

بهبود عملکرد عامل ها در شبیه ساز امداد با استفاده از کولونی مورچه ها و اتوماتای یادگیر

مصطفی اصغری	بهروز معصومی	محمدرضا میبدی
دانشگاه آزاد اسلامی	دانشگاه آزاد اسلامی علوم و تحقیقات تهران	دانشکده مهندسی کامپیوتر و فناوری اطلاعات
واحد میاندوآب	دانشگاه آزاد اسلامی قزوین	دانشگاه صنعتی امیرکبیر، تهران، ایران
m_asghary86@yahoo.com	bmasoumi@Qazviniau.ir	mmeiybodi@aut.ac.ir

چکیده: یکی از بسترهای مناسب برای تست و ارزیابی ایده ها و تکنیک های مربوط به سیستم های چند عامله، سیستم شبیه ساز امداد می باشد. در سیستم شبیه ساز امداد، وقوع یک حادثه مخرب شبیه سازی شده و عامل های نرم افزاری برای مقابله با اثرات حادثه در محیط مصنوعی فعالیت می کنند. اکثر الگوریتم های ارائه شده برای طراحی مسیر حرکت عامل ها در سیستم شبیه ساز امداد، بر اساس الگوریتم های کوتاهترین مسیر می باشند. مزیت استفاده از الگوریتم های کوتاهترین مسیر، کاهش میانگین زمان انتظار و عیب آن نیز کاهش قدرت کاوش در محیط است. در طراحی مسیر برای حرکت عامل ها از مبدا به مقصد، الگوریتمی مناسب خواهد بود که علاوه بر کاهش میانگین زمان انتظار، قدرت کاوش مناسبی نیز داشته باشد. در این مقاله الگوریتمی ترکیبی بر اساس کولونی مورچه ها و اتوماتای یادگیر ارائه شده است که با برقراری موازنه بین *Exploration* و *Exploitation* می تواند در شرایط مختلف خودش را با محیط وفق داده و قابلیت انعطاف بیشتری را به عامل ها بدهد. آزمایشات و شبیه سازی های انجام شده نشان می دهند که الگوریتم ارائه شده علاوه بر داشتن قابلیت انعطاف بالا، هم از نظر جستجو و کاوش و هم از نظر امتیاز کسب شده عملکرد بهتری نسبت به الگوریتم های مشابه و استفاده شده دارد.

واژه های کلیدی: طراحی مسیر، ترکیب کولونی مورچه ها و اتوماتای یادگیر، شبیه ساز امداد

۱- مقدمه

آمبولانس ها باید در سریعترین زمان ممکن به یاری مصدومین بشتابند و آنها را به پناهگاه منقل نمایند. آتش نشانها وظیفه دارند آتش را خاموش کرده و از انتشار آن به مکانهای دیگر جلوگیری کنند. وظیفه عامل های پلیس، جستجو در فضای حادثه و باز کردن راههای مسدود شده در اثر حادثه است تا سایر عامل ها در مسیر حرکت به سمت اهدافشان با مانعی روبرو نشوند. عامل های پلیس پس از اینکه به درخواست های عامل های دیگر پاسخ دادند و راههای مورد درخواست آنها را باز نمودند به جستجو در محیط پرداخته و در صورت مشاهده شهروندان مصدوم و یا ساختمان های آتش گرفته، موارد را به عامل های آمبولانس و آتش نشان گزارش می دهند. عاملها برای رفتن از یک نقطه به نقطه دیگر نیاز به یک الگوریتم مسیریابی دارند تا بر اساس آن مسیر حرکت بین مبدا و مقصد را مشخص کنند. اکثر تیم ها برای حرکت از یک نقطه به نقطه دیگر در محیط از الگوریتم های کوتاهترین مسیر استفاده می کنند [2,3,4].

آزمایشات و شبیه سازی ها نشان می دهد استفاده از

سیستم شبیه ساز امداد نمونه ای از سیستم های چند عامله بلادرنگ^۱ است که در آن محیط، عامل ها دارای پیچیدگی بالایی بوده و به شرایط واقعی نزدیکتر می باشد. عامل هایی که در این محیط فعالیت می کنند شامل عاملهای هم نوع و غیر هم نوع می باشند. هیچ عاملی اطلاعات کاملی از محیط نداشته و از نظر پهنای باند برای برقراری ارتباطات بین عاملها محدودیت وجود دارد [1]. فراهم بودن شرایط سیستم های چندعامله پیچیده و بزرگ در سیستم شبیه ساز امداد، آن را به بستر مناسبی برای تست و ارزیابی ایده ها و تکنیک های مربوط به سیستم های چند عامله پیچیده تبدیل کرده است. در سیستم شبیه ساز امداد، رویداد یک حادثه ویرانگر و مخرب مانند زلزله در یک محیط شهری شبیه سازی می شود. ساختمانها فرو می ریزند و یا آتش می گیرند، شهروندان در زیر آوار گیر می کنند و توسط آتش تهدید می شوند. راههای ارتباطی نیز به دلیل فرو ریختن آوار ساختمانها مسدود می شوند. در چنین شرایطی،

کوتاهترین مسیر برای حرکت عامل بین مبدا و مقصد، همیشه بهترین انتخاب نمی باشد. در صورت استفاده از الگوریتم های کوتاهترین مسیر در طراحی مسیر حرکت عامل، مکانهایی که عامل ها ملاقات می کنند محدود به مواردی می شود که در کوتاهترین مسیر بین مبدا و مقصد قرار گیرد. یعنی عامل برای حرکت از نقطه ای به نقطه دیگر از تمام مسیرهای ممکن استفاده نمی کند. مزیت استفاده از مسیرهای کوتاه، کاهش میانگین زمان انتظار اهداف و عیب آن نیز کاهش قدرت کاوش عامل هاست. با توجه به موارد بیان شده، برای کار در محیط های پیچیده مانند شبیه ساز امداد، الگوریتمی مناسب است که بتواند در موقعیت های مختلف خودش را با شرایط محیط مطابقت داده و موازنه متعادل بین exploitation (که در اینجا کاهش میانگین زمان انتظار اهداف است) و exploratio (جستجو و کاوش بیشتر در محیط) ایجاد نماید. برای این منظور الگوریتمی ارائه شده است که از قابلیت های الگوریتم های برمبنای کولونی مورچه ها و اتوماتای یادگیر به صورت ترکیبی استفاده نموده و قابلیت انعطاف و قدرت exploration بیشتری را به عامل ها داده و امتیاز نهایی کسب شده را نیز بهبود می دهد.

بخش بندی بقیه مطالب این مقاله به صورت زیر است: ابتدا در بخش ۲ کولونی مورچه ها به طور مختصر و خلاصه معرفی می شود. بخش ۳ در مورد اتوماتای یادگیر است. بخش ۴ به معرفی الگوریتم ارائه شده می پردازد. در بخش ۵ الگوریتم ارائه شده ارزیابی شده است. بخش ۶ مقایسه و نتیجه گیری می باشد.

۲- بهینه سازی با کولونی مورچه ها

کولونی مورچه ها الهام گرفتن از حل مسائل موجود در طبیعت توسط حس غریزی حشرات (مورچه ها) که یکی از جالب ترین سیستم های مورد مطالعه تا کنون است که کاربردهای فراوانی در علوم و مهندسی یافته است. یافتن کوتاهترین مسیر یک مسئله بهینه سازیست که گاه حل آن بسیار دشوار گاه نیز بسیار زمان بر است. مورچه ها قادر به یافتن کوتاهترین مسیر از لانه به منبع غذا هستند. راه حل مورچه ها برای پیدا کردن کوتاهترین مسیر بین لانه و منبع غذا به صورت زیر است: مورچه ها هنگام راه

رفتن از خود ردی از فرومون به جا می گذارند. فرومون با گذشت زمان تبخیر می شود اما در کوتاه مدت به عنوان رد مورچه بر سطح زمین باقی می ماند. یک رفتار بسیار ساده پایه ای در مورچه ها وجود دارد: آنها یک مسیر را از بین چندین مسیر بر اساس احتمال انتخاب می کنند. مسیری که فرومون بیشتری داشته باشد یا به عبارت دیگر مورچه های بیشتری قبلاً از آن عبور کرده باشند احتمال بیشتری برای انتخاب شدن خواهد داشت. در شروع کار انتخاب مسیر کاملاً تصادفیست اما بعد از اینکه در هر یک از مسیرها ردی از فرومون ایجاد شد مورچه ها به احتمال بیشتر مسیر با فرومون بیشتر را انتخاب می کنند (هر چه فرومون بیشتر باشد احتمال انتخاب مسیر بیشتر است). به همین دلیل پس از مدتی همه مورچه ها (با تقریباً همه مورچه ها) مسیر کوتاه تر را طی خواهند کرد.

۲-۱- مکانیزم انتخاب مسیر در الگوریتم های

کولونی مورچه ها

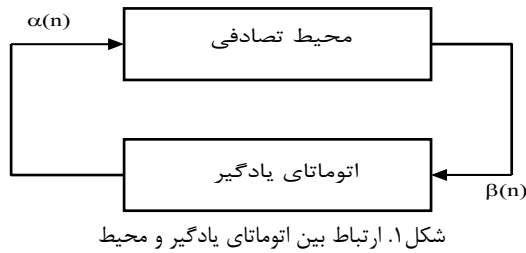
مورچه ها برای انتخاب گره بعدی از مقادیر فرومون ها و فاصله بین گره ها استفاده می کنند فرمول انتخاب گره بعدی رابطه مستقیمی با مقدار فرومون روی یال و عکس فاصله بین گره ها دارد. عکس فاصله بین دو گره را دید می نامیم و آنرا η_{ij} تعریف می کنیم. احتمال عبور از گره i به گره j طبق رابطه (۱) تعریف می شود. پارامترهای α و β به کاربر اجازه می دهند بر روی نسبت اهمیت مقدار دنباله فرومون به دید، کنترل داشته باشد.

$$p_{i,j} = \frac{[\tau_{i,j}]^{\alpha} \cdot [\eta_{i,j}]^{\beta}}{\sum_{j_k=1}^n [\tau_{i,j_k}]^{\alpha} \cdot [\eta_{i,j_k}]^{\beta}} \quad (1)$$

۲-۲- بهنگام سازی فرومون در کولونی مورچه ها

بعد از اینکه هر مورچه یک دور تولید کرد مقدار دنباله فرومون روی یال ها بهنگام می شود. در کولونی مورچه ها این عمل ابتدا با کاهش مقدار فرومون با یک فاکتور ثابت (ضریب تبخیر) و سپس قرار دادن فرومون توسط هر مورچه روی یال هایی که در دور خود پیموده است انجام می شود. ضریب تبخیر که مقداری بین صفر و یک دارد برای اجتناب از انباشتن بی حد دنباله فرومون است و

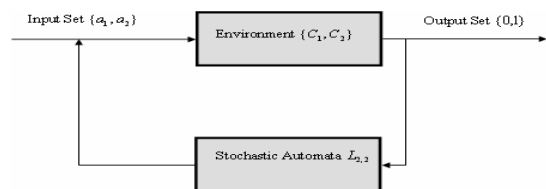
شده و فقط اتوماتای با ساختار ثابت به طور خلاصه معرفی می شود.



۳-۱- اتوماتای یادگیر با ساختار ثابت :

اتوماتای یادگیر با ساختار ثابت توسط ۵ تایی $\{\alpha, \beta, F, G, \phi\}$ نشان داده می شود که $\alpha \equiv \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه اعمال اتوماتا، $\beta \equiv \{\beta_1, \beta_2, \dots, \beta_m\}$ مجموعه ورودی های اتوماتا، $F: \phi \times \beta \rightarrow \phi$ تابعی که بر اساس پاسخ محیط، وضعیت جدید را می یابد، $G: \phi \rightarrow \alpha$ تابع خروجی که وضعیت کنونی را به خروجی بعدی می نگارد و $\phi(n) \equiv (\phi_1, \phi_2, \dots, \phi_k)$ مجموعه وضعیت های داخلی اتوماتای یادگیر می باشد. چند نمونه از اتوماتاهای یادگیر با ساختار ثابت در زیر معرفی می گردد:

اتوماتای دو حالته $L_{2,2}$: این اتوماتا دارای دو حالت ϕ_1 و ϕ_2 و دو خروجی α_1 و α_2 می باشد. ورودی های پذیرفته شده توسط اتوماتا مجموعه $\{0,1\}$ می باشد. ساختار این اتوماتا به شکلی طراحی شده است که تا زمان دریافت پاسخ مطلوب از محیط در یک اقدام باقی می ماند و به محض آنکه پاسخ نامطلوبی دریافت شود به سوی اقدام دیگر حرکت می نماید. نمودار عملکرد این اتوماتا در شکل ۲ نشان داده شده است.



شکل ۲- نمودار عملکرد اتوماتای دو حالته $L_{2,2}$

اتوماتای یادگیر $L_{2N,2}$ (اتوماتای ستلین): این نوع اتوماتا، تغییر یافته اتوماتای $L_{2,2}$ است و توسط ستلین معرفی شد که دارای $2N$ حالت و ۲ اقدام می باشد. سعی این نوع

الگوریتم را قادر می سازد تا تصمیم های بدی را که قبلاً گرفته شده است فراموش کند. روی یال هایی که توسط مورچه ها انتخاب نشده اند شدت فرمون با تعداد تکرارها بصورت نمایی کاهش می یابد. $t_{ij}(t+1)$ را شدت دنباله روی یال بین گره های i و j در لحظه $t+1$ می نامیم که با فرمول (۲) محاسبه می شود.

$$t_{ij}(t+1) = \rho \cdot t_{ij}(t) + \Delta t_{ij}^k(t, t+1) \quad (2)$$

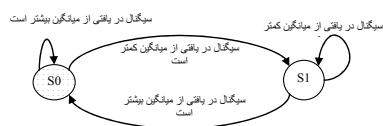
که در آن ρ ضریب تبخیر است و $\Delta t_{ij}^k(t, t+1)$ مقدار دنباله فرمون است که روی یال بین گره های i و j توسط مورچه k در زمان بین t و $t+1$ قرار داده می شود. مقدار دنباله فرمون در زمان 0 ، $t_{ij}(0)$ می تواند مقداری تصادفی باشد (عموماً مقداری کوچک). انتخاب های مختلفی در رابطه با چگونگی محاسبه $\Delta t_{ij}^k(t, t+1)$ و زمان بهنگام سازی $t_{ij}(t)$ وجود دارد که منجر به نسخه های مختلفی از الگوریتم کلونی مورچه ای می گردد. برای اطلاعات بیشتر در مورد الگوریتم های کلونی مورچه ها می توان به [5,6] مراجعه نمود.

۳- اتوماتای یادگیر

اتوماتای یادگیر یکی از مدل های یادگیری تقویتی است که در آن یک اتوماتا عملی بهینه را با توجه به اقدام انجام شده و بازخورد محیط فرا می گیرد. هدف نهایی اتوماتا، یادگیری انتخاب بهترین عمل از بین اعمال خود می باشد. بهترین عمل، عملی است که احتمال دریافت پاداش از محیط را به حداکثر برساند. کارکرد اتوماتای یادگیر در تعامل با محیط، در شکل ۱ مشاهده می شود. محیط را می توان توسط سه تایی $E \equiv \{\alpha, \beta, c\}$ نشان داد که در آن α مجموعه ورودی ها، β مجموعه خروجی ها و c مجموعه احتمال های جریمه می باشد. هرگاه β مجموعه ای دو عضوی $\{0,1\}$ باشد، محیط از نوع P است. در محیط از نوع Q ، $\beta(n)$ می تواند به طور گسسته مقداری از مقادیر محدود در فاصله $[0,1]$ را اختیار کند و در محیط از نوع S ، $\beta(n)$ متغیر تصادفی در فاصله $[0,1]$ است. اتوماتاهای یادگیر به دو دسته با ساختار ثابت و با ساختار متغیر دسته بندی می شوند. در این مقاله، به دلیل استفاده از اتوماتای یادگیر با ساختار ثابت، از پرداختن به اتوماتای با ساختار متغیر خودداری

۴- جستجو و کاوش در محیط با ترکیب کولونی مورچه ها و اتوماتای یادگیر

استفاده از الگوریتم کولونی مورچه ها برای جستجو و کاوش در محیط، علیرغم اینکه قدرت exploration بالایی را به عامل ها می دهد از نظر تنظیم پارامتر ها برای داشتن میانگین زمان مناسب، دارای پیچیدگی هایی می باشد که قابلیت انعطاف آن را در این مسئله کم می کند. الگوریتم ترکیبی ارائه شده، با تنظیم اتوماتیک پارامترهای الگوریتم کولونی مورچه ها توسط اتوماتای یادگیر، علاوه بر از بین بردن پیچیدگیهای تنظیم پارامترها، قابلیت انعطاف خوبی نیز به عامل ها در جستجو و کاوش محیط می دهد. الگوریتم ارائه ی شده برای جستجو و کاوش در محیط از ترکیب کولون مورچه ها و اتوماتای یادگیر استفاده می کند. بدین صورت که عامل پلیس علاوه بر اینکه برای جستجو و کاوش در محیط از الگوریتم کولونی مورچه ها استفاده می کند مجهز به یک اتوماتای یادگیر است که به صورت online پارامترهای الگوریتم کولونی مورچه ها را با توجه به وضعیت پیش آمده و به صورت اتوماتیک تنظیم می کند. می توان گفت مهمترین پارامتر الگوریتم کولونی مورچه ها که می تواند الگوریتم را به طرف موازنه exploration با exploitation سوق دهد پارامتر β می باشد که اهمیت هزینه یال (طول یال) را به عنوان جزئی از مسیر بیان می کند. در این الگوریتم مقدار پارامتر α ثابت بوده و اتوماتای یادگیر پارامتر β را بر اساس شرایط محیط تنظیم می کند. برای تنظیم پارامتر β از یک اتوماتای یادگیر با ساختار ثابت استفاده شده است. شکل ۵ ساختار اتوماتای استفاده شده را نشان می

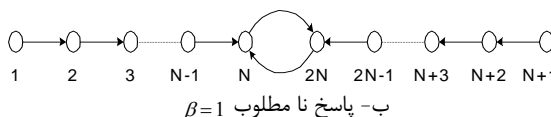
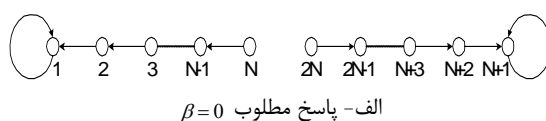


دهد:

شکل ۵- ساختار اتوماتای استفاده شده برای تنظیم پارامتر β

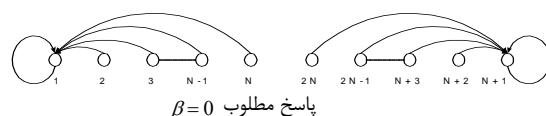
منظور از سیگنال دریافتی از محیط، مدت زمانی است که طول می کشد عامل از مبدا به مقصد برسد. برای قضاوت اتوماتا در مورد سیگنال دریافتی از محیط در قبال آخرین عمل انجام شده، از کمترین مقدار برای میانگین زمان

از اتوماتا برآن است که رفتار گذشته سیستم را جهت تصمیم گیری در مورد چگونگی انتقال از اعمال مورد استفاده قرار دهد. اتوماتای $L_{2N,2}$ هنگامی که یک جواب نامطلوب از محیط دریافت نماید از اقدامی به اقدام دیگر سوییچ می نماید در حالی که اتوماتای $L_{2N,2}$ اقدام به نگهداری تعداد موفقیت ها و شکست های دریافت شده به ازای هر اقدام می نماید و فقط زمانی که تعداد شکست ها بیش از تعداد موفقیت ها گردد (یا از یک مقدار حداکثر تجاوز نماید) به اقدام دیگر منتقل می شود. نمودار تغییر وضعیت این اتوماتای یادگیر مطابق شکل ۳ می باشد.



شکل ۳- نمودار تغییر وضعیت اتوماتای یادگیر $L_{2N,2}$

اتوماتای یادگیر Krinsky: این اتوماتای یادگیر زمانی که پاسخ محیط نامطلوب است، مانند اتوماتای یادگیر $L_{2N,2}$ رفتار می کند. اما برای پاسخ مطلوب هر وضعیت $\phi_i (i=1,2,3,...,N)$ به وضعیت ϕ_1 و هر وضعیت $\phi_i (i=N+1,N+2,...,2N)$ به وضعیت ϕ_{N+1} می رود. بنابراین همیشه N پاسخ نامطلوب متوالی لازم است تا اتوماتا یادگیر عمل خود را تغییر دهد. نمودار تغییر وضعیت این اتوماتای یادگیر برای پاسخ نامطلوب مانند اتوماتای یادگیر $L_{2N,2}$ بوده و برای پاسخ مطلوب مطابق شکل ۴ می باشد.



شکل ۴- نمودار تغییر وضعیت اتوماتای یادگیر Krinsky

برای مطالعه بیشتر در مورد اتوماتای یادگیر با ساختار ثابت و ساختار متغیر، می توان به منابع [7,8,9] مراجعه کرد.

انتظار که از الگوریتم کوتاهترین مسیر به دست آمده است استفاده می شود. فرآیند تنظیم پارامتر β توسط اتوماتای یادگیر به صورت زیر است:

شبیه سازی با مقادیر اولیه پارامترها شروع می شود. مقادیر اولیه مناسب باعث می شود الگوریتم سریعتر به جواب برسد و نیازی به یادگیری زیاد نباشد. مقادیر اولیه مناسب برای پارامترها با استفاده از روش آزمایش و خطا به دست می آیند. نتایج آزمایشات مربوط به محاسبه مقادیر اولیه مناسب در بخش بعدی به تفصیل ارائه شده است. در طول شبیه سازی، عامل پلیس هدفی را انتخاب نموده، زمان را یادداشت کرده و مطابق الگوریتم ارائه شده به سمت هدف حرکت می کند. موقعی که عامل پلیس به موقعیت هدف می رسد زمان سپری شده از لحظه انتخاب هدف تا رسیدن به هدف را به اتوماتای یادگیر می دهد (عکس العمل محیط برای اتوماتای یادگیر). اتوماتای یادگیر سیگنال دریافتی را با مقدار میانگین که از قبل به آن داده شده است مقایسه می کند که یکی از حالت های زیر پیش می آید:

اگر اتوماتا در حالت S_0 است و سیگنال دریافتی کمتر از مقدار میانگین است اتوماتای یادگیر در حالت S_0 باقی مانده و از مقدار پارامتر β به اندازه ضربی از اختلاف سیگنال دریافتی از محیط و مقدار میانگین کم می کند. این عمل باعث می شود تا در آینده قدرت exploration الگوریتم زیادتر شود.

اگر اتوماتا در حالت S_0 است و سیگنال دریافتی بیشتر از مقدار میانگین است اتوماتای یادگیر به حالت S_1 رفته و به مقدار پارامتر β به اندازه ضربی از اختلاف سیگنال دریافتی از محیط و مقدار میانگین را اضافه می کند. این عمل باعث می شود تا در آینده قدرت exploitation الگوریتم زیادتر شود.

اگر اتوماتا در حالت S_1 است و سیگنال دریافتی بیشتر از مقدار میانگین است اتوماتای یادگیر در حالت S_1 باقی مانده و به مقدار پارامتر β به اندازه ضربی از اختلاف سیگنال دریافتی از محیط و مقدار میانگین را اضافه می کند. این عمل باعث می شود تا در آینده قدرت exploitation الگوریتم زیادتر شود.

اگر اتوماتا در حالت S_1 است و سیگنال دریافتی کمتر از مقدار میانگین است اتوماتای یادگیر به حالت S_0 رفته و از مقدار پارامتر β به اندازه ضربی از اختلاف سیگنال دریافتی از محیط و مقدار میانگین کم می کند. این عمل باعث می شود تا در آینده قدرت exploration الگوریتم زیادتر شود.

در الگوریتم ترکیبی ارائه شده، از کولونی مورچه ها به صورت معکوس^۲ استفاده شده است. در شروع شبیه سازی بر روی تمام خیابانها فرومونی برابر با مقدار اولیه قرار داده می شود. (در این آزمایش ها این مقدار برابر ۵ قرار داده شده است). در هر واحد زمانی (ثانیه) که از شبیه سازی می گذرد فرومون تمام خیابان ها با یک ضربی (پارامتر Q یا نرخ تبخیر فرومون که در این آزمایشات ۰.۱ تعیین شده است) افزایش می یابد. عامل پلیس از هر خیابان که به عنوان جزئی از مسیر عبور می کند فرومون آن را به صورت ضربی از p کم می کند. مقدار این پارامتر برابر ۰.۳ تعیین شده است. از آنجا که در الگوریتم استفاده شده، اندازه فرومون یک خیابان با احتمال انتخاب آن خیابان رابطه مستقیم دارد کاهش فرومون یک خیابان، احتمال انتخاب آن را در آینده کاهش می دهد. عامل پلیس با این عمل (کاهش فرومون خیابان) سعی می کند از خیابانهایی که عبور کرده است عبور نکند. بنابراین از فرومون در جهت افزایش exploration الگوریتم استفاده می شود. اتوماتای یادگیر نیز با توجه به سیگنالی که از محیط می گیرد در صورت زیاد بودن میزان exploration ، مقدار پارامتر β را افزایش داده و آن را به سمت exploitation سوق می دهد. در صورت زیاد بودن exploitation ، اتوماتای یادگیر الگوریتم را به سمت exploration هدایت می کند و سعی در ایجاد موازنه بین این دو مورد دارد. الگوریتم ترکیبی استفاده شده مطابق شکل ۶ می باشد:

۵- ارزیابی الگوریتم جستجو و کاوش ترکیبی

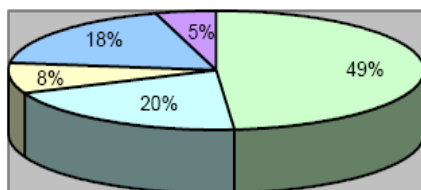
در این بخش، ابتدا نتایج آزمایشات مربوط به محاسبه مقدار اولیه مناسب برای پارامترها ارائه شده و سپس الگوریتم ارائه شده هم از نظر قدرت کاوش و هم از نظر امتیاز نهایی کسب شده ارزیابی می شود. همچنانکه قبلا نیز بیان شد، مقدار پارامتر p یا نرخ کاهش فرومون توسط

مورچه در کولونی مورچه ها با استفاده از روش سعی و خطا برابر ۰,۳ قرار داده شده است. جداول ۱ و ۲ به ترتیب تعداد گرهای ملاقات شده و میانگین زمان انتظار اهداف را به ازای مقادیر متفاوت ρ نشان می دهند.

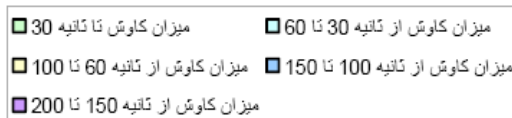
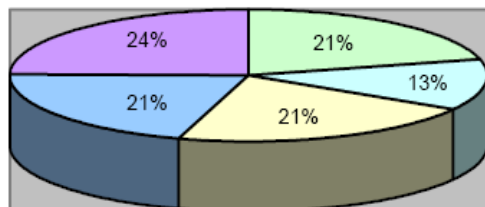
مقدار برای پارامتر ρ که هم قدرت کاوش بالایی به الگوریتم می دهد و هم میانگین زمان انتظار کمتری را نشان می دهد $\rho = 0,3$ می باشد.

۵-۱- ارزیابی از نظر قدرت کاوش و جستجو

عملکرد الگوریتم ارائه شده از نظر قدرت جستجو و کاوش در محیط، در نقشه های Kobe و VC (نقشه های استفاده شده در مسابقات ۲۰۰۸ چین) به ترتیب در شکل های ۷ و ۸ نشان داده شده است. برای پیاده سازی الگوریتم ارائه شده، از ساختار عامل های تیم Persia استفاده شده است [2,3]. کد عامل های آمبولانس و آتش نشان در تمام الگوریتم ها یکسان بوده و فقط کد و الگوریتم عامل های پلیس بازنویسی شده است. عملکرد الگوریتم تیم Persia که از کوتاهترین مسیر برای حرکت به سمت اهداف استفاده می کند (از نظر جستجو و کاوش در محیط) به عنوان معیار مقایسه در نقشه ای Kobe و VC به ترتیب در شکل های ۹ و ۱۰ ارائه شده است. دلیل ارائه عملکرد الگوریتم تیم Persia این است که عملکرد الگوریتم ارائه شده با آن مقایسه شده است.



شکل ۷. میزان کاوش عامل پلیس در الگوریتم ترکیبی - نقشه Kobe



شکل ۸. میزان کاوش عامل پلیس در الگوریتم ترکیبی - نقشه VC

- ۱- فرومون تمام یالها، مقادیر پارامترهای ρ و Q را مقداردهی اولیه کن. ($\alpha = 0.1, \beta = 0.1, \rho = 0.3, Q = 5$)
- ۲- هدفی را انتخاب کن.
- ۳- زمان را یادداشت کن.
- ۴- تا زمانی که به هدف (مقصد) نرسیده ای دستورالعمل های ۵ تا ۹ را اجرا کن.
- ۵- با هر پالس شبیه سازی، فرومون تمام یالها را افزایش بده.
- ۶- مقدار احتمال یال های متصل به گرهی را که در آن قرار داری طبق فرمول شماره (۱) محاسبه کن.
- ۷- یکی از یال های متصل به گرهِ فعلی را به تصادف انتخاب کن.
- ۸- فرومون یال انتخاب شده را مطابق فرمول شماره (۲) کاهش بده.
- ۹- به گرهی که در انتهای دیگر یال انتخاب شده قرار دارد برو.
- ۱۰- زمان صرف شده را به عنوان سیگنال ورودی به اتوماتای یادگیر بده.
- ۱۱- از اتوماتای یادگیر مقدار جدید پارامتر β را دریافت کن و در فرمول (۱) اعمال کن.

شکل ۶- الگوریتم ترکیبی ارائه شده برای طراحی مسیرعامل پلیس

جدول ۱- مقادیر مختلف پارامتر ρ و تاثیر آن در قدرت کاوش الگوریتم (تعداد گرهای ملاقات شده توسط عامل پلیس)

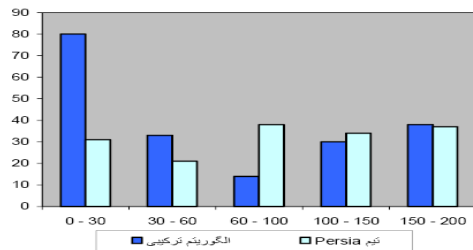
تعداد	$\rho=0.1$	$\rho=0.2$	$\rho=0.3$	$\rho=0.4$	$\rho=0.5$	$\rho=0.6$	$\rho=0.7$	$\rho=0.8$
15	175	118	168	138	152	188	173	184
30	198	165	233	148	157	207	185	192
45	198	215	255	182	162	209	198	192
60	202	215	236	197	197	209	267	192
100	265	245	237	235	201	253	291	206
150	281	281	249	234	230	269	291	219
200	291	291	265	291	235	291	291	245
250	291	291	274	291	235	291	291	257
300	291	291	287	291	280	291	291	287

تعداد کل گرهِ ها = ۲۹۳

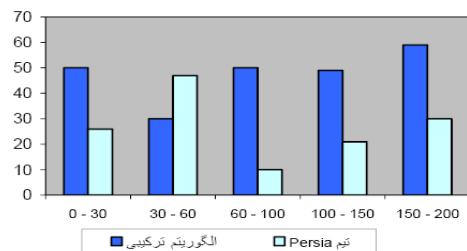
جدول ۲- مقادیر مختلف پارامتر ρ و تاثیر آن در میانگین زمان انتظار اهداف انتخاب شده توسط عامل پلیس

$\rho=0.8$	$\rho=0.7$	$\rho=0.6$	$\rho=0.5$	$\rho=0.4$	$\rho=0.3$	$\rho=0.2$	$\rho=0.1$	
250	146	65	250	95	36✓	66	91	میانگین زمان انتظار اهداف (بر حسب ثانیه)

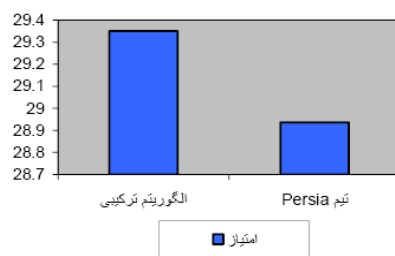
همچنانکه جداول ۱ و ۲ نیز نشان می دهند مناسبترین



شکل ۱۱. مقایسه فضای کاوش شده توسط الگوریتم ترکیبی و تیم Persia - نقشه Kobe



شکل ۱۲. مقایسه فضای کاوش شده توسط الگوریتم ترکیبی و تیم Persia - نقشه VC

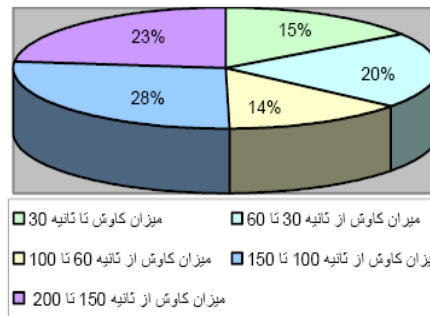


شکل ۱۳. مقایسه الگوریتم ترکیبی و تیم Persia از نظر امتیاز کسب شده

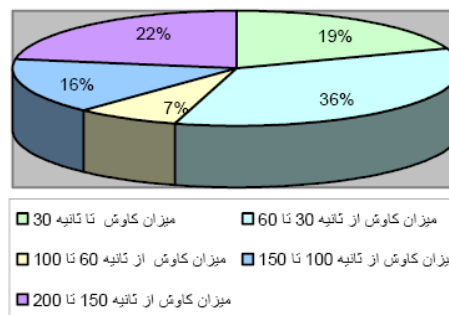
همچنانکه شکل های ۱۱ و ۱۲ نیز نشان می دهند الگوریتم ارائه شده از نظر قدرت کاوش در محیط، برتری چشمگیری را نسبت به الگوریتم هایی که از کوتاهترین مسیر استفاده می کنند نشان می دهد. آزمایش ها نشان می دهند (شکل ۱۳) که الگوریتم ارائه شده از نظر امتیاز نهایی کسب شده نیز بهتر از الگوریتم هایی عمل می کند که از کوتاهترین مسیر برای حرکت به سمت هدف استفاده می کنند.

منابع و مراجع :

- [1] A. Kleiner, M. Brenner, T. Bräuer, C. Dornhege, M. Göbelbecker, M. Luber, J. Prediger, J. Stuckler, and B. Nebel, "Successful Search and Rescue in Simulated Disaster Areas", *Institut für Informatik*, University at Freiburg, 79110 Freiburg, Germany, 2006.
- [2] M. R. Khojasteh and H. Heidari, "Persia 2005 Team Description", *Team Description Paper*, 2005.



شکل ۹. میزان کاوش عامل پلیس در تیم Persia - نقشه Kobe



شکل ۱۰. میزان کاوش عامل پلیس در تیم Persia - نقشه VC

۵-۲- ارزیابی از نظر امتیاز کسب شده:

برای ارزیابی الگوریتم ارائه شده از نظر امتیاز نهایی، امتیاز کسب شده از ۴ بار اجرا و شبیه سازی الگوریتم ارائه شده و تیم Persia یادداشت شده است که نتایج آن به صورت جدول ۳ می باشد.

جدول ۳- امتیاز کسب شده توسط الگوریتم ترکیبی و تیم Persia در ۴ بار اجرا

ردیف	1	2	3	4	میانگین
الگوریتم ترکیبی	28.37	28.45	29.22	31.84	29.35
تیم Persia	27.96	30.01	27.63	29.29	29.35

۶- مقایسه و نتیجه گیری

در این قسمت عملکرد الگوریتم ارائه شده با الگوریتم تیم Persia (که از الگوریتم کوتاهترین مسیر استفاده می کند) هم از نظر میزان کاوش و جستجو در محیط و هم از نظر امتیاز نهایی کسب شده مقایسه شده است. شکل های ۱۱ و ۱۲ این مقایسه را به ترتیب برای نقشه های VC و Kobe از نظر میزان کاوش در محیط نشان می دهند. شکل ۱۳ نیز دو الگوریتم را از نظر امتیاز نهایی کسب شده مقایسه می کند.

- [3] M. R. Khojasteh, A. Kazimi and Z. Ghaseminik, "Persia 2006, Towards a Full Learning Automata-Based Cooperative Team", *Team Description Paper*, 2006.
 - [4] S. B. M. Post and M. L. Fassaert, "A Communication and Coordination Model For 'RobocupRescue' Agents", *M.Sc. thesis, Department of Computer Science, University of Amsterdam*, 2004.
 - [5] M. Dorigo and T. Stützle, *Ant Colony Optimization*, Cambridge, MA: The MIT Press, 2000.
 - [6] M. Dorigo, G. Di Caro and M. L. Gambardella, "Ant Algorithms for Discrete Optimization", *Artif. Life*, vol. 5, no. 2, pp. 137–172, 1999.
 - [7] M. R. Meybodi, F. A. Mohammadi, "Optimizing ACS using Learning Automata", *Technical Report, Soft Computing Lab, Amirkabir University of Technology*, 2004.
 - [8] K. S. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*, Prentice Hall, Inc., 1989.
 - [9] M. A. L. Thathachar, P. S. Sastry, "Varieties of Learning Automata: An Overview", *IEEE Transaction on System, Man and Cybernetics, Part B*, Vol. 32, No. 6, P. 711-722, 2002.
 - [10] P. Mars, J. R. Chen and R. Nambiar, "Learning Algorithms: Theory and Application in Signal Processing, Control and Communications", *CRC press*, New York, 1998.
 - [11] The Robocup Rescue Technical Committee, "Robocup-Rescue Simulator Manual", 2000, Web page : <http://robomec.cs.kobe-u.ac.jp/robocup-rescue>.
 - [12] M. Bowling, "Robocup Rescue : Agent Development Kit", *Version 0.4*, 2001.
 - [13] T. Morimoto, "How to Develop a RoboCupRescue Agent for RoboCupRescue Simulation System", *version 0*, 1st edition, 2002.
 - [14] RoboCupRescue Official Site, <http://robomec.cs.kobe-u.ac.jp/robocup-rescue/>.
 - [15] H. Kitano, S. Tadokor, H. Noda, I. Matsubara, T. Takhasi, A. Shinjou, and S. Shimada, "Search And Rescue For Large Scale Disaster As a Domain For Multi-Agent Systems", *Kluwer Academic Publisher*, 2002.
-

1 Real-Time
2 Repelling