مجله علمی- پژوهشی "پدافند الکترونیکی و سایبری"
سال ششم، شماره ۲، تابستان ۱۳۹۷، ص ۴۸-۳۹

رنگ‌آمیزی گروندی خود تثبیت‌کننده با استفاده از نظریه بازی‌ها و یافتار مرتب‌سازی

حسن حیدری[۱]، سید محمود طاهری[۲*]

۱- کارشناس ارشد مهندسی کامپیوتر، ۲- استاد، دانشگاه تهران، دانشکده فنی، گروه علوم مهندسی

(دریافت: ۹۶/۰۲/۰۱، پذیرش: ۹۶/۰۷/۱۲)

چکیده

خرابی گذرا در سیستم‌های توزیع‌شده در شرایط مختلفی مانند خرابی پردازه‌ها و حمله‌های امنیتی رخ می‌دهد. یک الگوریتم خود تثبیت‌کننده با شروع از هر حالت دلخواه، در زمان متناهی به حالت قانونی می‌رسد و در مقابل خرابی گذرا مقاوم است. در این مقاله، نخست، برای مسئلهٔ رنگ‌آمیزی گروندی، اولین الگوریتم قطعی خود تثبیت‌کننده مبتنی بر نظریه بازی‌ها را ارائه می‌کنیم. در این الگوریتم، که از قابلیت اجرا روی شبکه‌های ناشناس برخوردار است، برای کاهش تعداد رنگ‌های مصرفی، از یافتارهای مرتب‌سازی استفاده می‌کنیم. با به‌کارگیری تعادل نش، ثابت می‌کنیم که الگوریتم روی شبح مرکزی با پیچیدگی زمانی $O(m)$ به رنگ‌آمیزی گروندی همگرا می‌شود که در آن m تعداد یال‌های شبکه است. نتایج شبیه‌سازی روی شبکه‌های مستقل از مقیاس، شبکه‌های تصادفی و شبکه‌های دنیای کوچک حاکی از آن است که به‌کارگیری یافتارهای مرتب‌سازی نسبت به عدم استفاده از آن‌ها موجب کاهش تعداد رنگ‌ها تا ۱۸٪ و بهبود سرعت همگرایی به جواب تا ۵٪ می‌گردد.

واژه‌های کلیدی: خرابی گذرا، امنیت شبکه، شبح مرکزی، تعادل نش، سیستم توزیع‌شده ناشناس

۱- مقدمه

در چند دهه اخیر، سیستم‌های توزیع‌شده کاربردهای وسیعی در حوزه‌های ارتباطات راه دور، پردازش اطلاعات توزیع‌شده، محاسبات علمی، کنترل بلادرنگ، شهر هوشمند و اینترنت اشیا یافته‌اند [۳-۱] . خرابی‌های مختلفی مانند خرابی پردازه‌ها، خرابی حافظه اشتراکی، خرابی در پیام‌های ارسالی- دریافتی و خرابی گذرا[۱] در این سیستم‌ها روی می‌دهند که ممکن است به‌صورت غیرعمدی یا عمدی و در اثر حملات امنیتی (توزیع‌شده و غیر توزیع‌شده) روی دهند. برای مقابله با خرابی سیستم‌های توزیع‌شده و به بیان دیگر، ساخت سیستم‌های توزیع‌شده‌ای که قابلیت تحمل‌پذیری خطا را دارند، از الگوریتم‌های خود تثبیت‌کننده استفاده می‌شود [۶-۴].

الگوریتم‌های خود تثبیت‌کننده در سال ۱۹۷۴ توسط دکسترا و برای مقابله با خرابی گذرا معرفی شدند [۷] . این الگوریتم‌ها تضمین می‌کنند که با شروع از هر حالت دلخواهی و در زمان متناهی به حالت قانونی[۲] می‌رسند و پس از رسیدن به حالت قانونی در آن باقی می‌مانند. تاکنون الگوریتم‌های خودتثبیت‌کننده در بسیاری از مسائل توزیع‌شده مانند ساخت درخت پوشا و جستجوی اول عمق [۸]، جورسازی حداکثری[۳] [۹]، پیش‌بینی فاصله در شبکه [۱۰]، انتخاب رهبر [۱۱] و تفکیک[۴] شبکه [۱۲] مورد استفاده قرار گرفته‌اند.

از سوی دیگر، مسئلهٔ رنگ‌آمیزی رأسی در مسائل مهمی از سیستم‌های توزیع‌شده مانند زمان‌بندی [۱۴-۱۳]، تخصیص منابع [۱۶-۱۵]، از بین بردن تقارن[۵] [۳ و ۷] و امنیت شبکه [۱۹-۱۸] کاربرد دارد. برای مسئلهٔ رنگ‌آمیزی رأسی، الگوریتم خود تثبیت‌کننده نیز ارائه شده است. گرادینارو و تیگزویل در سال ۲۰۰۰ برای مسئلهٔ رنگ‌آمیزی رأسی روی گراف‌های بدون جهت دلخواه سه الگوریتم خودتثبت‌کننده ارائه نمودند [۲۰]. الگوریتم اول،

* رایانامه نویسنده مسئول: sm_taheri@ut.ac.ir

1- Transient Fault
2- Legitimate
3- Maximal Matching
4- Disjunction
5- Symmetry Breaking

قطعی است و روی شبح مرکزی[1] با پیچیدگی زمانی $O(n\Delta)$ قابل اجراست. همچنین این الگوریتم قابلیت اجرا روی شبکه‌های ناشناس[2] را داراست. الگوریتم‌های دوم و سوم روی شبح توزیع‌شده قابل اجرا هستند و به‌ترتیب دارای پیچیدگی زمانی $O(n\Delta)$ و $O(logn\,\Delta)$ می‌باشند. الگوریتم‌های دیگری نیز برای مسئلهٔ رنگ‌آمیزی رأسی خودتثبیت‌کننده روی گراف‌های دلخواه ارائه شده است که این الگوریتم‌ها به همراه ویژگی‌های بارز آن‌ها در جدول (۱) آورده شده است. گفتنی است مسئلهٔ رنگ‌آمیزی رأسی خودتثبیت‌کننده، علاوه بر گراف‌های دلخواه روی گونه‌های دیگری از گراف‌ها مانند گراف‌های مسطح، حلقه و گراف‌های دو بخشی نیز بررسی شده است [۴].

جدول (۱): الگوریتم‌های رنگ‌آمیزی رأسی خود تثبیت‌کننده روی گراف‌های دلخواه

مرجع	نوع گراف	شبح	قطعی	ناشناس	گروندی	پیچیدگی زمانی	نوع
Gradinariu and Tixeuil [20]-1	بدون جهت	مرکزی	بله	بله	خیر	$O(n\Delta)$	خود تثبیت‌کننده
Gradinariu and Tixeuil [20]-2	بدون جهت	توزیع‌شده	بله	خیر	خیر	$O(n\Delta)$	خود تثبیت‌کننده
Gradinariu and Tixeuil [20]-3	بدون جهت	توزیع‌شده	خیر	بله	خیر	$O(logn\,\Delta)$	تثبیت‌کننده احتمالاتی
Hedetniemi et al. [28]-1	بدون جهت	مرکزی	بله	بله	بله	$O(m)$	خود تثبیت‌کننده
Hedetniemi et al. [28]-2	بدون جهت	مرکزی	بله	خیر	خیر	$O(n)$	خود تثبیت‌کننده
Goddard et al. [29]	بدون جهت	مرکزی	بله	بله	خیر	مشخص نشده	خود تثبیت‌کننده
Bernard et al. [30]	جهت‌دار	مرکزی	بله	بله	خیر	$O(n^2)$	خود تثبیت‌کننده
Bernard et al. [31]	جهت‌دار	مرکزی	خیر	بله	خیر	$O(n\Delta)$	تثبیت‌کننده احتمالاتی
Kosowski and Kuszner [32]-1	بدون جهت	مرکزی	بله	بله	خیر	$O(mn^3\mathrm{diam(G)})$	خود تثبیت‌کننده
Kosowski and Kuszner [32]-2	بدون جهت	توزیع‌شده	بله	خیر	خیر	$O(mn^3\Delta)$	خود تثبیت‌کننده
این مقاله	بدون جهت	مرکزی	بله	بله	بله	$O(m)$	خود تثبیت‌کننده

الگوریتم‌های خودتثبیت‌کننده، حالت تعمیم‌یافته الگوریتم‌های شبکه‌های پویا (شبکه‌هایی که تعداد گره‌ها و یال‌های آن در طول اجرای الگوریتم تغییر می‌یابند) هستند. در مقاله [۲۱]، علی‌رغم آن‌که عنوان مقاله ارائه یک الگوریتم خود تثبیت‌کننده برای مسئلهٔ رنگ‌آمیزی گروندی است، یک الگوریتم رنگ‌آمیزی گروندی برای شبکه‌های پویا ارائه می‌کند.

مسئلهٔ رنگ‌آمیزی رأسی توسط نظریه بازی‌ها نیز بررسی و حل شده است [۲۳-۲۲]. همچنین الـگـوریتم‌های خودتثبیت‌کننده مبتنی بر نظریه بازی‌ها، برای یافتن مجموعه غالب مینیمال[3] [۶] و یافتن مجموعه مستقل حداکثری و یافتن مجموعه مستقل وزن‌دار حداکثری [۵] ارائه شده‌اند.

در این مقاله برای مسئلهٔ رنگ‌آمیزی گروندی، یک الگوریتم قطعی خودتثبیت‌کننده و مبتنی بر نظریه بازی‌ها، ارائه می‌کنیم. گفتنی است که، تا جایی‌که نویسندگان اطلاع دارند، تاکنون چنین الگوریتمی ارائه نشده است. با استفاده از تعادل نش، ثابت می‌کنیم که پیچیدگی الگوریتم پیشنهادی روی شبح مرکزی از مرتبه $O(m)$

1- Central Daemon
2- Anonymous
3- Minimal Dominating Set

است. لازم به ذکر است که الگوریتم پیشنهادی قابلیت اجرا روی شبکه‌های ناشناس را داراست. سرعت همگرایی و کیفیت رنگ‌آمیزی (تعداد رنگ‌های استفاده‌شده) الگوریتم پیشنهادی را از طریق شبیه‌سازی آن روی شبکه‌های تصادفی [۲۴]، شبکه‌های دنیای کوچک [۲۵] و شبکه‌های مستقل از مقیاس [۲۶] بررسی می‌کنیم.

الگوریتمی که در این مقاله ارائه می‌کنیم شباهت زیادی به الگوریتم رنگ‌آمیزی حریصانه [۲۷] دارد. برای بهبود کیفیت رنگ‌آمیزی حریصانه از یافتارهای مرتب‌سازی[۱] مختلفی مانند *LF*، *LLF*، *Random*، SD و SL استفاده می‌کنیم [۳۳]. در این مقاله، با به کارگیری یافتارهای مرتب‌سازی نامبرده، کیفیت رنگ‌آمیزی را افزایش می‌دهیم. نتایج شبیه‌سازی بیانگر آن هستند که دو یافتار مرتب‌سازی SD و LF از کیفیت رنگ‌آمیزی بالاتری نسبت به سایر یافتارهای مرتب‌سازی برخوردار هستند. لازم به ذکر است که الگوریتم پیشنهادی با الگوریتم هدنیمی و همکاران [۲۸] از نظر مرتبه پیچیدگی زمانی یکسان است ولی الگوریتم پیشنهادی، مبتنی بر نظریه بازی‌ها و یافتارهای مرتب‌سازی است و نسبت به الگوریتم هدنیمی و همکاران به‌طور متوسط تعداد رنگ‌ها را تا ۱۸٪ کاهش داده و زمان همگرایی به جواب (زمان اجرای الگوریتم) را نیز تا ۵٪ بهبود داده است.

در ادامه، در بخش ۲، تعاریف و مفاهیم مقدماتی را درباره سیستم‌های توزیع‌شده، الگوریتم‌های خودتثبیت‌کننده، مسئلهٔ رنگ‌آمیزی گروندی، نظریه بازی‌ها و یافتارهای مرتب‌سازی ارائه می‌کنیم. در بخش ۳، با به کارگیری تعادل نش، یک الگوریتم مبتنی بر نظریه بازی‌ها برای مسئلهٔ رنگ‌آمیزی گروندی خود تثبیت‌کننده روی شبح مرکزی ارائه می‌کنیم. در بخش ۴، نتایج شبیه‌سازی را روی چند توپولوژی پرکاربرد- شبکه‌های تصادفی، شبکه‌های دنیای کوچک و شبکه‌های مستقل از مقیاس- آورده‌ایم. در انتها، در بخش ۵ جمع‌بندی و آینده تحقیق را بیان می‌نماییم.

۲- تعاریف و مفاهیم مقدماتی

گیریم $P = \{p_1, p_2, \dots, p_n\}$ مجموعه‌ی تمام پردازه‌ها ($|P| = n$)، و $E \subseteq P \times P$ مجموعه‌ی تمام لینک‌های ارتباطی بین پردازه‌ها ($|E| = m$) در یک سیستم توزیع‌شده باشند. گراف بدون جهت $G = (P, E)$ را که هر کدام از راس‌ها و یال‌های آن به‌ترتیب متناظر با یک پردازه و یک لینک از سیستم توزیع‌شده است در نظر می‌گیریم. بیشینه درجه گراف با Δ، مجموعه رئوس همسایه راس p با $N(p)$ و درجه راس p با $\deg(p)$ نشان داده می‌شوند. سیستم توزیع‌شده دارای مدل محاسباتی *Congest* [۳] است و ارتباط بین واحدهای محاسباتی از طریق حافظه اشتراکی صورت می‌گیرد. هر پردازه p، دسترسی خواندان- نوشتن را روی حافظه محلی خود دارد؛ در حالی‌که تنها دسترسی خواندن حافظه پردازه $q \in N(p)$ را دارد. اگر پردازه‌های یک سیستم توزیع‌شده فاقد شناسه یکتا باشند، آن سیستم توزیع‌شده را ناشناس می‌گوییم.

الگوریتم توزیع‌شده را غالبا در قالب دنباله‌ای از قوانین[۲] یا دستورات محفوظ[۳] بیان می‌کنیم. هر قانون یا دستور محفوظ از دو قسمت، محفوظ و اقدام[۴]، تشکیل می‌شود. محفوظ یک عبارت منطقی است که درصورت درست بودن، اقدام آن اجرا می‌شود. به قانونی که محفوظ آن درست است، قانون فعال‌شده[۵] و به اجرای یک اقدام، تغییر حالت[۶] می‌گوییم. پردازه‌ای را که حداقل یکی از قوانین آن فعال باشد، دارای امتیاز[۷] می‌نامیم. سیستم توزیع‌شده وارد حالت راضی[۸] خواهد شد اگر هیچ پردازه‌ی دارای امتیازی وجود نداشته باشد. به مدل اجرای همزمان[۹] پردازه‌ها، شبح می‌گوییم. برای یک سیستم توزیع‌شده سه نوع شبح- مرکزی، همگام و توزیع‌شده- قابل تصور است. در شبح مرکزی، در هر بازه زمانی دلخواه، تنها یک پردازه دارای امتیاز تغییر حالت می‌یابد. در شبح همگام، در هر دور[۱۰] تمام پردازه‌های دارای امتیاز اجازه تغییر حالت همزمان[۱۱] را دارند. در شبح توزیع‌شده، هر زیر مجموعه ناتهی از پردازه‌های دارای امتیاز اجازه تغییر حالت همزمان را دارند. در شبح توزیع‌شده، هر زیر مجموعه ناتهی از پردازه‌های دارای امتیاز می‌توانند به‌صورت موازی[۱۲] تغییر حالت یابند. در واقع، شبح مرکزی زیرمجموعه شبح همگام و شبح همگام زیرمجموعه شبح توزیع‌شده است.

1- Ordering Heuristic

2- Rules
3- Guarded commands
4- Action
5- Enabled
6- Move
7- Privileged
8- Acquiescent
9- Concurrent
10- Round
11- Simultaneous
12 Parallel

تعریف ۱ (k-رنگ‌آمیزی رأسی سره) k- رنگ‌آمیزی رأسی گراف G عبارت است از تخصیص k رنگ، ۱،۲،...،k، به راس‌های G. رنگ‌آمیزی سره است اگر هیچ دو راس مجاورِ متمایز، دارای یک رنگ نباشند.

تعریف ۲ (رنگ‌آمیزی گروندی) [۳۴] گوییم k-رنگ‌آمیزی رأسی سره، گروندی است اگر به ازای هر راس v که دارای رنگ $i > 1$ است، حداقل یک راس $u \in N(v)$ با رنگ $j < i$ وجود داشته باشد.

گفتنی است که رنگ‌آمیزی گروندی در بدترین حالت، یک $(\Delta + 1)$- رنگ‌آمیزی سره می‌باشد (یک خوشه[1] Δ رأسی که تمام رئوس آن به رأس v که عضو خوشه نیست متصل شده‌اند بدترین حالت را می‌سازد).

الگوریتم توزیع‌شده‌ای را که دارای دو ویژگی همگرایی[2] و بستار[3] باشد، خودتثبیت‌کننده می‌نامیم. خاصیت همگرایی تضمین می‌کند که با شروع از هر حالت غیرقانونی به حالت قانونی می‌رسد (در مسئلهٔ رنگ‌آمیزی رأسی، حالت‌های غیرقانونی، حالت‌هایی هستند که دو پردازه همسایه، دارای رنگ یکسان باشند و حالت‌های قانونی حالتی‌هایی هستند که در آن رنگ‌آمیزی، سره باشد). خاصیت بستار تضمین می‌کند که پس از همگرایی و وقوع حالت قانونی، سیستم در حالت قانونی باقی می‌ماند. به بیان دیگر، پس از همگرایی و تا قبل از رخ دادن یک خطا، سیستم در حالت راضی باقی می‌ماند (هیچ پردازه‌ای دارای امتیاز نمی‌شود). فرض می‌کنیم که در اجرای الگوریتم‌های خودتثبیت‌کننده، هر پردازه دسترسی بدون خطا به حافظه همسایه‌های خود دارد.

برای مدل‌سازی مسئلهٔ رنگ‌آمیزی گروندی، گیریم $Col : P \to \{0,1,\dots,\Delta\}$ یک تابع باشد که به هر یک از رئوس گراف یک رنگ از مجموعه $\{0,1,\dots,\Delta\}$ اختصاص می‌دهد. برای پیدا کردن رنگ‌آمیزی گروندی، بازی $\Gamma = [P; E; Col; \{S_i\}_{i=1}^{n};\ \{u_i\}_{i=1}^{n}]$ را حل می‌کنیم که در آن $S_i = \{0,1,\dots,\Delta\}$ مجموعه راهبردهای بازیکن (راس، پردازه) p_i و u_i مطلوبیت[4] او هستند. ترکیب راهبردی[5] یک nتایی $C = (c_1, c_2, \dots, c_n)$ است که در آن $c_i \in S_i$ راهبرد بازیکن p_i را نشان می‌دهد. برای نمایش ساده‌تر ترکیب راهبردی، راهبرد انتخاب‌شده توسط بازیکن p_i و راهبرد سایر بازیکن‌ها را با دوتایی مرتب $C = (c_i, c_{-i})$ نمایش می‌دهیم. گفتنی است که $Col(p_i) = c_i$.

اگر ارتباط بین بازیکنان توسط یک گراف قابل نمایش باشد (هر بازیکن تنها روی مطلوبیت بازیکنان همسایه خود تاثیر می‌گذارد)، به آن بازی، گرافیکی[6] می‌گوییم. به بازی‌ای که مطلوبیت هر بازیکن با انتخاب راهبردی که توسط سایر همسایه‌ها انتخاب نشده، بیشینه شود، بازی فشرده[7] می‌گوییم. اگر گراف بازی بدون جهت باشد، آن بازی را متقارن[8] می‌نامیم [۳۵].

تعریف ۳ (تعادل نش) ترکیب راهبردی $C^* = (c_i^*, c_{-i})$ را در بازی $\Gamma = [P; \{S_i\}_{i=1}^{n};\ \{u_i\}_{i=1}^{n}]$ یک تعادل نش گوییم اگر به ازای هر $i \in \{1,2,\dots,n\}$ و هر $c_i \in S_i$ رابطه $u_i(c_i^*, c_{-i}) \ge u_i(c_i, c_{-i})$ برقرار باشد.

تعریف ۴ (قانون بهترین-پاسخ[9]) گوییم بازیکن p_i، بر اساس قانون بهترین- پاسخ راهبرد $c_i^* \in S_i$ را انتخاب می‌کند اگر

$$c_i^* = BR_i(C) = \arg\max_{c_i \in S_i} u_i(c_i, c_{-i})$$

گفتنی است که قانون بهترین- پاسخ ایجاب می‌کند که هر بازیکن برای انتخاب راهبرد باید از راهبرد انتخاب‌شده توسط سایر بازیکنان مطلع باشد. همچنین گفتنی است که بازی رنگ‌آمیزی گروندی (Γ) در شرایط بازی فشرده‌ی گرافیکی متقارن صدق می‌کند.

قضیه ۱ [۳۵]. یک بازی فشرده گرافیکی روی یک گراف بدون جهت، یک بازی پتانسیل دقیق[10] است.

قضیه ۲ [۳۶]. در هر بازی پتانسیل دقیق، اگر راهبرد بازیکنان بر اساس قانون بهترین- پاسخ تغییر یابد، بازی به تعادل نش منجر می‌شود.

تعریف ۵ (یافتارهای مرتب‌سازی LF، LLF، $Random$، SL و SD) یک یافتار مرتب‌سازی، ترتیب بررسی رئوس در الگوریتم حریصانه رنگ‌آمیزی رأسی را مشخص می‌کند.

1- Clique
2- Convergence
3- Closure
4- Utility
5- Strategy Profile

6- Graphical Game
7- Congestion Game
8- Symmetric Game
9- Best-Response Rule
10- Exact Potential Game

- LF (Largest degree first): در این روش رئوس به ترتیب نزولی درجه، بررسی می‌شوند.
- LLF (Log largest degree first): در این روش رئوس به ترتیب نزولی $\lceil \log deg \rceil$ بررسی می‌شوند.
- $Random$ (تصادفی): در این روش، رئوس به‌صورت تصادفی بررسی می‌شوند.
- SL (Smallest degree last): در این روش، ابتدا رئوس با درجه کوچک بررسی می‌شوند و پس از بررسی از گراف حذف می‌شوند. در گراف باقیمانده، دوباره رئوس با درجه کوچک بررسی می‌شوند و این روند تا جایی که تمام رئوس بررسی شوند ادامه می‌یابد.
- SD (Saturation degree): رئوس گراف بر اساس ترتیب نزولی تعداد رنگ‌های متمایز همسایه‌های خود، بررسی می‌شوند.

۳- رنگ‌آمیزی گروندی خودتثبیت‌کننده روی شبح مرکزی

در این بخش، یک الگوریتم رنگ‌آمیزی گروندی خودتثبیت‌کننده را روی شبح مرکزی ارائه می‌کنیم. لازم به ذکر است که در این نوع از شبح، پردازه‌ها یکی پس از دیگری تغییر حالت می‌دهند و هیچ دو پردازه‌ی متمایزی، اجازه‌ی تغییر حالت همزمان را ندارند. لذا این الگوریتم، در دسته بازی‌های ترتیبی قرار می‌گیرد. همچنین عملکرد شبح مرکزی مشابه عملکرد یافتار مرتب‌سازی $Random$ می‌باشد.

۳-۱- تعریف عمومی بازی رنگ‌آمیزی گروندی

مطلوبیت بازیکن p_i را به‌ازای ترکیب راهبرد $C = (c_i, c_{-i})$ به‌صورت زیر تعریف می‌نماییم.

$$u_i(C) = -\left(c_i + \sum_{p_j \in N_i} \alpha \mathbb{1}_{\{c_i = c_j\}}\right) \quad (۱)$$

که در آن

$$\mathbb{1}_{\{c_i = c_j\}} = \begin{cases} 1 & c_i = c_j \\ 0 & c_i \neq c_j \end{cases}$$

و $\alpha \geq \Delta + 1$ یک ثابت است. راهبرد اولیه بازیکنان را به‌صورت دلخواه در نظر می‌گیریم. هر بازیکن بر اساس قانون بهترین-پاسخ می‌تواند راهبرد خود را تغییر دهد. به بیان دیگر، بازیکن p_i ($i \in \{1,2,\dots,n\}$)، راهبردی (رنگی) که $u_i(C)$ (تابع مطلوبیت او) را بیشینه کند، انتخاب می‌نماید.

لم ۱. زمانی‌که بازی رنگ‌آمیزی گروندی (Γ)، به تعادل نش می‌رسد، ترکیب راهبردی C، یک رنگ‌آمیزی سره است.

اثبات. برای اثبات، از برهان خلف استفاده می‌کنیم. برای فرض خلف، گیریم زمانی‌که الگوریتم به تعادل نش می‌رسد، رنگ‌آمیزی سره نباشد. پس حداقل دو راس p_i و p_j وجود دارند که $(p_i, p_j) \in E$ و $c_i = c_j$. گیریم $F_i = \{0,1,\dots,\Delta\} - \{c_k | p_k \in N(p_i)\}$ باشد. راس p_i حداکثر دارای درجه Δ است، لذا $|F_i| \geq 1$. به‌بیان دیگر، حداقل یک رنگ وجود دارد که توسط همسایه‌های p_i به کار گرفته نشده است. در بدترین حالت، این رنگ Δ است. اگر p_i رنگ خود را تغییر دهد و رنگ Δ را برگزیند، در این‌صورت مطلوبیت جدید p_i، $u_i(C') = -\Delta$ است (حاصل مجموع‌یاب، صفر می‌شود)، در حالی‌که $u_i(C)$ حداقل $-(\Delta + 1)$ بوده است (حاصل مجموع‌یاب، $-(\Delta + 1)$ است). لازم به ذکر است که پردازه p_j نیز همین شرایط را داراست. به تناقض خواهیم رسید، چون در تعادل نش، پردازه p_i نمی‌تواند با ثابت ماندن راهبرد سایر بازیکنان راهبرد دیگری را که موجب افزایش مطلوبیت او می‌شود انتخاب نماید. لذا فرض خلف باطل است و رنگ‌آمیزی سره است.

قضیه ۳. زمانی‌که بازی رنگ‌آمیزی گروندی (Γ)، به تعادل نش می‌رسد، ترکیب راهبردی C، یک رنگ‌آمیزی گروندی است.

اثبات. طبق لم ۱، رنگ‌آمیزی، سره است. لذا کافی است ثابت نماییم که هر راس دلخواه p_i، کوچکترین عضو (رنگ) مجموعه F_i را انتخاب می‌کند. برای اثبات از برهان خلف استفاده می‌شود. برای فرض خلف، گیریم راهبرد (رنگ) راس p_i، $c_i = k$، و راهبرد $j < k$ توسط هیچ‌کدام از همسایه‌های p_i استفاده نشده باشد. مطلوبیت بازیکن p_i در حالی‌که $c_i = k$ و $C = (c_i, c_{-i})$، برابر $u_i(C) = -k$ اما در حالی‌که $c_i = j$ و $C' = (c_i, c_{-i})$، برابر $u_i(C') = -j$ است. به تناقض خواهیم رسید چون p_i زمانی‌که در تعادل نش است، نمی‌تواند با تغییر راهبرد مطلوبیت خود را افزایش دهد. لذا فرض خلف باطل است و رنگ‌آمیزی گروندی است.

قضیه ۴. حاصل بازی رنگ‌آمیزی گروندی (Γ)، یک رنگ‌آمیزی گروندی خودتثبیت‌کننده است.

اثبات. چون بازی گروندی در شرایط بازی فشرده گرافیکی صدق می‌کند و چون گراف بازی، یک گراف بدون جهت است، لذا طبق قضیه‌های ۱ و ۲، بازی به تعادل نش منجر می‌شود. بر اساس

قضیه ۳، پس از رسیدن به تعادل نش، ترکیب راهبرد C، یک رنگ‌آمیزی گروندی خودتثبیت‌کننده است و اثبات کامل می‌شود.

قضیه ۵. بازی رنگ‌آمیزی گروندی، حداکثر در $O(m)$ تغییر حالت، به تعادل نش می‌رسد.

اثبات. اثبات قضیه مشابه اثبات قضیه ۱ در مرجع [۲۸] است. در اینجا تنها شمایی از اثبات را ارائه می‌کنیم. پس از آن‌که رنگ گره v در دور tام تغییر می‌یابد، ممکن است رنگ آن در دورهای $t+t'$ $(t' \geq 2)$ نیز تغییر یابد. نکته در این است که رنگ یک گره حداکثر یکبار افزایش می‌یابد (افزایش رنگ، تنها در اولین تغییر رنگ ممکن است روی دهد) و در تغییر رنگ‌های دیگر، روند کاهشی را در پیش می‌گیرد. لذا هر گره v، حداکثر $\deg(v)+1$ تغییر رنگ را انجام می‌دهد. بنابراین پیچیدگی الگوریتم از مرتبه $O(m)$ می‌شود.

۳-۲- تبدیل بازی به الگوریتم خودتثبت‌کننده

برای هر پردازه p_i، قوانین زیر را بر اساس قانون بهترین- پاسخ طراحی می‌کنیم.

$$\begin{aligned}
&for\ each\ c \in S_i\\
&\quad guard: \quad c_i \neq c \wedge u_i(c, c_{-i}) > u_i(c_i, c_{-i})\\
&\quad action: \quad c_i = c
\end{aligned}$$

بر پایه قوانین بالا، الگوریتم خودتثبیت‌کننده‌ی زیر را ارائه می‌کنیم.

R_0	$c_i \neq 0 \wedge \forall\, p_j \in N(p_i): c_j \neq 0$ $\rightarrow\ c_i = 0$
R_1	$c_i \neq 1 \wedge \forall\, p_j \in N(p_i): c_j \neq 1 \wedge \exists\, p_k \in N(p_i): c_k = 0$ $\rightarrow\ c_i = 1$
. .	. .
. .	. .
. .	. .
R_i	$c_i \neq i \wedge \forall\, p_j \in N(p_i): c_j \neq i \wedge \min\{c_j \mid p_j \in N(p_i)\} = i-1$ $\rightarrow\ c_i = i$
. .	. .
. .	. .
. .	. .
R_Δ	$c_i \neq \Delta \wedge \forall\, p_j \in N(p_i): c_j \neq \Delta \wedge \min\{c_j \mid p_j \in N(p_i)\}$ $= \Delta - 1$ $\rightarrow\ c_i = \Delta$

۳-۳- اعمال یافتارهای مرتب‌سازی

در یک سیستم توزیع‌شده، هر گره تنها به گره‌های همسایه خود دسترسی مستقیم دارد و دسترسی به گره‌هایی که در فاصله دو و بیشتر از آن قرار دارند هزینه زمانی و پیامی دارد. لذا برای کمینه کردن هزینه استفاده از یافتارهای مرتب‌سازی، هر گره با استفاده از اطلاعات خود و همسایه‌های خود، تابع یافتار مرتب‌سازی را فراخوانی می‌کند. از بین گره‌های همسایه، تنها گره (گره‌هایی) که دارای بیشترین مقدار تابع یافتار مرتب‌سازی باشد (باشند)، دارای امتیاز می‌شود. در صورتی‌که شبح گره‌ای را برای اجرا انتخاب نماید، گره بر اساس قانون بهترین-پاسخ اقدام خود را انتخاب می‌کند. در ادامه الگوریتم دقیق رنگ‌آمیزی گروندی مبتنی بر نظریه بازی‌ها و یافتارهای مرتب‌سازی را ارائه می‌کنیم.

هر پردازه p_i، الگوریتم زیر را اجرا می‌کند. گفتنی است که تابع $OrderingHeuristic$ می‌تواند هر یافتار مرتب‌سازی مانند LF یا $Random$ باشد. این تابع دو پارامتر را به‌عنوان ورودی می‌پذیرد. پارامتر اول، گره v و پارامتر دوم همسایه‌های گره v هستند.

$$\begin{aligned}
&w_i = OrderingHeuristic(p_i, N(p_i))\\
&if(\forall p_j \in N(p_i): w_j \leq w_i)\\
&\quad for\ each\ c \in S_i\\
&\qquad guard: \quad c_i \neq c \wedge u_i(c, c_{-i}) > u_i(c_i, c_{-i})\\
&\qquad action: \quad c_i = c
\end{aligned}$$

۴- نتایج شبیه‌سازی

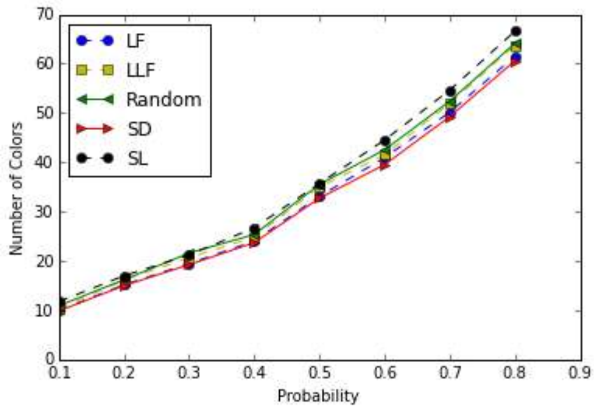
در این بخش، نتایج شبیه‌سازی روی سه توپولوژی- شبکه‌های مستقل از مقیاس (SF)، شبکه‌های تصادفی (ER) و شبکه‌های دنیای کوچک (WS)- را ارائه می‌کنیم. برای ساخت شبکه‌های SF، از الگوریتم مرجع [۳۷] استفاده کرده‌ایم. به ازای هر $\beta \in \{2.1, 2.4, \ldots, 4.4\}$، ۱۰۰۰ شبکه SF با ۲۰۰ گره تولید کرده‌ایم. همچنین به ازای هر $p \in \{0.1, 0.2, \ldots, 0.8\}$، ۱۰۰۰ شبکه ER و ۱۰۰۰ شبکه WS $(k=4)$ با ۲۰۰ گره تولید کرده‌ایم. گفتنی است که برای پیاده‌سازی الگوریتم خودتثبت‌کننده پیشنهادی و الگوریتم‌های ساختن شبکه‌ها، از زبان برنامه‌نویسی پایتون و برای ترسیم نمودارها از کتابخانه MatPlotLib این زبان، استفاده کرده‌ایم.

همچنین رایانه مورد استفاده دارای پردازنده Core-i5 4200 و ۶ گیگابایت رم می‌باشد.

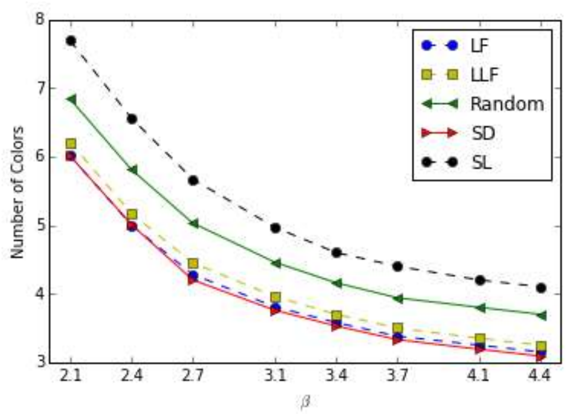
شکل‌های (۱- الف)، (۱- ب) و (۱- پ)، به ترتیب نتایج اجرای الگوریتم روی شبکه‌های ER، SF و WS هستند. الگوریتم را با ۵ یافتار مرتب‌سازی LF، LLF، $Random$، SD و SL پیاده‌سازی کرده‌ایم و هر نقطه در هر شکل، نتایج حاصل از میانگین تعداد رنگ‌های استفاده‌شده روی ۱۰۰۰ شبکه می‌باشد. همان‌طور که از شکل‌ها پیداست، یافتار مرتب‌سازی SD و LF نسبت به یافتار مرتب‌سازی $Random$ و دو یافتار مرتب‌سازی دیگر، به‌طور متوسط تا ۱۸٪ از تعداد رنگ کمتری استفاده می‌کنند. همچنین یافتار مرتب‌سازی SL، نسبت به سایر یافتارهای مرتب‌سازی، بطور متوسط از تعداد رنگ بیشتری استفاده می‌کند و کیفیت رنگ‌آمیزی آن پایین است.

شکل‌های (۲- الف)، (۲- ب) و (۲- پ)، به ترتیب نتایج اجرای الگوریتم روی شبکه‌های ER، SF و WS هستند. الگوریتم را با ۵ یافتار مرتب‌سازی LF، LLF، $Random$، SD و SL پیاده‌سازی کرده‌ایم و هر نقطه در هر شکل، نتایج حاصل از میانگین تعداد دورهای اجرای الگوریتم روی ۱۰۰۰ شبکه می‌باشد. همان‌طور که از شکل‌ها پیداست، یافتار مرتب‌سازی $Random$ روی شبکه‌های ER، یافتار مرتب‌سازی SL روی شبکه‌های SF و WS بهترین کارایی را داشته‌اند. استفاده از یافتار مرتب‌سازی SL روی شبکه‌های SF و WS نسبت به $Random$ تا ۵٪ موجب بهبود سرعت همگرایی به جواب می‌گردد.

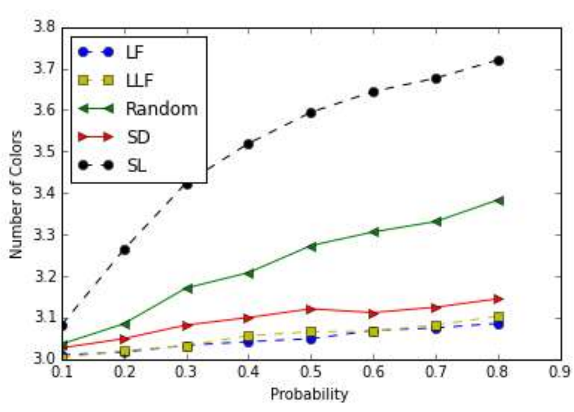
طبق جدول (۱) و بر اساس ویژگی‌های بارز الگوریتم‌های رنگ‌آمیزی خودتثبیت‌کننده که در این جدول درج شده است، ویژگی‌های الگوریتم هدنیمی و همکاران دقیقا با ویژگی‌های الگوریتم پیشنهادی یکسان است. لذا برای مقایسه الگوریتم پیشنهادی با الگوریتم‌های پیشین، صرفا آن‌را با الگوریتم هد نیمی و همکاران مقایسه می‌کنیم. گفتنی است که، خروجی الگوریتم هد نیمی و همکاران با خروجی الگوریتم پیشنهادی که از یافتار مرتب‌سازی $Random$ استفاده می‌کند، یکسان است. لذا درصورت استفاده از یافتارهای مرتب‌سازی SD و LF، کیفیت رنگ‌آمیزی الگوریتم پیشنهادی از الگوریتم هد نیمی و همکاران بیشتر خواهد شد.



الف- شبکه ER

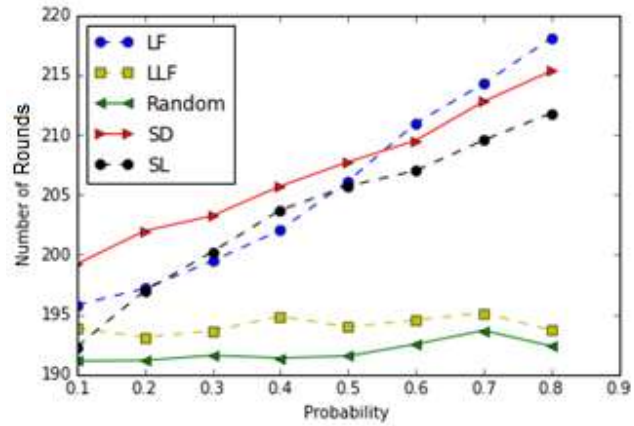


ب- شبکه SF

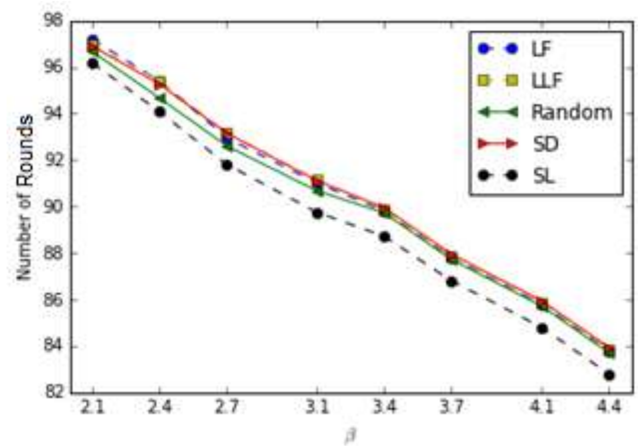


پ- شبکه WS

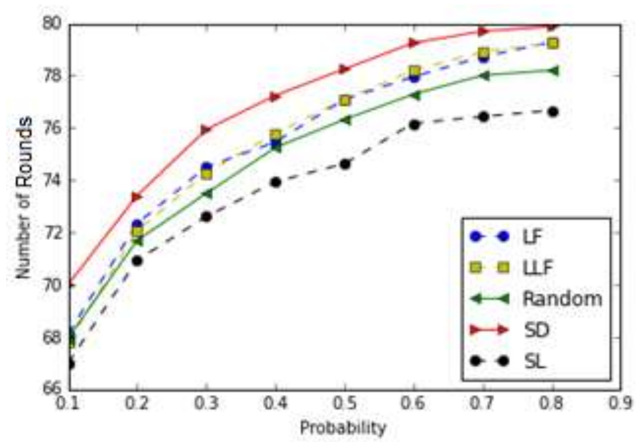
شکل (۱): مقایسه تعداد رنگ‌های استفاده شده توسط ۵ یافتار مرتب‌سازی



الف- شبکه ER



ب- شبکه SF



پ- شبکه WS

شکل (۲): مقایسه تعداد دورهای اجرای ۵ یافتار مرتب‌سازی

۵- جمع‌بندی و آینده تحقیق

با توجه به رشد روز افزونِ کاربرد سیستم‌های توزیع‌شده و ظهور حوزه‌های جدید مانند اینترنت اشیا، نیاز به ارائه الگوریتم‌های توزیع‌شده‌ی کارا بیش از پیش وجود دارد. از سوی دیگر، به دلیل امکان ایجاد خرابی در سیستم‌های توزیع‌شده (که به‌صورت عمدی و در پی حملات امنیتی روی می‌دهند و یا به‌صورت غیرعمدی و در اثر خطای گذرا و یا خطا در ارسال- دریافت پیام و غیره رخ می‌دهند) الگوریتم‌هایی که برای سیستم‌های توزیع‌شده ارائه می‌گردند باید قابلیت تحمل خرابی را داشته باشند.

در این مقاله، یک الگوریتم مبتنی بر نظریه بازی‌ها برای مسئلهٔ رنگ‌آمیزی گروندی خودتثبیت‌کننده روی شبح مرکزی و سیستم‌های توزیع‌شده ناشناس ارائه کردیم که در برابر خرابی گذرا مقاوم است. نوآوری‌ها و مزیت‌های روش پیشنهادی نسبت به روش‌های موجود از این قرار است:

۱. از یافتارهای مرتب‌سازی LF، LLF، $Random$، SD و SL برای بهبود کیفیت رنگ‌آمیزی خودتثبیت‌کننده استفاده کرده‌ایم. ترتیب‌های SD و LF نسبت به سایر یافتارهای مرتب‌سازی موجب افزایش بیشتر کیفیت رنگ‌آمیزی می‌شوند.

۲. مسئلهٔ رنگ‌آمیزی گروندی تا به حال برپایهٔ نظریه بازی‌ها بررسی نشده است. در مقاله حاضر، این مسئلهٔ را برای نخستین بار با استفاده از نظریه بازی‌ها حل کرده‌ایم.

برای آینده تحقیق، ارائه یک الگوریتم رنگ‌آمیزی گروندی خودتثبیت‌کننده و مبتنی بر نظریه بازی‌ها را روی شبح همگام و شبح توزیع‌شده پیشنهاد می‌کنیم. همچنین استفاده از راهبرد مختلط و بازی‌های بیزی را برای ارائه الگوریتم رنگ‌آمیزی گروندی خود تثبیت‌کننده احتمالاتی پیشنهاد می‌کنیم. یافتارهای مرتب‌سازی جدید نیز برای بهبود کیفیت رنگ‌آمیزی قابل ارائه هستند.

۶- مراجع

[1] N. A. Lynch, “Distributed Algorithms, Morgan Kaufmann,” 1996.

[2] L. Wang and R. Ranjan, “Processing distributed internet of things data in clouds,” IEEE Cloud Computing, vol. 2, pp. 76–80, 2015.

[3] D. Peleg, “Distributed Computing: A Locality-Sensitive Approach,” SIAM, 2000.

[4] N. Guellati and H. Kheddouci, “A survey on self-stabilizing algorithms for independence, domination, coloring, and matching in graphs,” Journal of Parallel and Distributed Computing, vol. 70, pp. 406–415, 2010.

[5] L. H. Yen, J. Y. Huang, and V. Turau, “Designing self-stabilizing systems using game theory,” ACM Trans. Auton. Adapt. Syst., vol. 11, pp. 1–27, 2016.

[6] L. H. Yen and Z. L. Chen, “Game-theoretic approach to self-stabilizing distributed formation of minimal multi-dominating sets,” IEEE Trans. Parallel Distrib. Syst., vol. 25, pp. 3201–3210, 2014.

[7] E. W. Dijkstra, “Self-stabilizing systems in spite of distributed control,” Communications of the ACM, vol. 17, pp. 643–644, 1974.

[8] S. Devismes and C. Johnen, “Silent self-stabilizing BFS tree algorithms revisited,” Journal of Parallel and Distributed Computing, vol. 97, pp. 11–23, 2016.

[9] J. Cohen, J. Lefèvre, K. Maâmra, L. Pilard and D. Sohier, “A self-stabilizing algorithm for maximal matching in anonymous networks,” Parallel Processing Letters, vol. 26, pp. 49–59, 2016.

[10] Y. Fu and X. Xiaoping, “Self-stabilized distributed network distance prediction,” IEEE/ACM Transactions on Networking, vol. 25, pp. 451–464, 2017.

[11] L. Blin and S. Tixeuil, “Compact deterministic self-stabilizing leader election on a ring,” Distributed Computing, vol. 1998, pp. 1-28, 2017.

[12] A. K. Datta, S. Devismes, and L. L. Larmore, “Self-stabilizing silent disjunction in an anonymous network,” Theoretical Computer Science, vol. 665, pp. 51–72, 2017.

[13] J. Behnamian, “Graph colouring-based algorithm to parallel jobs scheduling on parallel factories,” International Journal of Computer Integrated Manufacturing, vol. 29, pp. 622–635, 2015.

[14] B. Yüceoğlu, G. Şahin and S. P. van Hoesel, “A column generation based algorithm for the robust graph coloring problem,” Discrete Applied Mathematics, vol. 217, pp. 340–352, 2017.

[15] C. Zhao, X. Xu, Z. Gao, and L. Huang, “A coloring-based cluster resource allocation for ultra dense network,” in IEEE International Conference on Signal Processing, Communications and Computing, Hong Kong, pp. 1-5, 2016.

[16] S. Basloom, A. Nazar, G. Aldabbagh, M. Abdullah, and N. Dimitriou, “Resource allocation using graph coloring for dense cellular networks,” in International Conference on Computing, Networking and Communications, Kauai, pp. 1-5, 2016.

[17] L. Barenboim, M. Elkin, S. Pettie, and J. Schneider, “The locality of distributed symmetry breaking,” Journal of the ACM (JACM), vol. 63, 2016. doi: 10.1145/2903137

[18] A. S. Sairam, S. Roy, and R. Sahay, “Coloring networks for attacker identification and response,” Security and Communication Networks, vol. 8, pp. 751–768, 2015.

[19] B. L. Hartnell and C. M. Mynhardt, "Independent protection in graphs," Discrete Mathematics, vol. 335, pp. 100–109, 2014.

[20] M. Gradinariu and S. Tixeuil, “Self-stabilizing vertex coloring of arbitrary graphs,” in International conference on Principles of Distributed Systems, Paris, pp. 55-70, 2000.

[21] A. Mansouri and M. S. Bouhlel, “Efficient self-stabilizing grundy coloring algorithms,” in 2016 Future Technologies Conference FTC 2016, San Francisco, pp. 199-205, 2016.

[22] P. N. Panagopoulou and P. G. Spirakis, “A game theoretic approach for efficient graph coloring,” in International Symposium on Algorithms and Computation, Berlin, pp. 183-195, 2008.

[23] I. Chatzigiannakis, C. Koninis, P. N. Panagopoulou, and P. G. Spirakis, “Distributed game-theoretic vertex coloring,” in International Conference On Principles Of Distributed Systems, Tozeur, pp. 103-118, 2010.

[24] A. R´enyi. and P. Erdos, “On random graphs,” Publ. Math. Debr., vol. 6, pp. 290–297, 1959.

[25] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” Nature, vol. 393, pp. 440–442, 1998.

[26] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” Science, vol. 286, pp. 509–512, 1999.

[27] D. J. A. Welsh and M. B. Powell, “An upper bound for the chromatic number of a graph and its application to timetabling problems,” The Computer Journal, vol. 10, pp. 85–86, 1967.

[28] S. T. Hedetniemi, D. P. Jacobs, and P. K. Srimani, “Linear time self-stabilizing colorings,” Information Processing Letters, vol. 87, pp. 251–255, 2003.

[29] W. Goddard, S. T. Hedetniemi, D. P. Jacobs, and P. K. Srimani, “Self-stabilizing algorithm for orderings and colorings,” Int. J. Found. Comput. Sci., vol. 16, pp. 19–36, 2005.

[30] S. Bernard, S. Devismes, M. G. Potop-Butucaru, and S. Tixeuil, “Optimal deterministic self-stabilizing vertex coloring in unidirectional anonymous networks,” in IEEE International Symposium on Parallel & Distributed Processing, New York, pp. 1-8, 2009.

[31] S. Bernard, S. Devismes, K. Paroux, M. Potop-Butucaru, and S. Tixeuil, “Probabilistic self-stabilizing vertex coloring in unidirectional anonymous networks,” in International Conference on Distributed Computing and Networking, Kolkata, pp.167-177, 2010.

[32] A. Kosowski and Ł. Kuszner, “Self-stabilizing algorithms for graph coloring with improved performance guarantees,” in Artificial Intelligence and Soft Computing, London, pp. 1150-1159, 2006.

[33] W. Hasenplaugh, T. Kaler, T. B. Schardl, and C. E. Leiserson, “Ordering heuristics for parallel graph coloring,” in Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures , Prague, pp. 166-177, 2014.

[34] C. A. Christen and S. M. Selkow, “Some perfect coloring properties of graphs,” Journal of Combinatorial Theory, Series B, vol. 27, pp. 49–59, 1979.

[35] V. Bilò, A. Fanelli, M. Flammini, and L. Moscardelli, "Graphical congestion games," Algorithmica, vol. 61, pp. 274–297, 2011.

[36] D. Monderer and L. S. Shapley, "Potential games," Games and Economic Behavior, vol. 14, pp. 124–143, 1996.

[37] J. C. Miller and A. Hagberg, "Efficient generation of networks with given expected degrees," in International Workshop on Algorithms and Models for the Web-Graph, Atlanta, pp. 115-126, 2011.

Self-stabilizing Grundy Coloring Using Game Theory and Heuristics Ordering

H. Heydari, S. M. Taheri*

*University of Tehran, College of Engineering, School of Engineering Science

(Received: 21/04/2017, Accepted: 04/10/2017)

ABSTRACT

Transient faults in distributed systems can be occurred in many situations like process failure and security attacks. A self-stabilizing algorithm, regardless of the initial state, converges in finite time to legitimate states and tolerates transient faults. In this paper, we propose a self-stabilizing Grundy coloring using some concepts/results in the game theory. The proposed algorithm deals with autonomous networks, where nodes do not have identifiers. By using Nash equilibrium, we prove our proposed algorithm converges in $O(m)$ moves, where m is the number of network edges. Simulation results indicate that heuristics ordering leads to decrease the number of colors up to 18% and increase the Convergence up to 5%.

Keywords: Transient Fault, Network Security, Centralized Daemon, Nash Equilibrium, Autonomous Distributed System

* Corresponding Author Email: sm_taheri@ut.ac.ir