ارتباط و هماهنگی بین عاملها در سیستم های چند عاملی با استفاده از ساختار عصبی خودسازمانده و الگوریتم ژنتیک

علی پورحسین[۱]، محمدرضا اکبرزاده توتونچی[۲]
[۱]دانشجوی کارشناسی ارشد کنترل دانشگاه فردوسی مشهد، orgpoorhossein@ieee.
[۲]دانشیار گروه کنترل دانشگاه فردوسی مشهد، akbarzadeh@ieee.org

چکیده - در این مقاله برقراری ارتباط و تعامل بین عاملها در یک محیط چند عاملی دینامیک مورد بررسی قرار گرفته و راه حلی برای بهبود و تسریع در این ارتباط ارائه گردیده است. با توجه به اینکه ساختار عصبی خودسازمانده (*FALCON: fusion architecture for learning, cognition, and navigation*) توانایی فرمول بندی محیط های چند عاملی را به خوبی دارا می باشد، از این شبکه برای توصیف محیط و عاملها در یک سیستم چندعاملی استفاده گردیده است. استفاده ی منفرد از این شبکه با وجود مزیتهای فراوان نسبت به راه های گذشته، دارای حجم بالای محاسباتی و به وجود آمدن حالتهای فراوان می گردد. در این مقاله بکارگیری الگوریتم ژنتیک همراه با استفاده ی همزمان از شبکه های موازی *FALCON* به جای یک شبکه ی حجیم، سبب بهینه شدن ارتباط بین عاملها و برآورده شدن هدف های مشترک و کم شدن حجم محاسبات گردیده است. در نهایت راه حل ارائه شده بر روی مسئله ی شکار و شکارچی پیاده سازی شده است و نتایج شبیه سازی با روش های پایه مقایسه شده است.

کلید واژه- الگوریتم ژنتیک، ساختار عصبی خودسازمانده، سیستم های چندعاملی همکار، مسئله ی شکار و شکارچی، یادگیری Q.

۱- مقدمه

امروزه بسیاری از سیستم ها را می توان با نگاهی جدید به صورت سیستم چندعاملی(multiagent systems) مورد مطالعه قرار داد. همکاری گروهی عاملها(team work)، به دلیل افزایش توانایی در رسیدن به هدف نهایی سیستم و بهبود استراتژی کلی مورد نظر، از مهمترین مسائلی میباشد که همواره در سیستمهای چند عاملی مطرح است[۱].

این مسئله هنگامی اهمیت بیشتری پیدا میکند که محیط، پیچیده و دینامیک باشد. در چنین محیطی یک عامل هدفمند علاوه بر تغییرات محیط از عمل سایر عوامل نیز تاثیر می پذیرد. بنابراین محیط نسبت به قبل دارای ابعاد دینامیک بیشتری می گردد و عامل باید توانایی مدل کردن فرایند عمل و قدرت یادگیری و برقراری تعامل با سایر عوامل دیگر را داشته باشد.

استفاده از روشهای کلاسیک در توصیف عوامل و برقراری ارتباط بین آنها در یک محیط چندعاملی، به دلیل استفاده از معادلات فراوان، قدرت گسترش شبکه به سیستمهای با اندازه ی بزرگ را ضعیف تر می نماید[۲]. با تعریف یک سیستم هوشمند و استفاده از روشهای هوشمند در حل اینگونه مسائل، روشهایی نظیر استفاده از الگوریتم های ژنتیک و یادگیری تقویتی، سودمند بودن خود را در اینگونه محیط ها اثبات نمودند[۳-۶]. علاوه بر آن سیستم چندعاملی را میتوان به صورت یک ساختار عصبی خود سازمانده که به اختصار به آن FALCON اطلاق می شود، در نظر گرفت. استفاده از این ساختار توانایی فرمول بندی عاملها را به صورت مدون افزایش داده و گسترش آن را به سیستم های با اندازه های بزرگتر فراهم می آورد[۷][۸].

در این مقاله بهینه کردن ارتباط بین عوامل و یادگیری از محیط در ساختارهای FALCON با استفاده از یادگیری تقویتی و الگوریتم ژنتیک و همچنین تغییر در نگاه به سیستم از نقطه نظر حالتها و عاملهای موجود در جهت بهبود بخشی به ارتباط بین عوامل و تسریع در رسیدن به هدف مشترک، مورد نظر می باشد و راه حلی جهت اینکه چگونه تیمی از عاملها در یک محیط دینامیک و هوشمند با یکدیگر ارتباط داشته، از محیط فراگیری کرده و این فراگیری را در اختیار هم قرار دهند، ارائه شده است.

سایر قسمت های مقاله به شرح زیر است. در بخش ۲ به بررسی پیشینه ی یادگیری و ارتباط در سیستمهای هوشمند پرداخته شده است. بخش ۳ شامل ارائه و توصیف ساختار FALCON می باشد. سپس روش ارائه شده در بخش ۴ معرفی گردیده و بخش ۵ شامل گزارشی از پیاده سازی روش بر روی بازی شکار و شکارچی می باشد و درنهایت در بخش ۶ نتیجه گیری مطرح

گردیده است.

۲- یادگیری و ارتباط در سیستمهای هوشمند

توجه این مقاله بر یادگیری عاملهای همکار در محیطی که چندین عامل قصد رسیدن به یک هدف مشترک و یا بهینه کردن یک تابع هدف مشخص را با توجه به ارتباط با یکدیگر دارند[۳]، متمرکز شده است. بطور کلی یادگیری در سیستمهای چندعاملی به دو نوع RL (یادگیری تقویتی) و یا جستجوی اتفاقی (stochastic search) تقسیم بندی می گردد. جستجوی اتفاقی بر اساس یادگیری از مجموعه ی حلهای ممکن یک عمل به صورت انتخاب کاملا اتفاقی بوده و رفتار عاملها در یک جهت خاص اصلاح می گردد[۹][۱۰].

در یادگیری تقویتی، یک ویا دو تابع با عنوان تابع سیاست (policy function) که رابطه ی بین هر حالت و عمل موردنظر را توصیف می کند و تابع مقدار (value function) که ارزش هر جفت حالت و عمل را با یک مقدار معین بیان می کند، ارائه می شود. برای یادگیری تابع مقدار یک روش متعارف استفاده از الگوریتم Q می باشد[۱۱] که یک روش مرحله ای برای تعیین پاداش ها به ازای انجام عمل خاص در یک حالت خاص می باشد.

برای بهینه کردن و توسعه ی یادگیری Q تحقیقات فراوانی در زمینه های مختلف صورت گرفته است[۱۲-۱۴] که اگرچه سرعت یادگیری از محیط و انجام عمل را بهبود می بخشد اما به دلیل تغییرات یک محیط طبیعی و افزایش الگوهای مورد استفاده در این محیط، سرعت یادگیری کم شده و استفاده ی عملی از الگوریتم Q به تنهایی را مشکل می سازد[۸].

ساختارهای عصبی خودسازمانده اگرچه از ساختار Q استفاده می نمایند، اما به دلیل سرعت بیشتر و در عین حال پایدار ماندن سیستم[۷] قابلیت اعمال در محیط های واقعی را دارا می باشند[۸].

استراتژی همکاری در سیستم های چندعاملی به دو صورت امکان پذیر می باشد. استراتژی یادگیری تیمی (team learning) و یا یادگیری همزمان (concurrent learning)[۹]. یادگیری تیمی از یک عامل یادگیرنده برای یادگیری از سیستم استفاده می نماید و تمام اطلاعات توسط یک عامل جمع آوری شده و به بقیه ی عاملها ابلاغ می گردد. در یادگیری همزمان، هر عامل توانایی یادگیری به صورت جداگانه را دارا بوده بنابراین در این شرایط سه ویژگی عمده ی گرفتن پاداش، متغییر بودن فرایند یادگیری با زمان و مدل سازی سایر عوامل، برای یک عامل در نظر گرفته می شود[۱۵].

ساختار TD-FALCON به همراه الگوریتم ژنتیک ارائه شده در این مقاله از یادگیری همزمان عاملها بهره جسته، پاداش بر اساس در نظر گرفتن هدف های موجود و عوامل دیگر، متغیر بودن فرایند یادگیری در حین عمل بر اساس بهینه کردن پاداش تمام عاملها در هر لحظه از فرایند انجام عمل و مدل کردن عاملهای دیگر بر اساس ارائه ی شبکه های موازی FALCON صورت گرفته است.

۳- شبکه های FALCON

شبکه های FALCON ساختاری برای یادگیری و عمل در محیط های چند عاملی میباشند که چندین متغییر را به عنوان ورودی دریافت کرده و بر اساس آن سایر موارد را به روز رسانی میکنند[۷].

۳-۱- پیکره ی FALCON

این شبکه از مجموعه های ورودی، محیط عمل، مقادیر فعال و مقادیر ثابت تشکیل شده است.

۳-۱-۱- مجموعه های ورودی

مجموعه های ورودی شامل ۳ نوع میباشند:

حالتهای سیستم (State): حالتهائی که سیستم چند عاملی در حین عمل میتواند دارا باشد که به آن زمینهٔ F_1^{c1} می گوئیم که شامل مقدارهای $S=(s_1,s_2,...,s_n)$ میباشد.

عمل های ممکن(Action): عمل هائی که سیستم در حین کار میتواند انجام دهد که با F_1^{c2} نمایش میدهیم که شامل مقدارهای $A=(a_1,a_2,...,a_m)$ میباشد

پاداش(Reward): پاداشی که عامل ها از محیط به دلیل انجام عمل خود می گیرندکه با F_1^{c3} نمایش میدهیم که شامل بردارهای $R=(r,\bar{r})$ بوده که $r \in [0,1]$ و $\bar{r}=1-r$ است.

۳-۱-۲- محیط عمل (محیط کار)

محیطی میباشد که در آن بردارهائی از ورودی که فعال هستند به کار گرفته شده و بر اساس آنها تصمیمات بعدی و ارتباط و یادگیری از محیط صورت میگیرد که با F_2^c نمایش داده میشود.

۳-۱-۳- مقادیر فعال

این مقادیر در محیط عمل تعریف میگردند:

- x^{ck}، مقادیر مربوط به مجموعه های ورودی F_1^{ck} میباشند.

- y^c، مقدار مربوط به محیط عمل F_2^c میباشد.

- مقادیر وزنی w_j^{ck}، مقادیری جهت میزان یادگیری زمینه های مختلف ورودی F_1^{ck} در j امین گره از F_2^c می باشند.

۳-۱-۴- مقادیر ثابت

ضرایب قابل اعمالی هستند که تاثیر ویژگیهای مختلف در شبکه را به وسیله ی آنها می توان تنظیم نمود.

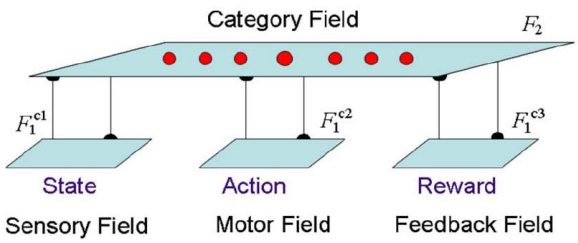
$\alpha^{ck} \succ 0$: پارامتر انتخاب هر کدام از زمینه های ورودی F_1^{ck}.

$\beta^{ck} \in [0,1]$: سرعت یادگیری هر کدام از زمینه های ورودی F_1^{ck}.

$\gamma^{ck} \in [0,1]$: پارامتر همکاری برای هر کدام از زمینه های ورودی F_1^{ck} بطوری که $\sum_{k=1}^{3} \gamma^{ck} = 1$.

$\rho^{ck} \in [0,1]$: ضریب احتیاط برای هر کدام از زمینه های ورودی F_1^{ck}.

در شکل ۱ ساختار FALCON ارائه شده است.



شکل ۱: ساختار FALCON

۳-۲- اساس کار FALCON

اساس کار FALCON شامل دو قسمت پیش بینی مقدار ها و یادگیری مقدارهاست.

۱-۱-۱- پیش بینی مقادیر

ابتدا مقادیر فعال ورودی دریافت شده و بر اساس آنها مقدار R محاسبه میشود. نحوهٔ محاسبه به صورت زیر است:

مقادیر فعال ورودی را به صورت زیر تعریف میکنیم:

$$x^{c1} = S, x^{c2} = A, x^{c3} = N \qquad (۱)$$

در این حالت مقدار x^{c3} که پاداش عملها می باشد معلوم نیست (N=1).

سپس سه مرحلهٔ زیر را انجام میدهیم:

مرحلهٔ ۱: فعال سازی کد

برای هر گره در زمینهٔ F_2^c یک کد با عنوان T_j^c تعریف می کنیم که j بیان کنندهٔ گره jام میباشد.

$$T_j^c = \sum_{k=1}^{3} \gamma^{ck} \frac{\left| x^{ck} \wedge w_j^{ck} \right|}{\alpha^{ck} + \left| w_j^{ck} \right|} \qquad (۲)$$

که عملگرهای $\wedge, |.|$ ،عملگرهای فازی بوده و به صورت زیر تعریف می گردند.

$$(p \wedge q) \equiv \min(p,q), |p| = \sum_i p_i \qquad (۳)$$

مرحلهٔ ۲: مقایسهٔ کد

با توجه به T_j^c های به دست آمده در مرحلهٔ قبل، گرهی انتخاب میشود که دارای بیشترین مقدار کد باشد:

$$T_i^c = \max\left\{ T_i^c : for\ all\ F_2^c\ node\ j \right\} \qquad (۴)$$

هنگامی که این گره انتخاب شد، آن را J قرار داده و $y_J^c = 1$ قرار میدهیم و بردار های سایر گره ها را برابر با صفر در نظر میگیریم.

مرحلهٔ ۳: فراخوانی کد

گره انتخاب شدهٔ J، بردارهای وزنی خود را فراخوانی کرده و بردارهای ورودی F_1^{ck} را بر اساس آن به صورت زیر تغییر میدهد:

$$x^{ck(new)} = x^{ck(old)} \wedge w_i^{ck} \qquad (۵)$$

عملگر $\wedge$ مانند مراحل قبل عملگر فازی می باشد.

با انجام ۳ مرحلهٔ بالا مقدار x^{c3} به R ارتقاء پیدا میکند.

۱-۱-۲- یادگیری مقادیر

هدف این قسمت ارتقاء مقادیر بردارهای وزنی w^{ck} می باشد.

با توجه به اینکه مقادیر x^{c3} در مرحله ی پیش بینی مقادیر بدست آمده است، ابتدا بردارهای ورودی را به صورت زیر تعریف میکنیم (رابطه ی ۱ به صورت زیر اصلاح می گردد).

$$x^{c1} = S, x^{c2} = A, x^{c3} = R \qquad (۶)$$

مانند قسمت پیش بینی مقادیر بردارها ۲ مرحله ی فعال سازی کد و مقایسهٔ کد را انجام میدهیم تا گره J انتخاب گردد. سپس الگوی یادگیری را به صورت زیر انجام می دهیم :

هنگامی که گره J انتخاب شد، w_j^{ck} برای هر کدام از زمینه های ورودی k به صورت زیر دوباره تعریف میشود:

$$w_j^{ck(new)} = (1-\beta^{ck}) w_j^{ck(old)} + \beta^{ck} (x^{ck} \wedge w_j^{ck(old)}) \qquad (۷)$$

۴- ارائه ی راه حل

برای پیاده سازی و استفاده از شبکهٔ FALCON در محیط چند عاملی با عاملهای مختلف و ارتباط عاملها با یکدیگر از ساختار چند مرحله ای FALCON استفاده می کنیم.

۴-۱- الگوریتم ساختار

سیستم را به ساختارهای موازی FALCON برای هر عامل تبدیل می کنیم. برای هر یک از ساختارها از الگوریتم زیر استفاده می کنیم.

۱- برای حالتی که عامل در آن قرار دارد، عمل های موجود را در نظر گرفته و با توجه به حالت و عمل های ممکن گره های FALCON را تعیین نموده و بر اساس قسمت پیش بینی مقدار بردارها، مقدار پاداش Q(s,a) را محاسبه می نمائیم.

۲- با توجه به مقدار پاداش و نوع انتخاب عمل (action selection policy) یک عمل را انتخاب می کنیم.

Action selection policy: به دلیل تعادل بین جستجو (exploration) و بهره برداری ازپاداش ها (exploitation) همیشه بهترین عمل را انتخاب نمی کنیم، بلکه با ضریب $0 \prec \varepsilon \prec 1$ سعی می کنیم که عمل های دیگر را نیز انجام دهیم($\varepsilon - greedy$).

۳- عمل را انجام داده و به حالت s' می رویم. با توجه به این حالت پاداش r را از محیط به دست می آوریم.

۴- مقدار پاداش جدید Q(s,a) را با توجه به Q-Learning و روابط زیر محاسبه می کنیم.

$$\begin{aligned}
&\Delta Q(s,a) = \alpha TD_{err} \\
&err = r + \gamma \max_{a'} Q(s',a') - Q(s,a) \\
&\Delta Q(s,a) = \alpha TD_{err}(1 - Q(s,a)) \\
&Q^{(new)}(s,a) = Q(s,a) + \Delta Q \\
&R = (Q^{(new)}(s,a), 1 - Q^{(new)}(s,a))
\end{aligned} \qquad (۸)$$

۵- قسمت یادگیری مقادیر را با توجه به مقادیر A,S,R محاسبه می نمائیم.

۶- حالت جدید را به حالت s منتقل می نمائیم.

۷- تا هنگام رسیدن به هدف برنامه را از مرحلهٔ ۲ تکرار می کنیم.

۴-۲- استفاده از الگوریتم ژنتیک

همانگونه که ذکر شد، مرحله ی ۱ از قسمت ۴-۱ نیاز به تعیین پاداش Q(s,a) بر اساس قسمت پیش بینی مقدارها می باشد. اگرچه محیط به صورت دینامیک و ناشناخته می باشد، اما برخی ویژگیهای محیط قابل شناسایی و اندازه گیری بوده و در بسیاری از موارد تعریف یک تابع برازش با توجه به ویژگیهای محیط، سایر عوامل و اهداف مورد نظر توانایی بهره گرفتن از محیط را بالا برده و بهینه کردن آن با استفاده از الگوریتم ژنتیک مفیدتر از مرحله ی پیش بینی مقادیر می باشد، در عین حال بر اساس یک تصمیم بهینه ی بدست آمده از الگوریتم ژنتیک مقادیر Q(s,a) و به تبع آن مقادیر w_j^{ck} با دقت و سرعت بیشتری تصحیح می گردد.

$$w_j^{ck(new)} = w_j^{ck(genetic)} \wedge w_j^{ck} \qquad (۹)$$

بر این اساس از چندین الگوریتم ژنتیک موازی برای عامل های سیستم استفاده شده، برای هر عامل جمعیت اولیه ای از حالتهای موجود و عملهای ممکن را تشکیل داده و تابع برازش مورد نظر را بهینه می کنیم.

در قسمت ۵ تابع برازشی برای بهینه شدن مسیر حرکت در یک مسئله ی مسیریابی ارائه شده است.

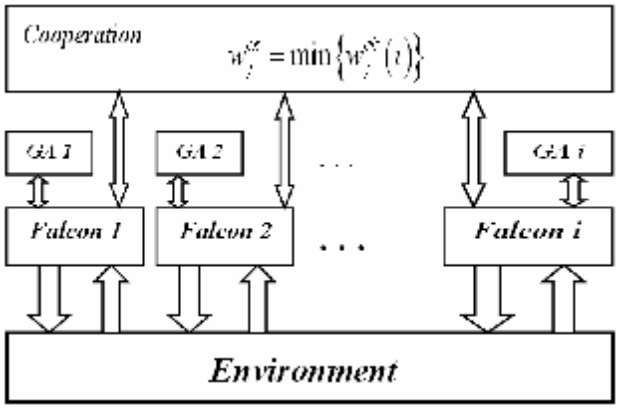
۴-۳- نحوهٔ ارتباط عاملها در ساختار عصبی خودسازمانده

جهت ارتباط عاملها با یکدیگر در مرحلهٔ ۶ که مقادیر w_j^{ck} به روز می شود به صورت زیر عمل می کنیم:

با توجه به اینکه هر عامل دارای شبکهٔ جدائی می باشد و گره های خاص خود را داراست، هنگام به روز کردن w_j^{ck} مربوط به یک عامل از گره های متناظر با آن در شبکه های مربوط به دیگر عاملها استفاده نموده و آن را بر اساس بهترین گره متناظر به روز می کنیم. این عمل سبب برقراری ارتباط بین شبکه های FALCON و تصمیم گیری بهتری شده و بدین معنی است که قبلا عامل دیگری در این شرایط از محیط قرار گرفته و اگر عمل او بهتر باشد عوامل دیگر از او پیروی میکنند.

$$w_j^{ck(new)} = \min \left\{ \begin{matrix} w_j^{ck}(i) \in FALCON(i) \\ ,i = number\ of\ Agents \end{matrix} \right\} \qquad (۱۰)$$

شکل ۲ ساختار ارائه شده را توصیف می کند.



شکل ۲: توصیف محیط حاوی چندین عامل با شبکه های موازی FALCON و ژنتیک و نحوه ی ارتباط آنها

۵- شبیه سازی محیط شکار و شکارچی

جهت بررسی روش ارائه شده در یک محیط چند عاملی، از بازی شکار و شکارچی استفاده گردیده است.

۵-۱- شرایط محیط

- محیط یک محدوده مربعی با ابعاد ۳۰ × ۳۰ می باشد .
- این بازی دارای ۲ شکارچی و یک شکار است.
- سرعت حرکت شکارچی ۲ برابر شکار است.
- شکار و شکارچی بر تمام محیط احاطه نداشته و دارای زاویهٔ دید محدود می باشند. زاویهٔ دید شکار و شکارچی برابر با ۴ در نظر گرفته شده است.
- شکار در صورت مشاهده ی شکارچی از آن فرار می کند.
- در صورتی شکار به دام می افتد که توسط هر دو شکارچی همزمان احاطه شود.

۵-۲- شبیه سازی بازی

در این بازی برای هر عامل یک ساختار FALCON با تعداد گره هایی به ابعاد محیط (30*30=900) و تعداد عمل ها نیز حرکت به جهات مختلف که شامل ۸ عمل ممکن است در نظر گرفته شد.

جهت تسریع و تقویت یادگیری از محیط و بهره برداری از حالت جهانی (global: شکار در محدوده ی دید شکارچی قرار می گیرد)، در زمانهایی از بازی که شکار در محدوده ی دید شکارچی قرار داشت از الگوریتم ژنتیک برای به روزکردن w_j^{ck} به صورت زیر استفاده گردید.

برای هر عامل جمعیت اولیه با طول کروموزوم برابر با میدان دید شکارچی از مسیرهای ممکن را تشکیل داده و با استفاده از الگوریتم ژنتیک تابع برازش زیر را مینیمم می کنیم[۱۲].

$$Path\ Evaluate = K_1 \times K_2 \times (W_d \times Dis\tan ce\cos t) \quad (۱۱)$$

$$K_1 = e^{\left(\frac{-\alpha \times Goaldis}{Startdis}\right)} \quad (۱۲)$$

Goaldis فاصله آخرین نقطه مسیر تا موقعیت هدف، Startdis فاصله آخرین نقطه مسیر تا موقعیت عامل جستجوگر و K_2 جریمه یا پنالتی می باشد که با توجه به محیط و مسیر حرکت، در مسیرهای نامناسب برابر با صفر تعریف می شود.

برای برقراری تعامل و ارتباط بین عاملها هنگامی که یک شکارچی به نقطه ای از محیط می رسد از اطلاعات سایر شکارچی ها در آن نقطه استفاده می کند.

نوع و مقدار پاداش از محیط نیز هنگامی که شکارچی ها قادر به دیدن شکار نباشند به ازای فاصلهٔ بیشتر از هم پاداش بهتری می گیرند که این امر سبب می شود که محیط وسیع تری جستجو گردد و هنگامی که شکار در محدوده ی دید باشد این نوع پاداش حذف شده و شکارچی ها به هم نزدیک می شوند. مقدار r در رابطه ی ۸ به صورت زیر محاسبه می گردد.

$$r = k\left(0.02k_2 d_{predator} + k_3 / d_{predator} + k_1 / d_{prey}\right) \quad (۱۳)$$

$d_{predator}$، فاصله تا شکارچی دیگر و d_{prey}، فاصله تا شکار می باشد. اگر شکار در محدوده ی دید شکارچی بود $k_3 = 0, k_2, k_1 = 1$ و در غیر این صورت $k_2 = 0, k_3, k_1 = 1$ می باشد.

۵-۳- نتایج به دست آمده

شبیه سازی را ۱۰۰۰ مرتبه و هرمرتبه تا رسیدن به هدف انجام دادیم که نتایج زیر به دست آمد:

در سریعترین حالت شکارچی ها پس از ۲۰ حرکت شکار را به دام انداختند و تنها در یک حالت پس از ۱۰۰۰ حرکت موفق به گرفتن شکار نشدند. به طور متوسط بعد از ۱۷۰ حرکت، شکار به دام می افتد.

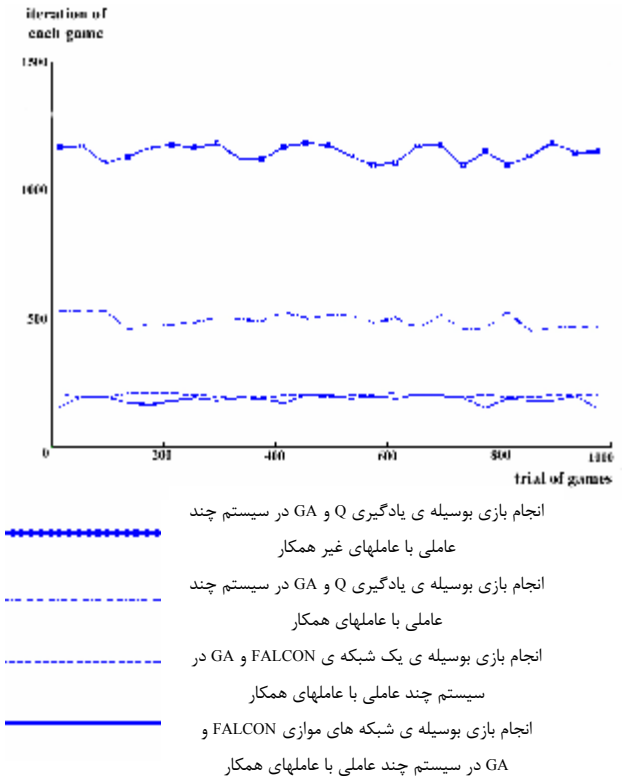
راه حل ارائه شده توانایی حل مسایل مربوط به مسیریابی را در محیط های محلی (ناشناخته و دارای محدوده ی دید کم)، در مقابل محیطهای جهانی (شناخته شده)، دارا می باشد[۵].

استفاده ی همزمان از شبکه های موازی FALCON سبب افزایش سرعت شبیه سازی و رسیدن به هدف نسبت به

حالتهایی گردید که از یک شبکه ی FALCON بزرگ استفاده شد[۸].تعداد حالتها به میزان ۹۰۰ حالت کاهش یافت.

استفاده از الگوریتم ژنتیک و یادگیری تقویتی به تنهایی، سبب کند شدن سرعت یادگیری و جستجوی محیط می گردد که تلفیق آن با شبکه های FALCON سبب افزایش سرعت جستجوی محیط و رسیدن به هدف تا دو برابر در مقایسه با کار قبلی شد[۶].

شکل ۳ روش ارائه شده را با چندین روش پایه مقایسه می کند[۵-۸]. محور افقی تعداد شبیه سازی بازی و محور عمودی تعداد حرکت شکارچی ها را در هر بار اجرا تا رسیدن به هدف نشان می دهد.



شکل۳: مقایسه ی روش ارائه شده با چندین روش پایه

همانگونه که مشاهده می شود، در حالتی که از یادگیری Q و GA در سیستم چند عاملی غیر همکار استفاده میشود، بطور متوسط در ۱۰۰۰ بار اجرای بازی، پس از ۱۱۵۰ حرکت و در حالتهای دیگر بترتیب ۵۰۰، ۲۰۰ و ۱۷۰ حرکت شکار به دام افتاده و بازی خاتمه می یابد.

۶- نتیجه گیری

در این مقاله از ساختارهای عصبی خودسازمانده، به صورت موازی برای توصیف یک محیط چند عاملی مسیریابی استفاده گردید. استفاده از شبکه های با قابلیت یادگیری نظیر FALCON توانایی آموزش سیستم در محیط های مجازی را بالا برده و نتیجه گیری بهتر در محیط های واقعی را فراهم می آورد. علاوه بر این از یادگیری تقویتی Q و الگوریتم ژنتیک برای هوشمندی بیشتر هریک از عاملها و بهبود ارتباط و تعامل بین آنها استفاده شد. استفاده از الگوریتم ژنتیک همراه با پاداش در نظر گرفته شده و به اشتراک گذاشتن w_j^{ck} ها سبب می شود که عوامل در رسیدن به هدف، نمونه ای از کار گروهی و تعامل در یک سیستم را ارائه دهند.

مراجع

[1] Li Jiang, Da-You Liu,"Advances in Multi-Agent Coordination", Proceedings of the Fifth International IEEE Conference on Machine Learning and Cybernetics, Dalian, 13-16 August 2006.

[2] Steven M. LaValle. Planning Algorithms. Cambridge University Press, 2006.

[3] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. Bradford Books, MIT, 1998.

[4] Jing Huang ,Bo Yang ,Da-You Liu , "A Distributed Q-LEARNING Algorithm for Multi-agent team Coordination" , Proceeding of the Fourth International Conference on Machine Learning and Cybernetics ,Dalian ,augest 2005.

[۵] عباس ولی زاده، حبیب رجبی مشهدی، علی پورحسین. "مسیریابی بهینه در حالت سراسری در محیط دینامیکی با کمک الگوریتم ژنتیک" ، اولین کنگره ی مشترک سیستمهای فازی و هوشمند ایران، مشهد، شهریور ۸۶.

[۶] علی پورحسین، محمد باقر نقیبی سیستانی. "تلفیقی از یادگیری تقویتی و الگوریتم ژنتیک جهت ارتباط عاملها در سیستمهای چندعاملی مسیریابی"، دومین کنگره ی مشترک سیستمهای فازی و هوشمند ایران، تهران، آبان ۱۳۸۷.

[7] A. H. Tan, "FALCON: A fusion architecture for learning, cognition, and navigation," in *ProcIJCNN*,Budapest, Hungary, 2004, pp. 3297–3302.

[8] KaoDan Xiao, Ah-Hwee Tan, "Self-Organizing Neural Architectures and Cooperative Learning in a Multiagent Environment," IEEE Transactions On Systems, Man, And Cybernetics—Part B, VOL. 37, NO. 6, December 2007.

[9] L. Panait and S. Luke, "Cooperative multiagent learning: The state of the art," George Mason Univ., Fairfax, VA, Tech. Rep. GMU-CS-TR 2003-1, 2003.

[10] M. Dubreuil, C. Gagne, and M. Parizeau, "Analysis of a master–lave architecture for distributed evolutionary computations," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36 no. 1, pp. 229–235, Feb. 2006.

[11] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 279-292, 1992.

[12] H. Kose, U. Tatlidede, C. Mericli, K. Kaplan, and H. L. Akin, "Q-learning based market-driven multi-agent collaboration in robot soccer," in *Proc. Turkish Symp. Artif. Intell. Neural Netw.*, Izmir, Turkey, 2004, pp. 219–228.

[13] V. Kononen, "Gradient descent for symmetric and asymmetric multiagent reinforcement learning," *Web Intell. Agent Syst.: Int. J. (WIAS)*, vol. 3, no. 1, pp. 17–30, 2005.

[14] E. F. Yang and D. B. Gu, "Multiagent reinforcement learning for multi-robot systems: A survey," Dep. Comput. Sci., Univ. Essex, Colchester, U.K., Tech. Rep. CSM-404, 2004.

[15] L. Bull and T. C. Fogarty, "Evolving cooperative communicating classifier systems," in *Proc. 4th Annu. Conf. Evol. Program.*, 1994, pp. 308–315.