www.NoavaranGermi.ir سایت نوآوران گرمی - مرجع فیلم های آموزشی و همیار دانشجو

همایش ملی مهندسی رایانه و مدیریت فناوری اطلاعات
National Conference On Computer Engineering & Information Technology Management

ارائه روشی جدید جهت بهبود تحمل پذیری خطا در سیستم های چند عامله

سیدحیدرعبودی[1] جاسم حزباوی[2]

[1]دانشجوی مهندسی تکنولوژی نرم افزار کامپیوتر، موسسه آموزش عالی جهاددانشگاهی خوزستان، pccode.ir@gmail.com

[2] دانشجوی مهندسی تکنولوژی نرم افزار کامپیوتر، موسسه آموزش عالی جهاددانشگاهی خوزستان، jsm.pioneer@gmail.com

چکیده: در این مقاله، روشهایی برای ایجاد تحمل پذیری خطا در سیستم های چند عامله ارائه شده است. ایجاد تحمل پذیری خطا در سیستم های چند عامله، امری دست یافتنی و در عین حال دشوار است. یک سیستم چند عامله به عنوان نمونهٔ مهمی از سیستم های چند عامله با بهره گیری از تکنیک های رایج تحمل پذیری خطا می تواند قابلیت کنار آمدن با انواع متفاوتی از خطاها را پیدا کند. اما آنچه در این مقاله مورد تاکید است استفاده از ذات چندعاملهٔ سیستم ها و بهره گیری از تکنیک هایی است که برخاسته از ویژگیهای خاص سیستم های چندعامله است.

کلید واژه- تحمل پذیری، چند عامله، کمک رسانی

مقدمه

در سال های اخیر، باتوجه به جذابیت هایی که در زمینهٔ هوش ماشین و رباتیک وجود دارد تحقیقات زیادی در این حوزه انجام شده است. بخش قابل توجهی از این تحقیقات درزمینهٔ سیستم های چند عامله است که مورد توجه محققین شاخه های مختلف علوم کامپیوتر و مهندسین کنترل می باشند. به طور کلی اگر ذات یک وظیفه که تنها از یک موجودیت مرکزی و توانمند برمی آید، قابل تقسیم میان چند موجودیت کوچک تر و ترجیحا هوشمند و خودمختار باشد، یک سیستم چندرباته یا در حالت کلی تر چند عامله، جایگزین بسیار مناسبی است.

لزوم ایجاد تحمل پذیری خطا در سیستمهای چند عامله:

با وجود پژوهشهای فراوانی که دردانشگاهها و مراکز تحقیقاتی در مورد سیستم های چندعامله یا چندرباته انجام شده، هنوز در عمل و در بسیاری از کاربردهای دنیای واقعی، این تکنولوژی جایگزین روشهای موجود نشده است و همچنان سیستم های متمرکز با تمامی انتقادهایی که به آنها وارد است، در اغلب کاربردها دیده می شوند. شاید دلیل عمدهٔ این امر را بتوان در آسیب پذیری فراوان آنها نسبت به اشکال ها و خرابی های رایجی دانست که در مکانیزیم های مختلف سازندهٔ رباتها رخ می دهد، یا اینکه کنارنیامدن عاملها را با تغییرات فراوانی که در محیط عملیاتی پویای آنها رخ می دهد، علت این مساله دانست. هر یک از این موارد یا دلایل مشابه اینها، لزوم افزودن قابلیت تحمل پذیری خرابی به رباتها و عاملها و در نتیجه طراحی سیستم های چند عاملهٔ با قوام بیشتر را تقویت می کند. در صورتی که این نیازمندی ها برآورده شود، با جایگزین نمودن سیستم های چند عامله یا در حالت خاص، چندعامله به جای سیستم های متمرکز، می توان تحمل پذیری خطا را بهبود بخشید و در نتیجه کارایی سیستم را حتی با وجود خرابی در سیستم، در حد قابل قبولی باقی نگه داشت. مقاله حاضر، به مسالهٔ ایجاد تحمل پذیری خطا به کمک روشهای ساده، عملی و در عین حال کم هزینه، در کاربردهایی می پردازد که قابلیت توزیع شدگی را دارند. منظور از کاربردهایی که قابلیت توزیع شدگی را دارند، کاربردهایی است که هدف هایی سیستم قابل تقسیم به چند زیر هدف باشد و این زیر هدف ها هر یک به

وسیلهٔ موجودیت های مستقل با دانش های محدود، قابل اکتساب باشد. چنین سیستم هایی با بهره گیری از راهکارهایی که در این مقاله معرفی شده اند، می توانند طوری تغییر یابند که حتی در صورت بروز خرابی عمده برای تعداد زیادی از عناصر سیستم تا جایی که سیستم حداقل تعداد عناصر سالمی را شامل باشد، از کار باز نماند و حداقل بحرانی ترین وظایفی که در زمان طراحی برایش تعیین شده را به درستی به انجام برساند.

تحمل پذیری خطا در سیستم های چندعامله با روشهای رایج:

اگر توجه خود را در حالت کلی به سیستم های چند عامله معطوف نماییم، هر سیستم چند عامله را می توان به منزلهٔ تیمی از عامل های همکار دانست که هر یک وظیفهٔ خاصی به عهده دارند و از تعامل آنها با یکدیگر یک عملکرد واحد حاصل می شود.

این شیوهٔ نگرش موجب می شود که بتوان از ایدهٔ سیستم های چندعامله برای توسعهٔ یک سیستم گستردهٔ تحمل پذیر خطا استفاده نمود. یکی از تکنیک های رایج برای ایجاد تحمل پذیری خطا در یک سیستم، استفاده از افزونگی است.

اما علاوه بر افزونگی صرف، در سیستم های چندعامله، تکنیک های پیشرفته تری هم برای ایجاد تحمل پذیری خطا وجود دارد و آن استفاده از خود عامل های موجود در سیستم برای رفع کاستی های ناشی از خطای عامل همتیمی آنهاست.

ایدهٔ جدیدی برای تحمل پذیری خطا در سیستم های چندعامله:

در سیستم های چندعامله، بهره گیری صحیح از خاصیت چند عاملی بودن و عمل نمودن عامل ها به صورت مستقل، می تواند نقطهٔ امیدی باشد که در صورت وقوع یک خطا، مشکل به کل سیستم توسعه نمی یابد.

بنابراین ایدهٔ اساسی این مقاله آن است که، به جای جایگزین نمودن یک عامل خراب با یک عامل دیگر از خارج، از قابلیت های در حال حاضر اضافهٔ دیگر عاملهای موجود در طرح استفاده کرده، نقش عامل خراب را نیز به نحوی بر دوش همکارانش قرار می دهیم تا شرایط بحرانی با کمترین تاثیر نامطلوب بر کارایی مرتفع گردد. در اصل علت استفاده از یک محیط چند عامله برای بیان ایده های این مقاله آن است که در چنین محیطی عناصر بالقوهٔ دیگری برای تقبل نقش یک عامل خراب در سیستم وجود دارد و استفادهٔ بهینه و منطقی از منابع موجود در سیستم، تحمل پذیری خطا را فراهم می کند. البته در هر سیستم چند عامله مسالهٔ تقسیم اولیهٔ وظایف باید با دقت انجام گیرد که این می تواند یا به صورت دستی در زمان طراحی یا به صورت اتوماتیک در زمان اجرا توسط خود عاملها انجام شود. اگر مسالهٔ انتخاب وظیفهٔ اولیه را به عهدهٔ خود عاملها بگذاریم،می توان با مانیتور نمودن کارایی در زمان راه اندازی سیستم توسط عاملها که قابلیت یادگیری دارند، انتظار داشت که هر عامل مناسب ترین کار را از لحاظ هماهنگی با ویژگی های خودش بر عهده بگیرد. آنچه در این مقاله انجام شده است، تقسیم وظیفهٔ دستی در زمان طراحی، میان عاملهاست.

مسالهٔ مهم دیگر که در مسائل مشابه، گاه به آن پرداخته می شود، مسالهٔ مهم تقسیم یک وظیفه میان چندعامل است. در صورتی که یک وظیفه به علت خرابی یک عامل بر زمین مانده باشد و این وظیفه قابلیت تقسیم میان چند عامل دیگر را داشته باشد، می توان بخشهای قابل انجام و مستقلی از آن وظیفه را به عاملهای مختلف محول نمود و در آخر نتایج خروجی عاملها را با هم ترکیب نمود تا یک نتیجهٔ کامل به دست آید. اما در این مقاله برای سادگی، وظیفه هایی برای عامل ها در نظر گرفته ایم که به اصطلاح اتمیک هستند و مستقلا باید توسط یک عامل انجام گیرند ولی تمامی عاملها در صورت داشتن زمان کافی و دارا بودن توان پردازشی کافی برای تکمیل آن قبل از تکمیل مهلت زمانی اش، قادر به انجام تمامی وظیفه ها هستند. البته بدیهی است که به علت دارا بودن قابلیتهای اعتماد مختلف احتمال تکمیل یک وظیفه توسط هر عامل با عامل دیگر متفاوت است. بنابراین در این مقاله، بهره گیری از خود عامل های درگیر در حل مساله به منظور رفع خطا در سیستم مورد تاکید است بدون اینکه از عامل واسطه ای به منظور ایجاد هماهنگی در تنظیم وظایف جدید استفاده شود.

به طورخلاصه، یک عامل در صورت مواجهه با مشکلی کـه وی را از ادامهٔ همکاری بازدارد، یک پیام درخواست

کمک صادر می کند و سایر هم تیمی های خود را از این واقعه مطلع می کند. به بیان کلی تر، دیگر عاملها از خرابی هر یک از هم تیمی های خود آگاهی می یابند. سپس از طریق فازهای

تصمیم گیری گسترده که اصل تاکید این تز است، اقدام به بررسی درخواست کمک و سپس انجام کمک رسانی می نمایند. لازم به توضیح است که در این مقاله، به مسالهٔ تشخیص خطا به شکل مستقل، توسط یک مولفهٔ جداگانه پرداخته نشده است و فرض بر این است که خرابی یک عامل توسط خودش قابل تشخیص است. این فرض برای سادگی انجام شده است، چرا که پرداختن به روشهای متعددی که برای تشخیص خرابی وجود دارد، این مقاله را از مسالهٔ اصلی خود که کنارآمدن با خرابی در سیستم های چند عامله است، منحرف می نمود. به بیان ساده ترین روش کمک رسانی می پردازد که تنها مبتنی بر درجه اهمیت وظیفهٔ عامل هاست. به بیان چند استراتژی سادهٔ کمک رسانی مبتنی بر ریسک می پردازد و در مفهوم بی صبری برای کمک رسانی با تعریف ساده ای معرفی شده است که این مفهوم در توسعه یافته و از دانش عاملها به نحو مناسبی استفاده می نماید. در استراتژی های ساده و ترکیبی برای تعیین ترتیب کمک رسانی معرفی شده اند. در روش غیرقطعی کمک رسانی مبتنی بر الگوریتم تقسیم وظیفهٔ نسبتا بهینه مطرح شده است و شامل جمع بندی روشهای مبتنی بر تصمیم گیری قطعی برای کمک رسانی و تقسیم وظیفه است و بالاخره به جمع بندی تمامی روشهای ارائه شده و مقایسهٔ آنها می پردازد

بررسی روشهای رایج تحمل پذیری خطا:

امروزه با هر چه پیچیده تر شدن سیستم های کامپیوتری، وجود عدم قطعیت در اغلب بخشهای یک سیستم و لزوم دخالت عوامل متعدد در حصول خروجی، طراحی سیستمی که با بروز مشکلی در هر یک از قسمتهای آن، دچار نقص عمده نشده و قادر باشد عملکرد صحیح خود را حفظ نموده وصرفاً با تغییری در کارایی کلی ، هدف هایی را برآورده سازد، بسیار ضروری است. با این ترتیب می توان تعریفی از یک سیستم تحمل پذیر خطا به شرح زیر ارائه نمود:

یک سیستم تحمل پذیر خطا، سیستمی است که بتواند وظایفی که برایش مشخص شده است، حتی با وجود خطاهای نرم افزاری و خرابی های سخت افزاری به درستی به انجام برساند. تحمل پذیری خطا، به دلیل مصرف رو به فزونی هر چه بیشتر کامپیوترها در کاربردهای مختلف در زندگی بشر، بسیار مورد توجه قرار گرفته است و در واقع به علت اهمیت کاربردها، تحمل پذیری خطا یک ویژگی مهم در سیستم ها محسوب می شود.

نتیجه گیری و کارهای آینده:

در این مقاله، روش هایی برای ایجاد تحمل پذیری خرابی در سیستم های چند عامله به ویژه سیستم های چندعامله ارائه شدکه تفاوت اصلی این روشها با آنچه قبلا و در اغلب پژوهشهای مرتبط انجام شده است، متکی نبودن بر افزونگی صرف و در عوض استفاده از قابلیتهای اضافه تر هر عامل موجود در طرح است به شکلی که عاملها در صورت بروز خرابی، با همکاری با یکدیگر و تعویض نقش خود از حالت عادی مسئولیتهای بیشتری را بر دوش بگیرند.

در طرح های پیشنهادی، افزونگی نه به صورت صریح و متمرکز به شکل عامل های افزونه، بلکه به صورت توزیع شده و ضمنی در قابلیت عاملهای اصلی سیستم قرار می گیرد تا با استفاده از این قابلیت ذخیره شده، عاملها بتوانند در صورت بروز خرابی احتمالی برای هر یک از هم تیمی های خود،وظیفهٔ یک یا تعدادی از آنها را نیز (بسته به میزان قابلیت های خودش) بر دوش گیرد و از کاهش کارایی سیستم تا زمان برطرف شدن خرابی، جلوگیری نماید.

بنابراین وجود افزونگی ضمنی در زمان تقسیم اولیهٔ وظیفه میان عاملها، در صورتی که در زمان بروز خطا به نحو شایسته ای به کار گرفته شود، موجب کاهش هزینهٔ زمان طراحی سیستم در مقایسه با سیستمی می شود که برای هر یک از عناصر خود، از یک یا دو عنصر افزونه استفاده نموده است.

علاوه بر این، استفاده از طرحهای پیشنهادی، معیارهای ارزیابی معرفی شده را تا حد قابل قبولی ارضا می نماید و این حاکی از مفیدبودن این ایده ها در سیستمهایی با شرایط مشابه است. به عنوان مقایسه ای اجمالی از هر یک از روشهای ارائه شده،

می توان موارد زیر را یادآور شد:

روش اول : تقسیم وظیفه بر اساس سرعت ذاتی – بدون ملاحظه از نظر سازگاری کمک و کمک رسان و با تصمیم گیری قطعی

روش دوم : تقسیم وظیفه بر اساس سرعت ذاتی – اعمال ملاحظاتی از نظر سازگاری کمک و کمک رسان با تصمیم گیری قطعی

روش سوم : تقسیم وظیفه بر اساس الگوریتم نسبتا بهینه با تصمیم گیری قطعی

www.NoavaranGermi.ir سایت نوآوران گرمی - مرجع فیلم های آموزشی و همیار دانشجو

روش چهارم : تقسیم وظیفه بر اساس الگوریتم نسبتا بهینه با تصمیم گیری غیرقطعی و سیاست اجرا تا تکمیل وظیفهٔ فعلی حتی در صورت رسیدن وظیفهٔ جدید

روش پنجم : تقسیم وظیفه بر اساس الگوریتم نسبتا بهینه با تصمیم گیری غیر قطعی و بررسی میزان اضطراب در صورت رسیدن وظیفهٔ جدید

به عنوان ادامهٔ کار هر یک از موارد زیر می تواند مدنظر باشد:

۱) کنار آمدن با خرابی های جزئی

۲) همکاری میان عاملهای کمک رسان در تکمیل یک وظیفه

۳) انتقال مسالهٔ تقسیم وظیفهٔ اولیه از فاز طراحی به ابتدای زمان اجرا

۴) استفاده از مکانیزم های بازار در ایجاد انگیزهٔ کمک برای عاملها

مراجع:

[۱] Barry Johnson, "Design and Analysis of Fault Tolerant Digital Systems", Adison Wesely, ۱۹۸۹

[۲] Dhiraj K. Pradhan, "Fault-Tolerant Computer System Design", ۱۹۹۵, Prentice Hall Levi Agrawala, "Fault Tolerant System Design", ۱۹۹٤, McGrawHill

[۳] Siewiorek and Swarz, "Reliable Computer Systems", ۱۹۹۲, Digital Press

[٤] Stothert and MacLeod, "Multi_Agent Techniques for Fault Tolerance in Computer Control Systems", Department of Electrical Engineering, University of The Witwatersrand, Johannesburg South Africa, ۱۹۹۷

[٥] Chrysanthos Dellarocas and Mark Klein, "An Experimental Evaluation of Domain-Independent Fault Handling Services in Open Multi-Agent Systems", ICMAS۲۰۰۰

[٦] Staffan Hugg, "A Sentinel Approach to Fault Handling in Multi-Agent Systems",Department of Computer Science, University of Karlskrona/Ronneby, Sweden

[۷] Sanjeev Kumar, Philip R. Cohen, "Towards a Fault-Tolerant Multi-Agent System Architecture", Oregon Graduate Institute, Autonomus Agent ۲۰۰۰