه ماتریس زیر حاصل خواهد شد.

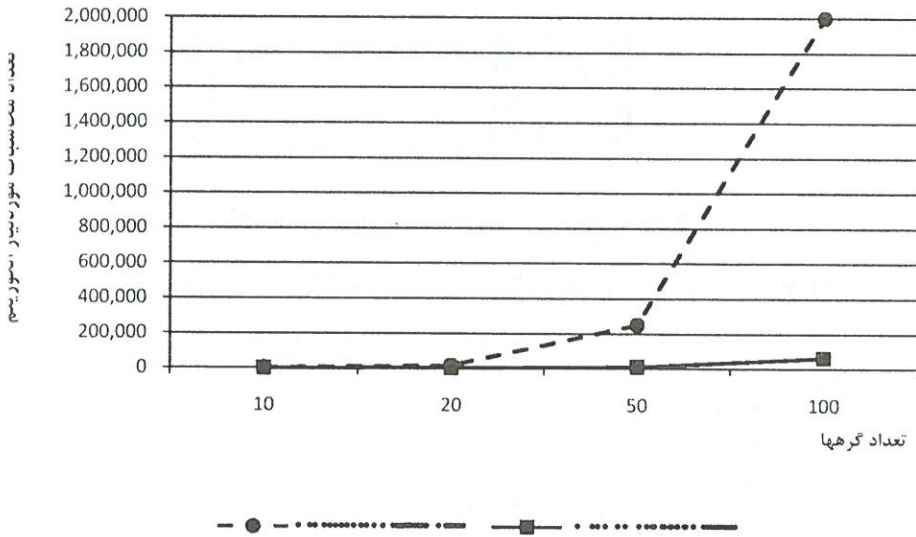
$$D_1 = \begin{bmatrix} 0 & 3 & 5 & \infty \\ 4 & 0 & 7 & 4 \\ 6 & 3 & 0 & 5 \\ \infty & 6 & 3 & 0 \end{bmatrix}$$

بهره‌گیری از رویه بهبود عملکرد ۲ در محاسبه R_1 خواهیم داشت:

$$R_1 = \begin{bmatrix} -2 & 3 & - \\ 1 & -3 & 4 \\ 1 & 2 & -4 \\ -2 & 3 & - \end{bmatrix}$$

حال با دنبال نمودن مراحل فوق می‌توان به دیگر ماتریس‌های D_2 و R_2 ، D_3 و R_3 و L و R_4 نیز دست یافت. بدیهی است مقادیر این ماتریس‌ها همانند مقادیر متناظر اصل از الگوریتم فلویید - وارشال است. توجه داشته‌باشید که استفاده از رویه‌های بود عملکرد ۱ و ۲ معرفی شده در الگوریتم مستطیلی بسیار موثر در کاهش حجم حسابات و بنابراین تسریع سرعت الگوریتم بوده‌اند. در الگوریتم مستطیلی نیز با در ست داشتن ماتریس‌های D_4 و R_4 می‌توان فاصله و مسیر هر گره i تا هر گره k را دست آورد.

همان‌طور که پیش‌تر نیز به آن اشاره شد با توجه به بهینگی الگوریتم مستطیلی، ل مسائل نمونه توسط این الگوریتم و مقایسه آنها با نتایج حاصل از الگوریتم وید - وارشال ضروری به نظر نمی‌رسد. لیکن به منظور نشان دادن حجم محاسبات م الگوریتم مستطیلی در مقابل الگوریتم فلویید - وارشال نتایج بررسی‌های بیه‌سازی شده توسط نویسندگان در شکل ۶ نشان داده شده‌است.



شکل ۶. عملکرد شبیه‌سازی شده الگوریتم‌های فلوید - وارشال و مستطیلی برای مسائل تا ابعاد ۱۰۰ گره

تعداد محاسبات الگوریتم فلوید - وارشال برای به‌دست آوردن هر یک از ماتریس‌های D_j و R_j ، $n^2 - n$ است. از طرفی تعداد ماتریس‌های تولید شده در این الگوریتم $2(n + 1)$ است. در نقطه مقابل تعداد محاسبات الگوریتم مستطیلی در بدترین عملکرد خود به اندازه تعداد محاسبات الگوریتم فلوید- وارشال خواهد بود (توجه داشته باشید که رویه‌های بهبود عملکرد ۱ و ۲ متناسب با شرایط مسئله می‌توانند منجر به کاهش بسیار چشمگیر و قابل توجهی در تعداد محاسبات لازم شوند). همان‌طور که نتایج شبیه‌سازی نشان می‌دهد زمان الگوریتم فلوید- وارشال با افزایش اندازه مسئله به‌طور نمایی افزایش می‌یابد. در نقطه مقابل زمان الگوریتم مستطیلی با اندازه مسئله و آن‌هم نه به‌طور نمایی تغییر می‌کند.

نتیجه‌گیری

در این مقاله الگوریتم بهینه جدیدی برای محاسبه کوتاه‌ترین هزینه و مسیر در شبکه‌های دارای حلقه ارائه شده‌است. الگوریتم ارائه شده با بهبود یکی از بهترین

گوریتم‌های موجود (الگوریتم فلوید - وارشال) منجر به کاهش قابل توجه حجم حسابات و زمان حل مسئله شده‌است. پرواضح است که در مسائل بزرگ این روش بسیار کارتر از الگوریتم فلوید - وارشال خواهد بود. علاوه بر این، فهم این الگوریتم ز برای اهداف آموزشی بسیار آسان‌تر از الگوریتم فلوید - وارشال است. به‌منظور حقیقات آتی، نویسندگان مقاله در حال مطالعه و بررسی بهبود الگوریتم مستطیلی از ریق کاهش تعداد مراحل لازم برای حصول ماتریس‌های R_n و D_n می‌باشند (در حال حاضر تعداد $n + 1$ مرحله جهت حصول این دو ماتریس لازم است).

منابع و مأخذ

۱. سبزه‌پرور، مجید. کنترل پروژه (گام به گام). انتشارات خانیران، تهران (۱۳۸۱).
۲. قاسمی، فرهاد. کنترل پروژه. جزوه انتشارات دانشگاه صنعتی شریف، دانشگاه صنعتی شریف، (۱۳۶۷).
۳. مهرگان، محمدرضا. پژوهش عملیاتی. انتشارات سالکان، تهران (۱۳۷۰).
4. Bellman, R. E. **On a routing Problem**. Quarterly of Applied Mathematics. 16(1): 87-90 (1958).
5. Cherkassky, B. V., Goldberg, A. V. and Radzik, T. **Shortest paths algorithms: Theory and experimental evaluation**. Mathematical Programming. 73(2): 129-174 (1996).
5. Dijkstra, E. W. **A Note on Two Problems in Connection with Graphs**. Numeriskche Mathematik. 1: 269-271 (1959).
7. Floyd, R. W. **Algorithm 97: Shortest path**. Communications of the ACM. 5(6): 345 (1962).
3. Ford, L. R. and Fulkerson, D. R. **Flows in Networks**. Princeton University Press, Princeton, NJ (1962).
9. Gallo, G. and Pallotino, S. **Shortest Paths Algorithms**. Annals of Operations Research. 13: 3-79 (1988).
0. Warshall, S. **A Theorem on Boolean Matrices**. Journal of the ACM. 9(1): 11-12 (1962).
1. Fua, L., Sunb, D., and Rilett, L.R. **Heuristic shortest path algorithms for transportation applications State of the art**. Computers & Operations Research 33: 3324-3343 (2006).
2. Altin, A., Fortz, B., Thorup, M., and Umit, H. **Intra-domain traffic engineering with shortest path routing protocols**. 4OR .7 (4): 301-335 (2009)
3. Klunder, G. A., and Post, H. N. **The shortest path problem on large-scale real-road networks**, Networks, 48 (4): 182-194 (2006).
4. Gamache, M., Grimard, R., and Cohen, P. **A shortest-path algorithm for solving the fleet management problem in underground mines**, European Journal of Operational Research 166: 497-506 (2005).
5. Roysset, J. O. , Carlyle, W. M., and Wood, R. K. **Routing Military Aircraft With A Constrained Shortest-Path Algorithm**, Military Operations Research, 14 (3): 31-52 (2009)