A new approach for Squad behavior in computer games

چکیده:

صنعت بازی‌های رایانه‌ای امروزه به یکی از پرطرفدارترین و سرگرم‌کننده‌ترین صنایع در جهان تبدیل شده است. یکی از اصلی‌ترین عوامل ایجاد هیجان و جذابیت در بازی‌های رایانه‌ای هوش مصنوعی است و سعی سازندگان بازی‌های رایانه‌ای استفاده هرچه بیشتر از این عامل در محصولاتشان است، زیرا باعث بهبود روند بازی می‌شود. هوش مصنوعی بازی‌ها خود شامل چندین بخش مانند شبیه‌سازی جمعیت، کنترل وسایل نقلیه و ... می‌باشد، که از جمله پر استفاده‌ترین آن‌ها می‌توان به طراحی هوش مصنوعی برای تعقیب و در نهایت دستگیری بازیکن اشاره کرد. در طراحی این سیستم‌ها که عموما چندعامله هستند، یکی از مهم‌ترین چالش‌ها قرار دادن هر عامل در جای مناسب برای هرچه بهتر رسیدن به هدف است.

مدل پیشنهادی ما از یک سیستم رخداد محور و چندین سیستم پیش‌بینی کلی تشکیل شده است و با استفاده از این سیستم‌ها سعی دارد که بهترین چیدمان را برای عامل‌های در اختیارش پیدا کند. در ادامه توضیحات این سیستم را بررسی می‌کنیم.

کلمات کلیدی:

هوش مصنوعی، سیستم‌های چند عامله، هوش مصنوعی مرکزی، هوش مصنوعی توزیع شده

۱. مقدمه

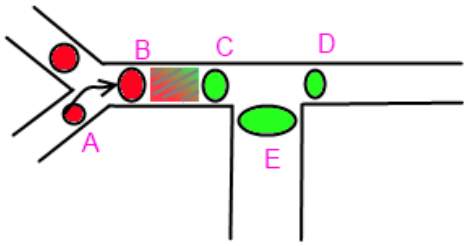
در این مدل هوش مصنوعی ما یک عامل مرکزی داریم که نقش فرماندهی و ارتباط یک عامل با عامل‌های دیگر را ایفا می‌کند و یک هوش توزیع شده که هرکدام از عامل‌ها را در انتخاب تصمیم‌ها یاری می‌کند. همچنین تعدادی عامل در اختیار داریم که هر کدام از آن‌ها در مکان‌های مختلفی از نقشه قرار گرفته‌اند. وظیفه‌ی کلی این سیستم دستگیری بازیکنی است که در مکانی از نقشه قرار دارد. بازیکن آزادانه می‌تواند در هر مسیری که بخواهد جا به جا شود. عامل‌ها در شهر بر اساس نقاطی که برایشان از قبل منظور شده است به گشت زنی می‌پردازند و با دیدن بازیکن در صورتی که بازیکن تحت تعقیب باشد، به تعقیب او می‌پردازند. مسیریابی عامل‌ها با استفاده از سیستم Navigation Mesh و الگوریتم A* انجام می‌گیرد. هر عاملی که به تعقیب بازیکن بپردازد ابتدا به فرمانده (هوش مصنوعی مرکزی) موقعیت و وضعیت خود را گزارش می‌دهد سپس فرمانده بر

مبنای اطلاعات رسیده، عامل‌های مفید را از بقیه جدا می‌کند. (منظور از عامل مفید عاملی است که توانایی کمک به تعقیب بازیکن را داشته باشد، برای مثال خیلی دور نباشد یا در ماموریت دیگری قرار نگرفته باشد).

سپس با محاسبه‌ی فاصله‌ی واقعی هرکدام از عامل‌های مفید تا بازیکن که با استفاده از الگوریتم A* بر روی خانه‌های ساخته شده با سیستم Navigation Mesh به دست می‌آید تصمیماتی را در خصوص اینکه هر کدام از عامل‌ها در کجا قرار بگیرند را اتخاذ می‌کند و به هر کدام از عامل‌ها تصمیم متناظر خودش را ارسال می‌کند و عامل آن را انجام می‌دهد. هدف اصلی فرمانده محاصره کردن بازیکن و بستن راه‌های فرار وی است و این کار را با فرستادن عامل‌ها به انتهای مسیرهایی که احتمال حضور بازیکن در آن‌ها بیشتر است انجام می‌دهد.

۲. سیستم ساخت نقشه

ما به منظور پیاده‌سازی ایده خود زمین و راه‌های آن را که معرّف نقشه بازی است، به یک گراف مدل کردیم. در این مدل راس‌ها به محل‌های ورود و خروج اطلاق می‌شود.



شکل ۱: یافتن مسیر عامل

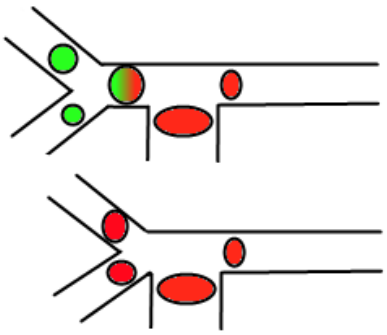
به عنوان مثال در شکل ۱ نقاط رنگی نشان دهنده یک رأس در گراف هستند (چرا که محل ورود یا خروج یک مسیر جدید می‌باشند) و خطوط درون شکل نشان دهنده یال‌ها می‌باشند. در این گراف، یال AB نشان می‌دهد که یک مسیر از رأس A به B و بالعکس وجود دارد که نیاز به عبور از هیچ رأس دیگری ندارد. استفاده از این شیوه مدل‌سازی به ما کمک می‌کند که مستقل از مختصات جغرافیایی فرد در حال تعقیب (بدون توجه به x و y مکانی) و تنها به کمک آگاهی از چند رأس آخری که از آن‌ها عبور کرده، محل نسبی فرد را تشخیص دهیم. همچنین به کمک روشی که در زیر معرفی می‌شود، ما قادر به تشخیص تغییر مسیرها خواهیم بود. به این منظور ما گراف را به کمک ماتریس مجاورت (Adjacent Matrix) پیاده‌سازی کردیم. در نتیجه، اگر فرض کنیم که می‌دانیم فرد در حال تعقیب به ترتیب از رأس‌های A و B عبور کرده باشد، مشخص می‌شود که این فرد در حال حرکت به سمت رأس C است و در ناحیه هاشور خورده مشخص شده در شکل است. این کار (تشخیص مکان بعدی و در نتیجه آن، محل نسبی فرد) به کمک تفریق مجموعه رئوس مجاور B از مجموعه رئوس مجاور رأس A به دست می‌آید، که مشخصا

پردازش بسیاری کمی را نیاز دارد. از آن جا که در مدل پیشنهادی ما، رئوس به عبارتی یک نقطه رهایی (trigger) می‌باشند، ما از نگرش پیشامد محور (Event Base) در انجام محاسبات فوق استفاده می‌کنیم. به این معنا که در هر بار عبور کردن فرد در حال تعقیب از یک گذرگاه، محاسبات فوق یک بار انجام می‌شود.

از آن جا که ما مسیر پیشروی فرد در دست تعقیب را به کمک ۲ رأس آخری که وی از آن‌ها عبور کرده است، محاسبه کنیم، این نکته مهم است که این ۲ رأس باید متوالی باشند. از طرف دیگر از آن جا که یکی از جنبه‌هایی که در این طرح پیشنهادی مبنا بوده است شبیه‌سازی بودن آن است، ما تنها در صورتی رخداد عبور از یک رأس را پردازش می‌کنیم که یکی از عامل‌های در حال تعقیب، شخص در حال فرار را در آن لحظه ببینند، در غیر این صورت مانند دنیای واقعی هیچ رخدادی صادر نمی‌شود.

به عنوان مثال اگر فرد در دست تعقیب، بعد از عبور از گذرگاه‌های A، B و C، از گذرگاه E عبور کند ولی عاملی به او نرسد و مدتی بعد در حال عبور از گذرگاه D دیده شود، نشان می‌دهد که او توسط عاملی در حال تعقیب نبوده است و ما اطلاعاتی از وی نداشتیم. پس نمی‌توانیم تنها بر اساس رأس D نظر بدهیم. در اینجا ما بر اساس نقشه واقعی بازی (و نه مدل گراف آن) مسیر بعدی را به دست می‌آوریم. این کار می‌تواند به شکل‌های مختلف انجام گیرد که ما از روش شبیه‌سازی حرکت واقعی با سرعت بسیار بالا استفاده کردیم.

همچنین ما در ادامه سعی کردیم برای بهینه‌سازی گراف، تغییراتی در انتخاب رئوس داشته باشیم. بدین منظور ما سعی کردیم بر اساس فاصله دو رأس متوالی و اهمیت ناحیه بین ۲ رأس (که نماینده یک یال در گراف، و یک مسیر در نقشه است) رئوس را با یکدیگر ادغام کنیم. در شکل ۲ دو نمونه از نحوه بهینه انتخاب رئوس شکل ۱ آورده شده است.

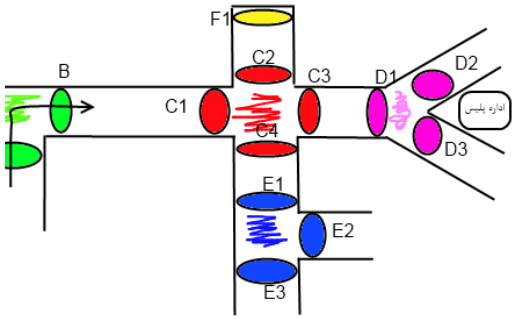


شکل ۲: نحوه بهینه انتخاب رئوس

۳. درخت پیش‌بینی مسیر

همان طور که در بخش ۲ به درخت پیش‌بینی اشاره شد، در هر بار عبور از رئوس، ما یک پردازش برای بررسی محل بعدی فرد در حال گریز انجام می‌دهیم. ما همچنین همزمان با به‌روزرسانی اطلاعات فوق، یک درخت پیش‌بینی مسیر نیز تشکیل می‌دهیم. علت همزمانی این دو عملیات، عدم امکان تغییر مسیر در بین دو رأس است. یعنی پس از این که فرد در حال تعقیب از یک گذرگاه عبور کرد، تا رسیدن فرد به رأس بعدی، تغییر مسیری نمی‌تواند صورت گیرد که این فرصت مناسبی را برای تشکیل درخت ما مهیا می‌کند. (شایان ذکر است که گذر از رئوس که منجر به ورود به یک ناحیه چند راهی می‌شود به شکل دیگری بررسی می‌شود که از بیان این حالت خاص صرف نظر می‌کنیم). هدف از تشکیل درخت فوق که به نوعی از گراف اصلی ما ولی به شکل خاص استفاده می‌کند، به دست آوردن احتمال حرکت فرد در حال گریز به محل‌های در پیش روی اوست. نتیجه‌ای که از این درخت برای ما اهمیت دارد، به دست آوردن مکان‌های مناسب برای قرار دادن عامل‌ها و بستن راه‌ها است.

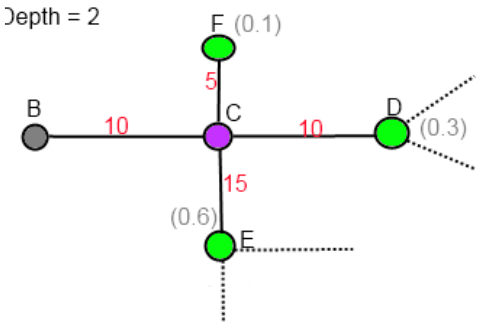
برای نمونه و بر اساس شکل ۳، رئوس C1 تا C4 از نظر مفهومی نمایانگر یک منطقه قابل پوشش توسط نیروهای تعقیب‌کننده است، پس ما در درخت خود، برای چنین رئوسی، یک رأس را به عنوان نماینده برای قرارگیری عامل استفاده می‌کنیم.



شکل ۳: منطقه های قابل پوشش توسط عامل تعقیب‌کننده

نحوه تشکیل این درخت را با یک مثال نشان می‌دهیم.

اگر فرض کنیم فرد در دست تعقیب به ترتیب از گذرگاه‌های A و B گذشته باشد، برای عمق ۲، درختی مانند شکل ۴ خواهیم داشت:



شکل ۴: مثال درخت جستجو در عمق ۲

در این درخت وزن یال‌ها نمایانگر طول مسیر و وزن برگ‌ها نشان دهنده احتمال آن مکان است. (احتمال آنکه فرد در حال فرار به آن جا بگریزد) در محاسبه این احتمالات ما ۲ معیار اصلی را مورد توجه قرار دادیم :

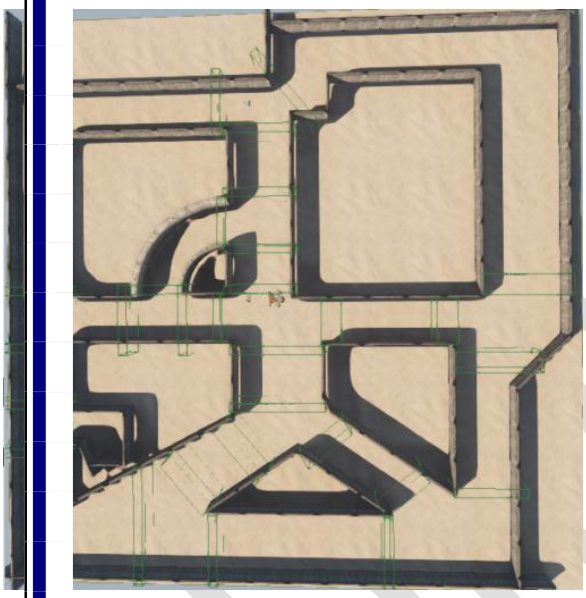
۱. **میزان خطر گذرگاه‌ها:**

این معیار توسط طراح بازی ارزش‌دهی می‌شود و به شرایط فیزیکی یک رأس و عامل‌های از این دست بستگی دارد. به عنوان مثال، یک مکان که دارای آیتم‌هایی برای پنهان شدن بازیکن باشد، نسبت به یک مسیر که به بن‌بست ختم می‌شود، مناسب‌تر در نظر گرفته می‌شود و از ارزش بالاتری برخوردار خواهد بود.

۲. **تاریخچه مسیرهای انتخابی:**

در این قسمت، یک سیستم ارزش‌دهی جداگانه برای گذرگاه‌ها در نظر گرفته می‌شود که بر اساس مشاهدات عامل‌های تعقیب‌کننده در حین تعقیب و بر اساس انتخاب‌های فرد در دست تعقیب، ارزش‌دهی می‌شود. به عنوان مثال، اگر بازیکن چند بار در یک دوراهی از مسیر سمت راست برود و این کار توسط عامل‌ها دیده شود، باعث می‌شود رأس انتهای مسیر سمت راست از ارزش بیشتری نسبت به رأس موجود در مسیر سمت چپ، برخوردار شود.

در شکل ۵ نمای نقشه‌ای که تیم نگارندگان مقاله آزمایش‌های خود را در آن انجام داده، در کنار گذرگاه‌های آن نمایش داده شده است:

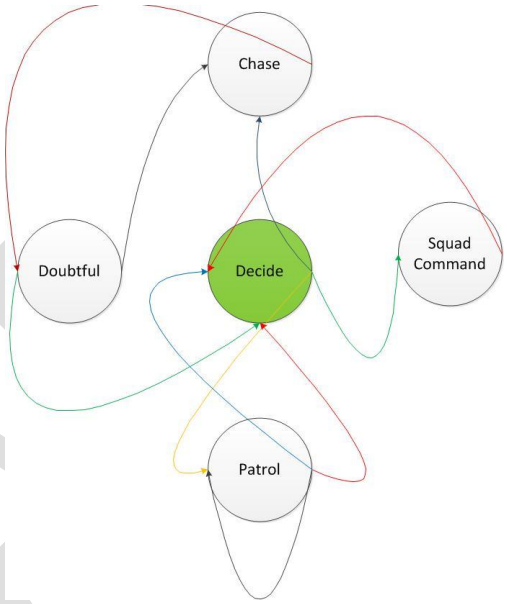


شکل ۵: نقشه مسیرها و گذرگاه‌های آزمایش

۴. آزمایش هوش مصنوعی فردی

سیستم هوش مصنوعی فردی با استفاده از ساختار Finite State Machine (FSM) ساخته شده است. در این سیستم هر عامل دارای چندین حالت است که با توجه به شرایط و فرمان‌هایی که از سیستم هوش مرکزی می‌گیرد، وارد یکی از این

حالت‌ها می‌شود. در شکل ۶ حالت‌های مختلف در نظر گرفته شده را مشاهده می‌کنید.



شکل ۶: ماشین گذر متناهی آزمایش

۴-۱ حالت تعقیب

این حالت هنگامی رخ می‌دهد که عامل کاربر را مشاهده کرده و از طرف فرمانده دستوری نداشته باشد. در این حالت بسته به موقعیت بازیکن و عامل اگر عامل از بازیکن عقب‌تر باشد و هر دو در یک جهت حرکت کنند (عامل در حال تعقیب از پشت سر بازیکن) عامل به مکانی که بازیکن را می‌بیند می‌رود و در هر لحظه مکان بازیکن را به روز رسانی کرده و به فرمانده اطلاع می‌دهد.

در صورتی که عامل از پشت سر بازیکن در تعقیب او نباشد (به طور مثال از رو به رو یا از طرفین) مکان احتمالی برخورد با بازیکن توسط فرمول زیر به دست آمده و عامل را به موقعیت Position می‌فرستیم تا در یک زمان به بازیکن برسد و او را دستگیر کند.

$$\Delta S = s_{agent} - s_{player}$$

$$\Delta V = V_{agent} - V_{player}$$

$$T = \frac{\Delta S}{\Delta V}$$

$$Position = s_{agent} + V_{agent} \times T$$

در صورتی که در حین مراحل تعقیب بازیکن از دید عامل ناپدید شد، عامل وارد حالت مشکوک شده و در صورت مشاهده بازیکن به حالت تعقیب باز می‌گردد.

۴-۲ حالت مشکوک

این حالت زمانی رخ می‌دهد که عامل در هنگام تعقیب دیگر موفق به دیدن بازیکن نشود. در این حالت ابتدا به آخرین جایی که بازیکن را دیده رفته و سپس در صورت مشاهده کردن بازیکن دوباره وارد حالت تعقیب می‌شود، در صورت عدم مشاهده بازیکن، مسیر روبه‌رو (در صورت وجود) یا یکی از مسیرهای ممکن را تا مدتی ادامه می‌دهد و در صورت عدم مشاهده بازیکن به حالت تصمیم‌گیری باز می‌گردد.

۴–۳ حالت گشت زدن

در این حالت وظیفه‌ی عامل رفتن به نقاط خاصی در نقشه و انجام کار خاصی در آن مکان به صورت تکرار شونده است. برای اجرای این حالت ما باید از قبل برای عامل‌ها مکان‌هایی را در نقشه مشخص کرده باشیم، برای مثال می‌توان گشت زنی در یک محله را با این سیستم به این صورت شبیه‌سازی کرد که عامل از اولین نقطه‌ی داده شده شروع کرده و به صورت ترتیبی به تمام نقاط مربوطه رفته و کار متناظر آن نقطه را انجام می‌دهد. (برای مثال انداختن نور چراغ قوه در مغازه‌ها و بررسی امنیت آن‌ها)

در صورت مشاهده مورد خاصی یا دریافت فرمانی از فرمانده وارد حالت تصمیم‌گیری می‌شود در غیر این صورت به گشت زدن خود می‌پردازد.

۴–۴ حالت دستور فرمانده:

این حالت مربوط به انجام دستورات فرمانده است و در هنگامی که فرمانی از فرمانده می‌رسد، عامل وارد این حالت می‌شود. این حالت بر طبق قابلیت‌هایی که برای سیستم در نظر گرفته‌ایم، دارای پیاده‌سازی‌های متفاوتی است، که عمده‌ترین تفاوت آن‌ها فرمان‌های پشتیبانی شده توسط عامل است. در مدل ما به دلیل آنکه هدف نهایی محاصره و دستگیری بازیکن است، تنها فرمان ارسالی فرمانده به هر عامل دستور حرکت به نقطه‌ای خاص است.

برای اجرای این دستور فرمانده باید نقطه‌ای را که برای محاصره‌ی بازیکن بهتر است، به عامل معرفی کند. سپس عامل به سمت آن نقطه حرکت می‌کند و در صورتیکه بازیکن را در طول پیمودن مسیر مشاهده نکند خود را به آن نقطه می‌رساند. سپس از فرمانده یا دستور سکون یا حرکت به نقطه ای جدید دریافت می‌کند که به دستور جدید عمل می‌کند.

۴–۵ حالت تصمیم‌گیری

این حالت که اولین و اصلی‌ترین حالت عامل و به نوعی مغز عامل است، بر اساس اولویت‌های تعیین شده در داخل آن تصمیم به رفتن به حالت دیگری می‌گیرد.

اولویت‌های قرار گرفته در مدل ما برای محاصره بازیکن به صورت زیر است:

۱. در صورت داشتن دستوری از فرمانده رفتن به حالت دستور فرمانده
۲. در صورت دیدن بازیکن و تحت تعقیب بودن آن رفتن به حالت تعقیب
۳. در صورت داشتن مسیر برای گشت زدن رفتن به حالت گشت زنی

همچنین اکثر حالت‌ها پس از انجام کارهای تعیین شده به حالت تصمیم‌گیری وارد

می‌شوند و به این صورت حلقه‌ی اصلی تفکر عامل تشکیل می‌شود.

۵. سیستم هوش مصنوعی مرکزی

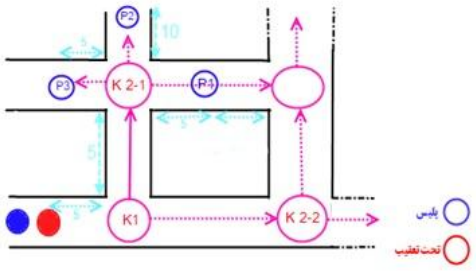
ما به منظور کنترل و هماهنگی بین نیروهای موجود از هوش جمعی استفاده می‌کنیم. این به اصطلاح واحد مرکزی وظیفه دارد که اطلاعات و مشاهدات تمام نیروها را گرفته و پس انجام پردازش‌ها، دستورت لازم را به عامل‌ها ارسال می‌کند. واحد مرکزی در دو مقطع زمانی به تصمیم گیری می‌پردازد:

۱. هنگامی که بازیکن در حال تعقیب از یک گذرگاه عبور کند و در عین حال توسط یک عامل قابل مشاهده باشد

۲. دریافت اطلاعات از عامل‌ها

حالت اول:

حالت ۱ همان طور که در بخش‌های قبل معرفی شد، با ایجاد یک درخت پیش‌بینی مسیر آغاز می‌گردد. عمق این درخت بر اساس معیارهای گوناگونی از قبیل تعداد عامل‌ها، موقعیت عامل‌ها و ... تعیین می‌گردد. در مرحله اول ما درختی با عمق یک ایجاد می‌کنیم. بر اساس برگ این درخت، امکان بسته شدن مسیر توسط یکی از نیروها برسی می‌گردد. اگر بتوان عاملی بدین منظور پیدا کرد مشکل حل شده هست، در غیر این صورت عمق درخت یک واحد افزایش می‌یابد. ما بر اساس عمق درخت و وزن یال‌ها و مکان فعلی عامل‌ها **جدول شماره ۱ را از شکل ۷** به وجود می‌آوریم.



شکل ۷: نقشه مسیرها و گذرگاه‌های آزمایش

	K1	K2-1	K2-2
P1	-5	+5	+5
P2	-10	0	-10
P3	-5	+5	-5

جدول ۱: جدول ساخته شده بر اساس شکل ۷

در توضیح این جدول، فرض کنید تمامی افراد از جمله فرد در حال فرار، بتوانند با سرعت ثابت ۱ واحد بر ثانیه بدوند. در این صورت مدت زمانی که برای پیمودن M واحد لازم است، برابر M ثانیه خواهد بود. با این فرض، خانه‌های جدول بالا، اختلاف زمان رسیدن فرد در حال فرار به یکی از مکان‌های حاصل از برگ درخت پیش‌بینی، نسبت به هر یک از عامل‌های P1 تا P3 را نشان می‌دهد. یعنی :

$$\Delta t = (Kx\text{-}T_{yP}) - (Kx\text{-}T_{yA}) \qquad (۱)$$

زمان رسیدن بازیکن به خانه y =T_{yP}

زمان رسیدن عامل P به خانه y =T_{yA}

همان طور که از شکل ۷ و جدول ۱ مشخص است، ما در مرحله اول سعی می‌کنیم که در لایه‌ی اول به موفقیت برسیم. به این منظور باید به دنبال عاملی باشیم که بتواند قبل از رسیدن بازیکن در حال گریز، خود را به مکان K1 برساند. ولی از آن جا که برای این لایه تمام اختلاف زمان‌ها منفی است، مشخص است که امکان چنین چیزی وجود ندارد. پس باید به بررسی لایه بعدی بپردازیم.

موفقیت در هر لایه منوط به بسته شدن تمام برگ‌های آن لایه است که در مورد لایه دو، بسته شدن مکان‌های K_{1-2} و K_{2-2} مدنظر است. همان طور که در جدول مشخص است، این کار با قرار دادن نیروی P1 در مکان K_{2-2} و یکی از دو نیروی P2 یا P3 در مکان K_{1-2} قابل انجام است.

اگر در شرایطی در این لایه نیز به قادر به بستن راه‌ها نمی‌شدیم، می‌بایستی لایه بعدی مورد بررسی واقع می‌شد.

روند افزایش لایه‌ها زمانی خاتمه می‌یابد که یا در لایه فعلی به موفقیت برسیم و بتوانیم تمامی مکان‌های آن لایه را (برگ‌های آن لایه را) پوشش دهیم و یا مجموعه اختلاف زمان‌ها لایه فعلی از لایه قبلی کمتر شود.

هم چنین به منظور کاهش محاسبات و نتیجه بهتر، گسترش درخت به صورت ناحیه‌ای (Partial) صورت می‌پذیرد. به این معنا که اگر در یک لایه قسمتی از مکان‌ها قابل پوشش و قسمتی غیر قابل پوشش باشند ما تنها بررسی لایه بعد را از سمت برگ‌های غیر قابل پوشش ادامه می‌دهیم و گسترش به صورت ناحیه‌ای صورت می‌پذیرد که باعث کاهش چشمگیری در هزینه پردازش‌ها می‌گردد.

در رابطه با نحوه چیدمان عامل‌ها در مکان‌ها، چند حالت ممکن است اتفاق بیفتد، که هر یک را جداگانه شرح می‌دهیم:

a. **تعداد عامل‌ها بیشتر یا برابر تعداد برگ‌ها باشد:**
این حالت به معنی وجود عامل‌های اضافی است. در این حالت ما از عامل‌ها با چند رویکرد متفاوت و بسته به مکان و فاصله‌ی آن‌ها از فرد در حال گریز و فاصله با سایر برگ‌ها استفاده می‌کنیم.

الف بستن مسیر از پشت (این کار به منظور پشتیبانی عامل‌های اصلی بکار می‌رود)
ب افزایش عامل‌ها در مکان‌های محتمل‌تر (با توجه به وزن برگ‌ها)

b. **تعداد** عامل‌ها **کمتر از تعداد برگ‌ها باشد:**
در این حالت که معمول‌تر از حالت قبلی است، سعی بر آن است که یک

عامل به دو یا چند ناحیه اختصاص داده شود. در این حالت یک عامل بر اساس میزان احتمال برگ‌هایی که به وی اختصاص داده شده و فاصله وی از هر یک از آن‌ها، در حد فاصل برگ‌ها قرار می‌گیرد. مثالی از انجام گیری این کار بدین شکل است که فرض کنید یک عامل داریم و دو نقطه باید بسته شود، ما مدت زمان رسیدن بازیکن به هر کدام از این دو نقطه را محاسبه و سپس بر مبنای کمترین زمان، عامل را در مسیر بین دو نقطه و در فاصله ای از نقطه‌ی نزدیک‌تر به بازیکن قرار می‌دهیم که اگر از طرف فرمانده متوجه شد که بازیکن وارد کدام مسیر شده است، وقت برای رسیدن به آن نقطه را داشته باشد و خود را به آنجا برساند.

حالت دوم:

برای کنترل بیشتر و طبیعی شدن رفتار عامل‌ها، ما از یک ارتباط دوطرفه بین واحد مرکزی و عامل‌ها استفاده می‌کنیم. دین منظور ما از مشاهدات عامل‌ها در موقعیت‌هایی مانند زیر استفاده می‌کنیم و از واحد مرکزی تقاضای تصمیم گیری می‌کنیم:
۱ دیدن فرد در دست تعقیب: ما این وضعیت را در دو حالت بررسی می‌کنیم:

a اطلاعات و موقعیت فرد در حال گریز در دست باشد:

اگر عاملی، به غیر از آن دست از عامل(ها) که صریحتا وظیفه تعقیب فرد در حال گریز را دارند، فرد در حال گریز را ببینند، سیگنالی به واحد مرکزی ارسال کرده و واحد مرکزی بسته به شرایط برای این عامل تصمیم می‌گیرد. به عنوان مثال در شرایطی که امکان حمله و دستگیری مهیا باشد، به سمت فرد در حال گریز می‌رود.

b زمانی که ما در حالت Suspicious باشیم و اطلاعات و موقعیت فرد در حال گریز در دست نباشد

در این حالت واحد مرکزی در درجه اول با اختصاص دادن همان عامل (و یا چند عامل نزدیک دیگر) سعی می‌کند مجددا وی را گم نکند و پس از حصول اطمینان از این مرحله، سایر عامل‌ها را مطابق آنچه در قسمت‌های قبل بین شد (با استفاده از درخت پیش‌بینی) تحت تعقیب وی قرار می‌دهد.

۲ گم کردن فرد در حال گریز توسط عامل(ها)یی که وظیفه تعقیب مستقیم وی را داشتند:

در چنین وضعیتی به حالتی که مشکوک نامید می‌شود می‌رویم.

۶. نتیجه‌گیری

هوش مصنوعی ارزش بالایی در ایجاد هیجان در بازی دارد و به همین دلیل ما سعی کردیم با ارائه‌ی

یک بستر مناسب برای هوش مصنوعی که هم بر پایه‌ی هوش توزیع شده و هم هوش مرکزی ساخته شده، هوش مصنوعی را به واقعیت نزدیک کرده و از این طریق هیجان بالاتری را به بازیکن القا کنیم.

مراجع:

Comment [m۱]: دادن یک مقاله بدون دیدن کارهای دیگران هیچگاه کار دقیقی نمی شود