DOR: 20.1001.1.2538161.1402.8.3.5.1

DOI : 10.22034/CSJ.2023.184644

پاییز ۱۴۰۲، دوره ۸، شماره ۳، پیاپی ۳۰

دریافت مقاله: ۱۴۰۲/۰۴/۱۴

پذیرش مقاله: ۱۴۰۲/۰۶/۲۸

نوع مقاله: پژوهشی

الگوریتمی برای رنگ‌آمیزی همسایه- مکان‌یاب درخت‌ها

رضا ندیمی*

استادیار گروه علوم کامپیوتر- دانشکده علوم ریاضی- دانشگاه مازندران- بابلسر- ایران

پست الکترونیکی: nadimi@umz.ac.ir

چکیده

در این مقاله، الگوریتمی برای رنگ‌آمیزی همسایه- مکان‌یاب درخت‌ها ارایه گردیده است. رنگ‌آمیزی گراف‌ها و کاربردهای آن از مباحث اصلی و پر کاربرد گراف‌هاست. رنگ‌آمیزی گراف‌ها در دو حوزه رنگ‌آمیزی گره‌ها و رنگ‌آمیزی یال‌های گراف مورد مطالعه قرار گرفته اند. در رنگ‌آمیزی گره‌ها، در سال‌های اخیر مفاهیم جدیدی از رنگ‌آمیزی گراف‌ها مانند "رنگ‌آمیزی مکان‌یاب" و "رنگ‌آمیزی همسایه- مکان‌یاب"، مطرح شده و مورد مطالعه قرار گرفته‌اند. تخصیص اعضای مجموعه رنگ $C = \{c_1, c_2, \dots, c_k\}$ به مجموعه گره‌های یک گراف را یک k- رنگ آمیزی (مناسب) گوییم اگر و فقط اگر به هیچ زوج همسایه‌ای رنگ یکسان اختصاص نیافته باشد. با اعمال محدودیت‌های بیشتر در رنگ‌آمیزی، به انواع دیگری از این مسئله خواهیم رسید. تعداد حداقل رنگ برای رنگ‌آمیزی مناسب یک گراف را عدد رنگی گراف گوییم؛ این عدد برای انواع رنگ‌آمیزی به طور مشابهی تعریف می‌شود. پیدا کردن عدد رنگی یک گراف در فرم بهینه‌سازی مسئله و همچنین تشخیص k-رنگ پذیری گراف برای $k>2$، در فرم تصمیم مسئله، از مسایل معروف np-hard هستند. نوع خاصی از رنگ‌آمیزی مناسب که موضوع این مقاله است، رنگ‌آمیزی همسایه- مکان‌یاب گره‌ها است. در این مسئله گره‌ها باید طوری رنگ‌آمیزی شوند که علاوه بر غیر یکسان بودن رنگ همسایه‌ها، مجموعه رنگ همسایه‌های گره‌های همرنگ، متمایز از هم باشند. در مبحث رنگ‌آمیزی همسایه- مکان‌یاب گره‌ها با وجود مطالعات وسیع صورت گرفته در زوایای نظری بحث، از جمله روابط بین عدد رنگی در انواع رنگ‌آمیزی‌ها و عدد رنگی گراف‌های خاص، از نظر الگوریتمی، در این زمینه نتیجه قابل توجهی وجود ندارد. در این مقاله، الگوریتمی برای رنگ‌آمیزی همسایه- مکان‌یاب درخت‌ها ارایه شده است. ثابت می‌کنیم الگوریتم از مرتبه زمانی چند جمله‌ای است و در مورد حداکثر رنگ‌های استفاده شده برای حالت‌های خاصی از درخت‌ها بحث خواهیم کرد.

واژه‌های کلیدی: رنگ‌آمیزی گراف‌ها، رنگ‌آمیزی درخت‌ها، رنگ‌آمیزی همسایه- مکان‌یاب.

۱. مقدمه

رنگ‌آمیزی گراف‌ها روشی موفق و معروف جهت متمایزسازی و شناسایی اجزای گراف شامل رئوس و یال‌ها است. غالبا با در نظر گرفتن کاربردهای مختلف، محدودیت‌های مختلفی در رنگ‌آمیزی گراف‌ها اعمال می‌شود. این محدودیت‌ها موجب مطرح شدن انواع مختلفی از رنگ‌آمیزی شده است. در رنگ‌آمیزی راسی گراف‌ها،

* نویسندهٔ مسئول

زمانی کــه تمایز رئوس مجاور مطرح باشــد، مســئله را رنگ‌آمیزی راسی گراف گویند[۱]. در صورتی که تمایز بین گره‌های همرنگ توســط مجموعه رنگ گره‌های با فاصله یکســان از آنها مدنظر باشد، مســئله تبدیل به رنگ‌آمیزی مکان‌یاب[1] می‌شــود[۲، ۳]. اگر الزام به تمایز در گره‌های با فاصله یک (همسایگی‌ها) وجود داشته باشد، آزادی عمل در رنگ‌آمیزی کمتر شــده و مسئله جدیدی به نام رنگ‌آمیزی همسایه- مکان‌یاب(NLC)[2] [٤، ٥، ٦] تعریف می‌شود.

اگر چــه قدمت مباحث رنگ‌آمیــزی در نظریه گراف‌ها به ســال ۱۸۵۲ برمی گردد، آغاز مباحــث الگوریتمی این حوزه به مراتب جدیدتر بوده و اوایل دهه ۱۹۷۰ را می‌توان ســرآغازی برای نگاه محاســباتی به رنگ‌آمیزی دانست. مســئله عدد رنگی، یکی از ۲۱ مسئله‌ای است که کارپ در سال ۱۹۷۲ NP-کامل بودن آنها را اثبات کرده است. همچنین در همان زمان الگوریتم‌های مختلفی با زمان اجرای نمایی برپایه روش عقب‌گرد توســعه یافتنــد. یکی از کاربردهای اصلی رنگ‌آمیزی گراف، تخصیص رجیستر در کامپایلرها در سال ۱۹۸۱ معرفی شــد. کاربردهای زیادی برای انواع رنگ‌آمیزی ذکر شده اســت؛ از اهم این کاربردها می‌توان به تشخیص خطا در شبکه‌ها [۷]، آزمایش بیولوژیکی [۸]، یادگیری ماشین [۹]، تجزیه و تحلیل بازی [۱۰] و آزمایش هم‌ریختی [۱۱] اشاره کرد.

۲. تعاریف و نمادها

در ایــن بخش به ارایــه تعاریف و معرفــی نمادهایی می‌پردازیم که در مقاله از آنها اســتفاده شــده است. برای حفظ ســادگی و انسجام، در سراســر مقاله از تعاریف و نمادهــای به کار رفته در کتاب "مقدمه‌ای بر نظریه گراف" [۱] استفاده خواهیم نمود.

گراف: گراف G، ساختاری متشکل از دو مجموعه است؛ مجموعــه‌ای از راس‌ها و مجموعــه‌ای از یال‌ها که هر یال توســط یک زوج از رئوس مشخص شده و نشانگر وجود رابطه‌ای بین آن دو راس اســت. غالبــا مجموعه رئوس با $V = \{v_1, v_2, \dots, v_n\}$، مجموعــه یال‌هــا بــا $E = \{e = (u, v) | \ u, v \epsilon V\}$ و خــود گراف بــه صورت G=(V, E) نشان داده می‌شود.

درخت: به گراف پیوســته بــدون دور، درخت گوییم. درخت‌ها در مقایســه بــا گراف‌های غیر درخــت، دارای ســاختار ساده‌تری هســتند که موجب می‌شود این دسته خاص از گراف‌ها، در مســایل مشابه توسط الگوریتم‌های کارآتری به جواب برسند. به گره‌های انتهایی درخت که یک همسایه دارند، برگ گویند. هر درخت حداقل دو برگ دارد زیرا در غیر این صورت گراف الزاما شامل دور خواهد بود. همچنین به مجموعه‌ای از درخت‌ها که به عنوان یک ساختار واحد در نظر گرفته شوند، جنگل گفته می‌شود.

k-رنگ‌آمیزی راسی: برای گراف G، یک k-رنگ‌آمیزی روی رئوس، تابعی اســت مانند f ازمجموعه رئوس $V(G)$ بــه مجموعه {k ,⋯ ,2 ,1} به طــوری که به ازای هر u و v مجاور، داشــته باشــیم: f(u) ≠ f(v). در اینجا مقدار f(v) را رنگ گره v می‌نامیم. جهت تمایز با رنگ‌آمیزی‌های خاص دیگر، این رنگ‌آمیزی «رنگ‌آمیزی مناسب»[3] نامیده می‌شود.

عــدد رنگی: عدد رنگــی گراف G که با $\chi(G)$ نشــان داده می‌شــود، حداقل مقدار k اســت که G به ازای آن یک k–رنگ‌آمیزی را می‌پذیرد.

افراز رنگی گراف: هر k–رنگ‌آمیزی از یک گراف، یک افــراز $\{S_1, S_2, \dots, S_k\}$ روی راس‌های گراف ایجاد می‌کند که در آن همه راسهای واقع در S_i دارای رنگ i–ام هستند.

فاصله از رنگ: فاصله گره u از مجموعه S_i یا به عبارتی رنگ i–ام به صورت $d(u, S_i) = \min\{d(u, v): v \in S_i\}$ تعریــف می‌شــود کــه در آن $d(u, v)$ تعــداد یال‌ها روی کوتاه‌ترین مسیر بین u و v است.

k-رنگ‌آمیــزی مکان‌یاب: یک k-رنگ‌آمیزی مناســب روی گره‌ها را k-رنگ‌آمیــزی مکان‌یاب گوییم اگر به ازای هر دو راس هم رنگ u و v، وجود داشــته باشــد i–ای که $d(u, S_i) \neq d(v, S_i)$.

1- Locating coloring
2- Neighbor-locating coloring
3- proper coloring

عـــدد رنگی مکان‌یاب: به کوچکترین عدد k که به ازای آن یک k–رنگ‌آمیزی مکان‌یاب برای گراف وجود داشـــته باشـــد، عدد رنگی مکان‌یـــاب آن گراف گوییـــم و آن را با $\chi_L(G)$ نشان می‌دهیم.

k–رنگ‌آمیزی همسایه– مکان‌یاب: مجموعه رنگ‌های اختصاص داده شـــده توســـط تابع f به همسایه‌های گره u را با $N_f(u)$ نشـــان می‌دهیم. اگـــر در یک k–رنگ‌آمیزی مناســـب، به ازای هر u و v با رنگ یکســـان داشته باشیم $N_f(u) \neq N_f(v)$ آنـــگاه آن رنگ‌آمیـــزی را رنگ‌آمیـــزی همسایه–مکان‌یاب گوییم.

عدد رنگی همسایه– مکان‌یاب: به کوچکترین عدد k که به ازای آن گراف G یک رنگ‌آمیزی همســـایه– مکان‌یاب را می‌پذیرد، عدد رنگی همســـایه– مکان‌یاب گراف G گوییم و آن را با $\chi_{NL}(G)$ نشان می‌دهیم.

با توجه به این که هر رنگ‌آمیزی همسایه– مکان‌یاب یک رنگ‌آمیزی معمولی است؛ همواره رابطه زیر برقرار است:

$$\chi_{NL}(G) \geq \chi(G)$$

همین موضوع در مورد رنگ‌آمیزی مکان‌یاب نیز صدق می‌کند و داریم:

$$\chi_L(G) \geq \chi(G)$$

از طرفـــی هـــر رنگ‌آمیـــزی همســـایه– مکان‌یاب یک رنگ‌آمیزی مکان‌یاب اســـت که در آن الزاما به ازای هر دو گره همرنگ رنگی وجود دارد که فاصله آن از یکی از گره‌ها دقیقا ۱ و از دیگری مخالف یک است. بنابراین داریم:

$$\chi_{NL}(G) \geq \chi_L(G)$$

و در نهایت داریم:

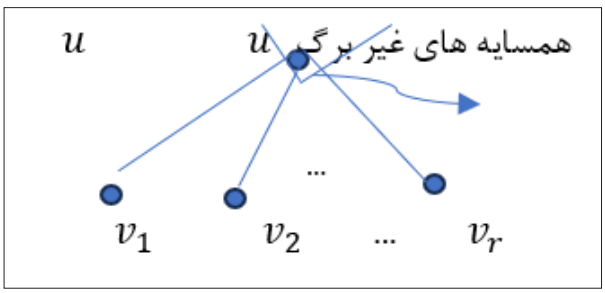
$$\chi_{NL}(G) \geq \chi_L(G) \geq \chi(G)$$

۳– الگوریتم رنگ‌آمیزی همســـایه– مکان‌یاب برای درخت‌ها

با توجه به Np-hard بودن مســـاله رنگ‌آمیزی مناسب در حالت کلـــی و عدم وجـــود الگوریتم‌هـــای رنگ‌آمیزی همســـایه–مکان‌یاب، پرداختن به حالت‌هـــای خاص مانند درخت‌ها می‌تواند گشایشـــی برای الگوریتم‌های مربوط به حالت کلی گراف‌ها یا حالت‌های خاص غیر درخت باشد.

در روش ارایـــه شـــده، با حداقل رنگ لازم شـــروع به رنگ‌آمیزی درخـــت می‌کنیم و در طول الگوریتم، به تدریج به تعداد رنگ‌ها افزوده خواهد شد.

ابتدا لیســـتی از گره‌های درخت ایجاد کـــرده و آنها را بر حســـب تعداد برگ‌های متصل به آنهـــا به صورت غیر صعودی مرتب می‌کنیم. حال از ابتدای لیســـت، گره دارای بیشـــترین برگ را به همراه برگ‌های متصـــل به آن رنگ می‌کنیم. فرض کنید گره انتخاب شده u، و برگ‌های متصل به آن $v_1, v_2, \ldots, v_r$ باشـــند (شکل ۱)؛ در این صورت u و برگ‌هایش را – چه u همسایه غیر برگ داشته باشد و چه نداشته باشد– نمی توان با کمتر از $r+1$ رنگ به صورت NLC رنگ‌آمیـــزی کرد. زیرا تنها همســـایه برگ‌ها گره u است و داشتن همسایه‌های با رنگ‌آمیزی غیر یکسان برای آنها غیر ممکن؛ بنابراین رنـــگ برگ‌ها باید دو به دو مجزا باشـــند و r رنگ برای رنگ‌آمیزی برگ‌ها مورد نیاز است. از طـــرف دیگر u همســـایهٔ همه برگ‌هاســـت و باید رنگی متفاوت ازهمه آنها داشـــته باشد، بنابراین در این مرحله از الگوریتم به $r+1$ رنگ نیاز داریم.



شکل ۱: یک گره با برگ‌ها و همسایه های غیر برگ آن

در ادامـــهٔ الگوریتم، رنگـــی را که به عنـــوان رنگ گره مرکزی (u) اســـتفاده شده اســـت، هرگز استفاده نخواهیم کرد. ولی رنگ‌های استفاده شده روی برگ‌ها را، تا زمانی که برای یک گره مرکزی مورد اســـتفاده قـــرار نگیرند، به دفعات اســـتفاده خواهیم کرد. به این منظور دو لیســـت L_1 و L_2 را بـــه ترتیب برای رنگ‌های فعـــال و رنگ‌های کنار گذاشته شده به کار می‌گیریم. در آغاز الگوریتم، دو لیست

خالی هستند. با ایجاد هر رنگ این رنگ وارد L_1 می‌شود و تا پیش از انتقال به L_2 به دفعات قابل استفاده خواهد بود. بعد از رنگ‌آمیزی گره مرکزی و برگ‌هایش، رنگ استفاده شده در مرکز را از L_1 حذف کرده و به L_2 اضافه می‌کنیم. لیست L_2 فقط برای جلوگیری از اضافه شدن رنگ تکراری مورد استفاده قرار می‌گیرد.

گره‌های رنگ شده را از درخت حذف می‌کنیم تا به درخت (یا جنگل) کوچکتری برسیم. به صورت تکراری سراغ پر برگ‌ترین گره در انتظار رنگ‌آمیزی می‌رویم و گره مرکزی و برگ‌های بی رنگش را با رنگ‌های موجود در L_1 رنگ‌های متفاوت می‌زنیم. اگر به تعداد کافی رنگ نداشتیم به تعداد مورد نیاز رنگ جدید به L_1 اضافه کرده و در رنگ‌آمیزی استفاده می‌کنیم. در ادامه الگوریتم رنگ‌آمیزی با جزییات بیان شده است و در قسمت بعد اثبات صحت الگوریتم و NLC بودن رنگ‌آمیزی آمده است.

الگوریتم NLC:

۱. $L_2 = L_1 = \phi$

۲. گره‌های درخت را در لیست Uncolored قرار داده، بر حسب تعداد برگ‌هایی که دارند به صورت غیر صعودی مرتب می‌کنیم.

۳. از ابتدای لیست Uncolored گره با بیشترین تعداد برگ را انتخاب کرده، به آن گره و تمامی برگ‌هایش رنگ‌های متمایزی از L_1 را اختصاص می‌دهیم. در صورت کمبود رنگ در L_1، به تعداد مورد نیاز رنگ جدید که در L_2 نباشند به L_1 اضافه می‌کنیم.

٤. رنگ اختصاص یافته به گره مرکزی را از L_1 به L_2 منتقل می‌کنیم.

۵. گره‌های رنگ شده را از لیست Uncolored حذف می‌کنیم.

٦. تعداد برگ‌های گره‌ها را بروز کرده صف را مرتب می‌کنیم.

۷. اگر Uncolored= ϕ، پایان الگوریتم، وگرنه به گام ۳ برمی گردیم.

قضیه ۱: در پایان الگوریتم NLC، یک رنگ‌آمیزی همسایه- مکان‌یاب کامل روی گره‌های درخت خواهیم داشت.

اثبات: اثبات قضیه را در سه قسمت زیر انجام خواهیم داد:

الف- همه گره‌ها رنگ شده اند.

ب- هیچ دو گره همسایه‌ای با رنگ یکسان وجود ندارند.

پ- اگر دو گره رنگ یکسانی داشته باشند آنگاه مجموعه رنگ‌های همسایه‌های آن دو گره متمایزند.

اثبات الف: هیچ گره‌ای پیش از رنگ شدن از لیست Uncolored حذف نمی‌شود، بنابراین با تهی شدن این لیست که شرط پایان الگوریتم است، همه گره‌ها رنگ شده‌اند.

اثبات ب: کلید اصلی اثبات ادعای ب این است که در هر مرحله از رنگ‌آمیزی، رنگ گره مرکزی از لیست رنگ‌های مجاز برای رنگ‌آمیزی حذف می‌شود. برای این که نشان دهیم هیچ دو همسایه‌ای با رنگ‌های یکسان رنگ نشده‌اند، گره‌ها را بر حسب این که در زمان رنگ شدن گره مرکزی یا برگ باشند به دو دسته مرکزی‌ها و برگ‌ها تقسیم می‌کنیم. مرکزی‌ها با همسایه‌های حذف شده خود همرنگ نیستند. زیرا همسایه‌های حذف شده، پیش از این گره رنگ شده‌اند، بنابراین مرکزی بوده‌اند و رنگشان از مجموعه رنگ‌های فعال حذف شده است. همچنین مرکزی‌ها رنگی متفاوت همسایه‌های برگ خود می‌گیرند. و نیز رنگی که به مرکزی می‌دهیم در مراحل بعدی مورد استفاده قرار نخواهند گرفت. بنابراین با همسایه‌های غیر برگ خود هم همرنگ نخواهد بود. حال ثابت می‌کنیم برگ‌ها هم با همسایه‌های خود همرنگ نیستند. برگ‌ها دو نوع همسایه دارند همسایه‌هایی که پیشتر رنگ شده‌اند و گره مرکزی مرحله جاری. همسایه‌های حذف شده، خود مرکزی بوده‌اند و رنگشان پیشتر غیر فعال شده است بنابراین با برگ کنونی همرنگ نیستند و همچنین همسایه مرکزی مرحله جاری طبق الگوریتم رنگ متفاوتی از برگ‌ها می‌پذیرد بنابراین همه گره‌ها رنگی متفاوت از همسایه‌های

خود را دارند.

اثبات پ: فرض کنیم در انتهای الگوریتم، دو گره u و v همرنگ هستند، ثابت می‌کنیم مجموعه رنگ‌های همسایه‌های این دو یکسان نیستند. از آنجایی که در طول یک مرحله، هیچ دو گرهی رنگ یکسان نمی‌گیرند، پس دو گره u و v در دو مرحله متفاوت رنگ شده‌اند. فرض کنیم گره u پیش از گره v رنگ شده باشد، از آنجایی که رنگ گره u حذف نشده است (در مراحل بعدی برای v استفاده شده است)، گره u، برگ بوده است. با توجه به آنچه گفته شد، دو گره u و v هم زمان برگ‌های یک گره مرکزی یکسان نبوده‌اند و همسایگی‌های یکسانی هم ندارند. بنابراین یا رنگ گره مرکزی برگ u در همسایه‌های v وجود ندارد و یا این که همسایه‌ای از v، در آینده رنگی به خود خواهد گرفت که در همسایه‌های گره u نبوده است زیرا همه رنگ‌های همسایه‌های u از مجموعه رنگ‌های فعال حذف شده‌اند.

قضیه ۲: الگوریتم NLC دارای پیچیدگی زمانی O(nlogn) است.

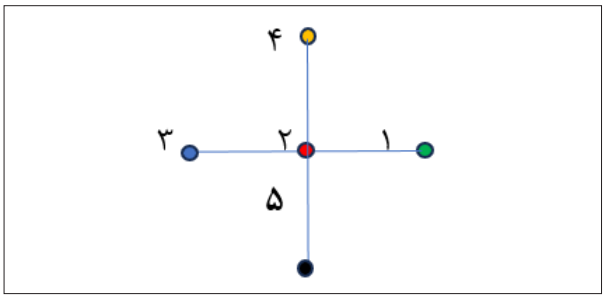
اثبات: گلوگاه محاسباتی الگوریتم، مرتب نگه داشتن گره‌های گراف بر حسب تعداد برگ است که در طول الگوریتم در حال تغییر هستند. این کار با استفاده از یک ساختار داده‌ای هرم بیشینه[4] در زمان O(nlogn) قابل انجام است. از آنجایی که هر گره حداکثر یک بار تبدیل به برگ می‌شود و بعد از رنگ‌آمیزی در هر مرحله فقط همسایه‌های غیر برگ گره مرکزی ممکن است به برگ تبدیل شوند، باقی اعمال الگوریتم از مرتبه O(n) خواهد بود.

۳-۱. رنگ‌آمیزی دسته هایی از درخت‌ها

در این بخش عملکرد الگوریتم بر روی دسته‌هایی از درخت‌ها را بررسی خواهیم کرد.

۳-۱-۱. گراف های ستاره

درهر گراف ستاره با n برگ، یک گره مرکزی داریم که به همراه برگ‌هایش در یک مرحله با n+1 رنگ متفاوت رنگ‌آمیزی می‌شوند. در شکل ۱ یک گراف ستاره با پنج گره داریم؛ گره ۲ بیشترین برگ را دارد که به همراه برگ‌های خود، طی یک مرحله با ۵ رنگ مجزا رنگ‌آمیزی می‌شوند.



شکل ۱: رنگ‌آمیزی گراف ستاره

قضیه ۳: الگوریتم در گراف‌های ستاره به جواب بهینه می‌رسد.

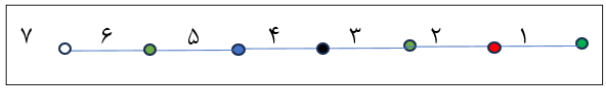
اثبات: در هر رنگ‌آمیزی NLC مفروض، گره مرکزی رنگی متفاوت از همه گره‌ها دارد. زیرا همسایه همه گره‌ها است. همچنین همه برگ‌ها دارای مجموعه همسایگی یکسان هستند (فقط یک همسایه دارند که مرکز است) بنابراین قابلیت تفکیک با رنگ همسایه را نداشته، الزاما دو به دو رنگ‌های متفاوت دارند.

۳-۱-۲. گراف‌های خطی

در گراف‌های خطی در هر مرحله، دو گره از یکی از دو سر گراف حذف شده و به یک گراف خطی کوچک‌تر می‌رسیم. در شکل ۲ یک گراف خطی با ۷ گره داریم. گره‌های ۲ و ٦ هر یک با داشتن یک برگ، پربرگ‌ترین گره‌های درخت هستند. فرض کنیم گره ۲ را انتخاب کرده، به همراه برگ ۱ به طور هم زمان با سبز و قرمز رنگ‌آمیزی کنیم؛ با توجه به مرکزی بودن گره ۲، رنگ قرمزاستفاده شده در آن در هیچ گره دیگری استفاده نخواهد شد و از لیست رنگ‌های فعال حذف می‌شود. با حذف گره‌های رنگ شده به یک گراف خطی به طول پنج با یک رنگ فعال(سبز) می‌رسیم، مرحله بعد گره ٤ به عنوان پر برگ‌ترین گره به همراه برگ خود یعنی گره ۳ به طور هم زمان با دو رنگ سبز و سیاه، رنگ‌آمیزی می‌شوند. رنگ سیاه از مجموعه رنگ‌های فعال حذف شده با فعال نگه داشتن رنگ سبز و حذف گره‌های رنگ شده به یک گراف خطی با سه گره

4- Max heap

می‌رسیم که با سه رنگ سبز و آبی و سفید رنگ‌آمیزی می‌شوند.



شکل ۲: رنگ‌آمیزی گراف خطی

قضیه ۴: تعداد رنگ‌های استفاده شده برای رنگ‌آمیزی گراف خطی برابر $\lceil n/2 \rceil+1$ است.

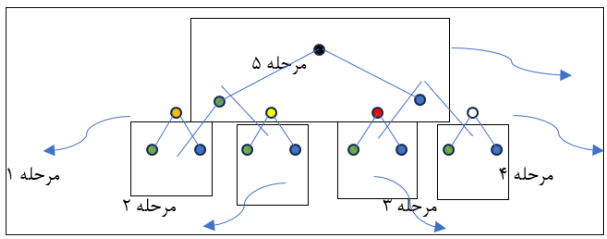
اثبات: در ابتدای الگوریتم دو رنگ تولید می‌شوند؛ در صورتی که n زوج باشد، در تمامی مراحل، دو گره حذف و یک رنگ جدید تولید می‌شوند. با توجه به این که دقیقا n/2 مرحله داریم، n/2 + 1 رنگ استفاده خواهد شد. به دلیل زوج بودن n این عدد برابر $\lceil n/2 \rceil+1$ خواهد بود. در صورتی که n فرد باشد، در مرحله آخر دو رنگ تولید و سه گره حذف می‌شوند؛ بنابراین تعداد مراحل الگوریتم برابر [n/2] است و یک رنگ در مرحله آخر بیشتر از بقیه مراحل تولید می‌شود. در نتیجه تعداد رنگ‌های استفاده شده برابر [n/2]+2 یا به عبارت دیگر(به دلیل فرد بودن n) $\lceil n/2 \rceil+1$ خواهد بود.

به نظر می‌رسد، تعداد رنگ‌های استفاده شده برای گراف خطی، با توجه به ساختار ساده آن زیاد است و می‌توان الگوریتم‌های اختصاصی بهتری استفاده کرد؛ ولی در اینجا هدف بررسی الگوریتمی است که بتواند از عهده رنگ‌آمیزی همسایه- مکان‌یاب همه درخت‌ها برآید.

۳-۱-۳. درخت‌ها ی دودویی کامل

در این بخش پیاده‌سازی الگوریتم روی درخت‌های دودویی کامل را بررسی می‌کنیم. برای تحلیل بهتر الگوریتم گره‌ها را از پایین به بالا رنگ‌آمیزی می‌کنیم. با توجه به ساختار درخت دودویی کامل همواره گره با بیشترین برگ در پایین‌ترین لایه درخت نیز وجود دارد (ممکن است در لایهٔ بالاتر هم وجود داشته باشد).

در شکل۳ یک درخت دودویی کامل داریم، در هر مرحله دقیقا سه گره رنگ شده و یک رنگ حذف می‌شود. رنگ‌های سبز و آبی تا انتهای الگوریتم فعال می‌مانند و همه سطوح فرد فقط با همین دو رنگ، رنگ‌آمیزی می‌شوند؛ در مقابل رنگ گره‌های واقع در سطوح زوج همگی رنگ متمایز می‌گیرند.



شکل ۳: رنگ‌آمیزی درخت دودویی کامل

قضیه ۵: الگوریتم NLC درخت دودویی کامل با n گره را حداکثر توسط $\frac{n}{3}+2$ رنگ، رنگ‌آمیزی می‌کند.

اثبات: درخت کامل با n گره، دارای log(n+1) سطح می‌باشد. نام این سطوح را از بالا به پایین، سطح ۰، سطح ۱، سطح ۲، ... و سطح $\log(n+1)-1$ در نظر می‌گیریم. گره‌های واقع بر نیمی از این سطوح فقط با دو رنگ رنگ‌آمیزی می‌شوند که بر حسب این که تعداد سطوح فرد یا زوج باشد، شامل $\frac{2n}{3}$ یا $\frac{2n}{3}+1$ گره هستند؛ زیرا در هر مرحله، از سه گره رنگ شده دو گره از رنگ‌های فعال قبلی استفاده می‌کنند. با توجه به این که تعداد گره‌های دیگر حداکثر $\frac{n}{3}$ است، بنابراین تعداد رنگ‌های استفاده شده حداکثر $\frac{n}{3}+2$ خواهد بود.

با اثباتی مشابه می‌توان قضیه فوق را به صورت زیر تعمیم داد.

قضیه ۶: الگوریتم NLC درختی را که همه گره‌های غیر برگ آن از درجه k باشند و شامل n گره باشد؛ حداکثر توسط $\frac{n}{k+1}+k$ رنگ، رنگ‌آمیزی می‌کند.

۴. نتیجه گیری

در این مقاله با ارایه الگوریتمی برای مسئلهٔ رنگ‌آمیزی همسایه-مکان‌یاب روی درخت‌ها، به صورت محاسباتی به مقوله‌ای که در سال‌های اخیر به صورت گسترده در پژوهش‌های مربوط به نظریه گراف دیده می‌شود، پرداخته شده است. با توجه به Np-hard بودن مسئلهٔ رنگ‌آمیزی

در حالت کلی و عدم وجود الگوریتم‌های رنگ‌آمیزی همسایه-مکان‌یاب، پرداختن به حالت‌های خاص مانند درخت‌ها می‌تواند گشایشی برای الگوریتم‌های مربوط به حالت کلی گراف‌ها یا حالت‌های خاص غیر درخت باشد. از طرفی کران‌های بالا در تعداد رنگ استفاده شده توسط الگوریتم می‌تواند توسط الگوریتم‌های جدید بهبود یابد. نگاه محاسباتی به این حوزه از نظریه گراف، می‌تواند موضوعات پژوهشی بسیاری را از پیچیدگی الگوریتم‌ها، تعداد رنگ‌های استفاده شده و حالت‌های خاص گراف‌ها در دسترس قرار دهد.

مراجع

1. West, D.B.: Introduction to graph theory, vol. 2. Prentice hall Upper Saddle River (2001)
2. Chartrand, G., Erwin, D., Henning, M., Slater, P., Zhang, P.: The locatingchromatic number of a graph. Bull. Inst. Combin. Appl. 36, 89 – 101 (2002)
3. Slater, P.J.: Leaves of trees. In: Proceedings of the 6th Southeastern Conference on Combinatorics, Graph Theory, and Computing. Congressus Numerantium, vol. 14, pp. 549–559 (1975)
4. Alcon, L., Gutierrez, M., Hernando, C., Mora, M., Pelayo, I.M.: Neighbor-locating colorings in graphs. Theoretical Computer Science 806, 144–155 (2020)
5. Slater, P.J.: Dominating and reference sets in a graph. Journal of Mathematical and Physical Sciences 22(4), 445–455 (1988)
6. D. A. Mojdeh. On the conjectures of neighbor locating coloring of graphs. Theoretical Computer Science, 922:300–307, 2022.
7. Karpovsky, M.G., Chakrabarty, K., Levitin, L.B.: On a new class of codes for identifying vertices in graphs. IEEE transactions on information theory 44(2), 599–611 (1998)
8. Moret, B.M.E., Shapiro, H.D.: On minimizing a set of tests. SIAM Journal on Scientific and Statistical Computing 6(4), 983–1003 (1985)
9. Chlebus, B.S., Nguyen, S.H.: On finding optimal discretizations for two attributes. In: Proceedings of the First International Conference on Rough Sets and Current Trends in Computing. vol. 1424, pp. 537–544. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
10. . Chv´atal, V.: Mastermind. Comb. 3(3), 325–329 (1983)
11. Babai, L.: On the complexity of canonical labeling of strongly regular graphs. SIAM J. Comput. 9(1), 212–216 (1980)