رنگ‌آمیزی گراف‌ها بااستفاده از شبکه شاره

علیرضا زارعی محمد قدسی

دانشکده‌ی مهندسی کامپیوتر
دانشگاه صنعتی شریف

چکیده

در این مقاله یک روش جدید برای حل مسئله‌ی رنگ‌آمیزی گراف براساس شبکه‌ی شاره ارائه می‌شود. در این روش، نشان می‌دهیم که می‌توان متناظر با هر گراف داده‌شده یک شبکه‌ی شاره با محدودیت‌هایی در شاره‌ی آن ایجاد کرد، به‌طوری‌که از محاسبه‌ی شاره‌ی بیشینه بتوان گراف اصلی را رنگ‌آمیزی کرد.

با توجه به NP‐تمام بودن مسئله‌ی رنگ‌آمیزی گراف، انتظار نمی‌رود که برای محاسبه‌ی شاره‌ی بیشینه با در نظر گرفتن محدودیت‌های فرض شده، الگوریتمی با هزینه‌ی زمانی چندجمله‌ای وجود داشته باشد، مگر آن‌که رده مسائل P و NP مساوی باشند. بااین حال ارائه‌ی الگوریتم‌های تقریبی برای محاسبه‌ی شاره، منجر به الگوریتم‌های تقریبی جدید برای مسئله‌ی رنگ‌آمیزی گراف و مسائل مشابه دیگری از جمله مجموعه‌ی مستقل رأسی و پوشش رأسی می‌شود.

دراین مقاله پس از بیان جزئیات الگوریتم ارائه شده و تحلیل و اثبات آن، روشی تقریبی برای حل مسئله‌ی شاره‌ی بیشینه ارائه می‌شود تا به‌کمک آن، روش تقریبی جدیدی برای رنگ‌آمیزی گراف به‌دست آید.

کلیدواژه‌ها: مسائل NP-Complete، رنگ‌آمیزی گراف، شبکه‌ی شاره، الگوریتم تقریبی.

۱ مقدمه

رنگ‌آمیزی گراف یکی از مسائل معروف در نظریه‌ی گراف‌ها می‌باشد که دارای کاربردهای متعددی از جمله زمان‌بندی، تخصیص ثبات و تخصیص پهنای باند است. این مسئله برای گراف‌های بدون جهت به این ترتیب تعریف می‌شود که هرکدام ازرأس‌های گراف با یک رنگ، رنگ‌آمیزی شود به‌طوری‌که هیچ دو رأس مجاور هم‌رنگ نباشند. این مسئله در حالت بهینه، یعنی رنگ‌آمیزی با حداقل رنگ ممکن، یک مسئله‌ی NP‐سخت است و راه حل چندجمله‌ای برای آن وجود ندارد مگراین‌که P و NP یکی باشند. سختی این مسئله به‌حدی است که حتی رنگ‌آمیزی یک گراف ۳‐رنگ‌پذیر با چهار رنگ نیز NP‐سخت است [۱].

با توجه به قدمت این مسئله و کاربردهای آن، الگوریتم‌ها و روش‌های مختلفی برای حل آن ارائه شده است که نمونه‌هایی از آن‌ها الگوریتم‌های موازی و ترکیبی [۲]، الگوریتم‌های ژنیتیکی تطبیقی[۱] [۳] و الگوریتم‌های تکاملی تطبیقی[۲] [۴] است. هم‌چنین الگوریتم‌های تقریبی متعددی برای این مسئله ارائه شده است [۵][٦][۷]، که در همه‌ی الگوریتم‌های تقریبی ارائه شده برای رنگ‌آمیزی گراف‌ها، ضریب n^ϵ در تعداد رنگ‌های لازم وجود دارد و طبق قضیه‌ی اثبات شده به‌وسیله‌ی Lund و Yannakakis نمی‌توان این ضریب n^ϵ را حذف کرد مگر این‌که P و NP یکی باشند [۸].

این مقاله روی‌کرد جدیدی برای حل این مسئله است که آن را به مسئله‌ی شبکه شاره تبدیل می‌کند. در این روش، برای گراف رنگ‌آمیزی شونده یک شبکه‌ی شاره ساخته می‌شود و با محاسبه‌ی شاره‌ی بیشینه برای آن، حداکثر رأس‌هایی که می‌توانند هم‌رنگ باشند، به‌دست می‌آیند و با تکرار این کار تمام رئوس رنگ‌آمیزی می‌شوند. برای رنگ‌آمیزی صحیح باید در به‌دست آوردن شاره، محدودیت‌هایی درانتخاب یال‌های دارای شاره رعایت شود که با این محدودیت‌ها محاسبه‌ی شاره‌ی بیشینه با روش‌های معمول آن ممکن نیست و در صورت وجود راه‌حل چندجمله‌ای برای آن باید P و NP یکی باشند. به همین دلیل در محاسبه‌ی شاره، باید از الگوریتم‌های تقریبی استفاده شود که در نتیجه یک جواب تقریبی برای رنگ‌آمیزی گراف به‌دست می‌آید.

در ادامه‌ی این مقاله، در قسمت ۲ نحوه‌ی تشکیل شبکه‌ی شاره‌ی متناظر با یک گراف توضیح داده می‌شود. در قسمت ۳ الگوریتم رنگ‌آمیزی گراف براساس شبکه شاره‌ی متناظر با گراف، ارائه می‌شود که در قسمت ۴ درستی آن اثبات و در قسمت ۵ بهینه بودن الگوریتم بررسی می‌شود.

در قسمت ٦ ضریب تقریب الگوریتم رنگ‌آمیزی و در قسمت ۷ پیچیدگی زمانی و حافظه مصرفی آن محاسبه می‌شود. در قسمت ۸ یک روش تقریبی برای حل مسئله‌ی شبکه شاره و در نتیجه روشی تقریبی برای رنگ‌آمیزی گراف می‌آید.

۲ تشکیل شبکه‌ی شاره‌ی متناظر با گراف دلخواه

برای گراف $G(V,E)$ شبکه شاره‌ی N به این ترتیب ایجاد می‌شود:

- در شبکه‌ی شاره رأس s به‌عنوان مبدأ و رأس t به‌عنوان مقصد اضافه می‌شوند.
- دو مجموعه‌ی رئوس LV و RV هرکدام با $n(V)$ رأس با ویژگی‌های زیر به شبکه‌ی شاره اضافه می‌شود:
  ۱. هر رأس $l_i \in LV$ و $r_i \in RV$ متناظر با دقیقاً یک رأس i از V می‌باشد.
  ۲. به‌ازای هر رأس $l_i \in LV$ یک یال (s,l_i) با جهت s به l_i به شبکه‌ی N اضافه می‌شود.
  ۳. به‌ازای هر رأس $r_i \in RV$ یک یال (r_i,t) با جهت r_i به t به N اضافه می‌شود.
- به‌ازای هر دو رأس i و j متمایز و غیر مجاور در گراف G یک زیرگراف به‌نام NA_{ij} بین دو رأس l_i و r_j و یک زیرگراف به‌نام NA_{ji} بین دو رأس l_j و r_i به N اضافه می‌شود که هر NA_{ij} به‌صورت زیر ساخته می‌شود:

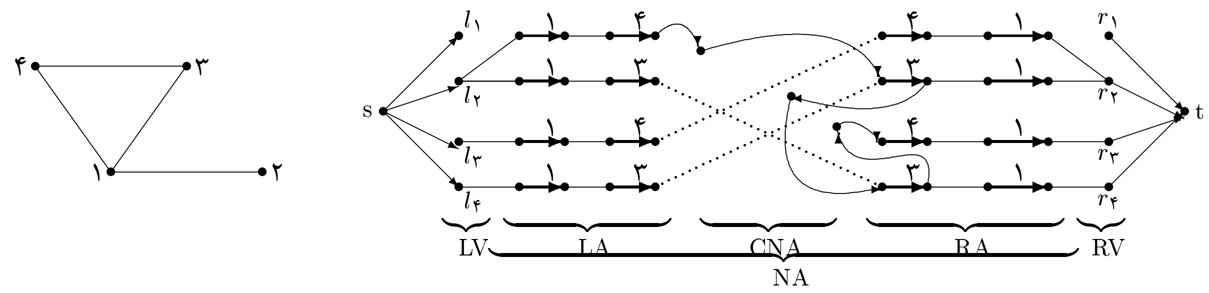
۱. به‌ازای هر رأس m مجاور i یا j در گراف G دو رأس به زیرگراف NA_{ij} اضافه شده و با یالی با برچسب m به‌هم وصل می‌شوند.

۲. همه‌ی جفت رأس‌های متصل به‌همِ به‌دست آمده، به صورت یک مسیر ساده به هم وصل می‌شوند و از این نوع مسیر دو مورد به زیرگراف NA_{ij} اضافه می‌شود که یکی RA_{ij} و دیگری LA_{ij} نامیده می‌شود. این دو زیرگراف با یک زیرگراف دیگر به‌نام CNA_{ij} به هم وصل می‌شوند که به‌صورت زیر ساخته می‌شود:

— زیرگراف نوع CNA_{ij} پس از ساختن تمام زیرگراف‌های نوع LA و RA ساخته می‌شود.

— به‌ازای هر یال با برچسب i در زیرگراف‌های نوع LA و هر یال با برچسب j در زیرگراف‌های نوع RA یک یال از رأسی از این زیرگراف به آن یال وصل شده و از انتهای آن یال یک یال به رأسی از زیرگراف CNA_{ij} وصل می‌شود. این دو یال موازی هم گفته می‌شوند.

---
[۱] Adaptive Genetic
[۲] Adaptive Evolutionary



شکل ۱: یک گراف و بخشی از شبکه‌ی شاره‌ی متناظر با آن

— — از هر رأس از زیرگراف CNA یک یال خارج شده و یک یال به آن وارد می‌شود.

۳. یک یال از l_i به ابتدای مسیر گراف LA_{ij} و یک یال از انتهای مسیر گراف RA_{ij} به r_j وصل می‌شود.

۴. انتهای مسیر زیرگراف LA_{ij} به ابتدای مسیر زیرگراف CNA_{ij} و انتهای این زیرگراف به ابتدای زیرگراف RA_{ij} وصل می‌شود.

- ظرفیت تمام یال‌های شبکه شاره «۱» می‌باشد.

در شکل ۱، یک گراف و شبکه‌ی شاره‌ی متناظر با آن نشان داده شده است. دراین شکل ظرفیت همه یال‌ها برابر «۱» است و اعداد نشان داده شده روی برخی از یال‌ها، برچسب مربوط به آن یال می‌باشد. همچنین برای خوانا بودن شکل فقط یکی از زیرگراف‌های نوع NA نشان داده شده است و سه‌تای دیگر به صورت نقطه‌چین آمده‌اند.

۳ الگوریتم رنگ‌آمیزی

برای رنگ‌آمیزی گراف دلخواه $G(V, E)$ به‌ترتیب زیر عمل می‌شود:

- شبکه‌ی شاره‌ی متناظر با گراف G ساخته می‌شود.
- شاره‌ی عبوری بیشینه از شبکه با این محدودیت که اگر از یک یال زیرگراف CNA شاره عبور کند از یال موازی آن نیز عبور کند، محاسبه می‌شود.
- رأس‌هایی از گراف G که از رئوس متناظر آن‌ها در شبکه، شاره عبور می‌کند رئوسی هستند که می‌توانند هم‌رنگ باشند. با یک رنگ، رنگ‌آمیزی شده و از گراف حذف می‌شوند.
- مراحل بالا تا هنگامی که شاره‌ی شبکه بیشتر از صفر است تکرار می‌شود.
- در گراف باقی‌مانده تمام رئوس مجاور هم هستند و هر کدام با یک رنگ جدید رنگ‌آمیزی می‌شوند.

در قسمت‌های بعدی درستی الگوریتم رنگ‌آمیزی ارائه شده توضیح داده می‌شود و تعداد رنگ‌های مورد نیاز و پیچیدگی زمانی و حافظه‌ی مصرفی آن بیان می‌شود.

۴ درستی الگوریتم

برای اثبات درستی الگوریتم دو موضوع را باید بررسی کرد:

۱. الگوریتم خاتمه‌پذیر است.
با توجه به این‌که در هر مرحله تکرار الگوریتم تعدادی از رئوس گراف حذف می‌شوند و نیز تعداد رئوس گراف متناهی است پس الگوریتم خاتمه‌پذیر است.

۲. مجموعه رئوسی که در هر مرحله‌ی تکرار الگوریتم به‌دست می‌آیند و با یک رنگ رنگ‌آمیزی می‌شوند در گراف G مجاور نمی‌باشند.
باتوجه به نحوه‌ی ساخت شبکه و نیز محدودیت درنظر گرفته شده هنگام عبور شاره، این مسئله نیز برقرار است و از قضیه‌ی ۱ به سادگی نتیجه می‌شود:

لم ۱ . هنگام عبور شار بیشینه، اگر از رأس $r_i(l_i)$ در شبکه شاره، شار عبور کند آنگاه از رأس $l_i(r_i)$ نیز شار عبور خواهد کرد.

اثبات. با توجه به این‌که ظرفیت همه‌ی یال‌ها در شبکه شاره «۱» است پس تعداد رأس‌های l_i و r_i که شاره از آن‌ها عبور می‌کند برابر است و اگر از مسیر $s \mapsto r_i \mapsto^* l_j \mapsto t$ شاره عبور کند در این‌صورت از مسیر $s \mapsto l_i \mapsto^* r_j \mapsto t$ نیز می‌تواند شاره عبور کند و چون در محاسبه‌ی شاره‌ی بیشینه به‌ازای هر مسیر از این نوع، یک واحد شاره‌ی عبوری شبکه اضافه می‌شود، پس هنگام به‌دست آوردن شاره‌ی بیشینه چنان‌چه رأس r_i در شبکه‌ی شاره دارای شاره باشد آن‌گاه رأس l_i نیز دارای شاره می‌باشد.
اثبات حالت برعکس آن نیز به همین ترتیب می‌باشد. □

قضیه‌ی ۱ . هنگام عبور شار بیشینه، رأس‌های متناظر با رئوس l_i یا r_i از شبکه شاره‌ی متناظر با گراف G که از آنها شار عبور می‌کند، مجاور هم نمی‌باشند.

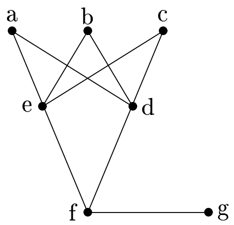
اثبات. فرض کنید دو رأس مجاور i و j در گراف G وجود دارد که رأس‌های l_i، l_j، r_i و r_j متناظر با آنها دارای شار باشند. در این صورت یک مسیر به‌صورت $s \mapsto r_i \mapsto^* l_k \mapsto t$ وجود دارد که مانع عبور شار از تمام مسیرهای $s \mapsto r_j \mapsto^* t$ می‌شود زیرا در تمام این مسیرها یالی با برچسب i وجود دارد که در مسیر $s \mapsto r_i \mapsto^* l_k \mapsto t$ نیز وجود دارد. پس از r_j شار عبور نمی‌کند و با توجه به لم ۱ از l_j نیز شار عبور نمی‌کند و این خلاف فرض است. □

۵ بهینه‌بودن الگوریتم

در الگوریتم ارائه شده در هر مرحله از تکرار الگوریتم حداکثر تعداد رأس‌هایی که می‌توانند هم‌رنگ باشند به‌دست می‌آید که علت آن نیز این است که در شبکه‌ی شاره‌ی N به‌ازای هر مسیر $s \mapsto r_i \mapsto^* l_j \mapsto t$ مقدار شار شبکه یک واحد افزایش پیدا می‌کند و چون هر رأس r_i یا l_i حداکثر یک بار می‌تواند در این نوع مسیرها قرار بگیرد، پس برای بیشینه شدن شاره باید تعداد این رأس‌ها ماگزیمم باشد و چون از هر مسیر $s \mapsto r_i \mapsto^* l_j \mapsto t$ در صورتی‌که r_i و l_j مجاور هیچ‌کدام از رأس‌های دارای شار نباشند، می‌تواند شاره عبور کند پس با به‌دست آوردن شاره‌ی ماکزیمم، حداکثر تعداد رأس‌های غیرمجاور که می‌توانند هم‌رنگ باشند به‌دست می‌آید.

موضوعی که در این‌جا مطرح است این است که اگر در هر مرحله حداکثر تعداد رأس‌های ممکن با یک رنگ، رنگ‌آمیزی می‌شوند پس رنگ‌آمیزی بهینه است. این مسئله درست نیست زیرا در شبکه‌ی شاره در هر مرحله ممکن است چندین جواب وجود داشته باشد که شبکه‌ی شاره فقط یکی از آن‌ها را به‌دست می‌آورد که ممکن است غیر از آن چیزی باشد که در رنگ‌آمیزی بهینه استفاده می‌شود. این مسئله در مثال زیر مشخص شده است.

در گرافی که در شکل ۲ نشان داده شده است باانتخاب $\{a, b, c, f\}$ و $\{d, e, g\}$ به عنوان رأس‌های هم‌رنگ، می‌توان با دو رنگ، گراف را به‌صورت بهینه رنگ‌آمیزی کرد در حالی‌که بااستفاده از الگوریتم ارائه‌شده، هر کدام از مجموعه‌های $\{a, b, c, f\}$ و $\{a, b, c, g\}$ می‌توانند در مرحله‌ی اول هم‌رنگ انتخاب شوند که باانتخاب آخری، حداقل دو رنگ دیگر برای رنگ‌آمیزی رأس‌های $\{f, d, e\}$ لازم است. پس الگوریتم بهینه نیست.



الگوریتم بهینه : { (a,b,c,f),(d,e,g)}

الگوریتم جدید : {(a,b,c,f),(d,e,g)} {(a,b,c,g),(e,d),(f)}

شکل ۲: مثالی برای بهینه نبودن الگوریتم.

۶ ضریب تقریب الگوریتم

یک حد بالا برای تعداد رنگ‌های بکار رفته در الگوریتم ارائه شده، برای رنگ‌آمیزی گراف k−رنگ‌پذیر G برابر است با $k+m$ که m برابر است با تعداد تکرار مراحل الگوریتم به‌طوری‌که پس از مرحله‌ی mام تعداد رأس‌های باقیمانده گراف، کوچکتر یا مساوی k باشد. زیرا در این صورت آن‌را می‌توان با k رنگ، رنگ‌آمیزی کرد و چون در هر بارِ تکرار الگوریتم، یک رنگ مصرف می‌شود، حداکثر تعداد رنگ‌های لازم $k+m$ می‌باشد.

برای محاسبه‌ی m، گراف k−رنگ‌پذیر G را در نظر بگیرید. با این فرض که تعداد رأس‌های گراف بزرگتر یا مساوی k است، در این گراف حداقل یک رنگ وجود دارد که تعداد رأس‌های رنگ‌آمیزی شده با آن، بزرگتر یا مساوی $\lceil \frac{n}{k} \rceil$ باشد زیرا در غیر این‌صورت همه‌ی رأس‌های گراف رنگ‌آمیزی نخواهند شد.

با توجه به مطلب بالا و نیز با توجه به این‌که شبکه‌ی شاره‌ی متناظر با گراف، ماگزیمم تعداد رأس‌هایی را که می‌توانند هم‌رنگ باشند را به‌دست می‌آورد، پس تعداد رأس‌هایی که توسط شبکه‌ی شاره‌ی متناظر با گراف، به عنوان رأس‌های هم‌رنگ به‌دست می‌آیند نیز بزرگتر یا مساوی $\lceil \frac{n}{k} \rceil$ است.

هم‌چنین، چون با حذف کردن یک سری رأس از یک گراف k−رنگ‌پذیر، گراف جدید نیز k−رنگ‌پذیر می‌باشد، پس در هر مرحله از الگوریتم، تعداد رأس‌های به‌دست آمده به‌عنوان رأس‌های هم‌رنگ، بزرگتر یا مساوی $\lceil \frac{n(G_i)}{k} \rceil$ می‌باشد که G_i تعداد رأس‌های باقی‌مانده از گراف G در آغاز مرحله‌ی iام اعمال الگوریتم می‌باشد.

برای به‌دست آوردن مقدار G_i از قضیه‌ی زیر کمک می‌گیریم :

قضیه‌ی ۲ . برای گراف k−رنگ‌پذیر G در آغاز مرحله‌ی iام از تکرار الگوریتم رنگ آمیزی ارائه شده، رابطه‌ی زیر در مورد تعداد رأس‌های آن برقرار است : $n(G_i) > k \Longrightarrow n(G_i) \leq \frac{n(k-1)^{i-1}}{k^{i-1}}$

اثبات. با استقرا روی i این مطلب را نشان می‌دهیم:

۱. به ازای $i=1$ که اولین مرحله تکرار الگوریتم است $G_1 = G$ که تعداد رأس‌های آن برابر n است و در رابطه صدق می‌کند.

۲. اگر $n(G_i) \leq \frac{n(k-1)^{i-1}}{k^{i-1}}$ و $n(G_i) > k$ باشد در این‌صورت در این مرحله نیز چون گراف k−رنگ‌پذیر است، پس حداقل تعداد رأس‌های هم‌رنگِ به‌دست آمده توسط الگوریتم، $\lceil \frac{n(G_i)}{k} \rceil$ می‌باشد که این تعداد رأس از گراف حذف می‌شوند و گراف G_{i+1} به‌دست می‌آید. پس:

$$\begin{aligned} n(G_{i+1}) &\leq n(G_i) - \lceil \frac{n(G_i)}{k} \rceil \\ \Rightarrow &\leq n(G_i) - \frac{n(G_i)}{k} \\ \Rightarrow &\leq \frac{n(G_i)(k-1)}{k} \\ \Rightarrow &\leq \frac{n(k-1)^{i-1}}{k^{i-1}} \times \frac{k-1}{k} \\ \Rightarrow &\leq \frac{n(k-1)^{i}}{k^{i}} \end{aligned}$$

با توجه به ۱ و ۲ اثبات کامل است. □

با توجه به قضیه‌ی ۲ و مطالب گفته شده در ابتدای این بخش مقدار m حداقل مقداری است که به ازای آن رابطه‌ی $\frac{n(k-1)^m}{k^m} \leq k$ برقرار است. پس:

$$\begin{aligned}&n(k-1)^m \leq k^{m+1}\\&\Rightarrow \log n + m\log(k-1) \leq (m+1)\log k\\&\Rightarrow m(\log(k-1) - \log k) \leq \log k - \log n\\&\Rightarrow m \geq \frac{\log k - \log n}{\log(k-1) - \log k}\\&\Rightarrow m \geq \frac{\log\frac{n}{k}}{\log\frac{k}{k-1}}\end{aligned}$$

و چون ما حداقل مقدارصحیح m را لازم داریم پس مقدار آن را برابر با $\left\lceil \frac{\log\frac{n}{k}}{\log\frac{k}{k-1}} \right\rceil$ می‌گیریم. پس حداکثر تعداد رنگ‌های لازم $k + \left\lceil \frac{\log\frac{n}{k}}{\log\frac{k}{k-1}} \right\rceil$ است.

۷ پیچیدگی زمانی و حافظه مصرفی الگوریتم

برای بیان پیچیدگی زمانی الگوریتم ابتدا مرتبه‌ی تعداد رأس‌های شبکه‌ی شاره‌ی متناظر با گراف را به‌دست می‌آوریم و سپس نشان می‌دهیم که تشکیل شبکه‌ی شاره در زمان چندجمله‌ای صورت می‌گیرد و پس از آن مرتبه زمانی اجرای الگوریتم را به‌دست می‌آوریم.

لم ۲. تعداد رأس‌های شبکه‌ی شاره متناظر با گراف n رأسی G،$O(n^5)$ می‌باشد.

اثبات. در شبکه شاره ۲n رأس برای مجموعه رئوس LV و RV و ۲ رأس برای s و t لازم است. همچنین O(n) رأس در هر زیرگراف از نوع RA و LA لازم است و چو ن تعداد مسیرهای از نوع $l_i \mapsto r_j$، $O(n^2)$ است پس $O(n^2 * n)$ رأس در زیرگراف‌های نوع CNA لازم است و تعداد زیرگراف‌های NA نیز $O(n^2)$ یعنی برابر با تعداد مسیرهای از نوع $l_i \mapsto r_j$ می‌باشد. پس در مجموع تعداد کل رئوس گراف $O(n^5)$ می‌باشد. □

نتیجه‌ی . تعداد یال‌های شبکه‌ی شاره متناظر با گراف n رأسی G، $O(n^5)$ می‌باشد.

قضیه‌ی ۳. شبکه‌ی شاره متناظر با گراف n رأسی G در مرتبه‌ی زمانی چندجمله‌ای قابل ساخت است.

اثبات. مرتبه‌ی زمانی تشکیل شبکه‌ی شاره شامل $O(n)$ برای ساخت شبکه، بجز زیرگراف‌های NA می‌باشد. همچنین $O(n^2)$ برای یافتن رئوس غیر مجاور و برای هرکدام از آنها $O(n)$ برای ساخت زیرگراف‌های نوع LA و RA می‌باشد. تولید زیرگراف‌های نوع CNA نیز مستلزم صرف هزینه‌ی $O(n^3)$ برای پیمایش شبکه به ازای هرکدام از زیرگراف‌های نوع CNA می‌باشد که در مجموع، مرتبه زمانی آن $O(n^5)$ می‌باشد. □

قضیه‌ی ۴. با فرض چندجمله‌ای بودن الگوریتم پیدا کردن شار بیشینه با محدودیت گفته شده برای مسئله، الگوریتم ارائه شده دارای پیچیدگی زمانی چندجمله‌ای می‌باشد.

اثبات. در الگوریتم ارائه شده هزینه‌ی زمانی در هر مرحله از تکرار، شامل $O(n^5)$ برای تشکیل شبکه شاره متناظر با گراف، بعلاوه‌ی هزینه‌ی محاسبه‌ی شار بیشینه و O(n) برای حذف رأس‌های رنگ‌آمیزی شده می‌باشد، و چون حداکثر تعداد دفعات تکرار الگوریتم $\left\lceil \frac{\log\frac{n}{k}}{\log\frac{k}{k-1}} \right\rceil$ است پس مرتبه زمانی اجرای الگوریتم چندجمله‌ای می‌باشد. □

در صورت صحیح بودن فرض صورت گرفته در قضیه ۴ و با توجه به قضیه‌ی اثبات شده توسط Lund و Yannakakis (ضریب n^{ϵ} در راه‌حل‌های تقریبی برای مسئله‌ی رنگ‌آمیزی حذف نمی‌شود مگراین‌که P و NP یکسان باشند)[۸] باید P و NP برابر باشند. بنابراین خود مسئله‌ی به‌دست آوردن شار بیشینه با محدودیت ذکر شده یک مسئله‌ی NP—تمام می‌باشد که در قسمت بعد یک راه‌حل تقریبی برای آن ارائه می‌شود تا به کمک آن مسئله‌ی رنگ‌آمیزی را به‌صورت تقریبی حل کرد.

۸ راه حل تقریبی برای مسألهٔ‌ی شبکه‌ی شاره، با ویژگی ذکر شده

راه حلی که در اینجا ارائه می‌شود روشی حریصانه برای به‌دست آوردن شار بیشینه می‌باشد که در آن امکان برگشت شاره(Flow Cancellation) وجود ندارد و به این ترتیب است که برای شبکه‌ی شاره‌ی N در انتخاب مسیرهای افزایشی مطابق الگوریتم Ford-Fulkerson [۹]، مسیر افزایشی که محدودیت گفته شده در آن صدق می‌کند و کمترین تعداد یالهای موازی را دارد انتخاب می‌شود و شار از آن عبور داده می‌شود.

با این روش در هر مرحله جفت رأس‌هایی که تعداد رأس‌های مجاور آنها کمتر است انتخاب می‌شوند تا امکان رنگ‌آمیزی تعداد بیشتری از رأس‌ها وجود داشته باشد و ضریب تقریب آن بستگی به نوع گراف دارد و به‌دست آوردن آن کار مشکلی است ولی موضوعی که مشخص است این است که کارایی آن حداقل برابر الگوریتم حریصانه‌ی رنگ‌آمیزی گراف [٦] می‌باشد.

۹ خلاصه و نتیجه‌گیری

در این مقاله روش جدیدی برای حل مسئله‌ی رنگ‌آمیزی گراف ارائه شده است که منجر به مسئله‌ی NP—تمام جدیدی یعنی محاسبه‌ی شار بیشینه در یک شبکه‌ی شاره با قراردادن محدودیت‌هایی در انتخاب یال‌های دارای شار می‌باشد. در راه‌حل جدید، متناظر با گراف رنگ‌آمیزی شونده یک شبکه‌ی شاره ساخته می‌شود و با به‌دست آوردن شار بیشینه، با درنظرگرفتن یک محدودیت برای آن، حداکثر تعداد راس‌هایی که می‌توانند هم‌رنگ باشند به‌دست می‌آید و با تکرار این کار تمام رأس‌های گراف رنگ‌آمیزی می‌شود. ولی با توجه به این‌که مسئله‌ی محاسبه‌ی شار بیشینه برای شبکه‌ی حاصل خود یک مسئله NP—تمام است برای حل آن باید از راه‌حل‌های تقریبی استفاده کرد که در قسمت ۸ یک راه‌حل حریصانه برای آن ارائه شده است.

در این مقاله روش برخورد جدیدی با مسئله رنگ‌آمیزی گراف و مسائل مشابه از جمله مجموعه‌ی مستقل رأسی و پوشش رأسی ارائه شده است که در صورت ارائه راه‌حل‌های تقریبی بهتر و با ضریب تقریب کمتر برای مسئله‌ی شبکه‌ی شاره‌ی به‌دست آمده، می‌توان راه‌حل‌های مشابهی برای این مسائل به‌دست آورد.

مراجع

[1] V. Guruswami and S. Kanna, "On the hardness of 4-coloring a 3-colorable graph", Proceeding of the 15th Annual IEEE Conference on Computational Complexity, July 2000.

[2] G. Lewandowski and A. Condon, "Experiments with Parallel Graph Coloring Heuristics and Applications of Graph Coloring", Second DIMACS Implementation Challenge, 1993.

[3] A. E. Eiben and J. K. van der Hauw, "Graph Coloring with Adaptive Genetic Algorithms", `http://www.wi.leidenuniv.nl/TechRep/tr96-11.html`.

[4] A. E. Eiben and J. K. van der Hauw and J. I. van Hemert, "Graph Coloring with Adaptive Evolutionary Algorithms", `http://www.wi.leidenuniv.nl/ğusz/graphcol.ps.gz`, 1996.

[5] A. Blum and D. R. Karger, ”An $\tilde{\mathrm{O}}(n^{۳/۱۴})$-coloring algorithm for 3-colorable graphs”, Information Processing Letters, 61(1):49-53, 1997.

[6] A. Blum, ”New approximation algorithms for graph coloring”, JACM 31(3):470-516, 1994.

[7] D. Karger and R. Motwani and M. Sudan, ”Approximate Graph Coloring by Semidefinite Programming”, In proceedings of the 35th Annual Symposium on Foundations of Computer Sience, November 1994.

[8] C. Lund and M. Yannakakis, ”On the hardness of approximating minimization problems”, Unpublished manuscript, 1992.

[9] Thomas H. Cormen and Charles E. Leiserson and Ronald L. Rivest, ”Introduction to Algorithms”, The MIT Press, 1990.