ISSN: 2821-157X htpps://www.jasd.khadu.ac.ir

Journal of Aerospace Defense
Volume 1, Issue 3
Spring 2024
P.P. 66-96

Research Paper;

Modeling the Distributed Incident Command System and Present a Learning Approach

Maryam Shokoohi[1], Mohsen Afsharchi[2], Hamed Shah-Hosseini[3]

1. Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. E-mail: maryam.shokoohi@srbiau.ac.ir
2. Department of Electrical and Computer Engineering, University of Zanjan, Zanjan, Iran. E-mail: afsharchi@znu.ac.ir
3. Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran, Iran. E-mail: shahhosseini@srbiau.ac.ir

Article Information

Accepted:
2023/05/26

Recceived:
2023/08/23

Keywords:

Incident command system, learning automata, distributed constraint optimization, passive defense

Corresponding Author:
Mohsen Afsharchi

Email:
afsharchi@znu.ac.ir

Abstract

The Incident Command System (ICS) requires agents who routinely deal with a large number of search and rescue tasks. In addition, responses must operate in highly uncertain and dynamic environments where new tasks emerge and may spread across the disaster landscape. Therefore, finding the optimal and correct allocation to complete all activities in a reasonable time is a big computational challenge.

In Iran, the Incident Command System acts asistematic and lack of intelligence, and the presence of a correct decision-making system that has functionality and makes correct and quick decisions is very important.

This article presents a method for solving the allocation problem, which is a distributed constraint optimization problem. This method uses Markov decision techniques and learning techniques such as learning automata.

The results of simulations and experiments show that the existence of the learning technique and the decentralized behavior of agents can replace the past methods and compensate for the lack of previous methods. The proposed method can perform 85% better than the centralized method and previous methods and is much better in terms of convergence and time.

Shokoohi, Maryam; Afsharchi, Mohsen and Shah-Hosseini, Hamed.(2024). Modeling the Distributed Incident Command System and Present a Learning Approach. *Journal of Aerospace Defense*, Journal of Aerospace Defense, Vol. 3, No1. 2024.

ISSN: 2821-157X | htpps://www.jasd.khadu.ac.ir

فصلنامه علمی دفاع هوافضایی
دورهٔ ۳، شمارهٔ ۱
بهار ۱۴۰۳
صص ۹۶-۶۶

مقاله پژوهشی؛ doi

مدلسازی سیستم فرماندهی حادثه توزیع شده و ارائه روشی برای حل آن با رویکرد یادگیری

مریم شکوهی[۱]، محسن افشارچی[۲]، حامد شاه‌حسینی[۳]

۱. دانشجوی دکترا، گروه مهندسی کامپیوتر، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران.
۲. استاد، گروه مهندسی برق و کامپیوتر، دانشگاه زنجان، زنجان، ایران.
۳. دکتر، گروه مهندسی کامپیوتر، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران.

اطلاعات مقاله

تاریخ دریافت:
۱۴۰۳/۰۳/۰۶

تاریخ پذیرش:
۱۴۰۳/۰۶/۰۲

کلیدواژه‌ها:
سیستم فرماندهی حادثه، آتوماتای یادگیر، بهینه‌سازی محدودیت توزیع‌شده، پدافند غیرعامل

نویسنده مسئول:
محسن افشارچی

ایمیل:
afsharchi@znu.ac.ir

چکیده

سیستم فرماندهی حادثه نیازمند وجود عامل‌هایی است که به طور معمول با تعداد زیادی از وظایف جستجو و نجات، روبرو می‌شوند. علاوه بر این، واکنش‌دهنده‌ها باید در محیط‌های بسیار نامطمئن و پویا که در آن وظایف جدید ظاهر می‌شوند و خطرات ممکن است در فضای فاجعه پخش شوند، کار کنند. بنابراین یافتن تخصیص بهینه و صحیح برای تکمیل تمام فعالیت‌ها در یک زمان مناسب یک چالش محاسباتی بزرگ است.

در ایران، سامانه فرماندهی حادثه به صورت غیرسیستماتیک و غیرهوشمندانه فعالیت می‌کند و حضور یک سامانه تصمیم‌ساز صحیح که دارای توانایی یادگیری بوده و توانایی تصمیم‌های صحیح و سریع را داشته باشد، از اهمیت بالایی برخوردار است.

در این مقاله روشی برای حل مسئله تخصیص، که از مسائل بهینه‌سازی محدودیت توزیع شده می‌باشد، ارائه می‌شود. در این روش از تکنیک‌های تصمیم‌گیری مارکوف و تکنیک‌های یادگیری همچون آتوماتای یادگیر استفاده می‌گردد.

نتایج شبیه‌سازی‌ها و آزمایشات نشان می‌دهد که وجود تکنیک یادگیری و رفتار غیرمتمرکز عامل‌ها، می‌تواند جایگزین روش‌های گذشته شده و کمبود روش‌های قبل را جبران کند. روش پیشنهادی به خوبی می‌تواند در مقابل روش متمرکز و روش‌های گذشته از لحاظ میانگین فعالیت انجام شده در حدود ۸۵ درصد و از لحاظ همگرایی و زمان بسیار بهتر عمل کند.

استناد: شکوهی، مریم؛ افشارچی، محسن و شاه حسینی، حامد.(۱۴۰۳). مدلسازی سیستم فرماندهی حادثه توزیع شده و ارائه روشی برای حل آن با رویکرد یادگیری. *دفاع هوافضایی*، دوره ۳(شماره۱)، صفحه۹۳-۸۳.

۱–مقدمه

مدیریت مؤثر شرایط اضطراری، دارای الزامات و نیازمندی‌های متعددی می‌باشد. یکی از مهم‌ترین این الزامات، وجود یک سیستم مشخص فرماندهی حادثه است تا بتوان با استفاده از آن، اختیارات، شرح وظایف و مسئولیت‌ها را به خوبی مشخص کرده و موضوعات درونی و بیرونی حادثه را کنترل نمود. یکی از بهترین الگوهایی که تاکنون برای فرماندهی حادثه ایجاد شده است، سیستم فرماندهی حادثه[1] است که تجارب موفقی در مدیریت مؤثر و کارای حوادث گوناگون داشته است. این سامانه، ساختاری است که توسط سیستم مدیریت حوادث ملی آمریکا مورد استفاده قرار گرفته و بسیاری از کشورهای دیگر دنیا هم از آن به عنوان یک راهنمای عملیاتی مناسب بهره می‌برند [۱].

ساماندهی فرماندهی حادثه که به عنوان ICS در کشورهای جهان شناخته می‌شود، مجموعه‌ای اصولی متشکل از افراد متخصص و سازمان‌های مربوطه و سیستم‌های ارتباطی و کامپیوتری است که برای فرماندهی، کنترل و هماهنگی در پاسخ به شرایط اضطراری استفاده می‌شود [۲].

در سیستم مذکور، فرمانده باید در تمام عملیات، قدرت تصمیم‌گیری سریع، ارزیابی لازم و کافی، تصحیح اشتباهات و پشتیبانی از اقدامات صحیح، سازماندهی موثر و جلوگیری از بروز بی‌نظمی و آشفتگی، اقدام به ریسک و عملیات ریسکی و استفاده درست و صحیح از منابع را داشته باشد. در واقع فرمانده باید در تاکتیک عملیاتی جزئیات تصمیم‌های استراتژیک را در نظر بگیرد [۳].

فرماندهان هنگام مدیریت بلایای بزرگ با تعدادی چالش مهم روبرو هستند. اولاً، تعداد کارهای نجات، معمولاً از تعداد پاسخ‌دهندگان و منابع در دسترس آنها بیشتر است. ثانیاً، هر کار احتمالاً به سطح متفاوتی از تلاش نیاز دارد تا در مهلت مقرر تکمیل شود. ثالثاً، وظایف جدید ممکن است به طور مداوم از محیط ظاهر یا ناپدید شوند، بنابراین پاسخ‌دهندگان باید به سرعت تخصیص منابع خود را محاسبه کنند. چهارم، تشکیل تیم‌ها یا ائتلاف‌هایی متشکل از چندین عامل، حیاتی است، زیرا هیچ سازمانی تمام منابع مورد نیاز برای نجات قربانیان، رفع انسداد جاده‌ها و خاموش کردن آتش‌هایی را که ممکن است در فضای فاجعه رخ دهد را در اختیار نخواهد داشت. با توجه به این موضوع فرماندهان باید مدیریت منابع را به گونه‌ای برنامه‌ریزی کنند که در محیط حادثه، تعداد جان‌ها و بخشی از زیرساخت‌های نجات‌یافته به

---

[1] Incident Command System- ICS

حداکثر برسد. به ویژه، مهم است که این برنامه‌ریزی‌ها به صورت غیرمتمرکز انجام شود تا از یک نقطه شکست در سیستم جلوگیری شود. ضمن این که سامانه ICS با کمک اپراتورها و به صورت غیرسیستماتیک در حال حاضر در کشور ایران به صورت غیرهوشمند انجام وظیفه می‌نمایند و در بسیاری از شهرها و سازمان‌ها این سامانه وجود ندارد [۴]. بنابراین طراحی سیستم هوشمندانه و دارای توانایی یادگیری، که بتواند در جهت برنامه‌ریزی صحیح تخصیص، به فرماندهان حادثه یاری نماید، از اهمیت بالایی برخوردار است. این سیستم می‌تواند در هنگام وقوع حادثه، با تصمیم‌گیری سریع و توزیع شده، فرماندهان را جهت پیشبرد سریع وظایف و تشکیل تیم‌های صحیح یاری دهد.

در این مقاله، یک راه‌حل غیرمتمرکز جدید برای فرآیند تشکیل ائتلافی که مدیریت بلایا را فراگرفته است و در جهت تخصیص فعالیت‌ها به عوامل حرکت می‌کند، ارائه می‌شود. در این مقاله یک سناریو حادثه، شبیه‌سازی شده که در آن عوامل به صورت غیرمتمرکز برای تخصیص فعالیت‌ها تصمیماتی اخذ می‌نمایند. در ابتدای امر این سناریو به صورت یک مسئله بهینه‌سازی محدودیت توزیع شده[۱](DCOP) فرموله می‌شود و سپس با استفاده از آتوماتای یادگیر[۲] (LA) حل می‌شود.

در این مقاله، یک محیط پویا در نظر گرفته شده و از مدل‌های مارکف برای مدل کردن وابستگی حالت‌ها در DCOP پویا استفاده می‌شود که در آن DCOP در گام بعدی تابعی از تخصیص مقادیر در مرحله زمانی فعلی است [۵–۷]. در این مدل‌ها، در هر گام زمانی عامل‌هایی که مسئول تخصیص مقادیر به متغیرهای خود هستند، به عنوان عناصر تصمیم‌گیری (عامل‌ها) شناخته می‌شوند و قادر به یادگیری دنباله‌ای از مقادیر هستند و سرانجام به یک سیاست مشترک سراسری همگرا می‌شوند.

همچنین از عواملی استفاده می‌شود که از شبکه واحدهای یادگیری مستقل به نام اتوماتای یادگیر تشکیل شده‌اند، چرا که دارای ویژگی‌های همگرایی بلندمدت و پویایی هستند [۸, ۹]. علاوه بر این، آتوماتای یادگیر در عین سادگی از لحاظ ارتباط بین اتوماتاها انعطاف‌پذیری زیادی ایجاد می‌کند. این مزایا بیشتر با این واقعیت تکمیل می‌شوند که الگوریتم‌های اتوماتا به اطلاعات کمتری نیاز دارند و حتی نشان داده شده است که در محیط‌هایی با قابلیت مشاهده جزیی نیز کار می‌کنند [۱۰, ۱۱]، به همین دلیل برای سیستم‌های چندعامله[۳] مناسب هستند.

---

[1] Distributed Constraint Optimization Problem
[2] Learning Automata
[3] Multi Agent Systems

ساختار مقاله به این شرح است: در بخش دوم کارهای انجام شده در پژوهش‌های گذشته مورد بررسی قرار می‌گیرد؛ در بخش سوم مروری بر موضوعات مطرح در پژوهش می‌باشد. روش بیان شده و الگوریتم ارائه شده در بخش چهارم مورد بحث قرار می‌گیرد. در بخش پنجم نتایج و شبیه‌سازی‌ها انجام می‌گیرد و در پایان نتیجه‌گیری خواهد بود.

۲- کارهای پیشین

هفت اصل مهم در سامانه فرماندهی حادثه مطرح می‌باشد: الف) استانداردسازی، ب) ویژگی عملکردی، ج) اداره کردن دامنه قابل کنترل، د) یکپارچگی واحد، ه) فرمان واحد، و) مدیریت اهداف، ز) مدیریت منابع جامع [۱۲]. این اصول برگرفته از منابع آتش‌نشانی کالیفرنیا برای مقابله با حوادث و بحران‌های بالقوه است [۱۳] و در دهه ۱۹۷۰ میلادی شکل گرفته و گسترش پیدا کرده است؛ اما به روش‌های گوناگون انجام می‌شده است [۱۴]. بعد از گذشت مدت‌ها بوناچینی با تجدید نظر در ساختار محدوده آتش و تاثیرات آن بر فرماندهی حادثه نسبت به بکارگیری ساختار مدیریت حادثه برای یک چنین سیستم‌هایی اقدام کرد [۱۵, ۱۶]. بنابراین در تمامی نقاط عملکردی یک سامانه فرماندهی حادثه به وضوح درک می‌شود که این سامانه دارای یک روند ارتقاپذیر بر اساس تحقیقات علمی و عملکردی در حوزه‌های مرتبط بوده و به مرور زمان نسخه‌هایی بهبودیافته از خود را جهت تثبیت شرایط بحرانی در موقعیتی قابل قبول ارائه داده است [۱۷].

طبق تحقیقات انجام شده در مرجع [۱۸] سامانه فرماندهی حادثه برای انجام تمام واکنش‌های خود ضعیف عمل کرده و باید مجموعه‌ای از شرایط پیچیده و گوناگون قبل از حادثه انجام گیرد تا سیستم طراحی شده قابلیت عملگرایی خود را حفظ کند. چرا که ضروری است این سیستم‌عامل‌ها به گونه‌ای با یکدیگر همگروه شوند که بتوانند یک هدف خاص را دنبال کنند. در زمان وقوع حادثه، پاسخ به حوادث و بلایا با توجه به تاثیرات اقتصادی و اجتماعی خود بر جوامع از ضروریات مهم است. برای انجام این کار معمولا تیمی از عوامل و پاسخ‌دهندگان نیاز است [۱۹, ۲۰]. بنابراین مجموعه منابع موجود (به عنوان مثال، پاسخ‌دهندگان یا تجهیزات) باید در بین تیم‌ها توزیع شود و باید هر کدام دارای تخصص و توانایی برای رعایت اهداف تعیین شده باشند. برای این کار می‌توان از بازی‌های تابع-مشخصه[1] (CFG) [۲۱] استفاده کرد که در آن فرد از یک تابع ارزش‌گذاری برای ارزیابی

[1] Characteristic-Function Games

میزان کارکرد عوامل با یکدیگر استفاده می‌کند. نتیجه یک ساختار ائتلافی[1] (CS) می‌باشد که مجموعه‌ای از عوامل را به ائتلاف‌ها تقسیم می‌کند. در مقاله [۲۲]، بر روی مدل‌سازی برنامه‌های کاربردی واکنش به بلایا از لحاظ مسئله ایجاد ساختار ائتلافی (CSG) [۲۳]، کار می‌شود. در مراجع [۱۳, ۲۴] کاربردهای خاصی که نیاز به استفاده از سیستم فرماندهی حادثه (ICS) دارند، چارچوبی عمدتاً برای هدایت واکنش به حوادث فاجعه به کار گرفته شده است.

معمولاً عملیات واکنش به بلایا، مسائلی را ایجاد می‌کند که در آن‌ها عواملی را برای انجام مجموعه‌ای از وظایف اعلام شده توسط مرکز فرماندهی، تعیین می‌کنند. در مرجع [۲۵]، نویسندگان یک حادثه واکنش به بلایا را مدل‌سازی می‌کنند که در آن وظایف، اولویت‌های متفاوتی دارند و از وظایف فرعی تشکیل شده‌اند. برای انجام یک کار، یک عامل به تعدادی منابع نیاز دارد و بنابراین عواملی که به رویداد پاسخ می‌دهند ائتلاف‌هایی را برای برآورده کردن یک نیاز تشکیل می‌دهند. نویسندگان در پژوهش [۲۶] به طور خاص بر روی روبات‌ها تمرکز می‌کنند. هر روبات دارای دو قابلیت است، یکی برای حس کردن (به عنوان مثال، دوربین) و دیگری برای انجام وظیفه (مانند بازوها). به طور مشابه، برای تکمیل یک کار، قابلیت‌های حسی و عملی مورد نیاز باید برآورده شود. برای مقابله با وظایف اعلام شده در طول یک رویداد فاجعه الهام گرفته از سیل (به عنوان مثال، جمع‌آوری نمونه‌های آب)، نویسندگان وظایف تشکیل شده را بررسی می‌کنند که هر کدام نقش‌های خاصی را برای انجام آن، می‌طلبد [۲۰]. نقش‌ها بر روی قابلیت‌هایی که ربات‌ها در سیستم دارند، ترسیم می‌شوند. وظایف تحت تاثیر محدودیت‌های مکانی و زمانی نیز مورد مطالعه قرار می‌گیرند [۲۷]. انگیزه این نوع بازی مسابقه نجات ربوکاپ [۱۹] است که از یک حادثه زلزله در ژاپن الهام گرفته شده است. مشکل ایجاد ساختار ائتلاف، قبلاً برای مدل‌سازی پاسخ به یک حادثه فاجعه استفاده شده است. نویسندگان در پژوهش [۲۱] سناریویی را مدل‌سازی می‌کنند که در آن ماهواره‌ای که با سوخت رادیواکتیو تغذیه می‌شود، در یک منطقه زیرشهری سقوط کرده است (این سناریو برای اولین بار به عنوان یک مسئله برنامه‌ریزی مبتنی بر عامل معرفی شد [۲۸]). خدمات اورژانسی که به این حادثه پاسخ می‌دهند متشکل از پزشکان، سربازان، حمل و نقل و آتش‌نشانان هستند. این سناریو به عنوان شبکه‌ای مدل‌سازی می‌شود که در آن پاسخ‌دهندگان باید مجموعه‌ای از وظایف نجات را با انداختن اهداف در مکان‌های خاص در شبکه انجام دهند. هدف می‌تواند قربانی، حیوان، سوخت یا منابع دیگر باشد. بنابراین، چهار هدف در نظر گرفته شده است که هر کار به مجموعه‌ای از نقش‌ها نیاز دارد تا انجام شود. هر

---

[1] Coalition Structure

عامل در حین ایفای هر یک از چهار نقش دارای سطح متفاوتی از قابلیت‌ها است. این اطلاعات ممکن است نشان‌دهنده آموزش آن برای آن نقش، تجربه گذشته، و غیره باشد.
جهت مدل‌سازی سیستم فرماندهی حادثه کار مشخصی به صورت هوشمندانه و دینامیک انجام نشده است. در این مقاله می‌توان جهت مدل‌سازی از Dynamic DCOP استفاده کرد و برای حل آن که جزو مسائل سیستم‌های چندعامله و در ضمن در ساختار مسئله Multi MDP می‌باشد، می‌توانیم از شبکه‌ی آتوماتاهای یادگیر و خصوصیت ارزشمند آن از جمله سادگی ساختار، نیاز کم به اطلاعات و بازخورد از محیط برای رفع محدودیت‌های تکنیک‌های موجود DDCOP استفاده کنیم. روش پیشنهادی می‌تواند برای مقابله با محیط‌های غیرمتمرکز و پویا استفاده شود. از آنجا که این مدل یک تکنیک جدید را فراهم می‌کند، قابل تصور است که می‌توان آن را برای حل مسائل دیگر DCOP در حوزه‌ها و زمینه‌های مختلف مورد استفاده قرار داد.

۳- ادبیات موضوع

۳-۱ مسئله تخصیص

مسئله تخصیص، مسئله‌ای است که در آن، یک پروژه با تعدادی فعالیت به زمانبندی و تخصیص منابع نیاز دارند. فعالیت‌ها در این مسائل می‌بایست طبق محدودیت‌های از پیش تعریف شده زمانبندی شوند [۲۹]. مسئله تخصیص در واقع یک مسئله NP کامل است و یکی از پرکاربردترین مسائل در زمینه تحقیق در عملیات می‌باشد [۳۰]. هدف از این مسئله تخصیص n عامل به m فعالیت می‌باشد که در آن تابع هدف کمینه کردن مجموع هزینه تخصیص است و مدل ریاضی این مسئله بدین شکل است:

$$c=\{c_1,c_2,..,c_r\} \qquad (۱)$$

به طوری که :

$$\sum_{i=1}^{n} x_{ij}=1 \qquad j=1,..,n,$$
$$\sum_{j=1}^{m} x_{ij}=1 \qquad i=1,..,m\ ,$$
$$x_{ij}=0or1$$

به این صورت که مقدار $x_{ij}=1$ خواهد بود در صورتی که عامل i به فعالیت j اختصاص یابد. در غیر اینصورت صفر خواهد بود. همچنین مقدار c_{ij}، برابر با هزینه تخصیص عامل i به فعالیت j می‌باشد. مجموعه اول محدودیت‌ها به این معنی است که هریک از فعالیت‌ها دقیقا

به یک عامل تخصیص می‌یابد و مجموعه دوم محدودیت‌ها مشخص می‌کند که هریک از تمامی عامل‌ها دقیقا به یک فعالیت مختص شوند. در این مقاله هدف تعیین تخصیص بهینه عامل‌ها به فعالیت‌هاست به طوری که در شرایط وقوع حادثه و با وجود محدودیت منابع، منجر به کمینه شدن میزان تاخیر انجام فعالیت‌های امدادی شود.

۳-۲ سامانه فرماندهی حادثه

سامانه فرماندهی حادثه[1] در حال حاضر رایج‌ترین نظام اِعمال مدیریت سوانح و حوادث جهان است که مقبولیت آن با توجه به نتایج، رو به افزایش است و در پدافند غیرعامل کارکرد محسوسی دارد. جایگزینی گسترده سامانه فرماندهی حادثه به عنوان مدل فرماندهی، نظارت و هماهنگی منابع نیروی انسانی در موارد خاص، بوده است. در واقع این سامانه عبارتست از مکانیسمی برای هماهنگی موثر عملیات مقابله در شرایط غیرمعمول و ترکیبی از تسهیلات، امکانات، پرسنل، فرآیندها و ارتباطات در یک ساختار فرماندهی واحد با مسئولیت مدیریت منابع تعیین شده که به منظور انجام اثربخش اهداف مربوط به یک حادثه می‌باشد [۴, ۱۸].

سامانه فرماندهی حادثه یکپارچگی فرماندهی را بگونه‌ای فراهم می‌آورد که کلیه پرسنل واقع در یک سیستم به نحو مطلوب، مدیریت شده و برای کاری مرتبط به نحو مطلوب مورد استفاده قرار گیرند. همچنین مجموعه‌ای استاندارد از مفاهیم و اصطلاحات برای برقراری ارتباط عناصر با هم و نحوه استفاده از منابع و تسهیلات را فراهم می‌آورد. همچنین این سامانه قابل توسعه بر اساس سیستم مدل‌سازی شده است، یا به عبارت دیگر قابلیت مدل شدن دارد.

از آنجایی که در سازمان‌های مختلف، مسائل مختلفی از جمله موارد ذیل وجود دارد، تشکیل سامانه فرماندهی حادثه از ضروریات سازمان می‌باشد:

- فقدان کلیت ساختاری و سازمانی: تدوین استراتژی برای ساختار سازمانی ناکارا، یکی از راه‌های موثر برای اصلاح نشانه‌های ضعف ساختار سازمانی است. در این روش، ابتدا باید مشکلات و نقاط ضعف ساختار سازمانی شناسایی شوند. سپس با توجه به این مشکلات، استراتژی‌هایی برای بهبود ساختار سازمانی تدوین شود.
- ارتباطات ضعیف درون‌سازمانی و در صحنه عملیات: یک ساختار سازمانی رسمی باید شامل تعریف دقیق از سطح‌های مختلف مدیریت، تعریف مسئولیت‌ها و اختیارات، و تعیین روابط بین اعضای سازمان باشد. در صورت عدم وجود یک ساختار سازمانی رسمی،

---

[1] Incident Command System (ICS)

اعضای سازمان ممکن است نتوانند به درستی ارتباط برقرار کنند و از تعاملات بهینه برای دستیابی به اهداف سازمان استفاده کنند.

- ناکافی بودن و نقصان در ساختار برنامه‌ریزی یکپارچه: یکی از اصلی‌ترین موانع مدیریت اصولی منابع در سازمان، نداشتنِ برنامه‌ریزی مناسب است. بدون برنامه‌ریزی دقیق، تخصیص منابع ممکن است تصادفی و بدون در نظر گرفتن اهداف و استراتژی‌های سازمان انجام شود.
- فقدان و خلاء دانش کافی و بدست آمده بر اساس تجربه: مدیریت دانش امکان دسترسی به تجربیات، دانش و تخصص را فراهم می‌کند که این عمل توانایی‌های جدیدی ایجاد می‌کند، عملکرد را بهبود می‌بخشد، نوآوری را افزایش می‌دهد، اطلاعات و سرمایه‌های دانش موجود در سازمان را به کار می‌گیرد، توزیع دانش و اطلاعات را در حوزه‌های مختلف سازمانی تسهیل می‌کند و اطلاعات و دانش را در فرایندهای روزانه‌ی کسب و کار ادغام می‌کند.
- مدیریت منابع ناکافی: مدیریت بهینه منابع به کاهش هدررفت منابع مالی، انسانی و مادی منجر می‌شود. با بهبود برنامه‌ریزی، تخصیص مناسب و کنترل دقیق منابع، از ضایعات و مصرف ناپسند منابع جلوگیری می‌شود.
- توانایی پیش‌بینی و برآورد محدود: اصولاً پیش‌بینی، عنصری کلیدی برای تصمیم‌گیری‌های مدیریتی است و به همین دلیل سیستم‌های مدیریتی برای طراحی و کنترل عملگرهای تشکیلاتی خود نیاز به پیش‌بینی دارند. به طور کلی می‌توان گفت که پیش‌بینی عبارت است از برآورد پیشامدهای آینده و هدف از پیش‌بینی، کاهش ریسک در تصمیم‌گیری است. پیش‌بینی‌ها معمولاً دارای مقداری خطا هستند که میزان این خطا با داشتن اطلاعات بیشتر در مورد سیستم کاهش می‌یابد.

این موارد با راه‌اندازی سامانه فرماندهی حادثه می‌تواند به شیوه‌ای صحیح و کارآمد مدیریت شده و سازمان را به سوی بهره‌وری و پیشرفت سوق دهد. با این حال ممکن است با راه‌اندازی سامانه فرماندهی حادثه به یک سری نقاط ضعف مانند موارد زیر نیز برخورد کنیم:

- خلاء و یا کمبود مسئولیت و پاسخگوئی شامل زنجیره‌ی فرماندهی[1] و نظارت نامفهوم
- ارتباطات ضعیف، شامل مشکلات سیستمی و واژه‌شناسی
- کمبود یک فرآیند برنامه‌ریزی سیستماتیک و منظم و سازمان‌یافته
- فقدان یک ساختار مدیریتی از پیش تعیین شده، انعطاف‌پذیر و مشترک

---

[1] Chain of Command

- عدم وجود روش‌های از پیش تعیین شده به منظور تلفیق نیازمندی‌های بین سازمانی در قالب ساختار مدیریتی و فرآیند برنامه‌ریزی

۳-۳ آتوماتای یادگیر

یکی از روش‌های یادگیری تقویتی، آتوماتای یادگیر[1] است. آتوماتای یادگیر بدون هیچگونه اطلاعاتی درباره اقدام بهینه (یعنی با در نظر گرفتن احتمال یکسان برای تمامی اقدام‌های خود در آغاز کار) سعی در یافتن پاسخ مساله دارد. یک آتوماتای یادگیر را می‌توان به صورت یک شئ مجرد[2] که دارای تعداد متناهی عمل است، در نظر گرفت. آتوماتای یادگیر با انتخاب یک عمل از مجموعه اقدام‌های خود و اِعمال آن بر محیط، عمل می‌کند. عمل مذکور توسط یک محیط تصادفی ارزیابی می‌شود و آتوماتا از پاسخ محیط برای انتخاب عمل بعدی خود استفاده می‌کند. در طی این فرایند آتوماتا یاد می‌گیرد که عمل بهینه را انتخاب نماید [۳۱].
محیط در اتوماتای یادگیر را می‌توان با سه‌تائی $E \equiv \{\alpha, \beta, c\}$ نمایش داد که در آن $\alpha \equiv \{\alpha_1, \alpha_2, .., \alpha_r\}$ مجموعه ورودی‌ها و $\beta = \{\beta_1, \beta_2, .., \beta_r\}$ مجموعه خروجی‌ها را نشان می‌دهد. $c = \{c_1, c_2, .., c_r\}$ نیز مجموعه احتمالات جریمه را نشان می‌دهد. این نوع از اتوماتا را اتوماتای یادگیر ثابت می‌نامند. اگر β دو عضوی باشد، $\beta_i = 1$ جریمه و $\beta_i = 0$ پاداش را نشان می‌دهد [۳۲] و در این صورت مدل آتوماتا P خواهد بود. در صورتی که β متغیر تصادفی در فاصله $[0,1]$ باشد، محیطو به عنوان مدل Q نامگذاری می‌شود.
علاوه بر اتوماتاهای با ساختار ثابت، اتوماتاهای با ساختار متغیر نیز وجود دارد که با چهارتائی $\{\alpha, \beta, p, T\}$ نشان داده می‌شود. $p = \{p_1, p_2, .., p_r\}$ بردار احتمال انتخاب هر یک از عمل‌ها و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری می‌باشد.

۳-۴ مسئله بهینه‌سازی محدودیت توزیع شده

در مسئله بهینه‌سازی محدودیت توزیع شده[3]، عامل‌ها مسئول کنترل متغیرها و محدودیت‌ها به منظور بهینه‌سازی یک تابع هدف کلی هستند [۳۳, ۳۴]. عامل‌ها برای تخصیص مقادیر به متغیرها همکاری می‌کنند. DCOP با چندتایی $< A, X, D, F, \sigma >$ تعریف می‌شود که در آن $X = \{x_1, x_2, .., x_N\}$ مجموعه‌ای از عوامل، $A = \{a_1, a_2, .., a_M\}$

---

[1] Learning Automata
[2] Object Abstract
[3] Distributed Constraint Optimization Problem

مجموعه‌ای از متغیرها و $D=\{D_1,D_2,..,D_N\}$ مجموعه‌ای از دامنه‌های محدود است که در آن D_i دامنه متغیر x_i است. $F=\{f_i,f_2,..,f_o\}$ مجموعه‌ای از توابع پاداش است که در آن هر تابع پاداش $f_i:D_{i_1}\times D_{i_2}\to\mathbb{N}\bigcup\{-\infty,0\}$ ، پاداش هر ترکیبی از مقادیر متغیرها را مشخص می‌کند. در محدوده تابع $\sigma(x)$ هر متغیر را به یک عامل نگاشت می‌کند و کنترل هر متغیر $x\in X$ را به یک عامل $\sigma(x)$ اختصاص می‌دهد. یک راه‌حل کامل یک تخصیص مقدار برای همه متغیرها است و هدف یافتن یک راه‌حل کامل با حداکثر پاداش است.

۴- روش پیشنهادی

در روش معرفی شده در این مقاله، ابتدا مسئله فرماندهی حادثه به صورت یک مسئله بهینه‌سازی محدودیت توزیع شده مدل می‌شود که در بخش ۴-۱ نشان داده شده است. شکل ۱ پویایی مسئله را نشان می‌دهد؛ همانطور که مشخص است، در هر گام زمانی یک مسئله DCOP خواهیم داشت که نتایج آن در هر مرحله، تابعی از تخصیص‌های مقدار در مرحله قبلی آن است و می‌توان این اتفاق را با استفاده از مدل‌های مارکوف مدل کرد (بخش ۴-۲). در این مدل‌ها در هر بازه‌ی زمانی، عامل‌ها مسئول تخصیص مقادیر به متغیرهایشان هستند که به عنوان عناصر تصمیم‌گیر می‌توانند دنباله‌ی مقادیر را یاد بگیرند و به یک سیاست مشترک سراسری همگرا شوند. در این حالت متغیرهای تصمیم فقط در مرحله اول مقداردهی می‌شوند؛

شکل ۱: یک اپیزود از مسئله فرماندهی حادثه پویا به صورت دنباله‌ای از مسائل بهینه‌سازی محدودیت توزیع شده در بازه‌های زمانی؛ τ_L حالت جذب، A عامل‌ها و T فعالیت‌ها

در این مقاله هر یک از گام‌های زمانی (τ)، یک حالت (state) می‌باشد که خروجی هریک از آنها بر اساس احتمال و انتخاب عامل‌ها مشخص می‌شود. شروع تا پایان این توالی یک اپیزود نامیده می‌شود و آخرین state از این توالی، حالت جذب نامیده می‌شود. این اپیزود به

صورت یک فرایند تصمیم‌گیری مارکوف[1] دیده می‌شود که ابزاری قدرتمند برای مدل کردن مسائل تصمیم‌گیری با تصمیمات متوالی و با وجود عدم قطعیت است.
برای حل فرایند تصمیم‌گیری مارکوف می‌توان برای هر عامل، از یک آتوماتای یادگیر بهره برد تا در هر state هریک از آتوماتاها با توجه به پاداش‌های دریافتی از محیط، فعالیت کنند. فعالیت هر یک از آتوماتاها با توجه به بردار احتمال تعریف شده در بخش ۳-۳ انجام می‌گیرد. جزئیات این موارد در بخش ۴-۳ به طور مبسوط توضیح داده شده است.

۴-۱ مدل‌سازی مسئله تخصیص در سیستم فرماندهی حادثه به صورت مسئله بهینه‌سازی محدودیت توزیع شده

برای مدل‌سازی مسئله تخصیص، عناصر اصلی DCOP باید به عنوان متغیرها، دامنه هر متغیر، محدودیت‌ها و توابع تعریف شوند. در این مسئله، "عامل‌ها" متغیرهای DCOP هستند که برای تصمیم‌گیری فعالیت‌ها تعیین می‌شوند. دامنه هر متغیر، "عمل هر عامل" است که به صورت "انتخاب یک فعالیت" یا " انتخاب هیچ فعالیت" تعریف می‌شود. جدول ۱ عناصر مسئله را به طور جداگانه نشان می‌دهد. دو محدودیت معرفی شده در بخش سوم، در جدول ۱ نمایش داده شده است.

جدول ۱: مدل‌سازی مسئله تخصیص به عنوان یک مسئله بهینه‌سازی محدودیت توزیع‌شده (DCOP)

عناصر اصلی DCOP	مسئله تخصیص منابع
متغیرها	عامل‌ها
دامنه هر متغیر	عمل هر یک از عامل‌ها: ("انتخاب یک فعالیت" یا " انتخاب هیچ فعالیت")
محدودیت‌ها	۱. هریک از فعالیت‌ها دقیقا به یک عامل تخصیص می‌یابد. ۲. هریک از تمامی‌عامل‌ها دقیقا به یک فعالیت مختص شوند.
هدف	$(S,\{\alpha_i\},Tr,R,\{\Omega_i\},O,\gamma)$

در این مقاله، ما مسئله را به عنوان یک DCOP در هر مرحله زمانی مدل‌سازی کردیم و با تکرار آن در گام‌های زمانی متوالی، یک DCOP پویا ایجاد شد که می‌توان با روشی بر پایه آتوماتای یادگیر آن را حل کنیم.

---

[1] Markov Decision Process (MDP)

۲-۴ مدل‌سازی مسئله تخصیص توزیع شده و پویا عنوان یک فرآیند تصمیم‌گیری مارکوف

با توجه به تعریف مسئله و رویکرد پیشنهادی، ما مسئله را به عنوان یک فرآیند تصمیم‌گیری مارکوف غیرمتمرکز جزئی قابل مشاهده (Dec-POMDP)[1] مدل‌سازی کردیم که تعمیم یک مسئله تصمیم‌گیری مارکوف (MDP)[2] [۳۵] برای در نظر گرفتن چندین عامل غیرمتمرکز است. Dec-POMDP یک چندتایی $(S,\{\alpha_i\},Tr,R,\{\Omega_i\},O,\gamma)$ است که در آن S مجموعه‌ای از حالات است. $\{\alpha_i\}$ مجموعه‌ای از اقدامات و Tr مجموعه‌ای از احتمالات انتقال شرطی بین حالت‌ها است. R تابع پاداش و Ω و O مجموعه‌ای از مشاهدات هستند. $\gamma \in [0,1]$ نرخ کاهش[3] است. از آنجایی که هر عامل دراین مسئله به طور کامل از وضعیت محلی خود آگاه است، مسئله Dec-MDP به صورت محلی کاملاً قابل مشاهده است و بنابراین، Ω و O را می‌توان حذف کرد [۳۶]. در ادامه هریک از قسمت‌های مسئله تصمیم‌گیری مارکوف نشان داده می‌شود:

فضای حالت

در فضای حالت هر عامل به طور مستقل عمل می‌کنند و از انتخاب دیگر عامل‌ها بی‌اطلاع است. در این حالت فضای حالت هر عامل به صورت زیر تعریف می‌شود:

$$S_i = \{V, SI, SJ\} \qquad such\,as \qquad v_i \in V = \{\rho_{ij}, V_j\} \tag{۲}$$

به طوری که ρ_{ij} احتمال قطعی عامل i برای انجام فعالیت j است، و V_j مقدار کار است (به عنوان مثال، مقدار کار).

حالت عامل α_i: (S_{α_i}) به صورت زیر تعریف می‌شود: اگر یک عامل در مرحله زمانی قبلی در حال انجام فعالیتی باشد، حالت آن "مشغول" خواهد بود که روی صفر تنظیم می‌شود، یعنی در مرحله زمانی فعلی درگیر یک فعالیت نیست. در مقابل اگر در مرحله زمانی قبلی هیچ فعالیتی را به خود تخصیص نداده باشد، در حالت «مشغول نبودن» خواهد بود و می‌تواند در مرحله زمانی فعلی فعالیت j را انجام دهد. در این حالت به حالت «مشغول» تبدیل می‌شود که با «j» نشان داده می‌شود.

$$s_{a_i} \in SI = \{0, j\} \tag{۳}$$

---

[1] Decentralized Partially Observable Markov Decision Process
[2] Markov Decision Process
[3] discount factor

حالت فعالیت t_j: (S_{t_j}) به صورت زیر تعریف می‌شود: اگر فعالیتی در حالت قبلی به هر عاملی تخصیص داده شده باشد، مقدار آن "صفر" خواهد بود و در غیراینصورت، i خواهد بود.

$$a_i = 1 - \sum_{j=1}^{N} x_{ij} \qquad i = 1,2\ldots,M \qquad j = 1,2\ldots,N \tag{۴}$$

فضای عمل

عملکرد هر عامل به صورت زیر تعریف می‌شود: "انتخاب یک فعالیت" یا "انتخاب هیچ یک از فعالیت‌ها"

احتمال انتقال

احتمال انتقال[1]، پویایی محیط را تعریف می‌کند. این مدل برای هر عاملی قطعی است و به انتساب انجام شده در حالت اول بستگی دارد:

$$p_q(\tau+1) = p_q(\tau) - d\left[\left(1-\beta_q(\tau)\right)p_q(\tau)\right] + b\beta_q(\tau)\left[\frac{1}{r-1} - p_q(\tau)\right] \qquad \forall q \;\; q \neq g \tag{۵}$$

به زبان ساده می‌توان گفت که عامل i در حالت دوم در دسترس خواهد بود اگر و تنها در صورتی که در حالت اول درگیر فعالیتی نشده باشد. این معادله "تکامل حالت عامل (α_i)" نامیده می‌شود.

مدل احتمال انتقال به طور تصادفی برای هر فعالیت تکمیل می‌شود. تکمیل تصادفی حالت فعالیت در هر مرحله زمانی به فعالیت‌های انجام شده در حالت قبلی بستگی دارد. احتمال اینکه $t_j = 1$، به این معنی است که فعالیت j در حالت اول انجام نشده است و بالعکس. توزیع متغیر تصادفی t_j به صورت زیر انجام می‌شود:

$$Pr\left[t_j = c\right] = c\prod_{i=1}^{M}\left(1 - pr_{ij}(\tau)\right)^{x_{ij}} + (1-c)\left(1 - \prod_{i=1}^{M}\left(1 - pr_{ij}(\tau)\right)^{x_{ij}}\right) \tag{۶}$$

$$\text{for} \quad c = 0,1 \qquad\qquad j = 1,2,\ldots,N$$

به طوری که $pr_{ij}(\tau)$ احتمالی است که عامل i، فعالیت j را در مرحله زمانی τ انجام می‌دهد و x_{ij} متغیر تصمیم مطابق رابطه (۱) است. احتمال اینکه $t_j = 1$ باشد به این معنی است که فعالیت j در حالت اول انجام نشده است. احتمال اینکه $t_j = 0$ باشد نشان دهنده

---

[1] Transition probability

این احتمال است که فعالیت j در حالت اول انجام شده است. توزیع متغیر تصادفی t_j در رابطه (۶) نمایش داده شده است.

تابع پاداش

فرآیند پاداش‌دهی با توجه به مدل‌های مختلف (مدل P و مدل S) مطابق بخش ۳ و روابط ۷ تا ۱۰ به دو فرآیند تقسیم می‌شود.

۴-۳ حل یک مسئله تخصیص توزیع شده پویا با استفاده از روش آتوماتای یادگیر با استفاده از الگوریتم DISC

در این بخش جزئیات حل یک مسئله تخصیص توزیع شده پویا با استفاده از روش آتوماتای یادگیر ارائه می‌شود:

در الگوریتم اصلی، در ابتدا مقداردهی اولیه متغیرها انجام می‌شود. سپس، برای هر فعالیت و بر اساس تعداد اقداماتی که یک عامل می‌تواند انجام دهد، یک احتمال اولیه ($1/r$) اختصاص داده می‌شود (r تعداد اقدامات است). عامل‌ها اقدامات خود را با توجه به مجموع احتمالات هر عمل موجود در مجموعه اقدامات، انتخاب می‌کنند، به طوری که اگر مقدار این مجموع کمتر از یک عدد تصادفی بود، همان عمل بر اساس شاخص بازگشتی اقدام انتخاب شده، توسط عامل انتخاب می‌شود. عمل انتخابی عامل، در محیط پویا حالت بعدی را تعیین می‌کند.

Algorithm 1. Distributed Incident Command System (DISC)

```
1.2  initialize Agents, Tasks, Actions, n=numOfActions, m = numOfTasks
1.3  initialize p=1/n for all Tasks
1.4  repeat for each iteration
1.5   for a_i in Agents
1.6    selectAct = SelectAction(n, m, Agents, Actions, Tasks, p_ij )
1.7    new_p = UpdateProbability (n, m, Agents, Actions, Tasks, p_ij )
1.8  end
1.7  end
```

در الگوریتم ۱ که الگوریتم اصلی ما می‌باشد، برای هر یک از کارها، بر اساس تعداد عمل‌هایی که عامل می‌تواند انجام دهد، یک مقدار احتمال ($1/r$) در نظر گرفته می‌شود (r تعداد فعالیت‌ها) (خط ۱.۲). در خط ۱.۶ عامل‌ها با توجه به تابع تعریف شده در الگوریتم ۲ عمل خود را انتخاب می‌کنند، به این صورت که برای هر یک از عمل‌های موجود در مجموعه عمل‌ها، مجموع احتمالات محاسبه می‌شود (خط ۲.۱ تا ۲.۳)، اگر مقدار این مجموع

کمتر از عدد تصادفی مدنظر باشد، همان عمل توسط عامل انتخاب می‌شود. (خطوط ۲.۴ تا ۲.۷).

Algorithm 2. SelectAction

Input: *n, m, Agents, Actions, Tasks, p*
Output: selectAct
2.1 Initialize *sumP=0*
2.2 for $i = 1 : n$
2.3 $sump \mathrel{+}= p(i)$
2.4 if ($rand <= sumP$)
2.5 selectAct = i
2.6 end
2.7 end

در خط ۱.۷ با استفاده از تابع تعریف شده در الگوریتم‌های ۳ تا ۶، مقدار احتمالات p را برای همه‌ی عمل‌ها آپدیت می‌کند؛ این کار با توجه به پاداشی که برای عامل‌ها بر اساس انتخابشان تعریف کرده‌ایم انجام می‌شود. یعنی اگر عامل‌ها در حالت جذب انتخاب‌های بهتری از تکرار قبل داشته باشند و نتیجه‌ی تخصیص کار با استفاده از رابطه ۱ بهتر از تکرار قبل باشد، پاداش دریافت کرده و در غیر اینصورت جریمه می‌شوند.

برای آپدیت احتمال (خط ۱.۷) با توجه به نوع مدل LA و زمان پاداش، می‌توان از طرق زیر عمل کرد:

۱- در صورتی که هر عامل بلافاصله پس از تعیین تکلیف پاداش خود را دریافت کرده و احتمالات خود را بر اساس مدل P به‌روز کند، بدین طریق عمل می‌شود: هر عامل پس از انتخاب یک عمل و با توجه به میزان فعالیت انجام شده، پاداشی از محیط دریافت می‌کند، یعنی اگر اثر جدید، بهتر از تکرار قبلی بود، پاداش $\beta = 0$ و در غیر این‌صورت $\beta = 1$ است. سپس، اگر پاسخ محیط مطلوب بود، احتمال اقدام انتخاب شده افزایش می‌یابد. در مقابل اگر واکنش نامطلوب محیط وجود داشته باشد، احتمالات بدون تغییر باقی می‌ماند، این موارد از طریق روابط (۷) و (۸) انجام می‌گیرد. (الگوریتم ۳)

Algorithm 3. UpdateProbability (P-model)

Input: *n, m, Agents, Actions, Tasks, p*
Output: *new_p*
3.1 *assignmentTask* = calculate Eq.1
3.2 if new *assignmentTask* better than previous *assignmentTask*
3.3 reward = 0
3.4 else
3.5 reward = 1
3.6 end
3.7 if reward == 0
3.8 *new_p* = calculate Eq.7 and Eq.8
3.9 else
3.10 *new_p = p*
3.11 end

۲- در صورتی که هر عامل در پایان هر اپیزود پاداش دریافت کرده و احتمالات خود را بر اساس مدل P به روز کند، بدین طریق عمل می‌شود: در این حالت با توجه به مدل P، پاداش بر اساس کل کار انجام شده توسط همه عوامل، داده می‌شود. در این حالت نیز برای آپدیت توابع احتمال از رابطه‌های (۷) و (۸) می‌توان بهره برد. (الگوریتم ۴)

$$p_g(\tau+1) = p_g(\tau) + d\left[1 - p_g(\tau)\right] \qquad (۷)$$
$$p_q(\tau+1) = (1-d)\,p_q(\tau) \qquad \forall q \quad q \neq g$$

$$p_g(\tau+1) = (1-b)\,p_g(\tau) \qquad (۸)$$
$$p_q(\tau+1) = b/_{r-1} + (1-b)\,p_q(\tau) \qquad \forall q \quad q \neq g$$

Algorithm 4. UpdateProbability (P-model Individually)

Input: *n, m, Agents, Actions, Tasks, p*
Output: *new_p*
3.1 for *i* in *Agents*
3.2 *effect*(i) = The amount of task done by the agent(i)
3.3 if new *effect(i)* better than previous
3.4 *Reward*(i) = 0
3.5 else
3.6 *Reward(i)* = 1
3.7 end
3.8 if *reward(i)* == 0
3.9 *new_p* = calculate Eq.7 and Eq.8
3.10 else
3.11 *new_p = p*
3.12 end
3.13 end

۳- در صورتی که هر عامل بلافاصله پس از تعیین تکلیف پاداش خود را دریافت کرده و احتمالات خود را بر اساس مدل S به روز کند، بدین طریق عمل می‌شود: هر عامل پس از انتخاب یک عمل و با توجه به میزان فعالیت انجام شده، پاداشی از محیط دریافت می‌کند. در مدل S، پاداش در محدوده $[0,1]$ داده می‌شود. در این حالت از رابطه‌های (۹) و (۱۰) برای به‌روزرسانی توزیع احتمالات عمل خود استفاده می‌کنند. (الگوریتم ۵)

Algorithm 5. UpdateProbability (S-model)

Input: *n, m, Agents, Actions, Tasks, p*
Output: *new_p*

```
3.1  assignmentTask = calculate Eq.1
3.2  if new assignmentTask better than previous assignmentTask
3.3     reward = 0
3.4  else
3.5     reward = 1
3.6  end
3.7  if reward == 0
3.8     new_p = calculate Eq.9 and Eq.10
3.9  else
3.10    new_p = p
3.11 end
```

۴- در صورتی که هر عامل در پایان هر اپیزود پاداش دریافت کرده و احتمالات خود را بر اساس مدل S به روز کند، بدین طریق عمل می‌شود: در این حالت با توجه به مدل S، پاداش بر اساس کل کار انجام شده توسط عوامل داده شد. در این حالت نیز از روابط (۹) و (۱۰) برای آپدیت مقادیر احتمالات خود استفاده می‌کند. (الگوریتم ۶)

$$p_g(\tau+1) = p_g(\tau) + d\left[\left(1-\beta_g(\tau)\right)\left(1-p_g(\tau)\right)\right] - b\beta(\tau)p_g(\tau) \qquad (۹)$$

$$p_q(\tau+1) = p_q(\tau) - d\left[\left(1-\beta_q(\tau)\right)p_q(\tau)\right] + b\beta_q(\tau)\left[\frac{1}{r-1} - p_q(\tau)\right] \quad \forall q \;\; q \neq g \qquad (۱۰)$$

Algorithm 6. UpdateProbability (S-model Individually)

Input: *n, m, Agents, Actions, Tasks, p*
Output: *new_p*

```
3.1  for i in Agents
3.2   effect(i) = The amount of task done by the agent(i)
3.3   if new effect(i) better than previous
3.4      Reward(i) = 0
3.5   else
3.6      Reward(i) = 1
3.7   end
3.8   if reward(i) == 0
```

```
3.9          new_p = calculate Eq.9 and Eq.10
3.10  else
3.11         new_p = p
3.12  end
3.13 end
```

در انتهای الگوریتم، هنگامی‌که هیچ گزینه‌ای برای هیچ عاملی باقی نماند، این حالت یک "حالت جاذب" خواهد بود و هر عامل بر اساس توانایی خود و وظایف محول شده اثر خود را محاسبه می‌کند. این اثر با توجه به میزان توانایی هر عامل برای انجام هر فعالیت اندازه‌گیری می‌شود.

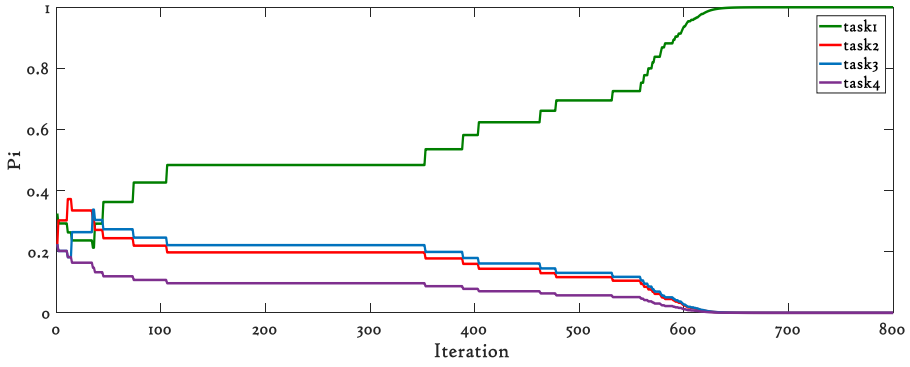
۵- شبیه‌سازی

در شبیه‌سازی فوق، m عامل و n فعالیت در نظر گرفته شده است به طوری که برای انجام هر یک از این فعالیت‌ها نیاز به نیرو و توان مشخصی توسط هر عامل می‌باشد. هر عامل به صورت غیرمتمرکز عمل می‌کند و تنها می‌تواند فعالیت‌هایی که در حوزه‌ی شناسایی و در توان خود باشد را به خود تخصیص داده و انجام دهد. هر عامل می‌داند به طور کلی چه میزان از فعالیت‌ها در محیط باقی‌مانده است و نتیجه‌ی تخصیص و کار خود را از محیط دریافت می‌کند. فرض بر این است که هر عامل فقط می‌تواند یک فعالیت را به خود اختصاص دهد. هر فعالیت دارای یک ارزش است و بنا به این ارزش، تخصیص فعالیت با ارزش بیشتر، اهمیت بالاتری دارد.

هر عامل به صورت مجزا به دنبال کاهش تعداد فعالیت‌های باقی‌مانده در محیط می‌باشد و هر بار بعد از تخصیص، از میزان فعالیت باقی‌مانده مطلع می‌گردد. بنابراین با استفاده از الگوریتم‌های یادگیری و بعد از هربار دریافت پاداش از محیط، به دنبال تخصیص فعالیتی می‌گردد که ارزش بیشتری داشته باشد.

۵-۱ نتایج

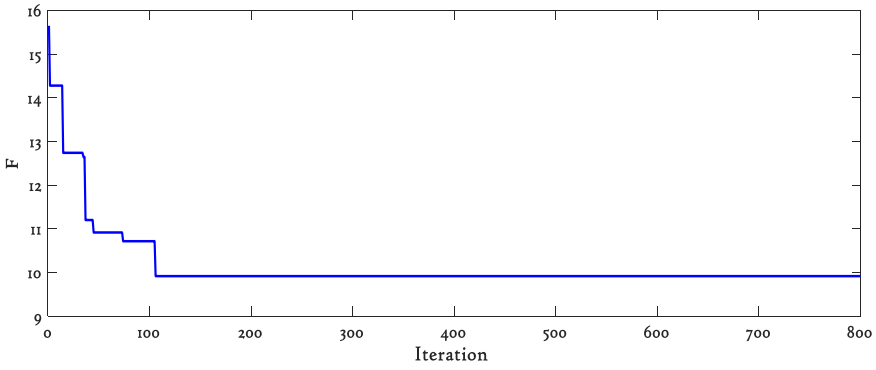
در **فاز اول شبیه‌سازی**، مسئله را با ۸ عامل و ۳ فعالیت پیاده‌سازی کرده‌ایم. مسئله کاملا به صورت توزیع‌شده و پویا شبیه‌سازی شده است. هر عامل در هر بازه زمانی می‌تواند از دامنه عمل‌های خود یک عمل انتخاب کند. مقدار احتمال برای هر یک از انتخاب‌ها با توجه به تعریف اتوماتای یادگیر برابر هم و به اندازه $p = 1/4$ در نظر گرفته شده است. بعد از انجام پیاده‌سازی و در طول تکرارهای برنامه (iteration)، با توجه به الگوریتم یادگیری و نحوه‌ی آپدیت کردن، احتمال (p_i) هر یک از فعالیت‌ها تغییر کرده و سرانجام فعالیتی که احتمال آن (p_i) بعد از تمام شدن تکرارها به سمت عدد "یک" میل کند، مورد انتخاب عامل خواهد بود. این تغییرات مربوط به هر یک از فعالیت‌ها، مرتبط به پاداشی است که با انتخاب عامل و اثری که بر روی رابطه‌ی ۱ می‌گذارد، می‌باشد. شکل ۲ همگرایی مسئله بعد از تکرارها را برای یکی از عامل‌ها به خوبی نشان می‌دهد.



شکل ۲: مقدار احتمال انتخاب فعالیت‌ها توسط یکی از عامل‌ها در روند یادگیری (۸ عامل و ۳ فعالیت)- p_i : بردار احتمال هر یک از فعالیت‌ها در الگوریتم آتوماتای یادگیر، iteration: تعداد تکرار الگوریتم

در پیاده‌سازی فوق با وجود ۸ عامل و ۳ فعالیت، برای هر یک از عامل‌ها ۴ انتخاب وجود دارد: انتخاب کردن "فعالیت ۱-task1"، انتخاب کردن "فعالیت ۲-task2"، انتخاب کردن "فعالیت ۳-task3" و انتخاب کردن"هیچ یک از فعالیت‌ها-task4". همانطور که از شکل مشخص است بعد از اتمام اجرای برنامه، "فعالیت ۱" که به رنگ سبز مشخص است دارای احتمال بالاتری است و بنابراین در انتهای یادگیری، انتخاب عامل خواهد بود. این فعالیت (فعالیت ۱) همان فعالیتی است که اگر عامل آن را نسبت به سایر فعالیت‌ها

انتخاب کند، در نتیجه‌ی سراسری اثر بیشتری خواهد داشت. این موارد برای سایر عامل‌ها نیز وجود دارد.



شکل ۳: مقدار رابطه ۱ بعد از روند یادگیری (۸ عامل و ۳ فعالیت) –
F: مقدار رابطه ۱، iteration: تعداد تکرار الگوریتم

شکل ۳ نیز مقدار رابطه ۱ (تابع F)، برای عامل را بعد از تکرارهای برنامه (iteration) نشان می‌دهد که به سمت مینیمم حرکت می‌کند. کاهش مقدار رابطه ۱ نشان می‌دهد که عامل‌ها توانسته‌اند فعالیت بیشتری در مجموع انجام دهند یا نه. از نمودار به خوبی مشخص است که پاداش‌ها به گونه‌ای تعریف شده است که یادگیری به درستی انجام شده و عامل‌ها به صورت توزیع‌شده و مجزا از هم، بدون اطلاع از انتخاب یکدیگر و به جهت مینیمم کردن تابع هدف، به سمت انتخاب درست حرکت می‌کنند.

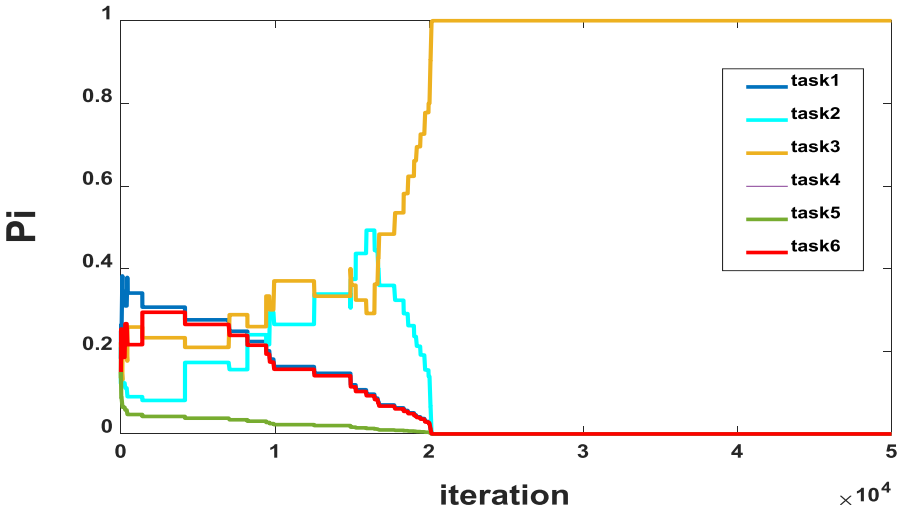
در این فاز، روش معرفی شده در این مقاله را با روش MA-DDCOP در مقاله [۷] از لحاظ تعداد انتخاب درست عامل‌ها و میزان رابطه ۱ مقایسه کرده‌ایم. هر اپیزود در شبیه‌سازی فوق، با وجود حالت‌های مختلف محیط و حضور ۸ عامل و ۴ انتخاب برای هر یک از آنها، در ۳۰ تکرار انجام شده است. در این تکرارها هر عامل با توجه به پاداش محیط، انتخاب خود را به‌روز نموده و توانسته است در انتهای این تکرار به سمت بهینه و انتخاب صحیح حرکت کند. می‌توان نتایج رابطه ۱ را در ده اجرای آخر هر دو الگوریتم و بعد از تکرارهای مختلف با توجه به تعداد درست انتخاب عامل‌ها در جدول ۲ مشاهده نمود.

جدول ۲ : مقدار رابطه ۱ در ده اجرای آخر دو الگوریتم DICS و MA-DDCOP با توجه به تعداد انتخاب درست عامل‌ها (۸ عامل و ٤ انتخاب)

تعداد تکرار	تعداد انتخاب درست عامل‌ها از ٤ انتخاب (روش MA-DDCOP)	مقدار رابطه ۱ (روش MA-DDCOP)	تعداد انتخاب درست عامل‌ها از ٤ انتخاب (روش معرفی شده در این مقاله(DICS))	مقدار رابطه ۱ (روش معرفی شده در این مقاله(DICS))
تکرار اول	۱	۰/۸	۱	۰/۶۷
تکرار دوم	۱	۰/۷۴	۱	۰/۵۵
تکرار سوم	۱	۰/۶۱	۲	۰/۵۶
تکرار چهارم	۲	۰/۶۳	۲	۰/۳۱
تکرار پنجم	۲	۰/۷۱۳	۲	۰/۴۰
تکرار ششم	۲	۰/۶۵	۲	۰/۲۷
تکرار هفتم	۲	۰/۴۳	۲	۰/۲۲
تکرار هشتم	۲	۰/۵۱	۳	۰/۱۹۵
تکرار نهم	۳	۰/۴۴	۳	۰/۱۸
تکرار دهم	۳	۰/۳	۳	۰/۱۰۳

جدول ۲ نشان می‌دهد که در الگوریتم معرفی شده در این مقاله، در تکرارهای کم، برخی عامل‌ها انتخاب‌های درستی نداشته و بنابراین مقدار رابطه ۱ نیز کمتر کاهش یافته است، اما در تکرارهای آخر مثل تکرار نهم و تکرار دهم، با توجه به افزایش یادگیری عامل‌ها و انتخاب درست آنها، میزان رابطه ۱ مینیمم شده و نسبت به الگوریتم MA-DDCOP، حدود ۸۹ درصد فعالیت‌ها انجام شده است.

در **فاز دوم شبیه‌سازی**، مسئله با تعداد عامل و فعالیت بیشتر انجام شده است. تعداد عامل‌ها ۲۰ و تعداد فعالیت‌ها ۵ می‌باشد. این بار برای هر یک از عامل‌ها ۶ انتخاب وجود خواهد داشت: انتخاب کردن "فعالیت ۱-task1"، انتخاب کردن "فعالیت ۲-task2"، انتخاب کردن "فعالیت ۳-task3" ، انتخاب کردن "فعالیت ۴-task4"، انتخاب کردن "فعالیت ۵-task5"و انتخاب کردن "هیچ یک از فعالیت‌ها-task6". بنابراین احتمال به صورت $p = 1/6$ تعریف می‌شود. نتایج تغییرات p_i در شکل ۴ برای یکی از عامل‌ها نشان داده شده است. با توجه به این شکل مشخص است که بعد از اتمام روند یادگیری (iteration)، عامل فعالیت ۳ را انتخاب می‌کند، چرا که احتمال (p_i) فعالیت ۳ به سمت ۱ میل می‌کند.

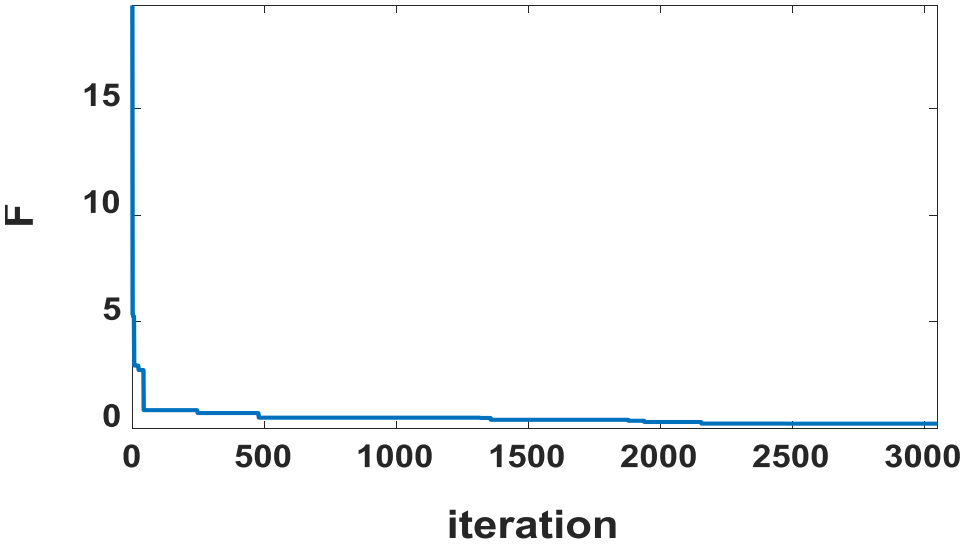


شکل ٤: مقدار احتمال انتخاب فعالیت‌ها توسط یکی از عامل‌ها در روند یادگیری (۲۰ عامل و ۵ فعالیت) - p_i : بردار احتمال هر یک از فعالیت‌ها در الگوریتم آتوماتای یادگیر، iteration: تعداد تکرار الگوریتم

در شکل ۵، مقدار رابطه ۱ (F) بعد از اجرای الگوریتم (iteration) را نشان می‌دهد که استفاده از آتوماتای یادگیر برای تعداد عامل‌های زیاد توانسته نتیجه بهینه‌ای را برای تخصیص‌های مناسب فعالیت به دست بیاورد. چرا که روند یادگیری عامل‌ها به درستی انجام شده و همین امر نتیجه کل را به سوی مینیمم شدن هدایت کرده است.

همچون فاز اول، دو الگوریتم DICS –معرفی شده در این مقاله– را با الگوریتم MA-DDCOP [۷] از لحاظ انتخاب درست عامل‌ها و انجام فعالیت‌ها، مقایسه کرده‌ایم. شبیه‌سازی با حضور ۲۰ عامل و ۶ انتخاب برای هریک از آنها، در ۳۰ تکرار انجام شده و مقدار رابطه ۱ در ده اجرای آخر در جدول ۳ نشان داده شده است.

از جدول ۳ مشخص است که در تکرارهای ابتدایی الگوریتم، به دلیل خطای عامل‌ها در انتخاب درست، میزان رابطه (۱) به میزان کافی کاهش نیافته است، اما در تکرارهای پایانی به دلیل یادگیری درست عامل‌ها، تعداد انتخاب درست عامل‌ها افزایش یافته (حدود ۵ انتخاب درست)، بنابراین در الگوریتم معرفی شده در این مقاله، تعداد فعالیت بیشتری انجام گرفته و میزان رابطه (۱) در حد ۸۷ درصد بهتر از الگوریتم قبل بوده است.



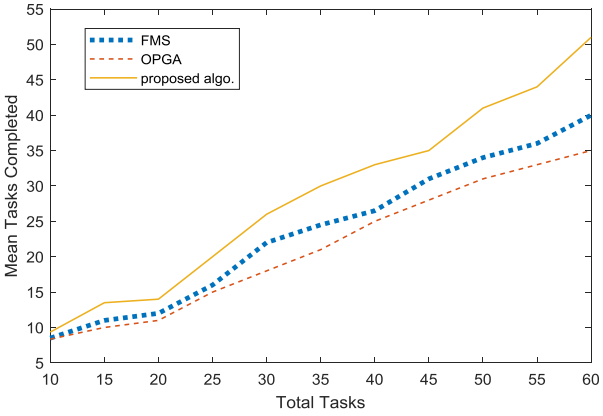
شکل ۵: مقدار رابطه ۱ بعد از روند یادگیری (۲۰ عامل و ۵ فعالیت) –
F: مقدار رابطه ۱، iteration: تعداد تکرار الگوریتم

جدول ۳ : مقدار رابطه (۱) در ده اجرای آخر دو الگوریتم DICS و MA-DDCOP با توجه به تعداد انتخاب درست عامل‌ها (۲۰ عامل و ۶ انتخاب)

تعداد تکرار	تعداد انتخاب درست عامل‌ها از ٦ انتخاب (روش MA-DDCOP)	مقدار رابطه ۱ (روش MA-DDCOP)	تعداد انتخاب درست عامل‌ها از ٦ انتخاب (روش معرفی شده در این مقاله(DICS))	مقدار رابطه ۱ (روش معرفی شده در این مقاله(DICS))
تکرار اول	۲	۱/۰۱	۳	۰/۷۱
تکرار دوم	۳	۰/۹۳	۳	۰/۵۹
تکرار سوم	۳	۰/۹۵	۳	۰/۶
تکرار چهارم	۳	۰/۷۶	۴	۰/۴۸
تکرار پنجم	۳	۰/۸	۴	۰/۳۳
تکرار ششم	۴	۰/۵۹	۴	۰/۳۵
تکرار هفتم	۴	۰/۶۷	۴	۰/۲۹
تکرار هشتم	۴	۰/۵۴	۵	۰/۲
تکرار نهم	۵	۰/۴	۵	۰/۱۸
تکرار دهم	۵	۰/۳۹۵	۵	۰/۱۱

در **فاز سوم شبیه‌سازی** روش خود را با دو روش F-Max-Sum و OPGA از لحاظ "درصد میانگین تعداد فعالیت انجام شده" نسبت به "کل فعالیت‌های موجود" مقایسه می‌کنیم [۲۷].

در این شبیه‌سازی، مجموعه‌ای از عامل‌ها را طوری انتخاب می‌کنیم که $|A|=10$ و مجموعه فعالیت‌ها را به صورت $|V|=\{10,15,..,60\}$ در نظر می‌گیریم. پنجاه نمونه از موقعیت‌های عامل و فعالیت به طور تصادفی برای هر مجموعه از فعالیت‌ها ایجاد می‌شود. علاوه بر این، مهلت هر فعالیت از یک توزیع یکنواخت که به تعداد فعالیت‌ها به صورت $d_v = U(0,10\times|V|)$ وابسته است و حجم فعالیت از یک توزیع یکنواخت که به تعداد $w_v = U(0,\frac{10\times|V|}{2})$ نیز وابسته است، ترسیم می‌شود. یعنی توزیع عادلانه فعالیت‌های آسان (مهلت طولانی حجم فعالیت کوچک) و سخت (مهلت کوتاه و حجم فعالیت زیاد) وجود دارد.



شکل ٦: مقایسه روش ارائه شده در مقاله با روش F-Max-Sum و OPGA از لحاظ میزان فعالیت انجام شده

همه استراتژی‌ها بر روی ۵۰ نمونه ارزیابی شدند و بیش از $10\times|V|$ مرحله زمانی (یعنی در هر مرحله زمانی، هر استراتژی برای تخصیص عوامل موجود به وظایف استفاده می‌شود). میانگین تعداد کل فعالیت تکمیل شده توسط هر استراتژی در شکل ۶ به همراه ۹۵٪ فواصل اطمینان نشان داده شده است. همانطور که از شکل مشاهده می‌شود، الگوریتم معرفی شده در این مقاله از دو الگوریتم F-Max-Sum و OPGA بهتر عمل می‌کند و نشان می‌دهد که این روند با افزایش تعداد فعالیت‌ها ادامه دارد. این موضوع به این دلیل است که OPGA

فقط به دنبال یافتن حداکثرهای محلی است، F-Max-Sum و الگوریتم معرفی شده راه‌حل بهینه را از همه راه‌حل‌های در نظر گرفته شده پیدا می‌کند، با این تفاوت که F-Max-Sum تعداد پیام زیادی مبادله می‌کند.

در الگوریتم‌های فوق تصمیم‌گیری‌ها در بازه‌های مختلف انجام می‌گیرد و پویایی محیط را در نظر می‌گیرند. از آنجا که هر یک از الگوریتم‌ها به دنبال کاهش تعداد فعالیت‌های باقی‌مانده در محیط هستند، از جدول ۴ مشخص است که روش معرفی شده در این مقاله بعد از اجرا، توانسته است میانگین فعالیت انجام شده‌ی بیشتری را نسبت به سایر الگوریتم‌ها انجام دهد. این مورد حتی با افزایش تعداد فعالیت‌ها نیز به قوت خود باقیست و همانطور که از جدول ۴ مشخص است مسئله‌ی ما می‌تواند به طور میانگین نزدیک ۸۵ درصد از فعالیت‌ها را به اتمام برساند.

جدول ٤: درصد میانگین تعداد فعالیت انجام شده توسط روش‌های OPGA، F-Max-Sum و روش معرفی شده در این مقاله نسبت به تعداد کل فعالیت‌ها

تعداد فعالیت ها	درصد میانگین تعداد فعالیت انجام شده (روش OPGA)	درصد میانگین تعداد فعالیت انجام شده (روش F-Max-Sum)	درصد میانگین تعداد فعالیت انجام شده (روش معرفی شده در مقاله)
۱۰	۰/۷۷	۰/۸	۰/۹
۱۵	۰/۶۶	۰/۷۳	۰/۹
۲۰	۰/۵۵	۰/۶۵	۰/۷
۲۵	۰/۵۸	۰/۶	۰/۸۴
۳۰	۰/۵۶	۰/۷۳	۰/۸۶
۳۵	۰/۵۷	۰/۶۷	۰/۸۲
٤۰	۰/۵۷	۰/۶۳	۰/۸۳
٤۵	۰/۶۲	۰/۶۸	۰/۷۵
۵۰	۰/۶۴	۰/۶۶	۰/۸۵
۵۵	۰/۶	۰/۶۳	۰/۸۳
٦۰	۰/۵۸	۰/۶۲	۰/۸۵

فاز چهارم شبیه‌سازی در رابطه با همگرایی و زمان اجرای الگوریتم برای تعداد عوامل مختلف می‌باشد. جهت این امر، نتایج را از نظر زمان اجرا و مقیاس‌بندی با یک روش پایه و متمرکز که از هیچ روش یادگیری استفاده نمی‌کند و الگوریتم F-Max-Sum و الگوریتم MA-DDCOP [۷] –که با استفاده از تکنیک‌های یادگیری تقویتی راه‌حل جایگزین برای حل‌کننده‌های DCOP معمولی استفاده می‌کند– مقایسه می‌کنیم.

جدول ۵: مقایسه روش ارائه شده با یک روش متمرکز، روش F-Max-Sum و روش MA-DDCOP از لحاظ سرعت اجرا

تعداد عامل‌ها	روش متمرکز	روش F-Max-Sum	روش MA-DDCOP	روش معرفی شده در مقاله
	(time)	(time)	(time)	(time)
۲	۲۲۳	۱۹۷	۴۷	۰.۸
٤	۴۸۹	۲۵۵	۹۵	۲.۱
٦	۵۵۴۷	۳۸۲	۱۴۵	۴.۱
۸	–	۷۲۰	۲۱۲	۶.۵
۱۲	–	۴۸۱۲	۴۳۱	۶۸
۱٦	–	۲۶۷۰۰۰	۶۵۹	۱۳۶

جدول ۵ زمان اجرای الگوریتم‌ها را با توجه به تغییرات در تعداد عامل‌ها و در نتیجه تعداد متغیرهای تصمیم‌گیری نشان می‌دهد. همانطور که در این جدول مشاهده می‌شود، تعداد عامل‌ها، زمان اجرای الگوریتم متمرکز را بسیار افزایش می‌دهد. روش F-Max-Sum برای مسائل با تعداد زیادی عامل اما افق کوچک مناسب است. روش MA-DDCOP هم با اینکه از تکنیک یادگیری بهره می‌برد اما نیازمند استفاده از یک هیوریستیک مناسب برای انجام کار می‌باشد. با این حال، ستون آخر جدول نشان می‌دهد که الگوریتم پیشنهادی هم برای تعداد زیادی عامل و هم برای یک افق بزرگ به دلیل استفاده از یادگیری خودکار و حذف روش‌های مبتنی بر جستجو و رویکرد اکتشافی مناسب است.

با توجه به اینکه سامانه فرماندهی حادثه جزو مواردی می‌باشد که سرعت پاسخگویی به فعالیت‌ها و تکمیل فعالیت‌ها توسط هر یک از عامل‌ها از اهمیت بالایی برخوردار است، نتایج شبیه‌سازی‌ها ما را امیدوار می‌سازد که می‌توان با استفاده از تکنیک‌های یادگیری، فعالیت‌های سامانه فرماندهی حادثه را بهتر و کارآمدتر نمود. ضمن این که در این سامانه غیرمتمرکز عمل نمودن عامل‌ها می‌تواند یکی از نکات برجسته و مهم باشد.

۶- نتیجه‌گیری

در این مقاله یک روش پویا و توزیع شده برای مدل‌سازی سیستم فرماندهی حادثه ارائه شده است. در این روش ابتدا مسئله به صورت یک مسئله بهینه‌سازی محدودیت توزیع شده مدل شده و سپس با استفاده از روش تصمیم‌گیری مارکوف و مفاهیم آتوماتای یادگیر به حل این مسئله پرداخته شده است. روش ارائه شده روشی غیرمتمرکز و توزیع شده است و می‌تواند

پویایی مسئله را پوشش دهد.
نتایج پیاده‌سازی‌ها و شبیه‌سازی‌ها نشان می‌دهد که روش معرفی شده در مقاله، از لحاظ همگرایی و توان انجام فعالیت‌ها از کیفیت بالایی برخوردار بوده و همچنین با افزایش مقیاس مسئله، توانایی پاسخ بلادرنگ را داراست. این موارد از جمله مواردی است که در سامانه فرماندهی حادثه از اهمیت بالایی برخوردار است.

۷–تعارض منافع

نویسندگان اعلام می‌دارند که در مورد انتشار این مقاله تضاد منافع وجود ندارد. علاوه بر این، موضوعات اخلاقی شامل سرقت ادبی، رضایت آگاهانه، سوء رفتار، جعل داده‌ها، انتشار و ارسال مجدد و مکرر توسط نویسندگان رعایت شده است.

۸–دسترسی آزاد

منابع

[۱] ا. کلات‌پور, "سیستم فرماندهی حادثه," انتشارات حک, ۱۳۹۹.
[۲] م. رضایی , ن. شمس کیا, "مدل تازه طراحی ساختار توسعه پایدار و مدیریت ایمنی در سامانه فرماندهی حوادث مدارس مطالعه موردی مدارس شهرستان شهریار," مدیریت بحران, vol. 10, no. 1, pp. 101-112, 2021.
[۳] ش. علمداری, "الگوها و دیدگاه‌ها در مدیریت بحران," بوستان حمید, ۱۳۹۶.
[۴] ف. رجبلو, "مدیریت بحران و سامانه فرماندهی حادثه," در سومین کنگره بین المللی بهداشت، درمان و مدیریت بحران در حوادث غیرمترقبه, ۱۳۸۵ doi/14746/https://civilica.com
[۵] W. L. Teacy *et al.*, "Decentralized Bayesian reinforcement learning for online agent collaboration," in *11th International Conference on Autonomous Agents and Multiagent Systems*, 2012: International Foundation for Autonomous Agents and Multiagent Systems.

[٦] D. T. Nguyen, W. Yeoh, H. C. Lau, S. Zilberstein, and C. Zhang, "Decentralized Multi-Agent Reinforcement Learning in Average-Reward Dynamic DCOPs," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014/06/21.٢٠١٤

[٧] M. Shokoohi, M. Afsharchi, and H. Shah-Hoseini, "Dynamic distributed constraint optimization using multi-agent reinforcement learning," *Soft Computing*, vol. 26, no. 8, pp. 3601-3629, 2022/03/16 2022.

[٨] K. Tuyls, "Learning in Multi-agent Systems. An Evolutionary Game Theoretic Approach," 2004.

[٩] M. A. L. Thathachar and P. S. Sastry, "Networks of Learning Automata," ed: Springer US, 2004.

[١٠] M. D. Pendrith, *An analysis of non-Markov automata games [electronic resource] : implications for reinforcement learning / Mark D. Pendrith and Michael J. McGarity* (PANDORA electronic collection, no. Accessed from https://nla.gov.au/nla.cat-vn3524867). [Sydney]: University of New South Wales, School of Computer Science and Engineering, 1997.

[١١] P. Vrancx, K. Verbeeck, and A. Nowé, "Networks of learning automata and limiting games," in *European Symposium on Adaptive Agents and Multi-Agent Systems*, 2005, pp. 224-238: Springer.

[١٢] M. K. Lindell, R. W. Perry, and C. S. Prater, "Organizing response to disasters with the incident command system/incident management system (ICS/IMS)," in *International workshop on emergency response and rescue*, 2005, vol. 31: Taipei.

[١٣] F. E. M. Agency, *National incident management system*. FEMA, 2017.

[١٤] C. W. Bahme and W. Kramer, *Fire Officer's Guide to Disaster Control*. National Fire Protection Association, 1978.

[١٥] A. V. Brunacini, "Fire Command, 1985 Quincy MA," *National Fire Protection Association*.

[١٦] A. V. Brunacini, *Fire command*. Jones & Bartlett Learning, 200.٢

[١٧] ب. ی. م. میرطاهری, "مقدمه ای بر سامانه فرماندهی حادثه در مدیریت بحران," طلیعه سبز, ١٣٨٨.

[۱۸] J. Jensen and S. Thompson, "The incident command system: a literature review," *Disasters*, vol. 40, no. 1, pp. 158-182, 2016.

[۱۹] A. Visser, N. Ito, and A. Kleiner, "Robocup rescue simulation innovation strategy," in *RoboCup 2014: Robot World Cup XVIII 18*, 2015, pp. 661-672: Springer.

[۲۰] T. L. Basegio and R. H. Bordini, "Allocating structured tasks in heterogeneous agent teams," *Computational Intelligence*, vol. 35, no. 1, pp. 124-155, 2019.

[۲۱] F. Wu and S. D. Ramchurn, "Monte-Carlo tree search for scalable coalition formation," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*, 2020, pp. 407-413.

[۲۲] T. Krausburg, R. H. Bordini, and J. Dix, "Modelling a chain of command in the incident command system using sequential characteristic function games," *Annals of Mathematics and Artificial Intelligence*, vol. 92, no. 4, pp. 925-951, 2024.

[۲۳] T. Rahwan, T. P. Michalak, M. Wooldridge, and N. R. Jennings, "Coalition structure generation: A survey," *Artificial Intelligence*, vol. 229, pp. 139-174, 2015.

[۲٤] R. L. Irwin, "The incident command system (ICS)," *Disaster response: Principles of preparation and coordination*, vol. 1, p. 30, 1989.

[۲٥] E. Ayari, S. Hadouaj, and K. Ghedira, "A dynamic decentralised coalition formation approach for task allocation under tasks priority constraints," in *2017 18th International Conference on Advanced Robotics (ICAR)*, 2017, pp. 25-۰ :۲٥٥IEEE.

[۲٦] C. Mouradian, J. Sahoo, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A coalition formation algorithm for multi-robot task allocation in large-scale natural disasters," in *2017 13th international wireless communications and mobile computing conference (IWCMC)*, 2017, pp. 1909-1914: IEEE.

[۲۷] S. D. Ramchurn, A. Farinelli, K. S. Macarthur, and N. R. Jennings, "Decentralized Coordination in RoboCup Rescue," *The Computer Journal*, vol. 53, no. 9, pp. 1447-1461, 2010/03/24 2010.

[۲۸] S. D. Ramchurn *et al.*, "Human–agent collaboration for disaster response," *Autonomous Agents and Multi-Agent Systems*, vol. 30, pp. 82-111, 2016.

[۲۹] ع. مجتبی , ع. کوروش, "یک الگوریتم فرابتکاری برای حل مساله تخصیص تصادفی," ۲۰۰۹.

[۳۰] R. M. Nauss, "Solving the Generalized Assignment Problem: An Optimizing and Heuristic Approach," *INFORMS Journal on Computing*, vol. 15, no. 3, pp. 249-266, 2003/08 2003.

[۳۱] K. S. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*. Dover Publications, 2013.

[۳۲] ح. س. سیدمیثم , م. محمدرضا, "تشخیص لبه در تصاویر با استفاده از اتوماتای یادگیر سلولی."

[۳۳] A. Petcu and B. Faltings, "A Scalable Method for Multiagent Constraint Optimization," *Ijcai'05*, pp. 266–271, 2005.

[۳٤] P. J. Modi, W.-M. Shen, M. Tambe, and M .Yokoo, "Adopt: asynchronous distributed constraint optimization with quality guarantees," *Artificial Intelligence*, vol. 161, no. 1-2, pp. 149-180, 2005/01 2005.

[۳٥] R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman, "Transition-independent decentralized markov decision processes," presented at the Proceedings of the second international joint conference on Autonomous agents and multiagent systems - AAMAS '03, 2003. Available: http://dx.doi.org/10.1145/860575.860583

[۳٦] R. Becker, S. Zilberstein ,V. Lesser, and C. V. Goldman, "Solving Transition Independent Decentralized Markov Decision Processes," *Journal of Artificial Intelligence Research*, vol. 22, pp. 423-455, 2004/12/01 2004.