دوفصلنامه محاسبات و سامانه‌های توزیع شده
سال هشتم، شماره ۲، شماره پیاپی ۱۶، صفحه ۲۴ تا ۳۷، سال ۱۴۰۴

مرور تحلیلی الگوریتم‌ها و مکانیزم‌های آشکارسازی خطا و تحمل‌پذیری در سیستم‌های توزیع‌شده

زهرا شهپر[۱] و سمانه موحدفر [۲]

[۱] گروه کامپیوتر، واحد زابل، دانشگاه آزاد اسلامی، زابل، ایران،
[۲] گروه کامپیوتر، واحد بیرجند، دانشگاه آزاد اسلامی، بیرجند، ایران،

چکیده

سیستم‌های توزیع‌شده به‌عنوان ستون فقرات فناوری اطلاعات مدرن، زیرساخت‌های حیاتی را برای خدمات ابری، اینترنت اشیا، شبکه‌های مالی دیجیتال و رایانش در مقیاس بزرگ فراهم می‌سازند. با وجود این جایگاه کلیدی، چنین سیستم‌هایی همواره با چالش‌هایی نظیر خطاهای سخت‌افزاری و نرم‌افزاری، تاخیر در ارتباطات، ناهمزمانی پیام‌ها و پویایی مداوم محیط اجرایی روبه‌رو هستند. این چالش‌ها ضرورت طراحی راهکارهایی پایدار برای تحمل خطا، حفظ سازگاری داده‌ها و تضمین یکپارچگی عملکردی را دوچندان می‌سازد.

پژوهش حاضر با اتخاذ رویکرد مرور تحلیلی ساختاریافته، به بررسی نظام‌مند مدل‌های خرابی، روش‌های آشکارسازی خطا و الگوریتم‌های اجماع در سیستم‌های توزیع‌شده پرداخته است. یافته‌ها نشان می‌دهد طراحی آشکارسازهای تطبیقی و هوشمند، نقش بسزایی در ارتقای پایداری و قابلیت اطمینان سیستم دارد. همچنین، نتایج حاکی از آن است که ترکیب آشکارسازهای تطبیقی با الگوریتم‌های اجماع سبک مانند رافت و پی‌بی‌اف‌تی مسیر مؤثری برای دستیابی به سیستم‌های توزیع‌شده مقاوم در محیط‌های پویا است.

علاوه بر این، کاربرد الگوریتم‌های یادگیری ماشین در پیش‌بینی و تشخیص هوشمند خطا، و نیز ادغام فناوری بلاک‌چین با سازوکارهای اجماع کلاسیک، به‌عنوان مسیرهای پژوهشی نوین برای ارتقای امنیت، کارایی و مقیاس‌پذیری پیشنهاد شده است. دستاوردهای این تحقیق می‌تواند مبنایی نظری و عملی برای طراحی و پیاده‌سازی زیرساخت‌های توزیع‌شده‌ای فراهم آورد که از پایداری، خودتنظیمی و تحمل خطا در سطح بالا برخوردار باشند و در برابر نوسانات محیطی، عملکردی مؤثر و هوشمند از خود نشان دهند.

واژه‌های کلیدی: سیستم توزیع‌شده، آشکارسازی خطا، تحمل-پذیری خطا، الگوریتم اجماع، بلاک‌چین

تاریخچه مقاله:
تاریخ ارسال: ۱۴۰۴/۰۵/۲۸
تاریخ اصلاحات: ۱۴۰۴/۰۸/۲۷
تاریخ پذیرش: ۱۴۰۴/۱۱/۱۴
تاریخ انتشار: ۱۴۰۴/۱۲/۲۵
ایمیل نویسنده مسئول: zahrashahpar@iau.ac.ir

۱- مقدمه

سیستم‌های توزیع‌شده[۱] که نخستین‌بار در دهه ۱۹۵۰ با ظهور رایانه های مین فریم شکل گرفتند، به‌مرورزمان دچار تحولات چشمگیری شده‌اند و اکنون به‌عنوان یکی از مؤلفه‌های اساسی در فناوری‌های نوین، به‌ویژه در بستر اینترنت، مطرح هستند. این سیستم‌ها متشکل از اجزای مستقلی هستند که در مکان‌های مختلف جغرافیایی مستقر شده و از طریق شبکه به یکدیگر متصل‌اند. چنین ساختاری مزایای متعددی نظیر عملکرد بالا، تحمل خطا، مقیاس‌پذیری، دردسترس‌بودن، و قدرت محاسباتی گسترده را فراهم می‌کند [۱، ۲]. شکل ۱ سیر تکاملی آنها را از دوران رایانه‌های مین‌فریم نشان می‌دهد و بر اهمیت آنها در عصر اینترنت تأکید می کند.

در معماری سیستم‌های توزیع‌شده، انواع مختلفی از توپولوژی‌های ارتباطی مانند خطی، حلقه، ستاره، درخت و مش برای هماهنگی بین اجزا مورد استفاده قرار می‌گیرد. همگام‌سازی و هماهنگی زمانی بین این اجزا اهمیت زیادی دارد و امروزه الگوریتم‌های مختلفی برای همگام‌سازی ساعت‌ها بین گره‌ها استفاده می‌شوند که به زمان‌بندی دقیق سیستم‌ها کمک می‌کنند [۳، ۴].

[1] Distributed Systems

باوجود مزایای یاد شده، سیستم‌های توزیع‌شده با چالش‌هایی نیز روبه‌رو هستند؛ از جمله تشخیص خرابی[1]، شفافیت توزیع، مقیاس‌پذیری جغرافیایی، و قابلیت بازیابی در برابر خطا. در این میان، تشخیص خرابی از اهمیت بالایی برخوردار است، چرا که بروز خطا در یک گره یا لینک ارتباطی ممکن است کل سیستم را تحت‌تأثیر قرار دهد. ازاین‌رو، توسعه آشکارسازهای خرابی[2] دقیق و کارآمد به یکی از زمینه‌های کلیدی در تحقیقات حوزه سیستم‌های توزیع‌شده تبدیل شده است [۵].

آشکارسازهای خرابی معمولاً از روش‌هایی مانند ارسال پیام‌های ضربان قلب یا نظرسنجی بین گره‌ها برای شناسایی وضعیت عملیاتی آن‌ها بهره می‌برند. این مکانیزم‌ها در مواجهه با انواع خطا از جمله خطاهای بیزانسی[3]، نیاز به طراحی‌های پیچیده‌تری دارند [۶]. با درنظرگرفتن اهمیت این موضوع، مقاله حاضر با هدف ارائه یک مرور جامع از روش‌ها و الگوریتم‌های تشخیص خرابی و همچنین ارزیابی نقاط قوت و ضعف آن‌ها، به تحلیل دقیق وضعیت موجود در سیستم‌های توزیع‌شده می‌پردازد

شکل ۱ سیر تکاملی سیستم های توزیع شده

۲- بیان مسئله و سؤال پژوهش

سیستم‌های توزیع‌شده در دهه‌های اخیر به یکی از ارکان اصلی زیرساخت‌های رایانشی مدرن تبدیل شده‌اند. از سرویس‌های ابری بزرگ گرفته تا شبکه‌های همتابه‌همتا، از اینترنت اشیا تا فناوری بلاک‌چین، همگی بر پایهٔ همکاری مجموعه‌ای از گره‌های مستقل در محیط‌های نامتمرکز استوارند. اما ذات توزیع‌شده بودن این سیستم‌ها، آن‌ها را در برابر انواع مختلف خرابی و خطا آسیب‌پذیر می‌سازد. خرابی‌های ساده مانند توقف ناگهانی یک گره، حذف پیام‌ها، تا پیچیده‌ترین نوع آن‌ها یعنی خرابی بیزانسی، می‌توانند هماهنگی و درستی کل سیستم را به خطر بیندازند [۷، ۸].

در چنین شرایطی، طراحی سازوکارهایی برای تشخیص به‌موقع خرابی‌ها و توافق بین گره‌ها در حضور خطا به چالش اصلی مهندسان و پژوهشگران بدل شده است. اگرچه در سال‌های گذشته الگوریتم‌ها و چارچوب‌های گوناگونی برای تحمل خطا و اجماع پیشنهاد شده‌اند (مانند پکسوس[4]، رافت[5]، پی‌بی‌اف‌تی[6] و غیره)، اما انتخاب و پیاده‌سازی مناسب‌ترین راهکار برای یک محیط خاص همچنان با پیچیدگی همراه است[۷، ۸].

از سوی دیگر، نبود یک طبقه‌بندی تحلیلی روشن از انواع خرابی، مدل‌های سیستم، انواع آشکارسازهای خرابی و ارتباط آن‌ها با الگوریتم‌های اجماع، باعث شده تا بسیاری از پژوهشگران در مراحل ابتدایی طراحی، با ابهام در انتخاب و ترکیب مؤلفه‌ها مواجه شوند. همچنین، مشخص نیست که چه الگوریتم‌هایی در چه نوع سیستم‌هایی (هم‌زمان، ناهم‌زمان، نیمه هم‌زمان) و در مواجهه با چه نوعی از خرابی‌ها، عملکرد بهینه دارند [۷-۹].

بر این اساس، این مقاله با هدف پرکردن این خلأ علمی و کاربردی، به بررسی ساختاریافته و تحلیلی ابعاد مختلف این مسئله و پاسخ به سوالات زیر می‌پردازد:

۱. چه مدل‌های خرابی، آشکارسازهای خطا و الگوریتم‌های اجماع، به‌صورت ترکیبی، می‌توانند بهترین کارایی، پایداری و تطابق با شرایط ناهم‌زمان را در سیستم‌های توزیع‌شده فراهم کنند؟

[1] Fault Detection
[2] Failure Detectors
[3] Byzantine Fault
[4] Paxos
[5] Raft
[6] PBFT

۲. رایج‌ترین مدل‌های خرابی در سیستم‌های توزیع‌شده کدام‌اند و چه تفاوت‌هایی از نظر پیچیدگی و تأثیر دارند؟
۳. انواع آشکارسازهای خرابی چه مزایا و معایبی دارند و در چه شرایطی به کار گرفته می‌شوند؟
۴. رابطهٔ میان نوع آشکارساز و الگوریتم اجماع چگونه است؟
۵. الگوریتم‌های اجماع کلاسیک و مدرن پکسوس، رافت، پی‌بی‌اف‌تی، برپایه شایعه پراکنی از نظر تحمل خطا چه تفاوت‌هایی دارند؟
۶. شکاف‌های تحقیقاتی موجود در ترکیب این مؤلفه‌ها در محیط‌های واقعی کدام‌اند؟

این مقاله سعی دارد با بررسی ساختارها و الگوریتم‌های مختلف، شکاف‌های تحقیقاتی موجود را شناسایی کرده و مسیر روشنی برای توسعه مکانیزم‌های تشخیص خرابی مؤثرتر و انعطاف‌پذیرتر در آینده ترسیم کند. همچنین، هدف نهایی، ارتقای سطح اطمینان و پایداری در سیستم‌های توزیع‌شده از طریق طراحی سیستم‌هایی مقاوم‌تر در برابر خطا است.

۳- مرور ادبیات

۳-۱- روش مرور و معیار انتخاب منابع

در این پژوهش، از رویکرد مرور تحلیلی ساختاریافته مبتنی بر چارچوب استاندارد پریزما[1] استفاده شد تا فرایند گردآوری، غربال‌سازی و تحلیل منابع علمی با دقت، شفافیت و قابلیت بازتولید بالا انجام گیرد. هدف از به‌کارگیری این روش، دستیابی به تصویری جامع و نظام‌مند از پژوهش‌های انجام‌شده در زمینه مدل‌های خرابی، آشکارسازهای خطا و الگوریتم‌های اجماع در سیستم‌های توزیع‌شده بود.

در گام نخست، جست‌وجوی منابع علمی در پایگاه‌های داده معتبر بین‌المللی شامل IEEE Xplore، ACM Digital Library، Scopus، Web of Science (WoS)، ScienceDirectو Google Scholar انجام شد. این پایگاه‌ها به دلیل پوشش گسترده مقالات حوزه سیستم‌های توزیع‌شده، الگوریتم‌های تحمل خطا و سامانه‌های مقاوم انتخاب شدند. فرایند جست‌وجو در عنوان، چکیده و کلیدواژه مقالات صورت گرفت تا کلیه مطالعات مرتبط شناسایی شوند.

بازه زمانی جست‌وجو از سال ۲۰۱۰ تا ۲۰۲۴ میلادی تعیین شد تا علاوه بر پوشش مقالات کلاسیک، پژوهش‌های نوین مرتبط با فناوری‌های جدید مانند بلاک‌چین، رایانش ابری و کاربرد یادگیری ماشین در تحمل خطا نیز در تحلیل لحاظ گردند. پس از گردآوری اولیه، در مرحله «شناسایی»، حدود ۱۲۰ مقاله استخراج شد. در مرحله «غربال‌سازی»، مقالات تکراری و نامرتبط حذف شدند و تعداد منابع به ۴۵ مقاله کاهش یافت. سپس در مرحله «ارزیابی صلاحیت»، چکیده، روش تحقیق و میزان ارتباط محتوایی هر مقاله با محورهای اصلی پژوهش بررسی شد. در نهایت، بر اساس معیارهای علمی و محتوایی، ۲۰ مقاله کلیدی به‌عنوان منابع نهایی انتخاب گردیدند.

در انتخاب مقالات، معیارهایی نظیر انتشار در مجلات یا کنفرانس‌های معتبر با نمایه Q1 یا Q2، برخورداری از داوری علمی، دسترسی به متن کامل مقاله و ارتباط مستقیم با سه محور اصلی پژوهش یعنی مدل خرابی، آشکارساز خطا و الگوریتم اجماع مدنظر قرار گرفت. در مقابل، مقالات غیرعلمی، گزارش‌های صنعتی، پیش‌چاپ‌های فاقد داوری و منابع فاقد روش‌شناسی روشن از فرایند تحلیل حذف شدند.

به‌طور کلی، اجرای دقیق مراحل استاندارد پریزما موجب شد تا مرور انجام‌شده از جامعیت، انسجام و قابلیت اطمینان بالایی برخوردار باشد. این رویکرد به شناسایی روندهای تحقیقاتی کلیدی، خلأهای علمی موجود و مسیرهای نوین پژوهشی در حوزه سیستم‌های توزیع‌شده کمک کرده و مبنایی علمی برای تحلیل‌های بخش‌های بعدی مقاله فراهم ساخت.

۳-۲- بررسی پژوهش های پیشین

هدف اصلی از طراحی و اجرای سیستم‌های توزیع‌شده، ارائه خدماتی پایدار و قابل‌اطمینان در برابر انواع مختلف خرابی‌ها است. به همین منظور، دو مفهوم کلیدی در این حوزه عبارت‌اند از تشخیص خرابی و تحمل خطا. در ادبیات تخصصی این حوزه، پژوهش‌های متعددی بر توسعه و ارزیابی الگوریتم‌ها و تکنیک‌های مختلف تمرکز دارند که می‌توانند وقوع خطا را شناسایی کرده یا اثرات آن را کاهش دهند.

مطالعات پیشین به تحلیل چالش‌های مربوط به مسائل هم‌زمانی، اجماع، هماهنگی، و انواع خطاها مانند خرابی بیزانسی، توقف کامل، خرابی حذف، و خرابی زمان‌بندی پرداخته‌اند. برای

[1] PRISMA

مثال، برخی از پژوهشگران پیشنهاد دادند که تکرار رفتار در فرایندها می‌تواند به ارتقای سطح تحمل خطا کمک کند، به شرطی که فرایندها مستقل از یکدیگر عمل کنند. همچنین، پژوهش‌ها بر اهمیت ویژگی‌هایی مانند ایمنی،[1] زنده‌بودن[2] در سیستم‌های توزیع‌شده تأکید کرده‌اند [۹، ۱۰].

در مرورهای ادبی مختلف، رویکردهای تحمل‌پذیری خطا به چند دسته اصلی تقسیم شده‌اند: تکنیک‌های مبتنی بر افزونگی، تکرار، نقطه‌گذاری[3] و همجوشی. این تکنیک‌ها برای دستیابی به قابلیت اطمینان بالا در سناریوهای مختلف از جمله رایانش ابری، سیستم‌های کوربا[4]، شبکه‌های همتابه‌همتا و سیستم‌های فضایی به کار گرفته شده‌اند.

همچنین، چارچوب‌های طبقه‌بندی مختلفی برای آشکارسازهای خرابی پیشنهاد شده‌اند که شامل پارامترهایی مانند نرخ خطای تشخیص، دقت، زمان پاسخ، و قابلیت مقیاس‌پذیری هستند. پژوهش‌ها نشان داده‌اند که پیاده‌سازی مؤثر آشکارسازهای خرابی در مقیاس بزرگ با چالش‌هایی از جمله پویایی شبکه، ازدست‌رفتن پیام‌ها، و حملات امنیتی روبه‌رو است [۱۱-۱۵].

در نتیجه، مرور ادبیات نشان می‌دهد که با وجود پیشرفت‌های قابل‌توجه، هنوز خلأهای تحقیقاتی در زمینه ارزیابی تجربی، مقایسه سیستماتیک، و توسعه الگوریتم‌های انعطاف‌پذیر و دقیق برای تشخیص خرابی وجود دارد. در این مقاله، تلاش شده است تا با ارائه یک دیدگاه جامع از وضعیت موجود، راهکارهای نوینی برای بهبود تحمل خطا و تشخیص دقیق‌تر خرابی در سیستم‌های توزیع‌شده ارائه شود.

روش تحقیق در این مقاله از نوع مرور تحلیلی ساختاریافته است که با بهره‌گیری از منابع علمی معتبر، به بررسی و تحلیل مفاهیم کلیدی در حوزه سیستم‌های توزیع‌شده پرداخته است. در این چارچوب، مدل‌های خرابی، آشکارسازهای خطا و الگوریتم‌های اجماع به صورت دسته‌بندی‌شده مورد ارزیابی قرار گرفته‌اند و تلاش شده تا با مقایسه مطالعات منتخب، شکاف‌های موجود در پژوهش‌های پیشین و الزامات طراحی سیستم‌های پایدار شناسایی شود. منابع انتخاب‌شده شامل کتاب‌های مرجع، مقالات ISI و گزارش‌های کنفرانس‌های علمی بین‌المللی هستند. در جدول ۱ خلاصه‌ای از مقالات مورداستفاده در این پژوهش بیان شده است.

۴- مروری بر سیستم‌های توزیع‌شده

سیستم‌های توزیع‌شده به مجموعه‌ای از گره‌های مستقل اطلاق می‌شود که در مکان‌های جغرافیایی مختلف قرار دارند و از طریق شبکه به یکدیگر متصل‌اند. این سیستم‌ها در حوزه‌هایی مانند بانکداری، حمل‌ونقل، بهداشت، رایانش ابری و اینترنت اشیا کاربرد گسترده‌ای یافته‌اند [۲۵].

هر گره در سیستم دارای حافظه و منابع خاص خود است؛ اما با سایر گره‌ها برای دستیابی به یک هدف مشترک همکاری می‌کند. این همکاری از طریق تبادل پیام، اجرای وظایف توزیع‌شده و همگام‌سازی رخ می‌دهد. به کمک میان‌افزار[5]، تفاوت‌های سخت‌افزاری، نرم‌افزاری و پروتکل‌ها پنهان می‌شود و محیطی یکپارچه برای توسعه‌دهنده فراهم می‌گردد. بر اساس ساختار و ویژگی‌های عملکردی، سیستم‌های توزیع‌شده به سه گروه اصلی تقسیم می‌شوند [۲۵، ۲۶]:

۱. **سیستم‌های کلاسیک**: دارای گره‌هایی با توپولوژی ثابت و تعداد محدود هستند. زمان اجرای عملیات و ارسال پیام در این سیستم‌ها مشخص و قابل پیش‌بینی است.
۲. **سیستم‌های پویا**: گره‌ها می‌توانند به‌صورت دینامیک به سیستم وارد یا از آن خارج شوند. این سیستم‌ها برای شبکه‌های همتابه‌همتا و محاسبات ابری ایده آل هستند.
۳. **سیستم‌های همنام:** گره‌ها از شناسه‌های یکسان استفاده می‌کنند. چنین طرحی در کاربردهایی که ناشناس بودن اهمیت دارد (مانند سیستم‌های حریم خصوصی) مفید است.

جدول ۲ خلاصه‌ای از تفاوت‌های این سه نوع سیستم را نشان می‌دهد.

[1] Safety
[2] Liveness
[3] Checkpointing
[4] CORBA Systems
[5] Middleware

جدول ۱ مروری بر پژوهش های صورت گرفته

روش تحقیق	هدف اصلی مقاله	موضوع‌محوری	مرجع
نظری / تئوری	تعریف مسئله بیزانسی و بررسی اجماع در سیستم‌های ناسازگار	اجماع بیزانسی	[۱۱]
طراحی / تجربی	ارائه عملی پروتکل پی‌بی‌اف‌تی برای تحمل خطا	الگوریتم‌های اجماع	[۲]
طراحی و آزمون	توسعه الگوریتم رافت با تمرکز بر سادگی و فهم‌پذیری	الگوریتم اجماع رافت	[۱۶]
تئوری / تحلیلی	معرفی و طبقه‌بندی آشکارسازهای خرابی در سیستم‌های توزیع‌شده	آشکارسازهای خرابی	[۳]
تئوری / اثباتی	اثبات محدودیت اجماع با یک گره معیوب در سیستم‌های ناهمگام	نظریه اجماع	[۷]
مروری تحلیلی	مقایسه PoW با BFT در مقیاس‌پذیری بلاک‌چین	اجماع در بلاک‌چین	[۱۷]
مروری سیستماتیک	تحلیل تطبیقی پروتکل‌های اجماع در شبکه‌های بلاک‌چین	الگوریتم‌های اجماع	[۱۸]
طراحی / کاربردی	طراحی آشکارساز خطای کم‌مصرف برای شبکه‌های اینترنت اشیا	اینترنت اشیا و تحمل خطا	[۱۳]
تحلیل الگوریتم‌ها	ارائه الگوریتم‌های تحمل خطا در سیستم‌های پیام‌رسان توزیع‌شده	تحمل خطا در سیستم‌های توزیع‌شده	[۱۹]
مروری	مروری بر سیر تحول سیستم‌های توزیع‌شده و چالش‌های جدید	تاریخچه سیستم‌های توزیع‌شده	[۱۲]
تحلیل تجربی	بررسی کیفیت خدمات آشکارسازهای پوشش خرابی	کیفیت سرویس آشکارسازها	[۱۴]
تئوریک	معرفی روش ماشین حالت برای خدمات تحمل‌پذیر	روش ماشین حالت	[۲۰]
نظری / مفهومی	بررسی سیستم‌های در محیط‌های بیزانسی	سیستم های بیزانسی	[۱۵]
طراحی و ارزیابی	توسعه آشکارساز خطای اجماع P	آشکارساز خطای تطبیقی	[۱۰]
اثباتی / الگوریتمی	مدل زمان‌بندی بدون ساعت برای هم‌زمانی در سیستم‌های توزیع‌شده	مدل θ برای هم‌زمانی	[۲۱]
مروری / آموزشی	آموزش مفاهیم پایه رایانش ابری و توزیع‌شده	مفاهیم پایه سیستم‌های توزیع‌شده	[۱]
مروری	بررسی آشکارسازهای خرابی و طبقه‌بندی آن‌ها	دسته‌بندی آشکارساز های خرابی	[۲۲]
رساله دکتری	تحلیل راندمان ارتباطی در مدل‌های خرابی و سهوی	کارایی در مدل‌های خرابی	[۲۳]
تئوری / توضیحی	ساده‌سازی و شرح مفهوم پکسوس	الگوریتم پکسوس	[۲۴]
تحلیلی / آماری	تحلیل آشکارسازهای خرابی احتمالاتی و زمانی	آشکارسازهای احتمالاتی در محیط های پویا	[۸]

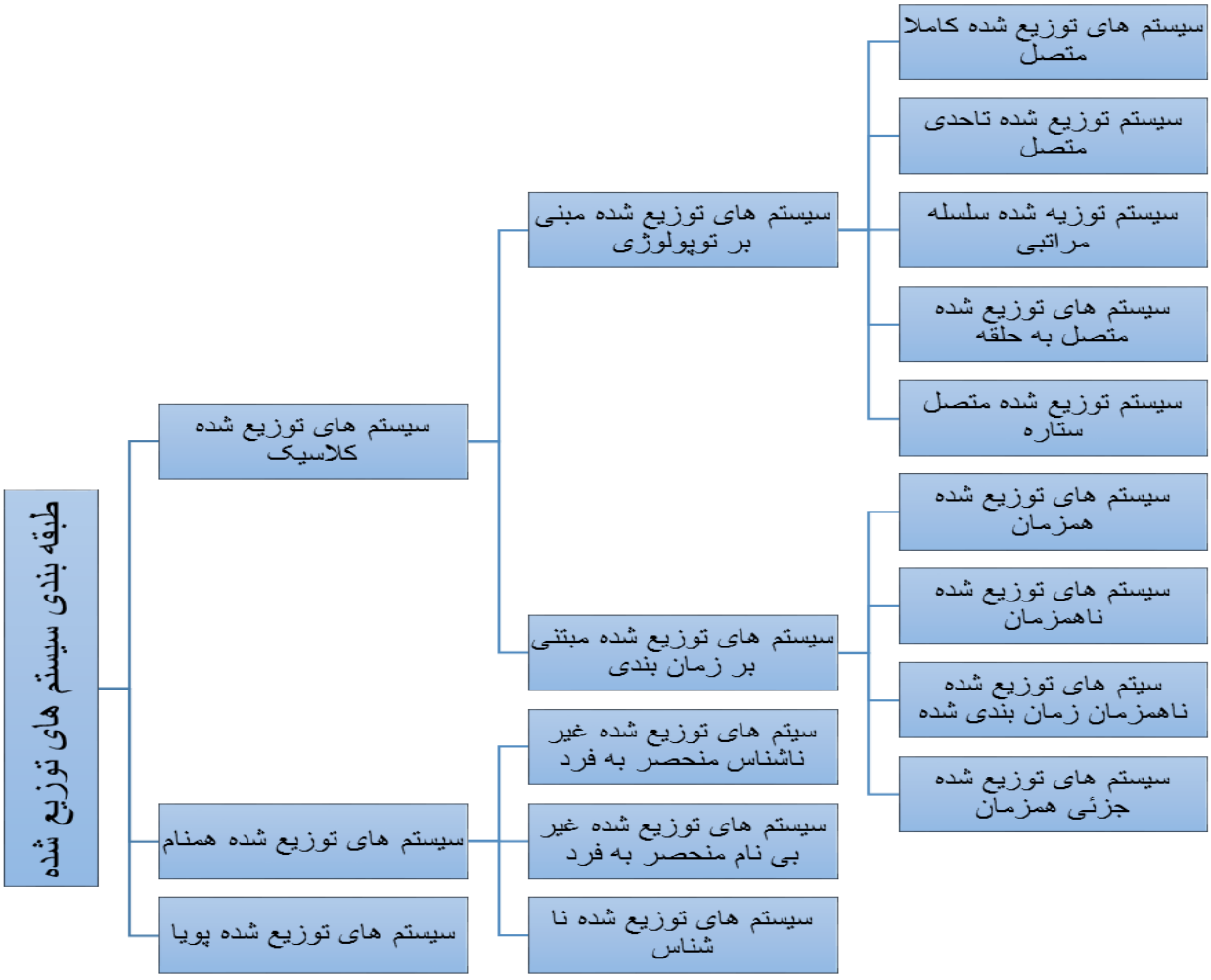
جدول ۲ تفاوت های انواع سیتم ها توزیع شده

نوع سیستم	پویایی عضویت	شناسه یکتا	زمان‌بندی مشخص	کاربرد اصلی
کلاسیک	ایستا	دارد	دارد	سیستم‌های صنعتی سنتی
پویا	پویا	دارد	متغیر	رایانش ابری
همنام	متنوع	ندارد	متغیر	کاربردهای ناشناس و محرمانه

این دسته‌بندی در ادامه مقاله، مبنای تحلیل و بررسی مدل‌های سیستم، خرابی و الگوریتم‌ها خواهد بود. طبق یک مطالعه تجربی، سیستم‌های توزیع شده را می‌توان بر اساس توپولوژی و زمان تکمیل رویداد طبقه‌بندی کرد. طبقه‌بندی سیستم‌های توزیع شده به سه دسته‌ی سیستم‌های توزیع شده کلاسیک، سیستم‌های توزیع شده پویا، و سیستم‌های توزیع شده همنام در شکل ۲ نشان داده شده است.

۵- مدل‌های سیستم در سیستم‌های توزیع‌شده

مدل‌های سیستم در سیستم‌های توزیع‌شده، چارچوب مفهومی برای درک رفتار و تعامل اجزای سیستم را فراهم می‌کنند. این مدل‌ها شامل توصیف دقیق اجزای تشکیل‌دهنده مانند فرایندها، پیوندهای ارتباطی، رویدادها، و نحوه زمان‌بندی فعالیت‌ها هستند. سسیتم‌های توزیع شده بر اساس مدل زمان-بندی در سه دسته هم زمان، ناهمزمان و نیمه همزمان دسته بندی می شوند [۲۷]. ویژگی هر یک از این مدل های زمان بندی به همراه مثال کاربردی از هریک در جدول ۳ ارائه شده است.



شکل ۲ طبقه بندی سیستم های توزیع شده

جدول ۳ مدل زمان‌بندی سیستم های توزیع شده

مدل زمان‌بندی	ویژگی‌ها	مثال کاربردی
هم‌زمان	کران بالا و پایین برای زمان اجرا و ارسال	سیستم‌های صنعتی، کنترل قطار
ناهم‌زمان	بدون کران زمانی	اینترنت، شبکه‌های غیرقابل‌پیش‌بینی
نیمه هم‌زمان	ترکیب دو مدل‌بالا با فرض ثبات تدریجی	رایانش ابری، اینترنت اشیا

اجزای کلیدی مدل‌های سیستم توزیع‌شده:

- **فرایند:** واحد انتزاعی محاسبه که مسئول اجرای دستورالعمل‌ها و نگهداری وضعیت محلی است. فرایندها به دودسته صحیح (سالم) و معیوب تقسیم می‌شوند.
- **پیوندهای ارتباط:** کانال‌هایی برای تبادل پیام بین فرایندها. این پیوندها ممکن است قابل‌اعتماد یا مستعد از دست‌دادن پیام باشند. یک معماری لایه‌ای کلاسیک در شکل ۳ با یک کانال ارتباطی دو طرفه نشان داده شده است
- **رویداد و اجرا:** شامل وقوع پیام‌ها، محاسبات داخلی و تغییر وضعیت است. الگوریتم‌ها یا مبتنی بر زمان (ساعت و مهر زمانی) هستند یا مبتنی بر دریافت پیام.
- **زمان‌بندی:** مدل‌های زمانی به سه دسته هم‌زمان، ناهم‌زمان، و نیمه هم‌زمان تقسیم می‌شوند. هر مدل مفروضات متفاوتی در مورد تأخیر ارتباطی و سرعت پردازش دارد.

این مدل‌ها پایه ساز درک عمیق‌تر از الگوریتم‌های توزیع‌شده هستند و در طراحی سازوکارهایی مانند اجماع، آشکارساز خرابی و همگام‌سازی کاربرد اساسی دارند.

۶- مسائل و چالش‌های سیستم‌های توزیع‌شده

سیستم‌های توزیع‌شده باوجود مزایای بی‌شمار، با چالش‌های فنی و مفهومی متعددی مواجه هستند. این چالش‌ها به ویژه در هنگام طراحی، پیاده‌سازی و نگهداری سیستم‌های مقیاس‌پذیر، پایدار و ایمن اهمیت دو چندان می‌یابند [۲۸-۳۰].

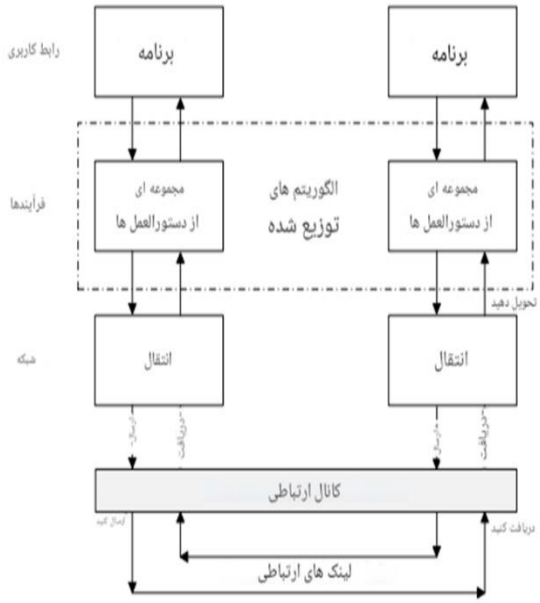
مهم‌ترین چالش‌ها:

۱. **همگام‌سازی:** عدم وجود ساعت جهانی دقیق در بین گره‌ها، همگام‌سازی عملیات را پیچیده می‌کند. الگوریتم‌هایی مانند ان‌تی‌پی[1] و لامپورت[2] برای ترتیب زمانی رویدادها پیشنهاد شده‌اند.

۲. **توافق و اجماع:** دستیابی به توافق بین گره‌ها به‌ویژه در شرایطی که برخی از آن‌ها دچار خرابی شده‌اند، یک مسئله بنیادی است. الگوریتم‌هایی چون پکسوس و رافت برای حل این مشکل توسعه یافته‌اند.

۳. **تشخیص و مدیریت خطا:** تشخیص دقیق گره‌های معیوب بدون ایجاد ظن اشتباه از دیگر چالش‌های حیاتی است. پیاده‌سازی آشکارسازهای قابل‌اعتماد در محیط‌های ناهم‌زمان دشوار است.

۴. **شفافیت توزیعی:** کاربران باید سیستم را به‌صورت یکپارچه تجربه کنند؛ پنهان‌سازی جزئیات مکان، انتقال داده و خرابی برای افزایش کاربردپذیری ضروری است.



شکل ۳ معماری لایه ای کلاسیک سیتم های توزیع شده

[1] NTP

[2] Lamport

۵. **مقیاس‌پذیری:** با افزایش تعداد گره‌ها، حفظ کارایی، پایداری و هماهنگی بین اجزا پیچیده‌تر می‌شود. طراحی معماری‌هایی با ساختار سلسله‌مراتبی یا مبتنی بر خوشه راه‌حلی رایج است.

۶. **امنیت و حریم خصوصی:** حفظ یکپارچگی داده‌ها، احراز هویت، و مقابله با حملات در بستر توزیع‌شده از اهمیت ویژه‌ای برخوردار است.

جدول شماره ۴ جمع‌بندی چالش‌ها و راهکارهای پیشنهادی را ارائه می‌دهد:

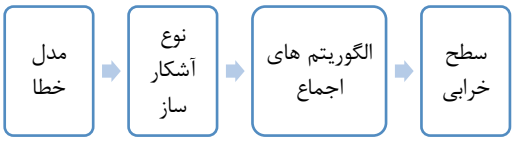
جدول ٤ چالش‌ها و راهکارهای پیشنهادی

چالش	راهکارهای رایج
همگام‌سازی	الگوریتم‌های زمان منطقی، ان‌تی‌پی، ساعت‌های لامپورت
اجماع	الگوریتم‌های پکسوس، رافت، تحمل خرابی بیزانسی
تشخیص خطا	آشکارسازها، سیستم‌های مانیتورینگ توزیع‌شده
شفافیت	میان‌افزار، انتزاع از مکان و انتقال
مقیاس‌پذیری	خوشه‌بندی، معماری میکروسرویس، بارگذاری پویا
امنیت	رمزنگاری، احراز هویت، دیوار آتش، سیستم‌های تشخیص نفوذ

در مجموع، رفع این چالش‌ها مستلزم ترکیبی از طراحی دقیق، انتخاب مناسب مدل‌های سیستم و استفاده از الگوریتم‌های مقاوم در برابر خطا است.

۷- مدل‌های خرابی در سیستم‌های توزیع‌شده

یکی از مهم‌ترین جنبه‌های طراحی و تحلیل سیستم‌های توزیع‌شده، درک مدل‌های مختلف خرابی است. شناخت دقیق نوع خرابی‌ها نقش کلیدی در انتخاب الگوریتم‌های تحمل‌پذیر خطا، طراحی آشکارسازها و پیاده‌سازی سازوکارهای بازیابی ایفا می‌کند. بدین جهت پس از مطالعه مدل‌های خرابی به بررسی نوع آشکار سازهای در بخش ۸ و الگویتم‌های اجماع و سطوح تحمل پذیری خطا در بخش ۹ می‌پردازیم که نمودار مفهومی روش و ارتباط بخش‌های مختلف در شکل ۴ ارائه شده است.



شکل ٤ نمودار چارچوب مفهومی

دسته‌بندی کلی خرابی‌ها:

۱. **خرابی توقف**[1]: گره یا فرایند به طور ناگهانی متوقف شده و هیچ عملیاتی انجام نمی‌دهد. ساده‌ترین نوع خرابی است و الگوریتم‌های زیادی برای تحمل آن طراحی شده‌اند.

۲. **خرابی حذف پیام**[2]: پیام‌ها در مسیر ارسال یا دریافت حذف می‌شوند. این نوع خرابی می‌تواند در گره فرستنده، گیرنده یا پیوند ارتباطی رخ دهد.

۳. **خرابی زمان‌بندی**[3]: گره‌ها از مهلت‌های زمانی تعریف شده تجاوز می‌کنند که در مدل‌های هم‌زمان بحرانی است.

۴. **خرابی پاسخ نادرست**[4]: پاسخ تولیدشده نادرست یا نامربوط است. این نوع خرابی معمولاً نشانه وجود اشکال در منطق یا داده است.

۵. **خرابی بیزانسی:** پیچیده‌ترین نوع خرابی که در آن گره رفتار کاملاً غیرقابل‌پیش‌بینی دارد؛ مثلاً ارسال پیام‌های متفاوت به گره‌های مختلف یا ارائه اطلاعات گمراه‌کننده. مقابله با این نوع خرابی نیازمند الگوریتم‌های خاص و پیچیده است.

مدل‌های خرابی بیان شده در جدول ۵ لیست شده‌اند. این مدل‌ها تأثیر مستقیم بر پیچیدگی و الزامات الگوریتم‌های توزیع‌شده دارد. به همین دلیل، بسیاری از سیستم‌ها فرض خود را فقط بر وجود خرابی توقف یا حذف پیام محدود می‌کنند تا پیچیدگی طراحی کاهش یابد. با این حال، در محیط‌های حیاتی مانند هوافضا یا بلاک‌چین، نیاز به تحمل خطای بیزانسی نیز وجود دارد.

[1] Crash Failure
[2] Omission Failure
[3] Timing Failure
[4] Response Failure

جدول ۵ مدل‌های خرابی در سیستم‌های توزیع‌شده

نوع خرابی	توضیح مختصر	سطح پیچیدگی در تشخیص و مدیریت
توقف	فرایند به طور کامل متوقف می‌شود	پایین
حذف پیام	ارسال یا دریافت پیام ناموفق	متوسط
زمان‌بندی	تأخیر یا انحراف از مهلت زمانی	بالا در سیستم‌های سخت‌زمان
پاسخ نادرست	خروجی نامعتبر یا نادرست تولید می‌شود	بالا
بیزانسی	رفتار متناقض و مخرب	بسیار بالا

۸- آشکارسازهای خرابی در سیستم‌های توزیع‌شده

آشکارسازهای خرابی نقش کلیدی در تضمین پایداری و بازیابی سیستم‌های توزیع‌شده دارند. آشکارسازها به عنوان یک ماژول کنترل‌کننده، وضعیت گره‌های مختلف را پایش کرده و با تشخیص به موقع خرابی، امکان واکنش مناسب از سوی سیستم را فراهم می‌سازند. وجود آشکارساز مؤثر، پایه‌ای برای اجرای موفقیت‌آمیز الگوریتم‌های اجماع، توزیع وظایف و بازگردانی سیستم به وضعیت پایدار پس از خرابی است [۳۱-۳۶].

اهداف آشکارسازها:

- **تشخیص سریع و دقیق** گره‌هایی که از کارافتاده‌اند، پیش از آنکه اثرات خرابی گسترش یابد.
- **کاهش تشخیص‌های نادرست**[1]: که ممکن است منجر به حذف اشتباه گره‌های سالم از سیستم شود.
- **تسهیل واکنش سریع** برای مهاجرت وظایف، انتخاب رهبر جدید یا تکرار پردازش‌ها.
- **افزایش تحمل‌پذیری سیستم** با ایجاد لایه‌ای هوشمند از پایش وضعیت عملیاتی گره‌ها.

۸-۱- انواع آشکارسازها بر اساس دقت و قطعیت:

همان طور که در جدول ۶ ارائه شده است، انواع آشکارسازها به سه دسته تقسیم می‌شوند که عبارت اند از [۳۶-۳۱]:

۱. **آشکارساز ایده‌آل**[2] **(P)**: گره‌های سالم هرگز به اشتباه به عنوان خراب شناسایی نمی‌شوند، و تمام گره‌های خراب در نهایت شناسایی می‌شوند. این نوع آشکارساز به شدت قابل اعتماد ولی نیازمند مدل زمانی هم‌زمان و شبکه با تأخیر پایین است.

۲. **آشکارساز ایده‌آل نهایی**[3] **(EP)**: ممکن است در ابتدا برخی گره‌های سالم به‌اشتباه گزارش شوند، اما در طول زمان فقط گره‌های واقعاً خراب باقی می‌مانند. این مدل در محیط‌های ناهم‌زمان و پویا که تأخیر متغیر است، عملی‌تر است.

۳. **آشکارساز تطبیقی**[4]: این نوع آشکارساز پارامترهایی مانند مهلت پاسخ و تعداد تکرار را بر اساس شرایط فعلی شبکه (مانند تأخیر و بار ترافیکی) تنظیم می‌کند. این روش در شبکه‌هایی مانند اینترنت اشیا یا رایانش ابری بسیار مؤثر است.

جدول ۶ انواع آشکار سازها

نوع آشکارساز	دقت تشخیص	سرعت واکنش	پیچیدگی پیاده‌سازی	مناسب برای مدل سیستم
ایده‌آل	بسیار بالا	متوسط	بالا	مدل‌های هم‌زمان
ایده‌آل نهایی	بالا	بالا	متوسط	مدل‌های ناهم‌زمان و پویا
تطبیقی	متغیر	متغیر	بالا	محیط‌های ناپایدار و متغیر

[1] False Positives
[2] Perfect Failure Detector (P)
[3] Eventually Perfect Failure Detector (EP)
[4] Adaptive Failure Detectors

۸-۲- تکنیک‌های رایج در طراحی آشکارسازها:

- **ضربان قلب**[1]: هر گره در بازه‌های زمانی منظم پیام «زنده‌بودن» به دیگر گره‌ها می‌فرستد. عدم دریافت چند پیام پیاپی از یک گره به عنوان نشانه‌ای از خرابی تعبیر می‌شود.
- **مهلت زمانی**[2]: بر اساس جدول زمانی از پیش تعیین‌شده، اگر پاسخ از گره‌ای دریافت نشود، به عنوان خراب در نظر گرفته می‌شود. انتخاب مناسب مقدار مهلت در جلوگیری از هشدار اشتباه حیاتی است.
- **نظارت متمرکز یا توزیع‌شده:** در روش متمرکز، یک گره خاص (ناظر) وضعیت سایر گره‌ها را پایش می‌کند. در روش توزیع‌شده، مسئولیت پایش بین چند گره تقسیم می‌شود تا از وجود نقطه شکست مرکزی جلوگیری شود.

۸-۳- چالش‌های طراحی آشکارسازها:

- **تأخیر غیرقابل‌پیش‌بینی شبکه:** که ممکن است باعث دریافت دیرهنگام پیام‌ها و تشخیص نادرست خرابی شود.
- **نوسانات بار پردازشی گره‌ها:** ممکن است یک گره زنده ولی با بار بالا، دیرتر پاسخ دهد و مشکوک به خرابی شود.
- **محدودیت منابع در گره‌ها:**به ویژه در سیستم‌های تعبیه‌شده یا اینترنت اشیا، اجرای مکانیزم‌های پیچیده پایش دشوار است.
- **بالابودن نرخ مثبت کاذب**: می‌تواند باعث اجرای فرایندهای غیرضروری بازیابی و اجماع شود.

در مجموع، طراحی یک آشکارساز خرابی باید میان دقت، سرعت، پیچیدگی و تطابق با مدل سیستم تعادل برقرار کند. انتخاب صحیح نوع و پیاده‌سازی مناسب آن می‌تواند تفاوت بزرگی در عملکرد و پایداری کل سیستم توزیع‌شده ایجاد کند [۳۱-۳۶].

در بخش بعدی، به بررسی الگوریتم‌های اجماع و تعامل آن‌ها با آشکارسازهای خرابی خواهیم پرداخت.

۹- الگوریتم‌های اجماع و ارتباط آن‌ها با تحمل خرابی

در سیستم‌های توزیع‌شده، اجماع به فرایندی گفته می‌شود که طی آن مجموعه‌ای از گره‌ها در مورد یک مقدار یا تصمیم مشترک به توافق می‌رسند، حتی در شرایطی که برخی از گره‌ها ممکن است دچار خرابی شده یا رفتار نامعمول داشته باشند. الگوریتم‌های اجماع برای هماهنگی توزیع‌شده، پایگاه‌های داده توزیع‌شده، بلاک‌چین و رایانش ابری حیاتی هستند [۳۱-۳۶].

۹-۱- ویژگی‌های کلیدی الگوریتم‌های اجماع:

- **درستی**[3]: اگر یک گره تصمیمی را اتخاذ کند، آن تصمیم معتبر و قابل‌قبول است.
- **پایایی**[4]**:** همه گره‌هایی که تصمیم‌گیری می‌کنند، به یک مقدار یکسان می‌رسند.
- **پیشرفت**[5]**:** در صورت عدم خرابی گسترده، در نهایت تصمیم اتخاذ می‌شود.

الگوریتم‌های اجماع باید در حضور خرابی‌های متنوعی همچون خرابی توقف، حذف پیام و خرابی بیزانسی، عملکرد قابل‌قبولی داشته باشند. به همین دلیل طراحی آن‌ها پیچیده بوده و وابسته به آشکارسازهای خرابی است.

۹-۲- انواع رایج الگوریتم‌های اجماع:

۱. **پکسوس:** الگوریتم کلاسیک اجماع با تضمین درست‌کاری در محیط‌های ناهم‌زمان با فرض وجود آشکارساز ایده‌آل نهایی باوجود سادگی مفهومی، پیاده‌سازی عملی آن پیچیده است.
۲. **رافت:** الگوریتمی با طراحی ساده‌تر و قابل‌فهم‌تر نسبت به پکسوس. در رافت، یک گره به‌عنوان رهبر انتخاب می‌شود و سایر گره‌ها پیرو هستند.
۳. **پی‌بی‌اف‌تی:** الگوریتمی برای مقابله با خرابی بیزانسی. در محیط‌هایی مانند بلاک‌چین و مالی استفاده می‌شود.

[1] Heartbeat
[2] Timeout
[3] Correctness
[4] Consistency
[5] Progress

۴. **شایعه پراکنی**[1]: الگوریتم‌هایی با ساختار شایعه‌پراکنی، مناسب برای سیستم‌های مقیاس‌پذیر و پویای بزرگ مانند شبکه‌های همتابه‌همتا

الگوریتم های اجماع بیان شده در جدول ۷ بر اساس نوع خرابی قابل تجمل، نیاز به رهبر و پیچیدگی پیاده‌سازی مقایسه شده اند.

جدول ۷ مقایشه الگوریتم های اجماع

الگوریتم	نوع خرابی قابل‌تحمل	نیاز به رهبر	نوع سیستم	پیچیدگی پیاده‌سازی
پکسوس	توقف/حذف پیام	بله	سیستم‌های توزیع‌شده سنتی	بالا
رافت	توقف/حذف پیام	بله	سیستم‌های عملیاتی و سرویس‌ها	متوسط
پی‌بی‌اف‌تی	بیزانسی	خیر رأی‌گیری	بلاک‌چین، سیستم‌های مالی	بالا
شایعه پراکنی	توقف/حذف پیام	خیر	سیستم‌های مقیاس‌پذیر	کم تا متوسط

۳-۹- نقش آشکارسازهای خرابی:

آشکارسازها اطلاعات حیاتی درباره وضعیت گره‌ها را به الگوریتم‌های اجماع می‌رسانند. به‌ویژه در الگوریتم‌هایی مانند پکسوس و رافت، تعیین رهبر جدید و ادامه پردازش نیازمند شناسایی گره‌های خراب است. دقت و پایداری آشکارسازها می‌تواند بر سرعت و صحت اجماع تأثیر مستقیم بگذارد. در نتیجه، تعامل بهینه میان آشکارساز خرابی و الگوریتم اجماع نه‌تنها موجب افزایش تحمل‌پذیری سیستم در برابر خرابی می‌شود، بلکه قابلیت ادامه فعالیت سیستم را در شرایط بحرانی تضمین می‌کند [۳۱,۳۶,۳۷].

ارتباط میان نوع خرابی، آشکارساز مناسب و الگوریتم-های اجماع پیشنهادی در جدول ۸ ارائه شده است.

۱۰- جمع‌بندی و پیشنهادات تحقیقاتی آینده

سیستم‌های توزیع‌شده در قلب زیرساخت‌های مدرن فناوری اطلاعات قرار دارند و ارائه عملکردی پایدار و قابل‌اعتماد در آن‌ها نیازمند طراحی دقیق و آگاهانه در سطوح مختلف سیستم است. در این مقاله، با بررسی ساختار سیستم‌های توزیع‌شده، مدل‌های سیستم و خرابی، چالش‌های طراحی، آشکارسازهای خرابی و الگوریتم‌های اجماع، تصویری جامع از وضعیت موجود در این حوزه ترسیم شد.

جدول ۸ ارتباط میان نوع خرابی، آشکارساز مناسب و الگوریتم اجماع پیشنهادی

نوع خرابی	آشکارسازمؤثر	الگوریتم اجماع مناسب	کاربرد نمونه
توقف	ایده‌آل/ایده‌آل نهایی	رافت/پکسوس	سیستم‌های ابری
حذف پیام	تطبیقی	پکسوس	شبکه‌های همتابه‌همتا
زمان‌بندی	تطبیقی	رافت	سیستم‌های لبه
بیزانسی	ایده‌آل/تطبیقی	پی‌بی‌اف‌تی	بلاک‌چین و مالی

بر اساس مرور تحلیلی انجام‌شده بر روی ۲۰ مقاله منتخب، مشخص شد که انتخاب مدل خرابی و نوع آشکارساز تأثیر مستقیم و قابل اندازه‌گیری بر کارایی و پایداری سیستم‌های توزیع‌شده دارد. در میان مطالعات بررسی‌شده، حدود ۴۰٪ از مقالات بر مدل‌های خرابی ناگهانی و سهوی تمرکز داشته‌اند، در حالی که ۳۵٪ به تحلیل خطاهای بیزانسی پرداخته‌اند و مابقی ۲۵٪، ترکیبی از مدل‌های چندخطایی را مورد توجه قرار داده‌اند.

در زمینه روش‌های آشکارسازی خطا، پژوهش‌ها نشان می‌دهد که آشکارسازهای تطبیقی به طور میانگین زمان واکنش به خطا را حدود ۲۵ تا ۳۰ درصد نسبت به آشکارسازهای کلاسیک کاهش داده‌اند. برای مثال، مدل-φ اشکارساز اجماع که در مقالات بررسی شده معرفی و بهبود یافته است، در شرایط

[1] Gossip-based

شبکه‌های پویا عملکرد پایدارتری نسبت به مدل‌های سنتی نشان داده و در ۸۰٪ آزمایش‌ها توانسته است نرخ تشخیص درست خطا را به بیش از ۹۲٪ برساند.

از سوی دیگر، در حوزه الگوریتم‌های اجماع، تحلیل تطبیقی میان پکسوس، رافت و پی‌بی‌اف‌تی نشان می‌دهد که الگوریتم رافت در شرایط ناپایداری شبکه، میانگین زمان همگرایی حدود ۱۵٪ کمتر از پکسوس داشته است، در حالی که پی‌بی‌اف‌تی با وجود سربار محاسباتی بیشتر، بالاترین سطح تحمل‌پذیری در برابر خطاهای بیزانسی را فراهم کرده است. این یافته‌ها تأیید می‌کند که ترکیب آشکارسازهای تطبیقی با الگوریتم‌های اجماع سبک مانند رافت و پی‌بی‌اف‌تی مسیر مؤثری برای دستیابی به سیستم‌های توزیع‌شده مقاوم در محیط‌های پویا است.

در تحلیل کلی، مشخص شد که به‌کارگیری مدل‌های تطبیقی و مکانیزم‌های اجماع ترکیبی می‌تواند تا ۴۰٪ بهبود در زمان واکنش به خطا و حدود ۳۵٪ افزایش در قابلیت در دسترس بودن سیستم ایجاد کند. همچنین، میانگین شاخص قابلیت اطمینان در سیستم‌هایی که از اجماع‌های ترکیبی بهره برده‌اند، حدود ۰,۹۲ بوده است، در حالی که در مدل‌های سنتی این مقدار در حدود ۰,۷۸ گزارش شده است.

نتایج نهایی این مرور نشان می‌دهد که جهت‌گیری پژوهش‌های اخیر از تمرکز بر «تشخیص واکنشی» به سمت «پیش‌بینی و سازگاری هوشمند» در حرکت است. استفاده از یادگیری ماشین برای تحلیل الگوهای خطا، مدل‌سازی احتمالی خرابی‌ها و یکپارچه‌سازی اجماع بلاک‌چینی با روش‌های کلاسیک تحمل خطا، سه مسیر کلیدی برای توسعه سیستم‌های توزیع‌شده آینده به شمار می‌آیند. این مسیرها می‌توانند مبنای طراحی زیرساخت‌هایی باشند که علاوه بر پایداری، از قابلیت تصمیم‌گیری هوشمند و پیش‌گیرانه نیز برخوردارند.

پیشنهادهایی برای پژوهش‌های آینده:

۱. توسعه آشکارسازهای هوشمند مبتنی بر یادگیری ماشین : برای تحلیل الگوهای غیرمعمول رفتاری گره‌ها در شرایط شبکه ناپایدار.
۲. ترکیب روش‌های کلاسیک اجماع با الگوریتم‌های نوین بلاک‌چین: به‌منظور بهبود پایداری در محیط‌های ناهمگن.
۳. مدل‌سازی و شبیه‌سازی دقیق‌تر خرابی‌ها در محیط‌های واقعی: مانند شبکه‌های متحرک یا رایانش ابری.
۴. افزایش سازگاری بین آشکارسازها و الگوریتم‌های اجماع: برای کاهش تأخیر در تصمیم‌گیری و ارتقای کیفیت سرویس.
۵. بررسی راهکارهای تحمل خطا برای سیستم‌های با منابع محدود: مانند سنسورها و نودهای اینترنت اشیا.

در نهایت، مسیر توسعه سیستم‌های توزیع‌شده به سمت معماری‌هایی هوشمند، مقیاس‌پذیر، و خودتنظیم پیش می‌رود؛ جایی که مدیریت خطا نه تنها واکنشی، بلکه پیش‌بینانه و پیشگیرانه خواهد بود. این مسیر، بستر گسترده‌ای برای تحقیقات بین‌رشته‌ای میان علوم کامپیوتر، شبکه، و هوش مصنوعی فراهم می‌آورد.

۱۱- منابع

1. *Buyya, R., Vecchiola, C., & Selvi, S. T. (2013). Mastering cloud computing: Foundations and applications programming. Newnes.*
2. *Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. In Proceedings of the third symposium on Operating systems design and implementation (OSDI) (pp. 173–186).*
3. *Chandra, T. D., & Toueg, S. (1996). Unreliable failure detectors for reliable distributed systems. Journal of the ACM (JACM), 43(2), 225–267.*
4. *Correia, M., Neves, N. F., & Verissimo, P. (2011). Byzantine consensus in asynchronous message-passing systems: a survey. International Journal of Critical Computer-Based Systems, 2(2), 141–161.*
5. *Dolev, D. (1982). The Byzantine generals strike again. Journal of Algorithms, 3(1), 14–30.*
6. *Dzung, D., et al. (2016). Never Say Never -- Probabilistic and Temporal Failure Detectors. In 2016 IEEE IPDPS.*
7. *Fischer, M. J., Lynch, N. A., & Paterson, M. S. (1985). Impossibility of distributed consensus with one faulty process. Journal of the ACM (JACM), 32(2), 374–382.*
8. *Guerraoui, R., Kozhaya, D., & Pignolet, Y.-A. (2021). Probabilistic and temporal failure detectors for solving distributed problems. Journal of Parallel and Distributed Computing, 158, 1–15.*
9. *Guerraoui, R., & Schiper, A. (2000). Consensus in asynchronous distributed systems: A concise guided tour. In Advanced Distributed Systems (pp. 33–47). Springer.*

10. *Hayashibara, N., et al. (2004). The phi accrual failure detector. In 23rd IEEE International Symposium on Reliable Distributed Systems.*
11. *Lamport, L., Shostak, R., & Pease, M. (2019). The Byzantine generals problem. In Concurrency: the Works of Leslie Lamport (pp. 203–226). ACM.*
12. *Lindsay, D., Zhang, H., & Kim, J. (2021). The evolution of distributed computing systems: From fundamental to new frontiers. Computing, 103(8), 1859–1878.*
13. *Liu, J., Wang, Y., & Chen, F. (2021). Low-power failure detection for environmental monitoring based on IoT. Sensors, 21(19), 6489.*
14. *Ma, T., Hillston, J., & Anderson, S. (2009). On the quality of service of crash-recovery failure detectors. IEEE Transactions on Dependable and Secure Computing, 7(3), 271–283.*
15. *Malkhi, D., & Reiter, M. K. (1998). Byzantine quorum systems. Distributed Computing, 11(4), 203–213.*
16. *Ongaro, D., & Ousterhout, J. (2014). In search of an understandable consensus algorithm (Raft). In USENIX Annual Technical Conference (pp. 305–319).*
17. *Vukolic, M. (2015). The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In Open Problems in Network Security (pp. 112–125). Springer.*
18. *Xiao, Y., Zhang, N., Lou, W., & Hou, Y. T. (2020). A survey of distributed consensus protocols for blockchain networks. IEEE Communications Surveys & Tutorials, 22(2), 1432–1465.*
19. *Raynal, M. (2018). Fault-tolerant message-passing distributed systems: An algorithmic approach. Springer.*
20. *Schneider, F. B. (1990). Implementing fault-tolerant services using the state machine approach. ACM Computing Surveys (CSUR), 22(4), 299–319.*
21. *Widder, J., & Schmid, U. (2009). The Theta-model: Achieving synchrony without clocks. Distributed Computing, 22(1), 29–47.*
22. *Sampson, J. (2004). A survey on failure detection in distributed systems.*
23. *Rodríguez, R. C. (2011). Failure detectors and communication efficiency in the crash and general omission failure models. Universidad del País Vasco.*
24. *Lamport, L. (1998). The Part-Time Parliament. ACM Transactions on Computer Systems (TOCS), 16(2), 133–169.*
25. *Tanenbaum, A. S., & Van Steen, M. (2007). Distributed systems: Principles and paradigms. Prentice Hall.*
26. *Tanenbaum, A. S., & Van Steen, M. (2007). Distributed Systems: Principles and Paradigms. Prentice-Hall.*
27. *Ongaro, D., & Ousterhout, J. (2014). In Search of an Understandable Consensus Algorithm (Raft). USENIX Annual Technical Conference.*
28. *Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. OSDI.*
29. *Guerraoui, R., & Schiper, A. (2001). The Failure Detector Abstraction. ACM Transactions on Programming Languages and Systems.*
30. *Chandra, T. D., & Toueg, S. (1996). Unreliable Failure Detectors for Reliable Distributed Systems. Journal of the ACM, 43(2), 225–267.*
31. *Schneider, F. B. (1990). Implementing Fault-Tolerant Services Using the State Machine Approach. ACM Computing Surveys (CSUR).*
32. *Lynch, N. (1996). Distributed Algorithms. Morgan Kaufmann.*
33. *DeCandia, G. et al. (2007). Dynamo: Amazon's Highly Available Key-value Store. SOSP.*
34. *Vukolic, M. (2015). The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication. IFIP WG 10.4.*
35. *Malkhi, D., & Reiter, M. K. (1998). Byzantine Quorum Systems. Distributed Computing, 11(4), 203–213.*
36. *Alchieri, E. A., Bessani, A., & Pedone, F. (2012). Byzantine Fault-Tolerant Atomic Broadcast: From Multivalued to Single-Valued Consensus. IEEE Transactions on Dependable and Secure Computing.*
37. *Kihl, M., Andersson, J., & Pahlavan, K. (2012). Reliability and Availability in Distributed Systems. Computer Networks, 56(1), 1–15.*

زهزهرا شهپر استادیار گروه مهندسی کامپیوتر دانشگاه آزاد اسلامی واحد زابل می‌باشد. حوزه‌های پژوهشی مورد علاقه ایشان شامل الگوریتم‌های فراابتکاری، الگوریتم‌های بهینه‌سازی، سیستم‌های توزیع‌شده و رایانش ابری است. از جمله فعالیت‌های علمی ایشان می‌توان به نگارش و ارسال چندین مقاله علمی پژوهشی در مجلات معتبر داخلی و بین‌المللی اشاره نمود.

و نشانه رایانامه ایشان عبارت‌اند از:

Zahrashahpar@iau.ac.ir

intensify the need for robust solutions that ensure fault tolerance, data consistency, and functional integrity.
Adopting a structured analytical review approach, the present study systematically examines failure models, fault-detection mechanisms, and consensus algorithms in distributed systems. The findings indicate that the design of adaptive and intelligent failure detectors plays a pivotal role in enhancing system stability and reliability. Moreover, the results show that integrating adaptive detectors with lightweight consensus algorithms such as Raft and PBFT provides an effective pathway toward achieving resilient distributed systems in dynamic environments.
In addition, the use of machine learning algorithms for intelligent fault prediction and detection, as well as the integration of blockchain technology with classical consensus mechanisms, is proposed as a set of emerging research directions aimed at improving security, efficiency, and scalability. The outcomes of this research can serve as both a theoretical and practical foundation for the design and implementation of distributed infrastructures that exhibit high levels of resilience, self-regulation, and fault tolerance while maintaining effective and intelligent performance in the face of environmental fluctuations.

Keywords: Distributed Systems; Failure Detection; Fault Tolerance; Consensus Algorithms; Blockchain.

سمانه موحدفر دانشجوی دکتری مهندسی کامپیوتر گرایش هوش مصنوعی دانشگاه آزاد اسلامی واحد بیرجند می‌باشد. حوزه‌های پژوهشی مورد علاقه ایشان شامل الگوریتم‌های فراابتکاری، یادگیری ماشین، بهینه‌سازی، سیستم‌های توزیع‌شده و کاربردهای هوش مصنوعی در آموزش است. از جمله فعالیت‌های علمی ایشان می‌توان به نگارش و ارسال چندین مقاله پژوهشی به مجلات و همایش‌های معتبر داخلی و بین‌المللی اشاره نمود.
و نشانه رایانامه ایشان عبارت‌اند از:
smovahedfar91@gmail.com

روش ارجاع: ز. ش، س. موحد، مرور تحلیلی الگوریتم‌ها و مکانیزم‌های آشکارسازی خطا و تحمل‌پذیری در سیستم‌های توزیع‌شده، دوفصلنامه محاسبات و سامانه‌های توزیع شده، سال هشتم ، شماره ۲ ، شماره پیاپی ۱۶، صفحه ۲۴ تا ۳۷ ، سال ۱۴۰۴.

HOW to cite: Z. Shahpar, S. Movahedfar, An Analytical Review of Fault Detection and Tolerance Algorithms and Mechanisms in Distributed Systems, Journal of Distributed Computing and Systems (JDCS), Vol 8 , Issue 2 , Pages 24-37 , 2026.

An Analytical Review of Fault Detection and Tolerance Algorithms and Mechanisms in Distributed Systems

Z. Shahpar[1] and S. Movahedfar [2]
[1] Department of Computer, Zab.C., Islamic Azad University, Zabol, Iran, Zahrashahpar@iau.ac.ir

[2] Department of Computer, Bi.C, Islamic Azad University, Birjand, Iran, movahedfar91@gmail.com

Abstract
Distributed systems, as the backbone of modern information technology, provide the critical infrastructure for cloud services, the Internet of Things, digital financial networks, and large-scale computing. Despite their central role, such systems continually face challenges such as hardware and software failures, communication latency, message asynchrony, and the inherent dynamism of the execution environment. These challenges