دوفصلنامه محاسبات و سامانه های توزیع شده
سال هفتم، شماره ۱، شماره پیاپی ۱۳، صفحه ۲۴ تا ۳۲، سال ۱۴۰۳

بررسی تحمل‌پذیری یک الگوریتم پخش علیّتی در برابر گم شدن پیام

لیلا ناموری تازه‌کند

دانشکده‌ی مهندسی برق و کامپیوتر، دانشگاه تبریز، تبریز، ایران

چکیده

در این پژوهش، الگوریتم LCUC از نظراحتمال نقض ترتیب علیتی و احتمال گم شدن پیام ها در مواجه با ارتباطات غیر قابل اعتماد تحلیل شده است. این الگوریتم با هدف کاهش پیچیدگی ارتباطی و افزایش تحمل‌پذیری در برابر ارتباطات نامطمئن طراحی شده است .الگوریتم LCUC برای سیستم‌های توزیع‌شده ناهمگام ارائه شده و قادر است ضمن حفظ کارایی در شرایط عادی، پیام‌های گم‌شده را در مواقع مواجه با ارتباطات غیر قابل اعتماد از گره های سالم دریافت کند. به منظور تحلیل تأثیر نقص‌های ارتباطی و فرآیندهای معیوب، از مدل توزیع پواسون برای بررسی احتمال نقض ترتیب علیتی استفاده شده است. تحلیل ها نشان می‌دهد که الگوریتم LCUC با کاهش بار ارتباطی تا مرتبه O(n) و حفظ صحت تحویل پیام‌ها حتی در حضور نقص‌های ارتباطی، عملکردی بهینه ارائه می‌دهد. علاوه بر این، با ارزیابی رفتار الگوریتم در شرایط خرابی گره‌ها، این الگوریتم در کنار حفظ ترتیب علیتی از گم شدن پیام جلوگیری می کند.

کلمات کلیدی:

سیستم‌های توزیع شده، الگوریتم پخش علیّتی، ارتباطات غیر قابل اعتماد، فرآیند سالم، فرآیند معیوب، تابع توزیع پواسون.

تاریخچه‌ی مقاله:

تاریخ ارسال: ۱۴۰۳/۰۱/۲۷

تاریخ اصلاحات: ۱۴۰۳/۰۵/۰۲

تاریخ پذیرش: ۱۴۰۳/۰۵/۲۲

تاریخ انتشار: ۱۴۰۳/۰۶/۳۰

ایمیل نویسنده مسئول: l.namvari@tabrizu.ac.ir

۱. مقدمه

این پژوهش بر روی طراحی و تحلیل یک الگوریتم پخش علیتی متمرکز است که به‌طور خاص به دو چالش اساسی در سیستم‌های توزیع‌شده می‌پردازد: کاهش پیچیدگی ارتباطی و تضمین تحمل‌پذیری در برابر ارتباطات غیرقابل اعتماد. با توجه به رشد سریع سیستم‌های توزیع‌شده و نقش کلیدی آن‌ها در حوزه‌هایی نظیر مدیریت داده‌های توزیع‌شده، اجرای انحصار متقابل و ارتباطات همزمان، ضرورت توسعه الگوریتم‌های کارآمد و قابل اطمینان برای حفظ روابط علت و معلولی بین پیام‌ها بیشتر از گذشته احساس می‌شود. این پژوهش الگوریتم جدیدی با نام LCUC ارائه می‌دهد که با الهام از الگوریتم‌های موجود و بهبود نقاط ضعف آن‌ها طراحی شده است [۳ـ۱].

امروزه، استفاده از سیستم‌های ارتباطی توزیع‌شده به‌طور فزاینده‌ای افزایش یافته است. این سیستم‌ها در انجام وظایف متنوعی نظیر سرگرمی، ارتباطات شخصی، فعالیت‌های علمی و امور تجاری نقش کلیدی دارند. در میان الگوریتم‌های ارسال پیام برای چنین سیستم‌هایی، پخش علیتی به‌ویژه در محیط‌های ناهمگام اهمیت خاصی دارد. این روش نقشی اساسی در حوزه‌هایی همچون تضمین سازگاری داده‌ها در سیستم‌های حافظه مشترک توزیع‌شده، مدیریت پایگاه داده، اجرای انحصار متقابل، ضبط وضعیت لحظه‌ای و ارسال و دریافت پیام در شبکه‌های توزیع‌شده ایفا می‌کند [۱]. ویژگی مهم پخش علیتی این است که پیام‌ها بدون توجه به ترتیب رسیدن، مطابق با روابط علت و معلولی در سیستم‌های ناهمگام تحویل داده می‌شوند [۱]. در این الگوریتم، هر فرآیند تنها پس از دریافت و تحویل پیام‌های ارسال‌شده توسط سایر فرآیندها اقدام به ارسال پاسخ می‌کند [۲]. با توجه به اهمیت الگوریتم‌های پخش علیتی، کاهش پیچیدگی ارتباطی و تضمین ارسال پیام در شرایط ارتباطات غیرقابل اعتماد از چالش‌های کلیدی است. تاکنون الگوریتم‌های متعددی برای بهبود

عملکرد پخش علیتی ارائه شده است. بسیاری از این الگوریتم‌ها دارای پیچیدگی ارتباطی $O(n^2)$ هستند.

بیشتر الگوریتم‌های موجود برای پخش علیتی پیچیدگی ارتباطی $O(n^2)$ دارند. Kshemkalyani و همکاران [۳] الگوریتم بهینه‌ای را پیشنهاد می‌دهند که با کانال‌های غیر_FIFO[۱] کار می‌کند. Gambhire و همکاران [۴] الگوریتم ترتیب علیتی را از نظر هزینه‌های سربار پیام و هزینه‌های ذخیره‌سازی پیام بهینه‌سازی کردند اما هنوز پیچیدگی ارتباطی $O(n^2)$ دارند. الگوریتم‌هایی که ارتباطات قابل اعتماد را پشتیبانی می‌کنند را می‌توانید در [۵_۷] مشاهده کنید که پیچیدگی ارتباطی $O(n^2)$ دارند. Auvolat و همکاران الگوریتمی را که در [۹] ارائه شده است برای یک سیستم با حداقل $t < n/3$ فرآیند بیزانتین/خائن[۲] که به طور تصادفی رفتار می‌کنند، توسعه دادند. از بین الگوریتم‌های پخش علیتی پیشنهاد شده برای فرآیندهای صحیح تنها الگوریتم ارائه شده توسط Mostefaoui و همکاران [۹] حداکثر پیچیدگی $O(n)$ را دارد. با این حال این الگوریتم برای سیستم هایی با ارتباطات غیر قابل اعتماد[۳] مناسب نیست.

در این مقاله، یک الگوریتم به نام LCUC معرفی می‌شود که از دو عبارت "پیچیدگی ارتباطی کم" و "ارتباطات غیرقابل اعتماد" الهام گرفته است. LCUC یک الگوریتم ارسال پیام است که رابطه‌ی علت و معلولی پیام‌ها را تضمین می‌کند. الگوریتم LCUC بر پایه‌ی الگوریتم پخش علیتی [۹] ارائه شده است. در این الگوریتم، فرآیند گیرنده موقع ارسال پیام جدید مثلاً m، آخرین پیام دریافت شده از هر فرآیند دیگر را به انتهای پیام جدید مورد ارسال پیوست می‌کند و این مسئله باعث می‌شود حتی اگر ارتباط بین چند فرآیند قطع شود، باز هم پیام m توسط فرآیندهای دیگر دریافت شود. در ادامه، ثابت می‌کنیم که الگوریتم LCUC دارای پیچیدگی ارتباطی از مرتبه $O(n)$ است و حتی در صورت بروز مشکلات در ارتباطات، این الگوریتم تحویل پیام‌های پخش شده توسط فرآیندها را تضمین می‌کند. علاوه بر این، در ادامه پژوهش، تأثیر خرابی فرآیندها به‌عنوان یک عامل احتمالی مورد بررسی قرار می‌گیرد و احتمال گم‌شدن پیام‌ها و تأثیر آن بر نقض ترتیب علیتی پیام‌ها برای نرخ‌های مختلف تبادل تحلیل می‌شود.

سازماندهی این مقاله به شرح زیر است: بخش دوم به بیان مبانی تحقیق و مفاهیم پایه‌ای می‌پردازد و کارهای گذشته را مرور می‌کند، سپس به معرفی یک الگوریتم پخش علیتی پایه می‌پردازد. در بخش سوم، الگوریتم پیشنهادی به‌طور دقیق توضیح داده می‌شود. در بخش چهارم درستی الگوریتم اثبات و ویژگی‌های آن تحلیل می‌شود. در بخش پنجم احتمال خرابی فرآیندها و تاثیر آن بر نقض ترتیب علیّتی تحلیل می‌شود. در پایان، بخش ششم نتیجه‌گیری این کار را بررسی می‌کند.

۲. مبانی تحقیق

در این بخش ابتدا الگوریتم‌های پخش علیتی گذشته را معرفی می‌کنیم، سپس ویژگی‌ها و جزئیات یکی از الگوریتم‌های اخیر [۹] را که مبنای الگوریتم پیشنهادی این مقاله است را شرح می‌دهیم.

۲_۱. مروری بر ادبیات گذشته

Birman و همکاران پخش علیتی را در سیستم‌های توزیع شده مقاوم در برابر خطا معرفی کردند [۱۰]. در سال ۱۹۹۱، این الگوریتم برای اولین بار با موفقیت در پروژه ISIS استفاده شد [۱۱] . Birman و همکاران یک پروتکل مبتنی بر پخش علیتی کارآمد و قابل اعتماد برای ارتقاء سیستم ISIS پیشنهاد شد که می‌تواند به آسانی به پخش کاملاً مرتب توسعه یابد [۱۲]. معرفی کاملی از ارتباطات فرآیندی در سیستم‌های توزیع شده و نمونه‌های برنامه‌های کاربردی این ارتباطات ارائه می‌دهد. همچنین، وی خواص آن‌ها را با تحلیل ترتیب کلی و علتی بررسی می‌کند. سپس، توضیح می‌دهد که چرا این الگوریتم‌ها در سیستم ISIS استفاده می‌شوند و به انتقادات Cheriton و Skeen پاسخ می‌دهد [۱۳].

Kshemkalyani و همکاران [۵] الگوریتم بهینه‌ای را که با کانال‌های غیر-FIFO کار می‌کند، پیشنهاد می‌دهند. این الگوریتم با کنترل و کاهش اندازه اطلاعات پیام بهینه‌سازی شده است. یک پیام فقط پیام‌های وابستگی علل لازم را حمل می‌کند. Gambhire و همکاران [۴] الگوریتم ترتیب علتی را از نظر هزینه اضافی پیام و هزینه ذخیره‌سازی پیام بهینه‌سازی کردند، اما هنوز پیچیدگی ارتباطی $O(n^2)$ دارند. الگوریتم‌هایی که ارتباط مطمئن را پشتیبانی می‌کنند، در [۵_۷] دیده می‌شوند. الگوریتم‌های پخش علتی بسیاری برای سیستم‌های ناهمزمان پیشنهاد شده‌اند، اما پیچیدگی ارتباطی آن‌ها $O(n^2)$ است. با این حال، پیچیدگی الگوریتم ارائه شده در [۹،۱۴]

---

۱ First-In- First-Out

۲ Byzantine

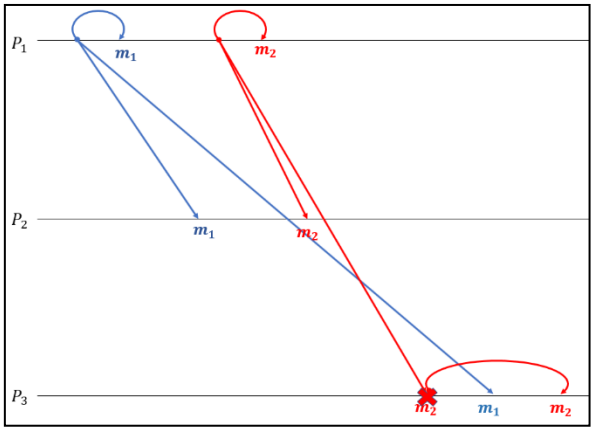
۳ Unreliable Communication

است. مفهوم علیّتی در انواع مختلفی استفاده می‌شود به شرح زیر: (۱)ارسال پیام بین فرآیندهای سیستم‌های توزیع شده که قبلاً ذکر شده است، (۲) ویرایش هماهنگ فایل‌ها [۱۵]، (۳) سازگاری علیّتی در داده‌های توزیع‌شده روی سرورها[۱۶]، (٤) الگوریتم پخش علیتی همچنین در سیستم‌های انتشار / اشتراک [۱۷] استفاده می‌شود. برای هر یک از این‌ها از روش‌های تضمین ترتیب علیّتی متفاوتی استفاده می‌شود که NamvariTazehkand و همکاران [۱۸] با مدل‌سازی و ارزیابی این روش‌ها آنها را مقایسه کرده‌اند.

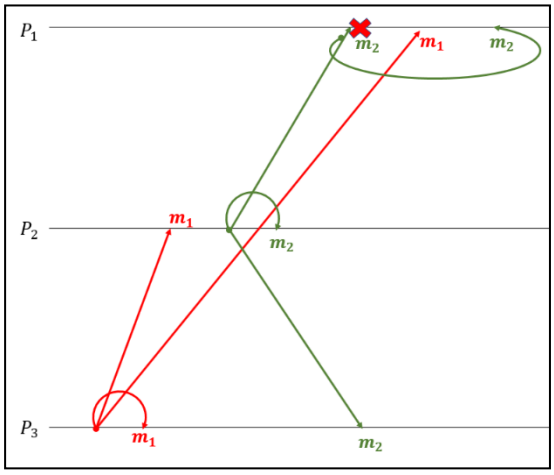
از آنجایی که مدل‌سازی و راستی‌آزمایی الگوریتم‌های توزیع شده امری ضروری است [۲۰ـ۱۹]، الگوریتم ارائه شده در [۹] با روش‌های رسمی مدل‌سازی و راستی آزمایی شده‌اند [۲۲ـ۲۱]. زیرا که الگوریتم ارائه شده در [۹] برای یک سیستم با حداقل تعداد ($t < n / 3$) فرآیند بیزانتین توسعه یافته است. به‌طوریکه که این فرآیندها به طور خودسرانه عمل می‌کنند، همچنین t تعداد فرآیندهای بیزانتین و n تعداد کل فرآیندهای سیستم را نشان می‌دهد [۸]. Guidec و همکاران الگوریتم پخش علیتی [۹] را برای شبکه‌های فرصت طلب بهینه‌سازی کرده‌اند که در آن برای هر گره عمر در نظر گرفته شده است [۱۸]. بسیاری از الگوریتم‌های پخش علیتی برای سیستم‌های ناهمزمان پیشنهاد شده‌اند اما پیچیدگی ارتباطی آنها $O(n^2)$ است و برای ارتباطات مطمئن مناسب هستند و درصورتی که ارتباطات قطع شود الگوریتم دریافت پیام را تضمین نمی‌کند.

۲-۲. معرفی یک الگوریتم پخش علیّتی پایه

یک الگوریتم پخش علیّتی شامل دو نوع پیام است، پیام‌های کاربردی[۱] و پیام‌های پروتکلی[۲] [۹]. فرض کنید یک فرآیند مثل P_i یک پیام کاربردی مثل m را در سطح برنامه پخش می‌کند. این پیام در سطح پروتکل به یک پیام پروتکلی سه تایی (m, i, seq_i) تبدیل می‌شود، به طوریکه m پیام کاربردی، i شناسه ی فرآیند فرستنده و seq_i شماره توالی پیام‌ها را نشان می دهد.



(الف)



(ب)

شکل ۱ : الف- پخش دو پیام توسط یک فرآیند. ب- پخش دو پیام توسط دو فرآیند جداگانه

این الگوریتم با چهار مشخصه ی مهم معرفی می شود که عبارتند از:

اعتبار[۳]: اگر یک فرآیند یک پیام m از یک فرآیند P_i تحویل بگیرد، P_i قبلاً m را پخش کرده است.

درستی[٤]: یک فرآیند یک پیام m را حداکثر یک بار تحویل می‌گیرد.

تحویل علیتی[٥]: اگر یک فرآیند یک پیام m را پخش کرده باشد، سپس بعداً یک پیام m' را پخش کند، یا یک فرآیند یک پیام m را تحویل

---

[۱] Application Messages
[۲] Protocol Messages
[۳] Validity
[٤] Integrity
[٥] Causal Delivery

بگیرد و بعداً یک پیام m' را پخش کند، هیچ فرآیندی پیش از تحویل گرفتن m' مطلقاً پیام m' را تحویل نمی‌گیرد.

پایان[1]: اگر یک فرآیند صحیح یک پیام را پخش کند، تمام فرآیندهای صحیح آن را تحویل می‌گیرند.

شکل ۱ مشخصه ی ۳ را به خوبی نشان می‌دهد. در ۱ ـ الف) فرآیند P_1 ابتدا پیام m_1 و سپس پیام m_2 را پخش کرده است، همه‌ی فرآیندها ابتدا m_1 و سپس m_2 را تحویل گرفته‌اند. در ۱ ـ ب) پیام‌های m_1 و m_2 به ترتیب توسط P_3 و P_2 پخش شده است و همه‌ی فرآیندها ابتدا m_1 و سپس m_2 را تحویل گرفته‌اند.

۲-۳. تعاریف

تعریف ۱. فرآیند سالم: فرآیندی که رفتار صحیح داشته و مطابق با استانداردهای الگوریتم کار کند، همچنین خراب نشود، به عنوان فرآیند سالم شناخته می‌شود.

تعریف ۲. فرآیند معیوب: فرآیندی که سالم نباشد را فرآیند معیوب می‌نامند (فرآیندی که از اجرای ادامه‌ی الگوریتم منصرف شده است، به عبارتی خراب شده است).

۲-٤. مفروضات

فرض کنید، مفروضات الگوریتم به شرح زیر باشد:

- سیستم از مجموعه‌ای از n فرآیند با نشان $P_1, \dots, P_n$ تشکیل شده است.
- هر فرآیند به صورت ناهمگام عمل می‌کند.
- مدل ارتباطی از کانال‌ها استفاده می‌کند.
- ارتباطات غیرقابل اعتماد هستند، به این معنی که ممکن است ارتباط قطع شود و پیام ارسالی گم شود، ولی پیام‌ها نه دوبار تکرار می‌شوند و نه دچار خرابی می‌شوند.
- تعداد پیام‌ها در میان فرآیندها متغیر است.
- همه‌ی فرآیندهای شرکت کننده سالم هستند.

۲-۵. نمادها

جدول ۱. نمادهای استفاده شده در الگوریتم LCUC را به طور کامل معرفی می‌کند.

۲-٦. الگوریتم LCUC

الگوریتم ۱ شبه کد مربوط به یک الگوریتم پخش علیتی با پیچیدگی ارتباطی کم و سازگار با ارتباطات غیر قابل اعتماد است. در خط ۱ و

۳. الگوریتم پیشنهادی

در این بخش، ابتدا فرضیات الگوریتم LCUC را شرح می‌دهیم، سپس نمادگذاری‌های لازم استفاده شده در الگوریتم را معرفی می‌کنیم. در انتها، توضیح مفصلی از الگوریتم LCUC ارائه می‌دهیم.

۲ متغیرهای مربوط به یک فرآیند P_i مقداردهی اولیه می‌شوند. یک فرآیند P_i پیام‌هایی که از فرآیندهای دیگر به ترتیب علیتی تحویل گرفته است را در $apmg_i$ ذخیره می‌کند که مقدار اولیه آن یک لیست خالی در نظر گرفته شده است (خط ۱). هر فرآیند پیام‌هایی که پخش می‌کند را با یک عدد طبیعی به نام seq_i شماره‌گذاری می‌کند به عنوان شماره توالی پیام هستند که مقدار اولیه آن صفر است (خط ۲). در این الگوریتم یک فرآیند هم در نقش فرستنده و هم در نقش گیرنده‌ی پیام است، زیرا که هر فرآیند پیامی که خودش پخش می‌کند را نیز تحویل می‌گیرد. وقتی فرآیند P_i پیامی را می خواهد پخش کند خط ۳ تا ۶ الگوریتم را اجرا می‌کند. وقتی این فرآیند می‌خواهد پیامی را تحویل بگیرد خط ۷ تا ۲۷ را اجرا می‌کند.

برای پخش پیام m ، فرآیند P_i ابتدا شماره توالی آن را یک واحد افزایش می‌دهد(خط ۳). سپس پیام پروتکلی MSG که حاوی پیام m و اطلاعات این پیام که به ابتدای آن $apmg_i$ پیوست شده (خط۵) را به همه‌ی فرآیندهای شرکت کننده با فراخوانی تابع $Broadcast()$ پخش می کند. فرآیند P_i در انتها لیست $apmg_i$ خالی می‌کند. برای دریافت پیام m توسط فرآیند P_i ، فرض کنید یک توالی از پیام‌ها از خود P_i یا فرآیند دیگری رسیده است که آخرین پیام، پیام مورد نظر برای پخش است و بقیه‌ی پیام‌های m_1 تا m_l پیام‌های پیوست شده به این پیام هستند (خط ۷).

[1] Termination

جدول ۱: نمادهای استفاده شده در الگوریتم LCUC

مفهوم	نماد
فرآیندهای شرکت کننده در سیستم	P_i و P_j
پیام کاربردی در سطح برنامه	m_i
مقدار عددی نشانگر شماره توالی پیام‌های پخش شده توسط P_i	seq_i
پیام کاربردی آماده‌ی پخش	(m, i, seq_i)
پیام‌های پیوست شده	$apmg_i$
تعداد کل فرآیندهای سیستم	n
مدت زمان تخمینی برای دریافت پیام در صورت وجود ارتباط مطمئن	τ
نماد الصاق پیام‌های پیوستی به انتهای پیام پروتکلی	∪
توالی خالی از پیام	{}
پیام پروتکلی	$MSG = apmg_i \cup (m, i, seq_i, t)$

الگوریتم ۱ پخش علیّتی با پیچیدگی ارتباطی کم برای ارتباطات غیر قابل اعتماد

```
init:
1)  apmg_i ← {}
2)  seq_i ← 0
    When C-broadcast (m) is invoked by process P_i do
3)      seq_i ← seq_i + 1 ;
4)      Broadcast (MSG);
5)      Let MSG be apmg_i ∪ (m , i , seq_i);
6)      apmg_i ← {}

When received (MSG) from P_j do
7)  Let MSG be (m_1, i_1, seq_1),..., (m_{l-1}, i_{l-1}, seq_{l-1}) , (m_l, i_l, seq_l ) , (m , i , seq_i )
8)      if P_j previously has been C-Delivered (m) then
9)          Drop(m);
10)     else
11)         For x = 1 to l − 1 do
12)             if MSG contain m_x that not yet C-Delivered then
13)                 if P_j previously has been C-Delivered (m_{x-1} , i_x , seq_x − 1) then
14)                     delivers(m_x);
15)                     apmg_j ← apmg_j ∪ {(m_x , i_x , seq_x)}
16)                     delivers(m);
17)                     apmg_j ← apmg_j ∪ {(m , i , seq_i)}
18)                 else
19)                     Wait for τ time;
20)                     delivers(m);
21)                     apmg_j ← apmg_j ∪ {(m , i , seq_i)}
22)                 end if;
23)             else
24)                 Drop(m);
25)             end if;
26)         end For;
27) end if;
```

در خط ۸ بررسی می کند اگر پیام m قبلا توسط فرآیند P_i تحویل گرفته شده است، پیام m را رها می‌کند[1] و از تحویل آن خودداری می کند (خط ۹). در غیر اینصورت بررسی می‌کند که در بین پیام‌های پیوست شده آیا پیامی وجود دارد که با پیام m رابطه‌ی علیتی داشته باشد و فرآیند گیرنده هنوز تحویل نگرفته باشد (خطوط ۱۱ تا ۱۳). آن را در خط ۱۴ دریافت می‌کند و به لیست $apmg_i$ اضافه می‌کند (خط ۱۵). سپس پیام‌های بعد از آن را از توالی پیام‌های رسیده تحویل می‌گیرد و به $apmg_i$ اضافه می‌کند (خطوط ۱۶و ۱۷). در غیراینصورت به اندازه‌ی t صبر می‌کند در صورتی که پیام در این فاصله‌ی زمانی رسید آن را تحویل می‌گیرد (خطوط ۱۹ تا ۲۲). اگر پیام‌های قبلی را به دلیل قطع ارتباط دریافت نکرده باشد، پیام m را نیز رها می‌کند (خط ۲۴) زیرا اگر بدون دریافت پیام‌های قبلی پیام m را دریافت کند، ترتیب علیتی پیام‌ها نقض می‌شود. در صورتی که یک فرآیند پیام m را رها کرده و دریافت نکرده باشد، این پیام در پیوست پیام‌های دیگر دوباره به فرآیند P_i خواهد رسید و می‌تواند از لیست پیوست آن پیام مورد نظر را تحویل بگیرد.

۴. اثبات درستی و تحلیل الگوریتم

این بخش ویژگی‌های الگوریتم LCUC را اثبات می‌کند.

لم ۱. پیچیدگی ارتباطی الگوریتم LCUC به‌صورت O(n) است.

اثبات. زمانی که هر فرآیند پیام m را پخش می‌کند، پیام m به انتهای $apmg_i$ اضافه می‌شود (خطوط ۱۵، ۱۷ و ۲۱). از آنجا که P_i پس از دریافت پیام، اطمینان حاصل می‌کند که پیام m_x با $Seq_i - 1$ در $apmg_i$ وجود ندارد (خط ۱۳)، هر پیام پروتکل شامل حداکثر n پیام است، یک پیام برای هر فرآیند. بنابراین، پیچیدگی ارتباطی الگوریتم LCUC به‌صورت $O(n)$ است. □

لم ۲. در صورت قطع ارتباط و نرسیدن پیام به یک فرآیند، الگوریتم همچنان دریافت پیام را تضمین می‌کند.

اثبات. زمانی که یک فرآیند توالی از پیام‌ها را از فرآیند دیگری دریافت می‌کند، در صورتی که آن پیام را قبلا دریافت نکرده باشد، طبق خط ۱۴ آن رادریافت می‌کند. □

لم ۳. الگوریتم دریافت علیتی پیام‌ها را تضمین می‌کند.

اثبات. طبق خط ۱۳ از الگوریتم، یک فرآیند قبل از دریافت یک پیام، دریافت پیام‌های قبلی با شماره توالی کوچکتر را بررسی می‌کند. □

قضیه‌ی ۱. الگوریتم LCUC یک الگوریتم پخش علیتی با پیچیدگی ارتباطی کم و مناسب برای ارتباطات غیر قابل اعتماد است.

اثبات. این قضیه به‌طور مستقیم از لم‌های ۱، ۲ و ۳ نتیجه می‌شود.□

۵. تحلیل احتمالاتی نقض ترتیب علیّتی پیام‌ها

این بخش ابتدا خراب شدن یک فرآیند را به عنوان یک رویداد معرفی می‌کند. سپس احتمال گم شدن پیام‌ها را بر اساس احتمال وقوع آن رویداد تحلیل می‌کند.

محاسبه‌ی میانگین نرخ خرابی: فرض کنید c رویداد خرابی یک فرآیند باشد و s تعداد فرآیندهایی که خراب می‌شوند. بازه زمانی $[t, t+\tau)$ را در نظر بگیرید، که در آن τ نشان‌دهنده یک بازه زمانی معیّن است. پارامتر λ میانگین نرخ رویداد c در بازه τ را نشان می‌دهد، و s تعداد وقوع رویداد c در بازه زمانی τ است که λ از رابطه (۱) به دست می‌آید.

$$\lambda = \frac{s}{\tau} \qquad (۱)$$

به عنوان مثال ۲ فرآیند در ۲۴ دقیقه خراب شوند، میانگین خرابی فرآیندها برابر با ۰/۰۸ است. $\lambda = \frac{s}{\tau} = \frac{2}{24} = 0.08$

اگر ۲ فرآیند در ۲۴ دقیقه خراب شوند، طبق رابطه‌ی (۲) اگر به طور متوسط هر ساعت ۵ فرآیند خراب شوند.

بنابراین $\lambda = \frac{2}{24} \times 60 = 5$.

$$\lambda = \frac{s}{\tau} \times \tau_1 \qquad (۲)$$

محاسبه‌ی از دست رفتن پیام بر اساس احتمال وقوع خرابی فرآیندها: با توجه به خط ۲۴ الگوریتم ۱ اگر فرآیندی پیام m را کنار بگذارد. همچنین فرآیندهای دیگری که قرار بود از پیوست پیام آنها پیام m دریافت شود نیز خراب شود و پیامی پخش نکنند. بنابراین احتمال از دست رفتن پیام وجود دارد. اگر بازه زمانی $[t, t+\tau)$، تعداد s فرآیند با نرخ خرابی λ خراب شوند، با توجه به توزیع پواسون $P(E = s)$. رابطه‌ی (۳) احتمال خرابی s فرآیند در بازه‌ی زمانی τ با نرخ λ خراب شوند و باعث از بین رفتن پیام می‌شوند.

[1] Drop

$$P(E = s) = \frac{e^{-(\tau\lambda)}.(\tau\lambda)^s}{s!} \qquad (۳)$$

معادله‌ی (۴) احتمال از دست رفتن پیام‌ها را با توجه به احتمال خرابی حداقل ۳ فرآیند در یک ساعت محاسبه می‌کند. نرخ خرابی فرآیند ها از رابطه‌ی (۲) بدست می‌آید. $\lambda = \frac{2}{15} \times 60 = 8$

$$P(s \geq 3) = 1 - P(s < 3)$$
$$= 1 - \sum_{i=0}^{2} \frac{e^{-(8)}.(8)^i}{i!} = 0.9862 \qquad (۴)$$

تاثیر احتمال از دست رفتن پیام بر احتمال نقض ترتیب علیّتی: فرض کنید یک فرآیند P_i ۵ پیام را طی نیم ساعت به ترتیب علیّتی از خود یا بقیه‌ی فرآیندها دریافت می‌کند، تعداد پیام‌هایی که یک P_j از P_i در یک ساعت دریافت می‌کند ۱۰ پیام است. رابطه‌ی (۵) احتمال نقض ترتیب علیّتی حداقل ۶ پیام اخیر در یک ساعت، به دلیل از دست رفتن بعضی از پیام‌ها را محاسبه می‌کند. نرخ نقض ترتیب علیّتی پیام‌ها از (۲) بدست می‌آید. $\varepsilon = \frac{5}{30} \times 60 = 10$

$$P(M \geq 6) = 1 - P(M < 6).P(s \geq 3)$$
$$= \left(1 - \sum_{i=0}^{5} \frac{e^{-(10)}.(10)^i}{i!}\right).P(s \geq 3) \qquad (۵)$$

جدول ۲ نشان می‌دهد که احتمال خراب شدن s فرآیند مستقیم با احتمال از دست دادن پیام‌ها مرتبط است. به‌عبارت‌دیگر، با افزایش احتمال خرابی حداقل s فرآیند، احتمال نقض ترتیب علیّتی پیام‌ها نیز افزایش می‌یابد و بالعکس.

جدول ۲: رابطه‌ی بین احتمال خرابی حداقل s فرآیند و احتمال نقض ترتیب علیّتی

$(\lambda = 8, \varepsilon = 10)$

$P(M = 18)$	$P(M = 16)$	$P(M = 14)$	$P(E \geq s)$	s
0.007089	0.021692	0.052060	0.999665	1
0.007070	0.021633	0.051920	0.996981	2
0.006994	0.021400	0.051361	0.986246	3
0.006791	0.020779	0.049870	0.957620	4
0.006385	0.019537	0.046889	0.900368	5
0.005735	0.017549	0.042118	0.808764	6
0.004869	0.014899	0.035757	0.686626	7
0.003879	0.011870	0.028488	0.547039	8
0.002889	0.008841	0.021219	0.407453	9
0.002009	0.006149	0.014757	0.283376	10

۶. نتیجه‌گیری

در این پژوهش، الگوریتم LCUC به‌عنوان راه‌حلی نوین برای مدیریت چالش‌های موجود در سیستم‌های توزیع‌شده ناهمگام معرفی شده است. این الگوریتم که با الهام از دو مفهوم کلیدی "کاهش پیچیدگی ارتباطی" و "مدیریت ارتباطات غیرقابل اعتماد" طراحی شده، قادر است رابطه علت و معلولی پیام‌ها را میان فرآیندها حفظ کند. LCUC با بهره‌گیری از پیام‌های الحاقی، دو چالش اصلی در سیستم‌های توزیع‌شده را مورد هدف قرار می‌دهد: کاهش پیچیدگی ارتباطی و تضمین ترتیب علیتی پیام‌ها حتی در مواجهه با ارتباطات نامطمئن. مطالعات گذشته در زمینه الگوریتم‌های پخش علیتی نشان می‌دهد که بسیاری از این روش‌ها با محدودیت‌هایی همچون پیچیدگی بالای $O(n^2)$ و نیاز به ارتباطات کاملاً قابل اعتماد مواجه هستند. در پاسخ به این محدودیت‌ها، الگوریتم LCUC با پیچیدگی ارتباطی پایین‌تر $O(n)$ طراحی شده است. این الگوریتم تضمین می‌کند که حتی در صورت بروز مشکلات ارتباطی، فرآیندها قادر به دریافت پیام‌ها به ترتیب علیتی خواهند بود. همچنین، تحلیل حضور فرآیندهای معیوب با استفاده از مدل توزیع پواسون نشان داده است که افزایش تعداد فرآیندهای خراب احتمال نقض ترتیب علیتی را افزایش می‌دهد. برای تحقیقات آینده، برنامه‌ریزی شده است تا الگوریتم LCUC با استفاده

از روش‌های فرمال مانند SPIN مدل‌سازی و تأیید شود. این رویکرد امکان تحلیل دقیق رفتار الگوریتم و اطمینان از صحت آن در سناریوهای مختلف را فراهم می‌آورد.

منابع:

[1] M. Raynal, Distributed algorithms for message-passing systems. Springer, 2013.

[2] M. Raynal, Fault-tolerant message-passing distributed systems: an algorithmic approach. springer, 2018.

[3] A. D. Kshemkalyani and M. Singhal, "An optimal algorithm for generalized causal message ordering," in Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing, 1996, p. 87.

[4] P. Gambhire and A. D. Kshemkalyani, "Evaluation of the optimal causal message ordering algorithm," in High Performance Computing—HiPC 2000: 7th International Conference Bangalore, India, December 17–20, 2000 Proceedings 7, 2000: Springer, pp. 83-95.

[5] A. D. Kshemkalyani and M. Singhal, "Necessary and sufficient conditions on information for causal message ordering and their optimal implementation," Distributed Computing, vol. 11, no. 2, pp. 91-111, 1998.

[6] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," in Concurrency: the Works of Leslie Lamport, 2019, pp. 179-196.

[7] C. Cachin, R. Guerraoui, and L. Rodrigues, Introduction to reliable and secure distributed programming. Springer Science & Business Media, 2011.

[8] A. Auvolat, D. Frey, M. Raynal, and F. Taïani, "Byzantine-tolerant causal broadcast," Theoretical Computer Science, vol. 885, pp. 55-68, 2021.

[9] A. Mostefaoui, M. Perrin, M. Raynal, and J. Cao, "Crash-tolerant causal broadcast in O (n) messages," Information Processing Letters, vol. 151, p. 105837, 2019.

[10] K. P. Birman and T. A. Joseph, "Reliable communication in the presence of failures," ACM Transactions on Computer Systems (TOCS), vol. 5, no. 1, pp. 47-76, 1987.

[11] K. Birman and R. Cooper, "The ISIS project: Real experience with a fault tolerant programming system," ACM SIGOPS Operating Systems Review, vol. 25, no. 2, pp. 103-107, 1991.

[12] K. Birman, A. Schiper, and P. Stephenson, "Lightweight causal and atomic group multicast," ACM Transactions on Computer Systems (TOCS), vol. 9, no. 3, pp. 272-314, 1991.

[13] K. Birman, "A response to Cheriton and Skeen's criticism of causal and totally ordered communication," ACM SIGOPS Operating Systems Review, vol. 28, no. 1, pp. 11-21, 1994.

[14] R. Vignesh, D. Deepa, P. Anitha, S. Divya, and S. Roobini, "Dynamic enforcement of causal consistency for a geo-replicated cloud storage system," International Journal of Electrical Engineering and Technology, vol. 11, no. 3, 2020.

[15] K. Huang, H. Wei, Y. Huang, H. Li, and A. Pan, "Byz-gentlerain: An efficient byzantine-tolerant causal consistency protocol," in Stabilization, Safety, and Security of Distributed Systems: 23rd International Symposium, SSS 2021, Virtual Event, November 17–20, 2021, Proceedings 23, 2021: Springer, pp. 495-499.

[16] J. P. de Araujo, L. Arantes, E. P. Duarte Jr, L. A. Rodrigues, and P. Sens, "VCube-PS: A causal broadcast topic-based publish/subscribe system," Journal of Parallel and Distributed Computing, vol. 125, pp. 18-30, 2019.

[17] F. Guidec, P. Launay, and Y. Mahéo, "Causal and Δ-causal broadcast in opportunistic networks," Future Generation Computer Systems, vol. 118, pp. 142-156, 2021.

[18] L. NamvariTazehkand, S. Pashazadeh, and A. Ebnenasir, "BCM-Broadcast: A Byzantine-Tolerant Causal Broadcast Algorithm for Distributed Mobile Systems," arXiv preprint arXiv:2401.16956, 2024.

[19] L. NamvariTazehkand and S. Pashazadeh, "Modeling, simulation, and evaluation of causal order assurance techniques in causal broadcast algorithms using timed colored petri nets," The Journal of Supercomputing, vol. 81, no. 1, pp. 1-33, 2025.

[20] L. NamvariTazehkand and A. Ebnenasir, "A Novel Approach for Specification and Verification of Symmetric Distributed Algorithms Using SPIN," in 2024 Third International Conference on Distributed Computing and High Performance Computing (DCHPC), 2024: IEEE, pp. 1-9.

[21] L. NamvariTazehkand and S. Pashazadeh, "Modeling and Verification of the Causal Broadcast Algorithm Using Colored Petri Nets," in 2024 Third International Conference on Distributed Computing and High Performance Computing (DCHPC), 2024: IEEE, pp. 1-12.

[22] L. NamvaiTazehkand, S. Pashazadeh 'Modeling and Formal Verification of a Distributed Mutual Exclusion Algorithm', Distributed Computing and Distributed Systems, Vol 6, Issue 2, Pages 1-11, 2024.

لیلا ناموری تازه‌کند دانشجوی دکتری مهندسی کامپیوتر در دانشکده مهندسی برق و کامپیوتر دانشگاه تبریز، تبریز، ایران است. وی مدرک کارشناسی ارشد خود را در رشته مهندسی کامپیوتر از دانشگاه تبریز دریافت کرده است. علاقه‌مندی‌های پژوهشی وی شامل الگوریتم‌ها و سیستم‌های توزیع‌شده، مدل‌سازی و راستی‌آزمایی رسمی، و تحلیل قابلیت اطمینان می‌باشد. نشانه رایانامه ایشان عبارتند از:

l.namvari@tabrizu.ac.ir

روش ارجاع: ل.ناموری تازه‌کند، بررسی تحمل‌پذیری یک الگوریتم پخش علیّتی در برابر گم شدن پیام، دوفصلنامه محاسبات و سامانه‌های توزیع شده، سال هفتم، شماره ۱، شماره پیاپی ۱۳، صفحه ۲۴ تا ۳۲، سال ۱۴۰۳

How to cite: L. NamvariTazehkand, **Evaluating the Fault Tolerance of a Causal Broadcast Algorithm Against Message Loss**, Journal of Distributed Computing and Systems (JDCS), Vol 7, Issue 1, Pages 24 - 32, 2024.

Evaluating the Fault Tolerance of a Causal Broadcast Algorithm Against Message Loss

L. NamvariTazehkand

Department of Computer Engineering, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran.

Abstract

This study focuses on the design and analysis of a causal broadcast algorithm that specifically addresses two fundamental challenges in distributed systems: reducing communication complexity and ensuring fault tolerance in the presence of unreliable communication. With the rapid growth of distributed systems and their critical role in areas such as distributed data management, mutual exclusion, and concurrent communications, the need for efficient and reliable algorithms to preserve causal relationships among messages has become more pressing than ever. This research introduces a novel algorithm called LCUC, designed by drawing inspiration from existing algorithms and addressing their shortcomings.

With the increasing development of distributed systems and the importance of interactions among their nodes, selecting message exchange algorithms that are compatible with system characteristics has become a necessity. Causal broadcast algorithms, as a key tool in this domain, preserve the causal relationship between messages. In this study, an algorithm called LCUC is introduced, designed with inspiration from the concepts of "reducing communication complexity" and "managing unreliable communications." This algorithm addresses two primary challenges in asynchronous distributed systems: reducing communication complexity and accommodating unreliable communications. This paper reviews previous causal broadcast algorithms, identifies their limitations, and proposes a new algorithm for distributed systems with unreliable communications that performs optimally under normal process conditions. The algorithm resolves existing issues using supplementary messages and demonstrates that its communication complexity is O(n), ensuring proper message delivery even in the event of communication problems. In another part of this research, the presence of faulty processes is considered, and the probability of message loss and its impact on violating causality order is analyzed using the Poisson distribution model. Findings indicate that as the number of faulty processes increases, the probability of violating causality order also increases.