دانشکده مهندسی کامپیوتر

پایان نامه کارشناسی ارشد

گرایش هوش مصنوعی

عنوان

یادگیری خودکار مهارت با استفاده از رویکرد تشخیص انجمن

نگارش

محسن غفوریان

استاد راهنما

دکتر حمید بیگی

مهر ماه ۱۳۹۱

تقدیم به

پدر و مادر عزیزم

و

همراه همیشگی ام، همسر مهربانم

که بی وجودشان هرگز انجام این پایان‌نامه ممکن نبود.

تقدیر و تشکر

در اینجا جا دارد از زحمات استاد محترمم، جناب آقای دکتر حمید بیگی کمال تشکر را داشته باشم. توصیه-های بی‌دریغ ایشان سهم غیرقابل انکاری در به ثمر نشستن این پایان‌نامه داشته است.

چکیده

یادگیری تقویتی یک روش یادگیری است که از بازخورد پاداش و جریمه، بدون نشان دادن شیوه‌ی صحیح، استفاده می‌کند. در این روش، نخست محیط حالت خود را در اختیار عامل قرار داده و عامل براساس حالت محیط و سیاست خود، یک کنش از بین کنش‌های مجاز انتخاب کرده و به محیط اعمال می‌نماید. محیط با دریافت این کنش حالت خود را تغییر داده و ارزیابی خود را در قالب یک سیگنال تقویتی و حالت جدید محیط به عامل می‌دهد و عامل براساس سیگنال دریافتی، سیاست خود را به‌روز می‌کند. هدف این روش، بیشینه نمودن پاداش بلندمدت دریافتی است. یادگیری تقویتی در محیط‌هایی با تعداد حالت‌ها و کنش‌های کم سریعاً به پاسخ بهینه همگرا می‌شود، اما در محیط‌های طبیعی، با تعداد بسیار زیاد حالت‌ها و کنش‌ها، سرعت همگرایی معمولاً بیش از حد مطلوب است.

برای مسائل با اندازه‌ی بزرگ، استفاده از انتزاع زمانی می‌تواند به عنوان یکی از راه‌کارها برای حل سریع‌تر مسائل یادگیری در مقایسه با شیوه‌ی مرسوم، یا حتی امکان‌پذیر نمودن آن در فضاهایی با حالت‌های بسیار زیاد، به کار رود. استفاده از انتزاع زمانی می‌تواند از راه یادگیری و بکارگیری مهارت در محیط انجام شود. مهارت را می‌توان به صورت ترتیبی از اعمال پایه در نظر گرفت که عامل یادگیر آن را برای رسیدن به یک حالت مناسب در محیط به کار می‌برد. اگر گراف گذر محلی را در نظر بگیریم، می‌توان نقاط مرزی انجمن‌های موجود در گراف را به عنوان اهداف میانی قلمداد کرد، که عامل برای رسیدن به هدف نهایی باید از آن‌ها گذر نماید.

در این پایان‌نامه، الگوریتمی ارائه می‌شود که از دسته روش‌های بهینه‌سازی کلونی مورچه برای پیدا کردن زیرهدف‌ها استفاده می‌کند. در این الگوریتم، ابتدا مسیرهای مختلفی توسط مورچه‌ها از حالت شروع به حالت پایانی ساخته شده و تغییرات میزان فرومون یال‌هایی که روی کوتاه‌ترین مسیر قرار دارند، در طول زمان، بررسی شده و بر اساس شکل این توزیع، یال‌های مجاور زیرهدف از باقی یال‌ها متمایز می‌شوند. در ادامه، انجمن‌هایی را که این مسیرها از آن‌ها می‌گذرند، پیدا شده و مهارت‌هایی برای رسیدن به زیرهدف‌های مفید با استفاده از چارچوب گزینه ساخته می‌شوند. برای ارزیابی روش پیشنهادی، کارایی آن با چند روش دیگر در محیط‌های اتاق‌ها، تاکسی، برج‌های هانوی و اتاق بازی سنجیده و مقایسه می‌شود. نتایج به‌دست آمده نشان از بهبود عملکرد عامل در بیش‌تر این محیط‌ها دارد.

کلمات کلیدی: یادگیری تقویتی، کسب مهارت، کشف زیرهدف، چارچوب گزینه، الگوریتم بهینه‌سازی کلونی مورچه

فهرست مطالب

فهرست شکل‌ها

فصل اول

مقدمه

سیستم‌های هوشمند نقش بسیاری مهمی را در دنیای امروز ایفا می‌کنند. سیستم‌های هوشمند کنترل ترافیک، روبات‌های صنعتی و خدماتی و بسیاری از ابزار هوشمند دیگر که در علوم پزشکی و دیگر زمینه‌ها به کار می‌روند، نمونه‌هایی از این دست هستند. گاهی این سیستم‌ها حاصل طراحی مهندسین و طراحان با توجه پیش‌بینی دقیق شرایط محیط و مطلوبات این ابزارها می‌باشند، اما در بسیاری از موارد، به علت پیچیدگی زیاد محیط[۱]، امکان طراحی یکسره‌ی عامل برای نیل به اهداف تعریف شده، وجود ندارد و به همین دلیل این سیستم‌ها باید طوری طراحی شوند که خود با سنجش و گرفتن بازخورد از محیط، بتوانند عملکرد خود را تغییر داده و کارایی خود را بهبود ببخشند.

یادگیری ماشین سعی می‌کند کنش[۲] صحیح را در هر حالت[۳] ممکن، هنگام تعامل با محیط، پیدا کند. یکی از از شاخه‌های فعال در این حوزه، یادگیری تقویتی[۴] است که در آن فرایند یادگیری حاصل کسب تجربه‌ی عامل در تعامل با محیط است. در یادگیری تقویتی کنش صحیح به طور صریح مشخص نمی‌شود، بلکه متناسب با نحوه‌ی عملکرد عامل در محیط، یک سیگنال تقویتی به عنوان پاداش یا جریمه به عامل داده می‌شود.

[1] Environment
[2] Action
[3] State
[4] Reinforcement Learning

۱-۱ تعریف مساله

هنگامی که عامل[5] با مسائلی از دنیای واقعی مواجه شود، یک رفتار بهینه، احتمالاً شامل یک تعداد زیادی از کنش‌های پایه[6] است. در چنین شرایطی یادگیری هر کدام از کنش‌های این رفتار بهینه، نیازمند زمان و فرایند بلندمدتی از تعامل با محیط است و به نوعی می‌توان گفت یادگیری تقویتی با استفاده از روش‌های کلاسیک از جمله یادگیری Q [7] که در فصل آینده مفصلاً توضیح داده می‌شود، در این شرایط عملکرد مناسبی ندارد.

در مقابل انسان‌ها از یک توانایی تحسین‌برانگیز در حل مسائل سود می‌برند و آن رویکرد حل کل به جز مسائل است. به عنوان مثال، هر یک از ما می‌دانیم برای خروج از یک ساختمان در شرایطی که فرد در یک اتاق واقع در طبقات فوقانی قرار دارد، وی باید خود را از به درب اتاق رسانده، درب را باز کند و از آن خارج شود، سپس خود را به آسانسور واقع در طبقه‌ی مربوط برساند و از آسانسور برای رسیدن به طبقه‌ی همکف استفاده نماید. سپس از برای خروج از ساختمان وی باید از محل آسانسور در طبقه‌ی همکف به درب خروجی ساختمان منتقل شود. بنابراین خروج از ساختمان شامل چند گام کلی خواهد بود: رساندن خود به درب اتاق، انتقال به محل آسانسور، استفاده از آسانسور و رسیدن به محل درب خروجی در طبقه‌ی همکف.

برای حل بهینه‌ی مسائل یادگیری تقویتی، می‌توان از همین رویکرد استفاده نمود. ابتدا باید مسئله را به مسائلی کوچک‌تر تقسیم کرد و سپس مهارت‌هایی برای حل هرکدام از آن‌ها آموخت. پس از این، می‌توان در وظایف دیگر، بارها از آن مهارت استفاده کرد. به حالت‌های هدف هر یک از مهارت‌ها مانند درب اتاق، درب آسانسور و درب ساختمان در طبقه‌ی همکف، حالت‌های زیرهدف[8] گفته می‌شود.

روش‌های متنوعی برای تحقق رویکرد تقسیم و حل ارائه شده‌اند. روش پیشنهادی در این پایان نامه، از دسته روش‌های مبتنی بر گراف است که در آن ابتدا تعاملات عامل با محیط به وسیله‌ی یک گراف مدل می‌شود.

[5] Agent
[6] Primitive Actions
[7] Q Learning
[8] Sub-goal state

بیش‌تر روش‌های موجود، برای پیدا کردن زیرهدف‌ها، با الگوریتم‌های گوناگونی به کشف انجمن[۹]های این گراف می‌پردازند[۳]. روش ارائه شده، با ساختن مسیرهایی با استفاده از دسته الگوریتم‌های بهینه‌سازی مورچه، هر یک از یال‌ها را بررسی می‌کند و با انتخاب برخی از این یال‌ها، زیرهدف‌های موجود در فضای حالت استخراج می‌شوند. گام بعدی، ساختن مهارت‌هایی است که عامل را به این نقاط زیرهدف می‌رسانند. برای این کار از روش بازنمایی تجربه استفاده شده است. در نهایت، از این مهارت‌ها برای حل بهینه‌ی مسائل استفاده خواهد شد. بررسی نتایج آزمایش‌های انجام شده، نشان‌دهنده‌ی افزایش سرعت یادگیری عامل در این روش‌ها می‌باشد.

۱-۲ ساختار پایان نامه

ادامه‌ی این پایان‌نامه، به صورت زیر سازمان‌دهی شده است: فصل دوم این پایان نامه به معرفی یادگیری تقویتی، روش‌های حل مسئله‌ی مارکوف و چارچوب‌های سلسله‌مراتبی آن می‌پردازد. در فصل سوم مروری خواهد شد بر برخی از پژوهش‌هایی که پیش از این در این زمینه انجام شده‌اند. در ادامه، یک روش جدید برای حل مسئله‌ی کسب خودکار مهارت پیشنهاد می‌شود، که فصل چهارم به آن اختصاص داده شده است. در فصل پنجم، کارایی روش پیشنهادی در چندین محیط شبیه‌سازی شده، مورد ارزیابی قرار گرفته و با برخی دیگر از روش‌های کسب مهارت مقایسه می‌شود. در پایان، فصل ششم برای نتیجه‌گیری و پیشنهاداتی برای کارهای آتی در نظر گرفته شده است.

[9] Community Detection

فصل دوم

یادگیری تقویتی

به طور ساده می‌توان یادگیری تقویتی را فرایند یادگیری عامل در حین تعامل با محیط با استفاده از رویکرد تربیتی پاداش و جریمه تعریف کرد. در این نوع یادگیری، تنها وسیله برای انتقال دانش به عامل، سیگنال تقویتی پاداش یا جریمه می‌باشد و به همین دلیل، در این نوع یادگیری هیچ نیازی به وجود ناظر برای مشخص کردن کنش درست در هر حالت نیست. یادگیری تقویتی را می‌توان یکی از اصلی‌ترین شکل‌های فرایند یادگیری و تربیت در انسان‌ها دانست. انسان‌ها با قرارگیری در شرایط مختلف، از نتیجه‌ی اعمال خود بازخورد می‌گیرند. به عنوان نمونه کودک با برداشتن گام‌های بلند روی یخ، تعادل خود را از دست می‌دهد و زمین می‌خورد و با گرفتن سیگنال درد در بدن خود، یاد می‌گیرد که در شرایط این چنین باید گام‌های کوتاه‌تر و محکم‌تری بردارد.

از آن‌جا که یادگیری تقویتی، بدون مشخص‌کردن کنش صحیح در هر کدام از حالت‌ها صورت می‌گیرد، می‌توان گفت انجام یادگیری با منتقل کردن حداقل میزان دانش به عامل انجام می‌شود. حتی در بسیاری از مسائل، نیازی به مشخص‌کردن هدف نهایی یادگیری نیست و نیل به هدف از طریق مشخص کردن میزان مطلوبت کنش‌ها صورت می‌گیرد. این مسئله، مزیت بزرگی در محیط‌های بزرگ و پیچیده محسوب می‌شود.

وظیفه‌ی عامل در محیط، رسیدن به بیشترین پاداش دریافتی است. با توجه به این‌که کنش‌ها علاوه بر پاداش لحظه‌ای دریافتی، حالت بعدی عامل را نیز تعیین می‌کنند، مشخص است که عامل برای رسیدن به عملکرد بهینه، در انتخاب کنش خود علاوه بر پاداش دریافتی بی‌درنگ، باید حالت بعدی را هم در نظر بگیرد. بنابراین

یکی از چالش‌هایی که یادگیری تقویتی با آن مواجه است، پاداش‌های تاخیری[10] است، به این معنی که ممکن است یک سلسله از کنش‌ها، در شروع منجر به پاداش‌های قابل توجهی نباشد، اما عامل در نهایت به مقادیر زیاد پاداش دست پیدا کند.

یک نوع نگرش برای تمایز روش‌های مختلف یادگیری، دسته‌بندی آن‌ها با توجه به شکل انتقال دانش به عامل است. از این نقطه نظر، روش‌های یادگیری را می‌توان به سه دسته‌ی یادگیری نظارت شده[11]، یادگیری بدون نظارت[12] و یادگیری نیمه‌نظارت شده[13] تقسیم کرد. از این دیدگاه باید یادگیری تقویتی را در دسته‌ی روش‌های نیمه‌نظارتی قرار داد، چراکه همان‌طور که قبل‌تر گفته شد، در این روش نیاز به ناظری که کنش صحیح را مشخص کند، وجود ندارد. از طرفی، برخلاف روش‌های بدون نظارت که هیچ گونه اطلاعی از ناظر یا محیط دریافت نمی‌کنند، سیگنال تقویتی از جانب محیط به عامل داده می‌شود.

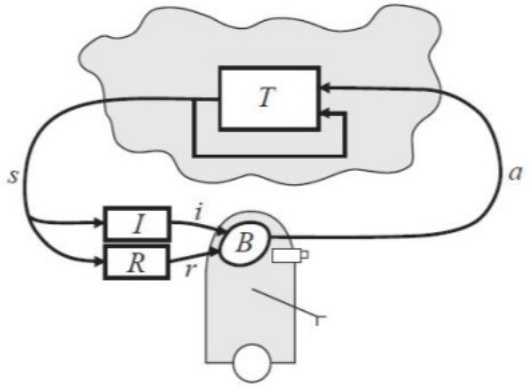
در ادامه‌ی این فصل، چارچوب یادگیری تقویتی و فرایند تصمیم‌گیری مارکوف[14] معرفی شده و برخی از روش‌های ارائه شده برای حل مسئله‌ی یادگیری تقویتی مرور می‌شوند. در نهایت چارچوب سلسله مراتبی و جایگاه آن در یادگیری تقویتی، توصیف خواهد شد.

۲-۱ فرایند تصمیم گیری مارکوف

در این قسمت به معرفی فرایند تصمیم گیری مارکوف متناهی و زمان‌گسسته[15] که یک چارچوب استاندارد برای یادگیری تقویتی می‌باشد، خواهیم پرداخت.

[10] Delayed Reward
[11] Supervised Learning
[12] Unsupervised Learning
[13] Semi-supervised Learning
[14] Markov Decision Process
[15] Finite Discrete Time Markov Decision Process

پیش‌تر گفتیم که عامل از طریق اعمال کنش‌ها و دریافت سیگنال تقویتی با محیط در تعامل است. در هر گام زمانی $t = 0, 1, 2, \ldots$ عامل، حالت محیط در آن لحظه (s_t)، را دریافت کرده و سپس کنش a_t، را به عنوان خروجی انتخاب می‌نماید. انجام این کنش سبب تغییر حالت محیط شده و عامل تاثیر آن را با دریافت حالت جدید s_{t+1} و سیگنال تقویتی r_{t+1}، درک می‌کند. شکل (۲- ۱) نمایی از این تعامل را ارائه می‌دهد.



شکل (۲-۱): نمایی از تعامل عامل با محیط، s سیگنال تقویتی، i حالت جدید، r پاداش دریافتی است[۲].

در صورتی که این روند در جایی منجر به رسیدن به حالت هدف شود، در این صورت تعامل عامل با محیط را می‌توان به دوره‌هایی تقسیم نمود. در این شرایط وظیفه را وظیفه‌ی دوره‌ای[16] می‌نامیم که در تقابل با وظایف پیوسته[17] می‌باشد.

تا این‌جا، برای توصیف شرایط محیط، از واژه‌ی حالت استفاده شده، که ممکن است تا حدی مبهم باشد. منظور از حالت، هرگونه اطلاعی است که محیط در اختیار عامل قرار می‌دهد. حالت محیط می‌تواند از پردازش اطلاعات آنی دریافتی، یا حتی از اطلاعات قبلی حاصل شود. به عنوان نمونه، عامل بعد از شنیدن کلمه‌ی بله در لحظه‌ی t، ممکن است بسته به سوالی که در لحظه‌ی قبلی پرسیده است، در حالت‌های متفاوتی قرار گیرد.

[16] Episodic Task
[17] Continous Task

برای ساده‌تر شدن مسئله، برای ما مطلوب است که s_t تمام اطلاعات مفید مربوط به حال و گذشته را در خود خلاصه کند. در این صورت سیگنال دریافتی عامل، باید چیزی فراتر از یک دریافت آنی باشد و در عین حال نیازی به نگه‌داشتن تاریخ تمام حالت‌ها و کنش‌ها نیست. چنین سیگنال حالتی دارای خاصیت مارکوف[۱۸] است. به عنوان نمونه، مکان و سرعت یک پرتابه، تمام اطلاعات مورد نیاز برای پیش‌بینی ادامه‌ی حرکت یک جسم را دارا می‌باشد. پس سیگنالی که حاوی بردار سرعت و مکان جسم باشد، با وجود نداشتن سرعت و مکان پرتابه در لحظات قبلی، برای عامل کفایت می‌کند و بنابراین دارای خاصیت مارکوف است.

یک مسئله‌ی یادگیری تقویتی که خاصیت مارکوف را داشته باشد، فرایند تصمیم‌گیری مارکوف (MDP) نامیده می‌شود و در صورتی که مجموعه‌ی کنش‌ها و حالت‌های آن متناهی باشد، به آن فرایند تصمیم‌گیری مارکوف متناهی گفته می‌شود. بیش‌تر مسائلی که یادگیری تقویتی امروزه با آن سروکار دارد به این صورت مدل می‌شوند.

به صورت رسمی، فرایند تصمیم گیری مارکوف به صورت چهارتایی (S, A, R, T) مشخص می‌شود. که در آن S مجموعه‌ی حالت‌ها، A مجموعه‌ی کنش‌ها، R یک تابع پاداش به صورت $R : S \times A \to \mathbb{R}$ که در آن $\mathbb{R}$ نشان‌دهنده‌ی مجموعه‌ی اعداد حقیقی است و T تابع انتقال به صورت $T : S \times A \to \Pi(S)$ است که $\Pi(S)$ یک توزیع احتمالاتی روی مجموعه‌ی S است که نگاشتی از هر حالت به یک احتمال می‌باشد.

با فرض متناهی بودن A و S، تغییر حالت‌های محیط و پاداش دریافتی را می‌توان با توزیع احتمالاتی یک گامی به ترتیب به صورت روابط (۲-۱) و (۲-۲) مدل کرد[۴]:

$$P_{ss'}^{a} = Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\} \qquad (۲\text{-}۱)$$

[18] Markov Property

$$R_{ss'}^{a} = E\{r_{t+1} | s_t = s, a_t = a, s_{t+1} = s'\} \quad (۲-۲)$$

در روابط بالا، $P_{ss'}^{a}$ نشان‌دهنده‌ی احتمال گذر از حالت s به s' بعد از انجام کنش a و $R_{ss'}^{a}$ بیان‌گر امیدریاضی مقدار پاداش دریافتی لحظه‌ای حاصل از انجام کنش a در حالت s و رفتن به حالت s' می‌باشد.

هدف عامل یادگیری رفتار بهینه در محیط است. رفتار عامل از سیاست[۱۹] عامل ناشی می‌شود. سیاست مارکوف با یک نگاشت از تمام کنش‌های مجاز هر حالت، به یک احتمال انتخاب مشخص می‌شود. طبق یکی از تعاریف ممکن، رفتار بهینه عبارت است از انتخاب کنش‌ها، به شکلی که درآمد تخفیف‌خورده[۲۰] عامل بیشینه شود. این معیار در رابطه‌ی زیر نشان داده شده است[۴]:

$$R = E\left\{\sum_{t=0}^{\infty} \gamma^t r_t\right\} \quad (۳-۲)$$

که در آن، $0 \leq \gamma < 1$ نرخ تخفیف[۲۱] نامیده می‌شود و برای کاهش اثر پاداش دریافتی در زمان‌های بسیار دور در نظر گرفته شده است. همچنین وجود آن، از واگرا شدن این سری جلوگیری می‌کند.

فرض کنید سیاست نه لزوماً بهینه‌ی π را در اختیار داشته باشیم، در این صورت تابع ارزش–حالت[۲۲] برای سیاست π در حالت s، به این صورت تعریف می‌شود[۴]:

$$V^{\pi}(s) = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s = s_t\right\} \quad (۴-۲)$$

[19] Policy
[20] Discounted Return
[21] Discounting Rate
[22] State-Value Function

مقدار تابع ارزش–حالت در حالت s، بیانگر امیدریاضی درآمد تخفیف‌خورده‌ی عامل است، در صورتی که با شروع از حالت s، از سیاست π پیروی کند. به شکل مشابه می‌توان تابع ارزش-کنش[۲۳] را نیز تعریف نمود[۴]:

$$Q^{\pi}(s,a) = E_{\pi}\left\{\sum_{k=0}^{\infty}\gamma^{k} r_{t+k+1} \middle| s_t = s, a_t = a\right\} \quad (۵-۲)$$

برای یادگیری سیاست بهینه در محیط، می‌توان از دو ترفند استفاده کرد: در شیوه‌ی نخست، تابع ارزش-حالت بهینه V^* و در روش دوم، تابع ارزش-کنش بهینه Q^* تقریب زده می‌شود. $V^*(s)$، نشان دهنده‌ی امید ریاضی بیشترین مقدار درآمد تخفیف‌خورده‌ای است که عامل می‌تواند با شروع از حالت s کسب کند. در مقابل $Q^*(s,a)$، بیان‌گر بیشترین مقدار درآمد تخفیف‌خورده‌ای است که عامل می‌تواند پس از اعمال کنش a در حالت s به‌دست آورد. در مورد این دو تابع تقریبی، باید گفت که هر کدام از دو تابع را می‌توان با داشتن دیگری به‌دست آورد[۴]:

$$V^*(s) = \max_{a\in A} Q^*(s,a) \quad (۶-۲)$$

$$Q^*(s,a) = \sum_{s'\in S} P^a_{ss'}\,(R^a_{ss'} + \gamma V^*(s')) \quad (۷-۲)$$

همچنین سیاست بهینه، با داشتن هر یک از این دو، قابل حصول است[۴]:

$$\pi^*(s) = \arg\max_{a\in A} Q^*(s,a) \quad (۸-۲)$$

$$\pi^*(s) = \arg\max_{a\in A}\left[\sum_{s'\in S} P^a_{ss'}\,(R^a_{ss'} + \gamma V^*(s'))\right] \quad (۹-۲)$$

برای سیاست غیرقطعی و نه لزوماً بهینه‌ی π، می‌توان تابع ارزش–حالت و ارزش-کنش را به صورت زیر نیز به-دست آورد[۴]:

$$V^{\pi}(s) = \sum_{a\in A}\pi(s,a)\Big(\sum_{s'} P^a_{ss'}(R^a_{ss'} + \gamma V^{\pi}(s'))\Big) \quad (۱۰-۲)$$

[23] Action-Value Fucntion

$$Q^{\pi}(s,a) = \sum_{s'} P^{a}_{ss'} \left[R^{a}_{ss'} + \gamma \sum_{a'} \pi(s',a') Q^{\pi}(s',a') \right] \quad (۱۱-۲)$$

با داشتن این روابط بازگشتی و مدل دقیقی از محیط، ابزار مناسبی برای حل مسئله‌ی مارکوف در اختیار خواهیم داشت. می‌توان با شروع از یک سیاست غیربهینه و مقادیر دلخواه اولیه برای تابع ارزش، از روابط بالا به صورت تکراری برای تقریب زدن تابع بهینه استفاده کرد. در ادامه، روش‌های حل مسئله‌ی مارکوف بیان می‌شوند.

۲-۲ حل مسئله‌ی مارکوف

برای حل مسئله‌ی یادگیری تقویتی، سه دسته روش معروف وجود دارد[۴] که از فرض خاصیت مارکوف استفاده می‌کنند:

۱. **برنامه‌نویسی پویا**[۲۴]: در این روش سیاست بهینه با مفروض دانستن مدل دقیق محیط، با استفاده از روابط (۲-۱۰) و (۲-۱۱) و بدون تعامل با محیط، محاسبه می‌شود. از آن‌جایی که به طور معمول، داشتن مدل دقیق محیط امکان‌پذیر نیست، این روش معمولاً به کار نمی‌رود.

۲. **روش مونت کارلو**[۲۵]: بدون نیاز به داشتن مدل دقیق محیط، پس از یک دوره تعامل با محیط، توابع ارزش تقریب زده می‌شوند. این روش در محیط‌های دوره‌ای به کار برده می‌شود.

۳. **روش اختلاف‌زمانی**[۲۶]: این روش ترکیبی است از ایده‌های دو روش قبلی. مشابه برنامه‌نویسی پویا، تخمین خود را از آن‌چه تاکنون دیده است به‌روز می‌کند و منتظر پایان دوره‌ی تعامل نیست. مانند روش مونت‌کارلو بدون استفاده از مدل محیط، توابع ارزش را تقریب می‌زند.

[24] Dynamic Programming
[25] Monte Carlo
[26] Temporal Difference

با توجه به مزایای ذکر شده، در این پایان‌نامه فرض ما بر استفاده از روش‌های یادگیری اختلاف زمانی خواهد بود. در این دسته روش‌ها دو الگوریتم مشهور Sarsa و یادگیری Q وجود دارند، که تمرکز ما در قسمت‌های بعدی، بر یادگیری Q به عنوان الگوریتم مورد استفاده خواهد بود.

۲-۳ یادگیری Q

همان‌طور که گفته شد، با داشتن تابع ارزش-حالت یا تابع ارزش-کنش بهینه می‌توان سیاست بهینه را به‌دست آورد. روش یادگیری Q سعی می‌کند با استفاده از تکرار، تخمینی از تابع ارزش-کنش بهینه (Q^*) به‌دست آورد. در بیشتر محیط‌ها، به‌دست آوردن مدل دقیق محیط، شامل $P^a_{ss'}$ و $R^a_{ss'}$ به ازای همه‌ی حالت‌ها و کنش‌ها امکان‌پذیر نیست. به همین دلیل، استفاده از رابطه‌ی (۲-۱۱) برای حل مسئله‌ی یادگیری تقویتی معمول نمی-باشد. به همین دلیل در چنین شرایطی باید از رابطه‌ای استفاده کرد که تنها از تخمین تابع ارزش و پاداش دریافتی استفاده کند. رابطه‌ی (۲-۱۲) دارای چنین مشخصاتی است:

$$\hat{Q}(s,a) = (1-\alpha)\hat{Q}(s,a) + \alpha\left[R(s,a) + \gamma \max_{a' \in A} \hat{Q}(s',a')\right] \qquad (۱۲-۲)$$

در این رابطه، $R(s,a)$ پاداش دریافتی حاصل از انجام کنش a در حالت s است. همچنین α که نرخ یادگیری[۲۷] نامیده می‌شود، ضریبی است متعلق به بازه‌ی $[۰,۱)$ که نشان‌دهنده‌ی سرعت تغییرات تابع ارزش-کنش می-باشد. هرچه این مقدار بزرگ‌تر باشد، تغییرات ناگهانی‌تر و هر چه کوچک‌تر باشد، تغییرات نرم‌تر خواهد بود. الگوریتم ۱، شبه‌کدی برای روش یادگیری Q ارائه می‌دهد.

[27] Learning Rate

الگوریتم ۱: یادگیری Q

ورودی: (α, γ)

مقادیر $\widehat{Q}(s, a)$ را به طور تصادفی مقداردهی کن.

برای هر دوره تکرار کن.

سیاست π را با استفاده از $\widehat{Q}$ به‌دست بیاور.

s را مقداردهی کن.

تکرار کن.

کنش a را با استفاده از سیاست π انتخاب کن.

کنش a را انجام بده و پاداش $R(s, a)$ و حالت بعدی s' را مشاهده کن.

$$\hat{Q}(s, a) \leftarrow \left(۱ - \alpha\right) \hat{Q}(s, a) + \alpha\left[R(s, a) + \gamma \max_{a' \in A} \hat{Q}(s', a')\right]$$

تا زمانی که s یک حالت پایانی باشد.

در این روش در هر مرحله برای انتخاب کنش‌ها، از سیاستی استفاده می‌شود که خود از تخمین تابع ارزش-کنش حاصل شده است. انجام کنش‌ها در هر یک از حالت‌ها، با در نظر گرفتن میزان پاداش لحظه‌ای و بهترین مقدار ارزش-کنش قابل‌دسترسی در ادامه، باعث بهبود این تخمین می‌گردد. با ادامه‌ی این روند به میزان کافی، تابع تخمین‌زده شده‌ی $(\hat{Q})$ به مقدار بهینه همگرا شده[۵] و به این ترتیب سیاست بهینه به‌دست می‌آید.

۲-۴ چارچوب سلسله مراتبی در یادگیری تقویتی

روش‌های یادگیری تقویتی که تاکنون بررسی شده‌اند، در بسیاری از مسائل، به موفقیت‌هایی دست پیدا کرده‌اند. این مسائل عموماً حاوی حالت‌ها و کنش‌های چندان زیادی نبوده‌اند. اما این روش‌ها در مسائلی که با حالت‌ها و کنش‌های بیشتری مواجه شوند، نیازمند زمانی بیش‌از مدت قابل قبول برای رسیدن به سیاست‌های بهینه می-

باشند و به این علت، کارایی چندانی ندارند. به همین دلیل، رویکرد فعلی به یادگیری تقویتی، به سمت گسترش‌پذیر[۲۸] نمودن هر چه بیش‌تر این روش‌ها، تغییر کرده است.

از آن‌جا که یادگیری رفتاری انسان، تا حد زیادی به یادگیری تقویتی شباهت دارد، بررسی مختصات تصمیم-گیری انتخاب کنش‌ها در انسان، می‌تواند برای این مسئله راه‌گشا باشد. به نظر می‌رسد انسان‌ها هنگام تصمیم-گیری، برای انتخاب کنش در شرایط مختلف از شکل‌هایی از انتزاع بهره می‌برند. یکی از از این انواع انتزاع، زمانی رخ می‌دهد که فرد برای انجام یک عمل خاص، بسیاری از جزئیات نامربوط به این کنش را نادیده می‌گیرد. به عنوان مثال، زمانی که فردی در حال تایپ کردن یک متن است، برای تصمیم‌گیری، هرگز به محل پارک اتومبیل خود یا درجه حرارت یخچال فکر نمی‌کند، گرچه این موارد ممکن است در موقعیت دیگری برای فرد بسیار مهم باشند. با الهام‌گیری از این مفهوم، دسته‌ای از روش‌ها برای کوچک‌تر کردن فضای یادگیری تقویتی به وجود آمده اند، که معروف به روش‌های انتزاع حالت[۲۹] هستند و به نوعی متغیرهای حالت نامربوط را در انتخاب کنش‌ها دخیل نمی‌کنند.

ایده‌ی دیگر برای بکارگیری انتزاع، استفاده از استراتژی تقسیم و حل[۳۰] است. این مثال را در نظر بگیرید: فردی برای حضور در یک کنفرانس در یک شهر دیگر برنامه‌ریزی می‌کند. وی برای انجام این کار در نظر دارد که ابتدا یک تاکسی به مقصد فرودگاه گرفته، سپس سوار هواپیما شود و در فرودگاه شهر مقصد یک تاکسی گرفته و خود را به محل کنفرانس برساند. در صورتی که بنا باشد این برنامه‌ریزی، از ابتدا با استفاده از اعمال پایه‌ای، مثل قدم برداشتن، چرخیدن و .. صورت بگیرد، انجام این برنامه‌ریزی امکان‌پذیر نیست. استفاده از شیوه-ی تقسیم و حل و سپس بکارگیری مهارت‌های از پیش آموخته شده، مثل تاکسی گرفتن، سوار هواپیما شدن و ... حل این مسئله را امکان‌پذیر می‌سازد. استفاده از ایده‌ی یادگیری سلسله مراتبی و آموختن و استفاده از

[28] Scalable
[29] State Abstraction
[30] Divide and Conquer

فراکنش[31]ها، ایده‌ی اصلی دسته‌ی بزرگ دیگری از تلاش‌هایی است که برای گسترش‌پذیر ساختن یادگیری تقویتی انجام شده است. اصطلاحاً می‌گوییم این دسته از روش‌ها از تکنیک انتزاع زمانی[32] بهره می‌برند.

با توجه به آن‌چه گفته شد، بکارگیری چارچوب سلسله مراتبی در مسئله‌ی یادگیری تقویتی به این معنی است که عامل یادگیرنده، در محیط‌هایی که شامل تعداد حالت‌های زیادی است، مسئله را به وظایف کوچک‌تری تقسیم کرده و برای هر یک از این وظایف یک سیاست جزئی به‌دست می‌آورد و بدین ترتیب با ترکیب این سیاست‌های جزئی، سیاست کلی بهینه حاصل می‌شود. برای نیل به اهداف یادگیری تقویتی سلسله مراتبی چارچوب گزینه[33][۶] را معرفی و از آن استفاده خواهیم نمود.

۲-۴-۱ چارچوب گزینه

به دلیل استفاده از انتزاع زمانی، این نیاز به وجود آمده است که بتوانیم یک توصیف رسمی از فراکنش‌ها که قطعه‌ی سازنده‌ی اصلی یادگیری تقویی سلسله مراتبی می‌باشد، ارائه کنیم. در این زمینه چند چارچوب از جمله MAXQ [۷]، چارچوب سلسله‌مراتبی از ماشین‌های انتزاعی[۱] و چارچوب گزینه ارائه شده است. در ادامه‌ی این بخش توضیحاتی در مورد چارچوب گزینه ارائه خواهد شد.

گزینه‌ی o، یک سه‌تایی به صورت (I, π, β) است که I زیرمجموعه‌ای از حالت‌هاست که o می‌تواند در آن‌ها آغاز شود. π بیان‌گر سیاست مورد استفاده برای اعمال گزینه می‌باشد که به صورت $\pi: S \times A \rightarrow [0, 1]$ توصیف می‌شود. پارامتر سوم گزینه، شرط پایان آن می‌باشد که در واقع یک نگاشت از حالت‌ها به احتمال پایان است. یعنی اگر عامل در حین انجام گزینه، در حالت s باشد، به احتمال $\beta(s)$ اجرای گزینه خاتمه می‌یابد و با احتمال $1 - \beta(s)$ ادامه پیدا کرده و کنش a با احتمال $\pi(s, a)$ انتخاب می‌شود.

[31] Macro Action
[32] Temporal Abstraction
[33] Option Framework

برای یکسان بودن نحوه‌ی نمایش، هرکنش پایه هم به صورت یک گزینه در نظر گرفته می‌شود که مجموعه‌ی آغازین آن، حالت‌هایی است که کنش a روی آن حالت‌ها انجام‌پذیر است. سیاست این گزینه نیز برای همه‌ی حالت‌های s در مجموعه‌ی I، $\pi(s,a)=1$ می‌باشد به این معنی که در صورت انتخاب این گزینه‌ی تک‌گامی، کنش a به احتمال ۱ انجام می‌شود. همچنین برای همه‌ی حالت‌های s مجموعه‌ی S شرط پایان به صورت زیر تعریف می‌شود:

$$\beta(s)=1 \qquad (۱۳-۲)$$

پایان یافتن گزینه در هر حالتی با احتمال ۱ انجام می‌پذیرد. این شرط برای تضمین تک‌گامی بودن گزینه، قرار داده شده است.

طبیعی است که بتوان تابع ارزش-کنش را تعمیم داد و تعریفی برای تابع ارزش-گزینه[۳۴] به‌دست آورد[۶]:

$$Q^{\mu}(s,o)=E_{\pi}\left\{\sum_{k=0}^{\infty}\gamma^{k}r_{t+k+1}\,\middle|\,E(o\mu,s,t)\right\} \qquad (۱۴-۲)$$

که در آن $o\mu$ سیاستی است که o را تا پایان دنبال می‌کند و سپس μ را در حالت نهایی آغاز می‌نماید. همچنین $E(o\mu,s,t)$ نشان‌دهنده‌ی اجرای $o\mu$ در حالت s و در زمان t می‌باشد.

مسئله‌ی اصلی در این‌جا یادگیری سیاست بهینه با در اختیار داشتن مجموعه‌ای از گزینه‌های O می‌باشد. فرض کنید بعد از این‌که اجرای گزینه‌ی o در حالت s آغاز شد، در حالتی مانند s' خاتمه می‌یابد. بر اساس این تعامل، می‌توان مشابه روابط موجود برای تابع ارزش-کنش، رابطه‌ای برای به‌روز رسانی جدول به‌دست آورد[۶]:

$$\hat{Q}(s,o)=(1-\alpha)\hat{Q}(s,o)+\alpha\left[R(s,o)+\gamma^{k}\max_{o'\in A}\hat{Q}(s',o')\right] \qquad (۱۵-۲)$$

[34] Value-Option Function

در رابطه‌ی بالا، $R(s,o)$ نشان‌دهنده‌ی مجموع با تخفیف پاداش دریافتی حاصل از انجام گزینه‌ی o در حالت s و همچنین k تعداد کنش‌های پایه‌ای گزینه‌ی o می‌باشد. می‌توان نشان داد که مقادیر$\widehat{Q}(s,o)$ به ازای همه‌ی حالت‌ها و گزینه‌ها به مقادیر بهینه همگرا خواهند شد[۸].

۲-۵ جمع‌بندی

در این فصل به معرفی مسئله‌ی مارکوف و رویکرد یادگیری تقویتی برای حل آن پرداخته شده است. همچنین نشان داده شد که در محیط‌های پیچیده، برای امکان‌پذیر نمودن یادگیری تقویتی، نیاز به داشتن دیدگاه کلان به حل مسئله و انجام آن از طریق شکستن پیچیدگی مسئله به مسائل کوچک‌تر می‌باشد. برای رسیدن به این مهم، چهارچوب گزینه به عنوان امکانی برای پیاده‌سازی یادگیری سلسله مراتبی و بکارگیری فراکنش‌ها مطرح شد.

در فصل بعدی به مرور برخی از روش‌های پیشنهادی به منظور کشف و ساخت مهارت‌ها برای تکمیل فرایند یادگیری سلسله مراتبی، خواهیم پرداخت.

فصل سوم

پژوهش‌های پیشین

در فصل گذشته بیان شد که برای حل بسیاری از مسائل یادگیری با خاصیت مارکوف، می‌توان از روش‌های یادگیری تقویتی از جمله یادگیری Q استفاده نمود. اما متاسفانه زمانی که پیچیدگی این مسائل زیاد شود، این روش‌ها نمی‌توانند در مدت زمان مناسب پاسخگوی فرایند یادگیری باشند. به همین منظور، ایده‌هایی برای مقیاس‌پذیری مسائل یادگیری تقویتی ارائه شده است.

همان‌طور که گفته شد، بنای اصلی روش‌های مقیاس‌پذیر بر انتزاع استوار است و در دو فرم کلی می‌توانند ظاهر شوند. نخست استفاده از انتزاع حالت است که به معنی کنار گذاشتن ویژگی‌هایی از محیط است که در تصمیم‌گیری عامل بی‌تاثیر می‌باشند. این کنارگذاری می‌تواند سبب کوچک‌تر شدن فضای حالت و در نتیجه، افزایش سرعت فرایند یادگیری شود. اما ایده‌ی دوم استفاده از انتزاع زمانی، به معنی به‌کارگیری تکنیک تقسیم و غلبه در مسئله می‌باشد.

در این فصل قصد داریم به صورت مفصل‌تر به پژوهش‌های انجام شده در مقوله‌ی کشف انجمن‌ها و برخی از معروف‌ترین روش‌های انتزاع زمانی بپردازیم. ساختار این فصل به این گونه می‌باشد که در ابتدا یک دسته‌بندی کلی از روش‌های انتزاع زمانی ارائه می‌شود و در ادامه برخی از معروف‌ترین کارهای انجام شده در این زمینه، بررسی می‌شوند.

۳-۱ دسته‌بندی کلی روش‌ها

از آن‌جایی که اساس کار انتزاع زمانی، استفاده از تقسیم و غلبه است، تمام روش‌هایی که در این دسته جای می‌گیرند از دو مرحله‌ی کلی تشکیل می‌شوند. نخست تبدیل مسئله به زیر مسائل کوچک‌تر و حل هر کدام از این زیر مسئله‌ها و نهایتاً ترکیب راه‌حل آن‌ها برای رسیدن به یک راه حل کلی برای مسئله‌ی اصلی.

در مسائل یادگیری تقویتی، هدف عامل بیشینه کردن پاداش دریافتی است که معمولاً متناظر با رسیدن به یک حالت نهایی در مسئله می‌باشد. بنابراین یک تعبیر مناسب از تبدیل مسئله به زیر مسائل کوچک‌تر در یادگیری تقویتی، کشف حالت‌هایی است که برای رسیدن به هدف نهایی، حتماً باید از آن‌ها گذر کرد. این حالت‌ها را زیرهدف[۳۵] می‌نامیم. به این ترتیب، طبیعی است که عمده‌ی روش‌هایی که از تکنیک انتزاع زمانی سود می‌برند، ابتدا به دنبال کشف حالت‌های زیرهدف باشند، سپس مهارت‌هایی برای رسیدن به این هدف شکل دهند. این روش‌ها را روش‌های مبتنی بر کشف زیر هدف می‌نامند. روش‌های مبتنی بر کشف زیرهدف خود به دو دسته‌ی کلی تقسیم می‌شوند:

۱. **روش‌های مبتنی برای بسامد ملاقات**: در این دسته، تشخیص زیرهدف‌ها بر مبنای بسامد ملاقات حالت‌های محیط می‌باشد و معمولاً حالت‌های با بسامد بیش‌تر با احتمال بیش‌تری به عنوان زیرهدف در نظر گرفته می‌شوند. الگوریتم تازگی نسبی[۹] در این دسته جای می‌گیرد که در ادامه، توضیح بیش‌تری بر آن داده خواهد شد.
۲. **روش‌های مبتنی بر گراف**: در این دسته روش‌ها، سابقه‌ی تعاملات عامل با محیط، به صورت یک گراف نگه‌داری می‌شود و این گراف به عنوان مبنایی برای پیدا کردن زیرهدف‌ها، بدون پرداخت هزینه-های بعدی تعامل واقعی با محیط، در نظر گرفته می‌شود. در این دسته می‌توان از الگوریتم‌های معروف

[35] Sub-goal

نظریه‌ی گراف برای پیدا کردن زیرهدف‌های محیط استفاده کرد. روش‌های افراز محلی گراف[۱۰]، برش Q [۱۱]، افراز گراف مبتنی بر بینابینی[۱۲]، افراز با مولفه‌های قویاً همبند[۱۳] و افراز به کمک معیار مرکزیت بردار ویژه[۲] در این دسته جای می‌گیرند.

در بخش ۳-۳ هر کدام از روش‌های نام‌برده، بسط داده خواهد شد.

۳-۲ مروری بر برخی از روش‌های کشف انجمن

همان‌طور که گفتیم، دسته‌ی عمده‌ای از روش‌های انتزاع زمانی، پس از مدل کردن محیط به وسیله‌ی گراف، زیرهدف‌ها را به عنوان نقاط مرزی انجمن‌ها اکتشاف می‌کنند. بنابراین، پیدا کردن انجمن‌های گراف، از اهمیت بسیار بالایی در این دسته روش‌ها برخوردار است. در این قسمت به بررسی و معرفی پژوهش‌های انجام شده در مورد اکتشاف انجمن‌ها خواهیم پرداخت[۳].

۳-۲-۱ تعریف انجمن

در ابتدا باید به ارائه‌ی تعریفی از انجمن بپردازیم. به عنوان یک مفهوم کلی یک انجمن مجموعه‌ای از راس‌های گراف می‌باشد که درون آن اتصالات زیاد و بین راس‌های انجمن و دیگر راس‌ها، اتصالات بسیار کم‌تری وجود دارد.

از دیدگاه محلی، می‌توان به انجمن، به صورت یک موجودیت مستقل، بدون در نظر گرفتن دیگر راس‌های گراف نگاه کرد. برای تعریف محلی انجمن، چهار معیار دوبه‌دویی کامل[۳۶]، قابلیت دسترسی[۳۷]، درجه راس[۳۸] و مقایسه‌ی پیوستگی درونی و خارجی[۳۹] پیشنهاد شده‌اند.

[36] Complete Mutality
[37] Reachability
[38] Vertex Degree
[39] Comparision of Internal Versus External Cohesion

انجمن‌های اجتماعی را می‌توان سخت‌گیرانه به صورت گروهی از افراد که هر دو نفر از آن‌ها با هم دوست هستند، تعریف نمود. این تعریف مطابق با تعریف خوشه[۴۰] در نظریه‌ی گراف می‌باشد. از آن‌جایی که به طور معمول در گراف‌ها، خوشه‌های بزرگ به ندرت یافت می‌شوند، این تعریف بسیار سخت‌گیرانه است؛ چرا که به عنوان مثال، زیرگرافی که تنها یک یال کم‌تر از گراف کامل دارد، طبق این تعریف به عنوان یک انجمن در نظر گرفته نمی‌شود. این تعریف را می‌توان تا حدی ساده‌تر کرد و از خواص مربوط به قابلیت دسترسی استفاده نمود. خوشه-n [۴۱] به مجموعه‌ی حداکثری از راس‌ها گفته می‌شود که فاصله‌ی هر دو راس آن حداکثر n باشد. معیار درجه‌ی راس، انجمن را مجموعه‌ی بیشینه‌ای از راس‌ها تعریف می‌کند که در آن هر راس با بیش از یک حد آستانه از راس‌ها دیگر زیرگراف مجاور است. نهایتاً ایده‌ی معیار چهارم، استفاده از تعریف اولیه‌ی انجمن است. یک نمونه از این دسته معیارها، انجمن قوی[۴۲] [۱۴] می‌باشد که مجموعه‌ای از راس‌ها است که در آن، درجه‌ی داخلی هر راس از درجه‌ی خارجی آن بیش‌تر می‌باشد.

انجمن‌ها را می‌توان با توجه به کلیت گراف نیز تعریف نمود. یک دسته از این تعاریف، گراف را دارای ساختار انجمنی می‌دانند، اگر ساختار آن با گراف تصادفی[۴۳] متفاوت باشد. گراف تصادفی گرافی است که هر زوج راس با احتمال یکسان مجاور هستند. از این تعریف در این فصل برای معیار پیمانه‌ای بودن[۴۴]، استفاده خواهیم نمود.

یک دسته‌ی دیگر از تعریف‌ها، از شباهت بین راس‌ها انجمن‌ها استفاده می‌کنند. هر راس متعلق به انجمنی است که راس‌های آن بیش‌ترین شباهت را به آن دارد. در این‌جا باید معیاری از شباهت راس‌ها تعریف شود. یک ایده بردن هر راس به فضای nبعدی اقلیدسی و استفاده از تعاریف مختلف فاصله در این فضاست. در صورتی که نتوان به هر راس یک نقطه در فضای nبعدی نسبت داد، می‌توان از مقایسه‌ی راس‌های همسایه‌ی هر راس استفاده کرد.

[40] Clique
[41] n-clique
[42] Strong Community
[43] Random Graph
[44] Modularity

در ادامه به مرور برخی از روش‌های مطرح شده خواهیم پرداخت، که از تعاریف فوق برای اکتشاف انجمن‌ها استفاده می‌کنند. دسته روش‌هایی که در این نوشتار معرفی می‌شوند، عبارتند از: افراز گراف[۴۵]، خوشه‌بندی سلسله مراتبی[۴۶]، خوشه‌بندی افرازی[۴۷]، خوشه‌بندی طیفی[۴۸]، روش‌های تقسیمی[۴۹] و روش‌های مبتنی بر پیمانگی[۵۰].

۳-۲-۲ افراز گراف

مسئله‌ی افراز گراف شامل تقسیم راس‌ها به گروه‌هایی با تعداد مشخص است به طوری‌که تعداد یال‌های بین گروه‌ها کمینه شود. به تعداد یال‌هایی که بین خوشه‌ها قرار دارند، اندازه‌ی برش[۵۱] می‌گوییم. مشخص نمودن تعداد گروه‌ها به عنوان یک شرط برای مسئله ضروری است؛ چرا که در صورت آزاد بودن این تعداد، در نظر گرفتن کل گراف به عنوان تنها خوشه، یک راه حل بدیهی و بهینه با اندازه‌ی برش صفر خواهد بود. از طرفی اعمال محدودیت روی اندازه‌ی هر یک از خوشه‌ها نیز لازم است چرا که در غیر این صورت یک راه حل ساده اما بی‌فایده، جدا کردن راس‌هایی با کم‌ترین درجات می‌باشد.

مسئله‌ی افراز یک گراف به دسته مسائل NP-سخت[۵۲] تعلق دارد و تاکنون اگوریتمی با زمان چندجمله‌ای برای حل آن ارائه نشده است، اما روش‌هایی وجود دارند که با پیچیدگی زمانی چندجمله‌ای، به یک جواب نسبتاً مناسب اما غیر بهینه می‌رسند[۱۵]. بسیاری از روش‌ها با دو نیم‌کردن گراف به افراز گراف می‌پردازند و در صورتی که به بیش از دو دسته نیاز باشد، دو نیم کردن روی دسته‌های حاصل تکرار می‌شود. در بیش‌تر روش‌ها

[45] Graph Partitioning
[46] Hierarchical Clustering
[47] Partitional Clustering
[48] Spectral Clustering
[49] Divisive Algorithms
[50] Modularity Based Methods
[51] Cut Size
[52] NP-Hard

شرط تساوی اندازه‌ی دو دسته اعمال می‌شود. مسئله‌ی معرفی شده، دو نیم‌کردن کمینه[۵۳] نام دارد و NP-سخت است.

در ادامه به معرفی الگوریتم کرنیگان-لین[۵۴] [۱۶] به عنوان یک نمونه از این دسته روش‌ها خواهیم پرداخت. این روش به طور کلی معیار سود Q را بیشینه می‌کند. Q نشان‌دهنده‌ی اختلاف تعداد یال‌های درون هر انجمن و تعداد یال‌های بین آن‌ها است. نقطه‌ی شروع روش، یک تقسیم بندی اولیه از راس‌های گراف به دو دسته است که می‌تواند تصادفی بوده یا از یک اطلاع اولیه از ساختار گراف حاصل شده باشد. سپس در هر مرحله، مجموعه-های هم اندازه‌ای از راس‌های بین دو گروه جابجا می‌شوند به طوری‌که بیش‌ترین افزایش ممکن در مقدار Q به-دست آید. برای کاهش ریسک گیر افتادن در بیشینه‌ی محلی، تغییراتی با کاهش مقدار Q هم بعضاً مجاز می-باشند. اجرای این روش دارای پیچیدگی زمانی $O(n^2 logn)$ است که در آن n تعداد راس‌های گراف است.

۳-۲-۳ خوشه‌بندی سلسله مراتبی

در حالت کلی، از قبل اطلاعات زیادی از ساختار انجمنی گراف از جمله تعداد انجمن‌ها در دست نیست. بنابراین روش‌هایی مانند افراز گراف به ندرت ممکن است که مفید واقع شوند. از طرف دیگر، بسیاری از گراف‌ها ممکن است دارای ساختار سلسله‌مراتبی باشند؛ به این معنی که هر انجمن خود از چندین انجمن قابل تشخیص دیگر تشکیل شده باشد. شبکه‌های اجتماعی[۵۵] نمونه‌هایی از گراف‌هایی با ساختار سلسله‌مراتبی هستند. در چنین شرایطی روش‌های خوشه‌بندی سلسله‌مراتبی[۱۷] می‌توانند به کار برده شوند.

نقطه‌ی شروع هر روش خوشه‌بندی سلسله مراتبی، تعیین معیاری برای شباهت هر زوج راس است، مستقل از این‌که به هم متصل هستند یا نه. بعد از این‌که یک معیار برای شباهت راس‌ها انتخاب شد، ماتریس مشابهت $X_{n\times n}$ محاسبه می‌شود.

[53] Minimum Bisection
[54] Kernighan-Lin Algorithm
[55] Social Networks

روش‌های خوشه‌بندی سلسله مراتبی را می‌توان به دو دسته‌ی کلی تقسیم کرد: الگوریتم‌های تراکمی[۵۶] که در آن خوشه‌های مشابه به صورت تکراری ترکیب می‌شوند و الگوریتم‌های تقسیمی[۵۷] که خوشه‌ها با برداشتن یال-هایی که راس‌های نامشابه را به هم وصل می‌کنند، تقسیم می‌شوند.

این دو دسته از نظر شکل فرایند، کاملا برعکس هم هستند. روش‌های تراکمی از پایین به بالا هستند و الگوریتم از راس‌های جدا به عنوان دسته‌های اولیه شروع می‌شود. اما روش‌های تقسیمی از بالا به پایین می-باشند و در شروع کل گراف به عنوان تنها دسته در نظر گرفته می‌شود. از آن‌جایی‌که روش‌های تقسیمی در گذشته به ندرت به کار رفته‌اند، بیش‌تر بر روش‌های تراکمی تمرکز خواهیم داشت.

در روش‌های تراکمی دسته‌هایی با بیش‌ترین میزان شباهت با هم ترکیب می‌شدند، بنابراین نیاز به وجود معیاری برای تعیین میزان شباهت دو دسته خواهیم داشت. در روش‌های خوشه‌بندی اتصال تک[۵۸] و خوشه‌بندی اتصال کامل[۵۹] به ترتیب مقدار کمینه و بیشینه‌ی $x_{i,j}$ انتخاب می‌شود که i در یک دسته و j در دسته‌ی دیگر باشد. در روش خوشه‌بندی اتصال میانگین[۶۰] میانگین تمام این مقادیر محاسبه می‌شود.

روش خوشه‌بندی سلسله مراتبی دارای این مزیت است که نیازی به مشخص نمودن تعداد انجمن‌ها از قبل ندارد. اما متقابلاً این روش دارای نقاط ضعف نیز هست. ممکن است ساختار سلسله مراتبی به دست آمده غیر واقعی باشد؛ به این دلیل که گراف مربوطه اصلاً دارای ساختار سلسله مراتبی نباشد. مشکل دیگر این روش، خوشه‌بندی جدای راس‌هایی با تنها یک همسایه است که در بسیار از شرایط معقول نیست. نهایتاً مشکل اصلی این روش پیچیدگی زمانی آن است که در حالت اتصال تک $O(n^2)$ و در حالت اتصال کامل و اتصال میانگین،

[56] Agglomerative Algorithms
[57] Divisive Algorithms
[58] Single Linkage Clustering
[59] Complete Linkage Clustering
[60] Average Linkage Clustering

$O(n^2 logn)$ می‌باشد. در شرایطی که تعریف شباهت در گراف بدیهی نبوده و این محاسبه هزینه‌بر باشد، پیچیدگی زمانی الگوریتم سنگین‌تر نیز خواهد بود.

۳-۲-۴ خوشه‌بندی افرازی

در این روش، هر راس از گراف به صورت نقطه‌ای در فضا در نظر گرفته می‌شود و تلاش می‌شود داده‌ها بر اساس فاصله‌ی آن‌ها از یکدیگر دسته‌بندی شوند. تعداد خوشه‌ها از قبل برابر با k مفروض است. معیار فاصله، مشخص-کننده‌ی میزان عدم شباهت بین نقاط می‌باشد. هدف جداسازی نقاط در k خوشه است به طوری که تابع هزینه-ای از فاصله‌ی نقاط یک خوشه از هم یا از مرکز جرم خوشه، کمینه شود.

چهار تابع هزینه‌ای که به طور عمده در این زمینه به کار می‌روند، عبارتند از: ۱) k-خوشه‌بندی کمینه[۶۱] که در آن تابع هزینه، قطر خوشه‌ها می‌باشد که به صورت بیش‌ترین فاصله‌ی بین دو عضو یک خوشه تعریف می-شود. ۲) مجموع k-خوشه‌بندی[۶۲] که در آن تابع هزینه به صورت میانگین فاصله‌ی بین همه‌ی زوج نقاط یک خوشه تعریف شده است. ۳) تابع هزینه‌ی k-مرکز[۶۳] که به ازای هر خوشه یک مرکز تعریف شده و هزینه‌ی یک خوشه برابر با بیشینه‌ی فاصله‌ی بین نقاط با مرکز در نظر گرفته می‌شود. ۴) تابع هزینه‌ی k-میانه[۶۴] که مشابه k-مرکز است با این تفاوت که بیشینه‌ی فاصله‌ی نقاط خوشه با نقطه‌ی مرکزی، به‌جای میانگین فاصله محاسبه می‌شود. اما معیاری که تاکنون بیش از معیارهای دیگر به کار برده شده است، k-میانگین[۶۵] می‌باشد که از رابطه-ی زیر به‌دست می‌آید[۳]:

$$Cost = \sum_{i=1}^{k} \sum_{x_j \in S_j} ||x_j - c_i||^2 \qquad (۱-۳)$$

[61] Minimum k-Clustering
[62] K-Clustering Sum
[63] K-Center
[64] K-Median
[65] K-Means

حل مسئله‌ی k-میانگین به راحتی توسط یک روش تکراری قابل انجام است. به این صورت که در مرحله‌ی اول به هر نقطه یک خوشه‌ی تصادفی نسبت داده شده و در هر مرحله مرکز جرم هر خوشه محاسبه شده و خوشه‌ها به‌روز شوند. برای انجام بهینه‌سازی با توجه به توابع هزینه‌ی فوق، نیاز به تعریف معیاری از فاصله خواهیم داشت. معیارهایی که در ادامه آمده است که از روابط همسایگی استفاده می‌کنند[۳]:

$$d_{ij} = \sqrt{\sum_{k \neq i,j} (A_{ik} - A_{jk)}^{\ 2}} \qquad (۲-۳)$$

همچنین می‌توان از معیار همپوشانی[۶۶] برای به‌دست‌آوردن فاصله استفاده کرد[۳]:

$$\omega_{ij} = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} \qquad (۳-۳)$$

در این رابطه، $\Gamma(i)$ مجموعه راس‌های همسایه‌ی راس i می‌باشد. از روی معیار همپوشانی می‌توان فاصله را به صورت رابطه‌ی (۳-۴) تعریف نمود[۳].

$$d_{ij} = 1 - \omega_{ij} \qquad (۴-۳)$$

ضعف عمده‌ی روش خوشه‌بندی افرازی، وابستگی آن به پارامتر k می‌باشد، که در عمل مشخص کردن این پارامتر در بسیاری از دامنه‌ها ممکن نیست. همچنین نیاز به معیاری برای تشخیص فاصله از دیگر مسائل پیش-روی این روش است، چراکه یک معیار ممکن است برای یک گراف مناسب و برای دیگری نامناسب باشد.

۳-۲-۵ خوشه‌بندی طیفی

ایده‌ی اصلی این روش الهام گرفته شده از برخی از ویژگی‌های جالب ماتریس لاپلاسین گراف G می‌باشد. در واقع این روش یک عضو خاص از خانواده روش‌های خوشه‌بندی افرازی است که در قسمت ۳-۲-۳ توضیح داده

[66] Overlap

شد. اگر فرض کنیم D، یک ماتریس $n \times n$ برای نمایش درجه راس‌های گراف و W یک ماتریس $n \times n$ برای وزن‌های گراف باشد، ماتریس L را به صورت زیر تعریف می‌کنیم:

$$L = D - W \qquad (۵-۳)$$

به این ماتریس، ماتریس لاپلاسین نرمال‌نشده گفته می‌شود. بردارهای ویژه و مقادیر ویژه‌ی ماتریس فوق خواص جالبی دارند. دو مورد از ویژگی‌های مقادیر ویژه ماتریس لاپلاسین به این شرح می‌باشد: اولاً مقادیر ویژه‌ی این ماتریس حقیقی و نامنفی هستند. ثانیاً دقیقاً به تعداد مولفه‌های همبندی گراف، مقادیر ویژه‌ی گراف برابر صفر می‌باشند. فرض کنید k بردار ویژه‌ی متناظر با k کوچک‌ترین مقادیر ویژه‌ی بردار را انتخاب کرده و با قراردادن آن‌ها به عنوان ستون، ماتریس U را تشکیل دهیم. با فرض این‌که k مولفه‌ی همبندی در گراف داشته باشیم، k بردار ویژه‌ی انتخاب شده متناظر با مقادیر ویژه‌ی صفر هستند و همه‌ی راس‌هایی در ماتریس U که متناظر با یک مولفه‌ی همبندی باشند، به یک بردار k تایی یکسان نگاشت خواهند شد. اما در صورتی که گراف همبند باشد و از k زیرگراف که با درجه‌ی ضعیفی به یکدیگر متصل شده باشند، تشکیل شده باشد، یک مقدار ویژه برابر صفر خواهد شد و $k-1$ مقادیر ویژه‌ی دیگر نزدیک به صفر خواهند بود. کوچک بودن مقادیر ویژه به این معنی است که مقادیر در سطرهای بردار ویژه‌ی متناظر نزدیک به هم خواهند بود. در نتیجه سطرهای متناظر با این راس‌ها در این ستون‌ها مقادیر نزدیکی خواهند داشت. پس می‌توانیم با استفاده از یک روش خوشه‌بندی مانند k-میانگین راس‌ها را در فضای k بعدی ماتریس U، خوشه‌بندی کنیم[۱۸]. در الگوریتم ۲، شبه‌کدی برای رویه‌ی ذکر شده، دیده می‌شود.

استفاده از این روش همراه چالش‌های خاصی هم خواهد بود. مسئله‌ی اول وابستگی روش به پارامتر k (تعداد انجمن‌ها) می‌باشد که در بسیاری از دامنه‌ها تعیین این مقدار از پیش، کار چندان ساده‌ای نیست. چالش بعدی مربوط به این است که ممکن است نقاط مرزی خوشه‌ها به درستی به خوشه‌ها انتساب نیابند. علت این امر استفاده از روش k-میانگین است که نقاط را به خوشه‌ای تناظر می‌دهد که فاصله‌ی کم‌تری از مرکز خوشه

داشته باشد، که ممکن است لزوماً منجر به انتساب درست نشود. مشکل نهایی این روش، بار محاسباتی نسبتاً سنگین آن برای محاسبه‌ی مقادیر ویژه می‌باشد. برای ماتریس‌های با بعد بزرگ‌تر از پنج، الگوریتم سریعی برای محاسبه‌ی مقادیر و بردارهای ویژه وجود ندارد و روش‌های موجود روش‌های تقریبی عددی هستند. از آن‌جایی که گراف گذر مورد استفاده در یادگیری تقویتی گراف خلوت می‌باشد، می‌توان از روش‌های بهینه‌ای از جمله روش لانکز[۶۷] برای محاسبه‌ی مقادیر و بردارهای ویژه استفاده کرد. هزینه‌ی زمانی این روش $O(n^3)$ می‌باشد.

الگوریتم ۲: خوشه‌بندی طیفی

ماتریس لاپلاسین را با استفاده از رابطه‌ی (۳-۵) برای گراف تشکیل بده.
k بردار ویژه‌ی اول ماتریس فوق را محاسبه کن.
با قرار دادن بردارهای ویژه به عنوان ستون‌ها، ماتریس U را تشکیل بده.
n نقطه‌ی k بعدی متناظر با n سطر ستون در نظر بگیر.
نقاط به‌دست آمده را با استفاده از الگوریتم k-میانگین خوشه‌بندی کن.

۳-۲-۶ روش‌های تقسیمی

روش‌های تقسیمی، دسته روش‌هایی هستند که برای پیدا کردن انجمن‌ها، یال‌هایی که بین انجمن‌ها قرار می‌گیرند را پیدا کرده و حذف می‌کنند و پس از چندین مرحله انجام این وظیفه، آن‌چه باقی می‌ماند انجمن‌های موجود در گراف می‌باشد.

معروف‌ترین روش این دسته، روش ارائه شده توسط نیومان و گیروان[۶۸] است[۱۹]. در این روش، یال‌ها بر اساس معیاری به نام مرکزیت[۶۹] که به نوعی بیان‌گر اهمیت یال مورد نظر است، انتخاب می‌شوند. مراحل این روش در الگوریتم ۳ آمده است:

[67] Lanczos
[68] Newman and Girvan
[69] Centrality

الگوریتم ۳: روش تقسیمی نیومان و گیروان

مرکزیت را برای همه‌ی یال‌ها پیدا کن.
تا زمانی که شرط پایان نرسیده باشد، تکرار کن.
یال با بیش‌ترین مرکزیت را خذف کن. در صورت وجود چند یال بیشینه، یکی را به صورت تصادفی حذف کن.
مرکزیت را برای گراف باقی‌مانده محاسبه کن.

برای محاسبه‌ی مرکزیت یالی، از مفهوم بینابینی یالی[۷۰] استفاده شده است. بینابینی یالی برای یال e برابر است با تعداد کوتاه‌ترین مسیرهایی که از e می‌گذرد. مشخصاً بینابینی یالی برای یال‌هایی که بین دو انجمن واقع شده‌اند، مقدار بزرگی خواهد بود، چراکه بسیاری از کوتاه‌ترین مسیرهایی که بین دو راس از دو انجمن متفاوت قرار دارند، از آن‌ها می‌گذرند. بینابینی یالی همه‌ی یال‌ها در یک گراف خلوت می‌تواند با پیچیدگی زمانی $O(n^2)$ محاسبه شود، در نتیجه، هزینه‌ی نهایی انجام این الگوریتم $O(n^3)$ خواهد بود[۳].

۳-۲-۷ روش‌های مبتنی بر پیمانگی

الگوریتم‌های موفق در زمینه اکتشاف انجمن‌ها، افراز مناسبی از راس‌ها ارائه می‌دهند. سوالی که در این‌جا ممکن مطرح شود، این است که چه معیاری برای میزان تناسب یک افراز وجود دارد؟ در صورتی که تابعی، در حالت کلی، نشان دهنده‌ی میزان مناسب بودن یک افراز باشد، این تابع، تابع کیفیت[۷۱] نامیده می‌شود. تاکنون توابع کیفیت متفاوتی بررسی و پیشنهاد شده‌اند. بیش‌تر این توابع جمع‌کننده[۷۲] هستند. این توابع کیفیت از رابطه‌ی (۳-۶) پیروی می‌کنند[۳]:

[70] Edge Betweenness
[71] Quality Function
[72] Additive Functions

$$Q(P) = \sum_{C \in P} q(C) \qquad (۶-۳)$$

که P یک افراز برای گراف، C یک انجمن تشخیص داده شده از این گراف و $q(C)$ یک تابع برازش برای انجمن C می‌باشد. در ادامه مثال‌هایی از تابع برازش انجمن C [۳] آمده است:

$$\delta_{int}(C) = \frac{\#\ of\ internal\ edges\ of\ C}{n_c\ (n_c - 1)} \qquad (۷-۳)$$

رابطه‌ی (۳-۷) مربوط به چگالی درون انجمن[۷۳] می‌باشد و در آن n_c نشان دهنده‌ی تعداد راس‌های C است. همچنین از رابطه‌ی (۳-۸) می‌توان برای محاسبه‌ی معیار چگالی نسبی[۷۴] استفاده کرد.

$$\rho(C) = \frac{\#\ of\ internal\ edges\ of\ C}{\#\ of\ total\ edges\ of\ C} \qquad (۸-۳)$$

البته تمام توابع کیفیت جمع شونده نیستند. به عنوان مثال تابع کیفیت زیر موسوم به کارایی، غیرجمع شونده است.

$$P(P) = \frac{\left|\{(i,j) \in E,\ C_i = C_j\}\right| + \left|\{(i,j) \notin E,\ C_i \neq C_j\}\right|}{n(n-1)/2} \qquad (۹-۳)$$

مثال دیگر تابع پوشایی[۷۵] است:

$$C(P) = \frac{\left|\{(i,j) \in E,\ C_i = C_j\}\right|}{n(n-1)/2} \qquad (۱۰-۳)$$

محبوب‌ترین تابع کیفیت، پیمانگی است که توسط نیومان و گیروان[۲۰] ارائه شده است. این تابع بر این فلسفه استوار است که گراف‌های تصادفی ساختار انجمنی ندارند و در نتیجه، وجود ساختار انجمنی توسط مقایسه‌ی چگالی واقعی یال‌ها در زیرگراف با چگالی یال مورد انتظار در صورتی که ساختار انجمنی موجود نباشد (مدل تهی[۷۴])، به‌دست می‌آید.

[73] Intra Cluster Density
[74] Null Model

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij})\delta(C_i, C_j) \quad (۱۱-۳)$$

در رابطه‌ی فوق A ماتریس مجاورت گراف، P_{ij} بیانگر امید ریاضی تعداد یال‌های بین راس i و j در مدل تهی و $\delta(C_i, C_j)$ تابعی است که در صورتی که i و j در یک انجمن قرار بگیرند، مقدار یک و در غیر این‌صورت مقدار صفر می‌گیرد.

برای مدل‌سازی مدل‌تهی، یک راه ثابت نگه‌داشتن دنباله‌ی درجات است. با این فرض که k_i نشان‌دهنده‌ی درجه‌ی راس i باشد، خواهیم داشت:

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(C_i, C_i) \quad (۱۲-۳)$$

مقدار بهینه‌ی پیمانگی، نشان‌دهنده‌ی یک افراز با کیفیت بسیار مناسب خواهد بود. بهینه‌سازی پیمانگی یکی از رایج‌ترین روش‌های کشف انجمن در گراف می‌باشد. جستجوی همه‌جانبه با توجه به بزرگی فضا میسر نخواهد بود، چرا که در حالت کلی، بهینه‌سازی پیمانگی ، یک مسئله‌ی NP -تمام[۷۵] است. به این علت، روش‌های بهینه-سازی تقریبی برای حل این مسائل به کار می‌روند.

یک روش تقریبی بهینه‌سازی، به‌کارگیری یک الگوریتم حریصانه است[۲۱]. در این الگوریتم، در ابتدا هر راس به تنهایی به عنوان یک انجمن در نظر گرفته می‌شود. سپس در هر دور از اجرا، دو انجمنی که ترکیب آن‌ها بیش‌ترین افزایش را به میزان پیمانگی می‌دهد، انتخاب شده و ترکیب می‌شوند.

در هر مرحله از اجرای الگوریتم کافی است ΔQ برای تمام یال‌های موجود محاسبه شود. زیرا ادغام دو انجمن که یالی در میان ندارند، قطعاً میزان پیمانگی را افزایش نمی‌دهد. پس هر مرحله از اجرای آن، دارای پیچیدگی زمانی $O(m+n)$ خواهد بود. از آن‌جا که این الگوریتم حداکثر n مرحله اجرا خواهد شد، اجرای آن منجر به هزینه‌ی کل $O(n(n+m))$ یا $O(n^2)$ در گراف خلوت خواهد شد.

[75] NP-Complete

۳-۳ مروری بر برخی از روش‌های انتزاع زمانی

در بخش پیشین، به بررسی برخی از روش‌های کشف انجمن پرداخته شد که می‌تواند به عنوان پایه‌ای برای روش‌های انتزاع زمانی به کار برده شود. در ادامه به چند روش معروف از دسته روش‌های انتزاع زمانی خواهیم پرداخت. ذکر این نکته بی‌فایده نیست که در برخی از این روش‌ها، نتایج روی چند محیط استاندارد از جمله محیط اتاق‌ها، محیط تاکسی، محیط هانوی و محیط اتاق بازی آورده شده است که شرح مفصل این محیط‌ها در بخش ۵-۱ آمده است.

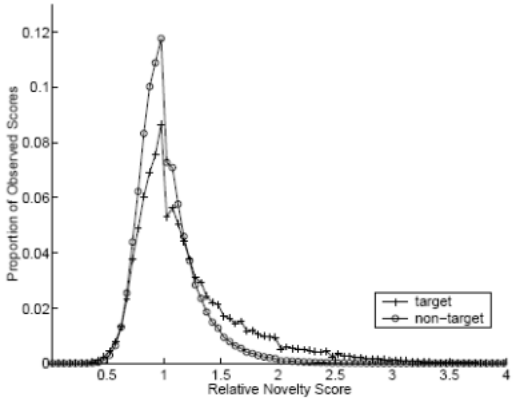
۳-۳-۱ الگوریتم تازگی نسبی

در این روش[۹]، برای کشف زیرهدف‌ها از مفهوم تازگی نسبی[۷۶] استفاده می‌شود. مطابق با تعریف، زیرهدف‌ها حالت‌هایی هستند که عامل را به مناطق دیگری از فضای حالت منتقل می‌کند. ابتدا باید به تعریف مفهوم تازگی[۷۷] بپردازیم: تازگی یک حالت، بیان می‌کند که یک حالت بعد از زمان شروع به چه میزانی دیده شده است. هر چه تکرار ملاقات‌های یک حالت کم‌تر باشد، آن حالت تازه‌تر است. مقدار تازگی برای حالت گسسته‌ی s، برابر با $\frac{۱}{\sqrt{n_s}}$ تعریف می‌شود، که در آن n_s تعداد ملاقات حالت s از زمان شروع است. همچنین این مفهوم برای مجموعه حالت‌های S، با رابطه‌ی $\frac{۱}{\sqrt{\overline{n_s}}}$ تعمیم می‌یابد. در این رابطه، $\overline{n_s}$ میانگین تازگی حالت‌های مجموعه‌ی S می‌باشد. تازگی نسبی یک حالت در یک مسیر، به صورت نسبت تازگی مجموعه حالت‌هایی که بعد از آن می‌آیند (شامل خود آن حالت) به تازگی مجموعه حالت‌های قبل از آن، تعریف شده است.

از دیدگاه شهود برمی‌آید که توزیع تازگی نسبی حالت‌های زیر هدف، متفاوت از حالت‌های دیگر خواهد بود. به صورت دقیق‌تر، انتظار می‌رود که حالت‌های زیر هدف، با تکرار بیش‌تری مقادیر تازگی نسبی بالاتر را داشته

[76] Relative Novely
[77] Novelty

باشند. شکل (۳-۱) میزان تازگی نسبی برای حالت‌های زیر هدف و دیگر حالت‌ها را در محیط دو اتاقه نشان می‌دهد.



شکل (۳-۱): مقایسه‌ی توزیع تازگی نسبی دو حالت هدف و غیرهدف[۹]

برای طبقه‌بندی حالت‌ها به دو دسته‌ی زیرهدف‌ها و دیگر حالت‌ها، از نظریه‌ی تصمیم‌گیری بیز[۷۸] استفاده شده است. این نظریه‌ی تصمیم‌گیری، برای کمینه‌سازی میزان کل هزینه، شامل هزینه‌ی تشخیص یک حالت غیرهدف به عنوان حالت هدف (λ_{fa}) و هزینه‌ی تشخیص ندادن یک حالت هدف (λ_{miss}) بکار می‌رود. در این روش، حالت s به عنوان هدف شناخته می‌شود، اگر:

$$\frac{P\{(s_1,\dots,s_n)|T\}}{P\{(s_1,\dots,s_n)|N\}} > \frac{\lambda_{fa}}{\lambda_{miss}}\frac{P\{N\}}{P\{T\}} \qquad (۱۳-۳)$$

در این رابطه، $(s_1,\dots,s_n)$ مقادیر تازگی نسبی حالت s و $P\{(s_1,\dots,s_n)|T\}$ و $P\{(s_1,\dots,s_n)|N\}$، به ترتیب برابر احتمال شرطیِ مقادیر تازگی نسبی، به شرط هدف بودن یا نبودن آن حالت است. همچنین $P\{T\}$ و $P\{N\}$، احتمال پیشین[۷۹] هدف بودن یا نبودن حالت‌ها می‌باشند.

[78] Bayes Decision Theory
[79] Prior Probability

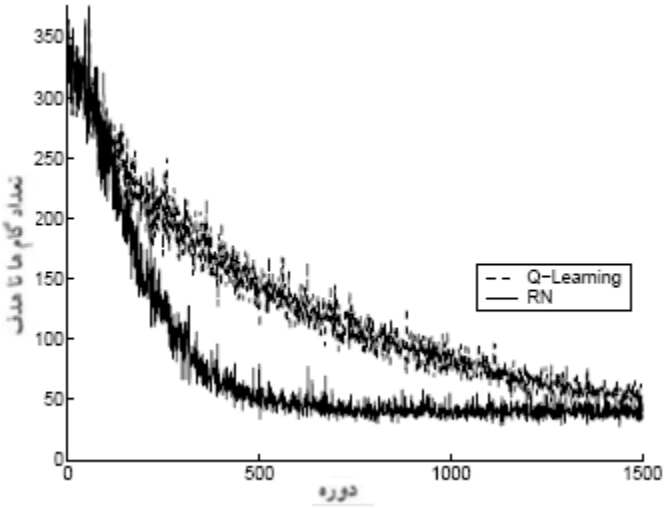
برای سادگی بیش‌تر، فضای پیوسته‌ی تازگی نسبی حالت‌ها به وسیله‌ی آستانه‌گیری به فضای گسسته دودویی تبدیل می‌شود. متغیر تصادفی x حاصل آستانه‌گیری تازگی نسبی با مقدار آستانه‌ی t_{RN} می‌باشد. در این صورت، رابطه‌ی (۳-۱۳) را می‌توان به صورت زیر درآورد:

$$\frac{p^{n_1}(1-p^{n-n_1})}{q^{n_1}(1-q^{n-n_1})} > \frac{\lambda_{fa}}{\lambda_{miss}}\frac{P\{N\}}{P\{T\}} \qquad (۱۴-۳)$$

در این رابطه، p و q به ترتیب، احتمال ۱ بودن مقدار x به شرط زیرهدف بودن و ۱ بودن مقدار x به شرط زیرهدف نبودن حالت است. همچنین n_1 تعداد مشاهداتی است که در آن x برابر ۱ و n تعداد کل مشاهدات می‌باشد. در نهایت، با استفاده از اعمال جبری ساده، قانون تصمیم‌گیری زیر حاصل می‌شود: یک حالت هدف است اگر:

$$\frac{n_1}{n} > \frac{ln\frac{1-q}{1-p}}{ln\frac{p(1-q)}{q(1-p)}} + \frac{1}{n}\frac{ln(\frac{\lambda_{fa}}{\lambda_{miss}}\frac{p(N)}{p(T)})}{ln\frac{p(1-q)}{q(1-p)}} \qquad (۱۵-۳)$$

در شکل (۳-۲) مقایسه‌ای از عملکرد این روش با یادگیری Q مشاهده می‌شود.



شکل (۳-۲): مقایسه‌ی روش تازگی‌نسبی و یادگیری Q در تعداد گام رسیدن تا هدف در محیط تاکسی [۹]

۳-۳-۲ الگوریتم افراز گراف محلی[۸۰]

تعریف زیر هدف در این مسئله، به این صورت است[۱۰]: محل گذر از یک ناحیه به ناحیه‌ی دیگر، که دارای این شرایط باشد: گذر از یک ناحیه به ناحیه‌ی دیگر احتمال کم، اما اکیداً مثبت دارد و بیش‌تر این گذرها از میان تعداد کمی از حالت‌ها می‌گذرند. خاصیت اصلی این روش نه در تعریف زیر هدف، بلکه در نحوه‌ی اکتشاف آن است. اکتشاف زیر هدف‌ها، با ساخت دوره‌ای گراف گذر محلی که نماینده‌ی تعاملات اخیر عامل است، انجام می‌شود. در ادامه، برشی از این گراف با احتمال کم گذر، بین نواحی یافت شده و زیرهدف‌ها به عنوان دو سر یال‌های متناظر با این گذرها پذیرفته می‌شوند. این روش را برای تاکید بر دیدگاه محلی در گراف ساخته شده، به اختصار روش برش L می‌نامند.

برای پیدا کردن حالت‌های زیر هدف، در ابتدا گراف تعاملات محلی ساخته می‌شود، که یک گراف وزن‌دار و جهت‌دار است. وزن هر یال برابر تعداد دفعاتی است که این گذر در یک مسیر قرار گرفته است.

برای یک گراف $G = (V, E)$ برش (A, B) یک افراز روی مجموعه راس‌های V می‌باشد. هدف الگوریتم، کمینه‌سازی معیار برش نرمال شده با تعریف زیر است:

$$NCut(A, B) = \frac{cut(A, B)}{vol(A)} + \frac{cut(B, A)}{vol(B)} \qquad (۱۶-۳)$$

در این رابطه، $cut(A, B)$ مجموع وزن همه‌ی یال‌هایی است که از راسی در A آغاز و به راسی در B می‌روند و $vol(A)$ مجموع وزن همه‌ی یال‌های است که از راسی در A آغاز می‌شوند. اولین جمله از این عبارت برابر با احتمال گذر از حالتی از مجموعه‌ی A به حالتی از مجموعه‌ی B و جمله‌ی دوم احتمال گذر عکس آن می‌باشد. بنابراین در مجموع، معیار $NCut$ بیان‌گر تخمینی احتمال گذر میان برش خواهد بود. کمینه کردن این معیار معادل با پیدا کردن یک برش مناسب است، چراکه برشی با معیار برش نرمال شده‌ی کمینه، برشی است که

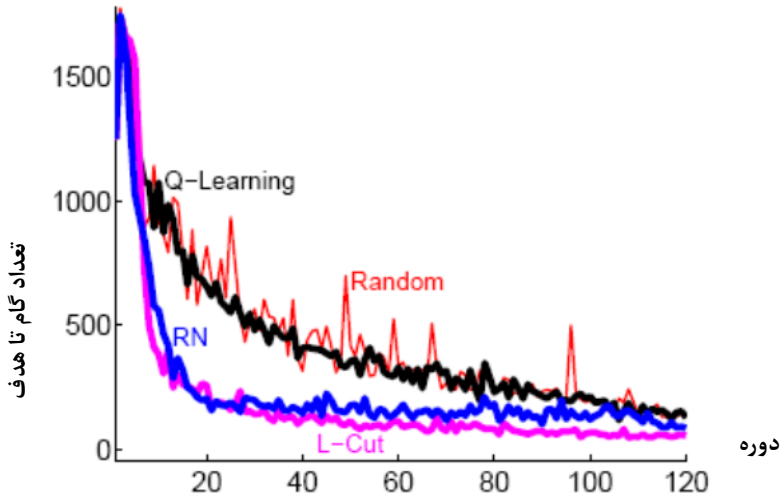
[80] Local Graph Partitioning

احتمال گذر بین حالت‌های درون هر ناحیه‌ی آن زیاد و احتمال گذر بین حالت‌های متعلق به دو ناحیه‌ی متفاوت، کم است و این به معنی یافتن برشی مناسب است.

پیدا کردن یک افراز که معیار فوق را کمینه کند، تنها در زمان نمایی انجام‌پذیر است. به همین دلیل از روش خوشه‌بندی طیفی که می‌تواند برش کمینه را برای تقریبی از معیار $NCut$ پیدا کند، استفاده می‌شود. این معیار تقریب زده شده روی گراف بدون جهت تعریف می‌شود و به شکل زیر می‌باشد:

$$\widehat{NCut}(A,B) = \frac{cut(A,B)+cut(B,A)}{vol(A)+cut(B,A)} + \frac{cut(B,A)+cut(B,A)}{vol(B)+cut(B,A)} \qquad (۱۷-۳)$$

این الگوریتم در زمان $O(N^3)$ اجرا می‌شود، اما باید به این نکته توجه نمود که در الگوریتم برش محلی به علت محلی بودن، N بسیار کوچک‌تر از تعداد حالت‌هاست و اجرای الگوریتم سریع‌تر خواهد بود. در شکل (۳-۳) مقایسه‌ای از الگوریتم برش محلی را با روش یادگیری Q و روش تازگی نسبی دیده می‌شود:



شکل (۳-۳): مقایسه‌ی تعداد گام تا هدف در دوره‌های متفاوت از الگوریتم‌های برش L، یادگیری Q، تازگی نسبی (RN) و یادگیری Q با مهارت‌های تصادفی[۱۰]

۳-۳-۳ روش برش Q

روش برش Q بر مبنای این تعریف اولیه از حالت‌های زیرهدف، بنا شده است[۱۱]: «زیرهدف، گلوگاهی[۸۱] است که حالت‌های مرزی دو ناحیه‌ی با همبندی بالا را به هم پیوند می‌دهد.» پیدا کردن این نقاط گلوگاه، به وسیله‌ی حل کردن مسئله‌ی برش کمینه-شار بیشینه[۸۲] انجام می‌شود. در شروع یک یادآوری بر این مسئله انجام می‌شود.

فرض کنید گراف $G = (V, E)$ موجود باشد. در مسئله‌ی شار بیشینه، سعی می‌شود حداکثر میزان شاری را به‌دست آورده شود که با توجه به ظرفیت هر یال، می‌توان از راس مبدا (s)، به راس مقصد (t)، فرستاد. یک برش $s - t$ مجموعه‌ای از یال‌هاست که با حذف کردن آن، گراف ناهمبند شده و به دو مولفه‌ی همبندی V_s و V_t تقسیم می‌گردد. مسئله‌ی پیدا کردن برشی با حداقل ظرفیت یال‌های برش، به عنوان مسئله‌ی برش کمینه شناخته می‌شود. اثبات می‌شود که این دو مسئله با یکدیگر معادلند[۲۲]. زمان اجرای برنامه، بر اساس تعداد یال‌ها و راس‌ها، چندجمله‌ای است. به صورت دقیق‌تر، پیچیدگی اجرای این الگوریتم $O(n^3)$ می‌باشد[۲۳]. که در آن، n تعداد راس‌های گراف است. در ادامه، شبه‌کدی برای الگوریتم برش Q ارائه می‌گردد:

الگوریتم ۴: برش Q

```
تکرار کن
    با محیط تعامل کن و سابقه‌ی گذر حالت را نگه‌داری کن
    اگر شرط فعال‌سازی ارضا شده است:
        دو حالت از مجموعه‌ی حالت‌ها برای s و t انتخاب کن.
        رویه‌ی برش(s و t) را انجام بده.
```

در الگوریتم ۴، از رویه‌ی برش استفاده شده که در الگوریتم ۵، بسط داده شده است:

[81] Bottleneck
[82] Min Cut-Max Flow Problem

الگوریتم ۵: رویه‌ی برش

ورودی: (s, t)

سابقه‌ی گذر حالت را به گراف تبدیل کن.
برش کمینه V_s و V_t را پیدا کن.
اگر کیفیت برش به اندازه‌ی کافی خوب است.
سیاست بهینه را برای رسیدن به حالت‌های گلوگاه از هر حالت V_s، به‌دست بیاور.

در مورد این الگوریتم، چند نکته را باید مشخص کرد. نخست آن‌که چه زمانی شرط فعال‌سازی ارضا می‌شود. یک گزینه این است که الگوریتم را با یک بسامد ثابت اجرا کنیم. انتخاب دیگر، می‌تواند زمانی باشد که راس‌های s و t انتخاب شده باشند. مسئله‌ی دیگر، نحوه‌ی انتخاب راس‌های s و t است. یک انتخاب منطقی برای آن‌ها، به ترتیب حالت شروع و حالت پایانی است. دو مسئله‌ی دیگری که باید به صورت دقیق مشخص شوند، چگونگی ساخت گراف و معیاری برای میزان مناسب بودن برش هستند.

در مورد اول، هر یال نشان‌دهنده‌ی گذر از حالتی به حالت دیگر است و وزن آن مطابق با رابطه‌ی زیر مشخص می‌شود[۱۱]:

$$c(i,j) = \frac{n(i \to j)}{n(i)} \qquad (۱۸-۳)$$

در رابطه‌ی بالا، $n(i \to j)$ تعدا گذرها از حالت i به j و $n(i)$، تعداد کل گذرها از حالت i می‌باشد. همچنین معیار زیر برای میزان تناسب یک برش V_s و V_t پیشنهاد شده است[۱۱]:

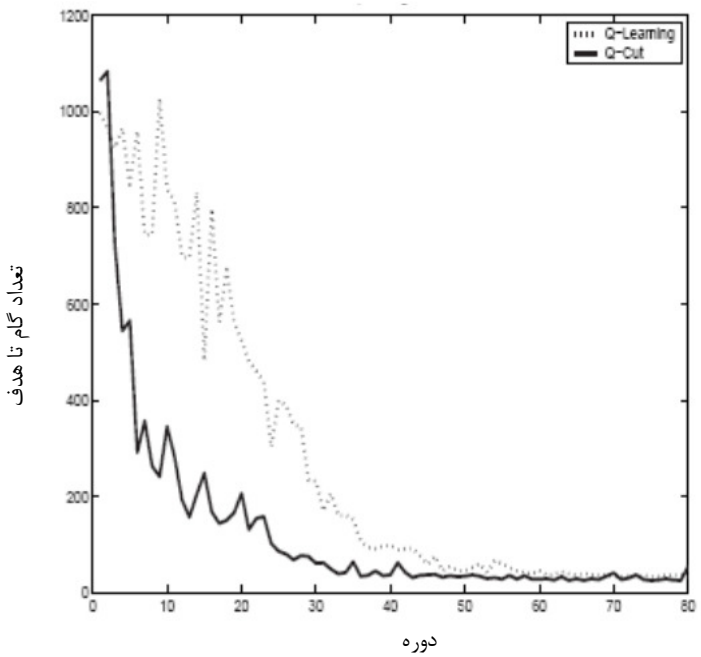
$$Q(V_s, V_t) = \frac{|V_s|\,|V_t|}{A(V_s, V_t)} \qquad (۱۹-۳)$$

در این رابطه، $A(V_s, V_t)$ نشان‌دهنده‌ی تعداد یال‌هایی است که در این برش وجود دارد و $|V_s|$ و $|V_t|$ تعداد راس‌های دو مجموعه‌ی V_s و V_t را نشان می‌دهد.

مطابق با این رابطه، هرچه تعداد یال‌های بین دو مجموعه راس‌ها کم‌تر باشد، این برش بهتر است. از طرفی در صورتی که اندازه‌ی مجموعه‌ی V_s کوچک باشد، پیدا کردن چنین گزینه‌ی کوچکی به اندازه‌ی کافی ارزشمند

نیست و اگر خیلی بزرگ باشد، مسئله‌ی یادگیری سیاست گزینه به اندازه‌ی کافی کوچک نشده و کمکی به مقیاس‌پذیری راه حل نمی‌کند. به همین دلیل، اندازه‌ی این دو مجموعه به صورت حاصل‌ضرب در صورت قرار داده شده‌اند، تا برش‌هایی با اندازه‌ی متناسب ارجحیت داشته باشند.

در شکل (۳-۴) مقایسه‌ای از تعداد گام‌های رسیدن تا هدف، برای دو روش برش Q و یادگیری Q در محیط دو اتاقه ملاحظه می‌فرمایید.



شکل (۳-۴): مقایسه‌ی تعداد گام تا هدف در دوره‌های متفاوت از الگوریتم‌های برش Q، یادگیری Q در محیط دو اتاقه[۱۱]

۳-۳-۴ روش مبتنی بر بینابینی

این الگوریتم[۱۲] نیز یکی دیگر از روش‌های مبتنی بر گراف است که در آن مانند دیگر روش‌های این دسته، ابتدا گراف تعاملات محیط ساخته می‌شود. در صورتی یال $u \to v$ در گراف موجود است، که این گذر حالت احتمال اکیداً مثبت داشته باشد. همچنین وزن این یال، برابر امیدریاضی هزینه‌ی این گذر خواهد بود.

مبنای اصلی روش مذکور، این است که حالت‌هایی که نقش محوری در مسیرهای بهینه در گراف گذر دارند، به عنوان زیرهدف‌های مفید در نظر گرفته می‌شوند. یک معیار مناسب برای میزان محوریت راس v به صورت زیر می‌باشد:

$$\sum_{s\neq t\neq v} \frac{\sigma_{st}(v)}{\sigma_{st}} w_{st} \qquad (۲۰-۳)$$

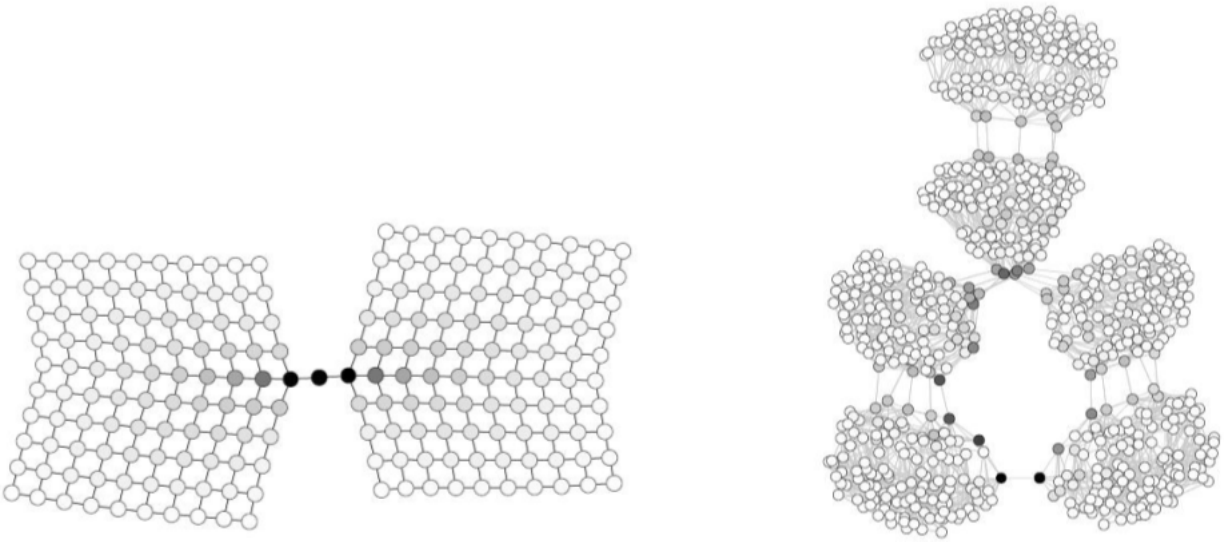
در این رابطه، σ_{st} تعداد کوتاه‌ترین مسیرهای از s به t، $\sigma_{st}(v)$ تعداد کوتاهرین مسیرهای از s به t، که از راس v می‌گذرند و w_{st} وزنی است که به مسیرهای از s به t نسبت داده می‌شود. با وزن‌های یکسان، عبارت بالا دقیقاً برابر معیار بینابینی[۸۳] خواهد بود، که معیاری است برای تعیین میزان مرکزیت یک راس در گراف. در معیار بینابینی، اگر کوتاه‌ترین مسیر یکتا نباشد، وزن آن بین همه‌ی این مسیرها، به صورت مساوی تقسیم می‌شود. محاسبه‌ی بینابینی برای همه‌ی راس‌ها در یک گراف بدون وزن، با پیچیدگی زمانی $O(mn)$ و پیچیدگی مکانی $O(m+n)$ انجام می‌شود[۲۴]. در گراف وزن‌دار، هزینه‌ی مکانی تغییر نمی‌کند، اما پیچیدگی زمانی به $O(mn+n^2 logn)$ افزایش پیدا می‌کند.

در این روش، وزن هر مسیر به میزان پاداشی که آن مسیر به‌دست می‌آورد، بستگی پیدا می‌کند و به این‌صورت ممکن است، بخشی از گراف که تاثیر بیش‌تری در رسیدن عامل به هدف داشته باشد، اهمیت بیش‌تری پیدا کرده و وزن بیش‌تری بگیرد. در نهایت، راس‌هایی به عنوان زیرهدف در نظر گرفته می‌شوند که در بیشینه‌های محلی معیار فوق قرار گیرند. شکل (۳-۵) مقدار بینابینی را برای حالت‌های مختلف در دو محیط متفاوت، نشان می‌دهد. در این شکل، راس‌هایی که تیره‌تر رسم شده‌اند، مقادیر بینابینی بیش‌تری دارند.

برای مقایسه‌ی این الگوریتم با روش‌های دیگر، در شکل (۳-۶) نموداری از تعداد کنش‌های انجام شده تا رسیدن به هدف بر حسب شماره‌ی دوره، به‌روی دو محیط اتاق بازی و محیط دو اتاقه، برای سه الگوریتم رسم شده است. این سه الگوریتم در شکل با نمادهای، $primitives$، $Random$ و $Skills$ مشخص شده‌اند که به

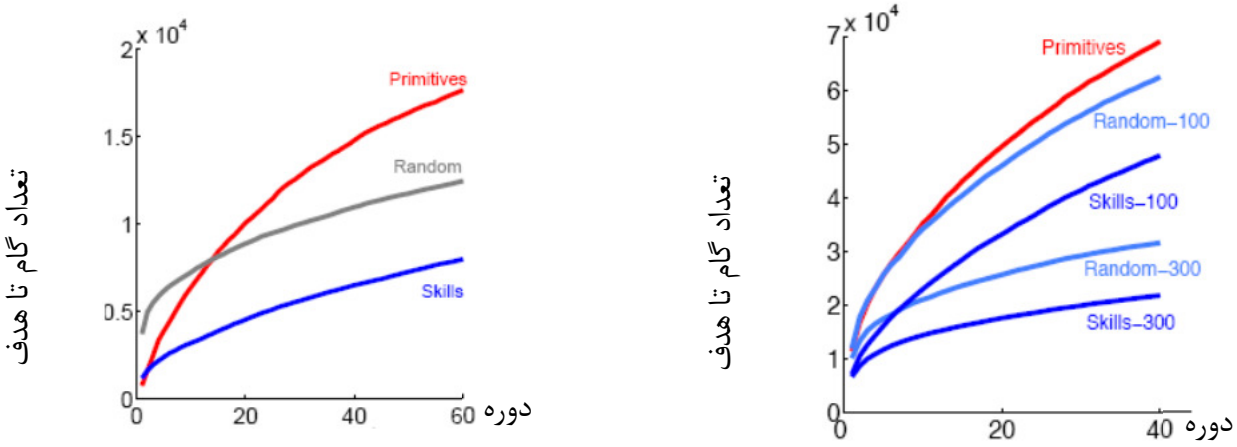
[83] Betweenness

ترتیب متناظرند با یادگیری Q بدون مهارت، یادگیری Q با مهارت‌های تصادفی و نهایتاً یادگیری Q با مهارت-های حاصل از این روش. اعداد ۱۰۰و ۳۰۰ نیز نشان‌دهنده‌ی اندازه‌ی مجموعه‌ی آغازین گزینه است.



(آ) محیط اتاق بازی (ب) محیط دو اتاقه

شکل (۳-۵): زیرهدف‌های به‌دست آمده با استفاده از معیار بینابینی[۱۲]

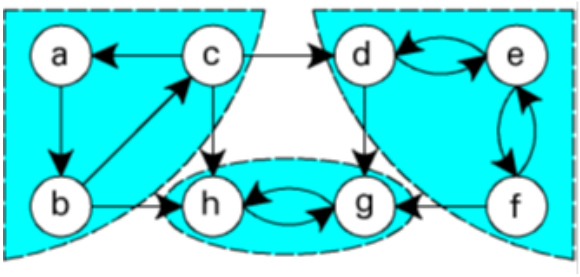


(آ) محیط اتاق بازی (ب) محیط دو اتاقه

شکل(۳-۶): مقایسه‌ی تعداد گام تا هدف در دوره‌های متفاوت از الگوریتم‌های مبتنی بر بینابینی، یادگیری Q و یادگیری Q با مهارت‌های تصادفی در محیط‌های دو اتاقه و اتاق بازی[۱۲]

۳-۳-۵ روش مولفه‌های قویاً همبند

روش مولفه‌های قویاً همبند برگرفته از این تعریف از حالت‌های زیرهدف است: «نقاطی که دو ناحیه با اتصالات درونی بالا را به یکدیگر متصل می‌کنند و علاوه بر این انتقال از طریق این حالت‌ها، از یک ناحیه به ناحیه‌ی دیگر نیز با احتمال بسیار کم صورت می‌گیرد.»[۱] برای رسیدن به ویژگی نخست این تعریف، یک ایده استفاده از تعریف مولفه‌های قویاً همبند است، که در یک گراف جهت‌دار، به مجموعه‌ی بیشنه‌ای از راس‌ها گفته می‌شود که به ازای هر دو راس u و v که عضو این مجموعه باشند، مسیری از u به v و همچنین مسیری از v به u موجود باشد. در شکل (۳-۷) افرازی از یک گراف به مولفه‌های قویاً همبند، ارائه شده است:



شکل (۳-۷): افرازی از یک گراف به مولفه‌های قویاً همبند[۱]

در تعریف بالا، کم بودن احتمال گذر از یک ناحیه به ناحیه‌ی دیگر به عنوان ویژگی دیگری از حالت‌های زیرهدف ذکر شد. این ویژگی کمک قابل توجهی در جهت به‌دست آمدن ساختار فضای حالت می‌کند، به این صورت که می‌توان برای جداسازی نواحی گراف فضای حالت، یال‌هایی با تعداد گذر کم‌تر از یک حد آستانه (t_t) را حذف نمود تا به این ترتیب یال‌های گذر میان نواحی برداشته شوند. بعد از این آستانه‌گیری، با حذف مسیرهای میان حالت‌های نواحی متفاوت، مولفه‌های قویاً همبند گراف، توصیف مناسب‌تری از نواحی فضای حالت خواهند بود.

مولفه‌های قویاً همبند یک گراف، معادل با یک افراز از راس‌های گراف می‌باشد، زیرا هر راس از گراف، حتماً عضو یک مولفه‌ی قویاً همبند خواهد بود، حتی اگر این مولفه، تک عضوی باشد. به این وسیله، می‌توان افرازی از

راس‌های گراف به انجمن‌هایی با اتصال درونی بالا، پیدا کرد. چندین الگوریتم برای پیدا کردن مولفه‌های قویاً همبند ارائه شده است، که در ادامه یکی از آن‌ها را بررسی می‌کنیم. این الگوریتم از یک نکته‌ی بسیار ساده استفاده می‌کند و آن نکته این است که مولفه‌های قویاً همبند یک گراف جهت‌دار، با مولفه‌های قویاً همبند معکوس آن یکسان می‌باشد. در ادامه، الگوریتم پیدا کردن مولفه‌های قویاً همبند، توضیح داده می‌شود.

ابتدا جستجوی عمق اول[۸۴] روی گراف G انجام می‌شود تا زمان پایان[۸۵] هر یک از راس‌ها به‌دست آید. سپس گراف معکوس (G^T) حاصل شده و این‌بار جستجوی عمق اول، روی گراف معکوس اجرا می‌شود. برای شروع جستجو، راسی با بیش‌ترین زمان پایان انتخاب می‌شود. در این مرحله، تعدادی از راس‌ها ملاقات می‌شوند، اما ممکن است بعضی از آن‌ها ملاقات‌نشده باقی بماند. در این‌صورت مجدداً جستجوی عمق اول با شروع از بیش-ترین زمان‌پایان روی راس‌های باقی‌مانده اجرا می‌شود و این روند ادامه پیدا می‌کند تا همه‌ی راس‌ها ملاقات شوند. نشان داده شده است که هر یک از زیردرخت‌های به‌دست آمده در جستجوی عمق اول نهایی، مولفه‌های قویاً همبند گراف G می‌باشند[۲۵]. در الگوریتم ۶، شبه‌کدی برای این روش، نمایش داده شده است.

الگوریتم ۶: بدست آوردن مولفه‌های قویاً همبند

ورودی: (G)

$DFS(G)$ را فراخوانی کن تا زمان پایان $f(u)$ برای همه‌ی راس‌ها محاسبه شود.
G^T را محاسبه کن.
$DFS(G^T)$ را فراخوانی کن، با این تفاوت که در حلقه‌ی اصلی تابع، همسایه‌ها را با اولویت برای $f(u)$های بیشتر، ملاقات کن.
درخت‌های شکل گرفته در جنگل حاصل را به عنوان مولفه‌های قویاً همبند، در نظر بگیر.

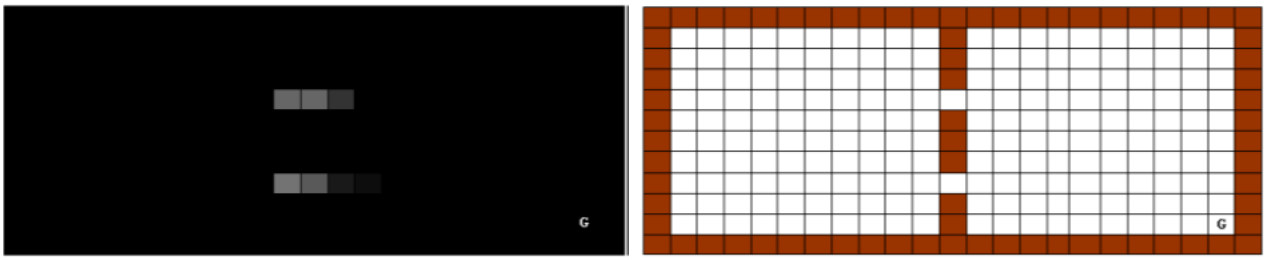
---

[84] Depth First Search
[85] Finishing Time

مزیت این روش، سرعت قابل توجه اجرای آن می‌باشد. در صورتی‌که برای پیاده‌سازی گراف گذر، از لیست مجاورت استفاده شود، به‌دست آوردن مولفه‌های قویاً همبند گراف، هزینه‌ی خطی براساس تعداد راس‌ها (n) و یال‌ها (m) خواهد داشت. به صورت دقیق‌تر، پچیدگی الگوریتم $O(n+m)$ می‌باشد.

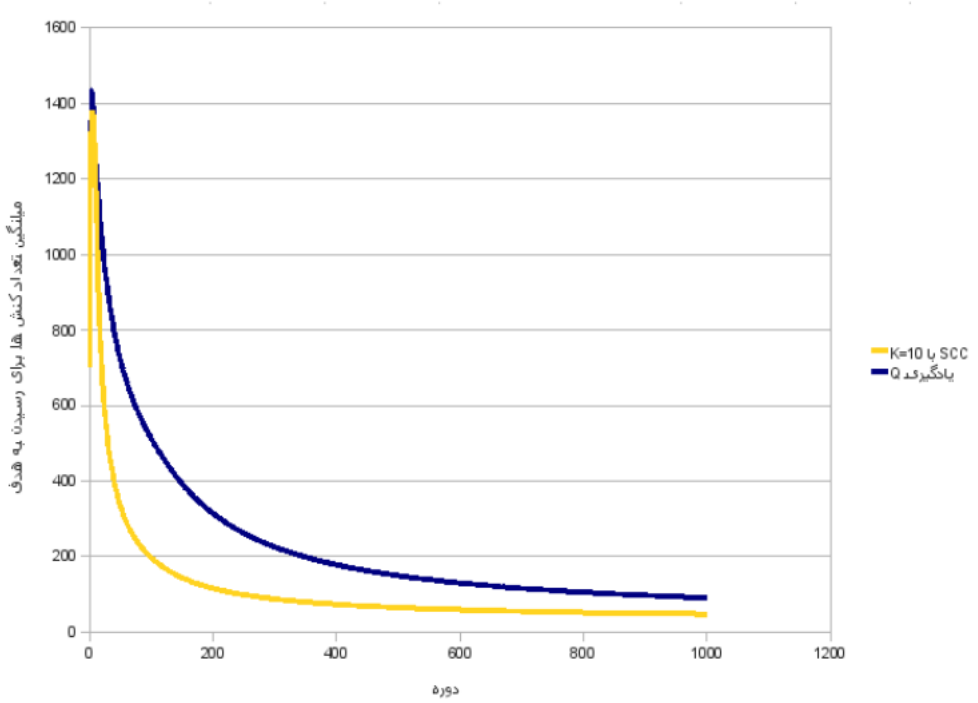
همان‌طور که گفته شد، هر راس در یک مولفه‌ی قویاً همبند قرار خواهد گرفت. اما بهتر است راس‌هایی به عنوان زیرهدف‌ها در نظر گرفته شوند، که نقاط مرزی مولفه‌های همبندی با اندازه‌ی بزرگ‌تری باشند. به عنوان نمونه می‌توان مولفه‌هایی با تعداد راس‌های بیش از حد مشخصی را در نظر گرفت. در [۲۶] مقدار $\mu + ۲\sigma$ برای این حد پیشنهاد شده است، که در آن μ و σ، به ترتیب میانگین و انحراف معیار اندازه‌ی مولفه‌ها می‌باشند.

در ادامه، شکل‌های (۳-۸) و (۳-۹)، به منظور بررسی نتیجه‌ی اعمال این الگوریتم و همچنین مقایسه‌ای از اجرای آن با یادیگری Q، آمده‌اند. شکل (۳-۸) زیرهدف‌های کشف شده در محیط دو اتاقه را نشان می‌دهد. در این شکل، نقاط روشن‌تر، حالت‌های مناسب‌تر برای زیرهدف را نشان می‌دهد.



(آ) محیط دو اتاقه با دو در میانی (ب) زیر هدف‌های کشف شده برای این محیط

شکل (۳-۸): محیط دو اتاقه با دو درب میانی و زیر هدف‌های احتمالی کشف شده[۱]



شکل (۳-۹): مقایسه‌ی تعداد کنش‌ها برای رسیدن به هدف، در روش مولفه‌های قویاً همبند و روش یادگیری Q که روی ۵۰ اجرا میانگین‌گیری شده است[۱].

۳-۳-۶ روش مرکزیت بردار ویژه

در الگوریتم‌های مربوط به مسائل شبکه‌های اجتماعی، معیارهای متنوعی برای اندازه‌گیری میزان محوریت یک راس با توجه به نیازهای مختلف، تعریف شده است. یکی از این معیارها که در فصل گذشته نیز به آن اشاره شد، معیار بینابینی است که میزان محوریت راس‌ها را در مسیرهای بهینه، نمایان می‌کند. یکی دیگر از معیارهای معروفی که روش فوق بر اساس آن بنا شده است، معیار مرکزیت بردار ویژه[۸۶] (EVC) می‌باشد. این معیار برای سنجش میزان قدرت انتشار گره‌ها، معرفی شده است[۲۷]. به صورت دقیق‌تر، این معیار نشان می‌دهد، یک گره، به چه میزان به گره‌های مهم دیگر متصل است. بنابراین طبیعی است که EVC معمولاً در شبکه‌های اجتماعی برای تبیین میزان اهمیت گره‌ها، برای انتشار اخبار، یا منتشر ساختن یک ویروس کامپیوتری به‌کار رود[۲۸].

[86] Eigenvector Centrality

همان‌طور که گفتیم، EVC نشان می‌دهد که یک گره به چه میزان به گره‌های مهم دیگر متصل است، بنابراین در صورتی که e_i را امتیاز یک گره برای این معیار بدانیم، می‌توانیم از رابطه‌ی زیر برای محاسبه‌ی آن استفاده کنیم:

$$e_i = \frac{1}{\lambda} \sum_{j \in N(i)} e_j \quad (۲۱-۳)$$

که در آن $N(i)$ مجموعه‌ی همسایگان راس i و λ یک عدد ثابت است. فرض کنید A ماتریس مجاورت گراف باشد، به‌صورتی که $A(i,j)$ برابر یک است اگر یالی از راس i به راس j وجود داشته باشد و در غیر این‌صورت صفر باشد. در این‌صورت، میزان مرکزیت بردار ویژه‌ی راس‌ها را می‌توان از رابطه‌ی زیر به‌دست آورد:

$$Ae = \lambda e \quad (۲۲-۳)$$

که حل این تساوی، معادل است با حل عبارت زیر:

$$det(A - \lambda I) = 0 \quad (۲۳-۳)$$

حل این معادله n جواب دارد که تحت عنوان مقادیر ویژه‌ی ماتریس A شناخته می‌شوند. با قرار دادن مقادیر ویژه در رابطه‌ی (۳-۲۲) بردارهای ویژه‌ی ماتریس A به‌دست می‌آیند. اگر ماتریس A متقارن باشد، مقادیر بردارهای ویژه‌ی A حقیقی خواهند بود، به همین دلیل، در این مسئله از گراف بدون جهت استفاده می‌شود تا ماتریس A خاصیت تقارنی داشته باشد. برای اطمینان از مثبت بودن همه‌ی مقادیر EVC، از بردار ویژه‌ی اول استفاده می‌شود. بردار ویژه‌ی اول، توسط الگوریتم لانکز با پیچیدگی زمان $O(n^3)$ قابل حصول است[۲۹] که در آن n تعداد راس‌های گراف است.

به این علت که EVC معیاری برای میزان اتصال به گره‌های مهم و نشان‌دهنده‌ی اهمیت یک راس برای انتشار اطلاعات می‌باشد، طبیعی است که راس‌های با بیشنه‌ی EVC راس‌های مرکزی نواحی گراف باشند. همچنین پیش‌بینی می‌شود یک راس دور افتاده، میزان EVC کم‌تری نسبت به سایر راس‌ها داشته باشد.

بنابراین به طور کلی می‌توان نتیجه گرفت که با حرکت از نقاط مرکزی یک خوشه به نقاط مرزی آن، میزان EVC کاهش پیدا کند. با توجه به آن‌چه گفته شد، می‌توان الگوریتمی برای به‌دست‌آوردن خوشه‌ها و در نتیجه حاصل شدن زیرهدف‌ها استخراج کرد. الگوریتم ۷، شبه‌کدی برای این روش ارائه می‌دهد:

الگوریتم ۷: خوشه‌بندی با استفاده از معیار مرکزیت بردار ویژه

ورودی (A)

اولین بردار ویژه ماتریس A را محاسبه کن. داریه‌یiام بردار ویژه، میزان EVC برای راس i ام می‌باشد.

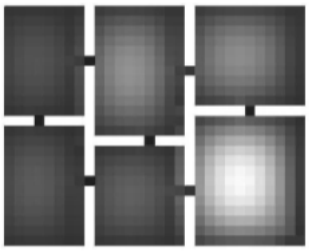
برای هر گره‌ای که بیشیه‌ی محلی EVC را دارد، یک خوشه‌ی جدید ایجاد کن و آن گره را به عنوان گره‌ی مرکزی خوشه در نظر بگیر.

خوشه‌هایی را که مراکز آن‌ها با هم همسایه هستند، با هم ادغام کن.

اگر گره‌ای عضو خوشه‌ی C است و همسایه‌ای با مقدار EVC کم‌تر دارد، آن همسایه را به خوشه‌ی C اضافه کن.

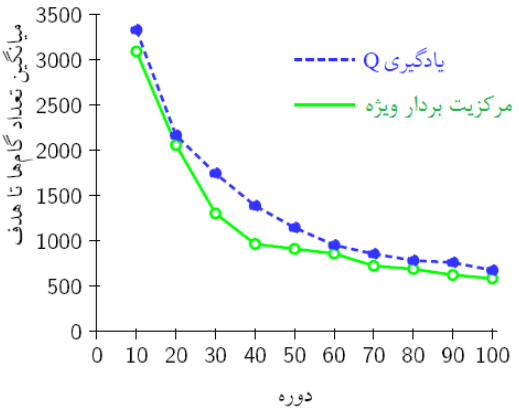
هر گره‌ای را که عضو بیش از یک خوشه است، به عنوان گره مرزی آن خوشه در نظر بگیر.

مطابق با الگوریتم ۵، برای به‌دست آمدن خوشه‌ها، از این قانون اساسی استفاده شده است که اگر i و j همسایه باشند و $EVC(i) > EVC(j)$ و i عضو خوشه‌ی C باشد، j نیز عضو این خوشه خواهد بود. پس از به‌دست آمدن خوشه‌ها می‌توان با توجه به حالت‌های مرزی هر خوشه، زیرهدف‌ها را نیز کشف کرد و مهارت‌هایی برای رسیدن به آن‌ها ساخت. در شکل (۳-۱۰) مقادیر بردار ویژه برای محیط ۶ اتاقه به نمایش در آمده است. در این شکل، نقاط روشن‌تر بیان‌گر حالت‌های با مرکزیت بردار ویژه‌ی بیش‌تر می‌باشد.



شکل (۳-۱۰): مقادیر مرکزیت بردار ویژه، برای محیط شش اتاقه[۲]

همچنین در شکل (۳-۱۱) مقایسه‌ای از تعداد کنش‌های انجام شده تا رسیدن به هدف، برای دو الگوریتم مرکزیت بردار ویژه و یادگیری Q در محیط شش اتاقه انجام شده است.



شکل (۳-۱۱): مقایسه‌ای از تعداد کنش‌های انجام شده تا رسیدن به هدف، برای دو الگوریتم مرکزیت بردار ویژه و یادگیری Q در محیط شش اتاقه[۲]

۳-۴ جمع‌بندی

در این فصل به دسته‌بندی روش‌های یادگیری سلسله مراتبی به کمک انتزاع زمانی پرداختیم. دیدیم که عمده‌ی این روش‌ها، مبتنی بر آنالیز گراف حاصل از مدل‌سازی محیط می‌باشند. بیش‌تر روش‌های ارائه شده دارای سه مرحله‌ی عمده هستند، مدل‌سازی محیط به وسیله‌ی گراف، پیدا کردن نقاط زیرهدف و نهایتاً ساخت مهارت-های برای رسیدن به نقاط زیرهدف.

بیش‌تر روش‌های ارائه شده، از پیدا کردن تمام خوشه‌های گراف برای یافتن زیرهدف‌ها استفاده می‌کنند. در فصل بعدی، خواهیم دید که روش پیشنهادی، بدون پیدا کردن تمام خوشه‌ها، تنها به کشف زیرهدف‌های مفید می‌پردازد.

فصل چهارم
روش پیشنهادی

یادگیری تقویتی فرایند یادگیری به‌وسیله‌ی سیگنال پاداش و جریمه است. بسیاری از مسائل یادگیری را که با مدل مارکوف توصیف می‌شوند، می‌توان با روش‌های یادگیری تقویتی حل نمود. اما در مورد مسائل با دامنه‌ی پیچیده، شامل تعداد حالت‌ها و کنش‌های زیاد، روش‌های عادی یادگیری تقویتی بسیار کند عمل می‌کنند. به همین منظور، در بسیاری از تلاش‌های برای مقیاس‌پذیر نمودن فرایند یادگیری تقویتی، از انتزاع زمانی بهره برده شده است. در این دسته روش‌ها، سعی می‌شود در ابتدا به نوعی مسئله را به زیر مسائل کوچک‌تر تقسیم نمود. برای انجام این کار عمدتاً زیرهدف‌های مسئله شناسایی می‌شود، تا بتوان مهارت‌هایی را برای حل سریع‌تر مسائل ساخت. بیش‌تر روش‌هایی که به دنبال کشف زیر هدف‌ها هستند، ابتدا سابقه‌ی تعاملات با محیط را با استفاده از یک گراف مدل می‌کنند، سپس با تحلیل گراف حاصل، حالت‌های زیرهدف یافت می‌شوند. با پیدا شدن حالت‌های زیرهدف، سیاست‌های جزئی برای رسیدن به هر یک از این زیرهدف‌ها به‌دست می‌آیند. سپس با استفاده از چارچوب یادگیری گزینه، می‌توان از این سیاست‌های جزئی در راستای رسیدن بهینه به هدف بهره جست. در این فصل، روش جدیدی برای استفاده از تکنیک انتزاع زمانی ارائه خواهد شد. این روش که از این پس آن را روش مبتنی بر فرومون می‌نامیم، شامل سه مرحله‌ی کلی است:

نخست مدل‌سازی محیط به وسیله‌ی گراف. در این مرحله، یک گراف وزن‌دار جهت‌دار ساخته می‌شود. هر یال از i به j بیان‌گر وجود گذر از حالت i به j در سابقه‌ی تعاملات و وزن آن نشان‌دهنده‌ی تعداد این گذرها می‌باشد.

در مرحله‌ی دوم، زیرهدف‌ها برای ساخت مهارت آماده می‌شوند. عمده‌ی روش‌هایی که بر اساس گراف زیرهدف-ها را کشف می‌کنند، ابتدا تمام خوشه‌های گراف را پیدا کرده و سپس نقاط مرزی آن‌ها را به عنوان زیرهدف‌ها معرفی می‌کنند. در روش پیشنهاد شده در این پایان‌نامه، از الگوریتم کلونی مورچه استفاده می‌شود و با تحلیل توزیع فرومون یال‌ها، زیر هدف‌های مفید برای رسیدن به هدف نهایی به‌دست می‌آیند. تاکید بر این نکته ضروری است که زیرهدف‌های یافت شده در این روش، نقاط مرزی بین تمام خوشه‌های گراف نیستند، بلکه زیرهدف‌هایی هستند که در مسیر رسیدن به هدف قرار دارند.

در مرحله‌ی آخر، با استفاده از چارچوب گزینه مهارت‌ها ساخته شده و بکار برده می‌شوند، که در فصل دوم، به تفصیل درباره‌ی آن توضیح داده شد. برای به‌دست آوردن سیاست هرکدام از مهارت‌ها، نیز از روش بازبینی تجربه استفاده شده است. در ادامه‌ی فصل، هر کدام از مراحل الگوریتم، شامل مدل‌سازی به‌وسیله‌ی گراف، پیدا کردن نقاط زیرهدف و ساخت مهارت‌ها در روش مبتنی بر فرومون به ترتیب بسط داده می‌شوند.

۴-۱ مدل‌سازی به‌وسیله‌ی گراف

در این‌جا، فرض می‌شود عامل با محیط خود در تعامل است و تعدادی وظیفه‌ی دوره‌ای را به انجام رسانده است. اکنون این امکان وجود دارد که تجربه‌ی چندین دوره‌ی ابتدایی از تعاملات، به عنوان مبنایی برای ساخت گراف در نظر گرفته شود.

از آن‌جا که فرض شده است عامل یادگیر با یک محیط مارکوف متناهی در تعامل است، تعداد حالت‌ها و کنش‌ها متناهی بوده و می‌توان محیط را با گراف مدل کرد. به ازای هر حالت مارکوف، یک راس و به ازای هر کنش پایه‌ای که یک گذر در فضای حالت ایجاد می‌کند، یک یال جهت‌دار در نظر گرفته می‌شود. وزن هر یال نیز برابر تعداد گذرهای صورت گرفته بین دو حالت در دوره‌ی تعاملات مورد نظر می‌باشد. این گراف را گراف گذر می‌نامیم.

روش ارائه شده و برخی دیگر از روش‌های موجود، ممکن است بتوانند بدون مدل‌سازی محیط توسط گراف، زیرهدف‌ها را کشف نمایند. با توجه به آن‌چه گفته شد، در این‌جا این سوال مطرح می‌شود که چه دلیلی برای مدل‌سازی محیط به وسیله‌ی گراف وجود دارد؟ در پاسخ به این مسئله، می‌توان دو دلیل ذکر کرد:

نخست این‌که مدل‌سازی محیط توسط گراف، عامل را از تعامل بیش‌تر با محیط بی‌نیاز می‌کند. تقریباً در تمام موارد، تعامل با محیط بسیار پرهزینه و زمان‌بر است و این در حالی است که ممکن است عملکرد غیربهینه در محیط به مدت طولانی، به هیچ وجه برای عامل مطلوب نباشد. به همین دلیل مدل‌سازی با گراف می‌تواند بخش قابل توجهی از یادگیری را که مربوط به کشف حالت‌های زیرهدف است، بدون تعامل با محیط و متحمل شدن هزینه‌های آن و با سرعت بسیار بیش‌تری ممکن سازد.

علاوه بر این، مدل‌سازی محیط با گراف، امکان استفاده از الگوریتم‌های نظریه‌ی غنی گراف را می‌دهد، که سال‌ها مورد مطالعه و تحقیقات قرار گرفته و تا حد زیادی به بلوغ رسیده است. به دلیل تنوع الگوریتم‌های موجود در این زمینه، در طی سال‌های اخیر روش‌های متعددی در زمینه‌ی کشف زیرهدف‌ها بر مبنای تئوری گراف پیشنهاد شده‌اند[۳].

مسئله‌ی دیگری که بهتر است مورد بررسی قرار گیرد، تصمیم‌گیری درباره‌ی نوع گراف می‌باشد، به این معنی که گراف جهت‌دار و وزن‌دار انتخاب شود یا خیر. در بعضی از روش‌ها از جمله روش مرکزیت بردار ویژه، گراف مدل شده، به دلیل شرایط خاص الگوریتمی که از آن استفاده می‌کند، بدون جهت و بدون وزن در نظر گرفته می‌شود. اما همان‌طور که پیش‌تر نیز به آن اشاره شد، در روش پیشنهادی از گراف وزن‌دار و جهت‌دار استفاده شده است. دلیل این امر، آن است که حذف این دو مقوله، ممکن است منجر به از دست رفتن بخشی از دانش مربوط به محیط شود. این مسئله را به تفصیل در ادامه بررسی می‌کنیم:

در صورت حذف جهت‌یال‌ها، این فرض به صورت ضمنی به مسئله القا می‌شود که گذر حالت‌ها به یکدیگر متقابلاً در هر دو جهت انجام‌پذیر است. این فرض در برخی از محیط‌ها از جمله محیط اتاق‌های مشبک فرض نادرستی نیست، اما در بعضی از محیط‌های دیگر مانند محیط‌های آزمایشی تاکسی و محیط اتاق بازی و بسیاری از محیط‌های واقعی دیگر، فرضی نادرست است، که همان‌طور که در فصل آینده به آن اشاره خواهد شد، موجب کاهش کارآیی الگوریتم‌هایی می‌شود که از گراف بدون جهت استفاده می‌کنند.

از طرفی وزن یال‌ها که در این‌جا تعداد گذر انجام شده از روی آن‌ها در فضای حالت است، می‌تواند موجب کسب دانش قابل توجهی در راستای روشن‌تر شدن ساختار فضای حالت و همچنین حالت‌ها و کنش‌های موثر در آن شود. همان‌طور که در بخش‌های بعدی خواهیم دید، از وزن یال‌ها به عنوان یک دانش‌مکاشفه‌ای برای انتخاب یال‌های مسیرها، در بهینه‌سازی کلونی مورچه، استفاده خواهیم نمود.

آخرین نکته‌ای که در این‌جا باید ذکر گردد، ساختمان‌داده‌ایست که برای گراف در نظر خواهیم گرفت. با توجه به نیاز مکرر روش ارائه شده، به ساخت مسیر و در نتیجه، نیاز به بررسی راس‌های همسایه‌ی یک راس خاص، از لیست مجاورت[۸۷] به جای ماتریس مجاورت[۸۸] استفاده شده است. به این ترتیب، هزینه‌ی ساخت مسیری به طول حداکثر n به $O(m)$ (تعداد یال‌ها) کاهش پیدا می‌کند، در صورتی که اگر از ماتریس مجاورت استفاده شود، این هزینه به $O(n^2)$ می‌رسید. با توجه به این‌که در بیش‌تر محیط‌ها، تعداد یال‌های گراف متناظر، بسیار کم‌تر از تعداد یال‌های یک گراف کامل است، این تمهید موجب کاهش پیچیدگی زمانی الگوریتم خواهد شد.

[87] Adjacency List
[88] Adjacency Matrix

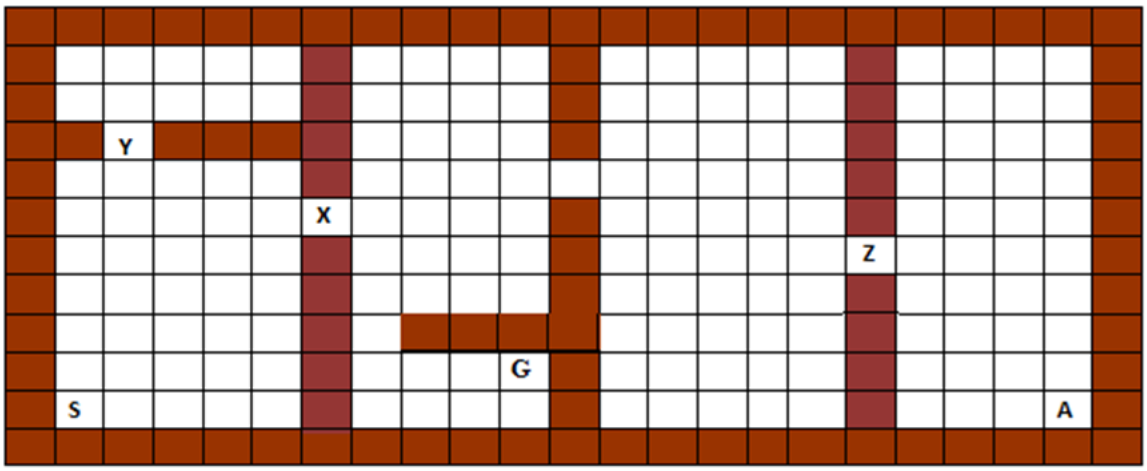
۴-۲ کشف زیرهدف‌ها

برخی از مسائل یادگیری تقویتی، دوره‌ای و برخی پیوسته هستند. پیش‌تر اشاره شد که تمرکز این پایان‌نامه، بر مسائل دوره‌ای می‌باشد که در آن هر دوره از یک حالت شروع شده و کنش‌ها مرتباً موجب تغییر حالت محیط می‌شوند. نهایتاً با ورود به حالت نهایی، دوره پایان یافته و پس از آن دوره‌ی بعدی با بازگشت محیط به حالت ابتدایی، مجدداً آغاز می‌گردد.

توجه به این نکته حائز اهمیت است که در بسیاری از مسائل یادگیری تقویتی، حالت آغازین دوره‌های مختلف یکسان می‌باشد. به عنوان مثال، حالت هر دوره‌ی عامل در محیط‌های رقابتی مانند شطرنج وتخته‌ی نرد، همواره از حالت مشخصی آغاز می‌شود. همین‌طور عاملی که سعی دارد یاد بگیرد یک آونگ معکوس را با وارد کردن نیرو به صورت عمودی نگاه دارد، نیز همواره وظیفه‌ی خود را از حالتی آغاز می‌کند که آونگ در حالت افتاده قرار دارد. نمونه‌های بسیار دیگری را می‌توان با شرایط مشابه مثال زد.

در فصل قبلی، تعدادی از معروف‌ترین روش‌های کشف زیر هدف مرور شد. مطابق آن‌چه در آن فصل دیدیم، مبنای بیش‌تر روش‌های مبتنی بر گراف برای کشف زیرهدف، خوشه بندی گراف و پیدا کردن نقاط مرزی آن‌ها می‌باشد. چنین رویه‌ای منجر به اکتشاف همه‌ی نقاط مرزی خوشه‌ها در کل فضای حالت خواهد بود. اما ممکن است بسیاری از این نقاط مرزی و مهارت‌های ساخته شده برای رسیدن به آن‌ها، کمکی به حل بهینه‌ی مسئله نکنند. برای روشن‌تر شدن این مطلب، فرض کنید در محیط پنج اتاقه‌ی شکل (۴-۱)، عامل قصد دارد از حالت شروع s، به حالت نهایی G نقل مکان کند. در چنین شرایطی، مهارتی که عامل را از حالت S به درب X برساند کمک شایانی برای یادگیری سریع‌تر خواهد نمود، اما مهارت متناظر با رساندن عامل از حالت A به درب Z، در این مسئله سودمند نیست، در صورتی‌که مهارت انتقال از حالت S به درب Y عامل را از هدف دور می‌کند. بنابراین در چنین مسائلی، بهتر است زیرهدف‌هایی شناسایی شوند که برای رسیدن به هدف مفید می‌باشند.

روش پیشنهادی با چنین رویکردی سعی در کشف و ساخت مهارت، برای زیرهدف‌های موثر در راه نیل به حالت نهایی دارد. برای انجام این کار، از دسته روش‌های بهینه‌سازی کلونی مورچه[۸۹] و به طور خاص الگوریتم سیستم مورچه[۹۰] استفاده شده است. بررسی‌های انجام شده، نشان داده است که تغییرات مربوط به میزان فرومون یال‌ها حین انجام بهینه‌سازی کلونی مورچه، اطلاعات مناسبی در راستای شناسایی زیرهدف‌های مربوط در اختیار می‌گذارد.



شکل (۴-۱): یک محیط ۵ اتاقه

۴-۲-۱ بهینه‌سازی کلونی مورچه

یکی از نخستین مطالعات رفتارشناسی انجام شده‌ی حشره‌شناسان، معطوف به توانایی مورچه‌ها در یافتن کوتاه-ترین مسیر از لانه به منبع غذا بوده است، که منجر به شکل‌گیری اولین مدل‌های الگوریتمی رفتار جستجوی غذای مورچه‌ها شده است. این مطالعات که روی چندین گونه از مورچه‌ها انجام شده، نشان‌دهنده‌ی ماهیت بی-نظم و تصادفی الگوی جستجوی مورچه‌ها می‌باشد[۳۰]. به محض این‌که محل غذا مکان‌یابی می‌شود، این جستجوها الگوهای منظم‌تری به خود می‌گیرند و مورچه‌های بیش‌تری از یک مسیر مشخص خود را به محل غذا

[89] Ant Colony Optimization
[90] Ant System

می‌رسانند، تا این‌که نهایتاً این روند به شکل اعجاب آوری به جایی می‌رسد که تمام مورچه‌ها از کوتاه‌ترین مسیر برای رسیدن به غذا استفاده می‌کنند.

این رفتار بهینه‌ی نهایی، حاصل نوعی از انتقال تجربیات و دانش از طرف مورچه‌های قبلی است که این مسیر را طی کرده‌اند. این انتقال دانش، در بیش‌تر انواع مورچه‌ها به طور غیرمستقیم و به صورت دقیق‌تر، از طریق واپس‌گذاری اثری از فرومون[۹۱] انجام می‌گیرد. مورچه‌های جستجوگر بر اساس میزان تمرکز فرومون در مسیرهای مختلف، تصمیم می‌گیرند که کدام مسیر را انتخاب کنند و به این شکل، مسیرهایی با تمرکز بیش‌تر فرومون، شانس بیشتری برای انتخاب شدن دارند.

آزمایش پل

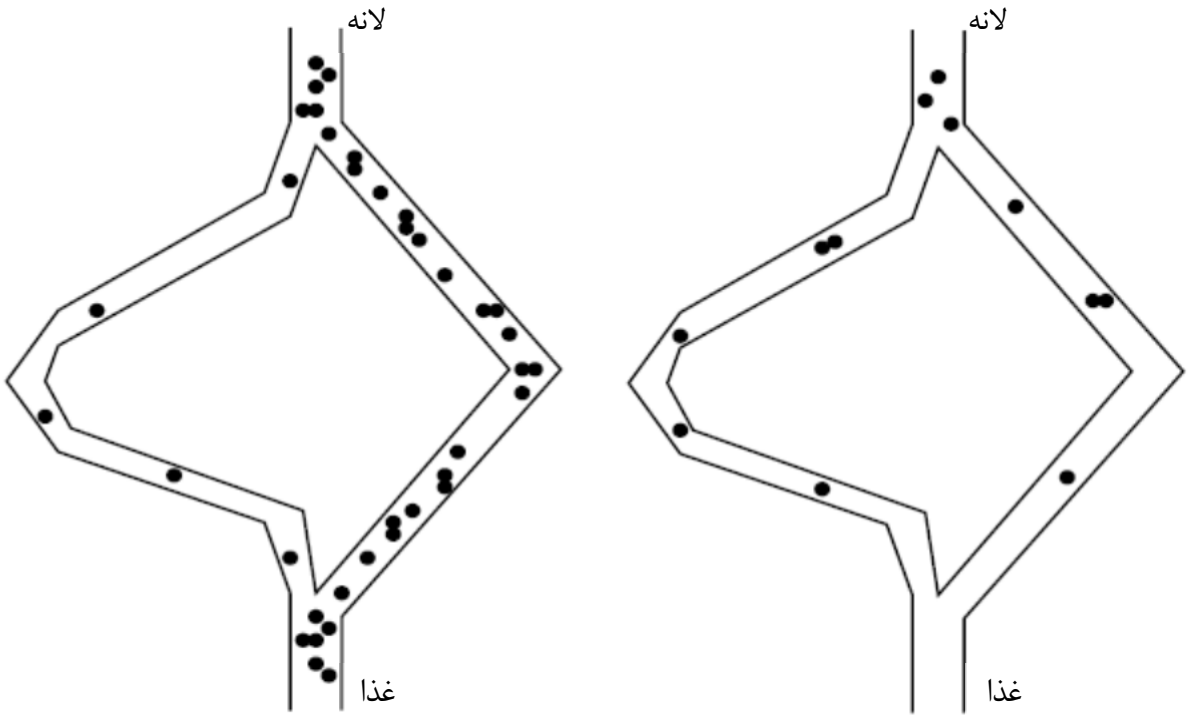
در آزمایشی که برای تشخیص ماهیت رفتار جستجوی مورچه‌ها انجام شد[۳۱]، محل غذا و لانه، از طریق دو پل به یکدیگر متصل شده بودند که طول یکی از آن‌ها بیش‌تر از دیگری بود. در شروع، هر دو مسیر به صورت تصادفی با احتمال تقریباً مساوی انتخاب می‌شدند و مطابق قسمت (آ) شکل (۴-۲)، در هر دو تعداد تقریباً برابری از مورچه‌ها تردد می‌کردند. با گذر زمان، مشابه شکل (۴-۲) قسمت (ب)، تعداد مورچه‌های بیش‌تری از مسیر کوتاه‌تر گذر کردند و انتخاب به سمت احتمال بیش‌تر، برای مسیر کوتاه‌تر، پیش رفت.

دلیل این اتفاق را می‌توان به این صورت توجیه کرد که مورچه‌ها در مسیر کوتاه‌تر سریع‌تر می‌توانند به لانه برگردند و به دلیل تبخیر طبیعی فرومون، میزان فرومون در مسیرهای بلندتر با آهنگ سریع‌تری به نسبت مسیرهای کوتاه‌تر کاهش می‌یابد.

در الگوریتم‌های برگرفته از این فرایند، از فرومون مجازی به عنوان تقلیدی از ویژگی‌های فرومون واقعی استفاده می‌شود، که نشان‌دهنده‌ی میزان «محبوبیت» یک راه حل برای یک مسئله‌ی بهینه‌سازی می‌باشد. می-

[91] Pheromone

توان گفت که فرومون مجازی، حاوی یک حافظه‌ی بلند مدت از کل فرایند جستجوی جواب می‌باشد. در این پایان‌نامه از همین خاصیت فرومون مجازی، برای به‌دست آوردن نقاط گلوگاه فضای حالت، استفاده خواهیم کرد.



(آ) توزیع انتخاب مسیر مورچه‌ها در شروع آزمایش (ب) توزیع انتخاب مسیر مورچه‌ها با گذشت زمان

شکل (۴-۲): آزمایش پل[۳۲]

۴-۲-۲ بهینه‌سازی کلونی مورچه ساده

اولین الگوریتمی که در این‌جا به آن می‌پردازیم، الگوریتم بهینه‌سازی کلونی مورچه‌ی ساده[۹۲] است که آن را به شکل اختصاری، SACO می‌نامیم. فرض کنید می‌خواهیم کوتاه‌ترین مسیر را از راس مبدا s، به راس مقصد t، روی گراف $G = (V, E)$ بیابیم. در این مسئله، برای نمایش میزان فرومون موجود در یال بین راس‌های i و j، از نماد τ_{ij} استفاده خواهیم کرد.

[92] Simple Ant Colony Optimization (SACO)

در ابتدای الگوریتم، فرض می‌کنیم که میزان فرومون بسیار کمی در هر یک از یال‌ها وجود دارد. در هر مرحله از اجرای الگوریتم، n_k مورچه به صورت متوالی در راس s قرار گرفته و هر یک بر اساس سیاستی مبتنی بر میزان فرومون یال‌های پیش رو، یک مسیر به سمت هدف می‌سازند.

نحوه‌ی ساختن مسیر به صورت افزایشی است و به این شکل است که اگر مورچه‌ی kام روی گره‌ی i قرار داشته باشد و N_i^k مجموعه گره‌های قابل دسترسی از گره‌ی i برای مورچه‌ی kام باشد، گره‌ی بعدی $j \in N_i^k$، با استفاده از رابطه‌ی احتمالی زیر انتخاب می‌شود[۳۲]:

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)}{\sum_{j \in N_i^k} \tau_{ij}^{\alpha}(t)} & if\ j \in N_i^k \\ 0 & if\ j \notin N_i^k \end{cases} \quad (۱-۴)$$

که در آن α یک عدد ثابت مثبت برای تعیین میزان تاثیر فرومون، در انتخاب یال بعدی است، که هر چه این مقدار بیش‌تر باشد، یال‌های با فرومون بیش‌تر با احتمال بیش‌تری انتخاب می‌شوند و این مسئله ممکن است باعث همگرایی سریع مسئله به یک جواب غیر بهینه گردد.

در این‌جا باید توجه شود که ممکن است بعضی از مسیرهای ساخته شده، شامل چندین حلقه باشد، که بعد از ساخت مسیرها، حلقه‌ها را باید از آن‌ها حذف کرد، تا تمام مسیرهای حاصل شده، ساده باشند.

بعد از آن‌که تمام n_k مورچه مسیر خود را به راس هدف ساختند، باید به شکلی میزان فرومون یال‌ها را به‌روز کرد که یال‌هایی که در مسیرهای بیش‌تری قرار داشته‌اند، فرومون بیش‌تری به نسبت سایر یال‌ها داشته باشند. به همین منظور، بعد از ساخته شدن تمام مسیرها، هر مورچه مقداری فرومون به یال‌های مسیر خود اضافه می‌کند[۳۲]:

$$\Delta\tau_{ij}^k(t) \propto \frac{1}{L^k(t)} \quad (۲-۴)$$

در این رابطه، $L^k(t)$ طول مسیری است که مورچه‌ی k ام در مرحله‌ی t می‌پیماید. به این ترتیب می‌توانیم تاثیر همه‌ی مورچه‌ها مطابق با رابطه‌ی زیر اعمال کنیم[۳۲]:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{n_k} \Delta\tau_{ij}^k(t) \quad (۳-۴)$$

آزمایش‌های متعددی که روی این الگوریتم انجام شد، نشان داد که این الگوریتم به سرعت به یک راه حل بهینه‌ی محلی همگرا می‌شود. برای جلوگیری از این رویداد و بالاتر بردن میزان کاوش[۹۳] به نسبت انتفاع[۹۴]، تبخیر فرومون‌ها هم در الگوریتم در نظر گرفته شد[۳۲].

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) \quad (۴-۴)$$

در این رابطه، ρ نرخی است که الگوریتم متناسب با آن، تجربیات قبلی را فراموش می‌کند. به عبارت دیگر این ضریب میزان تاثیر تاریخچه‌ی جستجو را تعیین می‌کند. هر چه این میزان بیش‌تر باشد، فرایند جستجو تصادفی‌تر می‌شود. در حالت حدی، اگر ρ برابر یک باشد جستجو کاملاً تصادفی و اگر برابر صفر باشد، کاملاً قطعی است.

۴-۲-۳ سیستم مورچه

نتایج آزمایش‌های انجام شده روی این الگوریتم، نشان داد که SACO برای گراف‌های ساده مسیرهای بهینه را به خوبی پیدا می‌کند، اما با پیچیده‌تر و بزرگ‌تر شدن گراف هزینه‌ی ساخت مسیرها به شدت افزایش پیدا می‌کند و الگوریتم ناپایدار و بسیار حساس به انتخاب پارامترها می‌گردد[۳۲].

به همین دلیل، تغییرات اندکی در این روش ایجاد شد تا در روش جدید، تا حدی مشکلات فوق برطرف شود. حاصل این تغییرات، الگوریتم سیستم مورچه (AS) نام گرفته است. دو تغییر عمده‌ی شکل گرفته روی الگوریتم

[93] Exploration
[94] Exploitation

SACO، به طور خلاصه، تعویض احتمال گذر به شکلی که دانش مکاشفه‌ای را شامل شود و افزودن لیست ممنوعه[۹۵] می‌باشد در الگوریتم AS احتمال گذر از گره‌ی i به گره‌ی j با عبارت زیر ارزیابی می‌شود[۳۲]:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{j\in N_i^k(t)} \tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)} & if\ j \in N_i^k(t) \\ 0 & if\ j \notin N_i^k(t) \end{cases} \quad (۵-۴)$$

در این رابطه، η_{ij} میزان تناسب پیشین حرکت از راس i به j می‌باشد، که توسط یک تابع مکاشفه‌ای تعیین می‌شود. این مقدار در واقع، میزان جذابیت یک حرکت را با استفاده از دانش پیشین[۹۶] در مقابل میزان فرومون که دانش پسین[۹۷] مسئله است، بیان می‌کند. دو پارامتر α و β که مقادیری در بازه‌ی $[0,1)$ می‌باشند، تعادل بین کاوش و انتفاع را برقرار می‌کنند. اگر $\alpha = 0$ باشد، از دانش فرومون استفاده نمی‌شود و جستجو تبدیل به یک جستجوی حریصانه می‌شود، که تنها از دانش پیشین استفاده می‌کند. اگر $\beta = 0$ باشد، الگوریتم کاملاً مشابه الگوریتمSACO خواهد بود.

برای کاستن از تعداد پارامترها، یک رابطه‌ی دیگر پیشنهاد شده است[۳۳]، که وابستگی را به یک پارامتر کاهش می‌دهد و به این ترتیب، کار تنظیم پارامتر ساده‌تر می‌شود. این رابطه همچنین بار محاسباتی را می‌کاهد:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}(t)}{\sum_{j\in N_i^k(t)} \alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}(t)} & if\ j \in N_i^k(t) \\ 0 & if\ j \notin N_i^k(t) \end{cases} \quad (۶-۴)$$

در این رابطه، α میزان اهمیت نسبی مقدار فرومون را نشان می‌دهد. مقادیر بیش‌تر α باعث در نظر گرفتن بیش‌تر دانش پسین خواهد بود.

[95] Tabu List
[96] Prior Knowledge
[97] Posterior Knowledge

دانش مکاشفه‌ای که از طریق تابع η به مسئله تزریق می‌شود، موجب جهت‌گیری ضمنی به سمت جواب‌های بهتر است و به همین دلیل، وابسته به مسئله خواهد بود. برای مسئله‌ی کوتاه‌ترین مسیر، می‌توانیم از معکوس اندازه‌ی مسیر برای آن استفاده کنیم:

$$\eta_{ij} = \frac{1}{d_{ij}} \qquad (۷-۴)$$

در این رابطه، d_{ij}، فاصله (یا هزینه) مسیر گذر از گره‌ی i به گره‌ی j می‌باشد. در این روش برای تولید سریع‌تر مسیرها و جلوگیری از تولید مسیرهایی با راس‌های تکراری، از یک لیست ممنوعه استفاده می‌شود. هر بار که مورچه‌ای از یک راس می‌گذرد، آن راس در لیست ممنوعه قرار می‌گیرد، تا بار دیگری تنواند وارد آن راس شود. الگوریتم سیستم مورچه در الگوریتم ۸ خلاصه شده است.

برای شرط پایان الگوریتم می‌توان چند حالت را در نظر گرفت: یکی این‌که تعداد مراحل، از حد خاصی (n_t) فراتر رود. یا این‌که به راه حلی دست پیدا کنیم که هزینه‌ی آن از حد قابل قبول کم‌تر باشد. گزینه‌ی دیگر زمانی است که همه‌ی مورچه‌ها یک مسیر را طی کنند. برای سادگی بیش‌تر معمولاً از شرط اول، به عنوان شرط پایان استفاده می‌شود.

پیچیدگی زمانی اجرای این الگوریتم، بستگی مستقیم به تعداد مراحل اجرا دارد. با فرض این‌که الگوریتم n_t مرحله‌ی اجرایی داشته باشد، پیچیدگی زمانی آن $O(n_t(m\, n_k + n))$ خواهد بود که در آن n تعداد کل راس-ها و m تعداد یال‌ها می‌باشد. البته در بسیاری از کاربردها از جمله کاربرد آن در روش پیشنهادی این پایان‌نامه، تعداد مورچه‌ها و تعداد مراحل، اعداد ثابتی خواهند بود و می‌توان میزان رشد زمان الگوریتم را با $O(n+m)$ نمایش داد.

الگوریتم ۸: الگوریتم سیستم مورچه

ورودی: $(n_k, n_t, \alpha, \ \rho)$

$t \leftarrow 0$

تکرار کن

برای مورچه‌های با شماره‌ی k از ۱ تا n_k

مسیر $x^k(t)$ را با استفاده از رابطه‌ی (۴-۶) بساز.

برای همه یال‌های (i,j)

تبخیر را مطابق رابطه‌ی (۴-۴) انجام بده.

میزان فرومون یال را مطابق رابطه‌ی (۴-۳) تغییر بده.

$t \leftarrow t + 1$

تا زمانی که شرط پایان فرا رسیده باشد. $(t \geq n_t)$

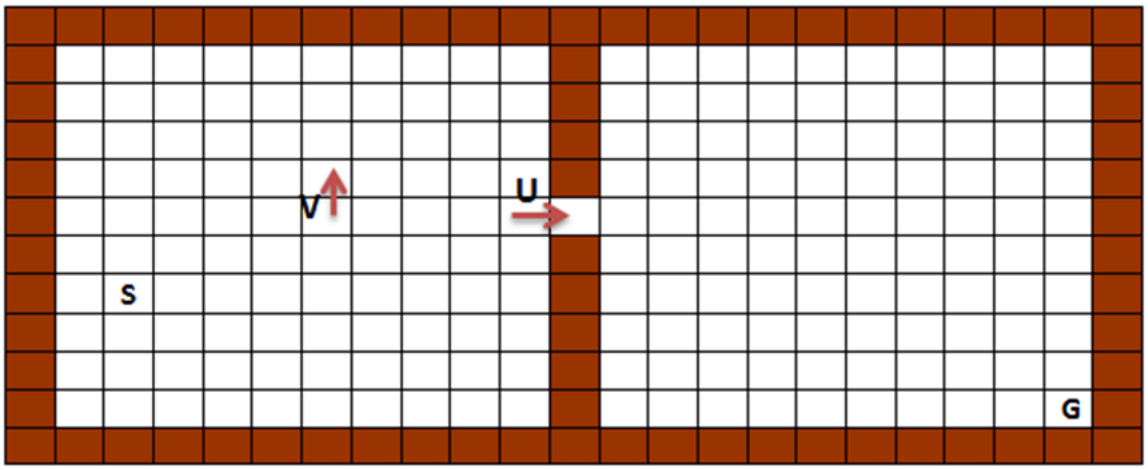
کوتاه‌ترین مسیر دیده شده را به عنوان جواب در نظر بگیر.

۴-۲-۴ الگوریتم کشف زیرهدف

برای طراحی روشی برای کشف زیرهدف‌ها، ابتدا باید مفهوم زیرهدف را تعریف کرد و سپس الگوریتمی طراحی نمود که حالت‌هایی با این ویژگی را پیدا کند. تعریف ما از زیرهدف، نقاط گلوگاهی است که بین نواحی با همبندی بالا قرار گرفته‌اند. طبیعتاً این نقاط در مسیرهایی که به سمت هدف ساخته می‌شود، نقش محوری و با ثباتی دارند، به این معنی که به نسبت حالت‌های دیگر، حضور منظم‌تری در مسیرهای مختلف گذر از حالت شروع به حالت هدف، دارا می‌باشند.

پیش‌تر بیان شد که ممکن است بسیاری از مهارت‌های ساخته شده برای انتقال از حالت‌های میانی هر کدام از خوشه‌ها، به حالت‌های مرزی آن خوشه، مفید نبوده و یا بعضاً عامل را از هدف دور کنند. در ادامه، الگوریتمی ارائه خواهد شد که با بررسی رفتار میزان فرومون یال‌ها در طول اجرای الگوریتم بهینه‌سازی مورچه، اهداف مفید که عامل را در راستای رسیدن به هدف پیش‌خواهند برد، پیدا کند.

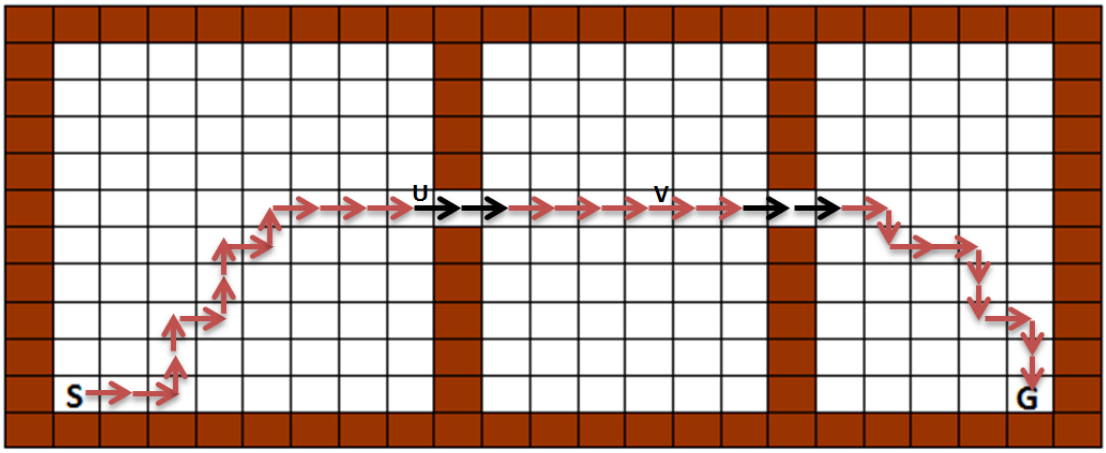
برای روشن‌تر شدن روش مبتنی بر فرومون، به شکل (۴-۳) توجه کنید. گراف گذر معادل این محیط را تصور کنید که در آن هر راس به حداکثر چهار راس همسایه‌ی خود متصل است. دو یال از این گراف، با نام‌های u و v، در شکل مشخص شده‌اند. فرض کنید الگوریتم سیستم مورچه، با حالت شروع s به عنوان راس مبدا و حالت پایانی G به عنوان راس مقصد، روی این گراف اجرا شود.



شکل (۴-۳): محیط دو اتاقه

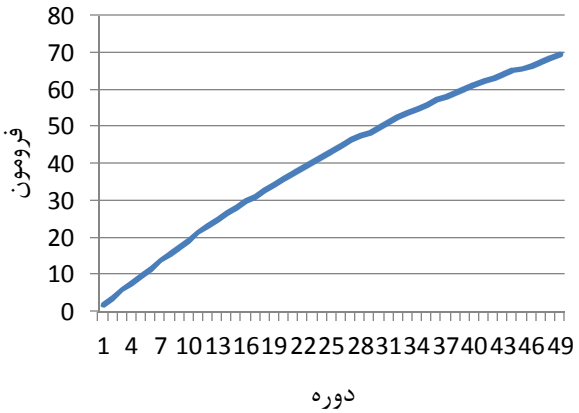
در دوره‌های اولیه، با توجه به مسیرهای غیر بهینه‌ای که تولید می‌شوند، تقریباً برای تمام یال‌ها (از جمله u و v) شانس قرار گیری در مسیرهای ساخته شده توسط مورچه‌ها وجود دارد. در ادامه، یال‌هایی مانند u که در مجاورت یک راس مرزی خوشه قرار دارند، به شکل منظم‌تری در مسیر قرار می‌گیرند و به میزان فرومون آن‌ها مرتباً افزوده می‌شود. اما یال‌هایی مانند v که در مرکز خوشه قرار دارند و برای آن‌ها جایگزین‌های زیادی برای ساخت مسیر وجود دارد، در بعضی از مسیرها موجود و در برخی دیگر غایب خواهند بود، به عبارت دیگر حضور کم‌نظم‌تری در مسیرهای منتهی به هدف دارند و میزان فرومون آن‌ها حین پیش‌روی الگوریتم گاهی کاسته شده و گاهی افزوده می‌شود. به این ترتیب می‌توان نتیجه گرفت که *یال‌هایی که تغییرات با بی‌نظمی کم‌تری در مقدار فرومون آن‌ها رخ داده، نقش اساسی‌تری در رساندن عامل از حالت شروع به حالت پایانی دارند و احتمالاً در نقاط مرزی خوشه‌ها قرار دارند.*

تمامی آن‌چه گفته شد مربوط به زمان اندکی بعد از شروع الگوریتم می‌باشد. با گذر زمان کافی، مسیرهای ساخته شده به کوتاه‌ترین مسیر همگرا شده و از جایی به بعد، تنها درصد کمی از یال‌ها که روی کوتاه‌ترین مسیر هستند، انتخاب می‌شوند. یال‌های دیگر به سرعت میزان فرومون خود را از دست داده و این مقدار به صفر همگرا می‌شود. با همگرایی میزان فرومون باقی یال‌ها به صفر، مقادیر این کمیت در توالی زمان، شکل منظم‌تری به خود گرفته و ممکن است به عنوان کاندید یال‌های منتهی به نقاط زیرهدف در نظر گرفته شوند. برای جلوگیری از این موضوع می‌توانیم تنها، یال‌های واقع بر کوتاه‌ترین مسیر را درنظر بگیریم. این راه‌کار از دو بابت می‌تواند سودمند باشد: نخست آن‌که یال‌هایی که مقدار فرومون آن‌ها به صفر میل کرده، از گزینه‌های موجود کنار گذاشته می‌شوند. دوم این‌که زیرهدف‌های یافت شده با این رویه، علاوه بر این‌که عامل را به سمت هدف رهنمون می‌کنند، روی کوتاه‌ترین مسیر تا هدف واقع هستند و به این شکل، ترتیبی از مهارت‌ها به‌دست می‌آید که با کم‌ترین تعداد گام ممکن، عامل را به هدف می‌رسانند، که خود موجب حذف مهارت‌های غیرسودمند خواهد شد. با توجه به آن‌چه گفته شد، برای یافتن یال‌های مجاور زیرهدف، کافیست بعد از گذر اندکی زمان از اجرای الگوریتم بهینه‌سازی مورچه، مطابق شکل (۴-۴) یال‌های کوتاه‌ترین مسیر را بررسی کرد و یال‌هایی با کم‌ترین میزان تغییرات فرومون را به عنوان یال‌های مجاور با زیرهدف برگزید. در شکل (۴-۴)، یال‌هایی که تیره‌تر رسم شده‌اند، دارای چنین شرایطی می‌باشند.

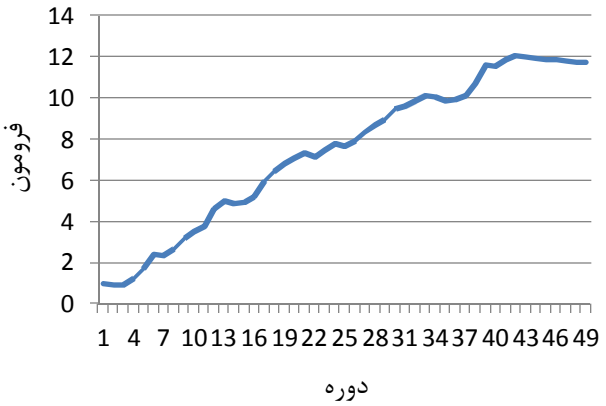


شکل (۴-۴): پیدا کردن حالت‌های زیرهدف در الگوریتم پیشنهادی

برای تایید آن‌چه گفته شد، در ادامه نموداری از تغییرات میزان فرومون دو یال مشخص شده در شکل (۴-۴)، نمایش داده شده است. نمودارهای شکل (۴-۵) (آ) و (ب) به ترتیب نمودار تغییرات میزان فرومون برای دو یال u و v می‌باشند. این دو یال نماینده‌ی دو نوع مختلف از یال‌های گراف می‌باشند، u از جمله یال‌های مجاور با زیرهدف و v یک یال در نواحی میانی خوشه‌ی خود می‌باشد. از بررسی این دو نمودار، تایید می‌گردد که میزان تغییرات فرومون در یال v به نسبت یال u بیش‌تر است.



(آ) تغییرات میزان فرومون برای یال u مشخص شده در شکل (۴-۴)



(ب) تغییرات میزان فرومون برای یال v مشخص شده در شکل (۴-۴)

شکل (۴-۵): تغییرات فرومون برای دو یال متفاوت

در این‌جا یک مسئله‌ی بسیار مهم، تعیین معیاری برای میزان تغییرات می‌باشد. یکی از معروف‌ترین معیارهای موجود برای بررسی میزان تغییرات یک متغیر، واریانس است که برای n نمونه داده‌ی $x_1, x_2, \dots, x_n$ به صورت زیر محاسبه می‌شود:

$$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n} \qquad (۸-۴)$$

استفاده از واریانس را به عنوان معیاری برای تمایز یال‌های مجاور با زیرهدف و باقی یال‌ها، از دو جهت می‌توان نقد کرد: نخست این‌که اگر به دو نمودار شکل (۴-۵) دقت شود، این مسئله روشن خواهد شد که مقیاس دو نمودار یکسان نیست، به این معنی که نمودار فرومون یال u که تغییرات نرم‌تری دارد، از نظر مقدار عددی در

دوره‌های متناظر بسیار بزرگ‌تر از مقدار فرومون یال v می‌باشد. به همین دلیل واریانس مقادیر فرومون یال u برخلاف آن‌چه برای ما مطلوب است، بیش‌تر خواهد بود.

به راحتی می‌توان نشان داد که اگر برای دو متغیر تصادفی X و Y داشته باشیم: $Y = aX$، در این صورت در مورد واریانس این دو، رابطه‌ی $Var(Y) = a^2 Var(X)$ برقرار خواهد بود. این رابطه به این معنی است که برای یکسان‌سازی مقیاس واریانس متغیرهای تصادفی، می‌توانیم واریانس هر متغیر را بر توان دوم میانگین آن تقسیم کنیم:

$$CV_X^2 = \frac{\sigma_X^2}{\mu_X^2} \qquad (۹-۴)$$

در ریاضیات به جذر این نسبت، ضریب تغییرات[۹۸] و به طور خلاصه CV گفته می‌شود. ضریب تغییرات در شرایطی که متغیر تصادفی بتواند شامل اعداد مثبت و منفی باشد، ممکن است به مشکل برخورد کند، چراکه این احتمال وجود دارد که میانگین متغیر تصادفی برابر صفر شده و به این صورت، ضریب تغییرات تعریف نشده باشد. در شرایط فعلی، می‌دانیم که میانگین میزان فرومون هیچ یالی صفر نخواهد بود و از این بابت استفاده از ضریب تغییرات مشکلی ندارد.

مسئله‌ی دوم این است که واریانس یا حتی مجذور ضریب تغییرات فرومون نمی‌تواند معیار کاملاً مناسبی برای جداسازی این دو دسته از یال‌ها باشد، چرا که این معیار به توالی مقادیر فرومون در طول زمان توجهی نمی‌کند و ممکن است یک ترتیب‌دهی متفاوت از مقادیر فرومون نمودار شکل (۴-۵) (ب)، تبدیل به نمودار همواری مشابه نمودار شکل (۴-۵) (آ) گردد. بنابراین لازم است توالی زمانی مقادیر فرومون در این معیار لحاظ گردد.

[98] Coefficient of Variation

برای حل مسئله‌ی دوم، می‌توان از واریانس شیب نمودار مقادیر فرومون در طول زمان استفاده کرد. با فرض این‌که F_i مقدار فرومون یال در لحظه‌ی iام باشد، مقدار شیب نمودار در لحظه‌ی iام، از رابطه‌ی $M_i = F_i - F_{i-1}$ حاصل می‌شود. با این تعریف σ_M^2 یک معیار مناسب‌تر برای این مسئله می‌باشد.

معیار فوق، همچنان مقیاس مقادیر فرومون‌ها را در نظر نمی‌گیرد. به همین دلیل باید مشابه آن‌چه قبلاً انجام دادیم، این مشکل را حل کنیم. به این دلیل که ممکن است میانگین شیب نمودار صفر گردد، نمی‌توانیم معیار ضریب تغییرات شیب نمودار (CV_M) را در نظر بگیریم. بنابراین معیار زیر برای آن پیشنهاد می‌شود:

$$R_F = \frac{\sigma_M^2}{(\max_i F_i - \min_i F_i)^2} \qquad (۱۰-۴)$$

در این رابطه، R_F را میزان ناهمواری مقادیر فرومون F نامیده‌ایم. در معیار ناهمواری، هم توالی زمانی مقادیر فرومون و همچنین مقیاس متغیر تصادفی F دیده شده است. بدیهی است که در صورتی‌که متغیر تصادفی F، k برابر شود، میزان ناهمواری آن تغییری نمی‌کند.

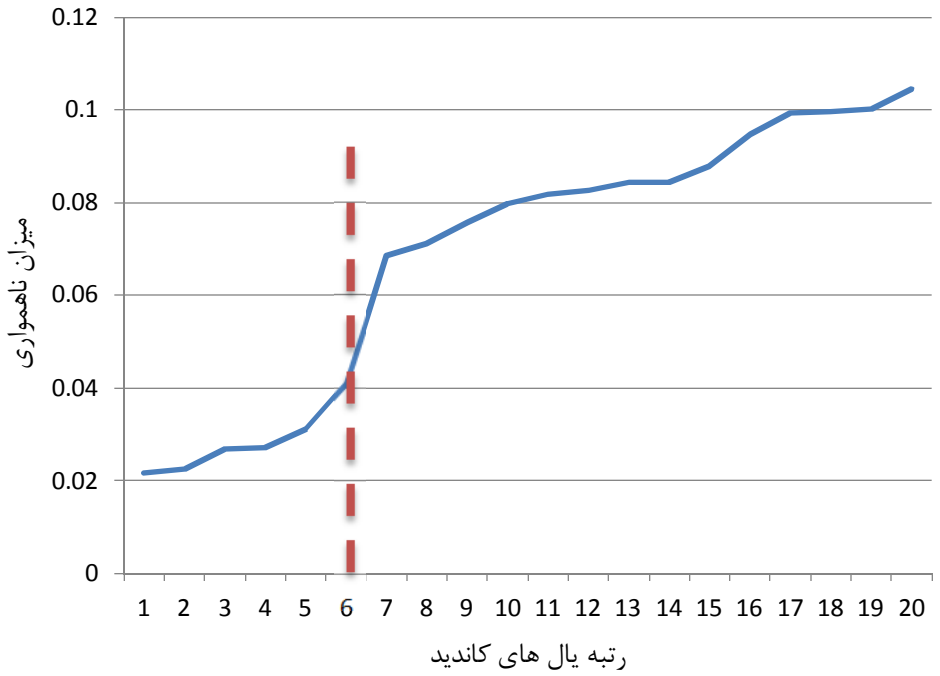
تا این‌جا گفته شد که با استفاده از معیار ناهمواری فرومون، به هر یک از یال‌هایی که در کوتاه‌ترین مسیر قرار دارند، عددی نسبت می‌دهیم که بیان‌گر میزان تناسب این یال‌ها برای مجاورت با زیرهدف‌های مسئله می‌باشد. اگر یال‌های کوتاه‌ترین مسیر را بر این اساس به صورت صعودی مرتب کنیم، انتظار داریم یال‌های مجاور با زیرهدف، در ابتدای لیست، کنار هم قرار بگیرند. مسئله‌ی جدیدی که در این‌جا با آن مواجه هستیم، نحوه‌ی جداسازی یال‌های مجاورِ با زیرهدف، از بقیه‌ی یال‌ها است.

یک راه برای حل این مسئله، این است که تعداد زیرهدف‌ها مثلاً k، به عنوان پارامتر ورودی الگوریتم به مسئله داده شده و k کاندید اول از ابتدای آرایه برداشته شده و نقاط زیر هدف به این شکل شناسایی شود. این

راه‌حل از این بابت که خروجی مسئله کاملاً تحت تاثیر مقدار ورودی آن خواهد بود، چندان مطلوب نمی‌باشد. همچنین در بسیاری از محیط‌ها تعداد زیرهدف‌های مسئله، به سادگی برای کاربر قابل تشخیص نیست.

در راه‌حل پیشنهادی، یک الگوریتم برای تشخیص خودکار یال‌های مربوطه ارائه شده است. ایده‌ی اصلی این روش، توجه به مقادیر و آهنگ تغییرات ناهمواری، در یال‌های کوتاه‌ترین مسیر است.

برای جداسازی یال‌های هدف، دو معیار در نظر گرفته می‌شود: معیار نخست شیب نمودار است. اولین زیرنویسی که در آن شیبِ ناهمواری یال‌ها به شکل جدی افزایش پیدا کند، می‌تواند کاندید مناسبی برای محل جداسازی یال‌های هدف و دیگر یال‌ها باشد، چرا که انتظار می‌رود یال‌هایی که با زیرهدف‌های گراف مجاور هستند، میزان ناهمواری نزدیک به هم داشته باشند. به عبارت دیگر، اولین نقطه‌ای که کیفیت یال‌ها به شکل محسوسی افت می‌کند، به نوعی نشان‌دهنده‌ی مرز بین یال‌های مجاور با زیرهدف و بقیه‌ی یال‌های مسیر خواهد بود. به دلیلی مشابه آن‌چه پیش‌تر ذکر شد، نباید یال‌هایی که به نسبت بهترین یال، بیش از حد مشخصی بی-کیفیت هستند را برگزید؛ چراکه ممکن است در مسئله‌ای، شیب تغییرات چندان فزاینده نباشد، اما به تدریج از کیفیت یال‌ها کاسته شود و این روند تدریجی در جایی باید منقطع گردد. شکل (۴-۶) نمایی از این تحلیل ارائه می‌کند.



شکل (۴-۶): نمودار میزان ناهمواری یال‌های کاندید، برحسب رتبه‌ی آن‌ها در محیط اتاق بازی

در این‌جا به دو مقدار آستانه‌ای τ_v و τ_d نیاز است تا حد مرزی مقدار و شیب نمودار تعیین گردد. به صورت دقیق‌تر b مرز مورد نظر است اگر و تنها اگر داشته باشیم:

$$Fail(b) = true\ and\ \forall i < b: Fail(i) = false \qquad (۴-۱۱)$$

در رابطه‌ی بالا، $Fail(i)$ یک تابع بولی است که مشخص‌کننده‌ی قبول نشدن iامین کاندید به عنوان یال مجاورِ زیرهدف می‌باشد. طبق این رابطه، b به عنوان مرز شناخته می‌شود، اگر خود پذیرفته نشود و تمام یال‌های قبلی پذیرفته شده باشند. تابع $Fail$، خود به صورت زیر تعریف می‌شود.

$$Fail(i) = (d_i > \tau_d . d_{init}\ or\ v_i > \tau_v . v_0) \qquad (۴-۱۲)$$

در این رابطه، v_i مقدار ناهمواری یال با رتبه‌ی i، d_i برابر $v_i - v_{i-1}$ و d_{init} نخستین d_i غیر صفر می‌باشد. به صورت خیلی خلاصه، این رابطه می‌گوید یال iام پذیرفته نمی‌شود، اگر و تنها اگر ناهمواری آن بیش از نسبتی از ناهمواری بهترین یال باشد، یا شیب نمودار در آن نقطه بیش از ضریبی از اولین شیب غیرصفر نمودار گردد.

از آن‌جایی‌که کارایی الگوریتم بستگی مستقیم به زیرهدف‌های یافت شده دارد و انتخاب بیش‌تر یا کم‌تر از تعداد واقعی زیرهدف‌ها، ممکن است به کلی روند حل بهینه‌ی مسئله را مختل کند، انتخاب درست مقادیر آستانه‌ای فوق در بهینگی راه حل ارائه شده توسط روش پیشنهادی تا حد زیادی موثر است و باید در انتخاب آن‌ها دقت شود. در فصل ششم، حساسیت به این پارامترها سنجیده خواهد شد.

برای انتخاب یال‌های مجاور زیرهدف، غیر از بررسی مقدار و شیب نمودار، مسئله‌ی دیگری نیز باید مورد بررسی قرار گیرد: در بسیاری از محیط‌ها، از جمله محیط اتاق‌ها، معمولاً به ازای هر حالت زیر هدف، دو یال مجاور در مسیر بهینه وجود دارد. این دو یال معادل کنش‌هایی هستند که عامل را به حالت زیرهدف برده و سپس از آن خارج می‌کنند. u و یال بلافصل بعدی آن در شکل (۴-۴)، نمونه‌ای از این زوج یال‌ها می‌باشند. در صورتی که یک دسته یال متوالی از فیلتر قبلی گذر کردند، در مرحله‌ی بعدی باید، یال‌های متوالی شناسایی شده و تنها یکی از آن‌ها انتخاب شود که ناهمواری کم‌تری دارد.

بعد از این‌که یال‌های کوتاه‌ترین مسیر، طی دو مرحله فیلتر شدند، یال‌های باقی‌مانده، مجاور زیر هدف‌هایی خواهند بود که در کوتاه‌ترین مسیر قرار دارند. در انتها می‌توان، حالت‌های متناظر با راس‌های یال‌های منتخب را به عنوان زیرهدف‌ها به‌دست آورد.

الگوریتم ۹ یک جمع‌بندی از تمام آن‌چه گفته شد، ارائه می‌دهد. به صورت خیلی خلاصه می‌توان گفت در ابتدا، الگوریتم سیستم مورچه فراخوانی شده و آرایه‌ای حاوی یال‌های کوتاه‌ترین مسیر به‌دست می‌آید. آرایه بر اساس مقدار ناهمواری یال‌ها به صورت صعودی مرتب می‌شود و در ادامه، نقطه‌ی مرزی b به‌دست می‌آید. نقطه‌ی b طوری انتخاب شود که شیب نمودار ناهمواری یال‌های مرتب شده، در آن نقطه بیش از حد مجاز

باشد، یا مقدار تغییرات آن از حد آستانه فراتر رفته باشد. از بین یال‌های باقی‌مانده، در صورتی که بین آن‌ها مجموعه‌ای از یال‌های مجاور وجود داشته باشد، کل مجموعه غیر از بهترین یال (یال با کمترین ناهمواری) حذف می‌شود. در پایان، راس‌های یال‌های منتخب نهایی به عنوان مجموعه‌ی زیرهدف‌ها در نظر گرفته می‌شود.

الگوریتم ۹: روش پیشنهادی برای کشف حالت‌های زیرهدف

ورودی: $(n_k, t_d, \alpha,\ \rho, \tau_d,\ \tau_v)$

الگوریتم سیستم مورچه$(t_d, \alpha,\ \rho)$ را اجرا کن و یال‌های کوتاهترین مسیر را در آرایه‌ی SP بریز.

آرایه‌ی SP را با توجه به مقدار فیلد R_F به صورت صعودی مرتب کن.

$v_0 \leftarrow SP[0].R_F$

به ازای i از ۱ تا طول آرایه‌ی SP

$d_{init} \leftarrow SP[i].R_F - SP[i-1].R_F$

اگر $d_{init} \neq 0$ از حلقه خارج شو.

به ازای b از ۱ تا طول آرایه‌ی SP

اگر $SP[b].R_F - SP[b-1].R_F > \tau_d.d_{init}$ یا $SP[b].R_F > v_0.\tau_v$

از حلقه خارج شو.

به ازای هر مجموعه یال‌های مجاور در $SP[0..b-1]$ مانند $Adjacent$

$best \leftarrow argmin_i\{Adjacent.Edges[i].R_F\}$

$SubGoals.add(Adjacent.Edges[best].head)$

محاسبه‌ی افزایشی[۹۹] واریانس

آخرین مطلبی که در این بخش به آن اشاره خواهیم کرد، در مورد نحوه‌ی محاسبه‌ی واریانس می‌باشد. در حالت عادی، برای محاسبه‌ی واریانس هر یک از یال‌ها با استفاده از رابطه‌ی (۴-۸)، نیاز به آرایه‌ای برای نگه‌داری مقادیر فرومون می‌باشد. در این صورت پیچیدگی حافظه‌ی مورد نیاز الگوریتم $\theta(mn_t)$ خواهد بود که در آن m تعداد یال‌ها و n_t تعداد دوره‌های الگوریتم است. با توجه به بزرگی اندازه‌ی فضای حالت در بیش‌تر مسائل

[99] Incremental

پیش‌رو، باید از یک روش افزایشی استفاده شود، که از حافظه‌ی ثابت برای هر یال استفاده کند و به عبارت دیگر، نیازی به نگه‌داری تمامی مقادیر فرومون یک یال نباشد. از روابط زیر برای محاسبه‌ی افزایشی واریانس استفاده شده است:

$$\sigma_n^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{X}_n)^2}{n} = \frac{y_n}{n} \quad (۱۲-۴)$$

$$y_n = \sum_{i=1}^{n}(x_i - \bar{X}_n)^2 = \sum_{i=1}^{n} {x_i}^2 + n{\bar{X}_n}^2 - 2\bar{X}_n \sum_{i=1}^{n} x_i \quad (۱۳-۴)$$

با فرض این‌که $s_n = \sum_{i=1}^{n} x_i$ باشد، خواهیم داشت:

$$y_n = \sum_{i=1}^{n} {x_i}^2 + n(\frac{s_n}{n})^2 - 2\frac{s_n}{n}s_n = \sum_{i=1}^{n} {x_i}^2 - \frac{{s_n}^2}{n} \quad (۱۴-۴)$$

به شکل مشابه می‌توان y_{n+1} را محاسبه کرد:

$$y_{n+1} = \sum_{i=1}^{n+1} {x_i}^2 - \frac{s_{n+1}^2}{n+1} \quad (۱۵-۴)$$

با تفریق دو عبارت فوق خواهیم داشت:

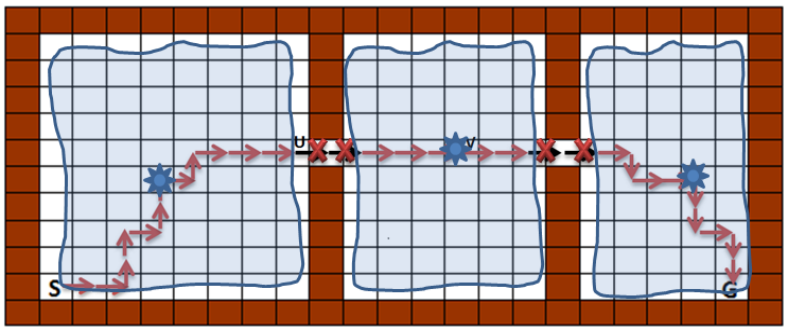
$$y_{n+1} = y_n + x_{n+1}^2 - \left(\frac{s_{n+1}^2}{n+1} - \frac{{s_n}^2}{n}\right) \quad (۱۶-۴)$$

عبارت (۴-۱۶) به این معناست که با داشتن جمله‌ی جدید (x_{n+1})، مجموع جملات قبلی (s_n) و y_n می‌توان y_{n+1} و در نتیجه σ_{n+1}^2 را محاسبه نمود.

۴-۳ ساخت مهارت

بعد از این‌که زیرهدف‌هایی که در مسیر رسیدن به هدف قرار دارند، توسط روش مبتنی بر فرومون به‌دست آمدند، باید انجمن‌هایی را که این زیرهدف‌ها مرز آن‌ها هستند، به‌دست بیاوریم، تا سیاست‌های جزئی بهینه در دامنه‌ی هر انجمن، برای رسیدن به نقاط زیرهدف حاصل شود. روش ارائه شده برای به‌دست آوردن انجمن‌ها بر این فرض استوار است که هر انجمن، یک ناحیه با همبندی بالاست که ارتباط آن با سایر نواحی محدود می‌باشد.

با توجه به این فرض، ایده‌ی اصلی روش ما حذف کردن یال‌های با ناهمواری کم است که بین نواحی قرار دارند، تا به این شکل، این نواحی ناهمبند شوند. سپس کوتاه‌ترین مسیرها، در صورتی که شامل i زیر هدف باشند، به $i+1$ قطعه شکسته می‌شوند، که هر یک از قطعات بخشی از مسیر هستند که بین دو زیر هدف قرار گرفته و در نتیجه همگی یک قطعه متعلق به یک انجمن خاص می‌باشند. حال کافیست به مرکزیت راس میانی هر کدام از این قطعات مسیر، جستجوی سطح اول[۱۰۰] انجام شود. از آن‌جایی که فرض کرده‌ایم نواحی گراف با حذف یال‌ها ناهمبند شده‌اند، انتظار می‌رود راسی خارج از ناحیه یافت نشود. برای اطمینان بیش‌تر، به این دلیل که ممکن است ناحیه‌ای کاملاً ناهمبند نشده باشد، می‌توان جستجوی سطح اول را به حداکثر عمق خاصی، که ضریبی از طول هر یک از قطعات است، محدود کرد. در شکل (۴-۷) نمایی از این روش ارائه شده است.



شکل (۴-۷): نمایی از جداسازی نواحی و به‌دست آوردن خوشه‌ها

[100] Breadth First Search (BFS)

برای ساخت سیاست‌های جزئی مهارت‌ها از روش بازبینی تجربه[۱۰۱] استفاده می‌شود. نحوه‌ی کار به این شکل است که سابقه‌ی تعاملات عامل شامل حالت‌های گذر و پاداش دریافتی مربوطه نگه‌داری شده و بعداً برای آموزش دادن سیاست‌های جزئی بهینه مورد استفاده قرار می‌گیرد. برای این‌که سیاست‌های جزئی به سمت زیرهدف‌ها رهنمون شوند، یک پاداش مجازی برای رسیدن به هر یک از زیرهدف‌ها در حالت‌های دامنه‌ی آن گزینه، در نظر گرفته می‌شود.

۴-۴ جمع‌بندی

در این فصل به بسط و تشریح روش پیشنهادی پرداختیم. دیدیم که بسیاری از مهارت‌هایی که در روش‌های معمول این رده وجود دارد، عملاً مفید نبوده و در صورت انتخاب هر یک از آن‌ها عامل از رسیدن به هدف خود دور می‌شود. به همین دلیل، در این روش سعی بر پیدا کردن مهارت‌هایی است که در جهت نیل عامل به هدف باشند. از دسته روش‌های بهینه‌سازی کلونی مورچه برای کشف زیرهدف‌ها استفاده کردیم و دیدیم که نوع تغییرات فرومون هر یک از یال‌ها حین انجام بهینه‌سازی، می‌تواند معیار جداساز مناسبی برای یال‌های مجاور با زیرهدف و دیگر یال‌ها باشد و از همین خاصیت برای کشف زیرهدف‌های مفید استفاده کردیم. در فصل آینده به بررسی عملکرد این روش مقایسه‌ی آن با دیگر روش‌ها خواهیم پرداخت.

[101] Experience Replay

فصل پنجم
نتایج عملی

این فصل به بررسی نتایج حاصله از روش پیشنهادی می‌پردازد. برای تحلیل میزان کارایی این روش، آن را با برخی از روش‌های دیگر یادگیری مقایسه خواهیم نمود. مقایسه‌ها در بستر محیط‌هایی انجام خواهد گرفت که از پیچیدگی نسبی و ساختار سلسله مراتبی برخوردار باشند.

در ادامه‌ی این فصل، ابتدا چهار محیط اتاق‌ها، تاکسی، اتاق بازی و برج‌های هانوی معرفی خواهند شد و در ادامه، نتایج عملی روش پیشنهادی روی این محیط‌ها بررسی می‌شوند. سپس مستنداتی مربوط به حساسیت روش به پارامترهای الگوریتم ارائه شده و در نهایت مقایسه‌ای بر نتایج خروجی انجام خواهد شد.

۵-۱ محیط‌های انجام آزمایش

در این قسمت، به معرفی و بسط محیط‌های انجام آزمایش خواهیم پرداخت. به ازای هر محیط، قوانین حاکم بر آن، گراف گذر فضای حالت و مجموعه‌ی مهارت‌های مربوط، ذکر خواهد گردید.

۵-۱-۱ محیط اتاق‌ها

محیط اتاق‌ها، یک جهان مشبک مستطیلی است که برخی از خانه‌های آن آزاد و باقی بسته می‌باشد. از نحوه‌ی قرارگیری خانه‌های بسته، معمولاً چندین اتاق شکل می‌گیرد که توسط یک یا چندین درب به یکدیگر متصل شده‌اند. عامل کار خود را از یکی از خانه‌ها شروع کرده و باید به یک خانه‌ی مشخص به عنوان هدف نقل مکان کند. کنش‌های مجاز، حرکت در یکی از چهار جهت بالا، چپ، راست و پایین می‌باشد. در صورتی که جهت

انتخاب شده‌ی عامل مسدود باشد، تغییری در مکان عامل رخ نمی‌دهد و در غیر این صورت با احتمال ۰/۹ در آن جهت حرکت کرده و به احتمال ۰/۱ به صورت تصادفی به یکی از خانه‌های مجاور برده می‌شود.

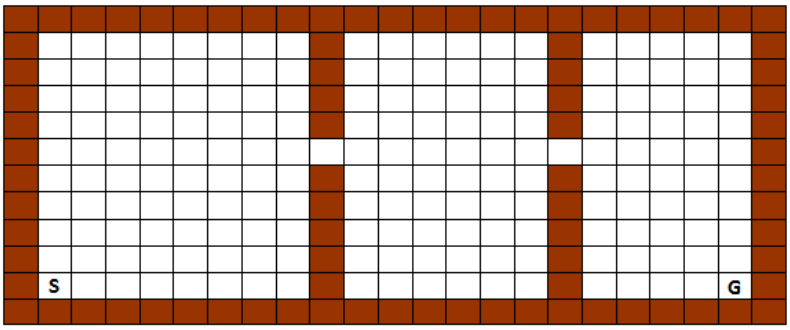
هر کنش جریمه‌ی ۱- در بر خواهد داشت، مگر این‌که عامل را به خانه‌ی هدف برساند که در این صورت پاداش ۱۰۰۰+ به عامل داده خواهد شد. در صورتی‌که محیط را به صورت گراف گذر مدل کنیم، هر راس معادل یک حالت خواهد بود و حداکثر با ۴ یال مجاورت خواهد داشت. بنابراین در این گراف، تعداد یال‌ها و راس‌ها دارای رابطه‌ی $m = \theta(n)$ می‌باشند.

دو چینش از اتاق‌ها برای آزمایش‌های عملی در نظر گرفته شده است. یکی محیط سه اتاقه با دو درب میانی و دیگری محیط شش اتاقه با هفت درب می‌باشد. محیط اول شامل ۲۷۶ حالت و محیط دوم دارای ۸۰۰ حالت می‌باشند. در شکل (۵-۱) (آ) و (ب) نمایی از این دو محیط آورده شده است.

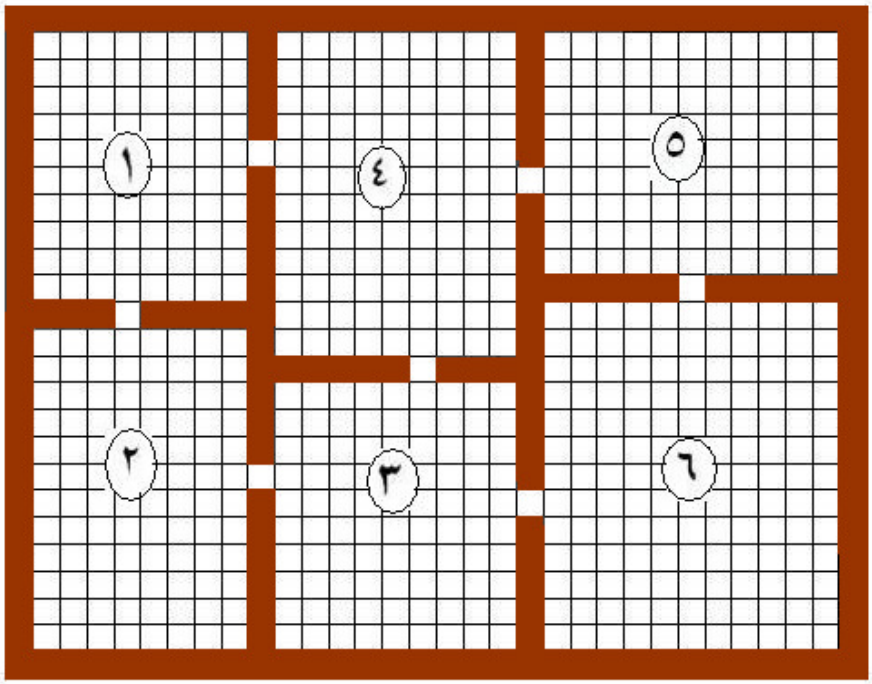
در این محیط‌ها، مهارت‌ها مربوط به انتقال از هر یک حالت‌های یک اتاق به یکی از درب‌های متصل به آن می‌باشد. همان‌طور که پیش‌تر هم گفته شده، همه‌ی مهارت‌ها برای رسیدن به هدف مفید نیستند بلکه تنها یادگیری مهارت‌هایی برای عامل مفید خواهند بود که در راستای هدف باشند.

در محیط سه اتاقه، رفتن از حالت‌های اتاق ۱ به درب مابین اتاق ۱ و ۲، رفتن از حالت‌های اتاق ۲ به درب مابین اتاق ۲ و ۳ و نقل مکان از حالت‌های اتاق ۳ به حالت هدف، مهارت‌های مفید می‌باشند.

در محیط شش اتاقه، با فرض این‌که حالت شروع در اتاق ۱ و حالت هدف در اتاق ۶ قرار گیرد، ۹ مهارت مفید می‌تواند در این محیط تعریف شود. باید دقت کرد که روش پیشنهادی، تنها مهارت‌های واقع روی کوتاه-ترین مسیر یافت شده را شناسایی خواهد کرد و به این صورت ۴ مهارت در این محیط توسط الگوریتم پیشنهادی باید کشف گردد.



(آ) محیط ۳ اتاقه با دو درب میانی



(ب) محیط ۶ اتاقه با ۷ درب میانی

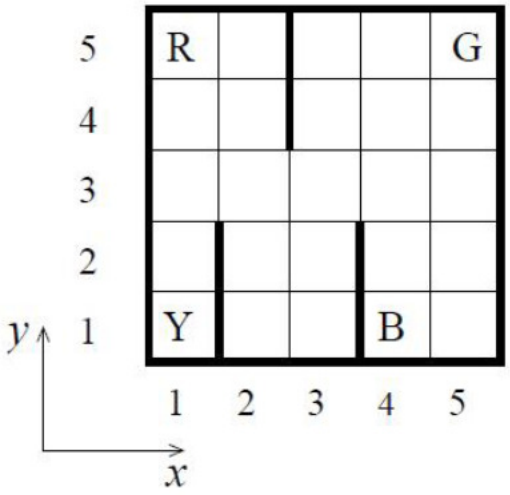
شکل (۵-۱): نمایی محیط‌های چنداتاقه‌ی مورد آزمایش

۵-۱-۲ محیط تاکسی

این محیط[۳۴] شامل یک جدول ۵×۵ می‌باشد که بین برخی از خانه‌های آن، مطابق شکل (۵-۲)، موانعی قرار گرفته است. در این جدول، چهار خانه با رنگ‌های آبی B، سبز G، قرمز R و زرد Y مشخص شده‌اند. مکان اولیه‌ی مسافر و مقصد او در دو خانه از این چهار خانه‌ی مشخص قرار دارند. برای تاکسی، به عنوان عامل در این محیط، شش کنش تعریف شده است. این شش کنش شامل چهار کنش حرکتی در جهت‌های بالا، راست، پایین

و چپ و دو کنش برای سوارکردن و پیاده‌کردن مسافر می‌باشد. کنش‌های حرکتی در راستای موانع، پیاده‌کردن مسافر در شرایطی که سوار ماشین نباشد و همچنین سوارکردن مسافر، در صورتی که سوار ماشین باشد، بی اثر خواهند بود. در صورتی که کنش حرکتی با مانع برخورد نکند، با احتمال ۰/۸ موفقیت آمیز خواهد بود و در غیر این صورت به سمت راست یا چپِ جهت درخواستی رانده خواهد شد.

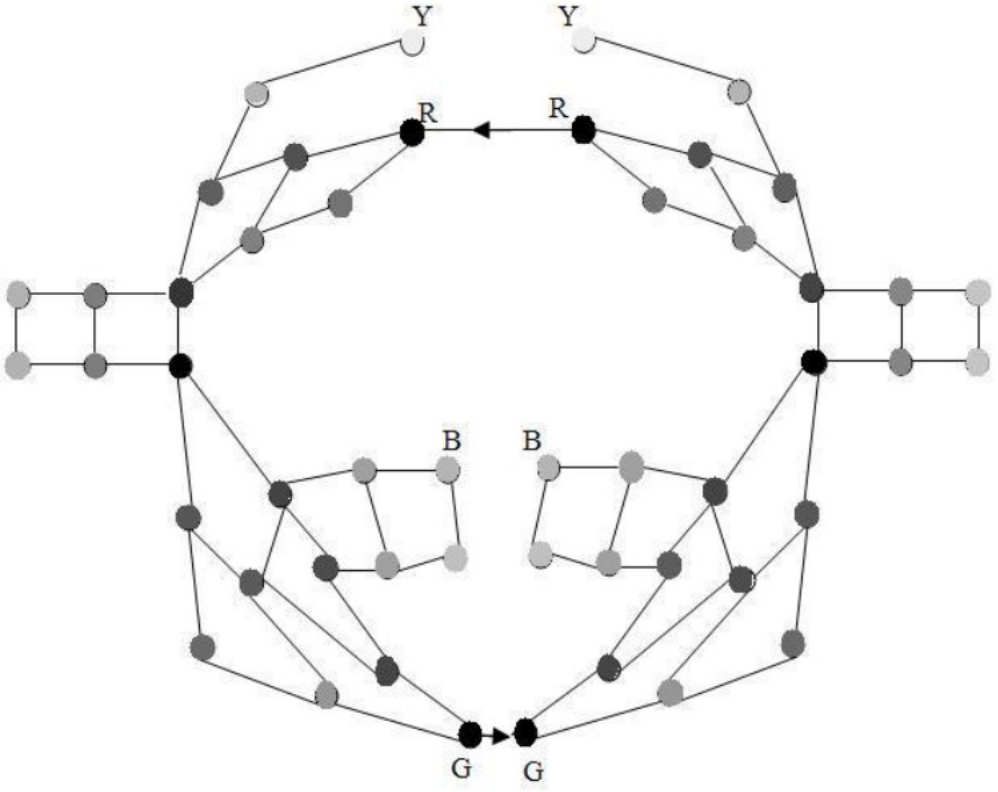
سوارکردن مسافر پاداش ۱۰+ و پیاده‌کردن وی در مقصد، پاداش ۲۰+ را در پی خواهد داشت. در صورتی که کنش سوارکردن در محلی غیر از محل مبدا مسافر صورت گیرد، یا عامل مسافر را در محلی غیر از مقصد پیاده نماید، کنش‌ها بی‌تاثیر بوده و عامل جریمه‌ی ۱۰- را دریافت خواهد کرد. هر کنش دیگر که شامل موارد فوق نباشد، جریمه‌ی ۱- خواهد داشت.



شکل (۵-۲): نمایی از محیط تاکسی[۳۴]

در شکل (۵-۳)، گراف گذر حالت محیط تاکسی مشاهده می‌شود. همان‌طور که دیده می‌شود، در این گراف، ۲۵ حالت نیمه‌ی سمت چپ، مربوط به شرایطی است که در آن تاکسی مسافر را سوار نکرده و نیمه‌ی متقارن سمت راست آن، نشان‌دهنده‌ی حالت‌هایی است که مسافر سوار تاکسی می‌باشد. همان‌طور که در این گراف مشاهده می‌شود، در این محیط، دو مهارت نقش اساسی در انجام هر دوره از تعامل با محیط خواهند داشت:

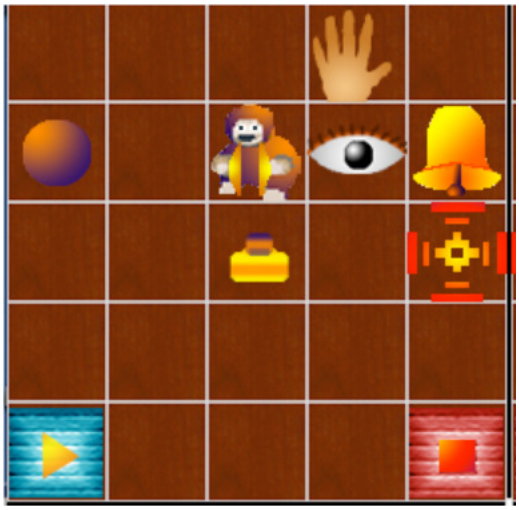
مهارت نخست، رساندن تاکسی از محل شروع به محل مبدا مسافر و سوار کردن وی خواهد بود. مهارت دوم نیز رساندن مسافر به مقصد و پیاده کردن وی می‌باشد.



شکل (۵-۳): گراف گذر محیط تاکسی، با این فرض که مبدا و مقصد به ترتیب در حالت‌های G و R می‌باشند[۲].

۵-۱-۳ محیط اتاق بازی

اتاق بازی[۳۵] یکی دیگر از محیط‌هایی است که مقایسه‌ها روی آن صورت خواهد گرفت و تاحدی به نسبت محیط‌های دیگر، قواعد پیچیده‌تری دارد. در این محیط مطابق آن‌چه در شکل (۵-۴) دیده می‌شود، تعدادی شی وجود دارد: یک کلید چراغ، یک توپ، یک زنگ، دو بلوک قابل جابجایی به رنگ‌های قرمز و آبی که می‌توانند به عنوان کلیدی برای قطع و وصل کردن موسیقی به‌کار روند و نهایتاً یک میمون که می‌تواند جیغ بکشد.



شکل(۵-۴): نمایی از محیط اتاق بازی[۳۵]

حس‌گرهای عامل شامل یک چشم، یک دست و یک نشان‌گر می‌باشد. عامل می‌تواند هر شی‌ای که در محل قرار گیری هر یک از حس‌گرهای خود وجود داشته باشد را دریابد.

در هر یک از گام‌های زمانی، عامل می‌تواند یکی از این کنش‌ها را انتخاب کند: ۱) چشم را به محل دست حرکت دهد. ۲) چشم را به محل نشان‌گر ببرد. ۳) چشم را یک قدم در جهت‌های بالا، راست، پایین و چپ حرکت دهد. ۴) چشم را به محل یک شی تصادفی ببرد. ۵) دست را به محل چشم ببرد. ۶) نشان‌گر را به محل چشم ببرد. علاوه بر این‌ها، در صورتی‌که چشم و دست، هر دو روی یک شی قرار داشته باشند، کنش مربوط به آن شی قابل انجام می‌گردد. اگر هردو روی کلید باشند، روشن یا خاموش کردن چراغ ممکن می‌گردد. در صورتی که هر دو روی توپ قرار گیرند، عامل می‌تواند به آن ضربه بزند، که باعث حرکت توپ در خط مستقیم به سمت نشان‌گر خواهد شد.

اشیا موجود در اتاق بازی ویژگی‌های خاصی دارند. در صورتی که توپ حین حرکت به زنگ برخورد کند، زنگ یک لحظه به صدا در آمده و در جهت تصادفی به یکی از خانه‌های مجاور خواهد رفت. کلید چراغ کنترل روشنایی اتاق را بر عهده دارد. رنگ هر کدام از بلوک‌ها در صورتی قابل مشاهده است که اتاق روشن باشد، در

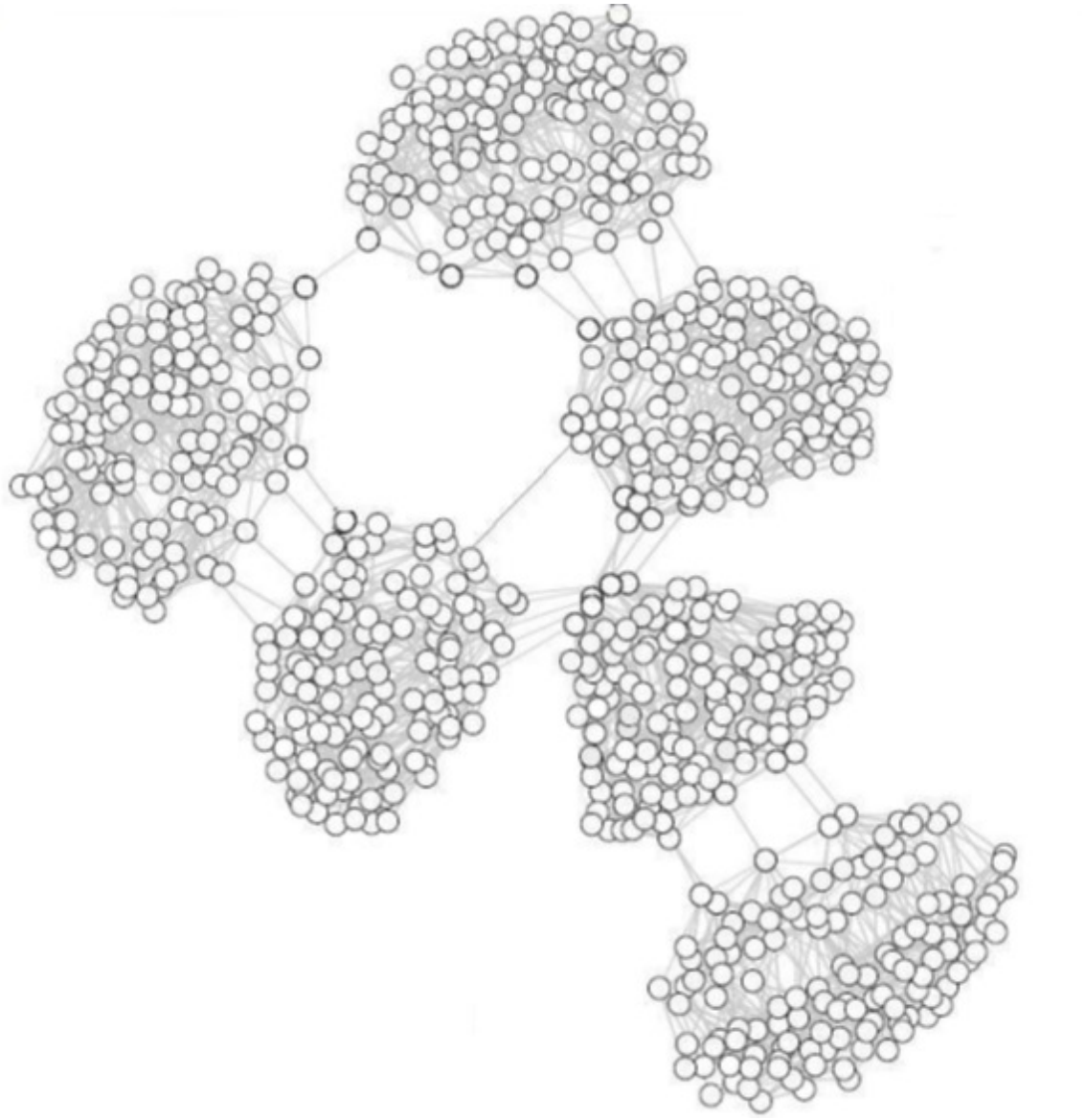
غیر این‌صورت، هر دو به صورت مشابه، خاکستری دیده می‌شوند. در صورتی که بلوک آبی فشار داده شود، پخش موسیقی وصل شده و مشابهاً بلوک قرمز می‌تواند پخش آن را قطع کند. هر کدام از بلوک‌ها می‌توانند فشار داده شوند و در این صورت به یک خانه‌ی تصادفی مجاور خواهند رفت. میمون نیز در صورتی که اتاق تاریک باشد و موسیقی و زنگ هم‌زمان به صدا در آمده باشند، می‌ترسد و جیغ می‌کشد.

هدف عامل در این محیط، ترساندن میمون برای جیغ کشیدن او می‌باشد. رسیدن به این حالت برای عامل پاداش ۱۰۰۰+ خواهد داشت، اما برای کمینه کردن تعداد کنش‌ها، برای هر عمل جریمه‌ی ۱- در نظر گرفته شده است.

برای آن‌که میمون جیغ بکشد، عامل باید این ترتیب از کنش‌ها را اجرا کند: ۱) بردن چشم خود به محل کلید چراغ ۲) بردن دست به محل چشم ۳) روشن کردن چراغ با استفاده از کلید چراغ ۴) پیدا کردن بلوک آبی با چشم ۵) بردن دست به محل چشم ۶) فشار دادن بلوک آبی برای وصل کردن پخش موسیقی ۷) پیدا کردن کلید چراغ با چشم ۸) بردن دست به محل کلید چراغ ۹) فشار دادن کلید برای خاموش کردن چراغ ۱۰) پیدا کردن زنگ با چشم ۱۱) بردن نشان‌گر به محل چشم ۱۲) پیدا کردن توپ با چشم ۱۳) بردن دست به محل چشم ۱۴) زدن ضربه به توپ برای به صدا در آوردن زنگ.

توجه کنید که در صورتی که عامل مهارت‌هایی برای روشن و خاموش کردن چراغ، وصل و قطع کردن موسیقی و به صدا در آوردن زنگ یاد بگیرد، می‌تواند از این مهارت‌ها برای رسیدن هر چه سریع‌تر به هدف خود استفاده کند.

در شکل (۵-۵) نمایی از گراف گذر این محیط آورده شده است. همان‌طور که دیده می‌شود این گراف خاصیت انجمنی دارد و هر انجمن متعلق به حالت‌های مربوط به انجام یک مهارت خاص می‌باشد.



شکل (۵-۵): گراف گذر حالت برای محیط اتاق بازی[۳۵]

۵-۱-۴ محیط برج‌های هانوی

محیط برج‌های هانوی، شامل سه میله و تعدادی دیسک با اندازه‌های متفاوت می‌باشد. در ابتدای کار، همه‌ی دیسک‌ها به ترتیب از کوچک به بزرگ روی هم قرار دارند. هدف بردن همه‌ی این دیسک‌ها به یک میله‌ی دیگر است به طوری چند قانون رعایت شود: ۱) در هر کنش تنها یک دیسک می‌تواند حرکت داده شود. ۲) نباید روی

دیسک برداشته شده، دیسک دیگری موجود باشد. ۳) هیچ دیسکی نمی‌تواند روی یک دیسک کوچک‌تر از خود قرار بگیرد.

کنش‌های ممکن برای عامل در هر حالت، جابجا کردن یکی از دیسک‌ها از یک میله به میله‌ی دیگر است به شکلی که قواعد فوق رعایت شوند. در صورتی که عامل موفق به انجام وظیفه‌ی خود شود، پاداش ۱۰۰۰+ و به ازای هر حرکت دیگر، جریمه‌ی ۱- دریافت خواهد کرد.

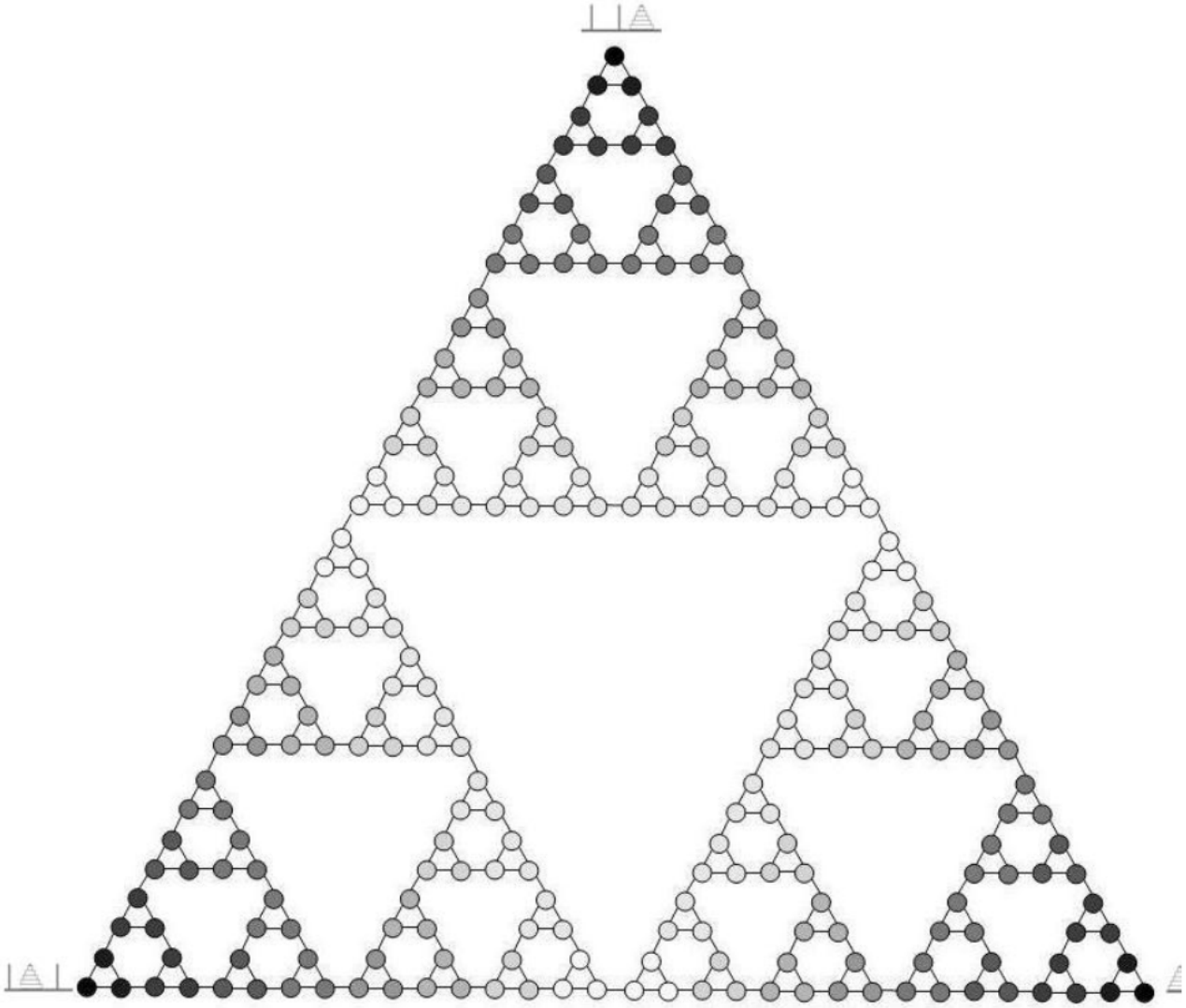
مسئله‌ای که در اینجا در نظر گرفته شده، محیط برج‌های هانوی با ۵ دیسک است، که شامل ۲۴۳ حالت می‌باشد. در شکل (۵-۶) نمایی از این محیط نمایش داده شده است.

شکل (۵-۶): نمایی از محیط برج‌های هانوی[۲]

در شکل (۵-۷) گراف مربوط به گذر در فضای حالت این محیط، نمایش داده شده است. در این گراف انجمن‌هایی به شکل سلسله مراتبی دیده می‌شود. به صورت معادل می‌توان مهارت‌هایی را به صورت سلسله مراتبی در آن تشخیص داد. حل مسئله برای n دیسک، شامل سه قدم می‌باشد: نخست منتقل کردن $n-1$ دیسک از میله‌ی مبدا به میله‌ی کمکی، دوم بردن بزرگ‌ترین دیسک از میله‌ی مبدا به میله‌ی مقصد و سوم بردن $n-1$ دیسک از میله‌ی کمکی به میله‌ی مقصد.

در این فرایند، دو مهارت قابل فراگیری موجود می‌باشد: بردن $n-1$ دیسک از میله‌ی مبدا به میله‌ی کمکی و انقال همین $n-1$ دیسک از میله‌ی کمکی به میله‌ی مقصد. به شکل مشابه می‌توان برای ساخت هر کدام از

این مهارت‌ها، از دو مهارت کوچک‌تر استفاده کرد. به عنوان مثال برای کسب مهارت دوم، می‌توان از دو مهارت استفاده کرد: انتقال $n-2$ دیسک از میله‌ی کمکی به میله‌ی مبدا و انتقال این $n-2$ دیسک از میله‌ی مبدا به میله‌ی مقصد.



شکل (۵-۷): گراف گذر فضای حالت برای محیط برج‌های هانوی با ۵ دیسک[۲]

۵-۲ سنجش حساسیت روش به پارامترها

نحوه‌ی عملکرد بسیاری از راه‌حل‌های خودکاری که سیستم‌های هوشمند برای مسائل ارائه می‌دهند، بستگی مستقیم به مقدار پارامترهای آن راه‌حل‌ها دارد. به شکلی که به نظر می‌رسد، بخشی از هوشمندی لازم برای حل مسئله در تعیین پارامترها نهفته است، که توسط طراح انجام می‌گیرد.

ماهیت روش مبتنی بر فرومون، به گونه‌ای است که پارامترهای بسیاری در آن دخیل هستند. به همین دلیل سعی شده است که تا جای ممکن، حساسیت روش به پارامترها به صورت دقیق سنجیده شود. برای این کار در مورد بیش‌تر پارامترها، مکانیسم ساده‌ای در نظر گرفته شده است. الگوریتم با مقادیر مختلف برای هر پارامتر اجرا شده و میانگین رتبه‌ی تخصیص داده شده توسط روش پیشنهادی به یال مجاور با زیرهدف واقعی سنجیده خواهد شد. واضح است که هرچه این میانگین کوچک‌تر باشد، الگوریتم موفق‌تر عمل کرده است.

پارامترهایی که در این الگوریتم وجود دارند عبارتند از: تعداد تعاملات با محیط برای کشف ساختار گراف گذر (N)، تعداد دوره‌ی اجرای الگوریتم مورچه (n_t)، تعداد مورچه‌ها (n_k)، ضریب تبخیر (ρ)، ضریب اهمیت انتفاع در برابر کاوش (α)، ضریب آستانه‌ی مقدار نمودار یال‌های کاندید (τ_v) و ضریب آستانه‌ی شیب نمودار یال‌های کاندید (τ_d). با توجه به این‌که تعداد پارامترهای الگوریتم زیاد است، غیر از تاثیر پارامتر N که روی همه‌ی محیط‌ها بررسی می‌شود، نتایج این آزمایش‌ها برای باقی پارامترها روی محیط تاکسی آورده و تحلیل خواهد شد. حساسیت‌سنجی پارامترها می‌تواند به صورت مشابه روی محیط‌های دیگر نیز انجام شود.

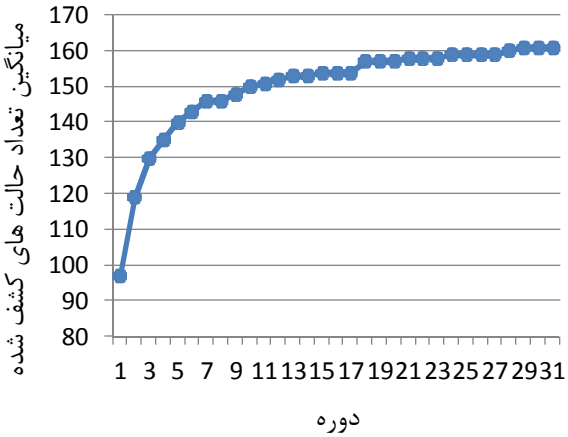
پارامترهای n_t، n_k، ρ و α مربوط به الگوریتم مورچه می‌باشند، همان‌طور که در ادامه دیده خواهد شد، این پارامترها به اندازه‌ی دو پارامتر بعدی (τ_v و τ_d) در کارایی روش پیشنهادی تاثیرگذار نخواهند بود، چراکه در صورت دادن مقدار نادرست به آن‌ها، اصلی‌ترین تفاوت در کوتاه‌ترین مسیر یافت شده خواهد بود که تاثیر چشم-گیری در کارایی الگوریتم ندارد.

۵-۲-۱ تنظیم پارامتر N

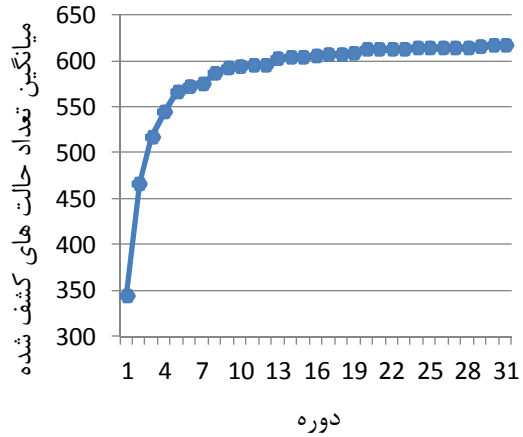
پارامتر N، بیان‌گر تعداد دوره‌های اولیه‌ی تعامل با محیط، برای به‌دست آوردن گراف گذر فضای حالت می‌باشد. تنظیم این پارامتر از دو جنبه برای ما حائز اهمیت است: نخست این‌که در صورتی‌که مقدار این پارامتر کم‌تر از مقدار لازم باشد، گراف گذر شامل تعداد کم‌تری از راس‌ها و یال‌ها خواهد بود و به این ترتیب ممکن است مبنای ادامه‌ی کار که تحلیل این گراف است، صحیح نباشد. از طرف دیگر، طی کردن دوره‌هایی بیش از حد لازم برای تعامل با محیط به دلایلی که در فصل گذشته ذکر کردیم، علاوه بر بی‌فایده بودن، هزینه‌ی زیادی در پی خواهد داشت و کارایی الگوریتم کسب مهارت را کم‌تر می‌کند.

برای بررسی مقدار بهینه برای پارامتر N، در محیط‌های مختلف، حالت‌های تصادفی برای شروع و پایان دوره در نظر گرفته شدند و میانگین تعداد حالات کشف شده در هر دوره به‌دست آمده است. نمودارهای موجود در شکل (۵-۸) (آ) تا (د) این مقادیر را برای محیط‌های مورد آزمایش نشان می‌دهد. نمودارهای فوق حاصل میان-گیری بین ۱۰۰ بار اجرای برنامه می‌باشند.

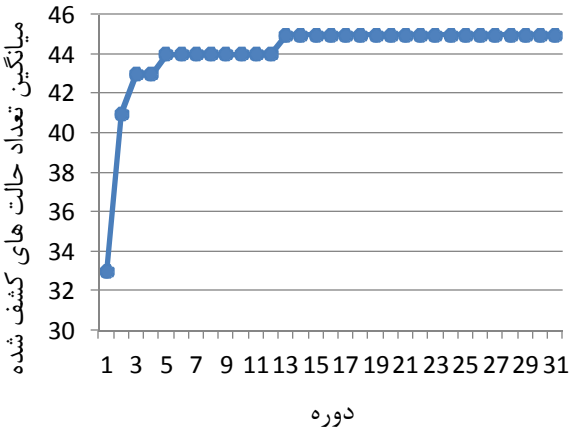
از آن‌جایی که راس‌هایی که بعد از چندین دوره در مسیرهای به سمت هدف قرار نمی‌گیرند، قطعاً جز راس-های مهم در یافتن مسیر در الگوریتم کلونی مورچه نخواهند بود، پیدا کردن اکثریت حالت‌ها کافی بوده و نیازی دیدن تمام حالت‌ها نیست. از روی شکل منحنی نمودارهای حاصله، می‌توان مقادیر مناسب N را برای هر محیط، تعیین کرد. این پارامتر در محیط‌های مورد آزمایش، می‌تواند این مقادیر را اختیار کند: در محیط سه اتاقه ۱۹، در محیط شش اتاقه و تاکسی، ۱۵، در محیط برج‌های هانوی، ۳ و در محیط اتاق بازی، ۲۰ دوره برای انجام تعامل اولیه کافی است.



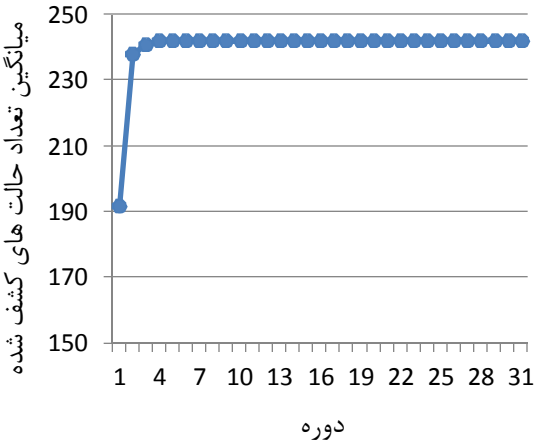
(آ) محیط سه اتاقه



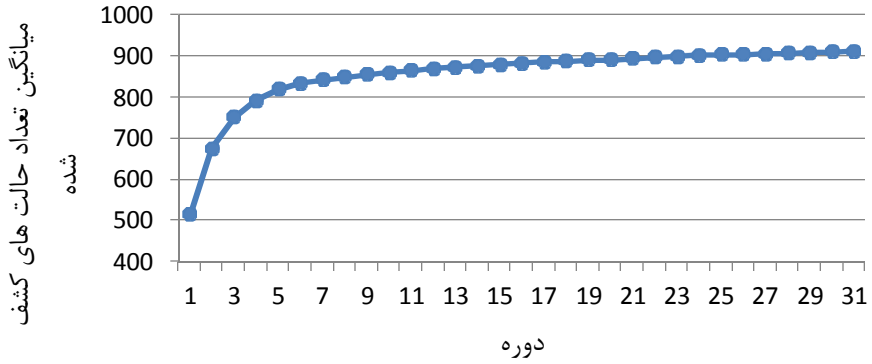
(ب) محیط شش اتاقه



(ج) محیط تاکسی



(د) محیط برج‌های هانوی



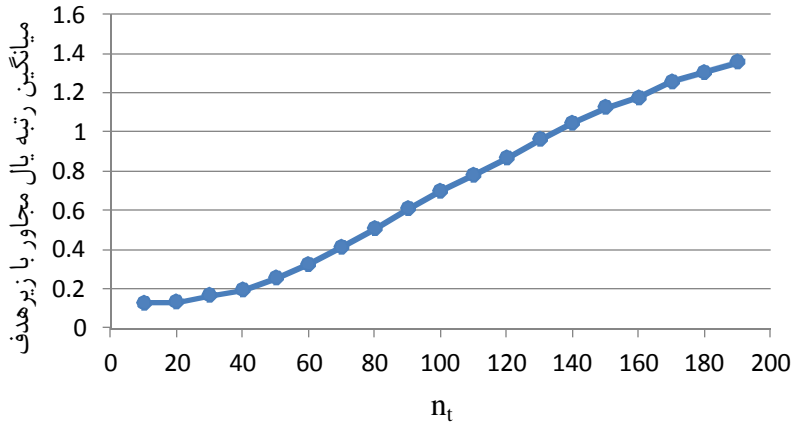
(ه) محیط اتاق بازی

شکل(۵-۸): نمودار میانگین تعداد حالت‌های کشف شده در دوره‌های مختلف برای محیط‌های مورد آزمایش

۵-۲-۲ حساسیت به پارامتر n_t

پارامتر n_t بیان‌گر تعداد دوره‌های اجرای الگوریتم مورچه می‌باشد. در صورتی که مقدار انتخاب شده برای این پارامتر فاصله‌ی زیادی با مقدار بهینه‌ی آن داشته باشد، این مسئله می‌تواند تا حدی بر سرعت و دقت اجرای الگوریتم تاثیرگذار باشد. باید توجه نمود که این پارامتر با مقادیر کم‌تر، پاسخ مناسب‌تری را از سیستم دریافت خواهد نمود، چراکه تفاوت اصلی میزان ناهمواری یال‌ها در دوره‌های اولیه خواهد بود، زیرا پس از گذشت زمان قابل توجه، میزان فرومون هر دو دسته یال، به یک مقدار همگرا خواهد شد و شیب نمودار فرومون به صفر میل خواهد نمود که باعث نزدیک‌تر شدن مقدار ناهمواری این دو دسته خواهد شد. بنابراین بهتر است در دوره‌های اولیه، این دو دسته را از هم جدا نمود. به این شکل، از نظر زمانی نیز برنامه با سرعت بیش‌تری به جواب خواهد رسید.

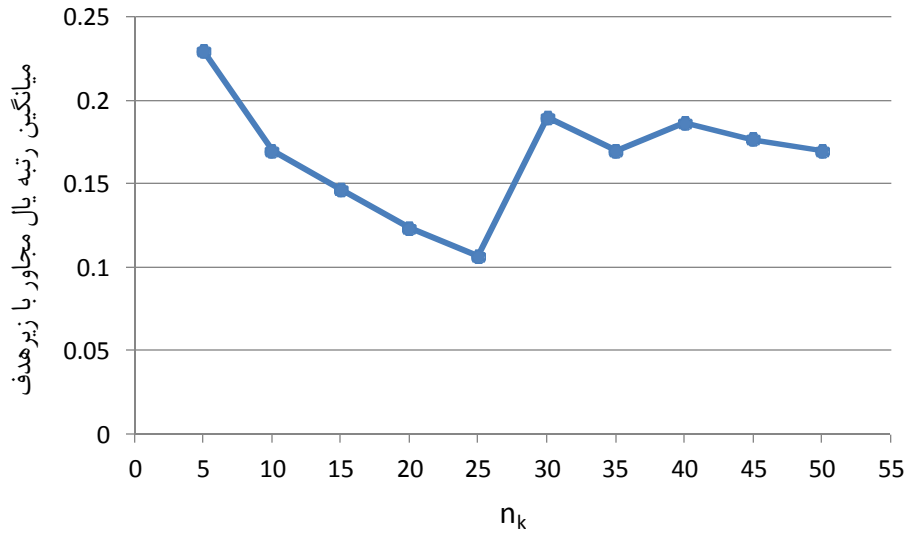
کارایی الگوریتم با معیار میانگین رتبه‌ی یال مجاور با زیرهدف و میانگین‌گیری آن روی ۱۰۰ بار تکرار، به ازای مقادیر مختلف برای n_t در بازه‌ی ۱۰ تا ۲۰۰ با گام ۱۰ اندازه‌گیری شد، که نتیجه‌ی آن در شکل (۵-۹) قابل مشاهده است. آن‌چه از آهنگ افزایش کارایی (کاهش میانگین تعداد گام تا هدف) در نمودار بر می‌آید، نشان می‌دهد، در محیط تاکسی، عدد ۱۰ مناسب‌ترین مقدار برای این پارامتر می‌باشد.



شکل (۵-۹): نمودار میانگین رتبه‌ی یال مجاور با زیرهدف، بر حسب مقادیر مختلف n_t

۵-۲-۳ حساسیت به پارامتر n_k

پارامتر n_k نشان‌دهنده‌ی تعداد مورچه‌هایی است که در هر دوره از اجرای الگوریتم، مسیری را به سمت هدف شکل می‌دهند. در رفتار واقعی مورچه‌ها، این مورد را می‌توان به حرکت دسته‌ای مورچه‌ها نسبت داد. سوالی که در این‌جا مطرح می‌شود، این است که چرا باید بهینه‌سازی کلونی مورچه را به بهینه‌سازی مورچه ترجیح داد و به جای یک مورچه از یک دسته مورچه استفاده کرد؟ در نظر گرفتن هم‌زمان چند مورچه در الگوریتم بهینه-سازی مورچه را به دو دلیل می‌توان توجیه کرد، نخست بالاتر بردن میزان کاوش به نسبت انتفاع در صورت وجود چند مورچه و دوم، پایین‌تر بردن میزان پیچیدگی زمانی. در آزمایش‌های عملی مشخص شده‌است[۳۲] که همگرایی به کوتاه‌ترین مسیر، با تعداد کم مورچه‌ها به خوبی حاصل می‌شود، اما برای مقادیر بزرگ n_k، الگوریتم این همگرایی را از خود بروز نمی‌دهد. نتایج تغییرات n_k در کارایی نهایی الگوریتم در شکل (۵-۱۰) نمایش داده شده است. این نمودار میانگین رتبه‌ی یال مجاور با زیرهدف که در ۳۰۰ بار تکرار مجدداً میانگین-گیری شده است، بر حسب مقادیر مختلف n_k نمایش می‌دهد. مقدار n_k از ۵ تا ۵۰ با گام ۵ تغییر کرده است.



شکل (۵-۱۰) نمودار میانگین رتبه‌ی یال مجاور با زیرهدف برحسب مقادیر مختلف n_k

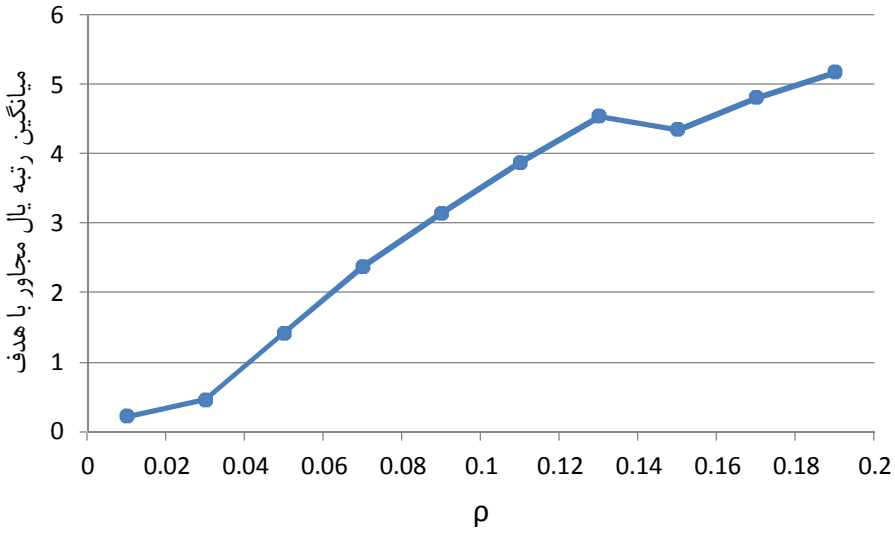
با توجه به آن‌چه گفته شد و مشاهده‌ی خروجی الگوریتم، مقدار ۲۵ برای پارامتر n_k در این مسئله، مناسب به نظر می‌رسد.

۵-۲-۴ حساسیت به پارامتر ρ

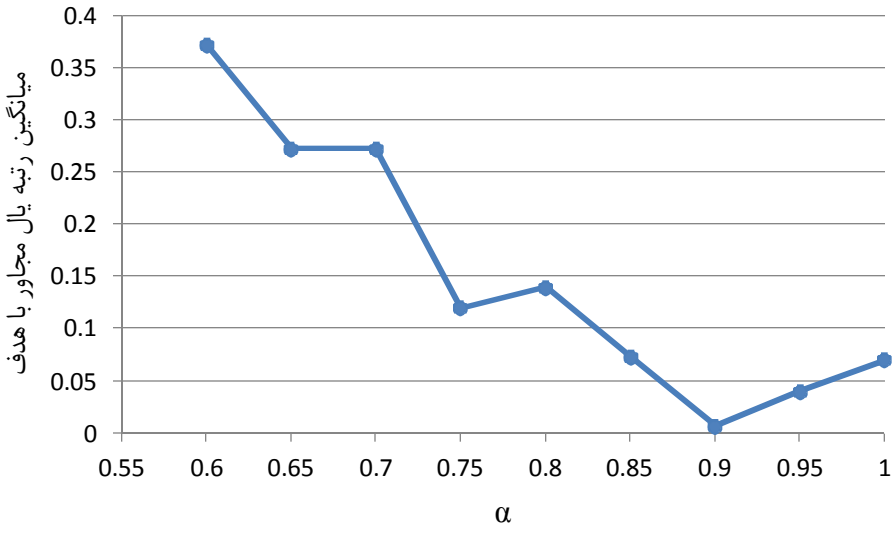
ضریب تبخیر، نرخی است که متناسب با آن تجربیات قبلی نادیده گرفته می‌شوند و به نوعی نشان‌دهنده‌ی میزان کاوش در الگوریتم می‌باشد. اگر برابر صفر باشد، الگوریتم به کوتاه‌ترین مسیر یا مسیر طولانی‌تر دیگری همگرا نخواهد شد[۳۲] و اگر این میزان زیاد باشد، به یک مسیر غیربهینه همگرا خواهد شد. بنابراین یک مقدار غیرصفر باید برای این پارامتر به‌دست آید. اما از آن‌جایی که در این روش، برای کارایی بهینه مقدار n_t کوچک در نظر گرفته می‌شود، باید در این تعداد دوره‌ی کم به خوبی از تجربیات کسب شده استفاده شود، چرا که درصورتی‌که این تجربیات بیش‌تر نادیده گرفته شود، با توجه به فرصت کم، الگوریتم همگرا نخواهد شد. بنابراین پیش‌بینی می‌شود مقادیر کوچک‌تر ρ در این‌جا مناسب‌تر باشد. شکل (۵-۱۱) نمودار حاصل از اجرای این الگوریتم را نشان می‌دهد. این نمودار بیان‌گر میانگین رتبه‌ی یال مجاور با زیرهدف می‌باشد که خود ۱۰۰ بار تکرار شده و میانگین گرفته شده است. مطابق با نمودار، مقدار بهینه برای این پارامتر ۰/۰۱ می‌باشد.

۵-۲-۵ حساسیت به پارامتر α

این پارامتر نشان می‌دهد به چه اندازه باید به مقدار فرومون یک یال، در برابر وزن آن بها داد. اگر این پارامتر خیلی کوچک باشد، تاثیر میزان فرومون که نقش اصلی را در الگوریتم ایفا می‌کند، کم‌تر خواهد شد و مقدار بیش‌از اندازه زیاد این پارامتر، آن را از انتفاع از دانش قبلی محروم خواهد کرد. نمودار شکل (۵-۱۲)، حاصل میانگین‌گیری رتبه‌ی یال مجاور با زیرهدف است که ۱۵۰ بار تکرار و مجدداً میانگین گرفته شده است. مقدار مناسب میانه‌ای برای این محیط، با توجه نمودار، ۰/۹ می‌باشد.



شکل (۵-۱۱): میانگین رتبه‌ی یال مجاور با زیرهدف بر حسب مقادیر مختلف ضریب تبخیر ρ



شکل (۵-۱۲): میانگین رتبه‌ی یال مجاور با زیرهدف بر حسب مقادیر مختلف α

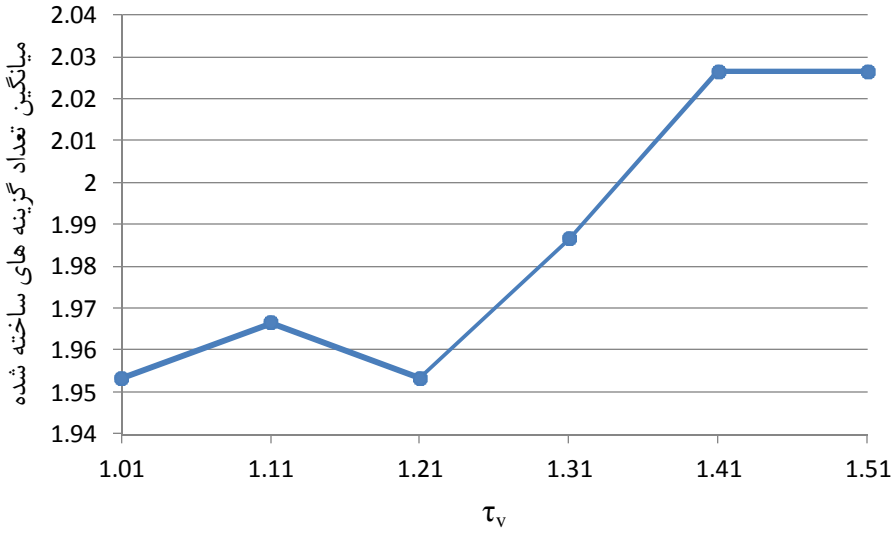
۵-۲-۶ حساسیت به پارامتر τ_v

این پارامتر برای جداسازی یال‌های مجاور با زیرهدف‌ها و دیگر یال‌ها در نظر گرفته شده و برای محدود کردن کیفیت یال‌های انتخاب شده، به نسبت بهترین یال به کار می‌رود. همان‌طور که در بخش ۵-۱-۲ گفته شد، در محیط تاکسی دو مهارت وجود دارد.

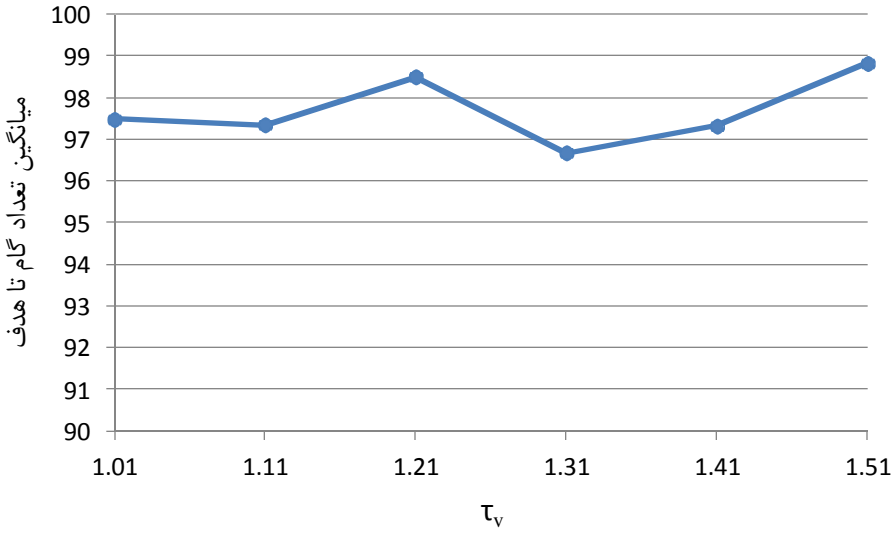
انتخاب پارامتر τ_v تاثیر مستقیم بر تعداد مهارت‌های ساخته شده خواهد داشت. در صورتی که مقدار این پارامتر بیش از حد مناسب باشد، احتمالاً حالت‌های بیش‌تری به عنوان زیرهدف در نظر گرفته می‌شوند و در نتیجه تعداد مهارت‌های ساخته شده بیش‌تر خواهد بود. نمودار شکل (۵-۱۳) (آ)، میانگین تعداد گزینه‌های ساخته شده در ۵۰ بار اجرای الگوریتم به ازای هر یک از مقادیر پارامتر τ_v نشان می‌دهد. هر چه این میانگین به عدد ۲ نزدیک‌تر باشد، الگوریتم عملکرد بهتری داشته و می‌توان نتیجه گرفت که پارامتر مقدار مناسب‌تری به خود گرفته است.

در نمودار نمایش داده شده در شکل (۵-۱۳) (ب)، میانگین تعداد گام‌ها تا رسیدن به هدف را در ۵۰ دوره‌ی اول که روی ۵۰ اجرا میانگین گرفته شده است، نمایش می‌دهد. طبق مشاهدات انجام شده، به نظرمی‌رسد مقدار ۱/۳۱ بهترین مقدار برای این پارامتر در محیط تاکسی می‌باشد. روش پیشنهادی برای حساسیت‌سنجی این پارامتر نیازمند دانستن تعداد مهارت‌های مفید می‌باشد.

دو پارامتر τ_d و τ_v به صورت متقابل در عملکرد دیگری تاثیر می‌گذارند. در بسیاری از محیط‌ها و از جمله در محیط تاکسی، توزیع میزان ناهمواری در یال‌های کاندید، به شکلی است که در صورتی که پارامتر τ_v به درستی مقداردهی شده باشد، نیازی به مقداردهی دقیق پارامتر τ_d نیست به همین دلیل برای رعایت اختصار، از آوردن نمودارهای مربوط به این پارامتر پرهیز می‌کنیم.



(آ) تاثیر پارامتر τ_v بر تعداد گزینه‌های به‌دست آمده



(ب) تاثیر پارامترτ_v بر میانگین تعداد گام تا رسیدن به هدف

شکل (۵-۱۳): تاثیر پارامترτ_v بر عملکرد برنامه

۵-۳ مقایسه با روش‌های دیگر

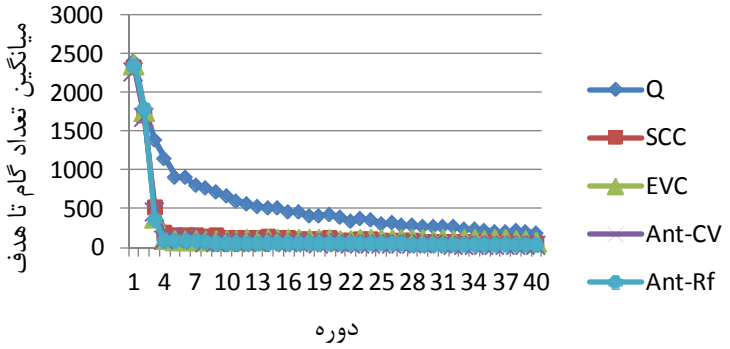
در این قسمت مقایسه‌ای از کارایی الگوریتم پیشنهادی با معیار ناهمواری (Ant-Rf) و معیار ضریب تغییرات (Ant-CV) با برخی از الگوریتم‌های دیگر، از جمله یادگیری Q، روش مولفه‌های قویاً همبند و روش مرکزیت بردار ویژه خواهیم دید که در نمودارها به ترتیب با نمادهای Q، SCC و EVC مشخص شده‌اند. معیار مقایسه، تعداد گام از شروع دوره تا رسیدن به هدف می‌باشد، که به جهت کاهش پدیده‌های تصادفی، روی۲۰ بار اجرا میانگین‌گیری شده‌اند.

پیش‌بینی می‌شود به علت در نظر گرفته شدن ترتیب مقادیر فرومون در معیار ناهمواری، کارکرد الگوریتم در مقایسه با معیار ضریب تغییرات، چه از نظر میانگین تعداد گام رسیدن تا هدف و چه از نظر زمان مورد نیاز برای همگرایی، بهتر باشد. برای حفظ اختصار از تکرار این مطلب برای مقایسه‌ی این دو روش در محیط‌های مختلف پرهیز خواهیم نمود.

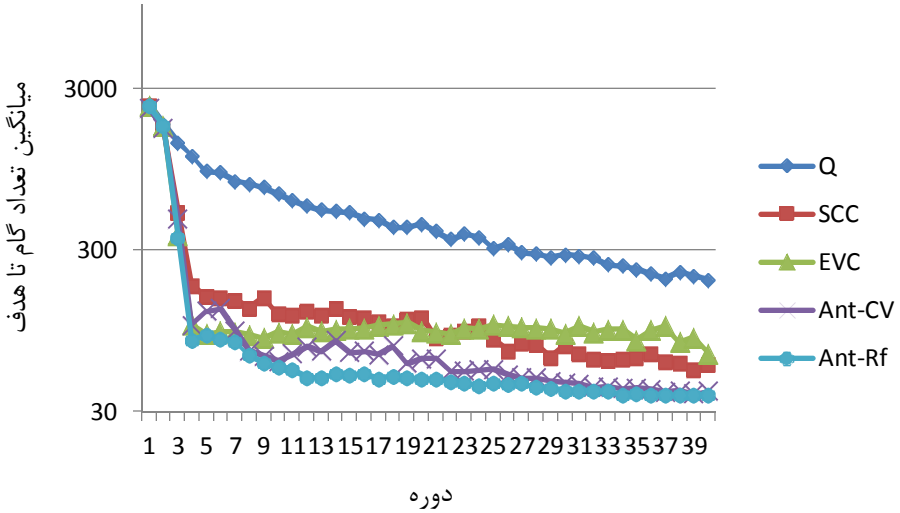
۵-۳-۱ محیط سه اتاقه

محیط سه اتاقه جزء محیط‌های ساده‌ی یادگیری تقویتی می‌باشد. در این محیط، با این فرض که اتاق‌ها از چپ به راست شماره‌گذاری شوند، الگوریتم‌های مولفه‌های قویاً همبند و مرکزیت بردار ویژه، مهارت‌هایی برای رفتن از اتاق ۱ به اتاق ۲، از اتاق ۲ به اتاق ۱، از اتاق ۲ به اتاق ۳ و از اتاق ۳ به اتاق ۲ تولید می‌کنند، اما در روش پیشنهادی به جای مهارت‌های غیرمفید رفتن از اتاق ۲ به اتاق ۱ و از اتاق ۳ به اتاق ۲، مهارتی برای رفتن از هر نقطه‌ای از اتاق ۳ به حالت هدف تولید می‌شود. نمودار مقایسه‌ی روش‌ها در شکل (۵-۱۴) (آ) دیده می‌شود. برای روش مولفه‌های قویاً همبند مقدار پارامتر t_t برابر ۲۲ است. در این شکل پیشرفت روش‌های کسب مهارت به نسبت روش یادگیری Q به خوبی مشاهده می‌شود. به دلیل نزدیکی نمودارهای مربوط که روش‌های کسب

مهارت در شکل (آ)، در شکل (۵-۱۴) (ب)، این مقایسه در مقیاس لگاریتمی صورت گرفته است. در این نمودار عملکرد تا حدی بهتر روش پیشنهادی با معیار ناهمواری دیده می‌شود، که می‌توان آن را به حذف مهارت‌های غیرمفید و اضافه کردن یک مهارت جدید مفید نسبت داد.



(آ) نمودار میانگین تعداد گام تا هدف در محیط سه اتاقه



(ب) نمودار با مقیاس لگاریتمی برای نمودار (آ)

شکل (۵-۱۴): مقایسه در محیط سه اتاقه با پارامترهای

$$Rf: n_t = 200, n_k = 10, \alpha = 0.98, \rho = 0.98, \tau_v = 1.01, \tau_d = 1.5$$

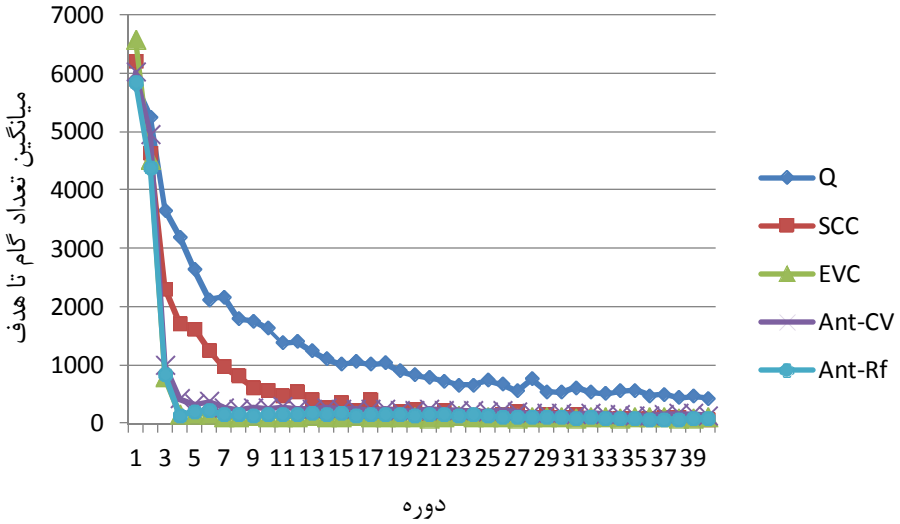
$$CV: n_t = 2000, n_k = 15, \alpha = 0.9, \rho = 0.98, \tau_v = 1.01, \tau_d = 1.5$$

۵-۳-۲ محیط شش اتاقه

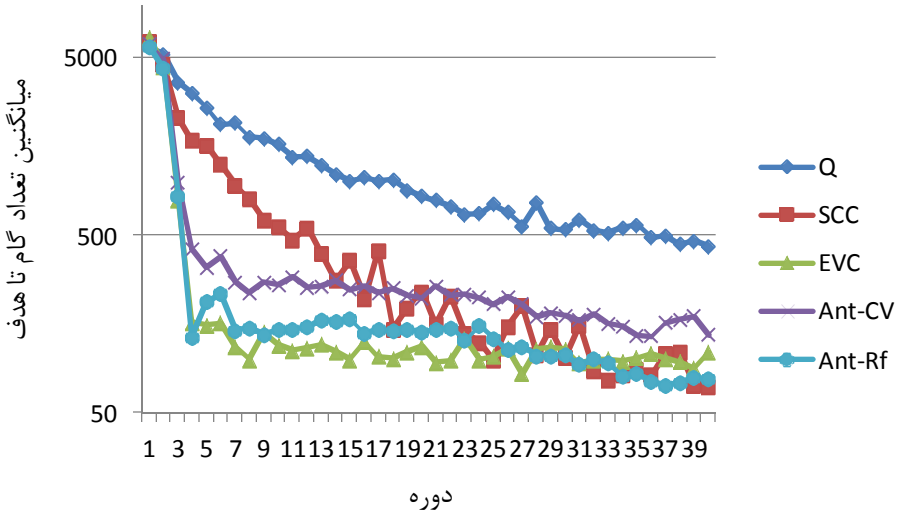
شکل (۵-۱۵) (آ) مقایسه‌ای از عملکرد پنج روش مذکور در محیط شش اتاقه ارائه می‌دهد. برای روش مولفه‌های قویاً همبند مقدار پارامتر t_t برابر ۲۲ است. در نمودار شکل (۵-۱۵) (ب) نیز برای تمایز بیش‌تر روش‌ها، از مقیاس لگاریتمی استفاده شده است. در این نمودار، عملکرد روش پیشنهادی با هر دو معیار ضریب تغییرات و ناهمواری، در مقایسه‌ی با روش مرکزیت بردار ویژه، مختصراً افت کرده است، گرچه همچنان بسیار سریع‌تر از روش یادگیری Q می‌باشد. به نظر می‌رسد در شرایطی که زیرهدف‌ها خاصیت گلوگاهی کم‌تری داشته باشند، یال‌های مجاور با آن در مقایسه با دیگر یال‌ها، به علت وجود گزینه‌های دیگر ممکن است تمایز کم‌تری داشته باشند و به این ترتیب روش پیشنهادی، به طور میانگین در تعداد کم‌تری از آزمایش‌ها، همه‌ی این زیرهدف‌ها را به درستی پیدا می‌کند.

۵-۳-۳ محیط تاکسی

در فصل قبل، پیش‌بینی کردیم که در محیط‌هایی که کنش‌ها لزوماً بازگشت‌پذیر نباشند، روش‌هایی که از گراف‌های بدون جهت برای مدل کردن محیط استفاده می‌کنند، احتمالاً موفقیت کم‌تری کسب می‌نمایند. این موضوع با توجه به شکل (۵-۳) از گراف گذر محیط تاکسی، برای این محیط صادق می‌باشد. این مطلب در پیشی گرفتن کاملاً محسوس روش پیشنهادی از روش مرکزیت بردار ویژه، علی‌رغم عملکرد بهتر آن روش در محیط شش اتاقه، در شکل (۵-۱۶) مشاهده می‌شود. بخشی از افت شدید کارایی روش مولفه‌های همبندی در این محیط را می‌توان به مسائل مرتبط با تنظیم پارامتر نسبت داد.



(آ) نمودار میانگین تعداد گام تا هدف در محیط شش اتاقه

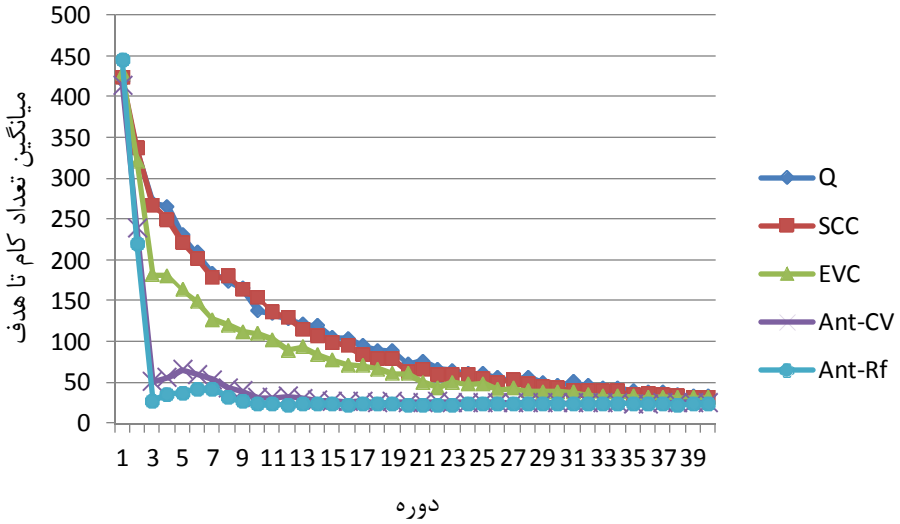


(ب) نمودار با مقیاس لگاریتمی برای نمودار (آ)

شکل (۵-۱۵): مقایسه در محیط شش اتاقه با پارامترهای

$Rf: n_t = 200, n_k = 10, \alpha = 0.98, \rho = 0.98, \tau_v = 1.15, \tau_d = 1.5$

$n_t = 2000, n_k = 15, \alpha = 0.9, \rho = 0.98, \tau_v = 1.15, \tau_d = 1.5$



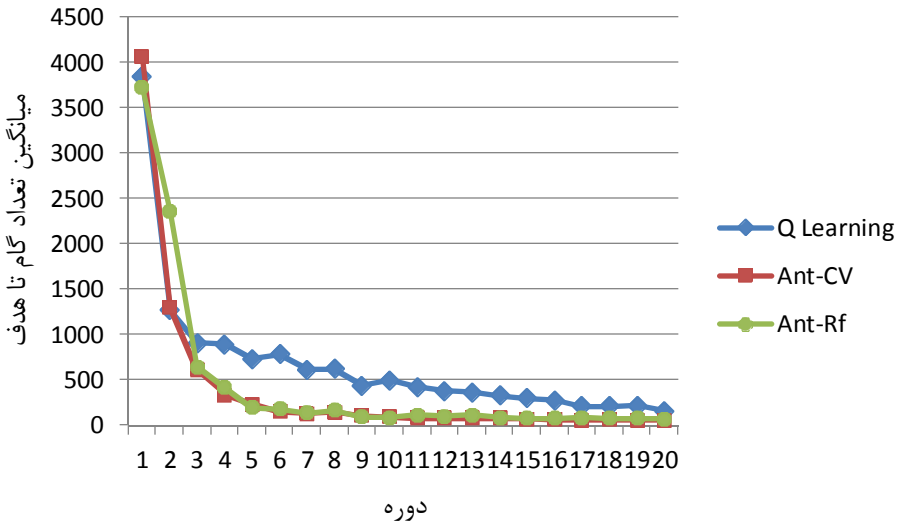
شکل (۵-۱۶): مقایسه‌ی روش‌ها در محیط تاکسی با پارامترهای

$$Rf: n_t = 10, n_k = 25, \alpha = 0.9, \rho = 0.98, \tau_v = 1.01, \tau_d = 1.5$$

$$CV: n_t = 2000, n_k = 15, \alpha = 0.75, \rho = 0.94, \tau_v = 1.01, \tau_d = 1.5$$

۵-۳-۴ محیط اتاق بازی

محیط اتاق بازی نیز مانند محیط تاکسی، محیطی با کنش‌های غیرقابل بازگشت است. به عنوان مثال، تاثیر زدن ضربه به توپ که باعث حرکت توپ، نواخته شدن زنگ و احتمالاً به صدا در آمدن میمون می‌شود، با یک کنش قابل بازگشت نیست. به همین دلیل انتظار می‌رود در این محیط نیز، روش پیشنهادی نتیجه‌ی بهتری کسب کند. متاسفانه به دلیل عدم امکان اجرای روش مرکزیت بردار ویژه و روش مولفه‌های قویاً همبند در این محیط، شکل (۵-۱۷) حاوی مقایسه بین روش‌های یادگیری Q و روش پیشنهادی می‌باشد.

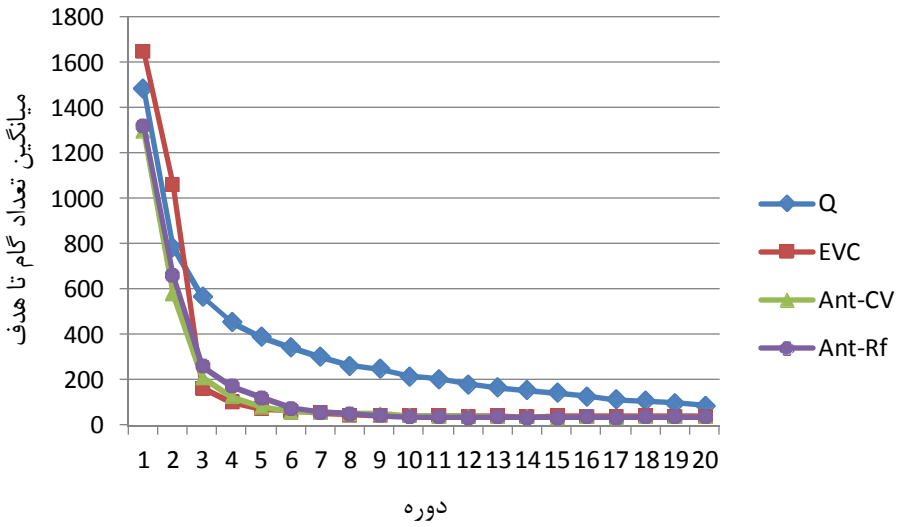


شکل (۵-۱۷): مقایسه‌ی روش پیشنهادی و یادگیری Q در محیط اتاق بازی با پارامترهای

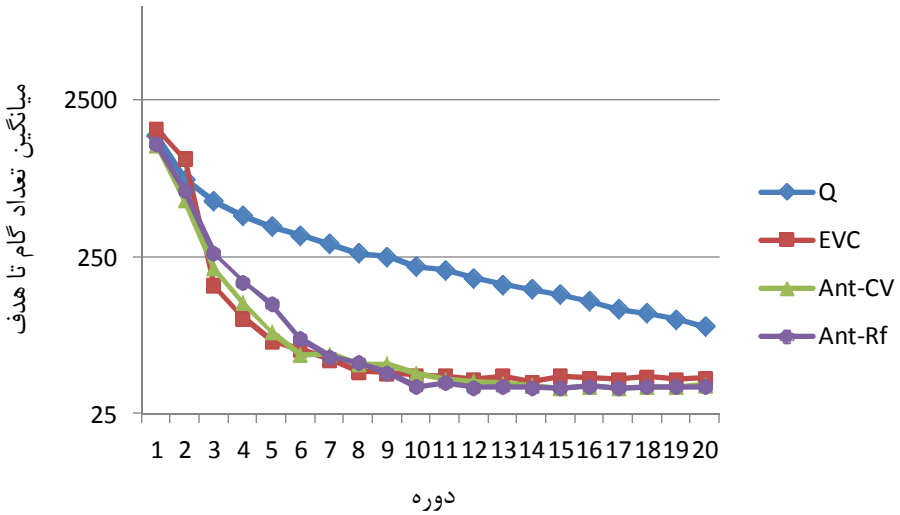
$$Rf: n_t = 200, n_k = 10, \alpha = 0.9, \rho = 0.98, \tau_v = 2.0, \tau_d = 1.5$$
$$CV: n_t = 2000, n_k = 10, \alpha = 0.9, \rho = 0.95, \tau_v = 2.0, \tau_d = 1.5$$

۵-۳-۵ محیط برج‌های هانوی

پیش‌تر در بخش ۵-۲-۴، هنگام معرفی این محیط گفتیم که در آن مهارت‌هایی به صورت سلسله مراتبی قابل تعریف است. بسته به مقدار پارامترها، به ویژه پارامترهای τ_v و τ_d، می‌توان سطوح مختلفی از زیرهدف‌ها را به‌دست آورد. در محیط برج‌های هانوی، تمام کنش‌ها بازگشت‌پذیر هستند. به همین دلیل روش پیشنهادی مزیت قابل توجهی در برابر روش مرکزیت بردار ویژه نخواهد داشت. نمودار شکل (۵-۱۸) مقایسه‌ای بر این روش‌ها را نشان می‌دهد. همان‌طور که در نمودار دیده می‌شود، در این محیط روش پیشنهادی با دو معیار ناهمواری و ضریب تغییرات عملکرد تقریباً مشابهی دارند.



(آ) نمودار میانگین تعداد گام تا هدف در محیط برج‌های هانوی



(ب) نمودار با مقیاس لگاریتمی نمودار (آ)

شکل (۵-۱۸): مقایسه در محیط اتاق برج‌های هانوی با پارامترهای

$Rf: n_t = 300, n_k = 10, \alpha = 0.95, \rho = 0.99, \tau_v = 1.9, \tau_d = 5$

$CV: n_t = 2000, n_k = 10, \alpha = 0.9, \rho = 0.9, \tau_v = 1.5, \tau_d = 5$

۵-۴ جمع‌بندی

در فصل پنجم به شرح و بسط محیط‌های مورد آزمایش پرداخته شد. سپس مقادیر بهینه برای پارامترهای برنامه به‌دست آمده و ارائه شد. در نهایت مقایسه‌ای از الگوریتم‌های یادگیری Q، مولفه‌های قویاً همبند، مرکزیت بردار ویژه و روش پیشنهادی با معیار ضریب تغییرات فرومون و ناهمواری مقادیر فرومون انجام گشت.

مطابق آن‌چه از نتایج آزمایش‌های عملی به‌دست آمد، روش پیشنهادی در محیط‌های سه اتاقه و محیط تاکسی عملکرد کاملاً بهتری به نسبت روش‌های دیگر داشت و در دیگر محیط‌ها نیز نتایج بسیار نزدیکی به دیگر روش-های مبتنی بر مهارت، کسب نمود. دلیل نتایج مناسب روش پیشنهادی را می‌توان به دو مزیت اصلی نسبت داد. مزیت نخست استفاده از وزن و جهت یال‌ها است که حاوی تمام اطلاعات تعاملی عامل با محیط می‌باشد. دومین مزیت روش پیشنهادی، کشف و استفاده از مهارت‌های مفید و اجتناب از مهارت‌های غیرضروری است.

در فصل آینده به یک جمع‌بندی کلی از این پایان نامه و کارهای آینده پرداخته خواهد شد.

فصل ششم
جمع‌بندی و کارهای آینده

در این پایان‌نامه روشی برای انجام یادگیری تقویتی سلسله مراتبی ارائه شد. در روش پیشنهادی از ویژگی جدیدی از زیرهدف‌ها برای اکتشاف آن‌ها استفاده شد: حالت‌های زیرهدف در فضای حالت حضور محوری پایدارتری در گذرهای موفق دارند.

عمده‌ی روش‌های قبلی با استفاده از خوشه‌بندی گراف تمام مهارت‌های مفید و غیرمفید را می‌سازند، سپس با استفاده از برخی از روش‌های هرس مهارت‌ها سعی بر حذف مهارت‌های غیر مفید دارند که این فرایند ممکن است زمان‌بر و غیر دقیق باشد. مهم‌ترین دست‌آوردهای این پژوهش را می‌توان، ارائه روشی جدید برای کشف زیرهدف‌های مفید محیط دانست. این نکته به خصوص در محیط‌های طبیعی بزرگ با هزاران مهارت نامربوط ممکن، که انتخاب هر کدام از آن‌ها می‌تواند عامل را از هدف دور سازد، اهمیت فوق‌العاده‌ای می‌یابد.

نتایج آزمایش‌های عملی نشان داد که معیار ناهمواری در مقایسه با معیار ضریب تغییرات معمولاً نتایج بهتری کسب می‌کند، ضمن این‌که با مقایسه‌ی نمودارهای مربوط به حساسیت‌سنجی روش به پارامتر n_t دیده می‌شود روش پیشنهادی با معیار ناهمواری در مقادیر کم‌تر n_t رفتار بهتری داشته در صورتی که این روش با استفاده از معیار ضریب تغییرات نیاز به مقادیر بیشتری از n_t دارد که این نشان می‌دهد که ناهمواری با در نظر گرفتن زمان اجرا نیز معیار مناسب‌تری است. این برتری را همان‌طور که در فصل چهارم توضیح داده شد، می‌توان به در

نظر گرفتن توالی مقادیر فرومون در طول زمان، نسبت داد. همچنین با مقایسه‌ی دیگر روش‌ها با روش ناهمواری فرومون، مشاهده شد که این روش علاوه بر کسب نتایج قابل قبول در تمام محیط‌های مورد آزمایش، در برخی از محیط‌ها مانند محیط تاکسی و محیط سه اتاقه، نتایجی به مراتب بهتر از روش‌های دیگر حاصل خواهد کرد.

علی‌رغم بهبود کارایی در روش پیشنهادی به نسبت برخی از روش‌های دیگر، هنوز مسائل باز زیادی وجود دارد که یادگیری تقویتی سلسله مراتبی با آن‌ها روبروست. بسیاری از محیط‌ها را می‌توان یافت که دارای اشتراکات فراوان هستند و به همین دلیل، می‌توان مهارت‌های آموخته شده در یکی از محیط‌ها را به دیگری انتقال داد. به عنوان مثال انسان‌ها، استفاده از راه پله یا آسانسور را، مستقل از ساختمانی که در آن قرار دارند، به عنوان مهارتی از پیش آموخته بکار می‌برند. این که چطور می‌توان این مهارت‌های عمومی را شناسایی کرد و در محیط‌های مشابه بکار برد که به آن مسئله‌ی انتقال مهارت در یادگیری مهارت گفته می‌شود، از جمله مقولات مهم و راه‌گشای یادگیری تقویتی سلسله مراتبی می‌تواند باشد که در این پژوهش دست نخورده باقی ماند.

با توجه به استفاده از روش بهینه‌سازی کلونی مورچه و تعداد زیاد پارامترهای مرتبط با آن، به‌دست آوردن مقدار بهینه برای هر کدام از پارامترها به صورت خودکار می‌تواند تاثیر زیادی در بهینگی اجرای الگوریتم داشته باشد. در این پژوهش حساسیت روش پیشنهادی به این پارامترهای سنجیده و گزارش شده است، اما مسئله‌ی مهم پیش رو، به‌دست آوردن مقدار مناسب پارامترهاست که در این پایان‌نامه روش ساختار یافته‌ای برای آن ارائه نشده است.

همان‌طور که گفتیم در این پژوهش سعی شده تا جای ممکن، مهارت‌های مفید ساخته و استفاده شوند. اما این مهم به بهای ایجاد وابستگی فرایند یادگیری به محیط‌هایی با حالت‌های آغاز و پایان مشخص و یا وابستگی به وظایف در محیط‌های کلی، به‌دست آمده است. بنابراین به نظر می‌رسد یادگیری تقویتی سلسله مراتبی نیازمند معیاری برای تشخیص مفید بودن مهارت‌ها می‌باشد و رسیدن به چنین معیاری را می‌توان گام بزرگی در جهت پیش‌رفت در این زمینه دانست.

از جمله کارهای دیگری که در راستای ایده‌ی اصلی این پژوهش، می‌توان انجام داد، استفاده از ایده‌ی مسیرهای تصادفی برای خوشه‌بندی گراف است. این الگوریتم خوشه‌بندی در ادامه می‌تواند به عنوان یک روش جدید برای کشف زیرهدف و ساخت مهارت به‌کار برده شود.

کتاب نامه

[۱] کاظمی‌تبار، سیدجلال، **کسب خودکار مهارت در یادگیری تقویتی**، پایان‌نامه کارشناسی‌ارشد، دانشگاه صنعتی شریف، تهران، دی‌ماه ۱۳۸۷

[۲] تقی‌زاده، نسرین، **کسب خودمختار مهارت در یادگیری تقویتی مبتنی بر خوشه‌بندی گراف**، پایان‌نامه کارشناسی ارشد، دانشگاه صنعتی شریف، تهران، آذرماه ۱۳۹۰

[3] S. Fortunato, "Community detection in graphs," Physics Reports, vol. 485, pp. 75-174, 2010.

[4] R. Sutton and A. Barto, Reinforcement Learning: An Introduction, Cambridge: MIT Press, 1998.

[5] T. Mitchel, Machine learning, Mac Graw Hill, 1997.

[6] M. Stolle and D. Precup, "Learning options in reinforcement learning," in *5th International Symposium on Abstraction, Reformulation and Approximation*, London, 2002.

[7] N. Mehta, S. Ray, P. Tadepalli and T. Dietterich, "Automatic discovery and transfer of MAXQ hierarchies," in *Twenty Fifth International Conference on Machine Learning*, 2008.

[8] R. Parr and S. Russell, "Reinforcement Learning with Hierarchies of Machines," in *NIPS*, 1997.

[9] O. Simsek and A. Barto, "Using relative novelty to identify useful temporal abstraction in reinforcement learning," in *twenty-first international conference on Machine learning*, 2004.

[10] O. Simsek and A. Barto, "Identifying useful subgoals in reinforcement learning by local graph partitioning," in *22nd international conference on machine learning*, 2005.

[11] I. Menache, S. Mannor and N. Shimkin, "Q-cut dynamic discovery of sub-goals in reinforcement learning," in *ECML*, 2002.

[12] O. Simsek and A. Barto, "Skill characterization based on betweenness," in *Twenty-Second Annual Conference on Neural Information Processing Systems*, 2009.

[13] S. J. Kazemitabar and H. Beigy, "Automatic Discovery of Subgoals in Reinforcement Learning Using Strongly Connected Components," in *ICONIP*, 2008.

[14] A. Pothen, "Graph partitioning algorithms with applications to scientific computing," Norfolk, 1997.

[15] B. W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *The Bell system technical journal,* vol. 49, pp. 291-307, 1970.

[16] T. Hastie, R. Tibshirani and J. H. Friedman, The Elements of Statistical Learning, Berlin: Springer, 2001.

[17] U. Von Luxburg, "A Tutorial on Spectral Clustering," *Statistics and Computing,* vol. 17(4), pp. 395-416, 2007.

[18] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *National Academy of Sciences,* vol. 99, pp. 7821-7826, 2002.

[19] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences,* vol. 103, pp. 8577-8582, 2006.

[20] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E,* vol. 69, 2004.
[21] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E,* vol. 69, 2004.
[22] P. K. Ahuja, T. L. Magnati and J. B. Orlin, Network Flows Theory, Algorithms and Applications, Prentice Hall Press, 1993.
[23] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum-flow problem," *Journal of ACM,* vol. 35(4), p. 921–940, 1988.
[24] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of Mathematical Sociology,* vol. 25(2), p. 163–177, 2001.
[25] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms, The MIT Press, 2009.
[26] S. J. Kazemitabar and H. Beigy, "Using Strongly Connected Components as a Basis for Autonomous Skill Acquisition in Reinforcement Learning," *Lecture Notes in Computer Sciences,* vol. 5551, pp. 794-801, 2009.
[27] P. Bonacich, "Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology," *The Journal of Mathematical Sociology,* vol. 2, p. 113–120, 1972.
[28] G. Canright and K. Engo-Monsen, "Spreading on networks: a topographic view," in *European Conference on Complex Systems*, 2005.
[29] G. H. Golub and C. F. Van Loan, Matrix computations, Baltimore: Johns Hopkins University Press, 1996.
[30] L. M. Gambardella, E. D. Taillard and M. Dorigo, "Ant Colonies for the QAP," *Journal of the Operational Research Sociery,* vol. 50, p. 167–176, 1999.
[31] S. Goss, S. Aron, J. L. Deneubourg and J. M. Pasteels, "Self-organized shortcuts in the Argentine ant," *Naturwissenschaften,* vol. 76, pp. 579-581, 1989.
[32] A. P. Engelbrecht, Computational intelligence: an introduction, John Wiley & Son,, 2007.
[33] V. Maniezzo and A. Colorni, "The Ant System Applied to the Quadratic Assignment Problem," *IEEE Transactions on Knowledge and Data Engineering,* vol. 11, p. 769–778, 199.
[34] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of Artificial Intelligence Research,* vol. 13, pp. 227-303, 199.
[35] S. P. Singh, A. Barto and N. Chentanez, "Intrinsically Motivated Reinforcement Learning," in *Advances in Neural Information Processing Systems 17*, 2005.

واژه‌نامه‌ی انگلیسی به فارسی

Action	کنش
Action-Value Function	تابع ارزش-کنش
Additive Function	تابع جمع‌کننده
Adjacency List	لیست مجاورت
Adjacency Matrix	ماتریس مجاورت
Agglomerative Algorithm	الگوریتم تراکمی
Ant Colony Optimization	بهینه‌سازی کلونی مورچه
Average Linkage Clustering	خوشه‌بندی اتصال میانگین
Bayes Decision Theory	نظریه‌ی تصمیم‌گیری بیز
Betweenness	بینابینی
Breadth First search	جستجوی سطح اول
Centrality	مرکزیت
Clique	خوشه
Coeficient of Variation	ضریب تغییرات
Community	انجمن
Comparision of Internal Versus External Cohesion	مقایسه‌ی پیوستگی درونی و خارجی
Complete Linkage Clustering	خوشه‌بندی اتصال کامل
Complete Mutality	دوبه‌دویی کامل
Continous Task	وظیفه‌ی پیوسته
Cut Size	اندازه‌ی برش
Delayed Reward	پاداش تاخیری
Depth First Search	جستجوی عمق اول
Discounted Return	درآمد تخفیف خورده
Discounting Rate	نرخ تخفیف

Divide and Conquer	تقسیم و حل
Divisive Algorithms	الگوریتم های تقسیمی
Dynamic Programming	برنامه‌نویسی پویا
Edge Betweenness	بینابینی یالی
EigenVector Centrality	مرکزیت بردار ویژه
Episodic Task	وظیفه‌ی دوره ای
Experience Replay	بازبینی تجربه
Exploitation	انتفاع
Exploratio	کاوش
Finishing Time	زمان پایان
Finite Discrete Time Markov Process	فرایند تصمیم‌گیری متناهی و زمان گسسته مارکوف
Graph Partitioning	افراز گراف
Hierarchical Clustering	خوشه‌بندی سلسله مراتبی
Incremental	افزایشی
k-Center	k-مرکز
k-Means	k-میانگین
k-Median	k-میانه
Learning Rate	نرخ یادگیری
Local Graph Partitioning	افراز گراف محلی
Macro Action	فراکنش
Markov Decision Process	فرایند تصمیم‌گیری مارکوف
Markov Property	خاصیت مارکوف
Min Cut-Max Flow Problem	مسئله‌ی برش کمینه-شار بیشینه
Minimum Bisection	دونیم‌کردن کمینه
Minimum k-Clustering	k-خوشه‌بندی کمینه
Modularity	پیمانه‌ای بودن

Modularity Based Methods	روش‌های مبتنی بر پیمانگی
Novelty	تازگی
NP-Complete	NP-کامل
NP-Hard	NP-سخت
Null Model	مدل تهی
Option Framework	چارچوب گزینه
Overlap	همپوشانی
Partition	افراز
Partitional Clustering	خوشه‌بندی افرازی
Pheromone	فرومون
Policy	سیاست
Posterior Knowledge	دانش پسین
Prior Knowledge	دانش پیشین
Prior Probability	احتمال پیشین
Quality Function	تابع کیفیت
Random Graph	گراف تصادفی
Reachability	قابلیت دسترسی
Reinforcement Learning	یادگیری تقویتی
Relative Novelty	تازگی نسبی
Scalable	مقیاس‌پذیر
Semi-Supervised Learning	یادگیری نیمه‌نظارت شده
Simple Ant Colony Optimization	بهینه‌سازی کلونی مورچه ساده
Single Linkage Clustering	خوشه‌بندی اتصال تک
Social Network	شبکه‌ی اجتماعی
Spectral Clustering	خوشه‌بندی طیفی
State	حالت

State Abstraction	انتزاع حالت
State-Value Function	تابع ارزش-حالت
Strong Community	انجمن قوی
Sub-goal	زیرهدف
Supervised Learning	یادگیری نظارت شده
Tabu List	لیست ممنوعه
Temporal Abstraction	انتزاع زمانی
Temporal Difference Methods	روش‌های اختلاف زمانی
Unsupervised Learning	یادگیری بدون نظارت
Vertex Degree	درجه‌ی راس

واژه‌نامه‌ی فارسی به انگلیسی

k-خوشه بندی کمینه	Minimum k-Clustering
k-مرکز	k-Center
k-میانگین	k-Means
k-میانه	k-Median
NP-کامل	NP-Complete
احتمال پیشین	Prior Probability
افراز	Partition
افراز گراف	Graph Partitioning
افراز گراف محلی	Local Graph Partitioning
افزایشی	Incremental
الگوریتم تراکمی	Agglomerative Algorithm
الگوریتم‌های تقسیمی	Divisive Algorithms
انتزاع حالت	State Abstraction
انتزاع زمانی	Temporal Abstraction
انتفاع	Exploitation
انجمن	Community
انجمن قوی	Strong Community
اندازه‌ی برش	Cut Size
بازبینی تجربه	Experience Replay
برنامه‌نویسی پویا	Dynamic Programming
بهینه‌سازی کلونی مورچه	Ant Colony Optimization
بهینه‌سازی کلونی مورچه ساده	Simple Ant Colony Optimization
بینابینی	Betweenness
بینابینی یالی	Edge Betweenness
پاداش تاخیری	Delayed Reward
پیمانه‌ای بودن	Modularity

تابع ارزش-حالت	State-Value Function
تابع ارزش-کنش	Action-Value Function
تابع جمع کننده	Additive Function
تابع کیفیت	Quality Function
تازگی	Novelty
تازگی نسبی	Relative Novelty
تقسیم و حل	Divide and Conquer
جستجوی سطح اول	Breadth First search
جستجوی عمق اول	Depth First Search
چارچوب گزینه	Option Framework
حالت	State
خاصیت مارکوف	Markov Property
خوشه	Clique
خوشه‌بندی اتصال تک	Single Linkage Clustering
خوشه‌بندی اتصال کامل	Complete Linkage Clustering
خوشه‌بندی اتصال میانگین	Average Linkage Clustering
خوشه‌بندی افرازی	Partitional Clustering
خوشه‌بندی سلسله‌مراتبی	Hierarchical Clustering
خوشه‌بندی طیفی	Spectral Clustering
دانش پسین	Posterior Knowledge
دانش پیشین	Prior Knowledge
درآمد تخفیف خورده	Discounted Return
درجه‌ی راس	Vertex Degree
دوبه‌دویی کامل	Complete Mutality
دونیم‌کردن کمینه	Minimum Bisection
روش‌های اختلاف زمانی	Temporal Difference Method
روش‌های مبتنی بر پیمانگی	Modularity Based Methods
زمان پایان	Finishing Time

زیرهدف	Sub-goal
NP-سخت	NP-Hard
سیاست	Policy
شبکه‌ی اجتماعی	Social Network
ضریب تغییرات	Coeficient of Variation
فراکنش	Macro Action
فرایند تصمیم‌گیری مارکوف	Markov Decision Process
فرایند تصمیم‌گیری متناهی و زمان گسسته مارکوف	Finite Discrete Time Markov Process
فرومون	Pheromone
قابلیت دسترسی	Reachability
کاوش	Exploratio
کنش	Action
گراف تصادفی	Random Graph
لیست مجاورت	Adjacency List
لیست ممنوعه	Tabu List
ماتریس مجاورت	Adjacency Matrix
مدل تهی	Null Model
مرکزیت	Centrality
مرکزیت بردار ویژه	EigenVector Centrality
مسئله‌ی برش کمینه-شار بیشینه	Min Cut-Max Flow Problem
مقایسه‌ی پیوستگی درونی و خارجی	Comparision of Internal Versus External Cohesion
مقیاس‌پذیر	Scalable
نرخ تخفیف	Discounting Rate
نرخ یادگیری	Learning Rate
نظریه‌ی تصمیم‌گیری بیز	Bayes Decision Theory
همپوشانی	Overlap
وظیفه‌ی پیوسته	Continous Task
وظیفه‌ی دوره ای	Episodic Task

یادگیری بدون‌نظارت	Unsupervised Learning
یادگیری تقویتی	Reinforcement Learning
یادگیری نیمه‌نظارت شده	Semi-Supervised Learning
یادگیری نظارت شده	Supervised Learning

Abstract

Reinforcement learning is a learning method that uses reward and penalty feedbacks, having no information about the right action. In this method, agent gets the state of environment and selects an action among its permissible set of actions, regarding its policy and the given state. Environment, expresses an evaluation, in form of a reinforcement signal and a change in state, as a response for agent's action. Afterward, the agent updates its policy considering received signal in order to maximize its long term reward. Reinforcement learning rapidly converges to the optimal solution, only if there are few states and actions, but there are lots of domains that consist of too many states and actions, which cause very slow convergence.

Using temporal abstraction can address this problem for large scale environments, and make the convergence much faster, compared to conventional methods. Temporal abstraction can be used through acquisition and utilization of skills. Briefly explained, skill can be defined as a sequence of primitive actions that can be applied to reach a suitable state in the environment. From another perspective, if the environment state transition is modeled as a graph, then the boundary points of communities of this graph may be regarded as sub-goals which the agent needs to pass over them, in order to reach the goal state.

In this thesis, an algorithm is presented which makes use of ant colony optimization methods to identify sub-goal states. Initially several paths from initial to goal state are generated by ants and then the alternation of pheromone deposited by ants on edges of shortest path is analyzed. Then edges with different distribution of pheromone over time are separated and known as bottleneck edges. Next, communities consisting of fragments of shortest path are detected. Finally, useful skills are learned on each detected community using option framework. To evaluate the results of the proposed method, its performance is compared with some other skill learning methods on 4 standard benchmarks, including Grid world, Taxi, Playroom and Hanoi environments. Results acquired from experimental results, shows improvements of performance in several environments.

Keywords: Reinforcement learning, skill acquisition, sub-goal detection, option framework, ant colony optimization algorithm.

Sharif University of Technology
Computer Engineering Department

M. Sc. Thesis

Automatic Skill Learning Using Community Detection Approach

By:
Mohsen Ghafoorian

Supervisor:
Dr. Hamid Beigy

October 2012