علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۷، شماره ۲، ۱۳۹۸
صفحات ۲۳-۱۵
مقاله عادی

توسعه الگوریتم بولی اصلاح‌شده جهت حذف پدیده افزونگی انتخابات در انتخاب هماهنگ‌کننده در سیستم‌های توزیع‌شده

رضا نورمندی پور*[۱]

*نویسنده مسئول، دریافت: ۹۸/۰۸/۲۱، بازنگری: ۹۸/۱۰/۱۸، پذیرش: ۹۸/۱۱/۲۸

[۱] استادیار، دانشکده فنی و مهندسی، واحد سیرجان، دانشگاه آزاد اسلامی، سیرجان، ایران

چکیده

در سیستم‌های توزیع‌شده که دارای فرایندها و منابع متعددی هستند، همگام‌سازی فرایندها برای برخی مسائل نظیر ورود به ناحیه بحرانی و استفاده از منابع مشترک امری اجتناب‌ناپذیر است. فرایند هماهنگ‌کننده نقش کلیدی را برای همگام‌سازی فرایندها ایفا می‌کند؛ بنابراین، انتخاب هماهنگ‌کننده یکی از مسائل مهم نه‌تنها در سیستم‌های توزیع‌شده بلکه در بسیاری از زمینه‌ها نظیر شبکه‌های ارتباطی، الگوریتم‌های انحصار متقابل، تخصیص منابع مشترک و غیره است. یکی از چالش‌های موجود در الگوریتم‌های انتخاب هماهنگ‌کننده کاهش پیام‌ها بین فرایندها است که تأثیر زیادی بر روی میزان ترافیک شبکه خواهد داشت. هرچند تلاش‌های زیادی در کاهش پیام‌ها بین فرایندها صورت گرفته است، ولی همچنان پدیده افزونگی انتخابات یکی از محدودیت‌هایی است که الگوریتم‌های انتخاب هماهنگ‌کننده با آن مواجه هستند. ایده اصلی در این مقاله ارائه رویکردهایی نوین در توسعه الگوریتم بولی اصلاح‌شده است با این هدف که پدیده افزونگی انتخابات در انتخاب هماهنگ‌کننده حذف شود. با به‌کارگیری این رویکردها در توسعه الگوریتم بولی اصلاح‌شده نه‌تنها پیچیدگی پیامی برابر $O(n)$ حفظ شده است، بلکه تضمین می‌کنند که تنها یک فرایند درگیر انتخابات شود که این باعث حذف پدیده افزونگی انتخابات و درنتیجه باعث کاهش ازدحام و ترافیک شبکه می‌شود.

کلمات کلیدی: سیستم‌های توزیع‌شده، انتخاب هماهنگ‌کننده، تبادل پیام، الگوریتم بولی اصلاح‌شده، افزونگی انتخابات.

۱- مقدمه

۱-۱- انگیزه

برخی الگوریتم‌های تخصیص منابع در سیستم‌های توزیع‌شده نیاز به یک هماهنگ‌کننده[۱] در کل سیستم دارند که برخی از فعالیت‌های هماهنگ‌کنندگی موردنیاز سایر فرایندهای موجود در سیستم را انجام دهد. تمام فرایندها در سیستم باید با فرایند هماهنگ‌کننده تعامل و ارتباط داشته باشند. همچنین همه آن‌ها باید و لازم است که بر وجود یک فرایند به‌عنوان هماهنگ‌کننده باهم به‌اتفاق آراء به توافق برسند. بعلاوه اگر فرایندی که نقش هماهنگ‌کننده جاری را بر عهده دارد به هر دلیل سقوط کند، آنگاه نیاز به برگزاری انتخابات برای انتخاب هماهنگ‌کننده جدید است. انتخابات روندی برای تعیین یک فرایند از میان تمام فرایندهای فعال در سیستم برای انجام برخی از وظایف توزیع‌شده است.

اگر همه فرایندهای موجود در سیستم بدون هیچ ویژگی متمایزی دقیقاً شبیه به هم باشند، هیچ راهی برای انتخاب یک فرایند خاص به‌عنوان هماهنگ‌کننده وجود ندارد. هماهنگ‌کننده یک فرایندی با شماره شناسه مشخص و متمایز است. بنابراین، مکانیسمی که برای انتخاب یک فرایند به‌عنوان هماهنگ‌کننده که نقش متمایز و متفاوتی از سایر فرایندها در سیستم ایفا می‌کند، به الگوریتم انتخابات[۲] معروف است [۳].

یکی از مهم‌ترین چالش‌ها در سیستم‌های توزیع‌شده اتخاذ یک الگوریتم کارا و مناسب برای انتخاب هماهنگ‌کننده است. مهم‌ترین نقش هماهنگ‌کننده در مدیریت استفاده از منابع مشترک به شیوه‌ای مطلوب است. این مسئله زمانی مطرح می‌شود که هماهنگ‌کننده جاری به هر دلیلی سقوط می‌کند و سیستم نیازمند انتخاب یک فرایند از میان همه فرایندها برای سپردن وظایف هماهنگ‌کنندگی است. از مشخصه‌های مهم انتخاب هماهنگ‌کننده تعداد پیام‌های مؤثر برای انتخاب

هماهنگ‌کننده، زمان تأخیر و مسئله بازگشت یک فرایند سقوط کرده به سیستم است که در این میان پرداختن به تعداد پیام‌ها و تلاش برای کاهش آن مهم‌ترین چالش پیش روی محققان است، زیرا تعداد پیام‌ها تأثیر مستقیم بر ترافیک شبکه و به دنبال آن پیچیدگی‌های شبکه خواهد گذاشت. ازاین‌رو تمرکز تحقیقات بیشتر روی یافتن الگوریتم‌هایی است که تعداد پیام‌ها برای یافتن هماهنگ‌کننده را کاهش دهند. هرچند موفقیت‌هایی در این خصوص حاصل شده است، ولی وقوع پدیده *افزونگی انتخابات*[۳] که در آن چندین فرایند هم‌زمان اقدام به برگزاری انتخابات می‌کنند که این خود نتیجه در بالا رفتن تعداد پیام‌های تبادلی و به دنبال آن افزایش ترافیک شبکه می‌شود، همچنان چالشی در پیش روی محققین است. بدین‌سان در این مقاله بر آنیم تا با پیشنهاد رویکردهای جدید به این مسئله علاوه بر کاهش تعداد پیام‌ها، فقط یک فرایند را در هر شرایطی درگیر انتخابات کنیم تا از افزونگی انتخابات جلوگیری شود.

۱-۲- مروری بر کارهای گذشته

با توجه به اینکه نقش مهم هماهنگ‌کننده در مدیریت استفاده از منابع مشترک به شکل بهینه است، لذا مسئله انتخاب هماهنگ‌کننده یک موضوع حیاتی نه‌تنها در محاسبات توزیع‌شده بلکه در ارتباطات شبکه‌ای، الگوریتم‌های متمرکز، ورود به ناحیه بحرانی و غیره است. مهم‌ترین چالش پیش رو تعداد پیام‌ها برای انتخاب هماهنگ‌کننده است. لذا تحقیقاتی که در این باب صورت گرفته همگی ناظر به تلاش برای کاهش تعداد تبادل پیام‌ها میان فرایندها در شبکه تا انتخاب یک فرایند به‌عنوان هماهنگ‌کننده جدید پس از سقوط هماهنگ‌کننده جاری است.

در سال ۱۹۷۷ نویسندگان الگوریتمی مبتنی بر حلقه را پیشنهاد نمودند که پیچیدگی پیام آن $O(n^2)$ و پیچیدگی زمانی آن $O(n)$ است [۱]. چانگ و روبرت در سال ۱۹۷۹ الگوریتمی ارائه کردند که مبتنی بر مجموعه‌ای از فرایندها است که به‌صورت حلقه به هم متصل هستند و پیام در حلقه به‌صورت یک‌سویه و در جهت عقربه‌های ساعت حرکت می‌کند. پیچیدگی پیام در این الگوریتم برابر $O(n^2)$است [۲].

در سال ۱۹۸۲ نویسندگان در مرجع [۳] الگوریتم بولی را برای انتخاب هماهنگ‌کننده پیشنهاد کردند که مبتنی بر تبادل پیام میان فرایندها بوده که هر فرایند برای فرایندهای بزرگ‌تر از خود پیام ارسال می‌کند و در پایان بزرگ‌ترین فرایند فعال به‌عنوان هماهنگ‌کننده جدید انتخاب می‌شود که مستلزم $O(n^2)$ پیام است. تابه‌حال تحقیقات زیادی در خصوص انتخاب هماهنگ‌کننده مبتنی بر الگوریتم بولی انجام شده است. برخی از تحقیقات انجام شده تمرکز روی الگوریتم بولی و سعی در بهبود عملکرد آن به‌منظور کاهش پیچیدگی پیامی داشته‌اند [۴-۸].

هرچند محققین الگوریتم بولی را اصلاح کرده‌اند و پیچیدگی پیامی را به $O(n)$ کاهش داده‌اند، ولی این الگوریتم اصلاح‌شده نیز دارای محدودیت‌هایی است که در ذیل به آن‌ها اشاره شده است [۸].

- اگر فرایند i بعد از فرستادن پیام انتخاباتی به فرایندها با اولویت بالاتر یا بعد از دریافت شماره شناسه فرایندها با اولویت بالاتر سقوط کند، فرایند با بزرگ‌ترین شناسه به‌اندازه $3D$ (D تأخیر به‌طور متوسط) برای پخش همگانی کردن شناسه هماهنگ‌کننده منتظر می‌ماند و اگر هیچ پیام تأییدی را دریافت نکند آنگاه الگوریتم را مجدد اجرا می‌کند. به عبارتی، اگر q فرایند مختلف با اولویت بالاتر از فرایند i وجود داشته باشد، بنابراین q نمونه الگوریتم در آن لحظه در سیستم اجرا می‌شود که باعث افزونگی انتخابات می‌شود.
- اگر فرایند i یک پیام تأیید را برای فرایند با شناسه بالاتر (فرایند انتخاب‌شده به‌عنوان هماهنگ‌کننده) بفرستد و فرایند i پیام هماهنگ‌کننده را در زمان D دریافت نکند، آنگاه فرایند i الگوریتم را تکرار خواهد کرد که این نیز باعث افزونگی انتخابات می‌شود.

در سال ۱۹۹۴ در مرجع [۹] الگوریتم پیشنهادی مبتنی بر حلقه و ارسال پیام در شبکه مبتنی بر بازه زمانی است که پیچیدگی زمانی و پیام را در حد $O(n)$ حاصل شده است. احسان کبیر و همکاران نیز در سال ۲۰۰۴ الگوریتمی مبتنی بر شبکه در مد هم‌زمان پیشنهاد دادند که پیچیدگی پیام در الگوریتم از مرتبه $O(n)$ است [۱۰]. در سال ۲۰۱۴ یک الگوریتم انتخاب رهبر مبتنی بر مدل احتمالی با استفاده از مفهوم هیئت انتخاباتی و همچنین یک الگوریتم پویا در انتخاب رهبر به ترتیب در مراجع [۱۱] و [۱۲] ارائه شده است که روی حل مسئله سقوط هماهنگ‌کننده جاری متمرکز هستند.

در مرجع [۱۳] یک روش جدید مبتنی بر انتخاب یک هماهنگ‌کننده و جانشینان آن ارائه داده‌اند، به‌طوری‌که با سقوط هماهنگ‌کننده، بدون آغاز انتخابات، جانشین هماهنگ‌کننده وظائف آن را به عهده گیرد.

۱-۳- اهداف و ساختار مقاله

در این مقاله دو رویکرد در توسعه الگوریتم بولی اصلاح‌شده پیشنهاد شده است که در هر یک از آن‌ها با درگیر کردن فقط یک فرایند در روند انتخابات، از وقوع پدیده *افزونگی انتخابات* جلوگیری و به دنبال آن، باعث کاهش چشمگیر ترافیک شبکه در روند انتخاب هماهنگ‌کننده جدید می‌شود. در ادامه، مفاهیم اولیه و رخداد پدیده افزونگی انتخابات در روند انتخاب یک هماهنگ‌کننده در بخش دوم مطرح شده است. در بخش سوم، جزئیات هر یک از رویکردهای پیشنهاد شده، که نوآوری‌های این مقاله هستند، و هرکدام می‌توانند به‌طور مجزا در توسعه الگوریتم بولی اصلاح‌شده بکار روند، تشریح شده است. در بخش چهارم پیچیدگی پیامی رویکردهای پیشنهادی به لحاظ تئوری مورد تجزیه‌وتحلیل قرار گرفته است. ارزیابی رویکردهای پیشنهادی با به‌کارگیری آن‌ها در الگوریتم بولی اصلاح‌شده در بخش پنجم انجام گرفته است. در انتها، نتیجه‌گیری در بخش ششم آورده شده است.

۲- مفاهیم اولیه و بیان مسئله

سیستم‌های توزیع‌شده متکی بر ارتباطات هستند و به‌طورکلی از دو سرویس انتقال پیام[۴] و فراخوانی از راه دور رویه‌ها[۵] استفاده می‌کنند.

یک سیستم توزیع‌شده مجموعه‌ای از پردازنده‌هاست که توسط یک شبکه، ارتباطی تنگاتنگ با هم دارند که در آن هر پردازنده حافظه محلی مستقل و لوازم جانبی مخصوص به خود را دارد و ارتباط بین آن‌ها از طریق تبادل پیام روی شبکه‌های ارتباطی است [۱۴]. الگوریتم‌های توزیع‌شده نیاز به یک گره هماهنگ‌کننده[۶] در کل سیستم دارند که برخی از فعالیت‌های هماهنگ‌کنندگی موردنیاز سایر گره‌های موجود در سیستم را انجام دهد. تمام گره‌ها در سیستم باید با گره هماهنگ‌کننده تعامل و ارتباط داشته باشند. همچنین همه آن‌ها باید و لازم است که بر سر رسیدن به یک گره به‌عنوان هماهنگ‌کننده با هم به‌اتفاق آراء به توافق برسند. بعلاوه اگر گرهی که نقش هماهنگ‌کننده جاری را بر عهده دارد به هر دلیل سقوط[۷] کند، آنگاه نیاز به برگزاری انتخابات برای انتخاب هماهنگ‌کننده جدید لازم است. انتخابات[۸] روندی برای تعیین یک فرایند از میان تمام فرایندهای فعال در سیستم برای انجام برخی از وظایف توزیع‌شده است. یکی از مهم‌ترین چالش‌ها در سیستم‌های توزیع‌شده اتخاذ یک الگوریتم کارا و مناسب برای انتخاب هماهنگ‌کننده است. از مشخصه‌های مهم انتخاب هماهنگ‌کننده تعداد پیام‌های مؤثر برای انتخاب هماهنگ‌کننده، زمان تأخیر و مسئله بازگشت یک فرایند سقوط کرده به سیستم است که در این میان پرداختن به تعداد پیام‌ها و تلاش برای کاهش آن مهم‌ترین چالش پیش رو محققان است، زیرا تعداد پیام‌ها تأثیر مستقیم بر ترافیک شبکه و به دنبال آن پیچیدگی‌های شبکه خواهد گذاشت. ازاین‌رو تمرکز تحقیقات بیشتر روی یافتن الگوریتم‌هایی است که تعداد پیام‌ها برای یافتن هماهنگ‌کننده را کاهش دهد.

تمامی الگوریتم‌های انتخاب هماهنگ‌کننده که تاکنون مطرح شده‌اند بر این فرض اساسی و مشترک استوار هستند که همواره بزرگ‌ترین فرایند در گروه به‌عنوان هماهنگ‌کننده انتخاب خواهد شد، بنابراین نیازی به شرکت دادن همه فرایندها در روند انتخابات نیست. در ادامه مروری بر الگوریتم بولی اصلاح‌شده [۸] خواهیم داشت به‌طوری‌که عملکرد این الگوریتم در انتخاب هماهنگ‌کننده تشریح شده است. سپس، پدیده افزونگی انتخابات که منجر به شراکت بیش از یک فرایند در روند انتخابات می‌شود، در الگوریتم بولی اصلاح‌شده مورد بررسی قرار می‌گیرد. لازم به ذکر است که این پدیده افزونگی انتخابات می‌تواند در هر الگوریتم انتخابات رخ دهد و منحصر به الگوریتم بولی اصلاح‌شده نیست. انتخاب الگوریتم بولی اصلاح‌شده در این مقاله صرفاً به دلیل کارا بودن و پایین بودن پیچیدگی آن است.

۲-۱- مروری بر الگوریتم بولی اصلاح‌شده

اساس الگوریتم بولی اصلاح‌شده بر فرضیات ذیل بنا شده است:

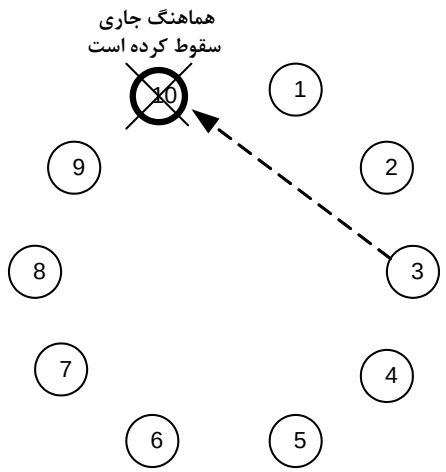
- زیرساخت شبکه اتصال کامل است، به عبارتی کلیه ماشین‌ها (فرایندها) در سیستم توزیع‌شده مستقیماً با یکدیگر ارتباط دارند.
- همه فرایندها دارای شماره شناسه منحصربه‌فرد هستند.
- زیرساخت این الگوریتم بر مبنای پیوستگی همه فرایندها است.
- کلیه فرایندها شماره و آدرس دیگر فرایندها را می‌دانند.
- همواره بزرگ‌ترین فرایند به‌عنوان هماهنگ‌کننده انتخاب خواهد شد.
- در هرلحظه تنها یک فرایند نقش هماهنگ‌کنندگی را خواهد داشت.

به‌محض اینکه فرایندی از سقوط هماهنگ‌کننده مطلع شود (فرایند ۳ در شکل ۱) یک *پیام انتخاباتی*[۹] را به فرایندی با بزرگ‌ترین شناسه (فرایند ۹ در شکل ۲) در سیستم ارسال می‌کند. فرایندی که پیام را دریافت می‌کند (فرایند ۹ در شکل ۲)، در صورت فعال بودن باید یک *پیام پذیرش*[۱۰] به فرایند آغازکننده انتخابات ارسال کند (شکل ۳ را ببینید). این پیام پذیرش، آمادگی فرایند را برای پذیرش نقش هماهنگ‌کننده و همچنین مطلع کردن فرایند آغازکننده جهت جلوگیری از ارسال پیام‌های اضافی مشخص می‌کند.

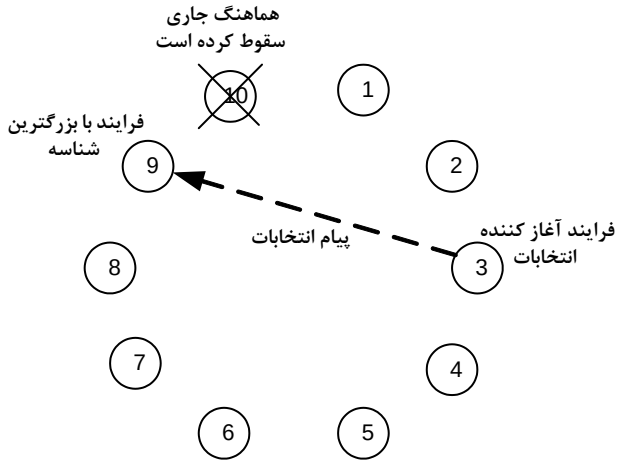
در پایان، فرایند دریافت‌کننده باید خودش را به‌عنوان هماهنگ‌کننده جدید معرفی کند که این کار با ساخت یک *پیام هماهنگ‌کننده*[۱۱] و ارسال آن به همه فرایندهای موجود در سیستم انجام می‌شود (شکل ۴ را ببینید).

۲-۲- بیان مسئله-پدیده افزونگی انتخابات در الگوریتم بولی اصلاح‌شده

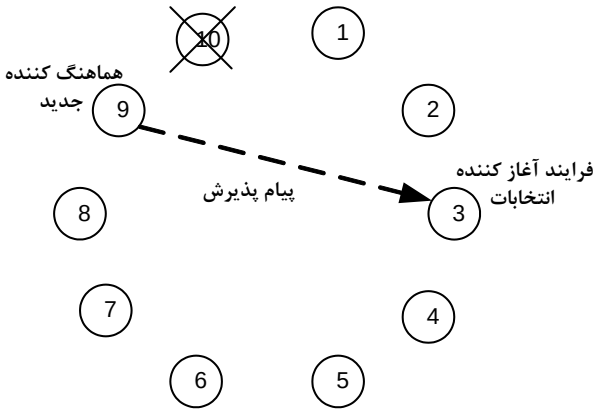
با توجه به شکل ۴، فرض شود که فرایند ۹ به‌عنوان هماهنگ‌کننده جاری در سیستم است و فرایند عادی با شناسه ۳ مطلع می‌شود که فرایند ۹ به هر علتی سقوط کرده است، لذا انتخابات را آغاز و به فرایند موجود در حلقه با بزرگ‌ترین شناسه، فرایند ۸، یک پیام انتخاباتی ارسال می‌کند و فرایند ۸ در صورت فعال بودن یک پیام پذیرش به فرایند ۳ ارسال می‌کند. فرایند ۸، آنگاه یک پیام هماهنگ‌کننده را ساخته و به فرایندهای موجود در سیستم پخش همگانی می‌کند. هر فرایند پس از دریافت این پیام، فیلد مربوط به هماهنگ‌کنندگی خود را، LEID[۱۲]، در جدول خود بروز رسانی می‌کند. حال مشکل زمانی است که یک فرایند مانند فرایند ۱ نیز در همین فاصله می‌خواهد با هماهنگ‌کننده تبادل پیام کند، درحالی‌که هیچ پیامی از هماهنگ‌کننده جدید، فرایند ۸، دریافت نکرده است و فرایند ۱ همچنان فرایند ۹ را به‌عنوان هماهنگ‌کننده جاری می‌شناسد. لذا یک پیام برای فرایند ۹ که سقوط کرده است، ارسال می‌کند و به طبع هیچ پاسخی دریافت نمی‌کند. حال اگر در این فاصله زمانی، پیام هماهنگ‌کننده جدید را از فرایند ۸ دریافت کند، اقدام به به‌روزرسانی فیلد LEID می‌کند، اما در صورت عدم دریافت هیچ پیامی اقدام به برگزاری انتخابات جدید می‌کند، به عبارتی، یک پیام انتخاباتی را برای فرایند فعال در حلقه با بزرگ‌ترین شناسه که همان فرایند ۸ است، ارسال می‌کند و فرایند ۸ نیز یک پیام پذیرش ارسال می‌کند. این عملیات تکراری ممکن است به تعداد فرایندها رخ دهد که این امر باعث پدیده‌ای بنام *افزونگی انتخابات* در سیستم می‌شود که افزایش پیام و ترافیک شبکه را در پی خواهد داشت.



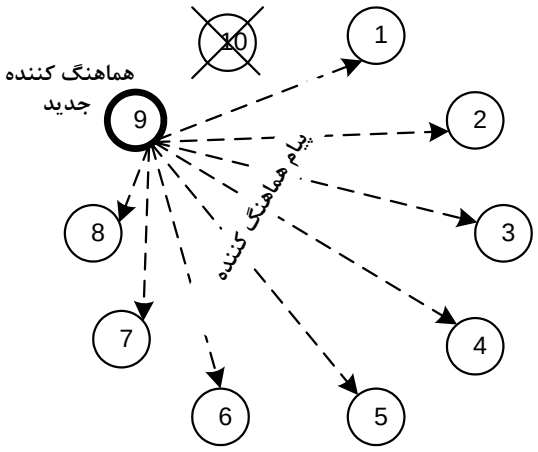
شکل ۱: فرایند ۳ تشخیص می‌دهد که هماهنگ‌کننده جاری سقوط کرده است.



شکل ۲: فرایند ۳ یک پیام انتخابات برای فرایند ۹ ارسال می‌کند.



شکل ۳: فرایند شماره ۹ با یک پیام پذیرش به فرایند ۳ پاسخ می‌دهد.



شکل ۴: فرایند ۹ یک پیام هماهنگ‌کننده را به همه فرایندهای موجود در سیستم پخش همگانی می‌کند.

۳- رویکردهای پیشنهادی در توسعه الگوریتم بولی اصلاح‌شده

الگوریتم‌هایی که تاکنون در این زمینه ارائه شده است، به‌ویژه الگوریتم بولی اصلاح‌شده، در خصوص این پدیده افزونگی انتخابات هیچ راه‌حلی ارائه نکرده‌اند. در این مقاله جهت اجتناب از این پدیده دو رویکرد پیشنهاد شده است؛ رویکرد *انتظار-مصلحتی*[۱۳] و رویکرد *تعداد-انتخابات*[۱۴] که با به‌کارگیری هریک از آن‌ها از وقوع افزونگی انتخابات جلوگیری می‌شود. در ادامه جزئیات این دو رویکرد مورد بحث قرار گرفته است.

۳-۱- رویکرد انتظار-مصلحتی

در این رویکرد هر فرایندی که بعد از انتخاب هماهنگ‌کننده جدید برای اولین بار یا بعد از بازیابی[۱۵] مجدد وارد سیستم می‌شود فاصله زمانی خود را تا هماهنگ‌کننده محاسبه می‌کند. به عبارتی، یک پیام ECHO به هماهنگ‌کننده ارسال و همان پیام را از هماهنگ‌کننده دریافت و متوسط زمانی ارسال و دریافت پیام را محاسبه می‌کند و آن را به‌عنوان زمان *انتظار-مصلحتی* ثبت می‌کند. حالتی را در نظر بگیرید که بیش از یک فرایند هم‌زمان تشخیص دهند که هماهنگ‌کننده جاری سقوط کرده است، یا حالتی را در نظر بگیرید که یک فرایند سقوط هماهنگ‌کننده را تشخیص دهد و انتخابات را آغاز کند و به دلیل وجود تأخیر حین ساخت و ارسال پیام هماهنگ‌کننده جدید به سایر فرایندها، ممکن است فرایندی یا فرایندهای دیگر سقوط هماهنگ‌کننده را قبل از دریافت پیام هماهنگ‌کننده منتخب تشخیص دهند که در هر دو حالت همه آن‌ها به‌طور مستقل اقدام به برگزاری انتخابات می‌کنند که باعث افزونگی پیام‌های انتخاباتی خواهد شد. بنابراین، برای اجتناب از رخداد این وضعیت، هر فرایند باید به‌اندازه زمان انتظار-مصلحتی خود صبر کند. ازآنجاکه این زمان یکتا و منحصربه‌فرد است، این تأخیر باعث می‌شود که فرایندها هم‌زمان انتخابات را آغاز نکنند و فرایندی که زمان انتظار-مصلحتی آن زودتر به پایان برسد با فرستادن پیام انتخاباتی به اولین فرایند فعال در سیستم با بزرگ‌ترین شناسه، اقدام به برگزاری انتخابات می‌کند.

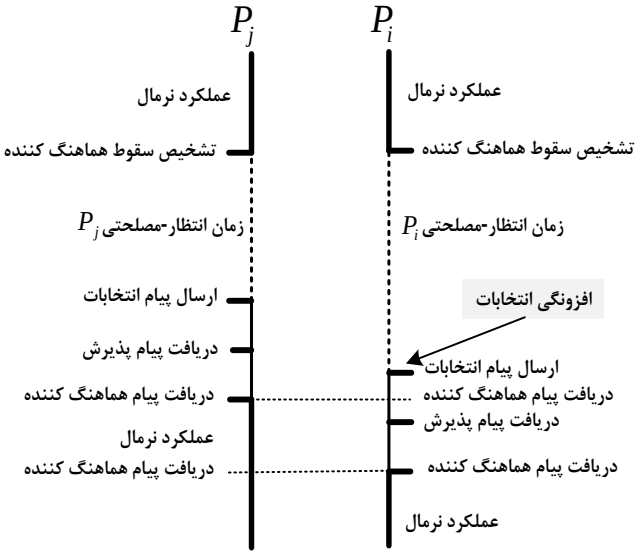
در این رویکرد، همان‌طور که در شکل ۵ نشان داده شده است، حالتی را در نظر بگیرید که فرایند i با ماکزیمم زمان انتظار-مصلحتی، سقوط هماهنگ‌کننده را تشخیص دهد و لذا بایستی به اندازه زمان انتظار-مصلحتی خود صبر کند و سپس اقدام به آغاز انتخابات کند. در همین حین، فرایند j که زمان انتظار-مصلحتی آن حداقل است، سقوط هماهنگ‌کننده را تشخیص و پس از اتمام زمان انتظار-مصلحتی خود و قبل از اتمام زمان انتظار-مصلحتی فرایند i، اقدام به انتخابات می‌کند. حال فرایند i نیز با اتمام زمان انتظار-مصلحتی خود و قبل از دریافت پیام هماهنگ‌کننده از طرف هماهنگ‌کننده جدید، اقدام به انتخاب می‌کند که این باعث رخداد پدیده افزونگی انتخاب می‌شود. پرواضح است که این وضعیت می‌تواند هم‌زمان برای بیش از دو فرایند نیز رخ دهد.

برای اینکه تضمین شود در رویکرد انتظار-مصلحتی فقط یک فرایند درگیر انتخابات می‌شود و از رخداد افزونگی انتخابات جلوگیری می‌شود، لازم است هماهنگ‌کننده بعد از ارسال پیام هماهنگ‌کننده، با ارسال پیام[۱۶]PCRQ، زمان انتظار-مصلحتی هریک از فرایندهای فعال را درخواست کند (شکل ۶ را ببینید) و سپس فرایندهای فعال با ارسال پیام[۱۷]PCRP، زمان انتظار-مصلحتی خود را به هماهنگ‌کننده اعلام کنند (شکل ۷ را ببینید)، در انتها، هماهنگ‌کننده با ارسال پیام[۱۸]MPC، ماکزیمم زمان انتظار-مصلحتی را به همه فرایندهای فعال اعلان می‌کند (شکل ۸ را ببینید). بنابراین، سه پیام PCRQ، PCRP و MPC بایستی پس از انتخاب هماهنگ‌کننده، بین هماهنگ‌کننده جدید و فرایندهای فعال تبادل شود تا عدم رخداد پدیده افزونگی انتخابات تضمین شود. البته پیام PCRQ می‌تواند به همراه پیام هماهنگ‌کننده در قالب یک پیام از طرف هماهنگ‌کننده جدید به فرایندهای فعال ارسال شود تا تعداد پیام‌های تبادلی کاهش یابد (شکل ۶ را ببینید). بنابراین، هر فرایند باید علاوه بر نگهداری زمان انتظار-مصلحتی مخصوص به خود، ماکزیمم زمان انتظار-مصلحتی را نیز نگاه دارد. حال هر فرایندی که سقوط هماهنگ‌کننده جاری را تشخیص دهد، بایستی به اندازه ماکزیمم زمان انتظار-مصلحتی صبر کند و سپس اقدام به آغاز انتخابات کند.

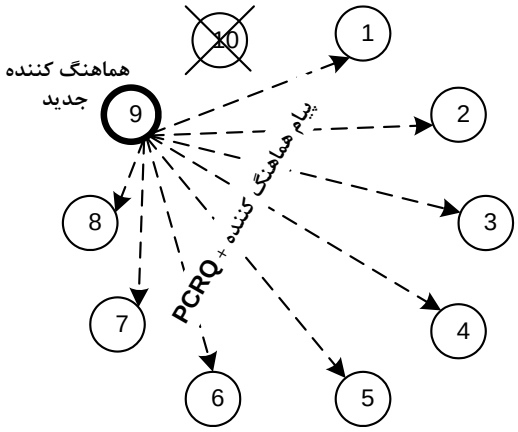
طبق آنچه در فوق تشریح شد، مزیت به‌کارگیری این رویکرد این است که فقط یک فرایند فعال درگیر انتخاب هماهنگ‌کننده می‌شود و تعداد پیام‌های تبادلی در انتخابات طبق رابطه (۱) به دست می‌آید، جایی که n تعداد فرایندهای فعال است که برابر یک هماهنگ‌کننده و n-1 فرایند فعال دیگر است.

$$\begin{aligned}\text{Total Message} = {} & \text{Election message} + \text{Accept message} + \\ & (\text{Coordinator \& PCRQ}) \text{ message} + \\ & \text{PCRP message} + \text{MPC message} \qquad (1)\end{aligned}$$

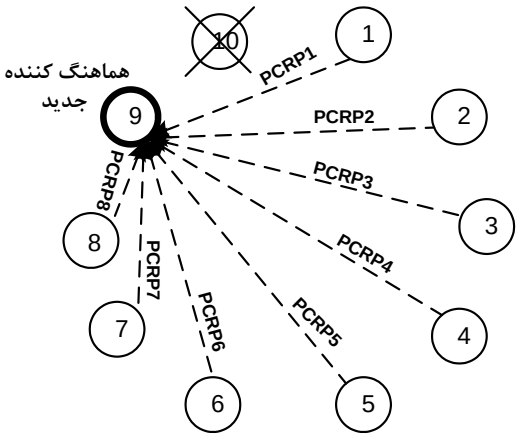
$$\text{Total Message} = 1+1+ (n-1) + (n-1) + (n-1) = 3n-1$$



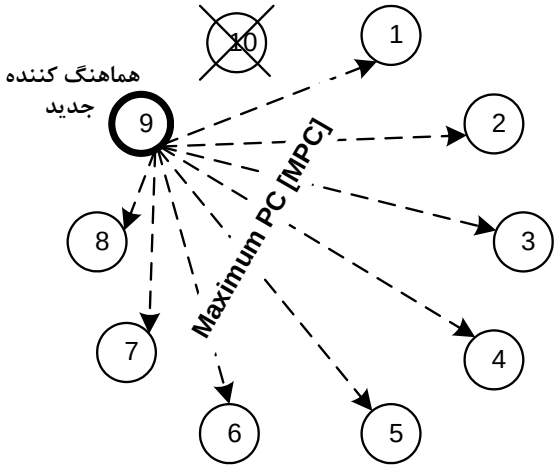
شکل ۵: حالتی از رخداد افزونگی انتخابات در رویکرد انتظار-مصلحتی



شکل ۶: درخواست زمان انتظار-مصلحتی هر یک از فرایندهای فعال توسط هماهنگ‌کننده جدید (ارسال پیام PCRQ و پیام هماهنگ‌کننده در قالب یک پیام)



شکل ۷: دریافت زمان انتظار-مصلحتی فرایندهای فعال توسط هماهنگ‌کننده جدید



شکل ۸: ارسال ماکزیمم زمان انتظار-مصلحتی توسط هماهنگ‌کننده جدید به فرایندهای فعال

شکل ۹ شبه کد فرایند i را با به‌کارگیری رویکرد انتظار-مصلحتی نشان می‌دهد. شایان‌ذکر است، هر فرایندی که بازیابی و یا به‌عنوان فرایند جدید وارد سیستم می‌شود، پس از شناسایی هماهنگ‌کننده جاری (درصورتی‌که فعال باشد) باید علاوه بر محاسبه زمان انتظار-مصلحتی خود (با ارسال و دریافت پیام ECHO)، ماکزیمم زمان انتظار-مصلحتی (MPC) را نیز از هماهنگ‌کننده جدید درخواست کند.

```
All New or Recovered process:
   send an ECHO Message to current coordinator;
   request and receive Maximum Pending_Convenience_Time;
   update Pending_Convenience_Time(i);
   update Maximum_Pending_Convenience_Time(i);
Process (i):
  while (1)
   {
    if (Current Coordinator has crashed)
     {
       wait Maximum_Pending_Convenience_Time(i);
       if (index i is maximum) // process(i) is new leader itself
        {
          broadcast Coordinator&PCRQ Message to all active processes;
          wait to receive PCRP Message from all active  processes;
          broadcast MPC Message to all active processes;
        }
       else
        {
          send Election Message to a process with  maximum index;
          wait to receive Accept message;
        }
     }
    if (receives Election Message) //index i is  maximum
     {
       send an Accept Message to the sender of the  first Received
            Election Message;
       broadcast Coordinator&PCRQ Message to all active processes;
       wait to receive PCRP Message from all active  processes;
       broadcast MPC Message to all active processes;
     }
    if (receives Coordinator&PCRQ Message)
     {
       send PCRP Message to the Coordinator;
       wait to receive MPC Message;
       update Maximun_Pending_Convenience_Time (i);
       update LEID(i)  field with ID of new coordinator;
     }
    Reminding Section;
   }
```

شکل ۹: شبه کد توسعه الگوریتم بولی اصلاح‌شده با رویکرد انتظار-مصلحتی

البته قابل‌ذکر است که این رویکرد نیز به‌نوبه خود دارای محدودیت‌هایی نیز است. به‌عنوان‌مثال، ممکن است بیش از یک فرایند در سیستم وجود داشته باشد که همگی دارای دقیقاً زمان تأخیر ماکزیمم یکسانی داشته باشند و دقیقاً در یک زمان یکسان از سقوط هماهنگ‌کننده مطلع و اقدام به آغاز انتخابات کنند، که این خود موجب افزونگی انتخابات خواهد شد. به‌هرحال، این شرایط ممکن است خیلی به‌ندرت در سیستم اتفاق بیافتد. از طرفی، هماهنگ‌کننده جاری باید در فواصلی منظم با ارسال پیام PCRQ، زمان انتظار-مصلحتی هریک از فرایندهای فعال را درخواست کند و سپس با ارسال پیام MPC، ماکزیمم زمان انتظار-مصلحتی را به فرایندهای فعال ابلاغ کند، زیرا ممکن است در حین عملکرد نرمال سیستم، برخی از فرایندهای فعال به دلایلی سقوط کنند.

۳-۲- رویکرد تعداد-انتخابات

در رویکرد تعداد-انتخابات هر فرایند دارای یک متغیر *تعداد-انتخابات*[۱۹] است. فرایند i به‌محض اینکه از سقوط هماهنگ‌کننده جاری مطلع شود، اگر شناسه خود فرایند i ماکزیمم باشد، یک واحد به متغیر تعداد-انتخابات خود اضافه می‌کند. اگر مقدار این متغیر برابر یک باشد (Election Number=1)، خود فرایند i اولین آغازکننده انتخابات و همچنین کاندیدای هماهنگ‌کننده است، لذا یک پیام هماهنگ‌کننده (Coordinator Message) را ایجاد و به سایر فرایندهای فعال پخش همگانی می‌کند و سپس با دریافت پیام پاسخ (Reply Message) از کلیه

فرایندهای فعال، متغیر تعداد-انتخابات را صفر می‌کند. دلیل استفاده از پیام پاسخ، اجتناب از بن‌بست است که در زیربخش ۳-۲-۱ به آن پرداخته شده است. حال درصورتی‌که شناسه فرایند i ماکزیمم نباشد، یک پیام انتخابات (Election Message) به اولین فرایند فعال در سیستم با بزرگ‌ترین شناسه ارسال می‌کند و فرایند دریافت‌کننده پیام انتخابات نیز قبل از هر کاری یک واحد به متغیر تعداد-انتخابات خود اضافه می‌کند. اگر مقدار این متغیر برابر یک باشد (Election Number=1)، فرایند فرستنده (در اینجا، فرایند i) اولین فرایند آغازکننده انتخابات است، لذا فرایند گیرنده (با بزرگ‌ترین شناسه) یک پیام پذیرش انتخابات (Accept Message) برای فرایند فرستنده (در اینجا، فرایند i) ارسال می‌کند. سپس یک پیام هماهنگ‌کننده (Coordinator Message) را ایجاد و به سایر فرایندهای فعال پخش همگانی می‌کند، چراکه خود به‌عنوان هماهنگ‌کننده جدید با بزرگ‌ترین شناسه بین فرایندها است. حال اگر مقدار متغیر تعداد-انتخابات فرایند مذکور (فرایند با بزرگ‌ترین شناسه) بزرگ‌تر از یک باشد؛ به عبارتی پیام انتخابات (Election Message) را از فرایندی دیگر نیز دریافت کرده است، پاسخی به آن فرایند نمی‌دهد و مانع از وقوع افزونگی انتخابات می‌شود، زیرا قبلاً توسط فرایند i به‌عنوان هماهنگ‌کننده انتخاب‌شده است. بنابراین در این رویکرد نیز تضمین می‌شود که در هر شرایطی، فقط یک فرایند درگیر فرایند انتخاب هماهنگ‌کننده می‌شود. در توسعه الگوریتم بولی اصلاح‌شده، با به‌کارگیری این رویکرد یک پیام جدید بنام «*پیام پاسخ*» اضافه شده است تا مانع از وقوع بن‌بست در روند انتخابات هماهنگ‌کننده شود. بنابراین، تعداد پیام‌های انتخاباتی مطابق با رابطه (۲) به دست می‌آیند.

$$\text{Total Message} = \text{Election Message} + \text{Accept Message} + \text{Coordinator Message} + \text{Reply Message} \quad (۲)$$
$$\text{Total Message} = 1+1+ (n-1) + (n-1) = 2n$$

شکل ۱۰ شبه کد فرایند i را با به‌کارگیری رویکرد تعداد-انتخابات نشان می‌دهد، جایی که n تعداد کل فرایندهای فعال در سیستم است.

۳-۲-۱ بررسی بن‌بست در رویکرد تعداد-انتخابات

در الگوریتم‌های انتخاب هماهنگ‌کننده دو سناریو می‌توان مطرح کرد که در آن‌ها انتخابات رخ می‌دهد. در سناریو اول زمانی که سقوط هماهنگ‌کننده جاری توسط یکی از فرایندهای فعال تشخیص داده شود، انتخابات آغاز می‌شود. سناریو دوم را می‌توان این‌طور مطرح کرد که اگر فرایندی وارد سیستم شود؛ که این فرایند یا از قبل به هر دلیلی سقوط کرده و مجدد بازیابی شده است، یا اینکه یک فرایند جدید است، که این فرایند بایستی با ارسال یک پیام به همسایگان خود از شماره هماهنگ‌کننده جاری مطلع شود. اگر شماره هماهنگ‌کننده جاری از شماره فرایند کوچک‌تر باشد، این فرایند اقدام به برگزاری انتخابات می‌کند. در رویکرد تعداد-انتخابات، در سناریوهای فوق هنگامی انتخابات انجام می‌شود که متغیر تعداد-انتخابات فرایندهای فعال ازجمله هماهنگ‌کننده جدید (در سناریو دوم) دارای مقدار صفر باشد، به‌طوری‌که از وقوع افزونگی انتخابات و همچنین بن‌بست جلوگیری شود. همان‌طور که در الگوریتم شکل ۱۰ مشاهده می‌شود، هر فرایندی که بعد از سقوط، مجدد بازیابی و یا به‌عنوان فرایند جدید وارد سیستم می‌شود، اقدام به صفر کردن متغیر تعداد-انتخابات می‌کند (سناریو اول). حال زمانی که هماهنگ‌کننده جدید نیز پیام هماهنگ‌کننده (Coordinator Message) را برای معرفی خود به سایر فرایندهای فعال ارسال می‌کند، زمانی اقدام به صفر کردن متغیر تعداد-اتخابات خود می‌کند، که پیام پاسخ (Reply Message) را از همه فرایندهای فعال دریافت کند، تا تضمین شود همه فرایندها از انتخاب هماهنگ‌کننده جدید مطلع شده‌اند و اقدامی مکرر برای انتخابات نکنند. لذا در سناریو دوم نیز اگر انتخاباتی صورت گیرد، متغیر تعداد-انتخابات هماهنگ‌کننده جاری نیز دارای مقدار صفر است.

```
All active and new processes:
  Election Number (i) = 0;  // i = 1... n;
Process (i):
  while (1)
   {
    if (current coordinator has crashed)
      {
       if (index i is maximum) // process(i) is new leader itself
         {
           Election Number (i) ++;
           if (Election Number (i) ==1)
            {
              broadcast Coordinator Message to all active  processes;
              wait to receive Reply Message from all  active
                    processes;
             Election Number (i) =0;
            }
         }
       else
         {
           send Election Message to a process with maximum  index;
           wait to receive Accept Message;
         }
      }
    if (receives Election Message) // index i is  maximum
      {
       Election Number (i) ++;
       if (Election Number (i) ==1)
         {
           send an Accept Message to the sender  of the first
                 received Election Message;
           broadcast Coordinator Message to all active processes;
           wait to receive Reply Message from all active
                 processes;
          Election Number (i) =0;
         }
      }
    if (receives coordinator Message)
     {
       send a reply Message to the sender of Coordinator Message;
       update LEID(i)  field with ID of new coordinator;
     }
    Reminding Section;
  }
```

شکل ۱۰: شبه کد توسعه الگوریتم بولی اصلاح‌شده با رویکرد تعداد-انتخابات

۴- پیچیدگی پیامی در رویکردهای پیشنهادی

۴-۱- بررسی رویکرد انتظار-مصلحتی

- **بهترین حالت:** زمانی است که فرایندی با بالاترین شناسه در سیستم از سقوط هماهنگ‌کننده جاری مطلع می‌شود که در این حالت خود فرایند به‌عنوان هماهنگ‌کننده جدید انتخاب می‌شود. لذا تعداد پیام‌ها طبق رابطه (۳) محاسبه می‌شود.

$$\text{Total Message} = \text{Election message} + \text{Accept message} + (\text{Coordinator \& PCRQ}) \text{ message} + \text{PCPR message} + \text{MPC message} \quad (۳)$$
$$\text{Total Message} = 0+0+(n-2)+(n-2)+(n-2) = 3n-6 = O(n)$$

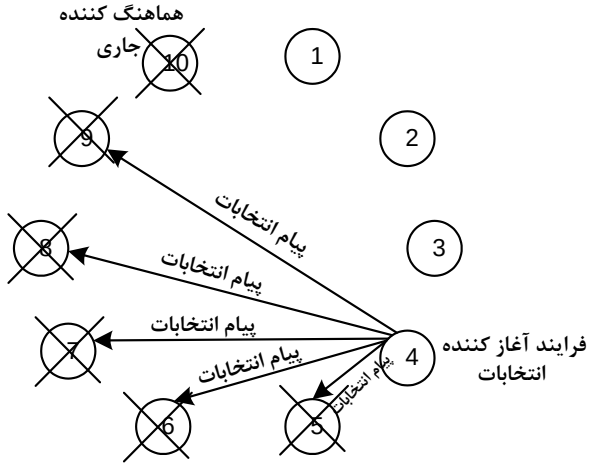
- **بدترین حالت:** زمانی رخ می‌دهد که فرایندی با کوچک‌ترین شناسه بفهمد که هماهنگ‌کننده جاری سقوط کرده و با ارسال یک پیام انتخابات به فرایندی با بزرگ‌ترین شناسه انتخابات را آغاز می‌کند و اگر این فرایند سقوط کرده باشد این عمل را برای دومین و بزرگ‌ترین فرایند ارسال می‌کند و اگر این فرایند هم سقوط کرده باشد این کار را تا زمانی انجام می‌دهد که خود فرایند آغازکننده خودش را به‌عنوان هماهنگ‌کننده جدید معرفی کند. حال اگر سیستم را در همین حال در نظر بگیریم، بدترین حالت و بیشترین تبادل پیام را داریم. درواقع در

بدترین حالت فوق عملاً حالتی در سیستم در نظر گرفته شده است که در آن به‌جز فرایند آغازکننده انتخابات همه فرایندهای با شناسه بزرگ‌تر سقوط کرده‌اند.

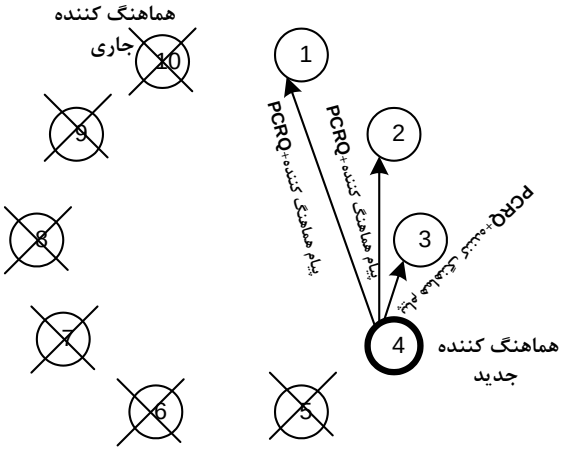
حال سیستمی را در نظر بگیرید که دارای ۱۰ فرایند است که فرایند با شناسه ۱۰ (هماهنگ‌کننده جاری) سقوط کرده است. پیام‌هایی که در سیستم بین فرایندهای فعال به ترتیب برای انتخاب هماهنگ‌کننده تبادل می‌شود در رابطه (۴) آورده شده است.

$$\text{Total Message} = \text{Election message} + \text{Accept message} + (\text{Coordinator \& PCRQ}) \text{ message} + \text{PCPR message} + \text{MPC message} \quad (۴)$$

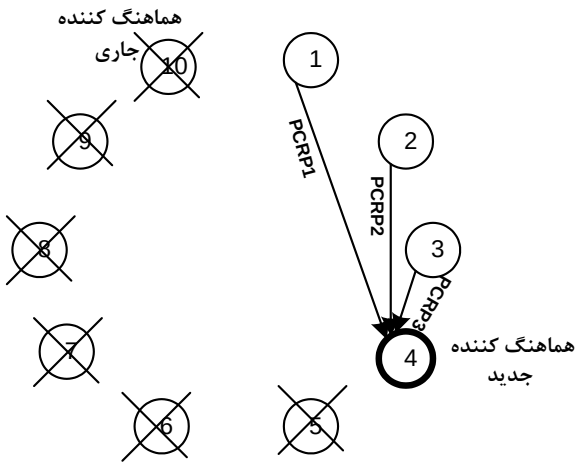
Process (1): Total Message = (n-2)+0+0+0+0 = n-2
Process (2): Total Message = (n-3)+0+1+1+1 = n
Process (3): Total Message = (n-4)+0+2+2+2 = n+2
Process (4): Total Message = (n-5)+0+3+3+3 = n+4
Process (5): Total Message = (n-6)+0+4+4+4 = n+6
Process (6): Total Message = (n-7)+0+5+5+5 = n+8
Process (7): Total Message = (n-8)+0+6+6+6 = n+10
Process (8): Total Message = (n-9)+0+7+7+7 = n+12
Process (9): Total Message = (n-10)+0+8+8+8 = n+14
Process (i): Total Message = (n-i-1)+3(i-1) = n + 2i – 4



شکل ۱۱: آغاز انتخابات توسط فرایند ۴ پس از سقوط هماهنگ‌کننده جاری
(تعداد پیام‌ها=n-5=10-5=5)

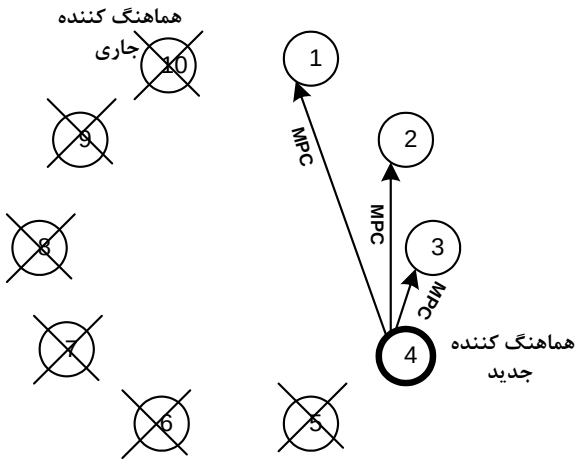


شکل ۱۲: ارسال پیام هماهنگ‌کننده+PCRQ توسط هماهنگ‌کننده جدید به ۳ فرایند فعال



شکل ۱۳: دریافت پیام PCRP از سه فرایند فعال توسط هماهنگ‌کننده جدید

شکل ۱۱ پیام‌های تبادلی را در یکی از حالات سیستم فوق بین فرایند ۴ و سایر فرایندهای فعال برای انتخاب هماهنگ‌کننده نشان می‌دهد. همان‌طور که در شکل نشان داده شده است، فرایند ۴ بعد از تشخیص سقوط هماهنگ‌کننده جاری و پایان محلت ماکزیمم زمان انتظار-مصلحتی، اقدام به برگزاری انتخابات می‌کند. لذا، پیام انتخاباتی را به فرایندی با بزرگ‌ترین شناسه، فرایند ۸، ارسال می‌کند و چون فرایند ۸ نیز سقوط کرده است، فرایند ۴ پاسخی دریافت نمی‌کند. مجدداً پیام انتخاباتی را به فرایند ۷ ارسال می‌کند و در صورت عدم دریافت پاسخ این عمل را تا فرایند ۵ تکرار می‌کند. بنابراین، تعداد پیام‌های انتخاباتی که ارسال می‌شود برابر با n-5=10-5=5 و تعداد پیام‌های پذیرش صفر است. در این مرحله فرایند ۴ خود به‌عنوان هماهنگ‌کننده جدید اقدام به ارسال پیام هماهنگ‌کننده بعلاوه پیام PCRQ به فرایندهای فعال می‌کند (شکل ۱۲ را ببینید). شکل‌های ۱۳ و ۱۴ نیز به ترتیب پاسخ فرایندهای فعال به هماهنگ‌کننده جدید و ارسال پیام MPC توسط هماهنگ‌کننده جدید به آن فرایندها را نشان می‌دهند.



شکل ۱۴: ارسال پیام MPC توسط هماهنگ‌کننده جدید به سه فرایند فعال

حال برای n فرایند در سیستم و با فرض اینکه احتمال انتخاب هریک از فرایندها به‌عنوان هماهنگ‌کننده جاری، p_i، یکسان و برابر مقدار ثابت $1/(n-1)$ است، و c_i تعداد پیام‌های ارسالی از طرف فرایند P_i برای سایر فرایندها تا انتخاب خود فرایند P_i باشد، پیچیدگی پیامی روش پیشنهادی طبق رابطه (۵) در بدترین حالت برابر $O(n)$ محاسبه می‌شود.

$$\sum_{i=1}^{n} c_i p_i = \sum_{i=1}^{n} \frac{1}{n-1}(n+2i-4) = \frac{1}{n-1}(n-2) + \frac{1}{n-1}n + \cdots + \frac{1}{n-1}(3n-4) = O(n) \quad (۵)$$

- **حالت میانگین:** در این حالت کلیه فرایندها، غیر از فرایند هماهنگ‌کننده، فعال هستند (n-1 فرایند فعال هستند) و فرایند i سقوط هماهنگ‌کننده را تشخیص می‌دهد و اقدام به آغاز انتخابات می‌کند. تعداد پیام‌هایی که بین فرایندها در انتخاب هماهنگ‌کننده تبادل می‌شود از رابطه (۶) به دست می‌آید.

Total Message = Election message + Accept message +
(Coordinator & PCRQ) message +
PCPR message + MPC message
Total Message = 1 + 1 + n-2 + n-2 + n-2 = 3n-4 = O(n) (۶)

۴-۲- بررسی رویکرد تعداد-انتخابات

- **بهترین حالت:** زمانی است که فرایندی با بالاترین شناسه در سیستم از سقوط هماهنگ‌کننده جاری مطلع می‌شود که در این حالت خود فرایند به‌عنوان هماهنگ‌کننده جدید انتخاب می‌شود. لذا تعداد پیام‌ها طبق رابطه (۷) محاسبه می‌شود.

Total Message = Election Message + Accept Message
+ Coordinator Message + Reply Message (۷)
Total Message = 0 + 0 + (n-2) + (n-2) = 2n-4

- **بدترین حالت:** زمانی رخ می‌دهد که فرایندی با کوچک‌ترین شناسه بفهمد که هماهنگ‌کننده جاری سقوط کرده و با ارسال یک پیام انتخابات به فرایندی با بزرگ‌ترین شناسه انتخابات را آغاز می‌کند و اگر این فرایند سقوط کرده باشد این عمل را برای دومین بزرگ‌ترین فرایند ارسال می‌کند و اگر این فرایند هم سقوط کرده باشد این کار را تا زمانی انجام می‌دهد که خود فرایند آغازکننده خودش را به‌عنوان هماهنگ‌کننده جدید معرفی می‌کند. حال اگر سیستم را در همین حال در نظر بگیریم، بدترین حالت و بیشترین تبادل پیام را داریم. درواقع در بدترین حالت فوق عملاً حالتی در سیستم در نظر گرفته شده است که در آن به‌جز فرایند آغازکننده انتخابات همه فرایندهای با شناسه بزرگ‌تر سقوط کرده‌اند.

حال سیستمی را در نظر بگیرید که دارای ۱۰ فرایند است که فرایند با شناسه ۱۰ (هماهنگ‌کننده جاری) سقوط کرده است. پیام‌هایی که در سیستم بین فرایندها به ترتیب برای انتخاب هماهنگ‌کننده تبادل می‌شود در رابطه (۸) آورده شده است.

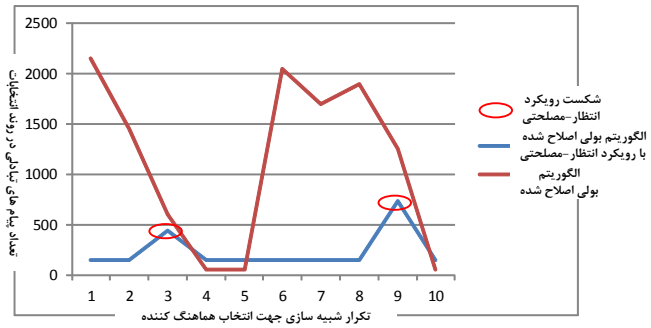
Total Message = Election Message + Accept Message
+ Coordinator Message + Reply Message (۸)
Process (1): Total Message = (n-2)+0+0+0 = n-2
Process (2): Total Message = (n-3)+0+1+1 = n-1
Process (3): Total Message = (n-4)+0+2+2 = n
Process (4): Total Message = (n-5)+0+3+3 = n+1
Process (5): Total Message = (n-6)+0+4+4 = n+2
Process (6): Total Message = (n-7)+0+5+5 = n+3
Process (7): Total Message = (n-8)+0+6+6 = n+4
Process (8): Total Message = (n-9)+0+7+7 = n+5
Process (9): Total Message = (n-10)+0+8+8 = n+6
Process (i): Total Message = n+i-3

حال برای n فرایند در سیستم و با فرض اینکه احتمال انتخاب هریک از فرایندها به‌عنوان هماهنگ‌کننده جاری، p_i، یکسان و برابر مقدار ثابت $1/(n-1)$ است، و c_i تعداد پیام‌های ارسالی از طرف فرایند P_i برای سایر فرایندها تا انتخاب خود فرایند P_i باشد، پیچیدگی پیامی روش پیشنهادی طبق رابطه (۹) در بدترین حالت برابر $O(n)$ محاسبه می‌شود.

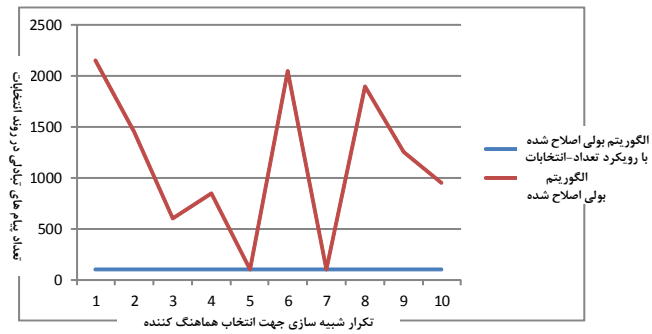
$$\sum_{i=1}^{n} c_i p_i = \sum_{i=1}^{n} \frac{1}{(n-1)}(n+i-3) = \frac{1}{n-1}(n-2) + \frac{1}{n-1}(n-1) + \cdots + \frac{1}{n-1}(2n-3) = \frac{2n^2-5n+3}{n-1} = O(n) \quad (۹)$$

- **حالت میانگین:** در این حالت کلیه فرایندها، غیر از فرایند هماهنگ‌کننده، فعال هستند (n-1 فرایند فعال هستند) و فرایند i سقوط هماهنگ‌کننده را تشخیص می‌دهد و اقدام به آغاز انتخابات می‌کند. تعداد پیام‌هایی که بین فرایندها در انتخاب هماهنگ‌کننده تبادل می‌شود از رابطه (۱۰) به دست می‌آید.

Total Message = Election Message + Accept Message
+ Coordinator Message + Reply Message
Total Message = 1 + 1 + n-2 + n-2 = 2n-2 = O(n) (۱۰)



شکل ۱۵: بررسی وقوع افزونگی انتخابات با به‌کارگیری رویکرد انتظار-مصلحتی در الگوریتم بولی اصلاح‌شده (n=50, Iteration=10)



شکل ۱۶: بررسی وقوع افزونگی انتخابات با به‌کارگیری رویکرد تعداد-انتخابات در الگوریتم بولی اصلاح‌شده (n=50, Iteration=10)

۵- ارزیابی رویکردهای پیشنهادی

در این بخش تعداد پیام‌های تبادلی در روند انتخابات در الگوریتم بولی اصلاح‌شده و همچنین در توسعه الگوریتم بولی اصلاح‌شده با به‌کارگیری هر یک از رویکردهای پیشنهاد شده در بخش ۳ مورد ارزیابی قرار گرفته است. در این ارزیابی، یک سیستم با ۵۰ فرایند در نظر گرفته شده است (n=50)، به‌طوری‌که فرایند با بزرگ‌ترین شناسه به‌عنوان هماهنگ‌کننده جاری ایفای نقش می‌کند. همچنین، با در نظر گرفتن حالت میانگین، به‌طوری‌که با سقوط هماهنگ‌کننده، یکی از فرایندهای فعال پس از تشخیص سقوط هماهنگ‌کننده جاری اقدام به آغاز انتخابات می‌کند، ارزیابی در ۱۰ مرتبه اجرای الگوریتم‌ها انجام شده است. با توجه به شکل ۱۵، مشاهده می‌شود که در الگوریتم بولی اصلاح‌شده، در اکثر موارد، بیش از یک فرایند هم‌زمان سقوط هماهنگ‌کننده جاری را تشخیص و اقدام به انتخابات کرده‌اند

و این وقوع افزونگی انتخابات منجر به افزایش ترافیک شبکه شده است، درحالی‌که با به‌کارگیری رویکرد انتظار-مصلحتی، فقط یک فرایند درگیر انتخابات شده است، مگر در اندک مواردی این رویکرد با شکست مواجه شده است، زیرا در این شبیه‌سازی، زمان تأخیر ماکزیمم ۱۰٪ از فرایندها دقیقاً یکسان در نظر گرفته شده است. اما همان‌گونه که در شکل ۱۶ مشاهده می‌شود، با به‌کارگیری رویکرد تعداد-اتخابات در الگوریتم بولی اصلاح‌شده، در همه موارد تکرار شبیه‌سازی، افزونگی انتخابات رخ نداده است و فقط یک فرایند آغازکننده انتخاب است که این خود باعث کاهش چشمگیر پیام‌های تبادلی و به دنبال آن کاهش ترافیک شبکه شده است.

۶- نتیجه‌گیری

در این مقاله دو رویکرد در توسعه الگوریتم بولی اصلاح‌شده پیشنهاد شده است تا از رخداد پدیده افزونگی انتخابات جلوگیری شود. هرچند در رویکرد اول که انتظار-مصلحتی نامیده شده است، علاوه بر تأخیری برابر ماکزیمم زمان انتظار-مصلحتی که در شروع روند انتخابات وجود دارد، سه پیام نیز به پیام‌های انتخاباتی در الگوریتم بولی اصلاح‌شده اضافه شده است، ولی همچنان پیچیدگی پیامی برابر $O(n)$ است، ولی با این مزیت که عدم رخداد افزونگی انتخابات تضمین شده است. در رویکرد دوم، تعداد-انتخابات، علاوه بر حفظ تعداد پیام‌های انتخاباتی در الگوریتم بولی اصلاح‌شده و پیچیدگی پیامی برابر $O(n)$، تأخیر رویکرد اول نیز حذف شده است. این رویکرد نیز تضمین می‌کند که فقط یک فرایند در روند انتخابات شرکت کند و درنتیجه از افزونگی انتخابات جلوگیری می‌کند.

۷- مراجع

[1] G. L. Lan, "Distributed System–Towards a Formal Approach", *IFIP world congress*, pp.155-160, 1977.

[2] E. J. Chang, R. Roberts, "An improved algorithm for the decentralized extrima-finding in circular configurations of processes," *Communication of ACM*, vol. 22, no.5, pp. 281-283, 1979.

[3] H. Garcia-Molina, "Elections in Distributed Computing System", *IEEE Transaction on Computer*, vol. 31,no.1, pp. 48-59, 1982.

[4] M. M. Rahman, A. Nahar, "Modified Bully Algorithm using Election Commission", *Journal of Computing*, vol. 1 no.3, pp. 439-446, 2009.

[5] A. Arghavani, E. Ahmadi, A. T. Haghighat, "Improved Bully Election Algorithm in Distributed Systems", *proceedings of the 5th international conference on information technology & multimedia*, pp. 1-6, 2011.

[6] R. Gajre, L. Ragha, "Optimized Bully Election Method for Selection of Coordinator Process and Recovery of Crashed Process", *International Journal of Scientific and Research Publications*, vol 3, no. 5, pp. 1-4, 2013.

[7] A. Datley, B. Patel, S. Soni, "Amelioration on Coordinator Election Algorithm in synchronous distributed system", *International Journal of Engineering Research & Technology*, vol. 2, no 11, pp. 2375-2378, 2013.

[8] M. S. Kordafshari, M. Gholipour, M.Jahanshahi, A.T. Haghighat, "Modified bully election algorithm in distributed system", *WSEAS Conferences, Cancun*, Mexico, pp.11-14, 2005.

[9] A. Silberschatz, P. Galvin, *Operating Systems Concetsp* ,4th EditionWesley publishing, 1994.

[10] Q. E. K. Mamun, S. M. Masum, "Modified Bully Algorithm for electing coordinator in distributed system", *proceedings of 3rd international conference on software Engineering, parallel & distributed system*, pp.13-15, 2004.

[11] P. D. Bhute, "A New Approach for Electing a Coordinator in Anonymous System", *International Journal of Computer Science and Information Technologies*, vol. 5, no. 3, pp. 3883-3886, 2014.

[12] M. Al Refai, "Dynamic Leader Election Algorithm in 2D Torus Network with Multi Links Failure", *International Journal of Computer Science Trends and Technology*, vol. 2, no. 5, pp. 150-156, 2014.

[13] M. Gholipour, M. S. Kordafshari, M. Jahanshahi, A. M. Rahmani, "A New Approach For Election Algorithm in Distributed Systems", *Second International Conference on Communication Theory, Reliability, and Quality of Service*, pp.70-74, 2009.

[14] P.K. Sinha, Distributed Operating Systems Concepts and Design, Prentice-Hall of India private Limited, 2008.

رضا نورمندی پور مدرک کارشناسی خود را در رشته مهندسی کامپیوتر گرایش سخت‌افزار در سال ۱۳۸۰ از دانشگاه شهید باهنر کرمان ، و کارشناسی ارشد و دکترای خود را در رشته مهندسی کامپیوتر گرایش معماری سیستم‌های کامپیوتری به ترتیب در سال‌های ۱۳۸۳ و ۱۳۸۹ از دانشگاه آزاد اسلامی واحد علوم و تحقیقات تهران اخذ کرده‌اند. ایشان اکنون استادیار و عضو هیئت‌علمی دانشگاه آزاد اسلامی واحد سیرجان هستند. زمینه تحقیقاتی مورد علاقه ایشان طراحی، آزمون و آزمون‌پذیری سیستم‌ها و شبکه های روی تراشه و هسته‌های تعبیه‌شده حافظه، سیستم‌های توزیع‌شده و مباحث زمان‌بندی و انحصارمتقابل است.

آدرس پست الکترونیکی ایشان عبارت است از:

noormandi_r@iausirjan.ac.ir

[1] Coordinator
[2] Election Algorithm
[3] Election Redundancy
[4] Message Passing
[5] Remote Procedure Call
[6] Coordinator
[7] Crash
[8] Election
[9] Election Message
[10] Accept Message
[11] Coordinator Message
[12] Leader ID
[13] Pending Convenience
[14] Number-of-Election
[15] Recovery
[16] Pending Convenience Request message
[17] Pending Convenience Reply message
[18] Maximum Pending Convenience message
[19] Election Number

The CSI Journal on Computing Science and Information Technology

The CSI Journal on
Computing Science and
Information Technology
Vol. 17, No.2, 2020

Extending Modified Bully Algorithm to Eliminate Election Redundancy in Coordinator Election in Distributed Systems

Reza Nourmandi-Pour

Department of Computer Engineering, Sirjan branch, Islamic Azad University, Sirjan, Iran

Abstract

In distributed systems which comprises of many processes and resources, process synchronization is unavoidable for issues such as mutual exclusion and allocation of shared resources. Coordinator process plays an essential role to processes synchronization. Therefore, coordinator election is not only important issue in distributed systems, but also in interconnecting networks, mutual exclusion algorithms, shared resources allocation, etc. One of the challenges in election coordinator algorithms is reducing the number of messages interchanging between processes, which has a direct influences over network traffic. Although much efforts has gone into reducing the number of interchanging messages between processes, but election redundancy is one of the limitations that coordinator election algorithms face. The main idea of this paper is to propose new approaches to extend the modified Bully algorithm in order to eliminate election redundancy. Hence, not only message complexity of the modified Bully algorithm equals to O(n), but also guarantees that only one process is involved in the election. This therefore, results in eliminating election redundancy and subsequently reduces network congestion and traffic.

Keywords: Distributed Systems, Coordinator Election, Message Passing, Modified Bully Algorithm, Election Redundancy