پردیس علوم
دانشکده ریاضی، آمار و علوم کامپیوتر

مسیرها در گراف های زمانی

نگارنده

ابوالفضل کبیری

استاد راهنما: دکتر مرتضی محمدنوری

پایان‌نامه برای دریافت درجه کارشناسی
در رشته علوم کامپیوتر

مرداد ۱۴۰۲

چکیده

این پژوهش با توجه به رویکرد پارامتری شده، به بررسی پیچیدگی محاسباتی مسیرهای زمانی بی‌استراحت(restless یا بی قرار) پرداخته و الگوریتم‌های موثر برای حل آن در برخی از حالت ها را ارائه کرده است.

سپاس‌گزاری

با سلام و احترام، در اینجا، می‌خواهیم از تمامی افرادی که به ما در این پژوهش کمک کردند، صمیمانه سپاسگزاری کنیم. ابتدا، باید از استاد راهنمای خود، آقای دکتر محمد نوری، که با توجه به دانش و تخصص خود، ما را در این مسیر هدایت کردند، قدردانی کنیم. همچنین، ما باید از تمامی اعضای گروه پژوهشی که با من در این پروژه همکاری کردند، و با پشتکار و تلاش بی‌وقفه خود، به ما در این پژوهش کمک کردند، سپاسگزاری کنیم. همچنین، باید از تمامی مراجع و منابعی که در این پژوهش استفاده شدند که اطلاعات مفیدی را به ما ارائه کردند، قدردانی کنیم. در کل، این پژوهش بدون کمک و همکاری این افراد، ممکن نبود و ما برای این کمک‌ها به آن‌ها بی‌تاب نیستیم.

پیشگفتار

یافتن مسیر بین دو رأس، یکی از بنیادی‌ترین اعمال در نظریه گراف بوده است.
در سال‌های اخیر، مطالعه مسیرها در گراف‌های زمانی(به معنای گراف‌هایی که مجموعه رئوس ثابت دارد اما مجموعه یال‌ها در طول زمان تغییر می‌کند) بیشتر و بیشتر توجه پژوهش‌گران را به خود جلب کرده است.به مسیری یک مسیر زمانی، یا (time-respecting) گوییم اگر از یال‌هایی با مقادیر زمانی صعودی استفاده کند.
ما در اینجا، محدودیت اساسی برای مسیرهای زمانی را مورد بررسی قرار می‌دهیم، که در آن زمان صرف شده در هر رأس نباید از مدت زمان مشخص شده یعنی Δ ، بیشتر باشد.
این محدودیت در مدل‌سازی فرایندهای دنیای واقعی مانند مسیریابی بسته در شبکه‌های ارتباطی و مسیرهای انتقال عفونت بیماری‌ها که بهبودی دائمی را ایجاد می‌کند، به طور طبیعی به وجود می‌آید.
در حالی که یافتن مسیرهای زمانی بدون محدودیت زمان انتظار، به صورت چندجمله‌ای انجام پذیر است، ما نشان می‌دهیم که "نوع بی‌استراحت"(ناآرام یا همان گراف هایی با یال های متغیر) این مشکل حتی در حالت های بسیار محدود دارای پیچیدگی محاسباتی سخت می‌شود.
به عنوان مثال، اگر این مسئله را بر حسب feedback vertex number یا طول مسیر گراف اصلی پارامتریزه شود، $W[1]$ ـسخت است.

پس از این، سؤال اصلی این است که آیا این مشکل در برخی از حالت های طبیعی، قابل حل است یا خیر؟ ما به بررسی چندین پارامتر طبیعی پرداخته‌ایم و الگوریتم FPT را برای سه نوع پارامتر ارائه داده‌ایم:

۱. پارامترهای مربوط به خروجی (در اینجا، حداکثر طول مسیر)

۲. پارامترهای کلاسیکی که به گراف اصلی اعمال می‌شوند (مانند شمارنده برگشتی)

۳. پارامتر جدیدی به نام شمارنده برگشتی زمانی (feedback vertex number) که ویژگی‌های زمانی دقیق‌تر گراف زمانی ورودی را در بر می‌گیرد.

در کل، این پژوهش با توجه به رویکرد پارامتری شده، به بررسی پیچیدگی محاسباتی مسیرهای زمانی بی‌استراحت پرداخته و الگوریتم‌های موثر برای حل آن در برخی از حالت ها را ارائه کرده است.

فهرست مطالب

۱ مقدمه **۱**
۱.۱ مشارکت ما ۳
۲.۱ گزاره ها ۴

۲ پیش نیاز ها و تعاریف **۵**
۱.۲ مشاهدات بیشتر ۸

۳ نتایج سختی برای مسیرهای زمانی بی‌استراحت **۱۰**

۴ یک الگوریتم FPT برای مسیر زمانی بی‌استراحتِ کوتاه **۱۲**
۱.۴ کاهش مسئله به گراف‌های جهت‌دار ۱۳

۵ نظریهٔ پیچیدگی محاسباتی برای گراف اصلی **۱۸**

۶ تعداد رئوس بازخوردی با زمان‌بندی شده(feedback vertex number) **۲۰**

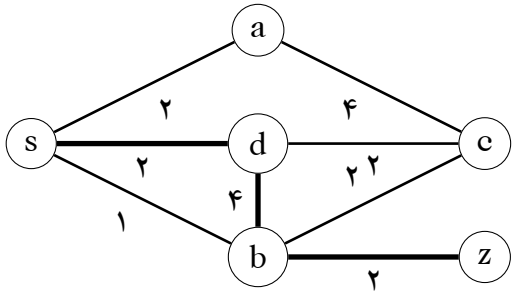
۷ نتیجه گیری **۲۶**

فصل ۱

مقدمه

یکی از روش‌های موفق برای کنترل (یا از بین بردن) شیوع بیماری‌های عفونی، بررسی ارتباطات [۱۸] است.
هرگاه فردی مثبت تشخیص داده شود، هر شخصی که با این فرد تماس فیزیکی داشته است به احتمال زیاد توسط این فرد عفونی شده است و بای در قرنطینه قرار گیرد.
با این حال، شیوع ویروس ممکن است به گونه‌ای سریع باشد که امکان پیگیری دستی آن وجود نداشته باشد، به عنوان مثال، اگر بیماری در یک مرحله پیش از علائم قابل انتقال باشد.
احتمالا هر فرد در هنگام تشخیص بیماری، زنجیره‌های عفونتی را قبلاً ایجاد کرده باشد. بنابراین، سیستم‌های دیجیتال بزرگی باید استفاده شوند که از شبکه‌های نزدیکی فیزیکی بر اساس داده‌های موقعیت و تماس استفاده می کنند [۲۰] ، که این به امکان پیگیری سریع و دقیق تماس‌ها منجر می‌شود.
حال شبکه‌های نزدیکی فیزیکی می‌توانند به عنوان گراف‌های زمانی [۱۳، ۲۸، ۳۰، ۳۶، ۴۰] درک شوند، به عبارت دیگر، گراف‌هایی که مجموعه رئوس (افراد) ثابت بماند، اما مجموعه یال ها (تماس‌های فیزیکی) در طول زمان ممکن است تغییر کند. در این مقاله، ما بر روی یکی از مهم‌ترین مسائل ترکیبیاتی پیش آمده در سناریوی فوق، که یک توسعه از یکی از مسائل بنیادی است، تحلیل پیچیدگی محاسباتی را گسترش می‌دهیم:
با داشتن یک گراف زمانی و دو فرد s و z، آیا زنجیره‌ای از عفونت(مسیری در گراف زمانی) از s به z ممکن است؟ به عبارت دیگر، آیا یک مسیر زمانی از s به z وجود دارد؟ به طور خاص، ما از یک مفهوم دسترسی یا همسایگی استفاده می‌کنیم که مدل استاندارد ۳ حالته SIR (قابل انعطاف - آلوده - بهبود یافته) را در بر می‌گیرد.
در این مقاله، با توسعه مفهوم دسترسی در گراف‌های زمانی و استفاده از مدل SIR، مسئله پیگیری زنجیره‌های عفونتی در شبکه‌های نزدیکی فیزیکی مورد بررسی قرار گرفته است. با استفاده از این روش، پیگیری تماس‌های بیماران به صورت سریع و دقیق انجام می‌شود و نیازی به عملیات قرنطینه عمومی نیست که می‌تواند تاثیرات منفی بر جامعه داشته باشد.

در گراف‌های زمانی، مفاهیم اساسی مسیر و همسایگی با توجه به مقوله زمان (time-respecting) [۳۴] تعریف می‌شوند:
یک مسیر زمانی (یا "walk") سخت، یک مسیر است که از یال هایی با قدم‌های زمانی بدون کاهش (افزایش) استفاده می‌کند. برای نمایش زنجیره‌های عفونت در مدل SIR، زمان انتظار یا توقف در هر راس وسط را به مدت مشخصی محدود می‌کنیم. این مسیرها را مسیرهای زمانی بیقرار می‌نامیم. آنها مدلی برای مسیرهای انتقال عفونت بیماری‌ها هستند که در صورت بهبود، ایمنی همیشگی را می‌دهند [۲۹] : فرد آلوده می‌تواند تا بهبود (با زمان انتظار محدود) بیماری را منتقل کند و بعد از آن دیگر هیچوقت دوباره دچار عفونت نمی‌شود (یعنی هر راس را فقط یک بار بازدید می‌کند).
یک مثال طبیعی دیگر از مسیرهای زمانی بیقرار، شبکه‌های مبتنی بر تحمل تأخیر بین موجودیت‌های متحرک است، که در آن مسیریابی یک بسته به مدت محدودی در گره‌های وسطی با زمان و مکان انجام می‌شود.
در ادامه، ما با ارائه یک مثال، تنظیم مسئله خود را به صورت غیررسمی توصیف می‌کنیم. در شکل ۱.۱ ، ما گراف زمانی نشان داده شده، رئوس s و z و محدودیت $\Delta = 2$ را داریم.
ما باید تصمیم بگیریم که آیا یک مسیر زمانی بیقرار از s به z وجود دارد یا خیر، به این معنی که یک مسیر وجود داشته باشد که در هر قدم، حداکثر یک بار به هر راس سر بزند و در بین قدم‌های پشت سر هم، حداکثر به مدت Δ واحد زمان توقف کند. در اینجا، (s, d, b, z) یک راه حل قابل قبول است، اما (s, b, z) به دلیل اینکه زمان انتظار در b از Δ بیشتر است، قابل قبول نیست. همچنین مسیر (s, b, c, d, b, z) یک راه‌حل معتبر نیست زیرا دو بار به راس b سر می‌زند و (s, a, c, d, b, z) نیز یک راه حل قابل قبول است.



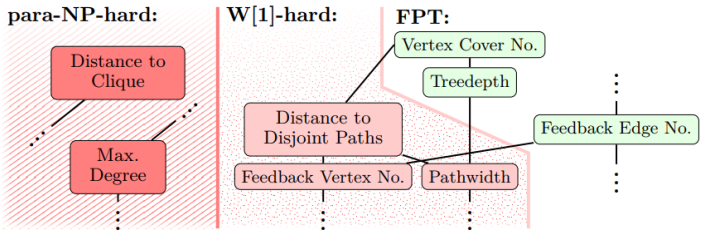
شکل ۱.۱: مثالی از یک گراف زمانی با یالهایی با مقادیر زمانی که در آن لبه‌های پررنگ، یک مسیر بیقرار زمانی (s, z) را نشان می‌دهند.

نشان داده شده است که محدودیت‌های زمان انتظار، تأثیر شگفت‌انگیزی روی قابلیت بیان گراف زمانی دارند، هنگامی که چنین گرافی را به عنوان یک ماشین محاسبه در نظر می‌گیریم و مسیرهای زمانی را به عنوان کلمات.
یک نوع متفاوت از مسیرهای زمانی بیقرار را مورد بررسی قرار دادیم که چندین walk به همان راس

مجاز است. نشان داده شده است که که مسیرهای چنینی می‌توانند به صورت چندجمله‌ای محاسبه شوند. [۲، ۱۹، ۲۷]
در حالت های مربوط به محدودیت های زمانی، مسائل مربوط به مسیرها بررسی های بسیاری شده‌اند و طبیعت مسیرهای زمانی باعث افزایش پیچیدگی محاسباتی بسیاری از آنها (نسبت به نسخه‌های استاتیک آنها) می‌شود.
در تنظیم زمانی، دسترسی به یک راس رابطه معادلی در بین رئوس نیست که باعث پیچیدگی بیشتر برخی مسائل می‌شود.
به عنوان مثال، پیدا کردن یک جزء زمانی متصل بیشینه(یک زیرگراف از گراف زمانی است که در آن هر دو راس در یک زمان قابل دسترسی هستند و همچنین بیشترین تعداد رئوس را هم دارد) از درجه NP-hard است [۱۰] .
همچنین، در یک گراف زمانی، درخت پوشا وجود ندارد. در واقع، حتی وجود زیرگراف با تعداد یالهای $o(n^2)$ که اتصال زمانی را تضمین می‌کنند هم تضمین نمی‌شود [۵] ، مگر آنکه گراف پایه کامل باشد [۱۴] و محاسبه spanner با حداقل جمعیت APX-hard است [۳، ۳۹].
مثال دیگر، مسئله تصمیم این است که آیا k مسیر زمانی جداگانه بین دو راس داده شده وجود دارد یا خیر. در مقاله‌ای، کمپ [۳۴] نشان داده که این مسئله که نسخه کلاسیک آن چندجمله‌ای است، NP-hard می‌شود. آنها همچنین مسئله مرتبط پیدا کردن جداکننده‌های زمانی را بررسی کردند که نیز NP-hard است [۲۱، ۳۴، ۴۹].

۱.۱ مشارکت ما

مشارکت ما در این مقاله شامل معرفی مسئله «مسیر زمانی بی‌قرار» است. به منظور درک بهتر پیچیدگی محاسباتی این مسئله، توجه شما را به پیچیدگی پارامتری آن معطوف می‌کنیم. علارغم تفاوت زیاد دو مسیر زمانی بی‌قرار و مسیر زمانی غیر بی‌قرار، نشان می‌دهیم که این مسئله در صورتی که بوسیله فاصله حذف راسی تا مسیرهای جدا از گرافِ پایه پارامتری شود NP-hard و W[1]-hard است(بخش ۳). این به معنای این است که مسئله ما در صورتی که گراف پایه یک جنگل است در زمان چندجمله‌ای قابل حل است. ما پارامترهایی با سه نوع مختلف را بررسی می‌کنیم. اولاً، نشان می‌دهیم که مسئله برای تعدادی از hope های مسیر زمانی قابل حل FPT است (بخش ۴). در ادامه، نشان می‌دهیم که مسئله زمانی که با تعداد یالهای بازخوردی گرافِ پایه پارامتری شود قابل حل است (بخش ۵). علاوه بر این، نشان می‌دهیم که در پارامترهایی که قبلاً به نتایج FPT منجر شده‌اند، مسئله به احتمال زیاد هسته چندجمله‌ای ندارد. نتایج ما یک توصیف دقیق از مرز قابلیت حل محاسبه مسیرهای زمانی بی‌قرار برای پارامترهای گراف پایه را همانطور که در نزدیکی پارامترهای مربوطه در شکل ۲.۱ نشان داده شده است ارائه می‌دهد.
سپس، بیشتر از پارامترهای مرتبط با خروجی و گراف پایه، نسخهِ زمانیِ جدیدی از تعداد راس



شکل ۲.۱: نشان دهنده بخش مربوطه از سلسله مراتب پارامترهای کلاسیک گراف پایه است (مرجع: [۴۵])

بازخوردی کلاسیک به نام تعداد راس بازخوردی زمانی(feedback vertex number) را تعریف می‌کنیم)به طور خلاصه، این تعداد، تعداد راس‌هایی را شمرده که باید از گراف زمانی حذف شوند تا گراف پایه آن بدون دور باشد. نشان می‌دهیم که پیدا کردن مسیرهای زمانی بی‌قرار با پارامتری شدن این پارامتر FPT است (بخش ۶). ما معتقدیم که این نتیجه، نسبت به نتایج سختی ما، یک تحول جالب است. به دلیل محدودیت فضا، بیشتر اثباتها (آنهایی که با ★نمایش داده شده اند) به داخل مراجع ارجاع داده شده است.

۲.۱ گزاره ها

در این پژوهش، پیچیدگی محاسباتی مسیرهای زمانی بی قرار بررسی شده و الگوریتم هایی برای حل آن در برخی موارد ارائه شده است.

هدف اصلی، بررسی پیچیدگی محاسباتی مسئله مسیر زمانی بی قرار و ارائه الگوریتم های کارآمد برای حل آن در برخی موارد است. فرضیه اصلی این است که این مسئله در مواردی قابلیت محاسباتی با پارامتر ثابت دارد.

متغیرها و مفاهیم اصلی شامل گراف زمانی، مسیر زمانی بی قرار، پیچیدگی محاسباتی، الگوریتم های کارآمد و پارامترهای ساختاری گراف است.

این پژوهش به بررسی نظری پیچیدگی محاسباتی مسئله مسیر زمانی بی قرار می پردازد و الگوریتم های ارائه شده، الگوریتم هایی نظری هستند.

روش پژوهش، تحلیل نظری پیچیدگی الگوریتم ها است و داده های مورد استفاده، مفاهیم نظری در حوزه نظریه پیچیدگی محاسباتی و الگوریتم ها است.

فصل ۲

پیش نیاز ها و تعاریف

در اینجا، ما به طور رسمی مفاهیم مهم مرتبط با گراف‌ها و مسیرهای زمانی را معرفی کرده و تعریف رسمی مسئله مسیر زمانی بی‌قرار (کوتاه) (Short Restless Temporal) را ارائه می‌دهیم. یک بازه، مجموعه مرتب $[a,b] := \{n \mid n \in \mathbb{N} \wedge a \leq n \leq b\}$ است، که در آن a و b اعداد طبیعی هستند و $a := [1,a]$ است. ما از نمادگذاری استاندارد و اصطلاحات تئوری گراف [۱۶] و تئوری پیچیدگی پارامتری [۱۵، ۱۷] استفاده می‌کنیم.

گراف‌های زمانی: یک *گراف زمانی* ساده و بدون جهت است که در آن برای تمام $i \in [\ell]$ داریم $\mathcal{G} = (V, E_1, \ldots, E_\ell)$ با $E_i \subseteq \binom{V}{2}$. ما $\ell := \ell(\mathcal{G})$ را *طول عمر* $\mathcal{G}$ می‌نامیم. مانند گراف‌های استاتیک، ما فرض می‌کنیم که تمامی گراف‌های زمانی در این مقاله بدون جهت و ساده هستند. ما گراف $G_i(\mathcal{G}) = (V, E_i(\mathcal{G}))$ را **لایه‌ی** i از گراف $\mathcal{G}$ می‌نامیم که در آن اگر $E_i(\mathcal{G}) := E_i$ باشد. واگر $E_i = \emptyset$، آنگاه G_i یک **لایه‌ی ساده** است. ما G_i و G_{i+1} را لایه‌های پشت سر هم می‌نامیم و i را یک *مرحله زمانی* می‌نامیم. اگر یک یال e در زمان i وجود داشته باشد، به عبارت دیگر، $e \in E_i$ باشد، ما می‌گوییم e *برچسب زمانی* i را دارد. همچنین بصورت $V(\mathcal{G}) := V$ نشان میدهیم. *گراف پایه* $G_\downarrow(\mathcal{G})$ از $\mathcal{G}$ به صورت $G_\downarrow(\mathcal{G}) := (V, \bigcup_{i=1}^{\ell(\mathcal{G})} E_i(\mathcal{G}))$ تعریف می‌شود. برای بهبود خوانایی، $(\mathcal{G})$ را از نمادگذاری‌های معرفی شده حذف می‌کنیم. برای هر $v \in V$ و هر مرحله زمانی $t \in [\ell]$، v در زمان t را با جفت (v,t) نشان می‌دهیم. برای هر $t \in [\ell]$، و هر $e \in E_t$ ما جفت (e,t) را یک *یال زمانی* می‌نامیم. مثلا برای یک یال زمانی $(\{v,w\},t)$ ، ما گره‌ها (v,t) و (w,t) را *نقاط پایانی* آن می‌نامیم. فرض می‌کنیم که اندازه $\mathcal{G}$ برابر با $|\mathcal{G}| := |V| + \sum_{i=1}^{\ell} |E_i|$ است، در حالت کلی فرض نمی‌کنیم که نمایش فشرده‌ای از گراف‌های زمانی را داریم. در نهایت اندازه $|V|$ را با n نمایش میدهیم.

پویش زمانی یک - (s,z) به طول k از گره شروع $s = v_0$ تا گره پایانی $z = v_k$ در یک گراف زمانیِ $\mathcal{G} = (V, (E_i)_{i \in [\ell]})$ یک دنباله از سه‌تایی‌هایی است که آن‌ها را *گذرگاه‌ها*

(transitions) می‌نامیم، به شکل $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ هستند. طوری که برای همه $i \in [k]$ ها داریم $\{v_{i-1}, v_i\} \in E_{t_i}$ و برای همه‌ی $i \in [k-1]$ ها داریم $t_i \leq t_{i+1}$. علاوه بر این، اگر برای همه‌ی i، j بصورت $i, j \in \{0, \ldots, k\}$ با $i \neq j$ داشته باشیم $v_i \neq v_j$ ، آنگاه P را یک *مسیر زمانی* (s, z) به طول k می‌نامیم. مجموعه‌ی رئوسی که توسط P بازدید شده‌اند، با $V(P) = \{v_0, v_1, \ldots, v_k\}$ نشان میدهیم. یک مسیر زمانی بی‌قرار اجازه ندارد که در یک گره مقدار دلخواهی از زمان منتظر بماند، بلکه باید هر گره‌ای که بازدید می‌کند، در حداکثر Δ واحد زمانی بعدی ترک کند.

تعریف ۱.۲. *یک مسیر زمانی (پویش زمانی) $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ با طول k ، اگر برای همه‌ی $i \in [k-1]$ داشته باشیم $t_i \leq t_{i+1} \leq t_i + \Delta$ ، یک مسیر بی‌قرار-Δ (مسیری با حداکثر مکث Δ) است.*

با داشتن این تعریف، ما آماده‌ایم تا مسئله‌ی اصلی این مقاله را تعریف کنیم.

مسیر زمانی بی‌قرار
ورودی: *یک گراف زمانی $\mathcal{G} = (V, (E_i)_{i \in [\ell]})$ ، دو گره متمایز $s, z \in V$ و یک عدد صحیح $\Delta \leq \ell$.*
سوال: *آیا در $\mathcal{G}$ یک مسیر زمانی بی‌قرار (s, z) با حداکثر Δ وجود دارد؟*

توجه کنید که زمان انتظار در گره مبدا s، نادیده گرفته می‌شود. این بدون کاستی از کلیت است، زیرا می‌توان یک گره مبدا درجه یک کمکی را که فقط در لایه اول با s همسایه است، اضافه کرد. ما همچنین یک نوع دیگر را در نظر می‌گیریم، که می‌خواهیم مسیرهای بی‌قرار با حداکثر طول مشخصی را پیدا کنیم. در مسئله مسیر کوتاه بی‌قرار زمان(Short Restless Temporal Path)، یک عدد صحیح اضافی، داده می‌شود و سوال این است که آیا در $\mathcal{G}$ یک مسیر زمانی بی‌قرار با طول حداکثر k از s به z وجود دارد؟ توجه کنید که مسیر زمانی بی‌قرار، حالت ویژه‌ای از مسیر زمانی کوتاه بی‌قرار برای $k = |V| - 1$ است و هر دو مسئله در NP هستند.

پیچیدگی پارامتری . ما از نمادگذاری و اصطلاحات استاندارد نظریه پیچیدگی پارامتری [۱۵] استفاده می‌کنیم و در اینجا یک مرور خلاصه از مهمترین مفاهیمی که در این مقاله استفاده می‌شود، ارائه می‌دهیم. یک *مسئله‌ی پارامتری* ، یک زبان $L \subseteq \Sigma^* \times \mathbb{N}$ است، که در آن Σ یک الفبای محدود است. ما دومین مولفه را *پارامتر* مسئله می‌نامیم. یک مسئله‌ی پارامتری،(با شرط ثابت بودن پارامتر) *قابل حل است* (در کلاس پیچیدگی FPT)، اگر الگوریتمی وجود داشته باشد که هر نمونه (I, r) را در زمان $f(r) \cdot |I|^{O(1)}$ حل کند. یک مسئله‌ی پارامتری قابل تبدیل به *هسته چندجمله‌ای* است، اگر یک الگوریتم چندجمله‌ای وجود داشته باشد که هر نمونه (I, r) را به یک نمونه (I', r) تبدیل کند، به طوری که (I, r) متعلق به L است اگر و تنها اگر (I', r)

متعلق به L باشد و $|(I', r')| \in r^{O(1)}$. اگر یک مسئله‌ی پارامتری سخت برای کلاس پیچیدگی پارامتری [۱]W باشد، آنگاه (احتمالاً) در FPT نیست. کلاس‌های پیچیدگی [۱]W زیر تحت کاهش پارامتری هستند، که ممکن است در زمان FPT اجرا شود و علاوه بر این، پارامتر جدید را به یک مقدار تنها بستگی به پارامتر قدیمی بدهد.

مشاهدات پایه . اگر یک مسیر زمانی بی‌قرارِ $P = ((v_{i-1}, v_i, t_i))_{i=1}^{k}$ در یک گراف زمانی $\mathcal{G}$ وجود داشته باشد، آنگاه $P' = (\{v_0, v_1\}, \ldots, \{v_{k-1}, v_k\})$ یک مسیر (s, z) در گراف پایه $\mathcal{G}_{\downarrow}$ است. به عبارت دیگر، راه دیگری وجود ندارد، اما برای هر مسیر (s, z) -path در $\mathcal{G}_{\downarrow}$ می‌توانیم به صورت خطی تصمیم بگیریم که آیا این یک مسیر $s - z$ تشکیل می‌دهد یا خیر. به عنوان یک نتیجه، ما می‌توانیم برای هر گراف زمانی در زمان خطی درحالتی گراف پایه یک جنگل باشد تصمیم بگیریم که در آن یک مسیر (s, z) یکتا در گراف پایه وجود دارد.

لم ۲.۲. *فرض کنید $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ یک گراف زمانی باشد که گراف پایه $\mathcal{G}_{\downarrow}$ یک مسیر (s, z) طوریکه s و z متعلق به V است. در این صورت، یک الگوریتم وجود دارد که در زمان $O(|\mathcal{G}|)$ مجموعه $A = \{t|$یک مسیر زمانی بی‌قرار با زمان ورودی tوجود دارد$\}$ را محاسبه می‌کند.*

اثبات. فرض کنید $V(G_{\downarrow}) = \{s = v_0, \ldots, v_n = z\}$ مجموعه رئوس و

$$E(G_{\downarrow}) = \{e_1 = \{v_0, v_1\}, \ldots, e_n = \{v_{n-1}, v_n\}\}$$

مجموعه یال‌های مسیر پایه باشد. همچنین، ما L_i را به عنوان مجموعه لایه‌های $\mathcal{G}$ تعریف می‌کنیم اگر در آن یال $e_i \in E(G_{\downarrow})$ وجود دارد، یعنی، $L_i := \{t|e_i \in E_t\}$. در ادامه، یک برنامه پویا روی مسیر اجرا می‌کنیم. برای هر گره v_i، ما ورودی جدول $T[v_i]$ را محاسبه می‌کنیم که به عنوان مجموعه‌ای از تمام لایه‌های t تعریف می‌شود که یک مسیر زمانی (s, v_i)ـبی‌قرار با زمان ورودی t وجود دارد. داریم $T[v_1] = L_1$. حال می‌توانیم ورودی‌های جدول را به صورت پشت سر هم به شکل زیر محاسبه کنیم:

$$T[v_i] = \{t \in L_i | t' \in T[v_{i-1}] \text{وجود دارد}, 0 \leq t - t' \leq \Delta\}$$

برای ورودی جدول $T[v_i]$، برای هر لایه بررسی می‌کنیم که آیا یک مسیر زمانی (s, v_{i-1}) ـبی‌قرار وجود دارد که در لایهِ $t' \in T[v_{i-1}]$ وارد شده و می‌توانیم مسیر را به گره v_i در لایه t بدون تجاوز زمان انتظار حداکثر Δ ادامه دهیم، یعنی: $0 \leq t - t' \leq \Delta$. واضح است که $T[v_i]$ شامل تمام لایه‌های t است که یک مسیر زمانی (s, v_i)ـبی‌قرار با زمان ورودی t دارد. پس از محاسبه آخرین ورودی یعنی $T[v_n]$، این ورودی شامل مجموعه A از تمام لایه‌های t است که یک مسیر زمانی (s, z)ـبی‌قرار با زمان ورودی t وجود دارد.
برای محاسبه ورودی جدول $T[v_i]$ در زمان خطی، ما لیست‌های مرتب شده لایه‌ها برای L_i و $T[v_{i-1}]$ را به صورت صعودی نیاز داریم. لیست‌های مرتب شده L_i از لایه‌ها را می‌توان در زمان $O(|\mathcal{G}|)$ محاسبه کرد: برای هر t متعلق به $[\ell]$، ما برای هر ورودی $e_i \in E_t$ عملیات را تکرار کرده

و t را به L_i اضافه می‌کنیم. حال فرض کنید L_i و $T[v_{i-1}]$ لیست لایه‌هایی باشند که به صورت صعودی مرتب شده‌اند، پس می‌توانیم ورودی جدول $T[v_i]$ را در زمان $O(|T[v_{i-1}]| + |L_i|)$ محاسبه کنیم.
الگوریتم:
ورودی جدول $T[v_i]$ را خالی تعریف می‌کنیم. اگر t اولین عنصر در L_i و t' اولین عنصر در $T[v_{i-1}]$ باشد داریم:

۱. اگر $t < t'$ باشد: t را با لایه بعدی در L_i جایگزین کنید و تکرار کنید.

۲. اگر $t - t' \leq$، باشد: t را به $T[v_i]$ اضافه کنید و t را با لایه بعدی در L_i جایگزین کنید و تکرار کنید.

۳. در غیر این صورت، t_0 را با لایه بعدی در $T[v_{i-1}]$ جایگزین کنید و تکرار کنید.

این کار تا زمانی ادامه می‌یابد که تمام عناصر در لیست‌ها پردازش شود.
در لیست نهایی چون $T[v_1](= L_1)$ مرتب است پس $T[v_i]$ دوباره مرتب می‌شود. هنگام محاسبه $T[v_i]$، می‌توانیم فرض کنیم که $T[v_{i-1}]$ به عنوان یک لیست مرتب شده از لایه‌ها داده شده است. بنابراین، ما می‌توانیم هر ورودی جدول $T[v_i]$ را به طور کارآمد در زمان $O(|T[v_{i-1}]| + |L_i|)$ محاسبه کنیم. همچنین داریم $|T[v_i]| \leq |L_i|$ و $\Sigma_{i=1}^{n}|L_i| = \Sigma_{i=1}^{\ell}|E_i|$. بنابراین، برنامه پویا در زمان $O(|\mathcal{G}|)$ اجرا می‌شود. □

۱.۲ مشاهدات بیشتر

اینجا مقاله‌ای در مورد مسئلهShort Restless Temporal Path و پیچیدگی محاسباتی آن ذکر شده است. در این بخش، چند مشاهدهٔ پایه در مورد گراف‌ها بیان شده است.

مشاهده ۳.۲. *با داشتن یک نمونهٔ مسئلهٔ مسیر زمانی نامستقر کوتاه، می‌توان نمونهٔ دیگری از همین مسئله با مقدار ثابت بزرگتر را در زمان خطی ساخت. به این صورت که با دریافت یک نمونهٔ $I = (\mathcal{G}, s, z, k, \Delta+1)$ از مسئله مسیر زمانی نامستقر کوتاه میتوان یک I' با مقدار ثابت بزرگتر ساخت. با این کار، می‌توان نتیجه گرفت که اگر I مثبت باشد، I' نیز مثبت است و بالعکس. این مشاهدات به ما کمک می‌کنند تا پیچیدگی محاسباتی مسیر زمانی نامستقر کوتاه را بهتر درک کنیم و با استفاده از آن‌ها، کاهش سختی برای مقادیر ثابت کوچک ایجاد کنیم.*

اثبات. اگر یک مسیر زمانی نامستقر کوتاه با طول Δ در گراف زمانی $\mathcal{G}$ وجود داشته باشد، آنگاه یک مسیر زمانی نامستقر کوتاه با طول $\Delta + 1$ در گراف زمانی $\mathcal{G}'$ وجود دارد و بالعکس. بنابراین،

اگر مسئله اصلی برای یک مقدار مشخص NP-complete باشد، آنگاه مسئله برای تمام مقادیر بزرگتر از آن نیز NP-complete خواهد بود. اما برای برخی از مقادیر خاص ، می‌توان مسیر زمانی نامستقر کوتاه را با زمان چندجمله‌ای حل کرد. به عنوان مثال، اگر $\Delta = 0$ باشد، مسیر بین s و z باید کاملاً در یک لایه قرار داشته باشد. بنابراین، مسئله معادل با آزمایش این است که آیا حداقل یکی از لایه‌های $\mathcal{G}_i$ شامل یک مسیر ثابت بین s و z است یا خیر. □

به عبارت دیگر، اگر مقدار بزرگتر مساوی Δ باشد، می‌توان مسئله مسیر زمانی نامستقر کوتاه را با زمان چندجمله‌ای بوسیله یکی از سه الگوریتم مربوط به محاسبهٔ مسیرهای زمانی بدون محدودیت زمانی حل کرد.

مشاهده ۴.۲. *مسیر زمانی نامستقر کوتاه در نمونه‌های $(\mathcal{G}, s, z, \Delta)$ اگر که $\Delta = 0$ یا $\Delta \geq \ell$ باشد در زمان چندجمله‌ای قابل حل است.*

اثبات. اگر Δ برابر با صفر باش به معنای این است که کل مسیر بین s و z باید در یک لایه واحد تحقق پیدا کند. بنابراین، مسئله معادل با آزمایش این است که آیا حداقل یکی از لایه‌های $\mathcal{G}_i$ شامل یک مسیر ثابت بین s و z است یا خیر.

اگر Δ بزرگترمساوی با ℓ هم باشد، محاسبهٔ مسیرهای زمانی بدون محدودیت زمانی برای سه معیار ممکن توسط Bui-Xuan ، Ferreira و Jarry [۱۱] حل شده است. □

فصل ۳

نتایج سختی برای مسیرهای زمانی بی‌استراحت

در این بخش، تحلیل دقیقی از سختی محاسباتی مسیر زمانی نامستقر کوتاه ارائه می‌دهیم. ما با نشان دادن Np-hardness برای چند لایه در مسیر زمانی نامستقر کوتاه آغاز می‌کنیم.

قضیه ۱.۳. (★) *مسیر زمانی نامستقرِ (Restless temporal) کوتاه برای همه $\ell \geq \Delta + 2$ و $\Delta \geq 1$ ها حتی اگر هر یال دارای یک برچسب زمانی باشد ، NP-Complete است.*

استقرا (reduction) استفاده شده در اثبات قضیه ۱.۳ با فرض زمان نمایی (ETH) حد پایین زمان اجرایی را [۳۱، ۳۲] نیز ارائه می‌دهد.

نتیجه گیری ۲.۳. *. مسیر زمانی نامستقر کوتاه(restless temporal path) هیچ الگوریتمی با زمان اجرای $f(\ell)^{o(|\mathcal{G}|)}$ برای هر تابع محاسبه‌پذیر f ندارد .*

اثبات. ابتدا توجه کنید که هر فرمول ۳-SAT با m مقدار قابلیت تعمیم به فرمول Exact (۳و ۴)-SAT با $O(m)$ مقدار قابلیت تعمیم را دارد [۴۶]. کاهشِ ارائه شده در اثبات قضیه ۱.۳ نمونه‌ای از مسیر زمانی نامستقر کوتاه با یک گراف زمانی به اندازه $|\mathcal{G}| = O(m)$ و $\ell = 3$ را ارائه می‌کند. بنابراین اگر یک الگوریتم برای مسیر زمانی نامستقر کوتاه با زمان اجرای $f(\ell)^{o(|\mathcal{G}|)}$ برای برخی تابع محاسبه‌پذیر f وجود داشته باشد، به معنای وجود یک الگوریتم با زمان اجرای $2^{o(m)}$ برای ۳-SAT است. که این با فرضیه زمان نمایی در تناقض است [۳۱، ۳۲]. □

علاوه بر این، کاهشِ پشت قضیه ۱.۳ به گونه‌ای تغییر داده می‌شود که می‌توان نشان داد مسیر زمانی نامستقر کوتاه NP-hard است، حتی اگر گراف پایه حداکثر درجه ثابت داشته باشد.

نتیجه گیری ۳.۳. *مسیر زمانی نامستقر کوتاه حتی اگر گراف پایه تمام یال‌هایش به جز یک یال یا حداکثر درجه شش باشد، NP-hard است.*

اثبات. این نتیجه گیری به طور مستقیم از نتیجه گیری ۲.۳ و اثبات قضیه ۱.۳ با توجه به توجه به تغییرات کوچک در کاهش استفاده شده در اثبات قضیه ۱.۳ نتیجه می‌شود. در واقع، ما می‌توانیم یک گراف پایه را با حداکثر درجه شش یا تمام یال‌ها به جز یک یال از یک مثال NP-hard موجود ساخته و سپس اثبات کنیم که این مثال نیز NP-hard است. اثبات. اینکه مسیر زمانی نامستقر کوتاه NP-hard است، حتی اگر گراف پایه حداکثر درجه شش باشد، به طور مستقیم از ساختار استفاده شده در اثبات قضیه ۱.۳ پیروی می‌کند. برای نشان دادن اینکه مسیر زمانی نامستقر کوتاهNP-hard است، از مسیر زمانی نامستقر کوتاه reduction میزنیم. ما یک نمونه $I = (\mathcal{G} = (V, (E_i)_{i\in[\ell]}), s, z, \Delta)$ از مسیر زمانی نامستقر کوتاه با $\ell = 3$ و یک نمونه $I' := (\mathcal{G} = (V, E'_1, E'_2, E'_3, E'_4, E'_5), s, z, \Delta)$ از مسیر زمانی نامستقر کوتاه ساخته‌ایم، که در آن $E'_1 = \binom{V\setminus\{s\}}{2}$ ، $E'_2 := E'_1$ ، $E'_3 := E'_2$ ، $E'_4 := E'_3$ و $E'_5 = \binom{V\setminus\{z\}}{2}$ است. توجه کنید که هیچ یک از یال‌های $E_1 \cup E_5$ نمی‌تواند در مسیر زمانی نامستقر (s, z) restless- استفاده شود. بنابراین، I یک نمونه ی مثبت است اگر و تنها اگر I' یک نمونه ی مثبت باشد. علاوه بر این، $E_1 \cup E_5$ شامل تمام یال‌های ممکن به جز $\{s, z\}$ است. □

قضیه ۴.۳. (★) *در مسیر زمانی نامستقر کوتاه که با پارامترِ فاصله تا مسیرهای جدا از هم از گراف پایه پارامتری شده است، به ازای $\Delta \geq 1$ حتی اگر هر یال دارای یک برچسب زمانی باشد، $W[1]$-hard است.*

فصل ۴

یک الگوریتم FPT برای مسیر زمانی بی‌استراحتِ کوتاه

در این بخش، به بحث درباره چگونگی پیدا کردن مسیرهای زمانی بی‌قرار کوتاه می‌پردازیم. میدانیم در مسیر زمانی بی‌قرار کوتاه، یک عدد صحیح k به عنوان ورودی اضافه شده است و پرسیده می‌شود که آیا یک مسیر زمانی (s, z) وجود دارد که حداکثر k یال زمانی را استفاده کند. با توجه به قضیه ۱.۳ این مسئله NP-hard است. توجه کنید که در سناریوی پیگیری تماس از ابتدا، می‌توانیم انتظار داشته باشیم k کوچک و گراف زمانی بزرگ باشد.

قضیه ۱.۴. *ویژگی های مسیر بی‌استراحت زمانی کوتاه:*

۱. *با خطای یک طرفه ی ثابت در زمان* $2^k \cdot |\mathcal{G}|^{O(1)}$ *قابل حل است.*

۲. *به صورت قطعی در زمان* $2^{O(k)} \cdot |\mathcal{G}|\Delta$ *قابل حل است.*

توجه کنید که می‌توانیم مسیر بی‌استراحت زمانی کوتاه را به گونه‌ای حل کنیم که زمان اجرای آن به مدت عمر گراف زمانی وابسته نباشد. برای اثبات قضیه ۱.۴ ، ابتدا مسئله را به یک مسئله مشخص مسیر در گراف‌های جهت‌دار کاهش می‌دهیم. سپس، از ابزارهای جبری معروف برای شناسایی چندجمله‌ای‌های خطی استفاده می‌کنیم. در اینجا، قضیه ۱.۴ (قسمت ۱) بر اساس ویلیامز [۴۷] است. برای به دست آوردن یک الگوریتم قطعی با زمان اجرای تقریباً خطی در $|\mathcal{G}|$، رویکرد متفاوتی بر اساس مجموعه‌های نماینده [۲۲] ارائه می‌دهیم که در نتیجه قضیه ۱.۴ (قسمت ۲) حاصل می‌شود.

۱.۴ کاهش مسئله به گراف‌های جهت‌دار

ما یک گسترش به نام Δ- (s,z) ـگسترش برای دو راس s و z از یک گراف زمانی با زمان‌های انتظار معرفی می‌کنیم. به عبارت دیگر، یک نسخه گسترده شده ی زمانی از گراف زمانی که سوالات مربوط به همسایگی یا دسترسی را به گراف‌های جهت‌دار کاهش می‌دهد. در حالی که رویکردهای مشابه چندین بار اعمال شده‌اند [۲، ۸، ۳۹، ۴۸، ۴۹] ، دانش ما نشان می‌دهد که این بار اولین بار است که زمان انتظار در نظر گرفته می‌شود.

به طور خلاصه، گسترش Δ-(s,z) برای هر راس v حداکثر $v^1,\dots,v^\ell$ کپی دارد و اگر یک مسیر (s,z) در حرکت خود v^i را دید، به این معنی است که یک walk بدون استراحت زمانی (s,z) که مربوط به آن زمان i باشد، در زمان i گره v را بازدید می‌کند.

تعریف ۲.۴. *(گسترش (s,z)-). اگر گراف زمانی $\mathcal{G}=(V,(E_i)_{i\in[\ell]})$ را با دو راس مجزای $s,z\in V$ به گونه‌ای که به ازای همه $t\in[\ell]$ ها $\{s,z\}\notin E_t$ است داشته باشید. فرض کنید $\Delta\le\ell$. گسترش‌ـ(s,z) گراف $\mathcal{G}$، گراف جهت‌دار $D=(V',E')$ است که دارای موارد زیر است:*

۱. $V' \coloneqq \{s,z\}\cup\{v^t | v\in e, e\in E_t, v\notin\{s,z\}\}$

۲. $E_s \coloneqq \{(s,v^t) | \{s,v\}\in E_t\}$

۳. $E_z \coloneqq \{(v^i,z) | v^i\in V', \{v,z\}\in E_t, 0\le t-i\le\Delta\}$

۴. $E' \coloneqq E_s\cup E_z\cup\{(v^i,w^t) | v^i\in V'\setminus\{s,z\}, \{v,w\}\in E_t, 0\le t-i\le\Delta\}$

علاوه بر این، $V'(s)\coloneqq s$، $V'(z)\coloneqq z$، و $V'(v)\coloneqq\{v^t\in V' | t\in[\ell]\}$ را برای همه $v\in V\setminus\{s,z\}$ تعریف می‌کنیم.

در ادامه، نشان می‌دهیم که یک گسترش از یک گراف زمانی می‌تواند به صورت کارآمد محاسبه شود.

لم ۳.۴. *با توجه به یک گراف زمانی $\mathcal{G}=(V,(E_i)_{i\in[\ell]})$ و دو راس مجزا $s,z\in V$ و $\Delta\le\ell$، می‌توانیم گسترش (s,z)- آن را با $|V(D)|\in O(|\mathcal{G}|)$ در زمان $O(|\mathcal{G}|\cdot\Delta)$ محاسبه کنیم.*

اثبات.

اثبات. ابتدا $V'\coloneqq\{s,z\}$ وE' خالی قرار داده می‌شود. به منظور انجام این کار به صورت کارآمد، برای هر راس $v\in V$ ، لیست مرتب شده‌ای L_v را حفظ می‌کنیم به گونه‌ای که $t\in L_v$ در صورت و تنها در صورتی است که $v^t\in V'$. فرض می‌کنیم که $|V|\le\Sigma_{i=1}^{\ell}|E_i|$ باتوجه به اینکه رئوسی که در هر لایه وجود دارند ایزوله هستند برای گسترش ـ(s,z) مهم نیستند و می‌توانند به صورت خطی حذف شوند. ما به شرح زیر ادامه می‌دهیم. برای هر $t\in\{1,\dots,\ell\}$ (به ترتیب صعودی)، بر روی E_t تکرار می‌کنیم. برای هر $\{v,w\}\in E_t$ سه حالت را متمایز می‌کنیم.

۱. $(w = s)$: ما v^t را به V' اضافه می‌کنیم و (s, v^t) را به E' اضافه می‌کنیم و t را به L_v اضافه می‌کنیم. این کار در زمان ثابت انجام می‌شود.

۲. $(w = z)$: ما v^t را به V' اضافه می‌کنیم و t را به L_v اضافه می‌کنیم. حال برای هر $i \in L_v$ (به ترتیب نزولی) تکرار می‌کنیم و تا زمانی که $t - i > \Delta$ باشد، (v^i, z) را به E' اضافه می‌کنیم. این کار در زمان $O(\Delta)$ انجام می‌شود.

۳. $(\{s, z\} \cup \{v, w\} = \emptyset)$: ما (v^i, w^t) را به V' اضافه می‌کنیم و t را به L_v و L_w اضافه می‌کنیم. حال برای هر $i \in L_v$ (به ترتیب نزولی) تکرار می‌کنیم و تا زمانی که $t - i > \Delta$ باشد، (v^i, w^t) را به E' اضافه می‌کنیم. سپس برای $i \in L_v$ (به ترتیب نزولی) تکرار می‌کنیم و تا زمانی که $t - i > \Delta$ شود، (w^i, v^t) را به E' اضافه می‌کنیم. این کار در زمان $O(\Delta)$ انجام می‌شود.

مشاهده می‌کنیم که پس از این روش، گراف جهت‌دار $D = (V', E')$ گسترش‌ـ(s, z) از $\mathcal{G}$ است و حداکثر دو راس برای هر یال زمانی در $\mathcal{G}$ اضافه شده است. بنابراین، $V' \leq |\mathcal{G}|$. این موضوع زمان اجرای کلی $O(|\mathcal{G}| \cdot \Delta)$ را ارائه می‌دهد. □

لم ۴.۴. *در یک گراف زمانی $\mathcal{G} = (V, (E_i)_{i \in [\ell]})$ اگر $s, x \in V$ دو راس مجزا باشند و $\Delta \leq \ell$ و $D = (V', E')$ باشد. یک مسیر زمانی (s, z) در $\mathcal{G}$ با طول k وجود دارد اگر و تنها اگر یک مسیر (s, z)-dipath با طول P' در D وجود داشته باشد به گونه‌ای که برای هر $v \in V$ داریم: $|V'(v) \cup V(P')| \leq 1$.*

اثبات. $(\rightarrow)$: فرض کنید $P = ((s, v_1, t_1), (v_1, v_2, t_2), \ldots, (v_{k'}, z, t_{k'}))$ یک مسیر زمانی بی‌قرار Δ-(s, z) در $\mathcal{G}$ به طول k باشد. می‌توان به صورت استقرایی یک مسیر جهت‌دار (s, z)-dipath P' را در D ساخت. توجه کنید که $P'_1 := ((s, v_1^{t_1}))$ یک مسیر جهت‌دار $(s, v_1^{t_1})$ به طول 1 در D است، زیرا یال $(s, v_1^{t_1})$ در E_s از D وجود دارد. حال فرض کنید $i \in [k' - 2]$ باشد و P'_i یک مسیر جهت‌دار $(s, v_i^{t_i})$ به طول i باشد به طوری که

۱. برای هر $j \in [i]$ داشته باشیم $|V'(v_j) \cap V(P'_i)| = 1$

۲. برای هر $v \in V \setminus \{s, v_1, \ldots, v_i\}$ داشته باشیم $|V'(v) \cap V(P'_i)| = 0$.

برای به دست آوردن یک مسیر جهت‌دار $(s, v_{i+1}^{t_{i+1}})$ به طول $i + 1$ به نام P'_{i+1}، مسیر P'_i را با یال $(v_i^{t_i}, v_{i+1}^{t_{i+1}})$ توسعه می‌دهیم. توجه کنید که $v_{i+1}^{t_{i+1}} \in V'$ به دلیل وجود یال زمانی $(\{v_i, v_{i+1}\}, t_{i+1})$ در $\mathcal{G}$ و اینکه یال $(v_i^{t_i}, v_{i+1}^{t_{i+1}}) \in E'$ است، زیرا $0 \leq t_{i+1} - t_i \leq \Delta$ داریم. توجه کنید که

۱. برای هر $j \in [i + 1]$ داریم $|V'(v_j) \cap V(P'_{i+1})| = 1$

۲. برای هر $v \in V \setminus \{s, v_1, \ldots, v_{i+1}\}$ داریم $|V'(v) \cap V(P'_{i+1})| = 0$.

پس یک مسیر جهت‌دار $(s, v_{k'-1}^{t_{k'-1}})$ به طول $k-1$ به نام P'_{k-1} داریم که شرایط ۱ و ۲ را ارضا می‌کند و می‌توان آن را (به شیوه مشابه) به یک مسیر جهت‌دار (s, z) به طول k توسعه داد به طوری که برای هر $v \in V$ داشته باشیم $|V'(v) \cap V(P')| \leq 1$.

($\Leftarrow$): فرض کنید P' یک مسیر جهت‌دار (s, z) به طول k در D باشد به طوری که برای هر $v \in V$ داشته باشیم $|V'(v) \cap V(P')| \leq 1$. فرض کنید $V(P') = \{s, v_1^{t_1}, \ldots, v_{k-1}^{t_{k-1}}, z\}$. توجه کنید که وجود یک یال از s به $v_1^{t_1}$ در D به این معنا است که یک لبه زمانی $(\{s, v_1\}, t_1)$ در $\mathcal{G}$ وجود دارد. به همین ترتیب، وجود یک یال از $v_i^{t_i}$ به $v_{i+1}^{t_{i+1}}$ به این معنا است که یک لبه زمانی $(\{v_i, v_{i+1}\}, t_{i+1})$ در $\mathcal{G}$ وجود دارد و $0 \leq t_{i+1} - t_i \leq \Delta$ است، برای هر $i \in [k-2]$. علاوه بر این، وجود یک یال از $v_{k-1}^{t_{k-1}}$ به z به این معنا است که t_k وجود دارد به طوری‌که لبه زمانی $(\{v_k, z\}, t_k)$ در $\mathcal{G}$ وجود دارد و $0 \leq t_k - t_{k-1} \leq \Delta$. پس $P = ((s, v_1, t_1), (v_1, v_2, t_2), \ldots, (v_{k'-1}, z, t_{k'}))$ یک گام زمانی بی‌قرار Δ-(s, z) به طول k در $\mathcal{G}$ است. در نهایت، $|V'(v) \cap V(P')| \leq 1$ برای هر $v \in V$ به این معنا است که $v_i \neq v_j$ برای هر $i \neq j$ با $i, j \in 0, \ldots, k$. پس P یک مسیر زمانی بی‌قرار Δ-(s, z) به طول k است.
□

درک قضیه ۱.۴. در اینجا الگوریتم Williams [۴۷] را با توجه به نیازهای خاص ما سازگار می‌کنیم. برای این منظور، نمادگذاری استانداردی از نظریه جبری معرفی می‌کنیم. یک مدار حسابی C بر روی یک حلقه جابجایی R یک گراف ساده برچسب‌دارِ جهت‌دارِ گراف دوره‌ای است که گره‌های داخلی آن با $+$ (گیت های جمع) یا $\times$ (گیت های ضرب) برچسب‌گذاری شده‌اند و گره‌های ورودی آن با عناصری از $R \cup X$ برچسب‌گذاری شده‌اند، جایی که X مجموعه‌ای از متغیرهاست. یک گره با درجه ورودی صفر (گیت های ورودی) عنصر $R[X]$ را نشان می‌دهد و یک گره با درجه خروجی صفر (گیت خروجی) وجود دارد. اندازه C تعداد رئوس در گراف است. یک مدار حسابی C بر روی R به صورت طبیعی یک چند جمله‌ای $P(X)$را محاسبه می‌کند که یک گیت ورودی چند جمله‌ای را که با آن برچسب گذاری شده باشد نشان می‌دهد. یک گیت جمع (ضرب) مجموعه چند جمله‌ای را که توسط همسایگان درجه ورودی آن نشان داده شده است، نشان می‌دهد. اگر چند جمله‌ای گیت خروجی C معادل $P(X)$ باشد، C مقدار $P(X)$ را نشان می‌دهد.

لم ۵.۴. (★) *فرض کنید k عددی طبیعی باشد و $D = (V, A)$ یک گراف جهت‌دار با جزئیات $V = \biguplus_{i=0}^{n} V_i$ باشد، به‌طوری‌که $V_0 = \{s\}$ و $V_n = \{z\}$، در این صورت یک مدار حسابی C وجود دارد که یک چند جمله‌ای $Q(X)$ با درجه حداکثر $k+1$را نشان می‌دهد و $Q(X)$ یک چند جمله‌ای با درجه حداکثر $k+1$ دارد، اگر و تنها اگر یک مسیر (s, z) با طول حداکثر k در D وجود داشته باشد به‌طوری‌که به ازای تمام $i \in [n]$ داشته باشیم $|V(P) \cup V_i| \leq 1$. علاوه بر این، $|X| = n+1$ اندازه $O(k(n+|A|))$ دارد و تمام گیت های ضربِ دو درجه ورودی دارند.*

این چندجمله‌ای در لم ۵.۴ به شکل بازگشتی به صورت $Q(X) = x_\perp Q_z^k$ با متغیرهای $X = \{x_\perp, x_0\} \cup \{x_i | i \in [n]\}$ تعریف می‌شود، که در آن برای تمام $v \in V \setminus \{s\}$ ها $Q_s^0 := x_0$ و $Q_v^0 := x_\perp$ است. برای تمام vهای متعلق V و jهای متعلق به $[k]$، تعریف می‌کنیم $Q_v^j = \Sigma_{(u,v)\in A} Q_u^{j-1} x_i$ ، که در آن $v \in V_i$ است. ایده این چندجمله‌ای مشابه با ایده‌ی Williams [۴۷] است، اما در اینجا به جای داشتن یک متغیر برای هر راس، ما فقط یک متغیر برای همه رئوس در یک قسمت از تقسیم Vداریم. حال ما می‌توانیم از نتیجه‌ی زیر از Williams [۴۷] استفاده کنیم.

قضیه ۶.۴. ([۴۷]). *فرض کنید $Q(X)$ یک چند جمله‌ای با درجه حداکثرk باشد که توسط یک مدار حسابی از اندازه n بدون ضرب اسکالر و همه گیت های ضرب دارای درجه ورودیِ دو باشد. یک الگوریتم تصادفی وجود دارد که در زمان $2^k n^{O(1)}$ اجرا می‌شود و با احتمال $(\geq 1/5)$ خروجی yes (مثبت) را می‌دهد، ولی اگر یک عبارت چندجمله‌ای در گسترش جمعی یا ضربی Q وجود داشته باشد و در صورت عدم وجود عبارت چندجمله‌ای همیشه خروجی no (منفی) را می‌دهد.*

قضیه ۱.۴ (۱) از لم ۳.۴ تا ۵.۴ و قضیه ۶.۴ بدست می‌آید. حال ما نشان می‌دهیم چگونه بخش چند جمله‌ای یک الگوریتم قطعی را بهبود ببخشیم.
برای اثبات قضیه ۱.۴ (۲)، ابتدا توجه می‌کنیم که در گسترش (s,z)، مسیر (s,z) در گرافِ جهت‌دار، یک مسیر زمانی (s,z)ـبی‌استراحت را به دقت توصیف می‌کند. سپس یک الگوریتم برای یافتن چنین مسیر P (اگر وجود داشته باشد) را نشان می‌دهیم. برای این منظور، یک مسئله به نام مسیر مستقل و برخی اصطلاحات استاندارد از نظر نظریه ماتروید(matroid) [۴۳] را معرفی می‌کنیم. یک زوج $(U, \mathcal{I})$ ،(که U مجموعه اصلی و $\mathcal{I} \subseteq 2^U$ یک خانواده از مجموعه‌های مستقل است) یک ماتروید است اگر شرایط زیر برقرار باشد: اگر $A' \subseteq A$ و $A \in \mathcal{I}$ باشد، آنگاه $A' \in \mathcal{I}$ است؛ و اگر $A, B \in \mathcal{I}$ و $|A| < |B|$ باشد، آنگاه یک $x \in B \setminus A$ وجود دارد طوری‌که $A \cup \{x\} \in \mathcal{I}$ برقرار است. یک مجموعه مستقل بصورت $A \in \mathcal{I}$ شامل‌بیشترین موارد یک ماتروید $M = (U, I)$ است. ماتروید یکنواخت با رتبه r بر روی U، ماتروید $\mathcal{I} = \{S \subseteq U | \, |S| \leq r\}$ است. یک ماتروید $(U, \mathcal{I})$ خطی یا قابل نمایش بر روی یک میدان F است اگر ماتریسی وجود داشته باشد که ورودی‌های آن در F قرار داشته باشد و ستون‌های آن با عناصر U برچسب گذاری شده باشند به‌طوری‌که $S \in \mathcal{I}$ در صورتی و تنها در صورتی است که ستون‌های ماتریس با برچسب‌های در S بر روی F مستقل خطی باشند. چنین ماتریسی را یک نمایش از $(U, \mathcal{I})$ می‌نامیم. حال ما آماده بیان مسئله مسیر مستقل هستیم. با دریافت یک گراف جهت‌دار$D = (V, E)$، دو رأس متمایز s،z از مجموعه V و یک نمایش A_M از یک ماتروید $M = (V, \mathcal{I})$ با رتبه r بر روی یک میدان متناهی F،می‌پرسیم که آیا یک مسیر (s,z) با طول حداکثر k در D وجود دارد به‌طوری که $V(P) \in \mathcal{I}$ باشد. برای بخش دیگر اثبات قضیه ۱.۴ (۲)، ما نیاز به یک الگوریتم نمایی با وابستگی خطی به اندازه ورودی داریم که فقط بر روی یک مجموعه مستقل از ماتروید و نه

مجموعه رئوسی که یک گراف بدون یال را ایجاد می‌کنند، مبتنی باشد. برای این منظور، با تکیه بر خانواده‌های نماینده(representative) [۲۲] ، موارد زیر را نشان می‌دهیم.

قضیه ۷.۴. (★). *یک نمونه (D, s, z, A_M) از مسیر مستقل می‌تواند در زمان $O(2^{wr}m)$ با عملیاتی بر روی میدان F که F میدان A_M که در آن r رتبه M و m تعداد یال‌ها در D است و $w < 2.373$.*

توجه کنید که با توجه به لم ۴.۴، در گراف زمانیِ $\mathcal{G}$ یک مسیر (s, z) وجود دارد اگر و تنها اگر یک مسیر P در گسترشِ $D = (V', E')$ از $\mathcal{G}$ وجود داشته باشد به طوری که $V(P)$یک مجموعه مستقل در ماتروید تقَسیمِ $M = (V', \{X \subseteq V' \mid \forall v \in V : |X \cap V'(v)| \leq 1\})$ باشد.

یک kـبرش ماتروید $(U, \mathcal{I})$ ، یک ماتروید $(U, \{X \in \mathcal{I} \mid |X| \leq k\})$ است به طوری که همه مجموعه‌های مستقل اندازه حداکثر k را دارند. در واقع kـبرش یک ماتروید خطی نیز یک ماتروید خطی است [۳۸]. ما در کاهش از مسیر زمانیِ بی‌استراحت کوتاه به مسیر مستقل، از یک $(k+1)$ـبرش از ماتروید M استفاده می‌کنیم. دو رویکرد کلی برای محاسبه نمایشی برای kـبرش از یک ماتروید خطی شناخته شده است - یکی تصادفی است [۳۸] و یکی قطعی است [۳۷]. هر دو رویکرد نیاز به یک میدان بزرگ دارند که یک عملیات روی آن میدان نسبتاً آهسته است. با این حال، برای ماتروید خاصِ ما، ما از ماتریس واندرموند برای محاسبه نمایش بر روی یک میدان متناهی کوچک استفاده می‌کنیم. توجه کنید که با استفاده از الگوریتم Lokshtanov [۳۷] یا Marx [۳۸] بر روی M، نمی‌توانیم یک زمان اجرای خطی در اندازه ورودی بدست آوریم.

لم ۸.۴. (★). *با توجه به یک جهان U با اندازه n، یک تقسیم بندیِ $U = P_1 \uplus \cdots \uplus p_q$ ، و یک عددِ طبیعیِ k، می‌توانیم در زمان $O(kn)$ یک نمایش A_M برای ماتروید*

$$M = (U, \{X \subseteq U \mid |X| \leq k \ , \ \forall i \in [q] : \ |X \cap P_i| \leq 1\})$$

که A_M بر روی یک میدان متناهی $\mathbb{F}$ تعریف شده است، محاسبه کنیم.

حال از لم های ۳.۴، ۴.۴ و ۸.۴ و قضیه ۷.۴، می‌توانیم به قضیه ۱.۴ (۲) برسیم.

فصل ۵

نظریهٔ پیچیدگی محاسباتی برای گراف اصلی

در این بخش، ما پیچیدگی محاسباتی پارامتری شده ی مسیر زمانی بی‌استراحت را بررسی خواهیم کرد. ما با نشان دادن نتایج اثبات‌پذیری پارامتر های ثابت شروع می‌کنیم که مستقیماً توسط قضیه ۷.۴ نشان داده می‌شود. می‌توانیم مشاهده کنیم که هر مسیرِ یک گراف می‌تواند حداکثر دو برابر تعداد راس‌های پوشش راسی گراف (به اضافه یک) راس داشته باشد، زیرا نمی‌توانیم دو راس خارج از پوشش راسی را پشت سر بازدید کنیم. این مشاهده اساساً در محیط زمانی نیز صادق است. اگر ما تعداد پوشش راسی گراف را با $vc_{\downarrow}$ نشان دهیم، می‌توانیم نتیجه بگیریم که هر مسیر زمانی بی‌استراحت می‌تواند حداکثر طول $2vc_{\downarrow}+1$ داشته باشد. از یک نظر طبقه‌بندی، ما می‌توانیم این محدودیت را با مشاهده کردن اینکه طول هر مسیر زمانی بی‌استراحت توسط طول هر مسیر در گراف زیرین محدود می‌شود، بهبود ببخشیم. طول یک مسیر در گراف زیرین می‌تواند با $2^{O(td_{\downarrow})}$ محدود (bound) شود [۴۱] ، جایی که $td_{\downarrow}$ عمق درختی گراف زیرین است.

مشاهده ۱.۵. *مسیر زمانی بی‌استراحت، با پارامتر $td_{\downarrow}$ از عمق درختی گراف زیرین، قابل حل شدن در پارامتر ثابت است.*

یکی از چالش‌های کمّی در پیچیدگی محاسباتی، تعداد یال‌های بازخوردی گراف زیرین است که به صورت زیر حل می‌شود.

قضیه ۲.۵. (★). *مسیر زمانی بی‌استراحت، در زمان $2^{O(f)} \cdot |\mathcal{G}|$ قابل حل است، که در آن f تعداد یال‌های بازخوردی گراف زیرین است.*

ما توجه می‌کنیم که طبق نتیجه ۲.۳، قضیه ۲.۵ به طور نسبی بهینه است، مگر اینکه ETH ناموفق باشد. به طور خلاصه، الگوریتم ما برای اثبات قضیه ۲.۵ شامل پنج مرحله زیر است:

۱. حذف کامل تمام رئوس با درجه ۱ از $G_{\downarrow}$ (به جز s و z).

۲. مجموعه یال‌های بازخوردی با کمترین اندازه F گراف $G_{\downarrow}$ را محاسبه کنید.

۳. مجموعه $\mathcal{P}$ از بزرگترین مسیرهای $G_{\downarrow} - F$ را محاسبه کنید و توجه داشته باشید که $|\mathcal{P}| = O(f)$.

۴. برای گراف $G_{\downarrow}$ به ازای یک مسیر s-z یال های بازخوردی در F و مسیرها در $\mathcal{P}$ راحدس بزنید .

۵. بررسی کنید که آیا مسیر (s, z) "حدس زده شده" یک مسیر زمانی بی‌استراحت (s, z) در $\mathcal{G}$ است یا نه.

نتایج بخش‌های ۳ تا ۵ تصویر خوبی از پیچیدگی محاسباتیِ پارامتری شده برای مسیر زمانی بی‌استراحت ارائه می‌دهند، به این معنی که برای بیشتر پارامترهای شناخته شده گراف (استاتیک) می‌دانیم که مسئله در FPT است یا [1]W-سخت یا para-NP-hard است. به شکل ۲.۱ نگاه کنید.

فهم ما از کلاس گراف‌های زمانی که می‌توانیم به طور کارآمد مسیر زمانی بی‌استراحت را حل کنیم به نقاط زیر خلاصه می‌شود: ما می‌توانیم به طور کارآمد بررسی کنیم که آیا یک مسیر زمانی بی‌استراحت در گراف زمانی $\mathcal{G}$ وجود دارد اگر تعدادی محدود از مسیرهای (s, z) در $G_{\downarrow}$ وجود دارد. علاوه بر این، ما با قضیه ۱.۳ و ۴.۳ و نتیجه ۳.۳ نتایج سختی برای گراف‌های زمانی با گراف زیرین محدود شده را ارائه کردیم، به شکل ۲.۱ نگاه کنید.

در نهایت، ما نشان می‌دهیم که برای تمام پارامترهای مورد بررسی و بیشترین پارامترهای ساختاری گراف زیرین، قرار نیست هسته های چندجمله‌ای به دست آوریم.

گزاره ۳.۵. (★). *مسیر زمانی بی‌استراحت، با پارامتر n (تعداد راس‌ها) برای همه $\Delta \geq 1$ ها هسته چندحمله ای ندارد مگر اینکه $NP \subseteq coNP/poly$ باشد.*

فصل ۶

تعداد رئوس بازخوردی با زمان‌بندی شده(feedback vertex number)

در این بخش، نسخه زمانی جدیدی از پارامتر "feedback vertex number" را معرفی می‌کنیم. طبق قضیه ۴.۳، می‌دانیم که مسیر زمانی بی‌استراحت، هنگامی که با پارامتر "feedback vertex number" گراف زیرین پارامتری شده باشد، W[1]-hard است. به همین دلیل، بررسی پارامترهای بزرگتر با هدف به دست آوردن نتایج قابلیت محاسباتی مفید است. ما یک پارامتر جدید به نام "timed feedback vertex number" ارائه می‌دهیم که تعداد نمایان شدن رئوسی را که باید از یک گراف زمانی حذف شوند تا گراف زیرین آن بدون چرخه باشد، اندازه‌گیری می‌کند. لازم به ذکر است که داشتن رئوس در مجموعه حذف، ما را به "حدس زدن" در مورد زمان ورود و خروج در مسیر زمانی بی‌استراحت (s, z)، کمک می‌کند، علاوه بر این،حدس زدن اینکه ترتیب نمایان شدن رئوس چگونه است باید بازدید شود. لازم به ذکر است که مطالعاتی نیز درباره حذف یال‌ها از گراف‌های زمانی برای از بین بردن چرخه‌های زمانی [۲۶] ، به عبارتی مسیرهای زمانی از یک راس به خودش، انجام شده است. قبل از تعریف رسمی پارامتر timed feedback vertex number، نشانه‌گذاری برای حذف شدن رئوس از یک گراف زمانی را معرفی می‌کنیم. به طور کلی، هنگامی که یک راس را از یک گراف زمانی حذف می‌کنیم، مجموعه رئوس آن را تغییر نمی‌دهیم، اما همه یال‌های زمانی را که راس حذف شده را به عنوان یکی از انتهایشان داشته باشند، حذف می‌کنیم. اگر $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ یک گراف زمانی باشد و $\mathcal{G} - X := (V, (E_i)_{i\in[\ell]})$ یک مجموعه از رئوس باشد، آنگاه $X \subseteq V \times [\ell]$ را خواهیم داشت که داریم:

$$E'_i = E_i \setminus \{e \in E_i \mid \exists (v, i) \in X, v \in e\}$$

تعریف ۱.۶. (*Timed Feedback Vertex Number*) *فرض کنید $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ یک گراف زمانی باشد. یک مجموعه timed feedback vertex برای $\mathcal{G}$، یک مجموعه*

$X \subseteq V \times [\ell]$ از رئوس است که در آن $G_{\downarrow}(\mathcal{G} - X)$ بدون چرخه باشد. تعداد timed feedback vertex یک گراف زمانی $\mathcal{G}$، کمینه ی اندازه یک مجموعه timed feedback vertex برای $\mathcal{G}$ است.

می‌توانیم مشاهده کرد که برای هر گراف زمانی، timed feedback vertex number حداقل به اندازه feedback vertex number گراف زیرین است و حد بالایی آن حاصلضرب feedback vertex number گراف زیرین و زمان حیات(عمر) است. علاوه بر این، توجه می‌کنیم که timed feedback vertex number در قابلیت تغییر ترتیب لایه‌ها ثابت است. در پایان این بخش، نشان می‌دهیم که چگونه می‌توان یک مجموعه timed feedback vertex را به صورت کارآمد محاسبه کرد.

الگوریتم ۱ الگوریتم FPT برای مسیر زمانی بی‌استراحت با پارامتر مجموعه گره‌های بازخورد زمان‌بندی شده.

ورودی: گراف زمانی $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ با $s, z \in V$، مجموعه گره‌های بازخورد زمان‌بندی شده X با شرط $s, z \notin \{v \mid (v,t) \in X\}$، و $\Delta \in \mathbb{N}$.
خروجی: بله، اگر یک مسیر زمانی بی‌استراحت Δ-(s,z) وجود داشته باشد، در غیر اینصورت خیر.

for each valid partition $O] \ I] \ U = X$ **do**
  $G' \leftarrow G - U$ and $x \leftarrow |I \cup O|$.
  $T \leftarrow G' - (v \in V|(v,t) \in O \cup I \cup s, z)$.
  for each Δ-ordering $(v_0, t_0) \leq \cdots \leq (v_{x+1}, t_{x+1})$ of $I \cup O \cup (s, z \times \perp)$ **do**
    $P_i \leftarrow \emptyset$, for all $i \in [x+1]$.
    for $i \leftarrow 1$ to $x+1$ **do**
      if $v_{i-1} = v_i$ **then**
        $P_i = \emptyset$.
      end if
      for each $e_1 = (v_{i-1}, w, t), e_2 = (u, v_i, t')$ of G' where $v_{i-1} \neq v_i$ **do**
        $T' \leftarrow T + e_1, e_2$.
        if $\exists$ (t_{i-1}, t_i)-valid Δ-restless temporal (v_{i-1}, v_i)-path P in T' **then**
          $P_i \leftarrow P_i \cup V(P) \setminus v_{i-1}, v_i$.
        end if
      end for
    end for
    $G \leftarrow$ intersection graph of the multiset $P_{(i)} \in P_i | i \in [x+1]$.
    Define $c : V(G) \to [x+1]$, $P_{(i)} \mapsto i$.
    if (G, c) has a multicolored independent set of size $x+1$ **then**
      return yes.
    end if
  end for
end for
return no.
=0

نتیجه اصلی این بخش این است که مسیر زمانی بی‌استراحت، هنگامی که با timed feedback vertex number پارامتری شده باشد، قابلیت محاسباتی با پارامتر ثابت دارد.

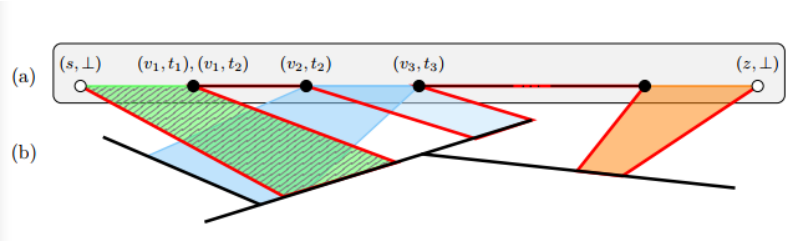
قضیه ۲.۶. (★). *به شرطی که یک مجموعه timed feedback vertex با اندازه x برای یک گراف زمانی $\mathcal{G} = (V, (E_i)_{i\in[\ell]})$ داشته باشیم، می‌توانیم در زمان $O(6^x x! \cdot max\{|\mathcal{G}|^3, |V|^4 x^2\})$ تصمیم بگیریم که آیا یک مسیر زمانی بی‌استراحت به طول در $\mathcal{G}$ با شرایط $s, z \in V$ و $\Delta \in \mathbb{N}$ وجود دارد یا خیر.*

الگوریتمی که برای نشان دادن قضیه ۲.۶ ارائه می‌دهیم، Chordal Multicolored Independent Set را حل می‌کند. در این مسئله، با گرفتن یک گراف $G = (V, E)$ و یک رنگ‌آمیزی راسی $c : V \to [k]$ را داریم، باید تصمیم بگیریم که آیا G یک مجموعه مستقل از اندازه k را دارای دقیقاً یک راس از هر رنگ می‌کند یا خیر. این مسئله به عنوان NP-complete شناخته شده است [[۹]، ۲ لم]. و در زمان $O(3^k \cdot |V|^2)$ قابل حل است [[۷]، گزاره ۶.۵]. الگوریتم ما برای Restless Temporal Path به طور خلاصه از این قدم‌های محاسباتی پیروی می‌کند:

۱. حدس میزنیم که کدامیک از رئوس timed feedback vertex در چه ترتیبی در restless temporal path ظاهر می‌شوند.

۲. بخش‌های مسیر بین دو راسِ timed feedback vertex را با حل یک نمونه Chordal Multicolored Independent Set محاسبه کنید.

ما یک شرح دقیق از الگوریتم خود را در الگوریتم ۱ ارائه می‌دهیم. در اینجا، یک تقسیم بندی $O \uplus I \uplus U$ از مجموعه نمایان شدن رئوس X معتبر است اگر $v \neq v'$ برای همه‌ی رئوس متمایز $(v,t), (v',t') \in I$ و برای همه‌ی رئوس متمایز $(v,t), (v',t') \in O$ باشد.



شکل ۱.۶ : نشان دهنده بخش مربوطه از سلسله مراتب پارامترهای نتایج الگوریتم ۱ نشان را می‌دهد. در بخش (a)، مجموعه $(\{s,z\} \times \{\perp\}) \cup I \cup O$ را نشان می‌دهد و (b) گراف پایه گراف زمانی $\mathcal{T}$ که یک جنگل است را نشان می‌دهد. مسیر زمانی بی‌قرار - restless (s,z) در مسیر ضخیم قرمزی است که از مسیرهای زمانی معتبر (تعریف ۳.۶) (s,v_1) و (v_1,v_2) بر روی $\mathcal{T}$ استفاده می‌کند.

تعریف ۳.۶. *فرض کنید $U \uplus I \uplus O$ یک تقسیم معتبر از X باشد و $(v_i, t_i), (v_{i+1}, t_{i+1}) \in (\{s, z\} \times \{\perp\}) \cup I \cup O$ باشد به‌طوری که $v_i \neq v_{i+1}$ و $t_i \leq t_{i+1}$ باشد و p_i یک مسیر (v_i, v_{i+1}) زمانی با زمان حرکت t_d و زمان ورود t_a باشد. در این صورت p_i، (t_{i-1}, t_i, I, O) ـ معتبر است اگر شرایط زیر برقرار باشد:*

۱. $(v_{i-1}, t_{i-1}) \in I \Rightarrow t_{i-1} \leq t_d \leq t_{i-1} + \Delta$،

۲. $(v_{i-1}, t_{i-1}) \in O \Rightarrow t_d = t_{i-1}$،

۳. $(v_i, t_i) \in I \Rightarrow t_a = t_i$،

۴. $(v_i, t_i) \in O \Rightarrow t_a \leq t_i \leq t_a + \Delta$.

توجه کنید که اگر یک مسیر زمانی (v_i, v_{i+1}) با شرایط معتبرِ (t_i, t_{i+1}) وجود داشته باشد و همچنین مسیر زمانی (v_{i+1}, v_{i+2}) با شرایط معتبرِ (t_{i+2}, t_{i+2}) وجود داشته باشد، ما می‌توانیم آن‌ها را به هم چسبانده و یک (t_i, t_{i+2}) زمانی با شرایط معتبر (v_i, v_{i+2}) بدست آوریم. بنابراین، اگر بین تمام جفت متوالی ها در که به جز انتها های گره‌ها اشتراک ندارند، مسیر زمانی بی‌قرار زمانی با شرایط معتبر وجود داشته باشد، مسیر زمانی s-z وجود دارد.

ایده الگوریتم ۱ این است که مسیر زمانیِ (s, z) تقسیم معتبری از مجموعه گره‌های زمانی X را ایجاد می‌کند به‌گونه‌ای که اگر P در زمان t به گره v برسد، $(v, t) \in I$ و اگر P از گره v در زمان t خارج شود، $(v, t) \in O$ و در غیر این صورت $(v, t) \in U$. علاوه بر این، اگر $M := I \cup O \cup (\{s, z\} \times \{\perp\})$ را براساس حرکت P (از s به z) مرتب کنیم، آنگاه این یک ترتیب برحسب Δ است به‌گونه‌ای که یک زیرمسیر P' از P متناظر با $(v, t), (v', t') \in M$ متوالی در M با $v \neq v'$ در یک گراف زمانی $\mathcal{T}'$ از خط (۹) معتبر است.

در این الگوریتم، تلاش می‌شود تمام تقسیم‌بندی‌های ممکن از X و تمام Δ- ترتیب‌های مربوطه را بررسی کنیم. برای هر یک از این‌ها، برای تمام مسیرهای زمانیِ معتبر (u, w) زمانی با حرکت‌های متوالی $(u, t), (w, t')$ گره‌های $V(P) \cap V(\mathcal{T})$ را در خانواده P_i ذخیره می‌کنیم. توجه کنید که اگر برای تمام $i \in \{1, \ldots, x+1\}$ ها $|\mathcal{P}_i| \geq 0$ برقرار باشد، آنگاه یک walk بی‌قرار (s, z) در $\mathcal{G}$ وجود دارد. بنابراین، برای پیدا کردن یک مسیر زمانی (s, z) restless-، باید $x+1$ جفت مجموعه $P_1^{(1)}, \ldots, P_{x+1}^{(x+1)}$ را پیدا کنیم، به‌طوری که $P_i \cap P_j = \emptyset$ برای همه $i \neq j$. در اینجا، مشاهده می‌کنیم که گراف اشتراکی در خط (۱۲) گره‌ای است که برای آن یک الگوریتم فرعی Chordal Multicolored Independent Set از Bentert و دیگران [۷] را برای پیدا کردن مجموعه‌های جفت بازیافتنی پی‌درپی استفاده می‌کنیم.

برای نتیجه گیری از قضیه ۲.۶ وجود الگوریتم ۱ به سادگی، باید یک مجموعه گره‌های بازگشتی زمانی را به طور کارآمد محاسبه کنیم. این به وضوح NP-hard است، زیرا مسأله Feedback

Vertex Set NP-complete را تعمیم می‌دهد [۳۳] . با این حال، ما به دنبال این هستیم که یک Feedback Vertex Set را محاسبه کنیم.

قضیه ۴.۶. (★). *یک مجموعه بازگشتی زمانی با حداقل تعداد گره‌ها در گراف زمانی $\mathcal{G}$، با زمان محاسباتی $4^x \cdot |\mathcal{G}^{O(1)}|$ محاسبه می‌شود، که در آن x تعداد گره‌های بازگشتی زمانی در $\mathcal{G}$ است. علاوه بر این، الگوریتمی با ضریب تقریب ۸ برای محاسبه مجموعه بازگشتی زمانی وجود دارد.*

توضیحات: در این قضیه، ما به دنبال محاسبه یک مجموعه بازگشتی زمانی با حداقل تعداد گره‌ها در گراف زمانی $\mathcal{G}$ هستیم. یعنی مجموعه‌ای از گره‌های بازگشتی زمانی که حذف آن‌ها باعث جدا شدن گراف می‌شود و این مجموعه را می‌خواهیم به صورت بهینه محاسبه کنیم. در ادامه، دو الگوریتم برای محاسبه این مجموعه بازگشتی زمانی ارائه می‌دهیم.

الگوریتم اول: الگوریتمی با زمان محاسباتی $4^x \cdot |\mathcal{G}^{O(1)}|$ برای محاسبه مجموعه بازگشتی زمانی با حداقل تعداد گره‌ها در گراف زمانی $\mathcal{G}$ وجود دارد. در این الگوریتم، از الگوریتم برنامه‌ریزی پویا برای محاسبه مقدار تابع داینامیکی استفاده می‌شود. این الگوریتم در چهار مرحله انجام می‌شود:

۱. محاسبه مقدار تابع داینامیکی برای هر یک از گره‌های زمانی $\mathcal{G}$.

۲. محاسبه مقدار تابع داینامیکی برای گره‌های بازگشتی زمانی $\mathcal{G}$.

۳. انتخاب یک گره بازگشتی زمانی با حداقل مقدار تابع داینامیکی و اضافه کردن آن به مجموعه بازگشتی زمانی.

۴. حذف گره انتخاب شده و تکرار مراحل ۱ تا ۳ تا زمانی که هیچ گره بازگشتی زمانی در گراف باقی نماند.

به‌طوری که در هر مرحله، مقدار تابع داینامیکی برای هر یک از گره‌های زمانی $\mathcal{G}$ با استفاده از مقادیر قبلی تابع داینامیکی محاسبه می‌شود.

الگوریتم دوم: الگوریتمی با ضریب تقریب ۸ برای محاسبه مجموعه بازگشتی زمانی با حداقل تعداد گره‌ها در گراف زمانی $\mathcal{G}$ وجود دارد. در این الگوریتم، ابتدا یک مجموعه اولیه بازگشتی زمانی با حداقل تعداد گره‌ها با استفاده از الگوریتم ابتکاری محاسبه می‌شود. سپس به صورت تکراری، گره با بیشترین درجه در مجموعه بازگشتی زمانی انتخاب شده و از مجموعه حذف می‌شود. این فرایند تا زمانی که هیچ گره بازگشتی زمانی در گراف باقی نماند، تکرار می‌شود.

با توجه به این که این الگوریتم با ضریب تقریب ۸ عمل می‌کند، ممکن است که مجموعه بازگشتی زمانی به دست آمده تعدادی گره اضافی نسبت به بهینه داشته باشد، اما حداکثر تعداد گره‌های اضافی هشت برابر تعداد گره‌های بازگشتی زمانی بهینه است.

در نتیجه، با توجه به این دو الگوریتم، می‌توان یک مجموعه بازگشتی زمانی با حداقل تعداد گره‌ها در گراف زمانی $\mathcal{G}$ را با زمان محاسباتی $4^x \cdot |\mathcal{G}^{O(1)}|$ یا با ضریب تقریب ۸ محاسبه کرد.

فصل ۷

نتیجه گیری

در پایان، ما پیچیدگی محاسباتی (پارامتری شده) مسئله‌ی Restless Temporal Path را، یک نسخه معمولی برای پیدا کردن مسیرهای زمانی، که زمان انتظار در هر راس محدود شده است، تحلیل کردیم. بر خلاف نسخه‌ی غیر بی‌استراحت یا نسخه‌ی "walking"، این مسئله حتی در موارد محدود نیز بسیار پیچیده است. در جنبه مثبت، الگوریتمی کارآمد برای پیدا کردن مسیرهای زمانی بی‌استراحت کوتاه ارائه دادیم و پارامترهای ساختاری گراف اصلی و گراف زمانی را شناسایی کردیم که الگوریتم‌های پارامتری شده را ممکن می‌سازند. این نتایج، بهترین درک از پیچیدگی محاسباتی مسئله‌ی Restless Temporal Path را فراهم می‌کنند و می‌توانند در توسعه الگوریتم‌های موثرتر برای این مسئله کمک کنند.

واژه‌نامه فارسی به انگلیسی

گراف زمانی Temporal graph; a graph in which the edges change over time, representing interactions that occur at different time steps.

گراف پویا Dynamic graph; a graph that evolves and changes structure and parameters over time.

شبکه زمانی Temporal network; a network with nodes and time-labeled edges representing contacts over time.

گراف متغیر با زمان Time-varying graph; a graph whose structure varies as a function of time.

مسیر زمانی Temporal path; a path through a temporal graph that respects the time ordering of edges.

مسیر پویا Dynamic path; a path in a dynamic graph that respects the time ordering of edge availabilities.

مسیر متغیر با زمان Time-varying path; a path in a time-varying graph that follows the time-labeled sequence of edges.

مسیر زمانی بی‌قرار Restless temporal path; a temporal path with a bound on the waiting time at each node.

مسیر زمانی ناآرام Restless time path; a temporal path that does not remain at any node for longer than the specified limit.

پیمایش زمانی Temporal walk; a walk along a temporal graph respecting the time ordering of edges.

قدم زدن زمانی Temporal walking; traversing a temporal graph over time while respecting time-ordering of edges.

انتظار زمانی Temporal waiting; waiting or pausing at a node in a temporal graph for some duration.

توقف زمانی Temporal stop; halting at a node in a temporal graph for a certain time period.

درنگ زمانی Temporal pause; an interval where a temporal path is paused at a node before continuing.

محدودیت زمان انتظار Waiting time constraint; an upper bound on the time a temporal path can wait at a node.

محدودیت توقف زمانی Temporal stop constraint; a constraint on the maximum duration of stopping at a node.

محدودیت درنگ زمانی Temporal pause constraint; a bound on how long a path can pause at a node.

گام زمانی Time step; an individual time unit in which edges of a temporal graph are present.

مرحله زمانی Time stage; a distinct time period in the lifetime of a temporal graph.

لایه زمانی Time layer; a snapshot of a temporal graph at a specific time step.

لحظه زمانی Time instant; a precise moment of time during the evolution of a temporal graph.

طول عمر Lifetime; the total duration of existence of a temporal graph.

دوره حیات Lifespan; the entire duration for which a temporal graph exists.

پیچیدگی محاسباتی Computational complexity; the amount of resources required to solve a computational problem.

پیچیدگی الگوریتمی Algorithmic complexity; the complexity of an algorithm expressed as a function of input size.

سختی محاسباتی Computational hardness; the difficulty of solving a computational problem as resource requirements grow.

سختی الگوریتمی Algorithmic hardness; difficulty of designing an efficient algorithm measured by asymptotic growth rate.

الگوریتم کارآمد Efficient algorithm; an algorithm that solves problems with minimal computational resources.

الگوریتم مؤثر Effective algorithm; an algorithm that capably solves problems within reasonable resource bounds.

پارامتری شدن Parameterization; analyzing computational complexity in terms of multiple parameters.

پارامتردار کردن Parameterizing; defining a problem using parameters to study its complexity.

پارامتر ثابت Fixed parameter; parameter assumed constant to restrict analysis to specific values.

پارامتر ثابت شده Fixed parameterized; computational problems with one or more fixed parameters.

کلاس پیچیدگی Complexity class; a set of problems with related resource-based time complexity.

قابلیت محاسباتی Computability; whether a problem can be solved by any algorithm within finite resources.

قابل حل بودن Solvability; whether an algorithmic solution exists to find a problem's solution.

هسته چندجمله‌ای Polynomial kernel; a reduced equivalent instance of fixed size solvable in polynomial time.

هسته چندجمله‌ای شدن Kernelization; the process of shrinking problem instances to polynomial kernels.

کاهش Reduction; transformation of one problem into another while preserving solution properties.

فروکاهش Reducibility; relating the inherent difficulty between two problems via transformations.

مشاهدات Observations; findings obtained by analysis and examination of a problem's structure and behavior.

بررسی‌ها Examinations; detailed investigations into computational problems to discover properties.

پارامتر ساختاری Structural parameter; a parameter derived from structural properties of a problem's input.

پارامتر سازه‌ای Structural parameter; parameter based on the structure of a problem instance.

گراف پایه Underlying graph; the static graph formed by aggregating all edges of a temporal graph.

گراف زیرین Underlying graph; an implicit static graph encapsulated by a temporal graph.

یال بازخوردی Feedback edge; an edge that creates a cycle when added to a graph.

رأس بازخوردی Feedback vertex; a vertex whose removal disconnects all cycles passing through it.

رأس بازخوردی زمانی Timed feedback vertex; a vertex breaking all temporal cycles if removed from all time layers.

مجموعه رأس‌های بازخوردی Feedback vertex set; a set of vertices whose removal results in an acyclic graph.

واژه‌نامه انگلیسی به فارسی

Temporal graph گرافی که در آن یال‌ها در طول زمان تغییر می‌کنند و نشان‌دهنده تعاملاتی هستند که در گام‌های زمانی مختلف رخ می‌دهند.

Dynamic graph گرافی که به مرور زمان ساختار و پارامترهای آن تغییر می‌کند.

Temporal network شبکه‌ای با گره‌ها و یال‌های دارای برچسب زمانی که نشان‌دهنده تماس‌های زمانی است.

Time-varying graph گرافی که ساختار آن به عنوان تابعی از زمان تغییر می‌کند.

Temporal path مسیری از یک گراف زمانی که ترتیب زمانی یال‌ها را رعایت می‌کند.

Dynamic path مسیری در یک گراف پویا که ترتیب زمانی در دسترس بودن یال‌ها را رعایت می‌کند.

Time-varying path مسیری در یک گراف متغیر با زمان که توالی برچسب‌های زمانی یال‌ها را دنبال می‌کند.

Restless temporal path مسیر زمانی با محدودیت زمان انتظار در هر گره.

Restless time path مسیر زمانی که در هیچ گره‌ای بیش از حد مشخصی نمی‌ماند.

Temporal walk پیمایشی در امتداد یک گراف زمانی با رعایت ترتیب زمانی یال‌ها.

Temporal walking عبور از یک گراف زمانی در طول زمان با رعایت ترتیب زمانی یال‌ها.

Temporal waiting انتظار زمانی؛ انتظار یا توقف در یک گره از گراف زمانی به مدتی مشخص.

Temporal stop توقف زمانی؛ توقف در یک گره از گراف زمانی به مدت معینی.

Temporal pause درنگ زمانی؛ فاصله‌ای که مسیر زمانی در یک گره مکث می‌کند قبل از ادامه مسیر.

Waiting time constraint محدودیت زمان انتظار؛ محدودیت بالایی برای مدت زمان انتظار یک مسیر زمانی در هر گره.

Temporal stop constraint محدودیت توقف زمانی؛ محدودیتی بر حداکثر زمان توقف در یک گره.

Temporal pause constraint محدودیت درنگ زمانی؛ محدودیتی برای مدت زمان توقف یک مسیر در یک گره.

Time step واحد زمانی مجزایی که در آن یال‌های گراف زمانی وجود دارند.

Time stage دوره زمانی مجزایی در طول عمر یک گراف زمانی.

Time layer تصویر لحظه‌ای از گراف زمانی در یک گام زمانی مشخص.

Time instant لحظه دقیقی از زمان در طول تکامل یک گراف زمانی.

Lifetime طول عمر؛ مدت زمان کل وجود یک گراف زمانی.

Lifespan دوره حیات؛ کل مدت زمان وجود یک گراف زمانی.

Computational complexity میزان منابع مورد نیاز برای حل یک مسئله محاسباتی.

Algorithmic complexity پیچیدگی یک الگوریتم بیان‌شده به عنوان تابعی از اندازه ورودی.

Computational hardness دشواری حل یک مسئله محاسباتی با افزایش نیازهای منابع.

Algorithmic hardness دشواری طراحی یک الگوریتم کارآمد اندازه‌گیری‌شده با نرخ رشد آسیمپتوتیک.

Efficient algorithm الگوریتمی که مسائل را با کمترین منابع محاسباتی حل می‌کند.

Effective algorithm الگوریتمی که به خوبی مسائل را در محدوده منابع معقول حل می‌کند.

Parameterization تحلیل پیچیدگی محاسباتی بر حسب چند پارامتر.

Parameterizing تعریف یک مسئله با استفاده از پارامترها برای مطالعه پیچیدگی آن.

Fixed parameter پارامتری که برای محدود کردن تحلیل به مقادیر خاص ثابت فرض می‌شود.

Fixed parameterized مسائل محاسباتی با یک یا چند پارامتر ثابت.

Complexity class مجموعه‌ای از مسائل با پیچیدگی زمانی مرتبط مبتنی بر منابع.

Computability امکان حل شدن یک مسئله توسط هر الگوریتمی در محدوده منابع متناهی.

Solvability وجود یک راه حل الگوریتمی برای یافتن پاسخ یک مسئله.

Polynomial kernel هسته چندجمله‌ای؛ مسئله کوچک و معادلی که با اندازه ثابت در زمان چندجمله‌ای قابل حل است.

Kernelization هسته چندجمله‌ای شدن؛ فرایند کوچک کردن مسائل به هسته‌های چندجمله‌ای.

Reduction کاهش؛ تبدیل یک مسئله به مسئله دیگر با حفظ ویژگی‌های راه حل.

Reducibility فروکاهش؛ برقراری ارتباط بین دشواری ذاتی دو مسئله از طریق تبدیل‌ها.

Observations یافته‌های به دست آمده از تحلیل و بررسی ساختار و رفتار یک مسئله.

Underlying graph گراف پایه؛ گراف ایستای حاصل از تجمیع همه یال‌های یک گراف زمانی.

Underlying graph گراف زیرین؛ گراف ایستای ضمنی مشتمل بر یک گراف زمانی.

NP-hard سخت؛-NP مسائلی که حداقل به اندازه مسائل کلاس NP دشوار هستند.

Feedback edge یال بازخوردی؛ یالی که با اضافه شدن به گراف موجب ایجاد دور می‌شود.

Feedback vertex رأس بازخوردی؛ رأسی که با حذف آن همه دورهایی که از آن می‌گذرند قطع می‌شوند.

کتاب‌نامه

[۱] Akanksha Agrawal، Pallavi Jain، Lawqueen Kanesh، and Saket Saurabh. *Parameterized complexity of conflict-free matchings and paths.* Algorithmica، pages ،۲۷–۱ .۲۰۲۰

[۲] Eleni C. Akrida، Jurek Czyzowicz، Leszek G□sieniec، Łukasz Kuszner، and Paul G. Spirakis. *Temporal flows in temporal networks.* Journal of Computer and System Sciences، ،۶۰–۱۰۳:۴۶ .۲۰۱۹

[۳] Eleni C. Akrida، Leszek G□sieniec، George B. Mertzios، and Paul G. Spirakis. *The complexity of optimal design of temporally connected graphs.* Theory of Computing Systems، ،۹۴۴–۹۰۷:(۳)۶۱ .۲۰۱۷

[۴] Eleni C. Akrida، George B. Mertzios، Sotiris Nikoletseas، Christoforos Raptopoulos، Paul G. Spirakis، and Viktor Zamaraev. *How fast can we reach a target vertex in stochastic temporal graphs*؟ Journal of Computer and System Sciences، ،۸۳–۱۱۴:۶۵ .۲۰۲۰

[۵] Kyriakos Axiotis and Dimitris Fotakis. *On the size and the approximability of minimum temporally connected subgraphs.* In Proceedings of the ۴۳rd International Colloquium on Automata، Languages، and Programming (ICALP ،(۱۶’ pages ،۱۴۹:۱۴–۱۴۹:۱ .۲۰۱۶

[۶] Albert-László Barabási. *Network Science.* Cambridge University Press، .۲۰۱۶

[۷] Matthias Bentert، René van Bevern، and Rolf Niedermeier. *Inductive k-independent graphs and c-colorable subgraphs in scheduling: a review.* Journal of Scheduling، ،۲۰–۳:(۱)۲۲ .۲۰۱۹

[۸] Kenneth A. Berman. *Vulnerability of scheduled networks and a generalization of Menger's theorem.* Networks: An International Journal، ۲۸ . ۱۹۹۶، ۱۲۵–۱۳۴:(۳)

[۹] René van Bevern، Matthias Mnich، Rolf Niedermeier، and Mathias Weller. *Interval scheduling and colorful independent sets.* Journal of Scheduling، ۱۸ . ۲۰۱۵، ۴۴۹–۴۶۹:(۵)

[۱۰] Sandeep Bhadra and Afonso Ferreira. *Complexity of connected components in evolving graphs and the computation of multicast trees in dynamic networks.* In International Conference on Ad-Hoc Networks and Wireless، pages ۲۵۹ . ۲۰۰۳، ۲۵۹–۲۷۰

[۱۱] B.-M. Bui-Xuan، Afonso Ferreira، and Aubin Jarry. "Computing shortest، fastest، and foremost journeys in dynamic networks." *International Journal of Foundations of Computer Science*، ۱۴ . ۲۰۰۳، ۲۶۷–۲۸۵:(۰۲)

[۱۲] Arnaud Casteigts، Paola Flocchini، Emmanuel Godard، Nicola Santoro، and Masafumi Yamashita. "On the expressivity of time-varying graphs." *Theoretical Computer Science*، ۲۷ . ۲۰۱۵، ۵۹۰–۳۷:

[۱۳] Arnaud Casteigts، Paola Flocchini، Walter Quattrociocchi، and Nicola Santoro. "Time-varying graphs and dynamic networks." *International Journal of Parallel، Emergent and Distributed Systems*، ۲۷ . ۲۰۱۲، ۳۸۷–۴۰۸:(۵)

[۱۴] Arnaud Casteigts، Joseph Peters، and Jason Schoeters. "Temporal cliques admit sparse spanners." In *Proceedings of the ۴۶th International Colloquium on Automata، Languages، and Programming (ICALP (۱۹'*، volume ۱۳۲ of LIPIcs، pages .۱۳۴:۱۴–۱۳۴:۱ Schloss Dagstuhl–Leibniz-Zentrum für Informatik، .۲۰۱۹

[۱۵] Marek Cygan، Fedor V. Fomin، Łukasz Kowalik، Daniel Lokshtanov، Dániel Marx، Marcin Pilipczuk، Michał Pilipczuk، and Saket Saurabh. *Parameterized Algorithms*. Springer، .۲۰۱۵

[۱۶] Reinhard Diestel. *Graph Theory*، ۵th Edition، volume ۱۷۳ of Graduate Texts in Mathematics. Springer، .۲۰۱۶

[۱۷] Rodney G Downey and Michael R Fellows. *Fundamentals of Parameterized Complexity*. Springer، ۲۰۱۳.

[۱۸] Ken TD Eames and Matt J Keeling. "Contact tracing and disease control." *Proceedings of the Royal Society of London. Series B: Biological Sciences*، ۲۷۰ (۱۵۳۳):۲۵۶۵–۲۵۷۱، ۲۰۰۳.

[۱۹] Jessica Enright، Kitty Meeks، George Mertzios، and Viktor Zamaraev. "Deleting edges to restrict the size of an epidemic in temporal networks." In *Proceedings of the ۴۴th International Symposium on Mathematical Foundations of Computer Science (MFCS ۲۰۱۹)*، volume ۱۳۸ of LIPIcs، pages ۵۷:۱–۵۷:۱۵ Schloss Dagstuhl–Leibniz-Zentrum für Informatik، ۲۰۱۹.

[۲۰] Luca Ferretti، Chris Wymant، Michelle Kendall، Lele Zhao، Anel Nurtay، Lucie Abeler-Dörner، Michael Parker، David Bonsall، and Christophe Fraser. "Quantifying SARS-CoV-۲ transmission suggests epidemic control with digital contact tracing." *Science*، ۲۰۲۰.

[۲۱] Till Fluschnik، Hendrik Molter، Rolf Niedermeier، Malte Renken، and Philipp Zschoche. "Temporal graph classes: A view through temporal separators." *Theoretical Computer Science*، ۸۰۶:۱۹۷–۲۱۸، ۲۰۲۰.

[۲۲] Fedor V. Fomin، Daniel Lokshtanov، Fahad Panolan، and Saket Saurabh. "Efficient computation of representative families with applications in parameterized and exact algorithms." *Journal of the ACM*، ۶۳ (۴):۲۹:۱–۲۹:۶۰، ۲۰۱۶.

[۲۳] Fedor V. Fomin، Daniel Lokshtanov،Fahad Panolan، and Saket Saurabh. "Representative families of product families." *ACM Transactions on Algorithms*، ۱۳ (۳):۳۶:۱–۳۶:۲۹، ۲۰۱۷.

[۲۴] Steven Fortune، John E. Hopcroft، and James Wyllie. "The directed subgraph homeomorphism problem." *Theoretical Computer Science*، ۱۱۱:۱۰–۱۲۱، ۱۹۸۰.

[۲۵] F̆anic̆a Gavril. "The intersection graphs of subtrees in trees are exactly the chordal graphs." *Journal of Combinatorial Theory، Series B*، ۱۶ (۱):۴۷–۵۶، ۱۹۷۴.

[۲۶] Roman Haag، Hendrik Molter، Rolf Niedermeier، and Malte Renken. "Feedback edge sets in temporal graphs." In *Proceedings of the ۴۶th International Workshop on Graph-Theoretic Concepts in Computer Science (WG (۲۰'*، volume ۱۲۳۰۱ of Lecture Notes in Computer Science، pages ۲۱۲–۲۰۰. Springer، ۲۰۲۰.

[۲۷] Anne-Sophie Himmel، Matthias Bentert، André Nichterlein، and Rolf Niedermeier. "Efficient computation of optimal temporal walks under waiting-time constraints." In *Proceedings of the ۸th International Conference on Complex Networks and their Applications*، volume ۸۸۲ of SCI، pages ۵۰۶–۴۹۴. Springer، ۲۰۱۹.

[۲۸] Petter Holme. "Modern temporal network theory: a colloquium." *The European Physical Journal B*، ۸۸(۹):۲۳۴، ۲۰۱۵.

[۲۹] Petter Holme. *Temporal network structures controlling disease spreading.* Physical Review E، ۹۴.۰۲۲۳۰۵:۲، ۲۰۱۶.

[۳۰] Petter Holme and Jari Saramäki (eds.). *Temporal Network Theory.* Springer، ۲۰۱۹.

[۳۱] Russell Impagliazzo and Ramamohan Paturi. *On the complexity of k-SAT.* Journal of Computer and System Sciences، ۶۲(۲):۳۶۷–، ۲۰۰۱. ۳۷۵،

[۳۲] Russell Impagliazzo، Ramamohan Paturi، and Francis Zane. *Which problems have strongly exponential complexity?* Journal of Computer and System Sciences، ۶۳(۴):۵۱۲–۵۳۰، ۲۰۰۱.

[۳۳] Richard M Karp. *Reducibility among combinatorial problems.* In *Complexity of Computer Computations*، pages ۸۵–۱۰۳. Springer، ۱۹۷۲.

[۳۴] David Kempe، Jon Kleinberg، and Amit Kumar. *Connectivity and inference problems for temporal networks.* Journal of Computer and System Sciences، ۶۴(۴):۸۲۰–۸۴۲، ۲۰۰۲.

[۳۵] William Ogilvy Kermack and Anderson G McKendrick. *A contribution to the mathematical theory of epidemics.* Proceedings of the Royal Society of London، Series A.، ۱۱۵(۷۷۲):۷۰۰–۷۲۱، ۱۹۲۷.

[۳۶] Matthieu Latapy، Tiphaine Viard، and Clémence Magnien. *Stream graphs and link streams for the modeling of interactions over time.* Social Network Analysis and Mining، ۸(۱):۶۱، ۲۰۱۸.

[۳۷] Daniel Lokshtanov، Pranabendu Misra، Fahad Panolan، and Saket Saurabh. *Deterministic truncation of linear matroids.* ACM Transactions on Algorithms، ۱۴(۲):۱:۱۴–۱۴:۲۰، ۲۰۱۸.

[۳۸] Dániel Marx. *A parameterized view on matroid optimization problems.* Theoretical Computer Science، ۴۱۰(۴۴):۴۴۷۱–۴۴۷۹، ۲۰۰۹.

[۳۹] George B Mertzios، Othon Michail، and Paul G Spirakis. *Temporal network optimization subject to connectivity constraints.* Algorithmica، ۸۱(۴):۱۴۱۶–۱۴۴۹، ۲۰۱۹.

[۴۰] Othon Michail. *An introduction to temporal graphs: An algorithmic perspective.* Internet Mathematics، ۱۲(۴):۲۳۹–۲۸۰، ۲۰۱۶.

[۴۱] Jaroslav Nešet□il and Patrice Ossona De Mendez. *Sparsity: Graphs، Structures، and Algorithms*. Springer، ۲۰۱۲.

[۴۲] Mark E J Newman. *Networks*. Oxford University Press، ۲۰۱۸.

[۴۳] James G. Oxley. *Matroid Theory*. Oxford University Press، ۱۹۹۲.

[۴۴] Raj Kumar Pan and Jari Saramäki. *Path lengths، correlations، and centrality in temporal networks*. Physical Review E، ۸۴(۱):۰۱۶۱۰۵، ۲۰۱۱.

[۴۵] Manuel Sorge and Mathias Weller et al. *The graph parameter hierarchy*، ۲۰۱۸ ۲۰۲۰. URL: https://manyu.pro/assets/parameter-hierarchy.pdf.

[۴۶] Craig A. Tovey. *A simplified NP-complete satisfiability problem*. Discrete Applied Mathematics، ۸(۱):۸۵–۸۹، ۱۹۸۴.

[۴۷] Ryan Williams. *Finding paths of length k in O□(۲k) time*. Information Processing Letters، ۱۰۹(۶):۳۱۵–۳۱۸، ۲۰۰۹.

[۴۸] H. Wu، J. Cheng، Y. Ke، S. Huang، Y. Huang، and H. Wu. *Efficient algorithms for temporal path computation*. IEEE Transactions on Knowledge and Data Engineering، ۲۸(۱۱):۲۹۲۷–۲۹۴۲، ۲۰۱۶.

[۴۹] Philipp Zschoche، Till Fluschnik، Hendrik Molter، and Rolf Niedermeier. *The complexity of finding separators in temporal graphs*. Journal of Computer and System Sciences، ۷۲:۹۲–۱۰۷، ۲۰۲۰.

Abstract

According to the parameterized approach, this research investigated the computational complexity of restless time paths and presented effective algorithms to solve it in some cases.

College of Science
School of Mathematics, Statistics, and Computer Science

Finding Temporal Paths Under Waiting Time Constraints

Abolfazl Kabiri

Supervisor: Morteza Mohammad-Nouri

A thesis submitted in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science

JULY, 2023