

الگوریتم مستطیل آبشاری و ماتریس انتقال در شبکه‌های کوتاه‌ترین مسیر با دور

اصغر عینی^{۱*}، کورش عشقی^۲

۱- دانشجوی دکتری، دانشگاه صنعتی شریف، دانشکده مهندسی صنایع، تهران، ایران

۲- استاد، دانشگاه صنعتی شریف، دانشکده مهندسی صنایع، تهران، ایران

رسید مقاله: ۱۸ خرداد ۱۳۹۶

پذیرش مقاله: ۱۹ آبان ۱۳۹۶

چکیده

مساله کوتاه‌ترین مسیر یکی از مسایل مشهور، بنیادی و پرطرفدار در نظریه گراف و شبکه است. در ادبیات این مساله، الگوریتم‌های کارای زیادی برای تعیین کوتاه‌ترین مسیر و مسافت بین هرزوج گره بر پایه جبر ماتریسی وجود دارد. در این مقاله، یک الگوریتم دقیق جدید با نام الگوریتم مستطیل آبشاری به کمک ساختار اصلی الگوریتم‌های قبلی و با بهبود روش‌های جدید، ارائه شده است. در الگوریتم ارائه شده سعی می‌شود تمام محاسبات و عملیات ریاضی الگوریتم در قالب تعدادی مستطیل پیاده‌سازی شود. الگوریتم مستطیل آبشاری، یک الگوریتم کاراست به‌طوری که روش اجرای ساده و زمان اجرای سریع دارد. بعلاوه، یک ماتریس جدید براساس ماتریس مسیر، با نام ماتریس انتقال تعریف شده که در تحلیل حساسیت و بهینه‌سازی مجدد شبکه‌های کوتاه‌ترین مسیر کاربرد دارد. در پایان، برای نشان دادن جزییات اجرای الگوریتم‌های مستطیل آبشاری، فلوید-وارشال، ماتریس تجدید نظرشده هو و ماتریس انتقال، یک مثال به صورت گام به گام حل شده است.

کلمات کلیدی: الگوریتم‌های شبکه کوتاه‌ترین مسیر با دور، الگوریتم مستطیل آبشاری، روش ماتریس تجدید نظر شده، الگوریتم فلوید - وارشال، ماتریس انتقال.

۱ مقدمه و مرور ادبیات

۱-۱ مقدمه

شبکه‌های کوتاه‌ترین مسیر از بنیادی‌ترین شبکه‌ها در جریان‌های شبکه بوده و کاربردهای فراوانی در حوزه‌های مختلف دارد. برای حل اینگونه شبکه‌ها، روش‌ها و الگوریتم‌های زیادی با رویکردهای متفاوت وجود دارد از

* عهده‌دار مکاتبات

آدرس الکترونیکی: ainiasghar@ie.sharif.edu

جمله می‌توان به الگوریتم‌های جذاب فلویید - وارشال [۱ و ۲]، آبشاری [۳]، ماتریس تجدید نظر شده [۴]، مستطیلی [۵]، دایجسترا [۶] و غیره اشاره کرد.

مساله کوتاه‌ترین مسیر از طرفی ساده‌ترین و از طرفی دیگر یکی از بنیادی‌ترین مسایل جریان‌های شبکه است [۷]. این مساله از رده‌ی مسایل P بوده، اما برای بعضی از نسخ آن از قبیل مساله کوتاه‌ترین مسیر با محدودیت و مساله کوتاه‌ترین مسیر با سیکل منفی NP-hard بودن آن‌ها ثابت شده است [۸-۱۰]. این مساله مورد علاقه پژوهشگران و مهندسان حوزه‌های مختلف است و در موارد کاربردی متعددی نظیر شبکه‌های حمل و نقل، شبکه‌های ارتباطی و مخابراتی، شبکه‌های لجستیک و پشتیبانی استفاده می‌شود. روش‌ها و الگوریتم‌های آسان و کارا برای حل مساله کوتاه‌ترین مسیر وجود دارد که جزو ساده‌ترین مدل‌های شبکه محسوب شده که در تبدیل شبکه‌های بزرگ و پیچیده به شبکه‌های ساده‌تر و در حل بسیاری از مسایل بهینه‌سازی ترکیبی نیز مورد استفاده قرار می‌گیرد. مطالعه مساله کوتاه‌ترین مسیر نقطه عطف شروع برای تولید مفاهیم مهم و اساسی جریان‌های شبکه شامل ساختار داده هوشمند و توسعه کارائی الگوریتم‌هاست [۷].

۲-۱ مرور ادبیات

پژوهشگران مسایل کوتاه‌ترین مسیر را به صورت زیر تقسیم‌بندی نموده‌اند:

- ۱- مساله کوتاه‌ترین مسیر و مسافت از یک گره به هر گره^۱
- ۲- مساله کوتاه‌ترین مسیر و مسافت از چند زوج گره^۲
- ۳- مساله کوتاه‌ترین مسیر و مسافت از هر زوج گره^۳
- ۴- مسایل کوتاه‌ترین مسیر تعمیم یافته^۴

طی چند دهه اخیر الگوریتم‌های زیادی برای مساله کوتاه‌ترین مسیر توسعه داده شده است. عموماً این الگوریتم‌ها به سه دسته زیر تقسیم می‌شوند [۱۱]:

- ۱- الگوریتم‌هایی براساس پیمایش شبکه^۵
- ۲- الگوریتم‌هایی براساس برنامه‌ریزی خطی^۶
- ۳- الگوریتم‌هایی برپایه جبر ماتریسی^۷

ابتدا در سال ۱۹۶۲ الگوریتم جذاب فلویید - وارشال ارایه شد. وارشال [۱] مبانی نظری الگوریتم را ارایه کرد و فلویید [۲] آن را پیاده‌سازی نمود. سپس الگوریتم معروف آبشاری معرفی شد [۳]. آنگاه هو [۴] روش ماتریس تجدیدنظر شده را با بهبود در الگوریتم آبشاری ارایه کرد. چند سال بعد، با ارایه ساختار جبری جدید به مدل‌سازی و حل مسایل شبکه‌های جریان از قبیل شبکه‌های کوتاه‌ترین مسیر پرداخته شد [۱۲].

¹ Single Source Shortest Path Problem – SSSP Problem

² Multiple Pairs Shortest Path Problem - MPSP Problem

³ All Pairs Shortest Path Problem – APSP Problem

⁴ Generalized Shortest Path Problem

⁵ Network Traversal Algorithms

⁶ LP – Based Algorithms

⁷ Matrix Algebra Algorithms

همچنین با استفاده از تکنیک‌های ماتریس متراکم، دو روش تولید کد و تولید آدرس - جدول برای اجرای بهتر برنامه‌های کامپیوتری جهت حل مساله کوتاه‌ترین مسیر هر زوج گره ارایه شد [۱۳]. تاکائوکا [۱۴] به کمک ضرب ماتریس مسافت تحت مدل RAM الگوریتم سریع تری برای این مساله معرفی نمود. سپس دوئین [۱۵] دو الگوریتم جدید بر اساس بهبود معادلات بلمن در شبکه‌های بزرگ و متراکم ارایه کرد. در همین سال، دمترسو و ایتالیانو [۱۶] الگوریتم جدیدی برای شبکه‌های کوتاه‌ترین مسیر پویا براساس خواص جبری و ساختار داده ارایه کردند. چان [۱۷] یک الگوریتم جدید و ساده با بهبود الگوریتم‌های زیرمکعبی از قبیل فریدمن ارایه نمود. موحمد و همکاران [۱۸]، به کمک الگوریتم فراابتکاری بهینه‌سازی توده ذرات^۱ مساله کوتاه‌ترین مسیر را حل کردند. دو سال بعد، هوگاردی [۸] با بهبود الگوریتم جذاب فلویید - وارشال توانست سیکل منفی را در شبکه‌های کوتاه‌ترین مسیر بادور و باوزن منفی شناسائی کند. در همین سال، موحامد و همکاران [۱۹]، به کمک الگوریتم فراابتکاری ژنتیک^۲ مساله کوتاه‌ترین مسیر دوهدفه را حل کردند. در اینگونه مسایل علاوه بر وزن مسافت، وزن زمان هم در نظر گرفته می‌شود. یاستر [۲۰] نیز یک الگوریتم تقریبی برای مساله کوتاه‌ترین مسیر با وزن‌های حقیقی ارایه کرد. پنگ و همکاران [۲۱] با بهبود الگوریتم دایجسترا به کمک دو رویکرد بهینه‌سازی آن را در حل شبکه‌های بزرگ کوتاه‌ترین مسیر نظیر شبکه‌های اینترنت، الکترونیک، حمل و نقل و اجتماعی استفاده کردند. همچنین دوئورو همکاران [۲۲] دو الگوریتم مختلف برای این مساله ارایه کردند. اولین الگوریتم با رویکرد مکانیزم‌های تعمیر و دومین الگوریتم بر اساس انتخاب والد است. در همین سال، هاگت و همکاران [۲۳]، یک الگوریتم کارا برای شبکه‌های کوتاه‌ترین مسیر چند ماژولی^۳ و وانگ و کمبکه [۲۴]، الگوریتمی جدید با زمان چندجمله‌ای برای محاسبه کوتاه‌ترین مسیر در سیستم‌های توزیع شده ارایه کردند. آمروسینی و سوماجن [۲۵]، یک الگوریتم جدید برای شبکه‌های کوتاه‌ترین مسیر چند ماژول شهری معرفی کردند.

۳-۱ مزایای الگوریتم مستطیل آبشاری

اساس الگوریتم مستطیل آبشاری برگرفته از الگوریتم‌های ماتریس تجدیدنظر شده هو و آبشاری فاربی و همکاران است. صحت اجرای الگوریتم‌های هو و آبشاری فاربی و همکاران برای محاسبه ماتریس‌های مسافت و مسیر قبلا اثبات شده است [۳ و ۴]. در نگارش الگوریتم مستطیل آبشاری، از ساختار گام‌ها، به‌ویژه نمایش ماتریس‌های مسافت و مسیر الگوریتم‌های جذاب فلویید - وارشال و مستطیلی استفاده شده است [۱، ۲ و ۵]. به عبارت بهتر، الگوریتم مستطیل آبشاری ترکیبی از الگوریتم‌های هو، فاربی و همکاران، الگوریتم جذاب فلویید - وارشال و الگوریتم مستطیلی است که با تغییرات زیر پیاده‌سازی شده است.

- الگوریتمی کردن روش انجام کار در سه گام و زیر گام در قالب مراحل F و B و با ساختار ماتریسی (بهبود روش ماتریس تجدیدنظر شده هو از نظر ساختار الگوریتم و نمایش ماتریسی بجای جدولی داده‌های مسیر و مسافت).

¹ Particle Swarm Optimization (PSO)

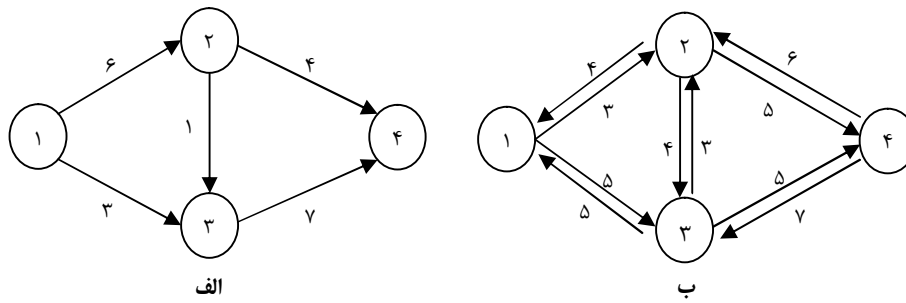
² Genetic Algorithm

³ Multi Modal

- وابسته کردن کلیه محاسبات عناصر ماتریس مسیر R به عناصر ماتریس مسافت D (بهبود کامل در الگوریتم‌های هو، فاربی و فلوید - وارشال با حذف محاسبات ماتریس مسیر R) این موضوع در بند دوم گام دوم و بند د گام سوم الگوریتم مستطیل آبشاری آمده است. این بهبود با کاهش تعداد محاسبات و در نتیجه افزایش سرعت اجرای الگوریتم همراه خواهد بود.
- انجام کلیه محاسبات و مقایسات ریاضی مورد نیاز الگوریتم مستطیل آبشاری در قالب $(n-2)$ مستطیل (این مستطیل‌ها که معرف محاسبات الگوریتم هستند به صورت ذهنی قابل رسم بوده و به سادگی محاسبه می‌شوند).
- کاهش حجم محاسبات از n محاسبه و مقایسه در الگوریتم‌های هو و فاربی به $(n-2)$ محاسبه و مقایسه در الگوریتم مستطیل آبشاری برای محاسبه هر عنصر ماتریس مسافت D . به عبارت بهتر، در الگوریتم هو و فاربی برای محاسبه هر عنصر ماتریس مسافت D ، به تعداد $2n(n^2-n)$ محاسبه و مقایسه نیاز است به نحوی که در مستطیل آبشاری، این تعداد به $2(n-2)(n^2-n)$ کاهش می‌یابد. در الگوریتم‌های هو، فاربی و فلوید - وارشال برای محاسبه ماتریس مسیر به تعداد محاسبات ماتریس مسافت مورد نیاز است بطوری که، در الگوریتم مستطیل آبشاری این محاسبات حذف شده و محاسبات عناصر ماتریس مسیر کاملاً به نتیجه محاسبات عناصر ماتریس مسافت وابسته شده است.
- مقایسه ذهنی عنصر d_{ik} با عناصر d_{ij} و d_{jk} . به عبارت بهتر، اگر $d_{ik} \leq d_{ij}$ یا $d_{ik} \leq d_{jk}$ باشد، آنگاه $d_{ik} \leq d_{ij} + d_{jk}$ خواهد بود. در نتیجه، این محاسبه و مقایسه قابل حذف خواهد بود. این مقایسه ذهنی باعث کاهش حجم محاسبات و در نتیجه افزایش سرعت الگوریتم مستطیل آبشاری خواهد شد.
- روش تصحیح مسیر در الگوریتم‌های جبر ماتریسی به ویژه الگوریتم فلوید - وارشال. الگوریتم‌های جبر ماتریسی از قبیل الگوریتم جذاب فلوید - وارشال در محاسبه صحیح عناصر ماتریس مسیر در برخی از شبکه‌ها با ساختارهای خاص دچار محدودیت محاسباتی می‌شود به نحوی که این موضوع، با ارایه روشی در مثال اول ۵-۲ برطرف شده است.
- رویکرد انتخاب صحیح مسیر در الگوریتم مستطیل آبشاری. با روش ارایه شده در آخر پاراگراف بند د از گام سوم، عناصر ماتریس مسافت در الگوریتم مستطیل آبشاری بدرستی محاسبه می‌شوند.
- ارایه ماتریس انتقال بر اساس ماتریس مسیر. ماتریس انتقال از مباحث جدید بوده که توسط نویسندگان ارایه شده به نحوی که در تحلیل حساسیت و بهینه سازی مجدد شبکه‌های کوتاه‌ترین مسیر کاربرد اساسی دارد. این ماتریس در هر رویکرد جبر ماتریسی که در آن ماتریس مسیر تولید می‌شود، قابل پیاده سازی است.

۲ شبکه‌های کوتاه‌ترین مسیر با دور و بدون دور

شبکه‌های کوتاه‌ترین مسیر از نظر ساختار شبکه‌ای به دو دسته‌ی شبکه‌های کوتاه‌ترین مسیر جهت‌دار بدون دور و شبکه‌های کوتاه‌ترین مسیر جهت‌دار با دور تقسیم می‌شوند. در شکل ۱، دو شبکه ساده نمونه از شبکه‌های کوتاه‌ترین مسیر آورده شده که در آن شکل ۱- الف ساختار یک شبکه کوتاه‌ترین مسیر جهت‌دار بدون دور و شکل ۱- ب ساختار یک شبکه کوتاه‌ترین مسیر جهت‌دار با دور را نشان می‌دهد.



شکل ۱. دو شبکه کوتاه‌ترین مسیر جهت‌دار ساده، الف. بدون دور و ب. با دور

شبکه‌های کوتاه‌ترین مسیر جهت‌دار با دور، نوع جامع و کامل‌تری از شبکه‌های کوتاه‌ترین مسیر هستند. در این شبکه‌ها حداقل یک دور وجود دارد. این نوع شبکه‌ها فاقد گره شروع و پایان مشخصی هستند، لذا هر گره می‌تواند معرف گره شروع یا ختم باشد [۵]. الگوریتم مستطیل آبشاری ارائه شده در این مقاله برای حل شبکه‌های کوتاه‌ترین مسیر با دور برای هرزوج گره است.

۳ الگوریتم مستطیل آبشاری

شبکه جهت‌دار با دور $G = (N, A)$ با مجموعه گره‌های $N = \{1, 2, \dots, n\}$ و مجموعه کمان‌های $A = \{(i, k) | i, k \in N, i \neq k\}$ و با وزن (مسافت) مثبت d_{ik} برای هر کمان $(i, k) \in A$ را در نظر بگیرید. هدف پیدا کردن کوتاه‌ترین مسافت و مسیر بین هر دو گره دلخواه i و k است. برای این منظور از دو ماتریس مسافت D_s و مسیر R_s با ابعاد $n \times n$ استفاده می‌شود که طی دو مرحله $s = B, F$ به جواب بهینه می‌رسند. مرحله B (مرحله پایه)^۱ و مرحله F (مرحله نهایی)^۲ نامیده می‌شود. به عبارت بهتر، در شروع اجرای الگوریتم، دو ماتریس D_B و R_B به ترتیب مقادیر اولیه مسافت و مسیر را که برگرفته از داده‌های شبکه است در خود ذخیره کرده و در پایان اجرای الگوریتم، دو ماتریس D_F و R_F به ترتیب مقادیر کوتاه‌ترین مسافت و مسیر را در خود ذخیره می‌کنند. در این الگوریتم محاسبه ماتریس‌های R_s به ماتریس D_s وابسته هستند که این موضوع، باعث افزایش سرعت و کاهش زمان اجرای الگوریتم مستطیل آبشاری می‌شود. دلیل نام‌گذاری الگوریتم به مستطیل آبشاری این است که تمام محاسبات و عملیات ریاضی الگوریتم در قالب $n-2$ مستطیل پیاده‌سازی می‌گردد که دارای راس مشترک d_{ik} هستند. به عبارت بهتر، در الگوریتم مستطیل آبشاری برای محاسبه هر عنصر d_{ik} ، از $n-2$ مقایسه و در قالب مستطیل که معرف روابط ریاضی هستند، استفاده شده در صورتی که در الگوریتم‌های فابری یا هو از n مقایسه در قالب روابط ریاضی برای محاسبه هر عنصر d_{ik} استفاده می‌شود در نتیجه، در الگوریتم پیشنهادی علاوه بر رویکرد مستطیلی که انجام محاسبات را ساده‌تر نموده تعداد مقایسات برای محاسبه هر عنصر نیز کاهش یافته است که این مزیت، باعث افزایش سرعت و کاهش زمان اجرای الگوریتم شده است. چون اساس الگوریتم مستطیل آبشاری برگرفته از الگوریتم‌های اثبات شده آبشاری فابری به‌ویژه ماتریس تجدید نظر شده هو است نیازی به

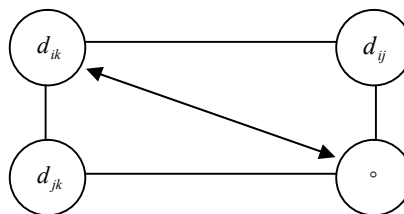
¹ Basic Stage

² Final Stage

اثبات صحت الگوریتم نخواهد بود. جهت مطالعه بیشتر به مراجع [۳ و ۴] رجوع گردد.

۳-۱ روش تشکیل مستطیل

در مرحله F ، برای محاسبه هر عنصر d_{ik} (عنصر سطر i و ستون k)، $n-2$ مستطیل با استفاده از صفرهای قطر اصلی طوری رسم می‌شوند که عنصر d_{ik} روبروی صفرهای قطر اصلی قرار گیرد. به عبارت بهتر، یک مستطیل از چهار راس تشکیل می‌شود به نحوی که یک راس آن را عنصر d_{ik} ، راس مقابل را صفر قطر اصلی و دو راس دیگر را عناصر d_{ij} و d_{jk} تشکیل می‌دهند. براساس مکان هر عنصر d_{ik} جایگاه صفرهای قطر اصلی می‌تواند متغیر باشد، اما همواره صفرها باید روبروی عنصر d_{ik} باشد. یکی از انواع این مستطیل‌ها در شکل ۲ نشان داده شده است.



شکل ۲. نحوه ایجاد مستطیل در الگوریتم مستطیل آبشاری. در این شکل، صفر همان صفرهای قطر اصلی ماتریس‌های D_F و D_B هستند.

به عبارت بهتر، به کمک این مستطیل‌ها، مقدار هر عنصر d_{ik} با $i \neq k$ ماتریس D_F به ترتیب رابطه (۱) محاسبه خواهد شد.

$$d_{ik} = \min(d_{ik}, d_{ij} + d_{jk}), \forall j, j \neq i, j \neq k \quad (1)$$

در رابطه (۱)، با تغییر مقدار هر عنصر d_{ik} ، مقدار جدید جایگزین مقدار قبلی در ماتریس‌های مسافت D_B و D_F شده، سپس محاسبات رفت یا برگشت ادامه می‌یابد.

یکی از روش‌های کاهش حجم محاسبات و در نتیجه افزایش سرعت الگوریتم مستطیل آبشاری مقایسه ذهنی عنصر d_{ik} با عناصر d_{ij} و d_{jk} است. به عبارت بهتر، اگر $d_{ik} \leq d_{ij}$ یا $d_{ik} \leq d_{jk}$ باشد، آنگاه $d_{ik} \leq d_{ij} + d_{jk}$ خواهد بود. در نتیجه، این محاسبه یا مقایسه قابل حذف خواهد بود. عناصر d_{ik} ، d_{ij} و d_{jk} رئوس مستطیل در شکل ۲ هستند.

۳-۲ گام‌های الگوریتم

گام اول: اگر n برابر تعداد گره‌های شبکه باشد، دو ماتریس مربع $n \times n$ ، D_s و R_s به ازای $s = B, F$ که در آن s را مرحله نامند، تعریف کنید.

گام دوم: در مرحله $s = B$ عناصر ماتریس‌های مسافت و مسیر D_B و R_B را به ترتیب زیر محاسبه کنید:

محاسبه عناصر ماتریس D_B

$$d_{ik} = \begin{cases} d_{ik} & \text{اگر کمان مستقیم از گره } i \text{ به گره } k \text{ وجود داشته باشد} \\ \infty & \text{اگر کمان مستقیم از گره } i \text{ به گره } k \text{ وجود نداشته باشد} \\ 0 & \text{اگر } i = k \end{cases}$$

محاسبه عناصر ماتریس R_B

$$r_{ik} = \begin{cases} k & \text{ماتریس در آن متناظر عنصر اگر } D_B, 0 \text{ یا } \infty \text{ نباشد} \\ - & \text{در غیر این صورت} \end{cases}$$

گام سوم: در مرحله $s = F$ ماتریس‌های D_F و R_F را براساس الگوی زیر پیاده‌سازی کنید.

الف. عناصر قطر اصلی ماتریس‌های D_F و R_F را از ماتریس‌های D_B و R_B تکرار کنید.

ب. برای محاسبه سایر عناصر ماتریس D_F و به‌طور مشخص برای هر عنصر d_{ik} با $i \neq k$ تعداد $n-2$ مستطیل در ماتریس D_B رسم کنید. این مستطیل‌ها طوری رسم می‌شوند که عنصر d_{ik} روی صفرهای قطر اصلی باشد. چون برای محاسبه عنصر d_{ik} از صفرهای موجود در سطر i و ستون k نمی‌توان استفاده نمود یا به عبارت بهتر، با اینگونه صفرها نمی‌توان مستطیل ساخت، به همین دلیل تعداد مستطیل‌ها برابر $n-2$ خواهد بود. این محاسبات که از عنصر d_{12} شروع شده و به عنصر $d_{n,n-1}$ ادامه می‌یابد را محاسبات رفت بنامید. به محض تغییر عنصر d_{ik} ، آن را در ماتریس مسافت جایگزین کرده، سپس محاسبات را ادامه دهید.

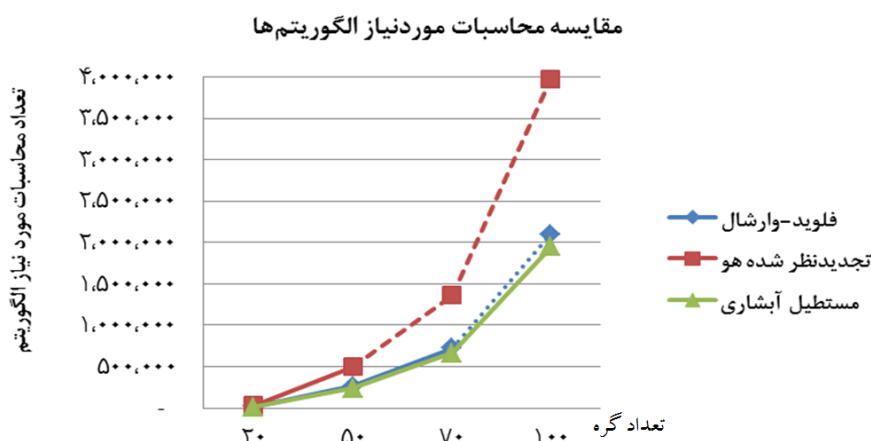
ج. محاسبات برگشت (همانند محاسبات رفت) را از عنصر $d_{n,n-1}$ ماتریس D_F شروع کرده تا به عنصر d_{12} ادامه دهید.

د. بر خلاف الگوریتم‌های هو، فاری و فلوید-وارشال، برای محاسبه عناصر ماتریس R_F ، محاسبات جدیدی انجام نمی‌شود، بلکه محاسبات عناصر ماتریس‌های مسیر R_F و R_B به ترتیب به محاسبات عناصر ماتریس‌های D_F و D_B وابسته شده است. به عبارت بهتر، اگر عنصری از ماتریس D_F تغییر نکند در آن صورت، عنصر متناظر آن در R_F نیز تغییر نخواهد کرد. در غیر این صورت، عنصر متناظر در ماتریس R_F نیز تغییر خواهد یافت و این تغییرات بستگی به مختصات صفرهای قطر اصلی در مستطیل‌های مربوطه خواهد داشت. در صورت برابری مقادیر مستطیل‌ها، مقدار r_{ik} را از سطر i و ستون‌های مرتبط با عناصر قطر اصلی مستطیل‌های برابر انتخاب کنید.

دلیل وابسته کردن محاسبات ماتریس مسیر به محاسبات ماتریس مسافت، تشابه محاسباتی ماتریس‌های مسیر و مسافت در الگوریتم‌های هو، فاری و فلوید-وارشال است. به عبارت بهتر، در این الگوریتم‌ها، از دو بیان برای یک منظور استفاده شده است. در نتیجه، صحت اجرای الگوریتم مستطیل آبخاری همانند الگوریتم‌های اثبات شده هو، فاری و فلوید-وارشال خواهد بود.

قضیه: پیچیدگی زمانی بدترین حالت الگوریتم مستطیل آبخاری، $O(n^3)$ است.

اثبات: برای محاسبه عناصر ماتریس D_B ، تعداد محاسبات برابر $n^2 - n$ و برای محاسبه عناصر ماتریس D_F تعداد محاسبات رفت و برگشت یکسان خواهد بود [۳ و ۴]. به عبارت بهتر، برای اجرای الگوریتم مستطیل آبخاری به تعداد $2(n^2 - n)$ محاسبه با $(n - 2)$ مستطیل برای هر عنصر d_{ik} ماتریس D_F نیاز است به طوری که محاسبات رفت در ماتریس D_B و محاسبات برگشت در ماتریس D_F انجام می‌شود؛ بنابراین کل محاسبات مورد نیاز، برای محاسبه عناصر ماتریس‌های مسافت D_F و D_B برابر $(n^2 - n) + (n - 2)(n^2 - n)$ خواهد بود. به جهت وابسته کردن ماتریس‌های مسیر به ماتریس‌های مسافت، برای تعیین عناصر ماتریس‌های مسیر R_F و R_B محاسبات جدیدی نیاز نخواهد بود. در نتیجه، تعداد کل محاسبات مورد نیاز الگوریتم مستطیل آبخاری برابر $(n^2 - n) + (n - 2)(n^2 - n)$ خواهد شد. به عبارت بهتر، می‌توان نتیجه گرفت که پیچیدگی زمانی الگوریتم مستطیل آبخاری در بدترین حالت، همانند الگوریتم‌های فلوید-وارشال و ماتریس تجدیدنظر شده هو $O(n^3)$ خواهد بود. قابل ذکر است که در اکثر شبکه‌ها که دارای ساختار خاص نیستند محاسبات رفت کفایت خواهد کرد، در این صورت حجم محاسبات الگوریتم مستطیل آبخاری به نصف کاهش خواهد یافت. در شکل ۳، تعداد محاسبات مورد نیاز سه الگوریتم برای شبکه‌ها با گره‌های مختلف نشان داده شده است.



شکل ۳. عملکرد محاسباتی الگوریتم‌های مستطیل آبخاری، فلوید-وارشال و ماتریس تجدیدنظر شده هو برای شبکه‌های مختلف

با توجه به شکل ۳، تعداد محاسبات مورد نیاز الگوریتم مستطیل آبخاری کم‌تر از الگوریتم فلوید-وارشال و بسیار کم‌تر از الگوریتم ماتریس تجدیدنظر شده هو به‌ویژه در مقیاس بزرگ خواهد بود.

۴ ماتریس انتقال

پس از حل شبکه‌های کوتاه‌ترین مسیر هرزوج گره، به کمک الگوریتم‌های جبر ماتریسی از قبیل الگوریتم‌های فلوید-وارشال، آبخاری، ماتریس تجدیدنظر شده هو، مستطیلی [۵]، مستطیل آبخاری و غیره دو ماتریس مسافت D و مسیر R محاسبه شده به طوری که کوتاه‌ترین مسیر و مسافت هرزوج گره در آن‌ها ذخیره می‌شوند. در این قسمت

ماتریس انتقال یا جابجائی^۱ ارایه شده که عناصر آن بر اساس عناصر ماتریس مسیر محاسبه می شود. عناصر ماتریس انتقال نشان دهنده تعداد جابه جایی کمان های شبکه در جواب بهینه است. به کمک این ماتریس، میزان ارزش هر کمان در جابه جایی بهینه شبکه قابل محاسبه است. این ماتریس در تحلیل حساسیت و بهینه سازی مجدد شبکه های کوتاه ترین مسیر هر زوج گره کاربرد مهمی دارد. ماتریس انتقال با T نشان داده شده است. تعداد جابه جایی یا انتقال کمان را "عدد انتقال کمان"^۲ نامیده و آن را با t_{ik} نشان می دهیم که همان عناصر ماتریس انتقال هستند. به عبارت بهتر، t_{ik} بیانگر تعداد جابه جایی بین گره های i و k در جواب بهینه است؛ یعنی با حذف کمان $i-k$ ، t_{ik} جابه جایی یا انتقال از جابه جایی های بهینه شبکه دچار اختلال خواهد شد. هرچه مقدار عددی آن بزرگ تر باشد اهمیت آن کمان در تعداد جابه جایی بهینه شبکه بیشتر خواهد بود و برعکس، مقدار عددی کوچک تر آن، معرف اهمیت کم تر آن کمان در جابه جایی های بهینه شبکه است. نقش یا ارزش هر کمان در جابه جایی شبکه را "ارزش انتقال کمان"^۳ نام گذاری کرده که w_{ik} نشان دهنده آن است. ارزش انتقال هر کمان به صورت نسبت عدد انتقال یا جابه جایی هر کمان به کل جابه جایی در جواب بهینه شبکه تعریف شده و بر حسب درصد نشان داده می شود. به عبارت بهتر، با حذف هر کمان دلخواه $i-k$ ، w_{ik} درصد از جابه جایی های بهینه شبکه دچار اختلال خواهد شد. عدد و ارزش انتقال هر کمان کاربرد اساسی در تحلیل حساسیت و بهینه سازی مجدد شبکه های کوتاه ترین مسیر دارد، زیرا به کمک آن ها می توان نقش کمان ها را در جابه جایی شبکه ها بررسی نمود و در نتیجه، کمان های زاید را شناسائی و حذف کرد. ماتریس انتقال T ، همانند دو ماتریس مسافت D و مسیر R یک ماتریس مربع $n \times n$ است. عناصر این ماتریس بر اساس عناصر ماتریس R محاسبه می شود. روش محاسبه عناصر ماتریس T به ترتیب رابطه (۲) است.

$$t_{ik} = \begin{cases} - & \text{if } i = k \\ \sum_{k=1, i \neq k, r_{ik}^F = i}^{k=n} r_{ik}^F + \sum_{i=1, i \neq k, r_{ik}^F = k}^{i=n} r_{ik}^F & \text{if } r_{ik}^F = k \\ 0 & \text{Otherwis} \end{cases} \quad (2)$$

اگر t معرف جمع عناصر ماتریس انتقال باشد در آن صورت به کمک رابطه (۳) ارزش انتقال هر کمان در شبکه قابل محاسبه است.

$$t = \sum_{i=1}^{i=n} \sum_{k=1, i \neq k}^{k=n} t_{ik} \quad (3)$$

$$w_{ik} = \frac{t_{ik}}{t} \times 100$$

عدد و ارزش انتقال برای گره ها هم قابل تعریف است. به عبارت بهتر، می توان ارزش جابه جایی هر گره i در کل جابه جایی شبکه را محاسبه نمود. هر گره i به $n-1$ گره دیگر جابه جایی دارد و همه ی $n-1$ گره به گره i هم

¹ Transposition Matrix² Arc Transposition Number³ Arc Transposition Worth

جابه‌جایی دارند، بنابراین کل جابه‌جایی از گره i به سایر گره‌ها و برعکس معادل $2(n-1)$ خواهد بود. اگر به مقدار $2(n-1)$ تعداد دفعاتی که گره i واسطه گره‌ها قرار می‌گیرد اضافه شود تعداد کل جابه‌جایی گره i محاسبه خواهد شد و این به شرطی است که در ماتریس R عنصری با مقدار ∞ وجود نداشته باشد، در غیر این صورت، تعداد ∞ ها از مقدار $2(n-1)$ کم خواهد شد. اگر in_i و t_i به ترتیب نشان‌دهنده تعداد واسطه‌ها و عدد جابه‌جایی گره i باشد، مقادیر آن‌ها به کمک روابط (۴) و (۵) قابل محاسبه است. برای محاسبه تعداد واسطه‌های هر گره i ، لازم است در ماتریس R_F به جز ستون i -ام، عدد i آن شمارش گردد.

$$in_i = \sum_{k=1, k \neq i}^n \sum_{i=1, i \neq k}^n r_{ik}^F \quad (4)$$

$$t_i = 2(n-1) + in_i, r_{ik} \neq \infty \quad (5)$$

اگر it و tt به ترتیب معرف کل واسطه‌ها و کل جابه‌جایی باشند، مقدار ارزش جابه‌جایی گره i یا w_i با استفاده از رابطه (۶) محاسبه می‌شود.

$$it = \sum_{i=1}^n in_i$$

$$tt = it + 2n(n-1)$$

$$w_i = \frac{t_i}{tt}, r_{ik} \neq \infty \times 100 \quad (6)$$

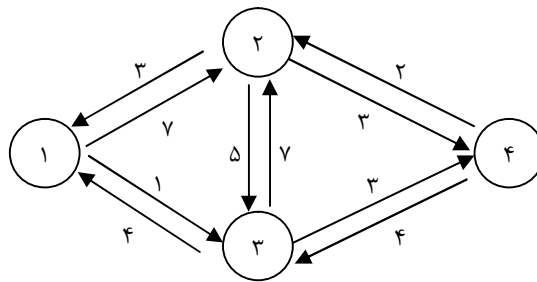
به عبارت بهتر، w_i نشان‌دهنده درصد ارزش یا اهمیت گره i در کل جابه‌جایی‌های بهینه شبکه است یعنی با حذف گره i ، مقدار w_i درصد از جابه‌جایی‌های شبکه دچار اختلال خواهد شد. بزرگی عددی آن معرف نقش مهمتر گره در جابه‌جایی بهینه خواهد بود. عدد و ارزش انتقال هر گره، همانند عدد و ارزش انتقال هر کمان، کاربرد مهمی در تحلیل حساسیت و بهینه‌سازی مجدد شبکه‌های کوتاه‌ترین مسیر دارد، زیرا به کمک آن‌ها می‌توان نقش گره‌ها را در جابه‌جایی بهینه شبکه تعیین نمود و در نتیجه، گره‌ها با ارزش جابه‌جایی کم‌تر را شناسایی کرد. برای محاسبه تعداد واسطه‌های هر گره i ، لازم است در ماتریس R_F به جز ستون i -ام، عدد i آن شمارش گردد.

۵ مثال نمونه

در ادامه، با ارایه یک مثال، روش اجرا، حجم محاسبات و صحت اجرای سه الگوریتم مستطیل آبشاری، فلویید-وارشال و ماتریس تجدید نظر شده هو بررسی می‌شود. ابتدا، مثال به کمک الگوریتم مستطیل آبشاری حل می‌شود.

۵-۱ حل مثال نمونه به کمک الگوریتم مستطیل آبشاری

شکل ۴ را در نظر بگیرید. هدف پیدا کردن کوتاه‌ترین مسیر و مسافت بین هر زوج گره به کمک الگوریتم مستطیل آبشاری است.



شکل ۴. مثال نمونه

گام اول: چون n برابر ۴ است، دو ماتریس مربع 4×4 مسافت و مسیر، D_s و R_s به ازای $s = B, F$ که در آن s را مرحله نامند، تعریف کنید.

گام دوم: در مرحله $s = B$ ، ماتریس های D_B و R_B را محاسبه کنید.

$$D_B = \begin{bmatrix} \circ & 7 & 1 & \infty \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix}, R_B = \begin{bmatrix} - & 2 & 3 & - \\ 1 & - & 3 & 4 \\ 1 & 2 & - & 4 \\ - & 2 & 3 & - \end{bmatrix}$$

گام سوم: در مرحله $s = F$ ، با استفاده از بند الف گام سوم، قطر اصلی ماتریس های D_B و R_B را تکرار کنید.

$$D_F = \begin{bmatrix} \circ & & & \\ & \circ & & \\ & & \circ & \\ & & & \circ \end{bmatrix}, R_F = \begin{bmatrix} - & & & \\ & - & & \\ & & - & \\ & & & - \end{bmatrix}$$

گام سوم: با استفاده از بند ب گام سوم، برای محاسبه هر عنصر d_{ik} ، $i \neq k$ از ماتریس D_F ، به تعداد $4-2$ مستطیل در ماتریس D_B رسم می شود. این مستطیل ها معرف محاسبات و مقایسات الگوریتم هستند. برای محاسبه عنصر d_{12} ، تعداد ۲ مستطیل به نحوی رسم کنید که یک راس آن عنصر d_{12} یا ۷ و رئوس روبروی آن صفرهای قطر اصلی یا عناصر d_{33} و d_{44} باشند دو راس دیگر بسادگی قابل استخراج بوده که همان مقادیر عناصر d_{ij} و d_{jk} هستند. برای محاسبه هر عنصر d_{ik} ، $i \neq k$ ، با صفرهای d_{ii} و d_{kk} نمی توان مستطیل ایجاد کرد، به همین دلیل تعداد مستطیل ها به اندازه $n-2$ خواهد بود.

$$D_B = \begin{bmatrix} \circ & \boxed{\begin{matrix} 7 & 1 \\ \circ & 5 \\ 7 & \circ \\ 2 & 4 \end{matrix}} & \infty \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix} d_{12} = \text{Min}(7, 7+1, 2+\infty) = 7 \rightarrow r_{12} = r_{12}$$

با مقایسه عنصر d_{ik} یا $d_{ir} = 7$ با عنصر d_{ik} یا $d_{rr} = 7$ از مستطیل رسم شده کوچک، چون $d_{ir} \leq d_{rr}$ است و با مقایسه عنصر d_{ik} یا $d_{ir} = 7$ با عنصر d_{ij} یا $d_{ir} = \infty$ از مستطیل رسم شده بزرگ، چون $d_{ir} \leq d_{ij}$ است در نتیجه، مقدار عنصر d_{ik} یا d_{ir} نیاز به محاسبه نداشته و ثابت خواهد ماند. به جهت وابسته بودن عناصر ماتریس مسیر R به عناصر ماتریس مسافت D ، چون مقدار عنصر d_{ir} تغییر نکرد، آنگاه مقدار عنصر r_{ir} هم تغییر نخواهد کرد. در اجرای عملی الگوریتم، برای محاسبه هر عنصر دلخواه d_{ik} با $i \neq k$ ، رسم مستطیل‌ها و در نتیجه، انجام محاسبات و مقایسات به صورت ساده انجام شده و نیازی به رسم مستطیل به صورت فیزیکی و محاسبات مربوطه نخواهد بود.

به طور مشابه می‌توان سایر عناصر ماتریس مسافت را محاسبه نمود. در ادامه چند عنصر به صورت تصادفی انتخاب و محاسبه شده است.

$$D_B = \begin{bmatrix} \circ & 7 & 1 & \infty \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix} d_{ir} = \text{Min}(\infty, 7+3, 1+3) = 4$$

چون مقدار عنصر d_{ir} تغییر کرد و این تغییر مربوط به مستطیل پررنگ، یعنی ۴ و وابسته با صفر عنصر d_{rr} است به همین دلیل مقدار عنصر $r_{ir} = 3$ خواهد شد. در ادامه محاسبات رفت، مقدار عنصر d_{ir} را با ۴ جایگزین کنید.

$$D_B = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix} d_{rr} = \text{Min}(7, 2+3, 4+7) = 5$$

چون مقدار عنصر d_{rr} تغییر کرد و این تغییر مربوط به مستطیل پررنگ، یعنی ۵ و وابسته با صفر عنصر d_{rr} است به همین دلیل مقدار عنصر $r_{rr} = 4$ خواهد شد. در ادامه محاسبات رفت، مقدار عنصر d_{rr} را با ۵ جایگزین کنید.

$$D_B = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 5 & 3 \\ 4 & 5 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix} d_{ri} = \text{Min}(\infty, 3+2, 4+4) = 5 \rightarrow r_{ri} = 2$$

با ادامه‌ی اجرای الگوریتم برای محاسبات رفت به سادگی می‌توان سایر عناصر ماتریس D_F را محاسبه کرد و به طور مشابه ماتریس R_F نیز محاسبه خواهد شد.

$$D_F = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 4 & 3 \\ 4 & 5 & \circ & 3 \\ 5 & 2 & 4 & \circ \end{bmatrix}, R_F = \begin{bmatrix} - & 2 & 3 & 3 \\ 1 & - & 1 & 4 \\ 1 & 4 & - & 4 \\ 2 & 2 & 3 & - \end{bmatrix}$$

برای انجام محاسبات برگشت، از عنصر d_{rr} ماتریس D_F شروع کرده و تعداد ۲-۴ مستطیل برای محاسبه هر عنصر d_{ik} رسم کنید. این محاسبات همانند محاسبات رفت است.

$$D_F = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 4 & 3 \\ 4 & 5 & \circ & 3 \\ 5 & 2 & 4 & \circ \end{bmatrix} \quad d_{33} = \text{Min}(4, 2+4, 5+1) = 4 \rightarrow r_{33} = r'_{33}$$

محاسبات برگشت را تا عنصر d_{12} ادامه دهید.

$$D_F = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 4 & 3 \\ 4 & 5 & \circ & 3 \\ 5 & 2 & 4 & \circ \end{bmatrix} \quad d_{12} = \text{Min}(7, 5+1, 2+4) = 6$$

چون مقدار عنصر d_{12} ، تغییر کرد در نتیجه مقدار عنصر r_{12} هم تغییر خواهد کرد. این تغییر هم مربوط به مستطیل اول، یعنی ۶ و متناسب با صفر d_{33} بوده و همینطور این تغییر می تواند مربوط به مستطیل دوم با مقدار ۶ و متناسب با صفر d_{44} باشد. معیار انتخاب مقدار عنصر r'_{12} ، عناصر r'_{13} و r'_{14} ماتریس R_F است چون مقادیر این عناصر برابر ۳ است؛ لذا مقدار عنصر $r'_{12} = 3$ خواهد بود. در ادامه محاسبات برگشت، مقدار عنصر d_{12} را با ۶ جایگزین کنید. در پایان اجرای الگوریتم، ماتریس های مسیر و مسافت نهایی به ترتیب زیر محاسبه خواهند شد:

$$D_F = \begin{bmatrix} \circ & 6 & 1 & 4 \\ 3 & \circ & 4 & 3 \\ 4 & 5 & \circ & 3 \\ 5 & 2 & 4 & \circ \end{bmatrix}, R_F = \begin{bmatrix} - & 3 & 3 & 3 \\ 1 & - & 1 & 4 \\ 1 & 4 & - & 4 \\ 2 & 2 & 3 & - \end{bmatrix}$$

با وجود ماتریس های D_F و R_F می توان کوتاه ترین مسافت و مسیر را از هر گره i تا هر گره j را به دست آورد. به عنوان مثال، کوتاه ترین مسافت بین دو گره ۱ و ۲ عبارت است از ۶ یا ($d_{12} = 6$) و کوتاه ترین مسیر به ترتیب زیر قابل محاسبه است:

$$r'_{12} = 3, r'_{33} = 4, r'_{44} = 2 \rightarrow 1-3-4-2$$

در این قسمت، مثال نمونه را به کمک الگوریتم فلوید - وارshall حل کرده تا روش اجرای این الگوریتم، حجم محاسبات و صحت اجرای آن بررسی گردد. در این الگوریتم، برای محاسبه عناصر ماتریس مسیر، محاسبات مشابه ماتریس مسافت انجام می شود.

۵-۲ حل مثال نمونه به کمک الگوریتم فلوید - وارshall

شکل ۴ را مجددا در نظر بگیرید. هدف پیدا کردن کوتاه ترین مسیر و مسافت بین هر زوج گره در شبکه به کمک الگوریتم فلوید - وارshall است [۱ و ۲].

ابتدا ماتریس های D_0 و R_0 را محاسبه کنید. با پیاده سازی گام ۲ الگوریتم فلوید - وارshall به راحتی می توان دو ماتریس D_0 و R_0 را به صورت زیر به دست آورد.

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

برای محاسبه ماتریس‌های D_1 و R_1 ، با توجه به اینکه $j=1$ است، لذا مقادیر سطر و ستون ۱ تکرار خواهد شد. قطر اصلی در کلیه مراحل ثابت خواهد ماند. سایر عناصر با پیمودن گام ۳ الگوریتم فلویید-وارشال قابل محاسبه است. در ادامه چند عنصر این ماتریس‌ها محاسبه شده‌اند.

$$d_{rr} = \min(d_{rr}, d_{r1} + d_{1r}) = \min(5, 3+1) = 4$$

$$d_{rf} = \min(d_{rf}, d_{r1} + d_{1f}) = \min(3, 4+\infty) = 3$$

با ادامه محاسبات، ماتریس D_1 مطابق زیر به دست خواهد آمد:

$$D_1 = \begin{bmatrix} \infty & 7 & 1 & \infty \\ 3 & \infty & 4 & 3 \\ 4 & 7 & \infty & 3 \\ \infty & 2 & 4 & \infty \end{bmatrix}$$

با انجام محاسبات زیر، ماتریس مسیر R_1 قابل محاسبه خواهد بود:

$$d_{rr} \geq d_{r1} + d_{1r} \equiv 5 \geq 3+1 \rightarrow r_{rr} = j = 1$$

$$d_{rf} \leq d_{r1} + d_{1f} \equiv 3 \leq 4+\infty \rightarrow r_{rf} = r_{rr} = 4$$

با ادامه محاسبات، ماتریس R_1 مطابق زیر به دست خواهد آمد:

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

برای محاسبه ماتریس‌های D_2 و R_2 ، با توجه به اینکه $j=2$ است؛ لذا مقادیر سطر و ستون ۲ تکرار خواهد شد. با پیمودن گام ۳ الگوریتم فلویید-وارشال، محاسبات چند عنصر به صورت زیر خواهد بود:

$$d_{rr} = \min(d_{rr}, d_{r2} + d_{2r}) = \min(1, 7+4) = 1$$

$$d_{rf} = \min(d_{rf}, d_{r2} + d_{2f}) = \min(3, 7+3) = 3$$

بر اساس محاسبات انجام شده، ماتریس D_2 مطابق زیر به دست خواهد آمد:

$$D_2 = \begin{bmatrix} \infty & 7 & 1 & 10 \\ 3 & \infty & 4 & 3 \\ 4 & 7 & \infty & 3 \\ 5 & 2 & 4 & \infty \end{bmatrix}$$

$$d_{rr} \leq d_{r2} + d_{2r} \equiv 1 \leq 7+4 \rightarrow r_{rr} = r_{rr} = 2$$

$$d_{rf} \leq d_{r2} + d_{2f} \equiv 3 \leq 7+3 \rightarrow r_{rf} = r_{rr} = 4$$

در نتیجه، ماتریس R_2 مطابق زیر به دست خواهد آمد:

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

با ادامه اجرای الگوریتم فلوید-وارشال ماتریس های D_f و R_f به ترتیب زیر محاسبه می شوند:

$$D_f = \begin{bmatrix} 0 & 6 & 1 & 4 \\ 3 & 0 & 4 & 3 \\ 4 & 5 & 0 & 3 \\ 5 & 2 & 4 & 0 \end{bmatrix}, R_f = \begin{bmatrix} - & 4 & 3 & 3 \\ 1 & - & 1 & 4 \\ 1 & 4 & - & 4 \\ 2 & 2 & 3 & - \end{bmatrix}$$

با در دست داشتن دو ماتریس D_f و R_f می توان مسافت و مسیر را از هر گره دلخواه i به هر گره دلخواه k به دست آورد. به عنوان مثال، کوتاه ترین مسافت بین دو گره ۱ و ۲ عبارت است از ۶ یا $(d_{12} = 6)$ و کوتاه ترین مسیر به ترتیب زیر قابل محاسبه است:

$$r_{12} = 4, r_{f2} = 2 \rightarrow 1-4-2$$

الگوریتم فلوید-وارشال مسیر ۱-۴-۲ را کوتاه ترین مسیر محاسبه کرده است در صورتی که مسیر صحیح، مسیر ۱-۳-۴-۲ است. برای تصحیح مسیر باید از وجود کمان مستقیم بین هر دو گره i و k استفاده نمود. به عبارت بهتر، اگر $r_{ik} = k$ باشد بین دو گره i و k کمان مستقیم وجود دارد در غیر این صورت، اگر $r_{ik} = l$ باشد حداقل یک گره واسطه l بین گره های i و k وجود دارد. مسیر استخراج شده از الگوریتم فلوید-وارشال ۱-۴-۲ بوده که از سه گره تشکیل شده است چون $r_{14} = 3$ است، لذا گره ۳ واسطه دو گره ۱ و ۴ خواهد بود. چون $r_{34} = 4$ است بنابراین گره واسطه بین دو گره ۳ و ۴ وجود ندارد، بنابراین زیر مسیر ۱-۴ به زیر مسیر ۱-۳-۴ تبدیل خواهد شد. چون $r_{13} = 2$ است لذا گرهی واسطه بین گره های ۱ و ۳ وجود ندارد. بنابراین، کوتاه ترین مسیر از گره ۱ به ۲ برابر مسیر ۱-۳-۴-۲ خواهد بود. در نتیجه، الگوریتم جذاب فلوید-وارشال در محاسبه ماتریس مسیر برخی از شبکه ها با ساختار خاص، دارای محدودیت محاسباتی است.

در ادامه، مثال نمونه را با الگوریتم ماتریس تجدیدنظر شده هو حل کرده تا روش اجرای این الگوریتم، حجم محاسبات و صحت اجرای آن بررسی شود. همانند الگوریتم فلوید-وارشال، در این الگوریتم نیز، برای محاسبه عناصر ماتریس مسیر محاسبات مشابه ماتریس مسافت انجام می شود.

۳-۵ حل مثال نمونه به کمک روش ماتریس تجدیدنظر شده هو

شکل ۴ را مجددا در نظر بگیرید. هدف پیدا کردن کوتاه ترین مسیر و مسافت بین هر زوج گره به کمک روش ماتریس تجدیدنظر شده هو است [۴].

ابتدا ماتریس مسافت را محاسبه کنید. چون n برابر ۴ است، این ماتریس مربع و 4×4 خواهد بود. در الگوریتم ماتریس تجدیدنظر شده هواز جدول برای نمایش عناصر مسافت و مسیر استفاده شده است. در اینجا، برای سهولت نگارش، بجای جدول از ماتریس استفاده می‌شود.

$$\begin{array}{c}
 \begin{array}{c} 1 & 2 & 3 & 4 \\
 \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array}
 \end{array}
 \begin{array}{|c|c|c|c|}
 \hline
 \circ & 7 & 1 & \infty \\
 \hline
 3 & \circ & 5 & 3 \\
 \hline
 4 & 7 & \circ & 3 \\
 \hline
 \infty & 2 & 4 & \circ \\
 \hline
 \end{array}
 \end{array}
 \rightarrow D = \begin{bmatrix} \circ & 7 & 1 & \infty \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix}$$

محاسبات رفت را به ترتیب زیر انجام دهید.

برای انجام محاسبات رفت، جهت محاسبه هر عنصر d_{ik}^F ، به تعداد $n=4$ مقایسه نیاز است. در ادامه تعدادی از عناصر ماتریس مسافت محاسبه شده‌اند.

$$d_{12}^F = \text{Min}(d_{11} + d_{12}, d_{12} + d_{22}, d_{12} + d_{13}, d_{12} + d_{14}, d_{12} + d_{24}) = \text{Min}(\circ + 7, 7 + \circ, 1 + 7, \infty + 2) = 7$$

$$d_{14}^F = \text{Min}(d_{11} + d_{14}, d_{12} + d_{24}, d_{13} + d_{34}, d_{14} + d_{44}) = \text{Min}(\circ + \infty, 7 + 3, 1 + 3, \infty + \circ) = 4$$

چون مقدار عنصر d_{14} ، تغییر کرد، لذا مقدار جدید عنصر را جایگزین مقدار قبلی کرده، سپس محاسبات رفت را ادامه دهید. به عبارت بهتر، ماتریس مسافت جدید به ترتیب زیر خواهد شد:

$$D = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 5 & 3 \\ 4 & 7 & \circ & 3 \\ \infty & 2 & 4 & \circ \end{bmatrix}$$

$$d_{23}^F = \text{Min}(d_{21} + d_{13}, d_{22} + d_{23}, d_{23} + d_{33}, d_{23} + d_{24}, d_{23} + d_{34}) = \text{Min}(3 + 1, \circ + 5, 5 + \circ, 3 + 4) = 4$$

مقدار عنصر d_{23} را به ۴ تغییر داده و محاسبات رفت را تا عنصر d_{34} ادامه دهید.

$$d_{34}^F = \text{Min}(d_{31} + d_{14}, d_{32} + d_{24}, d_{33} + d_{34}, d_{34} + d_{44}) = \text{Min}(4 + 4, 3 + 5, \circ + 3, 3 + \circ) = 3$$

در پایان محاسبات رفت، ماتریس مسافت به ترتیب زیر خواهد شد:

$$D^F = \begin{bmatrix} \circ & 7 & 1 & 4 \\ 3 & \circ & 4 & 3 \\ 4 & 5 & \circ & 3 \\ 5 & 2 & 4 & \circ \end{bmatrix}$$

محاسبات برگشت را از عنصر d_{33} ماتریس مسافت D^F شروع کرده و تا عنصر d_{12} ادامه دهید. در ادامه، محاسبه برگشت تعدادی از عناصر آمده است. برای انجام محاسبات برگشت، به تعداد $n=4$ مقایسه برای محاسبه هر عنصر d_{ik}^F نیاز خواهد بود.

$$d_{33}^F = \text{Min}(d_{31} + d_{13}, d_{32} + d_{23}, d_{33} + d_{33}, d_{33} + d_{34}, d_{33} + d_{43}) = \text{Min}(5 + 1, 2 + 4, \circ + 4, 4 + \circ) = 4$$

$$d_{41}^F = \text{Min}(d_{41} + d_{11}, d_{42} + d_{21}, d_{43} + d_{31}, d_{44} + d_{44}) = \text{Min}(\circ + 5, 2 + 3, 4 + 4, 5 + \circ) = 5$$

$$d_{12}^F = \text{Min}(d_{11} + d_{12}, d_{12} + d_{22}, d_{13} + d_{22}, d_{13} + d_{32}, d_{14} + d_{24}) = \text{Min}(0 + 7, 7 + 0, 1 + 5, 4 + 2) = 6$$

در پایان محاسبات برگشت، ماتریس مسافت به ترتیب زیر خواهد شد.

$$D^B = \begin{bmatrix} 0 & 6 & 1 & 4 \\ 3 & 0 & 4 & 3 \\ 4 & 5 & 0 & 3 \\ 5 & 2 & 4 & 0 \end{bmatrix}$$

برای محاسبه ماتریس مسیر، چون n برابر ۴ است، ماتریس مسیر مربع 4×4 ، R را به ترتیب زیر محاسبه کنید.

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

چون $d_{12} = d_{13} + d_{32} \equiv 6 = 1 + 5 \rightarrow r_{12} = j = 3$ خواهد بود. از طرفی دیگر، چون

$d_{12} = d_{14} + d_{42} \equiv 6 = 4 + 2 \rightarrow r_{12} = j = 4$ خواهد شد. فرض کنید مقدار $r_{12} = j = 3$ در نظر گرفته شود.

محاسبه تعدادی از عناصر ماتریس مسیر به ترتیب زیر خواهد بود.

$$d_{13} = d_{13} + d_{33} \equiv 1 = 1 + 0 \rightarrow r_{13} = 3$$

$$d_{14} = d_{13} + d_{34} \equiv 4 = 1 + 3 \rightarrow r_{14} = j = 3$$

$$d_{41} = d_{42} + d_{21} \equiv 5 = 2 + 3 \rightarrow r_{41} = j = 2$$

در پایان محاسبات، ماتریس مسیر R به ترتیب زیر به دست خواهد آمد:

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

با وجود ماتریس‌های D و R می‌توان کوتاه‌ترین مسافت و مسیر را از هر گره i تا هر گره k را به دست آورد.

به عنوان مثال، کوتاه‌ترین مسافت بین دو گره ۱ و ۲ عبارت است از ۶ یا ($d_{12} = 6$) و کوتاه‌ترین مسیر به ترتیب زیر

قابل محاسبه است:

$$r_{12} = 3, r_{32} = 4, r_{42} = 2 \rightarrow 1-3-4-2$$

اگر $r_{12} = j = 4$ فرض شود. الگوریتم هو، کوتاه‌ترین مسافت را به ترتیب زیر محاسبه خواهد کرد:

$$r_{12} = 4, r_{42} = 2 \rightarrow 1-4-2$$

به عبارت بهتر، الگوریتم هو همانند الگوریتم فلویید-وارشال دارای محدودیت محاسباتی خواهد بود. تصحیح مسیر همانند روش ارایه شده در الگوریتم فلویید-وارشال است.

در پایان، از مثال نمونه، برای محاسبه ماتریس انتقال و تحلیل آن استفاده می‌شود. یکی از کاربردهای این

ماتریس، شناخت کمان‌های کم‌اهمیت و زاید در شبکه بهینه است.

۵-۴ محاسبه ماتریس انتقال در مثال نمونه

شکل ۴ را مجدداً در نظر بگیرید. هدف، محاسبه ماتریس انتقال این شبکه است. فرض کنید ماتریس‌های D_B ، R_B ، D_F و R_F بر اساس یکی از الگوریتم‌های جبر ماتریسی جهت محاسبه ماتریس‌های مسافت و مسیر هر زوج گره از قبیل مستطیل آبشاری محاسبه شده باشند.

$$D_B = \begin{bmatrix} 0 & 7 & 1 & \infty \\ 3 & 0 & 5 & 3 \\ 4 & 7 & 0 & 3 \\ \infty & 2 & 4 & 0 \end{bmatrix} \rightarrow D_F = \begin{bmatrix} 0 & 6 & 1 & 4 \\ 3 & 0 & 4 & 3 \\ 4 & 5 & 0 & 3 \\ 5 & 2 & 4 & 0 \end{bmatrix}, R_B = \begin{bmatrix} - & 2 & 3 & - \\ 1 & - & 3 & 4 \\ 1 & 2 & - & 4 \\ - & 2 & 3 & - \end{bmatrix} \rightarrow$$

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

برای محاسبه عناصر ماتریس انتقال که براساس فرمت نام‌گذاری ماتریس‌های مسافت و مسیر الگوریتم مستطیل آبشاری با T_F نشان داده شده از ماتریس R_F استفاده کنید. قطر اصلی ماتریس T_F همانند ماتریس R_F برابر " - " خواهد بود. برای محاسبه عناصر $r_{ik} = k$ ، یا برای محاسبه عنصر t_{12} ، در سطر ۱ ماتریس R_F ، تعداد عناصری که برابر ۲ هستند + تعداد عناصر برابر ۱ ستون ۲ شمارش گردد، برای محاسبه عنصر t_{13} ، تعداد عناصری که برابر ۳ هستند + تعداد عناصر برابر ۱ ستون ۳ شمارش شده و غیره. برای عناصری که مقدار $r_{ik} \neq k$ است مقدار ۰ را جایگزین کنید. این محاسبات در ادامه آمده است.

$$R_F = \begin{bmatrix} - & 3 & 3 & 3 \\ 1 & - & 1 & 4 \\ 1 & 4 & - & 4 \\ 2 & 2 & 3 & - \end{bmatrix} \rightarrow T_F = \begin{bmatrix} - & 0 & 3+1 & 0 \\ 2+1 & - & 0 & 1+0 \\ 1+0 & 0 & - & 2+1 \\ 0 & 2+1 & 1+0 & - \end{bmatrix} = \begin{bmatrix} - & 0 & 4 & 0 \\ 3 & - & 0 & 1 \\ 1 & 0 & - & 3 \\ 0 & 3 & 1 & - \end{bmatrix}$$

عناصر ماتریس T_F نشان‌دهنده عدد انتقال کمان است. برای مثال، $t_{12} = 3$ به این معنی است که کمان $2 \rightarrow 1$ ، ۳ مرتبه در کل جابه‌جایی شبکه مورد استفاده قرار گرفته است. به عبارت بهتر، با حذف این کمان، ۳ مسیر بهینه دچار اختلال خواهد شد. در ادامه عدد انتقال و ارزش انتقال هر کمان آمده است.

$$\begin{array}{llllll} t_{12} = 0, & t_{13} = 4, & t_{14} = 0 & t_{21} = 3, & t_{23} = 0, & t_{24} = 1 \\ t_{31} = 1, & t_{32} = 0, & t_{34} = 3 & t_{41} = 0, & t_{42} = 3, & t_{43} = 1 \\ t = 0+4+0+3+0+1+1+0+3+0+3+1 = 16 \\ w_{12} = \frac{0}{16} = 0, & w_{13} = \frac{4}{16} = 25\%, & w_{14} = \frac{0}{16} = 0 \\ w_{21} = \frac{3}{16} = 18.75\%, & w_{23} = \frac{0}{16} = 0, & w_{24} = \frac{1}{16} = 6.25\% \\ w_{31} = \frac{1}{16} = 6.25\%, & w_{32} = \frac{0}{16} = 0, & w_{34} = \frac{3}{16} = 18.75\% \end{array}$$

$$w_{f1} = \frac{0}{16} = 0, \quad w_{f2} = \frac{3}{16} = 18.75\%, \quad w_{f3} = \frac{1}{16} = 6.25\%$$

در نتیجه، ارزش انتقال کمان‌های $1 \rightarrow 3$ برابر $\frac{4}{16}$ یا 25% ، ارزش انتقال کمان‌های $1 \rightarrow 2$ ، $2 \rightarrow 3$ و $2 \rightarrow 4$ برابر $\frac{3}{16}$ یا 18.75% ، ارزش انتقال کمان‌های $3 \rightarrow 1$ ، $3 \rightarrow 2$ و $3 \rightarrow 4$ برابر $\frac{1}{16}$ یا 6.25% و ارزش انتقال کمان‌های $1 \rightarrow 4$ ، $2 \rightarrow 1$ ، $3 \rightarrow 1$ ، $3 \rightarrow 2$ و $4 \rightarrow 1$ برابر 0 است. به عبارت بهتر، اگر کمان $1 \rightarrow 3$ حذف شوند، 25% درصد و با حذف هر یک از کمان‌های $1 \rightarrow 2$ ، $2 \rightarrow 3$ و $2 \rightarrow 4$ ، 18.75% و با حذف هر یک از کمان‌های $1 \rightarrow 4$ ، $2 \rightarrow 1$ ، $3 \rightarrow 1$ و $3 \rightarrow 2$ ، 6.25% از جابه‌جایی‌های بهینه شبکه دچار آسیب خواهد شد و با حذف کمان‌های زاید $1 \rightarrow 2$ ، $2 \rightarrow 3$ و $3 \rightarrow 1$ خللی در جابه‌جایی‌ها ایجاد نخواهد شد. ارزش انتقال 0 برای کمان‌ها به معنی عدم وجود کمان یا حذف آن کمان بدون ایجاد خلل در جابه‌جایی‌های بهینه شبکه است.

برای محاسبه عدد و ارزش انتقال هر گره، تعداد واسطه‌های گره را محاسبه کنید. برای محاسبه واسطه‌های گره 1 ، 2 ، 3 و 4 لازم است شماره گره i در غیر ستون i ماتریس R_F شمارش گردد. برای مثال، جهت محاسبه تعداد واسطه‌های گره 1 ، ستون 1 را حذف کرده و عدد 1 را در عناصر باقیمانده ماتریس مسیر شمارش کنید. به کمک ماتریس‌های زیر واسطه‌های هر گره بسادگی قابل استخراج خواهد بود.

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

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

بنابراین، تعداد واسطه‌ها برای گره 1 برابر 1 ، تعداد واسطه‌ها برای گره 2 برابر 1 ، تعداد واسطه‌ها برای گره 3 برابر 2 و تعداد واسطه‌ها برای گره 4 برابر 1 خواهد بود. در نتیجه کل واسطه‌ها برابر 5 است. چون در ماتریس R_F عنصر ∞ وجود ندارد در آن صورت، هر گره به تعداد $6 - (4 - 1) = 2$ جابه‌جایی دارد. عدد و ارزش انتقال هر گره به ترتیب زیر قابل محاسبه خواهد بود:

$$t_1 = 1 + 6 = 7, \quad t_2 = 1 + 6 = 7, \quad t_3 = 2 + 6 = 8, \quad t_4 = 1 + 6 = 7$$

$$w_1 = \frac{1+6}{5+2 \times 4 \times 3} = \frac{7}{29} = 24.14\%, \quad w_2 = \frac{1+6}{5+2 \times 4 \times 3} = \frac{7}{29} = 24.14\%$$

$$w_3 = \frac{2+6}{5+2 \times 4 \times 3} = \frac{8}{29} = 27.58\%, \quad w_4 = \frac{1+6}{5+2 \times 4 \times 3} = \frac{7}{29} = 24.14\%$$

در نتیجه، ارزش انتقال گره‌های ۱، ۲ و ۴ برابر ۲۴/۱۴ درصد و گره ۳ برابر ۲۷/۵۸ درصد خواهد بود. به عبارت بهتر، با حذف گره ۱، ۲ یا ۴، ۲۴/۱۴ درصد و با حذف گره ۳، ۲۷/۵۸ درصد از جابجایی‌های بهینه شبکه دچار اختلال خواهند شد.

۶ نتیجه و جمع‌بندی

در این مقاله الگوریتم جدیدی بانام الگوریتم مستطیل آبخاری براساس بهبود و بازتعریف چند الگوریتم دقیق برای محاسبه کوتاه‌ترین مسافت و مسیر بین هرزوج گره در شبکه‌های کوتاه‌ترین مسیر به همراه ماتریس انتقال که کاربرد مهمی در تحلیل حساسیت و بهینه‌سازی مجدد اینگونه شبکه‌ها دارد، ارائه شده است. مزیت اساسی الگوریتم مستطیل آبخاری نسبت به الگوریتم‌های مشابه این است که تمام محاسبات و عملیات ریاضی این الگوریتم، در قالب $n-2$ مستطیل که معرف روابط ریاضی هستند، پیاده‌سازی شده است. الگوریتم مستطیل آبخاری، از طرفی منجر به کاهش قابل توجه حجم محاسبات و در نتیجه کاهش زمان انجام محاسبات در الگوریتم‌های مشابه از قبیل الگوریتم‌های هو و فلوید-وارشال شده است (به جهت وابسته کردن محاسبات ماتریس مسیر به ماتریس مسافت) و از طرفی دیگر، به دلیل اجرای مستطیلی این الگوریتم، در مسایل بزرگ بسیار جذاب و ساده خواهد بود. علاوه بر این، فهم الگوریتم برای اهداف آموزشی بسیار مناسب و آسان‌تر خواهد بود.

منابع

- [1] Warshall, S., (1962). A theorem on boolean matrices. Journal of the ACM, 9, 11-12.
- [2] Floyd, R. W., (1962). Algorithm 97: Shortest path. Communications of the ACM, 5, 345.
- [3] Farbey, B. A., Land, A. H., Murchland, J. D., (1967). The cascade algorithm for finding all shortest distances in a directed graph. Management Science, 14(1), 19-28.
- [4] Hu, T., (1967). Revised matrix algorithms for shortest paths. SIAM Journal of Applied Mathematics, 15, 207-218.
- [5] Aini, A., Salehipour, A., (2012). Speeding up the floyd-warshall algorithm for the cycled shortest path problem. Applied Mathematics Letter, 25, 1-5.
- [6] Dijkstra, E. W., (1959). A note on two problems in connection with Graphs. Numerische Mathematik, 1, 269-271.
- [7] Ahuja, R. K., Magnanti, T. L., Orlin, J. B., (1993). Network flows: theory, algorithms, and applications. John Wiley & Sons.
- [8] Hougardy, S., (2010). The floyd-warshall algorithm on graphs with negative cycles. Information Processing Letters, 110, 279-281.
- [9] Lozano, L., Medaglia, A. L., (2013). On an exact method for the constrained shortest path problem. Computers & Operations Research, 40, 378-384.
- [10] Santos L., Coutinho-Rodrigues, J., Current J. R., (2007). An improved solution algorithm for the constrained shortest path problem. Transportation Research, 41, 756-71.
- [11] Wang, L., Johnson, E. L., Sokol, J. S., (2005). A multiple pairs shortest path algorithm. Transportation Science, 39, 465-476.
- [12] Carre, B. A., (1971). An algebra for network routing problems. Journal of the Institute of Mathematics and its Applications, 7, 273-294.
- [13] Goto, S., T., Ohtsuki, Yoshimura, T., (1976). Sparse matrix techniques for the shortest path problem, IEEE Transaction on CAS, 23, 752-758.
- [14] Takaoka, T., (2004). A faster algorithm for the all-pairs shortest path problem and its application. Lecture Notes in Computer Science, 3106, 278-289.
- [15] Duin, C. W., (2007). Two fast algorithms for all-pairs shortest paths. Computer & Operations Research, 34, 2827-2839.

- [16] Demetrescu, C., Italiano, G. F., (2007). Algorithmic techniques for maintaining shortest routes in dynamic networks. *Electronic notes in theoretical computer science*, 171, 3-15.
- [17] Chan, T. M., (2008). All-pairs shortest paths with real weights in $O(n^3 / \log n)$ time. *Algorithmica*, 50, 236-243.
- [18] Mohemmed, A. W., Sahoo, N. C., Geok, T. K., (2008). Solving shortest path problem using particle swarm optimization. *Applied soft computing*, 8, 1643-1653.
- [19] Mohamed, C., Bassem, J., Taicir, L., (2010). A genetic algorithms to solve the bicriteria shortest path problem. *Electronic notes in discrete mathematics*, 36, 851-858.
- [20] Yuster, R., (2012). Approximate Shortest Paths in Weighted Graphs. *Journals of Computer and System Science*, 78, 632-637.
- [21] Peng, W., Hu, X., Zhao, F., Su, J., (2012). A fast algorithm to find all-pairs shortest paths in complex networks. *Procedia Computer Science*, 9, 557 – 566.
- [22] Doerr, B., Johannsen, D., Kotzing, T., Neumann, F., Thiele, M., (2013). More effective crossover operators for the all-pairs shortest path problem. *Theoretical Computer Science*, 471, 12-26.
- [23] Huguet, M., Kirchler, D., Parent, P., Calvo, R. W., (2013). Efficient algorithms for the 2- way multi modal shortest path problem. *Electronic notes in discrete mathematics*, 41, 431-437.
- [24] Wang, J., Kaempke, T., (2004). Shortest route computation in distributed systems. *Computers & operations research*, 31, 1621-1633.
- [25] Ambrosino, D., Sciomachen, A., (2014). An algorithmic framework for computing shortest routes in urban multimodal networks with different criteria. *Procedia - Social and Behavioral Sciences*, 108, 139-152.