مدلی مبتنی بر آنتروپی و اتوماتاهای یادگیر برای حل بازی‌های تصادفی

بهروز معصومی و محمدرضا میبدی

چکیده: بازی‌های غیر قطعی (تصادفی) به‌عنوان توسعه‌ای از فرآیندهای تصادفی مارکوف با چندین عامل در سیستم‌های چندعامله و مدل‌سازی آنها حائز اهمیت بوده و به‌عنوان چارچوبی مناسب در تحقیقات یادگیری تقویتی چندعامله به‌کار رفته‌اند. در حال حاضر اتوماتاهای یادگیر به‌عنوان ابزاری ارزشمند در طراحی الگوریتم‌های یادگیری چندعامله به‌کار رفته‌اند. در این مقاله مدلی مبتنی بر اتوماتای یادگیر و مفهوم آنتروپی برای حل بازی‌های غیر قطعی و پیداکردن سیاست بهینه در این بازی‌ها ارائه شده است. در مدل پیشنهادی به‌ازای هر عامل در هر حالت از محیط بازی یک اتوماتای یادگیر با ساختار متغیر از نوع S قرار داده شده است که اعمال بهینه را در هر حالت یاد می‌گیرند. تعداد اعمال هر اتوماتا با توجه به همسایگان مجاور هر حالت تعیین شده و ترکیب اعمال اتوماتاها حالت بعدی محیط را انتخاب می‌کند. در مدل پیشنهادی از آنتروپی بردار احتمالات اتوماتای یادگیر حالت جدید برای کمک به پاداش‌دهی اتوماتاها و بهبود یادگیری استفاده شده است. برای بررسی و تحلیل رفتار الگوریتم یادگیری پارامتری به‌نام آنتروپی کلی تعریف گردیده که میزان همگرایی را در الگوریتم یادگیری بیان می‌کند. در نهایت الگوریتمی اصلاح‌یافته با ایجاد تعادل بین جستجو و استناد بر تجربیات پیشنهاد شده است. نتایج آزمایش‌ها نشان می‌دهد الگوریتم ارائه‌شده از کارایی مناسبی از هر دو جنبه هزینه و سرعت رسیدن به راه حل بهینه برخوردار است.

کلید واژه: **آنتروپی، اتوماتاهای یادگیر، بازی‌های تصادفی، سیستم‌های چندعامله.**

۱- مقدمه

یک سیستم چندعامله، در برگیرنده جامعه‌ای از عامل‌های هوشمند و خودمختار است که در یک محیط در کنار یکدیگر در حال کار بوده و سعی در انجام کاری خاص و رسیدن به هدفی مشخص دارند [۱]. سیستم‌های چندعامله برای مدل‌سازی، تحلیل و طراحی سیستم‌هایی که کنترل بین تصمیم‌گیرنده‌های خودمختار به‌صورت توزیع‌شده است مناسب هستند. امروزه در بسیاری از کاربردها و در زمینه‌های مختلف صنعتی، نظامی، مخابراتی و اطلاعاتی از سیستم‌های پیچیده و توزیع‌شده چندعامله استفاده فراوانی می‌شود [۲] و [۳]. برای حل بسیاری از مسایل مهم دنیای واقعی مانند برخی از کاربردهای رباتیک، مسیریابی در شبکه، زمان‌بندی و تصمیم‌گیری اقتصادی نیازمند برنامه‌ریزی در حالت غیر قطعی هستیم. مدل‌های فرآیند تصمیم‌گیری مارکف [۱](MDP)، چارچوب مناسبی برای مدل‌سازی این‌گونه مسایل و یافتن راه حل‌های بهینه برای آنها هستند. برای حل مسایل تصمیم‌گیری مارکوف، یادگیری تقویتی بسیار مورد استفاده قرار گرفته است. بازی‌های غیر قطعی (تصادفی) [۲](SG) به‌عنوان توسعه‌ای از فرآیندهای تصادفی مارکوف با چندین عامل در سیستم‌های چندعامله بسیار حائز اهمیت بوده و به‌عنوان چارچوبی مناسب در تحقیقات یادگیری‌های چندعامله به‌ویژه یادگیری تقویتی چندعامله به‌کار رفته‌اند [۴] و [۵]. معمولاً فرآیند یادگیری تقویتی چندعامله به‌صورت یک بازی تصادفی مدل می‌شود و به آن بازی مارکوفی [۳] می‌گویند که یک فرآیند تصمیم‌گیری مارکوف با چند عامل است [۶].

یادگیری تقویتی چندعامله به سرعت در حال توسعه بوده و روش‌های متنوع و مختلفی را در حوزه‌های رقابتی، همکارانه و ترکیبی در بر می‌گیرد به‌طوری که ارتباطی بین حوزه‌های مختلف علوم نظیر تئوری بازی‌ها، بهینه‌سازی و یادگیری در بازی‌ها برقرار ساخته است. بازی‌های تصادفی مدلی توسعه‌یافته از بازی‌های تکرارشونده هستند که در هر لحظه از زمان، بازی در یک حالت قرار دارد. گذار از حالتی به حالت جدید بر پایه تابع احتمالاتی با توجه به حالت قبلی و تعامل بین عامل‌ها در حالت قبلی انجام می‌گیرد. هر حالت در یک بازی تصادفی می‌تواند به‌صورت یک فرآیند تصمیم‌گیری مارکوف دیده شود و هر بازی تصادفی با یک عامل به‌صورت یک فرآیند تصمیم‌گیری مارکوف است.

بازی‌های تصادفی دارای انواع متفاوتی بوده و از نظر پاداش به دو نوع بازی‌های با مجموع صفر [۴](بازی‌های رقابتی) و بازی‌های با جمع کلی [۵] تقسیم‌بندی شده‌اند. در این بازی‌ها، عامل‌ها اعمالشان را به‌طور هم‌زمان انجام داده و پاداش هر عامل به عمل گروهی انتخاب‌شده توسط همه عامل‌ها بستگی دارد. در بازی‌های رقابتی دونفره ساختار ماتریس به‌گونه‌ای است که پاداش هر عامل منفی مقدار پاداش عامل دیگر است. در بازی‌های با جمع کلی، فرض می‌شود که برای پاداش عامل‌ها محدودیتی مطرح نیست. در حالت خاص بازی‌های مارکوف کلی، در صورتی که پاداش یکسانی برای همه عامل‌ها در نظر گرفته شود آنها را کاملاً همکارانه [۶] گویند و به آن فرآیندهای تصادفی مارکوف چندعامله [۷](MMDP) نیز گفته می‌شود.

در بازی‌های مارکوفی، راه حل به معنای پیداکردن سیاستی برای انتخاب اعمال توسط عامل‌ها در هر حالت است تا بتواند امید ریاضی مجموع کاهش‌یافته پاداش‌ها [۸] را برای همه عامل‌ها بیشینه نماید. در

این مقاله در تاریخ ۱۵ بهمن ماه ۱۳۸۷ دریافت و در تاریخ ۹ اردیبهشت ماه ۱۳۸۹ بازنگری شد.

بهروز معصومی، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی قزوین، قزوین، (email: masoumi@Qiau.ac.ir).

محمدرضا میبدی، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، (email: mmeybodi@aut.ac.ir).

1. Markov Decision Process
2. Stochastic Games
3. Markov Game
4. Zero - Sum Games
5. General - Sum Games
6. Fully Cooperative
7. Multi - Agent MDP
8. Sum of Discounted Expected Rewards

MMDP ها با توجه به این که همه عامل‌ها پاداش یکسان دریافت می‌کنند عامل‌ها بایستی یاد بگیرند تا در مورد سیاست بهینه توافق نمایند. در مقابل در بازی‌های مارکوفی کلی به‌دلیل وجود پاداش‌های متفاوت، پیداکردن راه حل بهینه مشکل بوده و لذا نقاط تعادل[1] (سیاست تعادل) در بازی مورد جستجو قرار می‌گیرند. به‌عبارت دیگر مسأله پیداکردن سیاست بهینه در بازی‌های تصادفی در حوزه تئوری بازی‌ها بوده و راه حل پایه در آنها رسیدن به نقاط تعادل نش[2] می‌باشد.

برای حل بازی‌های اتفاقی اعم از بازی‌های رقابتی و غیر رقابتی الگوریتم‌های یادگیری تقویتی متعددی به‌کار رفته‌اند. این الگوریتم‌ها از نظر خصوصیت ایستایی و پویایی محیط در دو رده قرار می‌گیرند. ما در این مقاله فرض محیط‌های پویا در بازی‌های تصادفی با چند حالت و غیر رقابتی را در نظر می‌گیریم. بیشتر الگوریتم‌های یادگیری چندعامله بر پایه پیداکردن نقاط تعادل عمل می‌کنند. Littman (۱۹۹۴) راه حل minmax را در بازی‌های با مجموع صفر ارائه می‌دهد [۵]. Hu و Wellman (۱۹۹۸) الگوریتمی برای حل بازی‌های با مجموع کلی ارائه دادند [۷]. آنها الگوریتمی به نام Nash - Q را پیشنهاد دادند که تحت شرایط خاص به سیاست تعادل نش همگرا می‌شد [۸]. Greenwald و همکارانش (۲۰۰۳) الگوریتم مبتنی بر تعادل همبستگی[3] به‌جای تعادل نش را مطرح نمودند که در آن با توجه به پیچیدگی زمانی خطی راه حل را برای بازی‌های تصادفی ارائه می‌دهد [۹]. Meiping Song و همکاران (۲۰۰۵) الگوریتمی به نام Pareto - Q را برای بازی‌های با مجموع کلی همکارانه مطرح کردند که از بهینه پارتو بر اساس قوانین اجتماعی استفاده می‌کرد [۱۰] و [۱۱]. بررسی دقیق الگوریتم‌های مختلف و رده‌بندی‌های دیگر این الگوریتم‌ها در [۴] به تفصیل گزارش شده است.

اتوماتاهای یادگیر نیز در حال حاضر به‌عنوان ابزاری ارزشمند در طراحی الگوریتم‌های یادگیری تقویتی بوده و به‌واسطه ویژگی‌هایی که دارند در بسیاری از کاربردهای چندعامله و محیط‌های ناشناخته مناسب هستند [۱۲] و [۱۳]. سادگی ساختار، نیاز به اطلاعات و بازخورد کم از محیط، مناسب‌بودن در سیستم‌های توزیع‌شده و سیستم‌های چندعامله با اطلاعات ناکامل و ارتباطات محدود، انعطاف‌پذیری و قابلیت تحلیل در بیشتر کاربردها از جمله این ویژگی‌ها هستند. در [۱۳] الگوریتم‌های مبتنی بر اتوماتاهای یادگیر برای حل فرآیندهای تصادفی مارکوف با شرایط مختلف و بازی‌های مارکوفی مطرح شده‌اند. بازی‌های اتوماتاها در حالت وجود چند عامل و یک حالت و بازی‌های تصادفی در شرایط وجود چندین عامل و چندین حالت از اهمیت خاصی برخوردار هستند. در [۱۴] روشی مبتنی بر اتوماتاهای یادگیر برای حل فرآیندهای تصادفی مارکوف چندعامله و بازی‌های مارکوفی در شرایط ارگودیک[4] مطرح شده و نشان داده شده است که شبکه‌ای از اتوماتاهای یادگیر[5] قادر به رسیدن به استراتژی‌های تعادل در بازی‌های مارکوفی می‌باشند.

در [۱۵] ترکیب تیمی اتوماتاها برای حل بازی‌های nنفره تیمی استفاده شده و نشان داده شده است در صورت استفاده از الگوریتم L_{RI} توسط تمام اعضای تیم الگوریتم به تعادل نش همگرایی پیدا می‌کند. هدف این مقاله طراحی الگوریتم یادگیری مبتنی بر اتوماتاهای یادگیر برای حل بازی‌های تصادفی با مجموع کلی است که در آنها پاداش کلی عامل‌ها از قبل مشخص نبوده و عامل‌ها از اطلاعات دیگران آگاهی کامل ندارند. در این راستا الگوریتمی با استفاده از اتوماتاهای یادگیر ارائه می‌شود تا با بهره‌گیری از مفهوم آنتروپی برای بهینه‌سازی استراتژی‌ها، الگوریتمی با قابلیت یادگیری بهتر و سریع‌تر برای یافتن خط مشی بهینه ارائه دهد. برای آزمایش مدل ارائه‌شده از محیط بازی‌های Grid Game استفاده شده است. در ادامه مقاله، بخش ۲ به تعریف یادگیری تقویتی چندعامله و تعریف بازی‌های تصادفی و مفهوم راه حل در آنها می‌پردازد. در بخش ۳ مفهوم اتوماتای یادگیر و الگوریتم پیشنهادی و در بخش ۴ نتایج و تحلیل آزمایش‌ها ارائه گردیده‌اند.

۲- تعریف بازی‌های تصادفی و مفهوم راه حل در آنها

در این بخش به‌طور خلاصه تعریف فرآیندهای تصادفی مارکوف و بازی‌های تصادفی مطرح می‌شود. در سیستم‌های چندعامله، الگوریتم‌های یادگیری تقویتی مانند یادگیری Q، با موفقیت در بسیاری از کاربردها مورد استفاده قرار گرفته‌اند [۴]. اکثر الگوریتم‌های یادگیری تقویتی چندعامله، بر پایه روش‌های تک‌عامله بنا نهاده شده‌اند. در حالت یادگیری تک‌عامله که در آن عامل به‌طور مستقل در حال یادگیری باشد، فرآیند یادگیری تقویتی را می‌توان به‌صورت فرآیند تصمیم‌گیری مارکوف تعریف نمود.

۲-۱ فرآیند تصادفی مارکوف (MDP)

به‌طور رسمی فرآیند تصادفی مارکوف را به‌صورت زیر تعریف می‌کنند:

تعریف ۱: فرآیند تصادفی مارکوف به‌صورت چندتایی $\langle S,A,R,T \rangle$ نشان داده می‌شود که در آن S مجموعه متناهی از وضعیت‌ها، A مجموعه عملیات قابل دسترس برای عامل، $T : S \times A \times S \to [۰,۱]$ احتمال انتقال از وضعیت جاری به وضعیت بعدی با انجام عمل a است و $R : S \times A \to \Re$ بیانگر تابع پاداش است که یک مقدار عددی را بر می‌گرداند.

علاوه بر این ما نیاز به مفهوم سیاست داریم که آنچه باید در وضعیت فعلی محیط انجام شود را بیان می‌کند. سیاست نگاشتی از وضعیت به عمل است. ما سیاست را در لحظه t به‌صورت $\pi : S_t \to A_t$ نشان می‌دهیم. سیاستی را ایستا گویند اگر وابسته به زمان نباشد. در یک MDP هدف عامل پیداکردن استراتژی $\pi : S \to A$ است که امید ریاضی مجموع کاهش‌یافته پاداش‌ها را بیشینه نماید. برای هر خط مشی π که عامل می‌تواند دنبال کند، بر روی وضعیت‌ها تابعی به نام تابع ارزیابی[6] به شکل رابطه زیر تعریف می‌شود

$$V(\pi,s) \equiv \sum_{t=۰}^{\infty} \gamma^t E(r_t \mid \pi, s_۰ = s) \qquad (۱)$$

این تابع، نگاشتی از مجموعه وضعیت‌ها به مقدار ارزش آنها می‌باشد. s یک وضعیت خاص، $s_۰$ وضعیت اولیه، r_t پاداش در زمان t و γ ضریب کاهش در محدوده $[۰,۱)$ است. $V(s,\pi)$ ارزش وضعیت s تحت استراتژی π را نشان می‌دهد. بنابراین هدف، یادگیری خط‌مشی بهینه π^* است. پیداکردن سیاست بهینه می‌تواند به‌صورت یک مسأله بهینه‌سازی مطرح شود که می‌تواند با استفاده از الگوریتم‌های برنامه‌ریزی پویا حل شود. در این صورت روشی تقریبی برای تخمین مقادیر بهینه $V(s)$ به‌کار می‌رود که روش Value Iteration نام دارد. راه حل استاندارد از طریق یک روش جستجوی تکرارپذیر است که بر پایه استفاده از معادله Bellman استوار است و به‌صورت (۲) تعریف می‌شود [۱۶]

1. Equilibrium Points
2. Nash Equilibrium
3. Correlated Equilibrium
4. Ergodic
5. Network of Learning Automata
6. Evaluation Function

	L	R
L	۶،۶	۲،۷
R	۲،۷	۰،۰

	T	L
T	۱،−۱	−۱،۱
L	−۱،۱	۱،−۱

شکل ۱: دو نوع بازی ماتریسی به نام‌های تطابق سکه‌ها (سمت چپ) و بازی Chicken (سمت راست).

$$V(\pi^*,s) \equiv \max_a \{r(s,a) + \gamma \sum_{s'} P(s'|s,a) V(s',\pi^*)\} \quad (۲)$$

$r(s,a)$ پاداش به‌دست آمده از انجام عمل a در وضعیت s است، s' وضعیت جدید و $P(s'|s,a)$ احتمال رفتن به s' پس از انجام عمل a در وضعیت s است. این روش هنگامی قابل استفاده است که عامل، توابع پاداش و گذار وضعیت را بشناسد، در غیر این صورت نمی‌توان این روش را به‌کار گرفت. در چنین حالتی از الگوریتمی به‌نام یادگیری Q استفاده می‌شود [۱۷].

۲-۲ بازی‌های تصادفی

بازی‌های تصادفی تعمیم فرآیندهای تصادفی مارکوف به حالت چندعامله بوده و می‌توانند به‌عنوان چارچوبی مناسب برای بررسی و تحقیقات یادگیری چندعامله استفاده شوند. این بازی‌ها توسعه‌ای از بازی‌های ماتریسی با چندین حالت بوده و به آنها بازی‌های مارکوفی نیز گویند. دو مثال از بازی‌های ماتریسی در شکل ۱ آورده شده‌اند. در هر بازی دو بازیکن وجود دارند که به‌صورت سطری و ستونی اعمالشان نشان داده شده است. درایه‌های ماتریس همان پاداش بازیکنان است. در بازی اول که یک بازی با جمع صفر است پاداش بازیکن دوم منفی پاداش اولی است. در حالت کلی ماتریس پاداش برای بازی‌های دونفره با دو ماتریس نشان داده می‌شود. هر حالت در بازی اتفاقی می‌تواند به‌صورت یک بازی ماتریسی بیان شود. در هر حالت از بازی بازیکنان پس از بازی و دریافت پاداش با توجه به عمل گروهی انجام‌شده به حالت دیگری از بازی می‌روند. در یک بازی تصادفی، اعمال بازیکنان هم‌زمان صورت می‌گیرد. به‌طور کلی تعریف رسمی بازی‌های تصادفی به‌صورت زیر است [۱۸]:

تعریف ۲: یک بازی تصادفی به‌فرم چندتایی $\langle n, S, A_{1...n}, T, R_{1...n} \rangle$ بیان می‌شود که n تعداد عامل‌ها، S مجموعه حالات، A_i مجموعه اعمال هر عامل i (در فضای اعمال گروهی $A_1 \times A_2 \times \ldots \times A_n$)، T تابع انتقال $S \times A \times S \to [0,1]$ و r تابع پاداش برای عامل i ام $S \times A \to \Re$ است.

در یک بازی تصادفی کاهش‌پذیر، هدف یک بازیکن بیشینه‌کردن مجموع پاداش کاهش‌یافته با پارامتر کاهش $\gamma \in [0,1)$ است. اگر π_i استراتژی بازیکن i باشد به‌ازای یک وضعیت اولیه s، بازیکن i سعی در بیشینه‌نمودن رابطه زیر دارد

$$V(s,\pi^1,\pi^2,\ldots,\pi^n) \equiv \sum_{t=0}^{\infty} \gamma^t E(r_t | \pi^1,\pi^2,\ldots,\pi^n, s_0 = s) \quad (۳)$$

با نگاه به (۳) و (۱) تشابه با مدل MDP دیده می‌شود منتها در اینجا چندین عامل وجود دارند و رفتن به وضعیت بعدی و دریافت پاداش به عمل گروهی عامل‌ها بستگی دارد. در حالت یادگیری تقویتی چندعامله، بیشینه‌نمودن سودمندی (پاداش) مورد انتظار هر عامل در بازی‌های مجموع کلی ممکن نبوده و هدف پیداکردن سیاست متعادل در بازی‌های مارکوف است. از جمله این سیاست‌ها می‌توان سیاست تعادل نش را نام برد. به‌عبارت دیگر پیداکردن یک سیاست متعادل به‌عنوان یک راه حل برای بازی‌های اتفاقی محسوب می‌شود.

۲-۳ استراتژی‌های تعادل

در بازی‌های با جمع کلی، راه حل مبنا تعادل نش است [۱۸]. نظریه تعادل‌های نش می‌گوید که در هر بازی به‌فرض آن که بازیکنان معقولانه استراتژی‌های خود را انتخاب کرده و به‌دنبال به‌دست آوردن حداکثر بهره (سود) از بازی باشند، حداقل یک استراتژی برای به‌دست آوردن بهترین نتیجه برای هر بازیکن قابل انتخاب است که اگر بازیکن راهکار دیگری به‌غیر از آن را انتخاب کند، نتیجه بهتری به‌دست نخواهد آورد [۱۹].

تعریف ۳: در یک بازی تصادفی Γ، یک تعادل نش به‌صورت چندتایی استراتژی‌های $(\pi_*^1, \pi_*^2, \ldots, \pi_*^n)$ تعریف می‌شود به‌طوری که برای هر $s \in S$ و $i = 1,\ldots,n$ داریم

$$V^i(s,\pi_*^1,\ldots,\pi_*^n) \geq V^i(s,\pi_*^1,\ldots,\pi_*^{i-1},\pi_*^i,\pi_*^{i+1},\ldots,\pi_*^n) \quad \text{for all } \pi^i \in \prod^i \quad (۴)$$

$\prod^i$ مجموعه استراتژی‌های قابل دسترس عامل i است. استراتژی‌هایی که یک تعادل نش را می‌سازند می‌توانند استراتژی‌های ایستا باشند. قضیه زیر نشان می‌دهد حداقل یک تعادل در استراتژی‌های ایستا وجود دارد:

قضیه ۱: (Fink ۱۹۶۴) هر بازی اتفاقی کاهش‌پذیر n نفره حداقل یک نقطه تعادل نش در استراتژی‌های ایستا را داراست [۲۰].

۲-۴ حل بازی‌های تصادفی

در بازی‌های اتفاقی بر خلاف MDPها یک راه حل بهینه‌ای که مستقل از عامل‌های دیگر باشد محتمل نیست. لذا نسبت به حالت تک‌عامله مسأله مشکل‌تر است. همچنین ممکن است که چندین نقطه تعادل وجود داشته و هماهنگی عامل‌ها برای توافق کار مشکلی است. مفهوم راه حل در بازی‌های اتفاقی به معنای پیداکردن نقطه تعادل منحصر به‌فرد می‌باشد. الگوریتم‌های مختلفی برای حل این بازی‌ها پیشنهاد شده‌اند که در یک رده‌بندی می‌توان به روش‌هایی که از تئوری بازی‌ها استفاده می‌کنند و روش‌های مبتنی بر یادگیری تقویتی اشاره نمود. در روش‌های مبتنی بر تئوری بازی‌ها، نیاز به مدل محیط داشته و توابع T و R نیز بایستی مشخص باشند. هدف این الگوریتم‌ها محاسبه مقدار تعادل بازی (پاداش مورد انتظار کاهش‌یافته برای هر عامل) است، لذا نیاز به فرضیات قوی‌تری از رفتار عامل‌ها دارند. در مقابل، الگوریتم‌های مبتنی بر یادگیری تقویتی چندعامله فرض می‌کنند محیط ناشناخته بوده و مشاهدات T و R از طریق تجربی قابل مشاهده هستند. هدف این الگوریتم‌ها پیداکردن سیاست تعادل بوده و معمولاً نیازهای کمتری را درباره رفتار عامل‌های دیگر دارند. یک چارچوب کلی برای یادگیری Q چندعامله در شکل ۲ نشان داده شده است. در قالب کلی، الگوریتم‌های مختلف یک ورودی تابع انتخاب تعادل را دریافت می‌کنند تا بتوانند با توجه به بردار ماتریسی نظیر $Q = (Q_1,\ldots,Q_n)$ تابع ارزش V را محاسبه نمایند. Hu و Wellman از تابع با نام Nash-Q برای محاسبه مقدار V طبق (۵) استفاده نمودند [۸]. Qio و همکاران نیز از مکانیزم چانه‌زنی[1] (۶) [۲۱] و Meiping Song نیز از الگوریتم Pareto-Q استفاده کردند [۱۱]

$$V_i(s) \in Nash_i(Q_1(s),\ldots,Q_n(s))$$
$$Nash_i(Q_i(s)) = \pi^1(s)\ldots\pi^n(s).Q_i(s) \quad (۵)$$

$$V_i(s) \in NBS_i(Q_1(s),\ldots,Q_n(s))$$
$$NBS_i(Q_i(s)) = \max_{\vec{a}}(Q_1(s,\vec{a}) \times \ldots, Q_n(s,\vec{a})) \quad (۶)$$

1. Nash Bargaining

Multi-agent Q-learning (Stochastic Game, α, γ, M)

Inputs: discount factor γ, learning rate α, M total training time

Output: state-value functions, action-value V_i^* function Q_i^*

Initialize: $s, a_1 \ldots a_N, Q_1 \ldots Q_N$

1. for k = 1 to M do
2. simulate actions $a_1,\ldots,a_n$ in state s
3. observe rewards $R_1,\ldots,R_n$ and next state s'
4. for i = 1 to N do
   (a) compute $V_i(S')$
   (b) $Q_i(s,\vec{a}) \leftarrow (1-\alpha)Q_i(s,\vec{a}) + \alpha[(1-\gamma)R_i + \gamma Vi(s')]$
5. agents choose action $a_1',\ldots,a_N'$
6. $s = s', a_1 = a_1', \ldots, a_N = a_N'$
7. decay α

شکل ۲: الگوریتم یادگیری Q چندعامله.

با توجه به پیچیدگی بالای محاسبات در هر یک از الگوریتم‌های محاسبه تعادل نش در بازی‌های اتفاقی، به نظر می‌رسد استفاده از الگوریتم‌های مبتنی بر اتوماتای یادگیر با توجه به سادگی و پیچیدگی محاسباتی کم بتوانند کارایی مطلوبی را از خود ارائه دهند. با توجه به این موضوع در بخش بعدی مدلی مبتنی بر اتوماتای یادگیر برای حل بازی‌های مارکوفی پیشنهاد و و نتایج آن ارائه گردیده‌اند.

۳- اتوماتاهای یادگیر و مدل پیشنهادی

با توجه به این که مدل‌های اتوماتاهای یادگیر ارتباطی نزدیک با بحث یادگیری تقویتی چندعامله دارند، در این بخش اتوماتاهای یادگیر به‌عنوان مدلی از یادگیری تقویتی به اختصار شرح داده می‌شود.

۳-۱ اتوماتاهای یادگیر

اتوماتای یادگیر، ماشینی است که می‌تواند تعدادی متناهی عمل را انجام دهد. هر عمل انتخاب‌شده توسط یک محیط احتمالی ارزیابی شده و نتیجه ارزیابی در قالب سیگنالی مثبت یا منفی به اتوماتا داده می‌شود و اتوماتا از این پاسخ در انتخاب عمل بعدی تأثیر می‌گیرد. هدف نهایی این است که اتوماتا یاد بگیرد تا از بین اعمال خود، بهترین عمل را انتخاب کند [۲۲]. بهترین عمل، عملی است که احتمال دریافت پاداش از محیط را به حداکثر برساند. کارکرد اتوماتای یادگیر در تعامل با محیط، در شکل ۳ مشاهده می‌شود.

محیط را می‌توان توسط سه‌تایی $E = \langle \alpha, \beta, c \rangle$ نشان داد که در آن $\alpha \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ مجموعه ورودی‌ها، $\beta \equiv \{\beta_1, \beta_2, \ldots, \beta_m\}$ مجموعه خروجی‌ها و $c \equiv \{c_1, c_2, \ldots, c_r\}$ مجموعه احتمال‌های جریمه می‌باشد. هرگاه β مجموعه‌ای دوعضوی باشد، محیط از نوع P است. در چنین محیطی $\beta_1 = 1$ به‌عنوان جریمه و $\beta_2 = 0$ به‌عنوان پاداش در نظر گرفته می‌شود. در محیط از نوع Q، $\beta(n)$ می‌تواند به‌طور گسسته یک مقدار از مقادیر محدود در فاصله [۰،۱] را اختیار کند و در محیط از نوع S، $\beta(n)$ متغیری تصادفی در فاصله [۰،۱] است. c_i احتمال این که عمل α_i نتیجه نامطلوب داشته باشد است. در محیط ایستا، مقادیر c_i بدون تغییر می‌مانند، حال آن که در محیط غیر ایستا این مقادیر در طی زمان تغییر می‌کنند. اتوماتاهای یادگیر می‌تواند به دو دسته اتوماتای یادگیر با ساختار ثابت اتوماتای یادگیر با ساختار متغیر (VSLA)[1] دسته‌بندی شود [۲۲].

شکل ۳: ارتباط بین اتوماتای یادگیر و محیط.

اتوماتای یادگیر با ساختار متغیر را توسط چهارتایی $\{\alpha, \beta, p, T\}$ می‌توان نشان داد که $\alpha \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ مجموعه عمل‌ها (اقدام‌های) اتوماتا، $\beta \equiv \{\beta_1, \ldots, \beta_r\}$ ورودی‌های اتوماتا، $p = \{p_1, \ldots, p_r\}$ بردار احتمال انتخاب هر یک از عمل‌ها و $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ الگوریتم یادگیری می‌باشد. الگوریتم زیر بر اساس (۷) و (۸) یک نمونه از الگوریتم‌های یادگیری خطی است. فرض می‌کنیم عمل α_i در مرحله nام انتخاب شود.

– پاسخ مطلوب از محیط

$$\begin{aligned} p_i(n+1) &= p_i(n) + a(1 - p_i(n)) \\ p_j(n+1) &= (1-a)p_j(n) \quad \forall j \quad , \quad j \neq i \end{aligned} \tag{۷}$$

– پاسخ نامطلوب از محیط

$$\begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad , \quad j \neq i \end{aligned} \tag{۸}$$

در (۷) و (۸) a بیانگر پارامتر پاداش و b پارامتر جریمه می‌باشند. با توجه به مقادیر مختلف برای a و b سه حالت را می‌توان در نظر گرفت: اگر a و b با هم برابر باشند، الگوریتم را L_{RP}، هنگامی که b از a خیلی کوچک‌تر باشد، الگوریتم را $L_{R\varepsilon P}$ و اگر b مساوی صفر باشد آن را L_{RI} می‌نامیم. شمای S-L_{RP} برای مدل‌های Q و S در (۹) بیان شده است. در مدل S، اگر عمل α_i در مرحله nام انتخاب شود در این صورت داریم

$$\begin{aligned} p_i(n+1) &= p_i(n) + a(1-\beta_i(n))(1-p_i(n)) \\ &\quad - b\beta_i(n)p_i(n) \\ p_j(n+1) &= p_j(n) - a(1-\beta_i(n))p_j(n) \\ &\quad + b\beta_i(n)(\frac{1}{r-1} - p_j(n)) \quad \forall j \quad , \quad j \neq i \end{aligned} \tag{۹}$$

که در آن r تعداد اعمال ممکن، a و b همان پارامترهای پاداش و جریمه هستند. برای اطلاعات بیشتر درباره اتوماتاهای یادگیر می‌توان به [۲۳] مراجعه نمود.

۳-۲ بازی اتوماتاها

در بازی اتوماتاها [۲۲] و [۲۴] تعدادی اتوماتا بدون آن که دانش کاملی از یکدیگر داشته باشند در محیط عمل می‌کنند. از نقطه نظر بازی‌ها بهتر است اعمال اتوماتا را به‌عنوان استراتژی آنها و ورودیشان را به‌عنوان نتایج در نظر بگیریم. با انجام نتایج هر اتوماتا یک ورودی را از محیط دریافت می‌کند. به‌دلیل طبیعت تصادفی‌بودن محیط‌ها، انتخاب اعمال توسط هر اتوماتا در یک بازی تعیین‌کننده احتمال نتایجشان است. همچنین فرض شده اتوماتای شرکت‌کننده در بازی، احتمالات نتایج را به‌عنوان دانش اولیه در اختیار ندارد. علاوه بر این فقط اطلاعاتی که هر اتوماتا در جریان بازی از ورودیش به‌دست می‌آورد به‌عنوان تابعی از اعمالش در بازی‌های موفق محسوب می‌شود. بنابراین در بحث تئوری بازی‌ها، در بازی‌های اتوماتا بازیکنان اطلاعات اولیه در ارتباط با بازی نظیر تعداد بازیکنان، اعمال احتمالی ایشان و عنصر ماتریس‌های بازی را در اختیار ندارند.

1. Variable Structure Learning Automata

در بازی‌های با مجموع صفر الگوریتم L_{RI} به نقطه تعادل همگرا می‌شود اگر در استراتژی‌های محض باشد در حالی که در الگوریتم $L_{R\varepsilon P}$ به استراتژی‌های تعادل مخلوط همگرایی دارد. در [۱۵] نشان داده شده است در بازی‌های با مجموع غیر صفر وقتی که اتوماتاهای یادگیر از الگوریتم L_{RI} استفاده می‌نمایند و بازی دارای یک نقطه تعادل محض باشد، همگرایی تضمین شده است.

۳-۳ مفهوم آنتروپی در یادگیری

آنتروپی مفهومی اساسی در ترمودینامیک است که بیانگر درجه بی‌نظمی در یک سیستم ترمودینامیکی است. این مفهوم توسط شانون در حوزه تئوری اطلاعات تحت عنوان آنتروپی اطلاعاتی[1] مطرح گردید. آنتروپی اطلاعاتی حالتی از آنتروپی متغیرهای تصادفی است که درجه عدم قطعیت را بیان می‌کند و به‌صورت زیر تعریف می‌شود [۲۵] و [۲۶]

$$H(X) = -\sum_{x_i \in \Theta} P(x_i)\log(P(x_i)) \qquad (۱۰)$$

X متغیر تصادفی با مجموعه مقادیر Θ و تابع تجمعی احتمال $P(x_i)$ است. آنتروپی متغیر تصادفی بسته به کاربردهای مختلف، معانی مختلف وابسته به مسأله دارد. در این مقاله آنتروپی در حوزه یادگیری در فضای گسسته به‌عنوان معیاری جدید در فرآیند یادگیری معرفی می‌شود. در یک فضای حالات گسسته، هر حالت s_i با در برگیرنده مجموعه‌ای از مقادیر احتمالات تصمیم است. هر حالت s_i متناظر با متغیر تصادفی D_i است. در فرآیند یادگیری آنتروپی D_i بیانگر عدم قطعیت تصمیم در حالت s_i است. مقادیر بیشتر آنتروپی عدم قطعیت بیشتر تصمیم‌گیرنده را مشخص می‌کند.

برای بررسی رفتار یادگیرنده، دو پارامتر آنتروپی استراتژی محلی و آنتروپی کلی را به‌عنوان پارامتر مستقل از مسأله به‌صورت زیر تعریف می‌کنیم: آنتروپی استراتژی محلی بیانگر عدم قطعیت تصمیم در یک حالت خاص بوده و به‌صورت زیر تعریف می‌شود

$$H_{local}^{s_i} = -\sum_{j=1}^{N} p_j(s_i)\log(p_j(s_i)) \qquad (۱۱)$$

که در آن N تعداد عناصر در مجموعه اعمال و $P_j(s_i)$ احتمال انتخاب j امین عمل تحت حالت s_i است. آنتروپی کلی محیط را میانگین تمام آنتروپی‌های استراتژی‌های محلی به‌صورت زیر تعریف می‌کنیم

$$H_{global}^{s_i} = -\frac{1}{M}\sum_{i=1}^{M} H_{local}^{s_i} = -\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N} p_j(s_i)\log(p_j(s_i)) \qquad (۱۲)$$

که در آن M تعداد عناصر در مجموعه حالات محیط می‌باشد. این پارامتر یک معیار مستقل از مسأله برای فرآیند یادگیری است به‌طوری که درجه همگرایی استراتژی را بیان می‌کند. برای بررسی رفتار الگوریتم این معیار در هر مرحله نمایش داده می‌شود. در یادگیری، بهینه‌سازی استراتژی فرآیندی است که احتمال مقادیر انتخاب عمل بهینه را افزایش می‌دهد به این معنا که استراتژی تصادفی اولیه به استراتژی بهینه تعریف‌شده همگرا می‌گردد. در نتیجه در فرآیند یادگیری به مرور عدم قطعیت استراتژی یادگیری و بالطبع آن آنتروپی کاهش می‌یابند. لذا مفهوم آنتروپی درجه همگرایی را در فرایند یادگیری بیان می‌نماید.

1. Information Entropy

۳-۴ معرفی مدل پیشنهادی

در این بخش روش پیشنهادی حل بازی‌های تصادفی با استفاده از اتوماتای یادگیر ارائه می‌شود که آن را MLA[2] می‌نامیم. می‌دانیم در یک بازی تصادفی عمل انتخاب‌شده در هر حالت با توجه به نتیجه اعمال گروهی عامل‌های مستقل در سیستم است. در مدل پیشنهادی، فرض می‌گردد که محیط شامل مجموعه‌ای ازحالات گسسته است. به ازای هر حالت S در محیط بازی و به ازای هر عامل نظیر K، یک اتوماتای یادگیر با ساختار متغیر و مدل S نظیر LA_k^s قرار داده می‌شود. حالت‌های مجاور با هر حالت، همسایگان آن در نظر گرفته می‌شود. با توجه به تعداد همسایگان هر اتوماتا، تعداد اعمال اتوماتا در هر حالت در نظر گرفته می‌شود. هر عمل گروهی ناشی از اعمال اتوماتاهای هر حالت متناظر با انتقال به یکی از حالات همسایه است. هر عامل برای تعیین حالت بعدی خود از اتوماتای یادگیر با توجه به عمل انتخابی عامل‌های محیط کمک می‌گیرد. عامل‌های یادگیرنده از یک حالت شروع‌کننده آغاز و به حالت پایانی یا هدف می‌رسند. در ابتدا اتوماتاهای یادگیر تمام عمل‌های خود را با احتمالی یکسان متناسب با تعداد اعمال آنها، انتخاب می‌کنند. در صورتی که عمل اتوماتاها منجر به ورود عامل به حالت نهایی (هدف) شود اتوماتای یادگیر پاداش کامل می‌گیرد، در غیر این صورت از آنتروپی بردار احتمال اتوماتای یادگیر حالت بعد برای تعیین پاداش یا جریمه میانی استفاده می‌شود.

در روش پیشنهادی، آنتروپی بردار احتمال میزان عدم قطعیت اتوماتای یادگیر حالت بعد را در انتخاب عمل خود نشان می‌دهد. هرچه آنتروپی بیشتر باشد میزان عدم قطعیت بیشتر است. عدم قطعیت بالا در بردار احتمال اتوماتای یادگیر به این معنی است که این اتوماتا دارای اطلاعات مفیدی برای رسیدن به هدف نبوده و عمل‌های خود را به‌صورت تصادفی انتخاب می‌کند (جستجو[3]). ولی چنانچه عدم قطعیت کم باشد به این معنی است که اتوماتا با احتمال بالایی یکی از اعمال خود را انتخاب می‌کند و دارای اطلاعات مفیدی برای رسیدن به هدف می‌باشد و از این اطلاعات بهره‌برداری می‌نماید[4]. فرض کنید بردار احتمال اعمال اتوماتای یادگیر i با r عمل در حالت s به‌صورت $P_i^s = \{P_i^s(1), P_i^s(2), \ldots, P_i^s(r)\}$ باشد. آنتروپی این بردار احتمال برای حالت s به شکل زیر تعیین می‌شود

$$H_i^s = \sum_{j=1}^{r} p_i^s(j)\log(p_i^s(j)) \qquad (۱۳)$$

آنتروپی زمانی بیشترین مقدار را خواهد داشت که تمام اعمال اتوماتای یادگیر احتمالی یکسان داشته باشند $P_{equal} = \{p_1 = \ldots = p_r = 1/r\}$ و زمانی کمترین مقدار (برابر با ۰) را خواهد داشت که بردار احتمالات یک بردار یکه باشد. برای این که مقدار آنتروپی را به مقداری بین ۰ و ۱ تبدیل کنیم تا به‌عنوان بردار تقویتی در اتوماتای ساختار متغیر مدل S قابل استفاده باشد از (۱۴) استفاده می‌شود. فرض کنید عامل i در حالت s و اتوماتای یادگیر متناظرش یعنی LA_i^s به حالت s' می‌رود در این صورت بردار تقویتی اتوماتای یادگیر β_i^s به‌صورت زیر محاسبه می‌شود

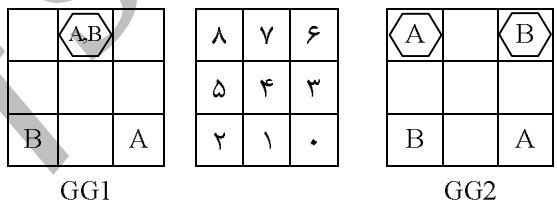
$$\beta_i^s = \frac{H_i^{s'}}{\max(H_i^{s'})^K} \qquad (۱۴)$$

2. Multi Learning Automata
3. Exploration
4. Exploitation

Multi Leaning Automata (Stochastic Game SG, *a*, *b*, *K*, *M*)

Inputs: *a*, *b*: reward and penalty parameter for each *LA*,
K: exploration Parameter, *M*: total training time
Initialize: In each state, a Learning Automaton of type *S* for each agent is placed. The set of actions of this *LA* is the set of permissible movements to other states.
The following steps are taken
1. for episode = 1 to *M* do
2. s = Start State
3. while not done
4. for each agent *i* do concurrently
5. Activate LA_i^s (*LA* residing in current state(s) of agent *i*)
6. Choose action a_i^s in state s
7. Observe Rewards r_i^s and Next State s'
8. Compute cumulative-reward1, cumulative-reward2
9. Compute β_i^s signal base on EQ (15)
10. Train LA_i^s residing in state s according β_i^s
11. end for
12. $s = s'$
13. end while
14. end for

شکل ۴: الگوریتم MLA.



شکل ۶: نمایش مختصات بازی دو نوع بازی Grid World.

که در آن $\max(H_i^{s'})$ بیشترین مقدار آنتروپی برای عامل i در حالت s' است و K پارامتر روش می‌باشد. میزان این پارامتر بر حسب کاربرد و نیاز مسأله باید به دقت تعیین شود. مقادیر بالای این پارامتر باعث می‌شود که β_i^s ها مقادیر کمی داشته باشند و اتوماتاهای یادگیر بیش از حد پاداش ببینند. این به معنی جستجوی بیشتر در محیط است. هرچه میزان K کمتر باشد β_i^s ها مقادیر بیشتری خواهند داشت. این امر باعث می‌شود که اتوماتاها بیشتر جریمه شوند. این مسئله باعث می‌شود که حتی حالت‌های مطلوب نیز پاداش لازم را نگیرند. فرض کنید که عامل i در حالت s باشد و اتوماتای یادگیر آن یعنی عامل را به حالت s' هدایت کند. در این صورت سیگنال تقویتی با توجه به (۱۵) تعیین می‌شود. الگوریتم پیشنهادی در شکل ۴ نشان داده شده است

$$\beta_i^s = \begin{cases} 0 & \text{if } s' = \text{EndState} \\ \dfrac{H_i^{s'}}{\max(H_i^{s'})^K} & \text{otherwise} \end{cases} \qquad (۱۵)$$

تنظیم پارامتر K: در روش پیشنهادی مقدار پارامتر K ثابت در نظر گرفته شده بود. در روش بهبودیافته که مشابه با روش بولتزمن در یادگیری Q عمل می‌کند پارامتر K برای تنظیم حالت جستجو یا استناد بر تجربیات قبلی استفاده شده است. با تنظیم اولیه پارامتر K در ابتدا عامل‌ها جستجو را انجام داده و به مرور زمان با توجه به تغییر پارامتر امکان بهره‌برداری از دانسته‌های خود را پیدا می‌کنند. الگوریتم بهبودیافته در شکل ۵ نشان داده شده است. یادگیری موقعی پایان می‌یابد که عامل به هدف برسد یا تعداد مراحل پایان یابد.

Modified MLA (Stochastic Game SG, *a*, *b*, *M*)

Inputs: *a*, *b*: reward and penalty parameter for each *LA*,
M: total training time
Initialize: In each state, a Learning Automaton of type *S* for each agent is placed. The set of actions of this *LA* is the set of permissible movements to other states. Coefficient *c* is an increment parameter.
The following steps are taken
1. *K* is set to initial value
2. for episode = 1 to *M* do
3. s = Start State
4. while not done
5. for each agent *i* do concurrently
6. Activate LA_i^s (*LA* residing in current state(s) of agent *i*)
7. Choose action a_i^s in state s
8. Observe Rewards r_i^s and Next State s'
9. Compute cumulative-reward1, cumulative-reward2
10. Compute β_i^s signal base on EQ (15)
11. Train LA_i^s residing in state s according β_i^s
12. end for
13. $s = s'$
14. end while
15. Increment *K* with coefficient *c*
16. end for

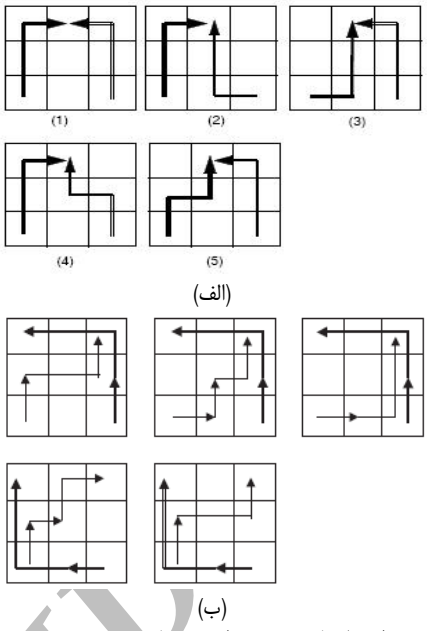
شکل ۵: الگوریتم MLA تغییریافته.

٤- آزمایش‌های انجام‌گرفته در محیط بازی‌های GRID GAME و نتایج آنها

چارچوب عمومی به‌کار رفته در سیستم‌های چندعامله شامل بازی‌های تصادفی و ماتریسی است. در یادگیری چندعامله، یکی از انواع بازی‌های تصادفی غیر رقابتی برای آزمایش، بازی‌های Grid Game است که توسط Hu و Wellman ارائه شده و بسیار مورد استفاده قرار گرفته است [۸]. این بازی دارای انواع متفاوتی است که همه آنها یک بازی دونفری از نوع جمع کلی می‌باشند.

٤-١ دو نوع بازی Grid Game

دو نوع بازی Grid Game در شکل ۶ نشان داده شده‌اند. بازی نوع اول (GG۱) نسخه‌ای از بازی Chicken است که دارای چندین حالت بوده و در آن فقط یک هدف وجود دارد. بازی دوم (GG۲) یک بازی هماهنگی چندحالتی است که دو عامل و دو نوع هدف وجود دارند. در بازی ۱، دو عامل از دو گوشه یک صفحه شروع کرده و سعی می‌کنند با کمترین تعداد حرکت به هدف برسند. اعمال بازیکنان به‌طور هم‌زمان انجام می‌گیرند. اعمال بازیکنان یعنی A۱ و A۲ به‌صورت چهار عمل (بالا، پایین، چپ، راست) تعریف می‌شود. مجموعه فضای حالات به‌صورت $S = \{s/s = (l_1, l_2)\}$ تعریف می‌شود که هر حالت $s = (l_1, l_2)$ مختصات عامل‌های ۱ و ۲ را نشان می‌دهد. عامل‌ها نمی‌توانند در یک مختصات یکسان قرار گیرند. اگر دو عامل سعی در حرکت به یک مربع یکسان داشته باشند حرکت هر دو با شکست مواجه می‌شود. تعداد حالات ممکن در نوع اول برابر $8 \times 7 = 56$ و نوع دوم برابر $72 = 8 - 7 - 57$ است. اگر عامل‌ها به دو خانه مختلف غیر هدف بروند هر دو پاداش صفر را دریافت می‌کنند. اگر هر یک از عامل‌ها به هدف برسد ۱۰۰ واحد پاداش می‌گیرد و در صورت رفتن به خانه یکسان (تداخل) هر دو یک واحد جریمه می‌شوند و در موقعیت قبلی باقی می‌مانند. در هر دو بازی گذار از



شکل ۷: نمونه راه حل‌های بهینه برای دو بازی Grid World، (الف) GG۱ و (ب) GG۲.

جدول ۱: مقایسه کارایی الگوریتم‌های یادگیری.

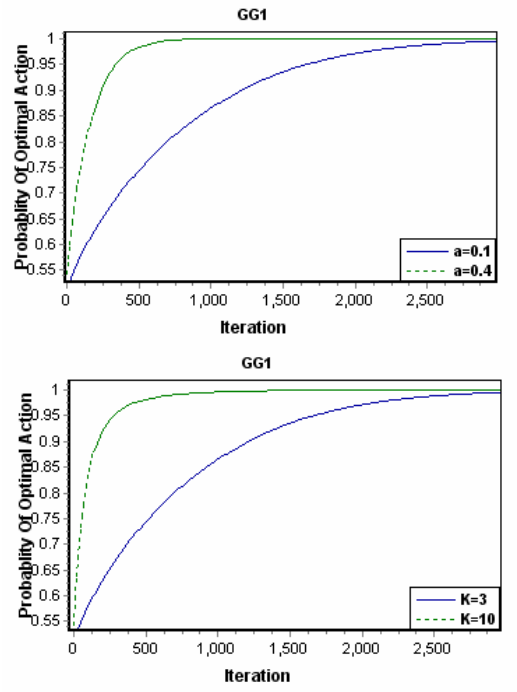
استراتژی یادگیری		نتیجه یادگیری
عامل ۱	عامل ۲	درصد رسیدن به مسیر بهینه
Single Q-learning	Single Q-learning	۲۳
First Nash	Single Q-learning	۴۵
First Pareto	Single Q-learning	۵۴
First Nash	First Nash	۱۰۰
First Pareto	First Pareto	۱۰۰
MLA	**MLA**	**۱۰۰**

حالتی به حالت دیگر با قطعیت انجام می‌شود، یعنی حالت جاری و عمل مشترک عامل‌ها منحصراً حالت بعدی را تعیین می‌کنند.

در این بازی فرض می‌شود عامل‌ها از موقعیت هدف در ابتدای بازی و همچنین از پاداش کلی یکدیگر اطلاع ندارند. عامل‌ها اعمالشان را هم‌زمان انتخاب نموده و فقط می‌توانند از اعمال قبلی عامل‌های دیگر و حالت فعلی (موقعیت مشترک هر دو عامل) آگاهی داشته باشند.

یک مسیر دنباله‌ای از اعمال از نقطه شروع تا پایان را نشان می‌دهد. در اصطلاح بازی چنین مسیری را استراتژی یا سیاست می‌نامند. کوتاه‌ترین مسیری که با مسیر عامل دیگر تداخل نداشته باشد را استراتژی بهینه می‌نامند که یک موازنه نش را می‌سازند زیرا هر مسیر (استراتژی) بهترین پاسخ در قبال دیگری است. استراتژی‌های ایستا به‌صورت $\pi^i = (\pi^i(s_1),\ldots,\pi^i(s_m))$ تعریف می‌شود که برای هر حالت s_i یک توزیع احتمالاتی را اختصاص می‌دهد. $\pi^i(s_j)$ را استراتژی محض گویند هرگاه احتمال ۱ را به اعمال بدهد. یک موازنه نش شامل زوج استراتژی‌های ایستا به‌صورت (π_*^1,π_*^2) است که هر استراتژی بهترین پاسخ به دیگری باشد.

برای حل مسأله با توجه به روش ارائه‌شده به‌ازای هر حالت (l_1,l_2) برای هر عامل یک اتوماتای یادگیر در نظر گرفته شده است. هر عامل با توجه به عمل مشترک گروهی از یک حالت به حالت جدید می‌رود (در صورت عدم برخورد دو عامل با یکدیگر و با توجه به عدم برخورد با موانع



شکل ۸: نمودار تغییرات احتمال انتخاب عمل بهینه برای اتوماتای یادگیر ۱ در خانه شروع ($state = (۰۲)$) با توجه به مقادیر مختلف پارامتر یادگیری و مقادیر متفاوت K.

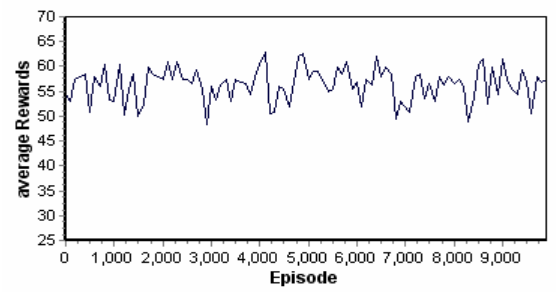
و با توجه به (۱۵) مقدار β (پارامتر پاداش یا جریمه) تعیین شده و اتوماتای آن حالت به‌روز می‌گردد. عامل‌ها اعمالشان را هم‌زمان انجام داده و هر دو قابلیت مشاهده حالت جدید، پاداش آنی به‌دست آمده و عمل انجام‌شده توسط دیگری را دارند. شکل ۷ راه حل‌های بهینه را برای دو بازی نشان می‌دهد.

۴-۲ نتایج آزمایش‌ها و تحلیل آنها

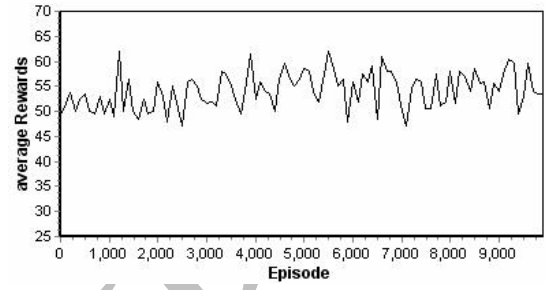
برای بررسی رفتار الگوریتم پیشنهادی و ارزیابی آن چند سری آزمایش انجام گرفته است. بازی مارکوفی در نظر گرفته شده همان دو نوع بازی Grid Game مطرح‌شده در بخش قبلی است.

آزمایش ۱: بررسی همگرایی اتوماتای یادگیر. این آزمایش برای بررسی همگرایی اتوماتا انجام گرفته است. برای این منظور تغییرات احتمال انتخاب عمل حرکت بهینه در خانه شروع برای عامل ۱ در دو حالت ثابت‌گرفتن پارامتر K و ثابت نگه داشتن پارامتر یادگیری مشاهده می‌شود. در حالت اول پارامترهای یادگیری a برابر $\{۰/۱,۰/۴\}$ و پارامتر b برابر صفر و با فرض ($K = ۳$ ثابت) و حالت دوم پارامتر a ثابت و پارامتر $k = \{۱۰,۳\}$ بررسی شده است. شکل ۸ نمودار تغییرات را برای این عامل نشان می‌دهد. با توجه به نمودار می‌بینیم هرچه ضریب پارامتر یادگیری افزایش یابد احتمال انجام عمل نیز افزایش یافته و در نتیجه همگرایی نیز سریع‌تر خواهد شد و از طرفی هرچه پارامتر جستجو بیشتر شود سرعت همگرایی بیشتر می‌شود.

آزمایش ۲: مقایسه کارایی الگوریتم پیشنهادی با سایر الگوریتم‌ها. برای مقایسه الگوریتم با الگوریتم‌های مشابه از الگوریتم Nash - Q که یکی از مهم‌ترین و عمومی‌ترین الگوریتم‌ها در این زمینه می‌باشد، استفاده شده است. در بخش اول ابتدا مقایسه‌ای بین الگوریتم پیشنهادی و سایر الگوریتم‌ها از نظر درصد رسیدن به مسیر بهینه انجام می‌گیرد. جدول ۱ مقایسه نتایج به‌دست آمده را از نظر میزان همگرایی نشان می‌دهد. آزمایش‌های جدول ۱، ۴۰۰۰۰ مرحله و برای استحکام بیشتر ۵۰ مرتبه اجرا در نظر گرفته شده است.

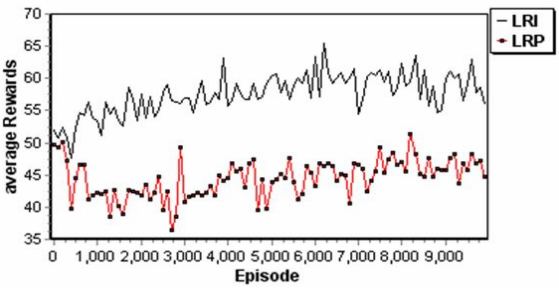


(الف)

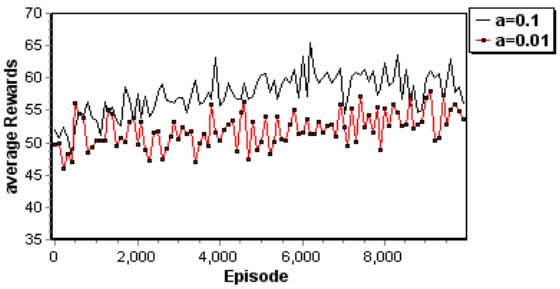


(ب)

شکل ۹: مقایسه روش پیشنهادی با الگوریتم Nash - Q از نظر میانگین پاداش دریافتی برای عامل ۱ در بازی GG۱، (الف) الگوریتم Nash - Q و (ب) الگوریتم پیشنهادی.



(الف)

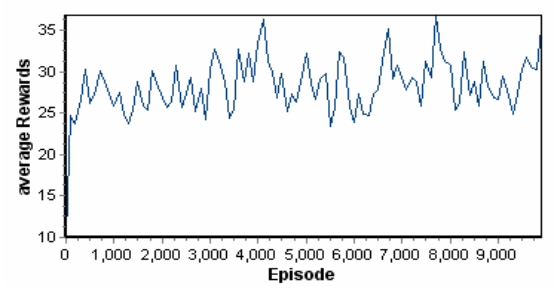


(ب)

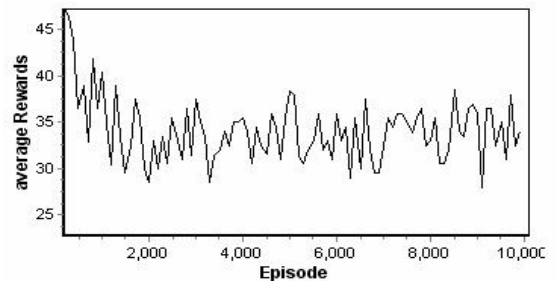
شکل ۱۱: بررسی رفتار روش پیشنهادی با توجه به (الف) الگوریتم یادگیری و (ب) پارامتر یادگیری برای بازی GG۱.

در بخش دوم این آزمایش، از میانگین پاداش به‌دست آمده در هر اپیزود برای مقایسه استفاده شده است. شکل‌های ۹ و ۱۰ میانگین پاداش را در الگوریتم Nash - Q و MLA برای هر دو نوع بازی نشان می‌دهند. آزمایش‌ها به‌خاطر استحکام در نتایج ۲۰۰ بار تکرار شده و در هر بار ۱۰۰۰۰ اپیزود آزمایش شده است. در هر اپیزود جدید هر عامل به‌طور تصادفی یک مختصات جدید (به‌جز هدف) را خواهد گرفت. پارامتر یادگیری برای روش پیشنهادی $a = ۰/۱$ و الگوریتم L_{RI} بوده و برای روش NASH - Q پارامتر یادگیری همان ۰/۱ در نظر گرفته شده است. همان‌طور که در شکل‌های ۹ و ۱۰ دیده می‌شود رفتارهای مشابهی از الگوریتم پیشنهادی و الگوریتم Nash - Q دیده می‌شود.

آزمایش ۳: بررسی نقش پارامتر یادگیری در کارایی الگوریتم

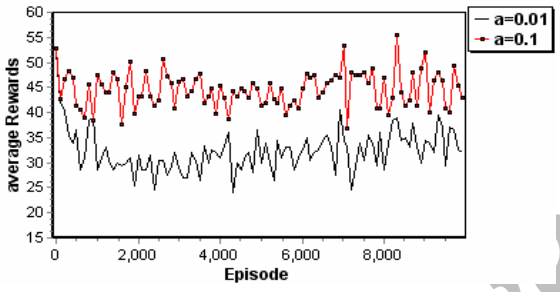


(الف)

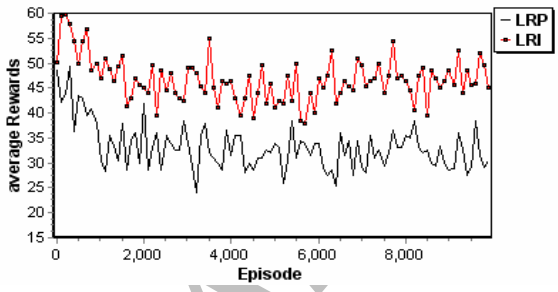


(ب)

شکل ۱۰: مقایسه روش پیشنهادی با الگوریتم Nash - Q از نظر میانگین پاداش دریافتی برای عامل ۱ در بازی GG۲، (الف) الگوریتم Nash - Q و (ب) الگوریتم پیشنهادی.



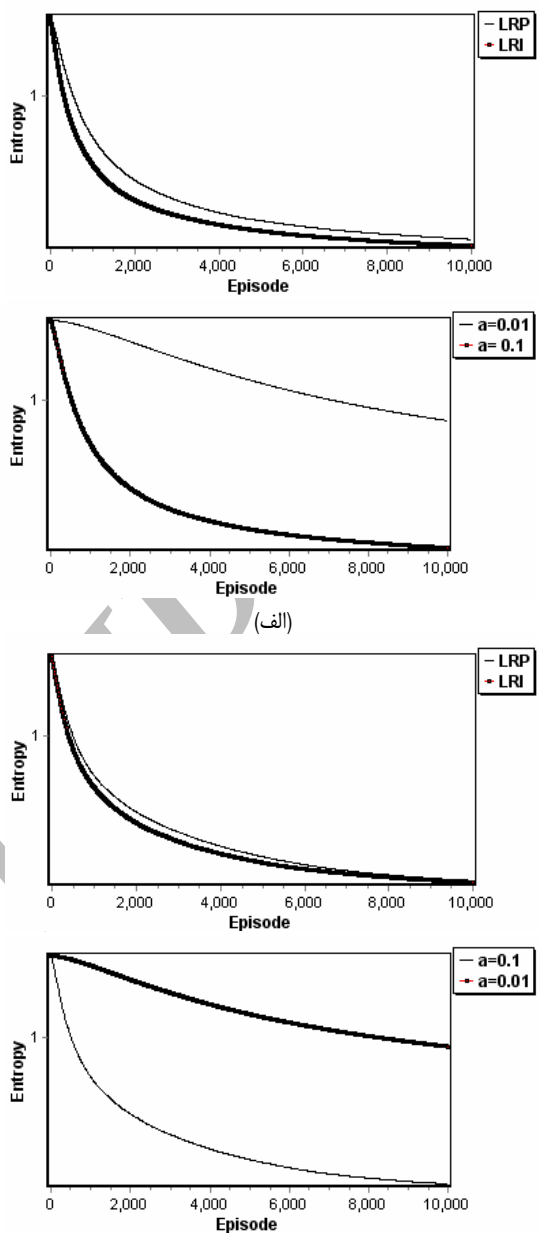
(الف)



(ب)

شکل ۱۲: بررسی رفتار روش پیشنهادی با توجه به (الف) الگوریتم یادگیری و (ب) پارامتر یادگیری برای بازی GG۲.

پیشنهادی. در این آزمایش نقش پارامترهای یادگیری a و b در همگرایی و کارایی الگوریتم پیشنهادی در دو نوع بازی مورد بررسی قرار می‌گیرد. با در نظر گرفتن مقدار $a = \{۰/۱, ۰/۰۱\}$ و $b = ۰$ می‌بینیم با افزایش پارامتر a شیب رسیدن به نقطه تعادل بالا رفته و در فاصله زمانی کمتری به پاداش مطلوب‌تر می‌رسد. در بخش دوم آزمایش به نقش پارامتر b در کارایی الگوریتم پرداخته می‌شود. با در نظر گرفتن مقادیر $b = \{۰, ۰/۰۱\}$ نقش الگوریتم L_{RI} و سپس الگوریتم L_{RP} از نظر میانگین پاداش برای عامل ۱ مشاهده می‌گردد. با توجه به نتایج به‌دست آمده دستاورد قضیه مطرح‌شده در [۱۵] الگوریتم L_{RI} در بازی‌های مارکوفی نیز تأیید می‌گردد و الگوریتم L_{RI} رفتار بهتری را از نظر همگرایی ارائه می‌دهد. شکل‌های ۱۱ و ۱۲ این مقایسه را نشان می‌دهند.

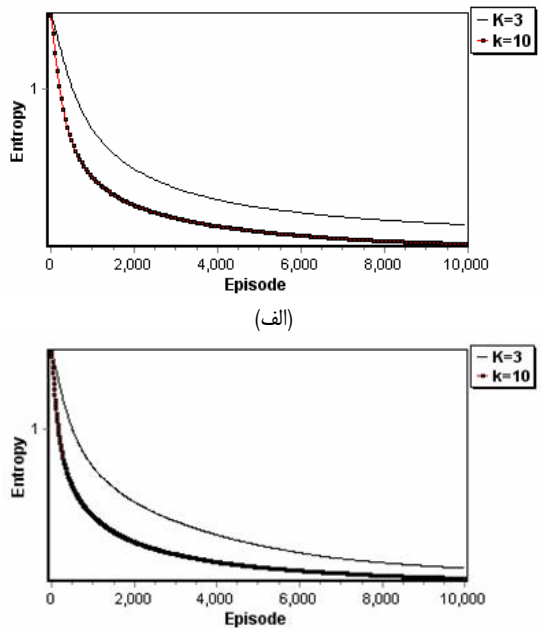


(الف)

(ب)

شکل ۱۳: تأثیر پارامترهای مختلف بر آنتروپی سراسری محیط برای دو بازی (الف) GG۱ و (ب) GG۲.

آزمایش ٤: بررسی رفتار الگوریتم با توجه به معیار آنتروپی کلی. در این آزمایش از معیار آنتروپی کلی سیستم به‌عنوان یک پارامتر کمی مستقل از مسأله در یاگیری استفاده شده که بیانگر درجه همگرایی سیستم است. برای این منظور آنتروپی کلی محیط با توجه به پارامترهای مختلف یادگیری برای هر دو نوع بازی مورد بررسی قرار گرفته است. همان‌طور که در شکل ۱۳ دیده می‌شود میزان آنتروپی کلی سیستم در هر مرحله از یادگیری به مرور نشان داده شده است. با توجه به گذشت زمان یادگیری می‌بینیم رفتار سیستم به مرور از آنتروپی سراسری بالا که بیانگر بی‌نظمی سیستم است به سمت آنتروپی پایین تغییر می‌کند. در ابتدا که تمام احتمالات اعمال اتوماتاها با یکدیگر برابرند آنتروپی هر حالت دارای بیشترین مقدار بوده و لذا باعث می‌شود آنتروپی کلی محیط نیز در حداکثر باشد. به مرور با توجه به یادگیری عامل‌ها می‌بینیم از میزان آنتروپی کلی محیط کاسته می‌شود. این تغییرات همان همگرایی الگوریتم یادگیری را نشان می‌دهد. همان‌طور که در شکل دیده می‌شود رفتار الگوریتم در مورد آنتروپی کلی نیز در اینجا صدق می‌کند.



(الف)

(ب)

شکل ۱۴: تأثیر پارامتر K بر آنتروپی سراسری محیط برای الگوریتم MLA در دو بازی (الف) GG۱ و (ب) GG۲.

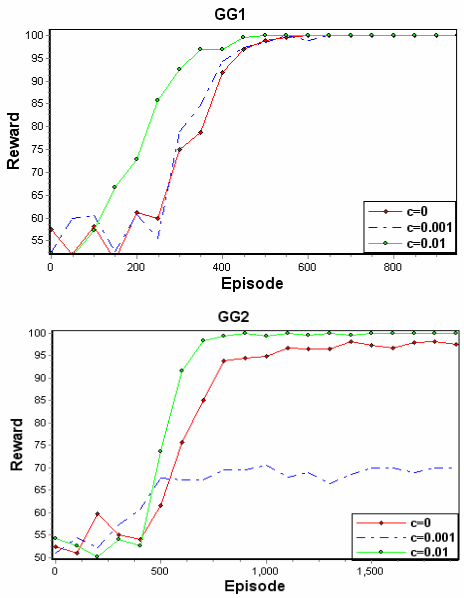
آزمایش ٥: بررسی نقش پارامتر K در آنتروپی کلی محیط. در این آزمایش به بررسی نقش پارامتر K در تغیرات آنتروپی کلی پرداخته می‌شود. برای این منظور پارامتر یادگیری ۰٫۱ و پارامتر $K=\{3,10\}$ در نظر گرفته شده است. با توجه به شکل ۱۴ می‌بینیم افزایش پارامتر K باعث کاهش سریع‌تر آنتروپی کلی می‌گردد.

آزمایش ٦: در این آزمایش به بررسی الگوریتم اصلاح‌شده پرداخته می‌شود. برای این منظور برای بررسی رفتار بازی در هر اپیزود فرض می‌شود که عامل‌ها از همان نقطه شروع دوباره آغاز می‌کنند (تصادفی نیست) و میانگین پاداش دریافتی در هر اپیزود ترسیم می‌شود. ابتدا مقدار K ثابت و مساوی مقدار ۱ نگه داشته شده و نقش پارامتر c مورد بررسی قرار می‌گیرد. آزمایش‌ها در دو نوع بازی با مقادیر $c=\{0,0.001,0.01\}$ انجام گرفته‌اند. در حالت $c=0$ همان الگوریتم اولیه است. با توجه به نتایج به‌دست آمده در شکل ۱۵ بهترین حالت موقعی است که مقدار $c=0.01$ داشته باشد.

همان‌طور که در شکل دیده می‌شود با مقدار دهی اولیه K برابر یک، ابتدا الگوریتم حالت جستجوگرانه داشته و با افزایش مقدار K به مرور زمان، امکان بهره‌برداری عامل از دانسته‌هایش میسر شده و کارایی آن بهبود می‌یابد.

۵- نتیجه‌گیری

در این مقاله مدلی مبتنی بر مفهوم آنتروپی و اتوماتاهای یادگیر برای حل بازی‌های تصادفی پیشنهاد گردید و سپس الگوریتمی مبتنی بر آن با در نظر گرفتن شرایط جستجو و استناد بر تجربیات عامل‌ها ارائه گردید. در روش پیشنهادی به‌ازای هر عامل یک اتوماتای یادگیر در هر حالت از محیط قرار می‌گیرد که این یادگیرنده‌ها، عامل‌ها را برای رسیدن به اهداف خود هدایت می‌کنند. در روش پیشنهادی از مفهوم آنتروپی بردار احتمالات اتوماتای یادگیری برای پاداش‌دادن به عامل‌ها استفاده گردید. نتایج آزمایش‌های انجام‌گرفته نشان می‌دهند آنتروپی می‌تواند به‌عنوان یک عامل تسریع‌کننده در یادگیری به یادگیری بهتر و تطبیق‌پذیرتر کمک



شکل ۱۵: بررسی رفتار بازی از نظر پاداش دریافتی با توجه به الگوریتم یادگیری تغییریافته.

نماید. همچنین معیاری به‌نام آنتروپی کلی برای بررسی رفتار الگوریتم یادگیری در محیط چندعامله پیشنهاد گردید. تنظیم پارامترهای پاداش و جریمه اتوماتاهای یادگیر می‌تواند کارایی حل مسأله را افزایش داده به‌طوری که استفاده از تغییرات پارامتر K در طول اجرای بازی می‌تواند زمان رسیدن به نقاط تعادل را بهبود بخشد. مدل ارائه‌شده می‌تواند با ایجاد تغییرات در نحوه پاداش‌دهی عامل‌ها و با توجه به پویایی محیط مسائل غیر همکارانه و همکارانه نیز مورد استفاده قرار گیرد. با توجه به نتایج آزمایش‌ها و مقایسه با الگوریتم‌های مشابه اتوماتاهای یادگیر مدل مناسب یادگیری بین عامل‌ها در سیستم‌های چندعامله بوده و می‌تواند به‌عنوان راه حلی مناسب و کارا در بازی‌های مارکوفی به کار روند.

مراجع

[1] G. Weiss, Multi - Agent Systems: A Modern Approach to Distributed Artificial Intelligence, Cambridge, MA: MIT Press, 1999.

[2] H. Van Dyke Parunak, "A practitioners' review of industrial agent applications, autonomous agents and multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 4, pp. 389-407, Dec. 2000.

[3] A. K. Goel, V. Kummar, and S. Sirivasan, "Application of multi - agent system & agent coordination," in *Proc. 2nd National Conf. Mathematical Techiniques: Emerging Paradigms for Electronics and IT Industries, MATEIT'2008*, pp. 328-337, New Dehli, India, Sep. 2008.

[4] L. Busniu, R. Babuska, and B. Schutter, "A comprehensive survey of multi-agent reinforcement learning," *IEEE Trans. on System, Man, Cybern.*, vol. 38, no. 2, pp. 156-171, Mar. 2008.

[5] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proc. ICML'94*, pp. 157-163, Mar. 1994.

[6] J. Osborne and A. Rubinstein, *A Course in Game Theory*, Cambridge, MA: MIT Press, 1994.

[7] J. Hu and M. P. Wellman, "Online learning about other agents in a dynamic multi-agent system," *J. of Cognitive System Research*, vol. 2, no. 1, pp. 67-79, Apr. 2001.

[8] J. Hu and M. P. Wellman, "Nash Q-learning for general - sum stochastic games," *J. of Machine Learning Research*, vol. 4, pp. 1039-1069, Nov. 2003.

[9] A. Greenwald and K. Hall, "Correlated Q - learning," in *Proc. of the Twentieth Int. Conf. on Machine Learning, ICML'2003*, pp. 242-249, Aug. 2003.

[10] M. Song, G. Gu, and G. Zhang, "Pareto-Q learning algorithm for cooperative agents in general-sum games," in *Proc. CEEMAS 2005*, pp. 576-578, Sep. 2005.

[11] M. Song, J. Bai, and R. Chen, "A new learning algorithm for cooperative agents in general - sum games," in *Proc. of the Sixth Int. Conf. on Machine Learning and Cybernetics*, pp. 50-54, Hong Kong, Aug. 2007.

[12] M. R. Khojasteh and M. R. Meybodi, "Evaluating learning automata as a model for cooperation in complex multi - agent domains," Lecture Notes in Artificial Intelligence, Springer Verlag, LNAI 4434, pp. 409-416, ???. 2007.

[13] A. Now´e, K. Verbeeck, and M. Peeters, "Learning automata as a basis for multi - agent reinforcement learning," Lecture Notes in Computer Science, vol. 3898, pp. 71-85, ???. 2006.

[14] P. Vrancx, K. Verbeeck, and A. Nowe, "Decentralized learning in markov games," *IEEE Trans. on Systems, Man and Cybernetics-pt. B: Cybernetics*, vol. 38, no. 4, pp. 976-981, Aug.. 2008.

[15] P. S. Sastry, V. Phansalkar, and M. A. L. Thathachar, "Decentralized learning of Nash equilibria in multi-person stochastic games with incomplete information," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 24, no. 5, pp. 769-777, May 1994.

[16] R. A. Howard, Dynamic Programming and Markov Processes, MIT Press, 1960.

[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning I: an Introduction*, MIT Press, 1998.

[18] F. Thusijsman, *Optimality and Equilibria in Stochastic Games*, CWI-tract 82, Centre for Mathematics and Computer Science, Amsterdam, 1992.

[19] J. F. Nash Jr., "Non-cooperative games," *Annals of Mathematics*, vol. 54, no. 2, pp. 286-295, Sep. 1951.

[20] A. M. Fink, "Equilibrium in a stochastic N-person game," *J. of Science in Hiroshima University, Series A - I*, vol. 28, pp. 89-93, 1964.

[21] H. Qio, F. Szidarovszky, J. Rozenblit and L. Yong, "Multi-agent learning model with bargaining," in *Proc. of the 38th Conf. on Winter Simulation*, pp. 934-940, Dec. 2006.

[22] K. S. Narendra and M. A. L Thathachar, *Learning Automata: an Introduction*, Prentice Hall, 1989.

[23] M. A. L. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," *IEEE Trans. on Systems, Man, and Cybernetics - Pt B.: Cybernetics*, vol. 32, no. 6, pp. 711-722, Dec. 2002.

[24] M. A. L. Thathachar and P. S. Sastry, Networks of Learning Automata: Techniques for Online Stochastic Optimization, Kluwer Academic Publishers, 2004.

[25] M. Costa, A. L. Goldberger, and C. K. Peng, "Multi-scale entropy analysis of complex physiologic time series," *Physical Review Letters*, vol. 89, pp. 68101-68104, Aug. 2002.

[26] Z. Dianhu, F. Shaohui, and D. Xiaojun, "Entropy - a measure of uncertainty of random variable," *Systems Engineering and Electronics*, vol. 11, pp. 1-3, Dec. 1997.

بهروز معصومی در سال ۱۳۷۴ مدرک کارشناسی مهندسی کامپیوتر–نرم افزار خود را از دانشگاه شهید بهشتی تهران و در سال ۱۳۷۷ مدرک کارشناسی ارشد مهندسی کامپیوتر خود را از دانشگاه آزاد اسلامی واحد تهران جنوب دریافت کرد. ایشان در سال ۱۳۸۹ موفق به اخذ درجه دکترا در مهندسی کامپیوتر–نرم افزار از دانشگاه آزاد اسلامی علوم و تحقیقات تهران گردید. نام‌برده از سال ۱۳۷۷ تاکنون، عضو هیات علمی دانشکده مهندسی کامپیوتر و فناوری اطلاعات دانشگاه آزاد اسلامی قزوین می‌باشند. زمینه‌های علمی مورد علاقه ایشان عبارتند از: سیستم‌های چند عامله، یادگیری در سیستم‌های چند عامله، طراحی سیستم‌های پایگاه داده و محاسبات نرم.

محمدرضا میبدی تحصیلات خود را در مقاطع کارشناسی و کارشناسی ارشد اقتصاد بترتیب در سالهای ۱۳۵۲ و ۱۳۵۶ از دانشگاه شهید بهشتی و در مقاطع کارشناسی ارشد و دکتری علوم کامپیوتر به‌ترتیب در سالهای ۱۳۵۹ و ۱۳۶۲ از دانشگاه اوکلاهمای آمریکا به پایان رسانده است و هم‌اکنون استاد دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر می‌باشد. نام‌برده قبل از پیوستنش به دانشگاه صنعتی امیرکبیر در سالهای ۱۳۶۲ الی ۱۳۶۴ استادیار دانشگاه میشیگان غربی و در سالهای ۱۳۶۴ الی ۱۳۷۰ دانشیار دانشگاه اوهایو در ایالات متحده آمریکا بوده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: الگوریتم‌های موازی، پردازش موازی، محاسبات نرم و کاربردهای آن، شبکه‌های کامپیوتری و مهندسی نرم افزار.