به نام پروردگار دانایی

نام و نام خانوادگی: امیرمسعود فرهمند
عنوان پایان‌نامه: یادگیری و تکامل در معماری‌های رفتارگرای سلسله‌مراتبی
استاد راهنما: دکتر مجید نیلی احمدآبادی
استادان مشاور: دکتر کارو لوکس – دکتر بابک اعرابی

اینجانب امیرمسعود فرهمند نویسنده پایان‌نامه کارشناسی ارشد در گروه مهندسی برق و کامپیوتر گرایش کنترل بدین‌وسیله اعلام می‌نمایم که توسط اساتید راهنما و سمینار در جریان قوانین کپی‌رایت و روش درست اقتباس و نقل مطالب از سایر مراجع و مواخذ قرار گرفته‌ام و متعهد به حفظ امانت‌داری و قدردانی از زحمات سایر محققین و نویسندگان می‌باشم. بدین‌وسیله اعلام می‌نمایم که مسوولیت کلیه مطالب درج شده با اینجانت می‌باشد و در صورت استفاده از اشکال، جداول و مطالب از سایر منابع بلافاصله مرجع آن ذکر شده است و سایر مطالب از کار تحقیقی اینجانب استخراج شده است و امانت‌داری را به صورت کامل رعایت نموده‌ام. در صورتی که خلاف این مطلب ثابت شود، مسوولیت کلیه عواقب قانونی با شخص اینجانب است.

امیرمسعود فرهمند

دانشگاه تهران

دانشکدهٔ فنی

گروه مهندسی برق و کامپیوتر

عنوان:

یادگیری و تکامل در معماری‌های رفتارگرای سلسله‌مراتبی

نگارش:

امیرمسعود فرهمند

استاد راهنما:

دکتر مجید نیلی احمدآبادی

استادان مشاور:

دکتر کارو لوکس

دکتر بابک اعرابی

پایان‌نامه برای دریافت درجهٔ کارشناسی ارشد در رشتهٔ
مهندسی برق - گرایش کنترل

امرداد ۱۳۸۴

دانشگاه تهران
دانشکده فنی
گروه مهندسی برق و کامپیوتر

عنوان:

یادگیری و تکامل در معماری‌های رفتارگرای سلسله‌مراتبی

نگارش:
امیرمسعود فرهمند

پایان‌نامه برای دریافت درجهٔ کارشناسی ارشد در رشتهٔ
مهندسی برق - گرایش کنترل

از این پایان‌نامه در تاریخ ۱۳۸۴/۵/۵ در مقابل هیات داوران دفاع به عمل آمد و مورد تصویب قرار گرفت.

معاونت تحصیلات تکمیلی دانشکده فنی	دکتر جواد فیض
مدیر گروه آموزشی	دکتر پرویزجبه‌دار مارالانی
سرپرست تحصیلات تکمیلی گروه	دکتر سید مهدی فخرایی
استاد راهنما	دکتر مجید نیلی احمدآبادی
استاد مشاور ۱	دکتر کارو لوکس
استاد مشاور ۲	دکتر بابک نجار اعرابی
عضو هیات داوران	دکتر بهزاد مشیری
عضو هیات داوران	دکتر علیرضا فاتحی

چکیده

هدف اصلی‌ی این پژوهش، خودکارسازی‌ی فرآیند طراحی‌ی عامل‌های هوش‌مند رفتارگرای سلسله‌مراتبی است. برای انجام این‌کار فرض می‌شود که سیگنال تقویت‌ای توسط طراح مشخص شده که بیان‌گر عمل‌کرد عامل است. روش‌های پیش‌نهادی در این گزارش سعی در بهبود آن معیار دارند. در این پایان‌نامه مساله‌ی طراحی عامل‌هایی با معماری رفتارگرای سلسله‌مراتبی به دو زیرمساله‌ی طراحی‌ی رفتارهای معماری و طراحی‌ی ساختار معماری تقسیم شده است که حلِ هم‌زمان این دو منجر به حلِ مساله‌ی طراحی‌ی معماری می‌شود. در این پژوهش دو روی‌کرد مبتنی بر یادگیری و مبتنی بر تکامل به عنوان روش‌های خودکارسازی‌ی فرآیند طراحی معرفی شده است.

با استفاده از ایده‌های مبتنی بر یادگیری، هر دو مساله‌ی یافتن ساختار و رفتار مطلوب مطرح شده و سپس پاسخ مناسب برای حلِ آن‌ها ارایه می‌شود. برای حلِ مساله‌ی یادگیری ساختار، دو بازنمایی مختلف دانش موسوم به بازنمایی مرتبه صفر و مرتبه یک معرفی می‌شود که با تجزیه‌ی ارزش کلِ عامل به اجزای ساده‌تر، مسایل دشواری چون تقسیم امتیاز و به‌روزرسانی‌ی دانش را قابل حل می‌کند. ایده‌های یادگیری برای ایجاد نگاشت مناسب هر رفتار نیز به کار رفته است و با معرفی‌ی عمل‌ای مجازی، اجازه‌ی هم‌کاری‌ی بین رفتارها داده شده است.

تکامل هم‌کارانه‌ی رفتارها نیز به عنوان روش‌ای برای تولید رفتارهای جدید استفاده شده است. در این پژوهش برخلاف اکثر پژوهش‌های پیشین، رفتارها به طور مجزا تکامل می‌یابند و سپس با استفاده از یادگیری ساختار به نحوی در معماری قرار می‌گیرند که به بیش‌ترین بازده ممکن منجر شوند.

روش‌های الهام‌گرفته از فرهنگ به نام الگوریتم‌های ممتیک نیز معرفی شده‌اند که باعث بالا رفتن بازده استفاده‌ی هم‌زمان از یادگیری و تکامل می‌شود.

روش‌های پیش‌نهادی بر دو بستر بلندکردن جسم توسط چند روبات و مساله‌ای مجرد آزموده شده‌اند و نتایج قابل رقابت با طراحی‌های انسان به دست آمده است.

در این پژوهش علاوه بر نتایج اصلی مربوط به طراحی‌ی خودکار معماری‌ی رفتارگرای سلسله‌مراتبی، نتایج فرعی‌ای چون تحلیل احتمالاتی‌ی معماری‌ی رفتارگرای سلسله‌مراتبی‌ی به‌کار رفته در این گزارش و هم‌چنین تاثیر عدم قطعیت سیگنال تقویت بر تابع ارزش و تابع سیاست عامل نیز بررسی شده است.

تشکر

این پایان‌نامه نوشته نمی‌شد مگر با کمک استادان و دوستان عزیزم که در این مدت مرا همه‌گونه یاری کردند.

تجربه‌ی کار مشترک با مجید نیلی یکی از به‌ترین تجربه‌های علمی‌ی من بوده است. او نه تنها افق‌های تازه‌ای از هوش‌مصنوعی را به روی‌ام گشود، بلکه شیوه‌ی کار آکادمیک را نیز به من یاد داد. او مرا در انتخاب مسیر پژوهش آزاد گذاشت و در عوض به عنوان نقادی بسیار خوب در مورد ایده‌های‌ام – چه آن‌هایی که جا افتاده بودند و چه آن‌هایی که تنها جرقه‌ای بوده‌اند و من یک دفتر پر از آن‌ها ثبت کرده‌ام– نظر داد. ما با وجود آن‌که در بعضی از زمینه‌های علمی با هم اختلاف نظر داشتیم اما به خوبی در بسیاری از موارد به تفاهم رسیدیم و با کم‌ترین تقابل‌ای پروژه را به پیش بردیم. علاوه بر این ساعت‌ها وقتی که با هم صرف تصحیح مقالات کردیم جزو تجربه‌های نادری بود که تنها از یک استاد راه‌نمای ایده‌آل بر می‌آید.

کارو لوکس یکی از به‌ترین استادانی بوده است که تاکنون داشته‌ام. او مخزن دانش‌ای است که هر کس به اندازه‌ی توانِ خود از آن برداشت می‌کند. هر بار که با او صحبت می‌کردم یا سر کلاس‌های‌اش می‌نشستم، انبوه‌ای از ایده‌های جدید به ذهن‌ام خطور می‌کرد. او بدون شک یکی از شکل‌دهنده‌های اصلی‌ی اندک دانش‌ای است که من در این مدت کسب کرده‌ام.

بابک اعرابی، دیگر استاد مشاورم، با اشاره‌های مفیدش درک مرا نسبت به آن‌چه انجام می‌دهم و ارتباطش با دیگر حوزه‌های مربوط –مانند شناسایی الگو و یا کنترل– بالا برد. علاوه بر این، برخورد مثبت و امیدبخش او همیشه برای‌ام ارزش‌مند بوده و مرا به ادامه‌ی کار –چه به پروژه‌ام مربوط بوده باشد و چه به کارهای دیگر– امیدوار می‌کرده است.

در این دوران سه ساله دوستان خوبی داشته‌ام. دوستانی که وقت نداشتن‌ها و بداخلاقی‌های یک دانش‌جوی فوق‌لیسانس را تحمل کرده و او را فراموش نکردند. دوستانی که ساعت‌ها با من سر موضوعات علمی، فلسفی و یا هر چیز دیگری که بتوان تصور کرد بحث می‌کردند و ساعات هیجان‌انگیز

و لذت‌بخشی را برای‌ام به وجود آوردند. نمی‌توانم همه‌شان را نام ببرم و بگویم چرا از بابت وجودشان خوش‌حال‌ام. هر کدام دلیل خاصِ خود را دارند که با یکی دو کلمه خلاصه‌پذیر نیست. در این‌جا تنها نام آن‌هایی را می‌آورم که در این مدت به طور پیوسته با آن‌ها ارتباط داشته‌ام. از احمد ایمانی‌پور، احمد پورصابری، آیدین کرویی، بتسابه درودیان، پریسا درویش‌زاده، پریسا رشیدی، پریسا میرشمس، پویان شیوا، پیمان حامدحسینی، حسین مباهی، رامین مهران، رضا پدرامی، روزبه دانشور، روزبه فضل، سعید بیتی، محسن محمدی تاکامی، محمد سلیمانی، محمد شاه‌محمدی، محمد قشلاقی، محمد میلاسی، مریم گودرزی، هاجر همایی و خیلی‌های دیگر که نام‌شان را نیاورده‌ام متشکرم. تشکر مخصوص من به لنا عباسی است که هر جا ناامید شده بودم مرا امید داد و هر جا مایوس شده بودم به پیش راند و در ضمن پای ثابت و همیشگی‌ی نظریه‌ها و خیال‌پردازی‌های من بوده است. از بابت وجود او خوش‌حال‌ام!

در نهایت و مهم‌تر از همه می‌بایست از دلشاد، نوید، کیان و مادربزرگ‌ام تشکر کنم که همیشه پشتیبان من بودند و به‌ترین شرایط ممکن را برای فعالیت علمی برای‌ام فراهم کردند. متاسفانه نوشتار قادر به بیان احساس من نسبت به آن‌ها نیست و در نتیجه بیش از این سعی در بیان‌اش نمی‌کنم. متشکرم!

فهرست

فهرست جداول

فهرست شکل‌ها

شکل ۵-۶ (مساله‌ی مجرد) مقایسه احتمال خطای یادگیری رفتار (ساختار ثابت) و یادگیری هم‌زمان ساختار و رفتار.

شکل ۶-۶ (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۱۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.

شکل ۷-۶ (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۵۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.

شکل ۸-۶ (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۱۵۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.

شکل ۹-۶ (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۳۰۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.

شکل ۱۰-۶ (مساله مجرد) مقایسه‌ی بین متوسط و حداکثر سازگاری جمعیت به ازای روش‌های مختلف اشتراک سازگاری هم‌راه با یادگیری ساختار. اشتراک سازگاری یک‌نواخت (آبی)، اشتراک سازگاری ارزش‌محور (سیاه). خط سبز متوسط سازگاری معماری‌ای با ساختار و رفتار کاملا تصادفی است (با احتمال انتخاب عمل و NA برابر) و خط قرمز حداکثر سازگاری دست‌یافتنی است. خطوط پررنگ متوسط سازگاری جمعیت را نشان می‌دهند و خطوط نقطه‌چین، حداکثر سازگاری را.

شکل ۱۱-۶ (مساله مجرد) مقایسه‌ی متوسط و حداکثر سازگاری به ازای روش‌های مختلف‌ای که از اشتراک سازگاری‌ی یک‌*نواخت* استفاده می‌کنند: تکامل رفتارها و یادگیری ساختار بدون استفاده از مم (آبی)، تکامل رفتارها و یادگیری ساختار با استفاده از مم (سیاه)، تکامل رفتارها و ساختار ثابت (بنفش). خطوط پررنگ متوسط سازگاری‌ی جمعیت و خطوط نقطه‌چین حداکثر سازگاری را نشان می‌دهند. حداکثر سازگاری‌ی ممکن با خط قرمز نشان داده شده است.

شکل ۱۲-۶ (مساله‌ی مجرد) مقایسه‌ی چگالی احتمال سازگاری برای تکامل با اشتراک سازگاری یک‌*نواخت* در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (آبی تیره)، عامل‌هایی با ساختار اولیه‌ی تصادفی (سبز) و عامل‌هایی با ساختار ثابت (قرمز) انجام گرفته است.

شکل ۱۳-۶ (مساله‌ی مجرد) مقایسه‌ی توزیع احتمال تجمعی ($P\{Fitness \leq f\}$) سازگاری برای تکامل با اشتراک سازگاری یک‌نواخت در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (سیاه)، عامل‌هایی با ساختار اولیه تصادفی (آبی) و عامل‌هایی با ساختار اولیه ثابت (بنفش) انجام گرفته است. تمایلِ نمودار به سمت راست نشان‌گر بالاتر بودن احتمال تولید عامل‌های سازگارتر است.

شکل ۱۴-۶ (مساله مجرد) مقایسه‌ی متوسط و حداکثر سازگاری به ازای روش‌های مختلف‌ای که از اشتراک سازگاری‌ی *ارزش‌محور* استفاده می‌کنند: تکامل رفتارها و یادگیری ساختار بدون استفاده از مم (آبی)، تکامل رفتارها و یادگیری ساختار با استفاده از مم (سیاه)، تکامل رفتارها و ساختار ثابت (بنفش). خطوط پررنگ متوسط سازگاری‌ی جمعیت و خطوط نقطه‌چین حداکثر سازگاری را نشان می‌دهند. حداکثر سازگاری‌ی ممکن با خط قرمز نشان داده شده است.

شکل ۱۵-۶ (مساله‌ی مجرد) مقایسه‌ی چگالی احتمال سازگاری برای تکامل با اشتراک سازگاری *ارزش‌محور* در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (آبی تیره)، عامل‌هایی با ساختار اولیه‌ی تصادفی (سبز) و عامل‌هایی با ساختار ثابت (قرمز) انجام گرفته است.

شکل ۱۶-۶ (مساله‌ی مجرد) مقایسه‌ی توزیع احتمال تجمعی ($P\{Fitness \leq f\}$) سازگاری برای تکامل با اشتراک سازگاری ارزش‌محور در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (سیاه)، عامل‌هایی با ساختار اولیه تصادفی (آبی) و عامل‌هایی با ساختار اولیه ثابت (بنفش) انجام گرفته است. تمایلِ نمودار به سمت راست نشان‌گر بالاتر بودن احتمال تولید عامل‌های سازگارتر است.

شکل ۱۷-۶ سه روبات در حال بلند کردن جسم‌ای بزرگ

شکل ۱۸-۶ (مساله‌ی بلندکردن جسم) مقایسه متوسط سیگنال تقویتی دریافتی بین روش‌های یادگیری ساختار مبتنی بر بازنمایی مرتبه صفر (ZO) و بازنمایی مرتبه یک(FO)، ساختار از پیش طراحی شده و ساختار اتفاقی. رفتارها توسط انسان طراحی شده‌اند.

شکل ۱۹-۶ (مساله‌ی بلندکردن جسم) مقایسه معیار رفتارگرایانه بین روش‌های یادگیری ساختار مبتنی بر بازنمایی مرتبه صفر (ZO) و بازنمایی مرتبه یک(FO)، ساختار از پیش طراحی شده و ساختار اتفاقی. رفتارها توسط انسان طراحی شده‌اند.

شکل ۲۰-۶ (مساله‌ی بلندکردن جسم) مقایسه متوسط سیگنال تقویت دریافتی بین یادگیری ساختار (رفتار طراحی‌شده)، یادگیری رفتار (ساختار طراحی‌شده) و یادگیری هم‌زمان ساختار و رفتار.

شکل ۲۱-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی سیگنال تقویت دریافتی ($P\{\text{average gained reward} \le x\}$) در فاز یادگیری (بین اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای ترکیب‌های مختلف روش‌های یادگیری.

شکل ۲۲-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی سیگنال تقویت دریافتی ($P\{\text{average gained reward} \le x\}$) در فاز آزمون (بین اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای ترکیب‌های مختلف روش‌های یادگیری.

شکل ۲۳-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی معیار رفتارگرایانه ($P\{\text{behavioral performance} \le x\}$) در فاز یادگیری (بین اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای ترکیب‌های مختلف روش‌های یادگیری.

شکل ۲۴-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی معیار رفتارگرایانه ($P\{\text{behavioral performance} \le x\}$) در فاز آزمون (بین اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای ترکیب‌های مختلف روش‌های یادگیری.

شکل ۲۵-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی پاداش دریافت شده در فاز یادگیری (اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای روش‌های مختلف یادگیری.

شکل ۲۶-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی پاداش دریافت شده در فاز آزمون (اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای روش‌های مختلف یادگیری.

شکل ۲۷-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی معیار رفتارگرایانه در فاز یادگیری (اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای روش‌های مختلف یادگیری.

شکل ۲۸-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی معیار رفتارگرایانه در فاز آزمون (اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای روش‌های مختلف یادگیری.

شکل ۲۹-۶. (مساله‌ی بلندکردن جسم) نمونه‌ای از مسیر حرکت نقاط تماس روبات-جسم، زاویه‌ی کج‌شده‌گی به هنگام بلند کردن جسم، و رفتارهای کنترل‌کننده‌ی روبات در هر واحد زمانی. این مسیر حاصل چندین اپیزود یادگیری هم‌زمان رفتار و ساختار است. تناظر رفتارها و شماره‌های روی نمودار پایین بدین‌گونه است:

0 (No Behavior), 1 (Push More), 2 (Don’t Go Fast), 3 (Stop), 4 (Hurry up), 5 (Slow down).

شکل ۳۰-۶. (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی به ازای طراحی‌های مختلف: ۱) تکامل رفتارها (اشتراک سازگاری‌ی یک‌نواخت) و یادگیری ساختار (آبی)، ۲) تکامل رفتارها (اشتراک سازگاری ارزش‌محور) و یادگیری ساختار (سیاه)، ۳) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۴) رفتارها و ساختار طراحی شده (قرمز). خطوط نقطه‌چین اطراف حالت‌های طراحی‌شده (۳ و ۴) ناحیه‌ی یک انحراف معیار از متوسط را نشان می‌دهند.

شکل ۳۱-۶. (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی و متوسط سازگاری کلِ دوره‌ی زندگی عامل در طراحی‌های مختلفی که از اشتراک سازگاری *یک‌نواخت* استفاده می‌کنند: ۱) تکامل رفتارها و یادگیری ساختار بدون کمک مم (آبی)، ۲) تکامل رفتارها و یادگیری ساختار با کمک مم (سیاه)، ۳) تکامل رفتارها و ساختار طراحی‌شده (بنفش)، ۴) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۵) رفتارها و ساختار طراحی‌شده (قرمز). خطوط پررنگ بیان‌گر متوسط سازگاری در پنج اپیزود انتهایی زندگی‌ی عامل هستند و خطوط نقطه‌چین سازگاری‌ی کلِ زندگی عامل را نشان می‌دهند. با وجود آن‌که عمل‌کرد آخرین اپیزودهای همه‌ی عامل‌ها تقریبا یک‌سان است اما عمل‌کرد کلِ دوره‌ی زندگی‌ی عامل‌هایی که از اطلاعات مم استفاده می‌کنند بسیار بالاتر است.

شکل ۳۲-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *یک‌نواخت*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.

شکل ۳۳-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری کلِ دوره‌ی زندگی عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *یک‌نواخت*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.

شکل ۳۴-۶ (مساله‌ی بلندکردن جسم) مقایسه‌ی احتمال تجمعی سازگاری ($P\{Fitness \leq f\}$) عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *یک‌نواخت*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (سیاه)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (آبی) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (بنفش) انجام شده است. خطوط نقطه‌چین سازگاری‌ی کلِ دوره‌ی زندگی‌ی عامل را نشان می‌دهند. هر چه توزیع بیش‌تر به سمت راست گرایش داشته باشد، شانس ایجاد عامل‌های خیلی خوب بیش‌تر است.

شکل ۳۵-۶ (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی و متوسط سازگاری کلِ دوره‌ی زندگی عامل در طراحی‌های مختلفی که از اشتراک سازگاری *ارزش‌محور* استفاده می‌کنند: ۱) تکامل رفتارها و یادگیری ساختار بدون کمک مم (آبی)، ۲) تکامل رفتارها و یادگیری ساختار با کمک مم (سیاه)، ۳) تکامل رفتارها و ساختار طراحی‌شده (بنفش)، ۴) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۵) رفتارها و ساختار طراحی‌شده (قرمز). خطوط پررنگ بیان‌گر متوسط سازگاری در پنج اپیزود انتهایی زندگی‌ی عامل هستند و خطوط نقطه‌چین سازگاری‌ی کلِ زندگی عامل را نشان می‌دهند. با وجود آن‌که عمل‌کرد آخرین اپیزودهای همه‌ی عامل‌ها تقریبا یک‌سان است اما عمل‌کرد کلِ دوره‌ی زندگی‌ی عامل‌هایی که از اطلاعات مم استفاده می‌کنند بالاتر است.

شکل ۳۶-۶ (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.

شکل ۳۷-۶ (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری کلِ دوره‌ی زندگی عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.

شکل ۳۸-۶ (مساله‌ی بلندکردن جسم) مقایسه‌ی احتمال تجمعی سازگاری ($P\{Fitness \leq f\}$) عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از

۱) مقدمه و پیشینه‌ی پژوهش

۱-۱) مقدمه

عمومیت استفاده از عامل‌های[1] خودکاری که در محیط‌هایی مانند کارخانه یا منازل به کار می‌روند روز به روز بیش‌تر می‌شود. به تدریج روبات‌ها از آزمایش‌گاه‌ها خارج شده و وارد محیط‌های واقعی می‌شوند. این فرآیند نیازمند وجود مکانیزم‌های تصمیم‌گیری‌ی مقاوم‌ای است که بتواند به خوبی از عهده‌ی نایقینی‌ها و خطاهای پیش‌آمده برای روبات برآید. روبات‌ای که در محیط واقعی قرار می‌گیرد با بسیاری از عوامل اختلال‌زا مانند نویز سنسور، خطای محرک[2] و پیچیدگی و تغییر دایم محیط مواجه است. هم‌چنین روبات در اکثر موارد به همه‌ی اطلاعات لازم برای تصمیم‌گیری دست‌رسی ندارد و میزان اطلاعات‌ای که سنسورها به او می‌دهند محدود است. تصمیم‌گیری در چنین شرایطی بسیار دشوار است و تلاش‌های اولیه‌ای که خواهان به‌کارگیری روش‌های کلاسیک هوش‌مصنوعی برای حل این‌گونه مسایل بودند با شکست مواجه شده‌اند. خوش‌بختانه معماری‌های رفتارگرا[3] که از اواسط دهه‌ی هشتاد میلادی مطرح شده‌اند بسیاری از این مشکلات عملی را حل کرده‌اند. معماری‌های رفتارگرا با تجزیه‌ی کل مساله به تعدادی رفتار مجزا (به جای تقسیم به تعدادی کارکرد[4]) بسیاری از مسایل دشوار پیشین را حل می‌کنند. با این‌همه هم‌چنان طراحی‌ی معماری‌ی مناسب‌ای برای حلِ مساله‌ای خاص چندان ساده نیست و در عملْ طراح با سعی و خطای بسیار به تدریج به حلِ مطلوب نزدیک خواهد شد. این پرسش مطرح می‌شود که آیا می‌توان بخش‌ای از این فعالیت وقت‌گیر را به خود عامل واگذار کرد تا با توجه به راه‌نمایی‌های طراح بتواند به معماری‌ی مطلوب برسد؟ تلاش برای پاسخ مثبت دادن به چنین پرسش‌ای انگیزه‌ی اصلی‌ی این پژوهش بوده است. هدف اصلی‌ی این پژوهش ایجاد مکانیزم‌هایی برای طراحی‌ی خودکار معماری‌های رفتارگرا است. به عبارت دیگر، قصد داشتم تا روش‌های کلی‌ای ایجاد کنم تا به وسیله‌ی آن طراح بتواند بخش عمده‌ای از دشواری‌ی فرآیند طراحی را به خود عامل واگذارد. در این

[1] Agents
[2] Actuator
[3] Behavior-based Architectures
[4] Function

پژوهش ایده‌های گوناگونی را برای خودکارسازی‌ی طراحی‌ی معماری‌های رفتارگرای سلسله مراتبی پیش‌نهاد می‌کنم.

برای خودکار کردن فرآیند طراحی لازم است معیاری برای برتری‌ی طرح‌ای بر طرح دیگر وجود داشته باشد. این معیار که توسط طراح مشخص می‌شود بیان‌گر دیدگاه او به راه حل مناسب مساله است. به عنوان مثال در اکثر موبایل روبات‌ها برخورد کردن با اجسام رفتار مطلوب‌ای نیست. طراح می‌تواند این خواست را به صورت امتیاز منفی‌ای مشخص کند: هر برخورد معادل یک واحد امتیاز منفی باشد. آن‌گاه با این معیار، طرح‌ای به‌تر است که امتیاز منفی‌ی کم‌تری بگیرد و در نتیجه به‌ترین طرح ممکن معادل کنترلری است که کم‌ترین برخورد را موجب شود. با این دیدگاه، فرآیند طراحی معادل با بهینه‌سازی‌ی معیار مشخص شده توسط طراح است.

در این پژوهش برای حل مساله‌ی بهینه‌سازی یاد شده دو روی‌کرد متفاوت را بررسی کرده‌ام. اولین روی‌کرد، استفاده از یادگیری تقویتی[۵] بوده است. روبات با استفاده از سیگنال تقویت[۶] —که به صورت پاداش یا تنبیه‌ای برای عامل خواهد بود– سعی در ایجاد به‌ترین معماری‌ی رفتارگرای ممکن می‌کند. این فرآیند —که در آینده به طور دقیق به آن خواهم پرداخت– در دو سطح تطبیق ساختار معماری (فصل ۳) و تطبیق رفتارهای قرارگرفته در معماری (فصل ۴) صورت می‌پذیرد. روی‌کرد دیگری که برای حل مساله‌ی بهینه‌سازی مطرح شده، بهره‌جویی از تکامل مصنوعی[۷] است. با استفاده از تکامل، جمعیت‌ای از عامل‌ها سعی در پیدا کردن به‌ترین معماری‌ی ممکن می‌کنند. در پژوهش حاضر فرآیند تکامل در سطح تطبیق رفتارها و به صورت تکامل هم‌کارانه انجام می‌شود (فصل ۵). علاوه بر این دو نگاه، استفاده از ایده‌های فرهنگی و انتقال دانش از طریق فرهنگ مشترک بین عامل‌ها با معرفی‌ی مفهوم‌ای به نام مم[۸] و الگوریتم‌های ممتیک[۹] نیز مورد مطالعه قرار گرفته است (فصل ۵). نتایج این دو روی‌کرد را در فصل مربوط به آزمایش‌ها (فصل ۶) آورده‌ام. در فصل ۷ به تحلیل احتمالاتی‌ی معماری‌ی

[5] Reinforcement Learning
[6] Reinforcement Signal
[7] Artificial Evolution
[8] Meme
[9] Memetic Algorithm

شکل ۱-۱. تجزیه کارکردگرایانه‌ی یک وظیفه (روی‌کرد هوش مصنوعی کلاسیک)

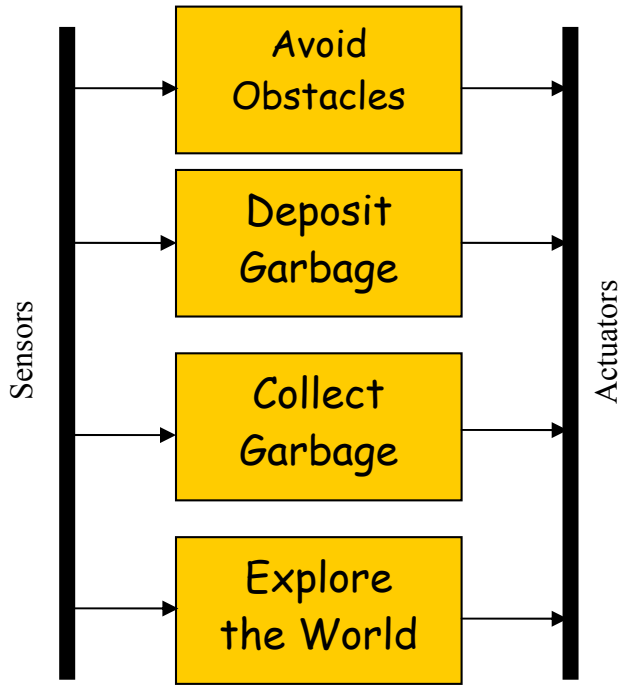
سلسله‌مراتبی‌ی به کار رفته در این پایان‌نامه می‌پردازم و خواص آن را مطالعه می‌کنم. در فصل ۸ نیز به تاثیر وجود عدم قطعیت در سیگنال تقویت بر تابع ارزش و تابع سیاست عامل می‌پردازم.

در بخش‌های پیش رو بیش‌تر درباره‌ی معماری‌های رفتارگرا و هم‌چنین پژوهش‌های پیشین‌ای که برای خودکار کردن طراحی‌ی معماری‌های رفتارگرا صورت گرفته است خواهم نوشت.

۱-۲) معماری‌های رفتارگرا

برخلاف عامل‌هایی که در محیط‌های مجرد[۱۰] تصمیم‌گیری می‌کنند (مانند عامل جستجوگر ماز، عامل شطرنج‌باز و ...)، عامل جسم‌دار[۱۱] در محیط واقعی قرار گرفته[۱۲] با بسیاری از عدم قطعیت‌ها روبرو است: ورودی‌ی اطلاعات چنان عامل‌ای نویزی و نادقیق است، در اکثر مواقع دانش‌اش نسبت به محیط بسیار جزئی و ناکامل است (از دید عامل، محیط به طور جزئی رویت‌پذیر[۱۳] است)، عمل‌ای که انتخاب می‌کند الزاما به همان صورت اجرا نمی‌شود و هم‌چنین معمولا محیط پیچیده و متغیر با زمان است. همه‌ی این‌ها باعث می‌شود که روی‌کرد متداول هوش مصنوعی‌ی کلاسیک که مبتنی بر ادراک[۱۴] محیط، مدل مجردسازی از محیط، برنامه‌ریزی[۱۵] در آن مدل مجرد و سپس اجرای فرمان‌ها باشد (شکل ۱-۱) غیرعملی باشد. عدم قطعیت محیط از دید عامل آن چنان دقت مدل ساخته شده را پایین می‌آورد که برنامه‌ریزی‌ی انجام گرفته تقریبا بی‌فایده خواهد شد. طراحی‌ی رفتارگرا بسیاری از این مشکلات را حل

[10] Abstract
[11] Embodied
[12] Situated
[13] Partial Observable
[14] Perception
[15] Planning



شکل ۲-۱. تجزیه رفتارگرایانه‌ی یک وظیفه (روی‌کرد طراحی رفتارگرا)

می‌کند و به کمک آن دیدگاه روبات‌هایی طراحی شده‌اند که تا پیش از آن ممکن نبودند ([Brooks86]، [Brooks91a]، [Brooks91b]، [Brooks91c] و [Arkin98]). . برای نمونه‌ای از کارهای موفق‌ای که با استفاده از معماری‌ای رفتارگرا انجام گرفته است به [Brooks89]، [Wang96]، [Matarić98]، [Parker98]، [Nili01] مراجعه کنید.

این‌که طراحی‌ی رفتارگرا شامل چه مولفه‌هایی است و چه زمینه‌ی فلسفه‌ای دارد به طور دقیق مورد توافق همه‌ی پژوهش‌گران نیست. با این وجود شاید بتوان گفت که مهم‌ترین ویژگی‌ی آن، نوع تجزیه‌ی رفتار کلی‌ی عامل به تعدادی رفتار جزئی‌تر است. این رفتارهای جزئی هستند که در نهایت رفتار کلی‌ی عامل را ایجاد می‌کند. رفتارهای جزئی با تعامل با محیط و هم‌چنین تعامل با یک‌دیگر باعث تظاهر[۱۶] رفتار هوش‌مندانه می‌شوند. در این دیدگاه به جای نگاه‌ای کارکردگرایانه[۱۷] به اجزای تصمیم‌گیری (کارکرد درک محیط، کارکرد مدل‌سازی، کارکرد برنامه‌ریزی و ...)، نگاه‌ای رفتارگرایانه[۱۸] به

[16] Emergence
[17] Functionalism
[18] Behaviorism

مساله داریم. نگاه‌ای که با دیدی عینی[19] نه به ساز و کار داخلی‌ی ذهن که به نتیجه‌ی تصمیم ذهنی‌ی عامل در محیط –با توجه به همه‌ی قیدهای فیزیکی‌ی عامل و محیط– می‌پردازد.

برای روشن‌تر شدن قضیه، فرض کنید که قصد ساخت روبات‌ای را داریم که آشغال‌ها را از روی زمین جمع می‌کند و به مکان مشخص‌ای می‌برد. هم‌چنین می‌دانیم که روبات نباید به اجسام محیط برخورد کند. دیدگاه کلاسیک (و کارکردگرایانه) ابتدا سعی در درک محیط با استفاده از سنسورهای روبات می‌کند و سپس با استفاده از اطلاعات استخراج شده به ساختن مدل‌ای مجرد از دنیای اطراف‌اش تلاش می‌کند. چنین کاری ساده نیست چون عدم قطعیت سنسورها باعث پیچیده و یا حتی غیرممکن شدن عمل تشخیص دقیق ساختار محیط و مکان آشغال‌های دور و بر می‌شوند. پس از انجام این‌کار –در صورت امکان– می‌بایست در فضای بسیار بزرگ ایجاد شده به گونه‌ای مسیر روبات را برنامه‌ریزی کرد تا نه تنها همه‌ی آشغال‌ها جمع شوند بلکه قیدهای دیگری چون برخورد نکردن با موانع نیز برقرار شود. به دلیل ابعاد بزرگ فضای‌ی برنامه‌ریزی، چنین کاری الزاما خیلی ساده نیست. مهم‌تر از آن به دلیل مجرد بودن ورودی‌ی برنامه‌ریز، در صورت‌ای که مرحله‌ی مدل‌سازی دقیق انجام نشده باشد –که در اکثر موارد این‌چنین است– خطای مدل‌سازی باعث برنامه‌ریزی در محیطی می‌شود که متناظری در دنیای واقعی ندارد. حتی در صورت‌ای که بتوان از این موانع گذشت، نتیجه‌ی نهایی نه تنها دقت بالا و مورد انتظار روش‌های کلاسیک را ندارد بلکه به دلیل کند بودن عملیات انجام شده در تجزیه‌ی کارکردگرایانه‌ی وظیفه، عامل بسیار کند و لَخت خواهد بود. چنین عمل‌کردی برای مسایل دنیای واقعی قابل قبول نیست.

در دیدگاه طراحی‌ی رفتارگرا، وظیفه‌ی مورد نظربه تعدادی جزء *رفتاری* تجزیه می‌شود (شکل ۱–۲). جزء رفتاری آن چیزی است که ناظر بیرونی به هنگام مشاهده‌ی عامل‌ای که آن مساله را به درستی حل می‌کند آن را یک رفتار مجزا تشخیص می‌دهد. مثلا اگر عامل‌ای –مثلا انسان– عمل مورد نظر را به درستی انجام دهد، ناظر رفتارهای "به سمت آشغال حرکت کردن"، "دوری جستن از اجسام"، "دور زدن موانع"، "به سمت هدف حرکت کردن" و ... را در فعالیت آن عامل تشخیص می‌دهد. توجه کنید که این‌ها الزاما آن‌چیزی نیستند که در ذهنِ عامل می‌گذرد اما از آن‌جا که ما به ذهنِ عامل –مثلا

[19] Objective

انسان- دست‌رسی‌ی مستقیم نداریم، تنها فرآیند عینی و اندازه‌پذیرمان —در تعریف غیرریاضی‌ی آن- همین رفتارهای مشخص شده است. حال اگر بیاییم و هر کدام از این رفتارها را به طور مجزا تولید کنیم، مجموعه‌ای از رفتارهایی خواهیم داشت که با استفاده از آن رفتارها و انتخاب صحیح و به موقع‌شان می‌توان رفتار کلی‌ی عامل را بازتولید نمود. این کار —یعنی تولید رفتار- فرآیندی است به ظاهر متفاوت با فلسفه‌ی رفتارگرایی. بعضی از پژوهش‌گران (مثلا [Matarić01]) اعتقاد دارند که طراحی‌ی رفتارگرا نمونه‌ای از رفتارگرایی نیست. دلیل آن نظر این است که فلسفه‌ی رفتارگرایی اجازه‌ی گمانه‌زنی درباره‌ی ماهیت داخلی‌ی ذهنِ منجر به این رفتار را به تحلیل‌گر رفتار نمی‌دهد. اما شاید توجه به این نکته که هدف رفتارگرایی تحلیل —و نه طراحی- بوده است تا حدی راه‌گشا باشد. طراح معماری‌ی رفتارگرا با الهام از این ایده که حلِ مساله را به صورت تعدادی رفتار مجزا ببیند وام‌دار رفتارگرایی است اما به هنگام طراحی‌ی هر کدام از رفتارها از رفتارگرایی جدا شده و به صورت زمینه‌ی پژوهشی‌ی متفاوت‌ای در می‌آید. در نتیجه —به اعتقاد من- طراحی‌ی رفتارگرا با وام‌گیری از بخش‌ای از ایده‌ی رفتارگرایی سعی در حلِ مساله‌ی ساخت عامل هوش‌مند می‌کند.

ویژگی‌ی خاص و اساسی‌ی این تجزیه، ساده‌سازی‌ی فرآیند طراحی است. معمولا در اکثر مسایل، طراحی‌ی یک جزء رفتاری بسیار ساده‌تر از طراحی‌ی سیستم کلاسیک‌ای است که برای حل مساله، همه‌ی ابعاد آن را با هم در نظر می‌گیرد. این کار در مسایل دنیای واقعی بسیار پیچیده و غیرقابل پی‌گیری[20] است. اما برخلاف آن طراحی‌ی یک رفتار ساده —مانند "دوری جستن از اجسام"- نیازی به تحلیل دقیق و کامل دنیای اطراف ندارد و به جای کارکردن در کلِ فضای مساله، تنها بخش کوچک ولی مربوطی مانند فاصله با اجسام نزدیک مورد توجه قرار می‌گیرد. پس از طراحی‌ی اجزای رفتاری مورد نیاز می‌بایست شرایط فعال شدن هر کدام از آن رفتارها را مشخص کرد. این‌که چه زمان‌ای رفتار خاص‌ای می‌بایست فعال شود و کنترل عامل را در اختیار بگیرد، مساله‌ای است که بخش مهم‌ای از فرآیند طراحی‌ی رفتارگرا را شامل می‌شود. در اکثر موارد این تصمیم‌گیری به عهده‌ی طراح است و او با روش‌ای پایین به بالا[21] سعی در پیدا کردن به‌ترین ترکیب رفتارها می‌کند: طراح ابتدا با انتخاب یک رفتار

[20] Intractable
[21] Bottom up

شروع می‌کند و رفتار کلی‌ی عامل را مشاهده می‌کند. در صورت مطلوب نبودن نتیجه یا شرایط کنترل‌کننده بودن رفتار را تغییر می‌دهد یا رفتار جدیدی ایجاد می‌کند (یا از جعبه ابزار رفتارهای‌اش، رفتار جدیدی را انتخاب می‌کند) تا در مواردی که مجموعه رفتارهای پیشین به خوبی عمل نکرده‌اند وارد عمل شده و باعث رفتار مطلوب عامل شود. این عمل آن‌قدر تکرار می‌شود تا رفتار کلی‌ی مورد نظر به دست آید.

روی‌کرد طراحی‌ی رفتارگرا معمولا برای طراحی‌ی کنترل‌کننده‌های هوش‌مندی که قرار است برای عامل‌های محیط واقعی عمل کند موفق‌تر از روی‌کرد کلاسیک نتیجه می‌دهد. طراحی‌ی رفتارگرا نه تنها ساده‌تر است بلکه از آن‌جا که فرآیند تصمیم‌گیری‌ی درونی‌ی هر رفتار در اکثر مواقع ساده‌تر و کم محاسبه‌تر از فرآیند لازم برای حلِ کلِ مساله است، کنترلر حاصل بسیار سریع‌تر واکنش می‌دهد. این نکته مخصوصا با در نظر گرفتن این‌که طراحی‌ی رفتارگرا ذاتا گسترده[۲۲] است (هر رفتار یک پروسه‌ی مجزا است) آشکارتر می‌شود.

هدف پژوهش فعلی همان‌طور که پیش‌تر گفته شد خودکار کردن فرآیند طراحی‌ی معماری‌ای رفتارگرا است. سعی کرده‌ام با استفاده از یادگیری و تکامل روش‌ای برای طراحی‌ی خودکار معماری‌ی رفتارگرا —که تا پیش از این توسط طراح طرح می‌شد– معرفی کنم.

۱-۳) کارهای مرتبط

در این بخش مروری بر کارهای مشابه که به نحوی به پژوهش فعلی مربوط است خواهم داشت. در ابتدا مروری بر روش‌های طراحی‌ی خودکار عامل‌های رفتارگرا می‌کنم و سپس بعضی از روش‌های یادگیری‌ی سلسله مراتبی را بررسی خواهم کرد.

۱-۳-۱) طراحی‌ی خودکار عامل‌های رفتارگرا

تقسیم روش‌های ممکن برای طراحی‌ی معماری‌های رفتارگرا به دو دسته‌ی روش‌های مبتنی بر یادگیری و روش‌های مبتنی بر تکامل ویژه‌ی این پژوهش نیست.

[22] Distributed

روش‌های مبتنی بر یادگیری، نتایج اولیه‌ی قابل توجه و موفقیت‌آمیزی در طراحی‌ی خودکار سیستم‌های رفتارگرا داشته‌اند. اما اکثر آن‌ها مساله‌ی یادگیری را در فرم چندان کامل و جامع‌ای بررسی نکرده‌اند. هم‌چنین تحلیل ریاضی و دقیق‌ای بر چگونگی‌ی حل مساله صورت نگرفته است و اکثر روش‌ها تنها به بهینه‌سازی قسمت‌ای از معماری پرداخته‌اند. به عنوان نمونه‌ای از کارهای انجام شده می‌توان به [Maes90] اشاره کرد که از یادگیری برای تنظیم پیش‌شرط فعال شدن مجموعه‌ای از رفتارهای از پیش تعیین شده استفاده کرده است. این پژوهش یکی از اولین کارهایی است که در این زمینه صورت گرفته و از روی‌کردی مشابه —اما نه دقیقا یک‌سان– با آن‌چه در تئوری‌ی محاسباتی شده‌ی یادگیری تقویتی معمول است بهره جسته (برای شرحی از روش‌های یادگیری تقویتی به [Sutton98] مراجعه کنید). در [Mahadevan92]، از یادگیری برای تغییر رفتارها در معماری‌ای بسیار مشابه با معماری‌ای که در این پژوهش استفاده کرده‌ام —یعنی معماری‌ی Subsumptionی موازی– استفاده شده است. در آن پژوهش ساختار از پیش ثابت است و هر رفتار دارای سیگنال تقویت مجزایی است. این برخلاف روی‌کرد پژوهش فعلی است که در آن نه معماری ثابت است و نه هر لایه — یا رفتار– دارای سیگنال تقویت مجزایی است. در [Matarić92] از ساختاری کاملا رفتارگرا برای یادگیری نقشه‌ی توپولوژیک محیط استفاده شده است. در آن کار ساختار معماری ثابت نیست و در صورت لزوم گسترش می‌یابد. افزودن هر رفتار معادل یافتن زمین‌نشان‌ای[۲۳] جدید است. در [Matarić94] و [Matarić97]، از سیگنال تقویت شکل‌دهی‌شده[۲۴] و تخمین‌زن پیش‌رفت[۲۵] برای سرعت بخشیدن به یادگیری در معماری‌ای رفتارگرا بهره گرفته شده است. البته معماری‌ی به‌کار رفته در آن کار سلسله مراتبی نیست و رفتارها در یک سطح با هم رقابت می‌کنند. هم‌چنین در [Michaud98] نیز برای انتخاب رفتارها از روی‌کردی حافظه‌محور[۲۶] استفاده شده است. بدین معنا که با توجه به تاریخ‌چه‌ی حالت‌هایی که روبات در آن قرار گرفته است و با توجه به نرخ موفقیت روبات در آن حالت‌ها، امتیازی به آن حالت داده می‌شود. این‌کار شبیه به کارهای مک‌کالوم[۲۷] در Nearest Sequence Matching

[23] Landmark
[24] Shaped Reinforcement Signal
[25] Progress Estimator
[26] Memory-based
[27] McCallum

([McCallum95a]) و Utile Suffix Memory ([McCallum95b]) برای در نظر گرفتن حافظه و تاثیر تاریخچه‌ی حالت‌ها در انتخاب عمل مناسب است. برای مروری بر کارهای مشابه که از یادگیری برای تطبیق در معماری‌های رفتارگرا استفاده کرده باشند به [Matarić01] مراجعه کنید.

روش‌های مبتنی بر تکامل نیز در روباتیک استفاده شده است. در این روش‌ها کنترلر روبات – که معمولا به صورت یک‌پارچه[۲۸] توصیف می‌شوند – با استفاده از میزان سازگاری‌ی[۲۹] عامل استفاده‌کننده از آن کنترلر تکامل می‌یابد. به دلیل یک‌پارچه بودن اکثر کنترلرهای تکاملی و نبود چندین رفتار مختلف به صورت مشخص و بارز (و نه فقط به شکل نمود خارجی) می‌توان در رفتارگرا بودن این نوع طراحی شک داشت و اکثر کنترلرهای طراحی شده را بیش‌تر جزو خانواده‌ی کنترل‌های رآکتیو[۳۰] در نظر گرفتن. در بسیاری از پژوهش‌ها از کنترلرهای مبتنی بر شبکه‌ی عصبی استفاده می‌شود و پارامترهای تکاملْ وزن‌های شبکه‌ی عصبی است. به عنوان نمونه به [Harvey93]، [Cliff93]، [Floreano94]، [Floreano96]، [Urzelai98]، [Floreano99]، [Floreano00]، [Nolfi00]، [Chernova04] مراجعه کنید. به عنوان کارهایی که شباهت بیش‌تری با کار فعلی دارند می‌توان به [Koza94]اشاره کرد. در آن پژوهش از برنامه‌ریزی ژنتیکی[۳۱] برای تولید معماری‌ی Subsumption مشابه با معماری‌ی فعلی استفاده شده است. هدف استفاده از تکاملْ ایجاد کنترلری بوده است که عمل تعقیب دیوار[۳۲] را به درستی انجام دهد. هم‌چنین [Togelius04] نیز از تکامل لایه‌ای[۳۳] اضافه‌شونده[۳۴] برای ایجاد معماری‌ای مشابه با معماری‌ی این پژوهش استفاده کرده است.

با وجود این‌که روش‌های متداول روباتیک تکاملی به نظر توان‌مند می‌آیند اما ایرادهای مشخص‌ای نیز دارند. به عنوان یکی از بزرگ‌ترین مشکلات آن‌ها می‌توان به سرعت پایین‌شان نسبت به روش‌هایی مبتنی بر یادگیری اشاره کرد. هم‌چنین اکثر کنترلرهای به‌کار رفته در روباتیک تکاملی به

[28] Monolithic
[29] Fitness
[30] Reactive
[31] Genetic Programming
[32] Wall Following
[33] Layered
[34] Incremental

دلیل یک‌پارچه بودن‌شان قابلیت استفاده‌ی مجدد ندارند. در این‌باره در فصل ۵ بیش‌تر توضیح خواهم داد.

۱-۳-۲) یادگیری تقویتی سلسله مراتبی[۳۵]

ایده‌ی استفاده از یادگیری در ساختارهای سلسله مراتبی ایده‌ی چندان تازه‌ای نیست. در چند سال اخیر پژوهش‌های بسیاری در حوزه‌ای به نام یادگیری تقویتی سلسله‌مراتبی انجام شده است که نتایج موفق‌ای نیز داشته است. از آن‌جا که در این پژوهش از معماری‌ای سلسله مراتبی استفاده شده است، مروری مختصر بر روش‌های متداول یادگیری‌ی آن معماری‌ها مفید می‌نماید.

برای شروع، به این مثال توجه کنید: فرض کنید که موبایل روبات‌ای قصد یادگیری‌ی رفتار خروج از ساختمان را داشته باشد. در روش‌های معمول یادگیری، عمل[۳۶]های ممکن روبات بسیار جزیی هستند: مثلا حرکت در یکی از چهار جهت اصلی. روبات‌ای که در اتاقی قرار گرفته است برای خروج از آن می‌بایست با حرکت‌های کوچک و پایه‌ای[۳۷] خود آن‌قدر در محیط به کاوش[۳۸] بپردازد تا به خروجی‌ی ساختمان برسد. در آن هنگام روبات پاداش‌ای دریافت می‌کند و انتخاب‌های قبلی‌اش تقویت می‌شود. حال اگر طراح این روبات به جای حرکت‌های پایه‌ای، او را به حرکت‌های مرکب‌ای مانند "حرکت به سمت در"، "حرکت در طول راه‌رو"، "دور زدن از موانع" و ... نیز مجهز کرده باشد، روبات می‌تواند با انتخاب چند حرکت مرکب به خروجی برسد و پاداش لازم را دریافت کند. این حرکت‌های لازم در واقع مجموعه‌ای از حرکت‌های پایه‌ای است که توسط برنامه‌ای ایجاد می‌شود. این مثال بیان‌گر روش متداول‌ای به نام options است[Sutton99]. Options که به آن‌ها Macro Action نیز می‌گویند در اصل تعمیم‌ای بر عمل‌های ساده‌ای است که در یادگیری تقویتی مطرح می‌شود. هر option گسترش زمانی‌ی یک عمل است و به همین دلیل optionها را اعمال گسترش-زمانی یافته[۳۹] نیز می‌نامند. برخلاف انتخاب عمل‌ها در چارچوب MDP[۴۰] که هر انتخاب تنها باعث انتقال از یک حالت به حالت

[35] Hierarchical Reinforcement Learning
[36] Action
[37] Primitive
[38] Exploration
[39] Temporally Extended Action
[40] Markov Decision Problem

دیگر می‌شود ، هر option می‌تواند باعث گذشت چند واحد زمانی و انجام چند عمل پایه‌ای[۴۱] و در نتیجه چندین انتقال حالت متوالی شود. هنگامی که عامل یک option را انتخاب می‌کند، مجموعه‌ای از عمل‌های از پیش تعیین شده پشت سر هم اجرا می‌شود. از دیدگاهی هر option را می‌توان یک زیربرنامه[۴۲] دانست که مطابق با برنامه‌ای از پیش مشخص به انتخاب اعمال می‌پردازد. هر option حتی می‌تواند درون خود optionهای دیگر را نیز فراخوانی کند و بدین‌ترتیب تبدیل به روش‌ای بازگشتی[۴۳] — و در نتیجه سلسله مراتبی– می‌شود.

optionها دارای دو ویژگی‌ی مطلوب هستند: اول این‌که طراح در صورت داشتن دانش از پیش نسبت به رفتارهای مطلوب در آن محیط می‌تواند به طراحی‌ی optionهای مناسب آن مساله دست یازد. در این صورت سرعت یادگیری —یا برنامه‌ریزی– بسیار بالا می‌رود. ویژگی‌ی دیگر این است که optionها با کوچک کردن فضای تصمیم‌گیری باعث ساده‌تر و سریع‌تر شدن برنامه‌ریزی و یادگیری می‌شوند. البته در این بین حداکثر بازده ممکن است کم‌تر شود[۴۴].

یادگیری و برنامه‌ریزی با optionها –که تئوری‌شان بر پایه‌ی Semi-Markov Decision Process است[Puterman94]– بسیار مشابه با یادگیری در سیستم‌های متداول مبتنی بر MDP است و الگوریتم‌های مشابه با Q-Learning و SARSA نیز در آن وجود دارد.

یکی از مسایل بسیار مهم در این روش سلسله مراتبی پیدا کردن optionهای مناسب است. با وجود آن‌که در اکثر موارد optionها توسط طراح از پیش مشخص می‌شوند، اما چنین optionهایی الزاما به بالاترین بازده منجر نمی‌شوند: طراح ممکن است درک دقیقی از مساله نداشته باشد و زیربرنامه‌هایی بنویسند که به‌ترینِ ممکن برای آن مساله نباشد. روش‌های مختلفی برای طراحی‌ی خودکار option مطرح شده است. به عنوان مثال در [Precup00]، روش یادگیری intra-option معرفی شده که با استفاده از تابع پاداش ویژه‌ای برای هر زیرهدف[۴۵] در سیاست[۴۶] خود optionها نیز

---

[41] Primitive Action
[42] Subroutine
[43] Recursive
[44] اگر طراح به عامل اجازه دهد تا یادگیری را بر روی مجموعه‌ی حرکات پایه‌اش نیز انجام دهد (و آن‌ها را حذف نکند)، آن‌گاه بازده کم نخواهد شد..
[45] Subgoal
[46] Policy

تغییر ایجاد می‌کند. از دیدگاهی بسیار کلی این شبیه به روش یادگیری رفتار این پایان‌نامه است. گرچه ساختار ویژه و سلسله مراتبی‌ی معماری‌ی این پژوهش تغییرات اساسی‌ای را در روش یادگیری مناسب ایجاد می‌کند. در [McGovern01] نیز روش‌ای برای ایجاد تدریجی زیرهدف‌ها در طول یادگیری معرفی شده است. هم‌چنین در چارچوبIntrinsically Motivated Reinforcement Learning [Singh04] نیز فرآیندی برای تولید optionهای جدید به هنگام رسیدن به شرایطی که عامل به حالت‌های تازه و ناآشنا می‌رسد پیش‌نهاد شده است.

یکی دیگر از روش‌های یادگیری تقویتی سلسله‌مراتبی، روش Feudal RL است [Dayan93]. در این روش چند لایه، لایه‌های بالاتر –که دید کلی‌تری نسبت به مساله دارند– تصمیم می‌گیرند که کدام لایه‌ی پایین‌تر می‌بایست کنترل عامل را در اختیار بگیرد. لایه‌های بالاتر دیدی کلی‌تر ولی مجردتر نسبت به مساله دارند ولی لایه‌های پایین‌تر –که فرمان‌بردارند– دیدی جزیی ولی دقیق‌تر دارند. روش یادگیری‌ای که در این پژوهش به آن خواهم پرداخت نیز دارای ساختاری لایه به لایه است اما برخلاف روش Feudal RL، دیدی رفتارگرایانه نسبت به مساله دارد: مساله به صورت اجزاء رفتاری (و نه کارکردی) تقسیم می‌شود، هر لایه می‌تواند کنترل کل عامل را در اختیار بگیرد و هر کدام از رفتارها نگاشت‌ای مستقیم بین ورودی سنسور و خروجی‌ی حرکتی‌ی عامل است. هم‌چنین با وجود آن‌که لایه‌های بالاتر توانایی محدود کردن لایه‌های پایین را دارند، اما به طور مستقیم فرمان‌ای برای انتخاب یا تغییر حالت لایه‌های پایین نمی‌دهند. به عبارت دیگر معماری‌ی پیش‌نهادی –برخلاف Feudal RL– معماری‌ای گسترده است و به دلیل وجود نداشتن هماهنگ‌کننده‌ی مرکزی نسبت به خطاهای پیش‌آمده مقاوم‌تر است. علاوه بر این‌ها تفاوت حالت حس‌شده‌ی هر رفتار فقط در میزان تجرید آن نیست، بلکه هر رفتار می‌تواند از مجموعه‌ای کاملا متفاوت از حالت‌های موجود در عامل برای تصمیم‌گیری استفاده کند.

روش MaxQ Value Decomposition –که آن را می‌توان نسخه‌ی پیش‌رفته‌تر و کامل‌تر Feudal RL دانست– یکی دیگر از روش‌های متداول یادگیری تقویتی سلسله‌مراتبی است [Dietterich00]. در این روش وظیفه‌ی عامل به تعدادی زیر–وظیفه‌ی[۴۷] ساده‌تر –ولی کلی‌تر– تقسیم

[47] Subtask

می‌شود. هر کدام از این زیر–وظیفه‌ها مانند زیربرنامه‌هایی هستند که توسط قسمت اصلی‌ی برنامه‌ای فراخوانی می‌شوند. عامل برای انجام دادن هر وظیفه با توجه به حالت محیط یکی از این زیربرنامه‌ها را فراخوانی می‌کند. مثلا برای روبات‌ای که می‌بایست جسمی را از قسمتی از اتاق به قسمت‌ای دیگر ببرد، می‌توان چنین زیر–وظیفه‌هایی را تعیین کرد: " حرکت به سمت مکان‌ای مشخص"، " برداشتن جسم"، "قرار دادن جسم" و "دور زدن موانع". عامل در بالاترین سطح تصمیم‌گیری با توجه به حالت فعلی‌ی عامل به‌ترین زیر–وظیفه را انتخاب می‌کند. هر کدام از زیر–وظیفه‌ها خود نیز می‌توانند زیر–وظیفه‌های دیگری را فراخوانی کنند. به مانند options، روش MaxQ را می‌توان به صورت SMDP فرمول‌بندی کرد.

روش MaxQ شباهت‌هایی با روش یادگیری‌ی ساختار و رفتاری که در این پژوهش معرفی می‌شود دارد. شباهت دو روش در این است که هر دوی آن‌ها ارزش[۴۸] هر حالت را به صورت ارزش اجزای ساده‌تری می‌نویسند: MaxQ ارزش کلی را بر حسب ارزش زیر–وظیفه‌ها می‌نویسد و روشِ من ارزش کلی را بر حسب ارزش رفتارها و لایه‌ها می‌نویسد (یادگیری ساختار). این موضوع پس از توضیح روش واضح خواهد شد. از دیدی دیگر می‌توان آن را مشابه با روش یادگیری رفتار دانست: در هر دو روش تخمین‌ای از ارزش زیر–وظیفه‌های پایین‌تر (لایه‌های پایین‌تر در یادگیری رفتار) در هر تصمیم‌گیر نگه‌داری می‌شود. با این‌همه روش MaxQ روش‌ای است که ساختار در آن ثابت است و یادگیری‌ای بر روی ساختار انجام نمی‌شود. در حالی که در روش‌ای که معرفی خواهم کرد ساختار تغییرپذیر خواهد بود. هم‌چنین روش پیش‌نهادی –همان‌طور که در توضیحات Feudal RL گفته شد– روشی رفتارگرا است در حالی که MaxQ این‌گونه نیست. در نتیجه اگر بخشی از معماری‌ی MaxQ برداشته شود، عامل دیگر قادر به ادامه‌ی کار نیست در حالی که با برداشته شدن یا خراب شدن بخشی از معماری‌ی رفتارگرا، عامل هم‌چنان به وظیفه‌ی خود عمل می‌کند – گیریم با بازده پایین‌تر.

فرض کنید که کنترلر عامل‌ای به صورت یک برنامه با زبان برنامه‌نویسی‌ی متداولی –مانند Lisp– یا یک Finite State Machine (FSM) توصیف شده باشد. بدیهی است که تصمیم‌هایی که عامل اخذ می‌کند بستگی به برنامه‌ای دارد که کنترلر آن عامل را توصیف می‌کند و آن برنامه نیز به

[48] Value

صورت از پیش توسط برنامه‌نویس مشخص شده است. به عنوان نمونه، طراح تشخیص می‌دهد که برای پوشش دادن[۴۹] کلِ یک محوطه می‌بایست همیشه به موازات دیوار سمت راست حرکت کرد و البته گاهی –که با احتمالی مشخص می‌شود– از دیوار کنده شد تا به دیوار دیگری رسید. با وجود سادگی و موثر بودن این روش، عیب آن تطبیق‌ناپذیر بودن کنترلر طراحی‌شده است. ممکن است سیاست دیگری به عامل‌ای با بازده بالاتر منجر شود و یا این‌که طراحی‌ای که در محیط قدیم صورت گرفته دیگر چندان مناسب محیط تغییر یافته نباشد. ایده‌ی دو روش Hierarchies of Abstract Machines (HAM) و ALisp بر افزودن قابلیت یادگیری بر کنترلری این‌چنین توصیف شده است.

در HAM از توصیف FSMای استفاده می‌شود که در گره‌های انتخاب‌اش[۵۰] امکان یادگیری اضافه شده است ([Parr98a] و [Parr98b]). عامل با توجه به سیگنال تقویت به‌ترین انتخاب از بین چندین انتخاب ممکن در هر چنان گره‌ای را یاد می‌گیرد. این روش که تئوری‌ی ریاضی‌اش –درست مانند options و MaxQ– بر پایه‌ی SMDP است با مقید کردن سیاست‌های تحقق‌پذیر، ابعاد فضای فرضیه[۵۱،۵۲] را کوچک می‌کند. این کاهش ابعاد فضا باعث سریع‌تر شدن یادگیری می‌شود. ALisp گسترش‌ای از HAM است که در آن برخلاف HAM کنترلر با زبان‌ای سطح بالا مانند Lisp توصیف می‌شود ([Andre02] و [Andre03]). گفتنی است روش ALisp از تجزیه تابع ارزش‌ای مشابه با MaxQ استفاده می‌کند.

همان‌طور که اِعمال ساختار (برنامه) از پیش مشخص برای FSM (Lisp) باعث کاهش قابل ملاحظه‌ی ابعاد فضای فرضیه می‌شود، به همین صورت مقید کردن نتیجه به سیاست‌های محدود نیز ممکن است باعث پایین آمدن بازده نهایی عامل شود. اگر دانش اولیه‌ی طراح دقیق نباشد و ساختاری (برنامه‌ای) نه چندان خوب طراحی کند وجود یادگیری در نقاط انتخاب نیز باعث رسیدن به بیشینه‌ی بازده ممکن نخواهد شد. به عبارت دیگر بهینه‌گی‌ی سیاست نهایی کاملا به دانش طراح از مساله باز می‌گردد. این خانواده‌ی روش‌ها برخلاف روش‌های یادگیری‌ای که در فصل‌های بعد پیش‌نهاد خواهد شد قابلیت

[49] Coverage
[50] Choice Node
[51] Hypothesis Space
[52] برای دیدن تعریف فضای فرضیه به کتاب‌های یادگیری ماشینی مانند [Mitchell97] یا [Vidyasagar03] مراجعه کنید

طراحی‌ی خودکار سلسله‌مراتب را ندارند و مشابه با یادگیری رفتار –بدون یادگیری ساختار– معرفی شده در این پایان‌نامه‌اند. قابل ذکر است که یادگیری رفتار پژوهش فعلی تا حدی مشابه با یادگیری‌ای است که در این دو روش مطرح می‌شود و به نظر می‌آید که این روش‌ها به دلیل قدرت توصیفی‌ی بالایی که دارند قادر به ایجاد معماری‌ای مشابه با معماری‌ای که در این پایان‌نامه به آن می‌پردازم هستند. البته هم‌چنان مشابه با قبل تفاوت در رفتارگرا بودن معماری‌ی این پژوهش با بقیه‌ی معماری‌ها و روش‌های یادگیری تقویتی سلسله مراتبی به قوت خود باقی است.

۲) معرفی مساله

در این فصل بستر ریاضی‌ی روش‌های مورد بحث در این پایان‌نامه را معرفی می‌کنم. با استفاده از این بستر ریاضی می‌توان به توسعه‌ی یادگیری ساختار، یادگیری رفتار و تکامل رفتار پرداخت.

فرض کنید که مجموعه‌ای از n رفتار مختلف در اختیار داریم. این مجموعه را با $\{B_i\}(i=1,...,n)$ نمایش می‌دهیم. هر رفتار را –که نگاشت‌ای بین ادراک عامل و عمل آن است– بدین‌گونه توصیف می‌کنیم:

$$
\begin{aligned}
&B_i : S_i' \to A_i' \qquad \mathrm{i} = 1,...,\mathrm{n} \\
&A_i' = A_i \cup \{\text{No Action}\} \\
&S_i' = \{s_i' | s_i' = M_i(s); \forall s \in S_i\} \\
&S_i \subset S, A_i \subset A \\
&M_i : S \to S_i'
\end{aligned}
\qquad (۲-۱)
$$

که در آن S بیان‌گر حالت کلی‌ی عامل است و A نیز مجموعه‌ی اعمالی است که عامل می‌تواند با استفاده از آن محرک‌های‌اش[1] را تغییر دهد. مثلا در یک روبات متحرک، S معادل اطلاعات مقادیر خوانده شده توسط فاصله‌یاب‌های سونار[2]، سنسورهای برخوردی[3]، اطلاعات دریافتی از دوربین، تخمین از مکان فعلی‌ی روبات و ... است. A نیز می‌تواند مجموعه‌ی دستورهای حرکتی‌ای باشد که به موتور چرخ‌ها داده می‌شود. در روابط بالا S_i زیرمجموعه‌ای از کل فضای حالت عامل است که توسط رفتار B_i رویت‌پذیر[4] است. در حالت کلی این زیرمجموعه برای رفتارهای مختلف یک‌سان نیست $(S_i \neq S_j)$ و البته هیچ دلیلی هم وجود ندارد که زیرفضایی که رفتارهای مختلف حس می‌کنند بدون اشتراکی با هم

[1] Actuators
[2] Sonar
[3] Bumper
[4] Observable

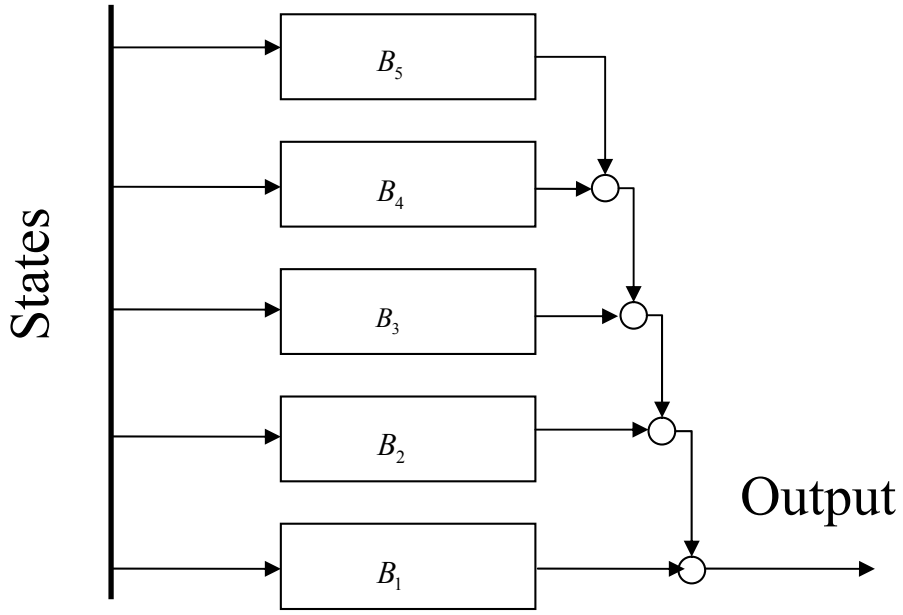
باشند ($S_i \cap S_j \neq \varnothing$). معمولا هر رفتار تنها به بخش کوچک‌ای از کل فضای حالت عامل دست‌رسی دارد. مثلا رفتاری که برای سریع واکنش دادن و جلوگیری از برخورد با اجسام ساخته شده است کافی است به سنسورهای برخوردی و فاصله‌یاب‌ها دست‌رسی داشته باشد و مثلا اطلاع از این‌که در فاصله‌ی دورْ جسم‌ای قرار دارد معمولا لازم نیست. مشابه با قبل A_i نیز مجموعه‌ی اعمال‌ای است که رفتار B_i می‌تواند انجام دهد؛ انتخاب عضوی از A_i باعث تغییر وضعیت یکی از خروجی‌های عامل می‌شود. M_i نیز نگاشت‌ای است که حالت کلی‌ی عامل را به حالت درونی‌ای که رفتار B_i برای تصمیم‌گیری‌های‌اش استفاده می‌کند تبدیل می‌کند. مثلا M_i می‌تواند معادل حذف تعدادی از حالت‌های S (با فرآیند تصویر کردن[5] به زیرفضایی از فضای اصلی) یا استخراج تعدادی ویژگی از حالت‌های اصلی باشد.

هر رفتار B_i با عمل‌ای فرضی به نام "عمل پوچ" (یا ناعمل[6]) که آن را با NA نشان خواهم داد توسعه یافته است. NA عمل‌ای است که حتی در صورت انتخاب نیز باعث تغییر محرک‌ها نمی‌شود[7].

برای این‌که ببینید یک رفتار چگونه عمل می‌کند، فرض کنید که عامل درحالت $s \in S$ است. هر کدام از رفتارهای B_i از طریق نگاشت M_i به زیرفضای نگاشت‌یافته‌ی S_i' از S دست‌رسی دارند. اگر $s_i \in S_i$ (یا $s_i' \in S_i'$ در بازنمایی[8] داخلی هر رفتار)، آن‌گاه رفتار B_i **تحریک**[9] می‌شود. حال اگر رفتار B_i یکی از اعمال حقیقی‌ی خود و نه NA را انتخاب کند، آن‌گاه به آن **فعال‌شده**[10] می‌گوییم: اگر $a_i = B_i(M_i(s_i)) = B_i(s_i') \neq NA$. اما در صورت‌ای که آن رفتار تحریک نشود و یا تحریک بشود ولی عمل NA را انتخاب کند، رفتار هیچ عمل‌ای را بر روی محرک‌های عامل انجام نداده است.

---

[5] Projection
[6] No Action
[7] دلیل وجود NA در فصل یادگیری (فصل ۴) رفتار مشخص می‌شود اما به طور خلاصه NA باعث می‌شود تا رفتار گاهی حتی در صورت تحریک شدن نیز خروجی‌ای ندهد و اجازه دهد تا رفتارهای پایین‌تر فعال شوند. بدین‌گونه رفتارهای پایین‌تر ممکن است بتوانند عمل‌ای را پیش‌نهاد دهد که نتیجه‌ی به‌تری برای عامل دارد و بازده بالا رود. این‌که در چه شرایطی می‌بایست چنین انتخاب‌ای صورت پذیرد در همان فصل بررسی می‌شود.
[8] Representation
[9] Excite
[10] Activated



شکل ۱-۲. نمونه‌ای از معماری Subsumption موازی

تا به حال به معرفی‌ی تک‌رفتار پرداخته بودم. هدف این پژوهش بررسی‌ی معماری‌های رفتارگرا است. معماری‌هایی که در آن تعدادی رفتار با کنش با هم رفتار کلی‌ی عامل را مشخص می‌کنند. همان‌طور که پیش‌تر گفته‌ام، هدف این پروژه خودکارسازی‌ی تولید معماری‌های رفتارگرای سلسله مراتبی است. به همین دلیل، معماری‌ی انتخاب‌شده حالت خاص‌ای از معماری‌ی متداول و مشهور *Subsumption* است. در این این حالت خاص که به آن معماری *Subsumption* کاملا موازی[11] می‌گوییم، همه‌ی رفتارها به صورت موازی هم قرار گرفته‌اند (شکل ۱-۲). رفتارهای که در لایه‌های بالایی معماری قرار دارند نسبت به رفتارهای لایه‌های پایین‌تر خود اولویت دارند. اولویت‌شان نیز بدین صورت است که در صورت فعال‌شدن دو رفتار B_1 و B_2 (به خاطر داشته باشید که فعال شدن رفتار معادل پیش‌نهاد خروجی دادن است)، رفتار بالاتر (مثلا B_2) کنترل عامل را در اختیار قرار می‌گیرد و اجازه‌ی در اختیار داشتن کنترل عامل را به رفتار B_1 نمی‌دهد و خود به تغییر محرک‌های عامل دست

[11] Purely Parallel SubSumption Architecture (PPSSA)

می‌یازد. در این حالت گفته می‌شود که B_2 رفتار **کنترل‌کننده**[۱۲] در معماری است و B_1 نیز توسط B_2 **سرکوب**[۱۳] شده است.

اینک می‌خواهم نمایش لازم برای معرفی‌ی یک معماری PPSSA را معرفی کنم. فرض کنید که n رفتار $\{B_i\}$ داریم و قصد بیان معماری‌ی m–رفتاری T را که شامل m تا از کل رفتارها است داریم ($m \le n$). $T(i)$ نمایان‌گر رفتار لایه‌ی iام معماری T است. در این نوشتار، شماره‌گذاری از پایین‌ترین لایه آغاز می‌شود: $T(1)$ پایین‌ترین رفتار معماری است و $T(m)$ بالاترینِ آن. به عبارت دیگر داریم:

$$T = [B_{index(1)}\ B_{index(2)}\ \dots\ B_{index(m)}]^{\mathrm{T}} \quad m \le n$$
$$index(i): j \quad (\text{that indicates } B_j \text{ is in the } i^{th} \text{ layer}) \qquad (۲-۲)$$

پس مثلا معماری‌ی

$$T = [Wandering \quad BallCollection \quad ObstacleAvoidance] \qquad (۲-۳)$$

بیان‌گر ساختار معماری‌ی عامل‌ای است که در پایین‌ترین و کم اولویت‌ترین لایه رفتار پرسه‌زدن قرار گرفته است و در بالاترین و پر اولویت‌ترین لایه رفتار جلوگیری از برخورد.

اکنون به مساله‌ی طراحی خودکار عامل هوش‌مند می‌پردازم. همان‌گونه که پیش‌تر گفتم این مساله معادل با پیدا کردن طرحی‌ای است که معیار مشخص شده توسط طراح را بهینه کند. برای یافتن این بهینه‌گی باید جستجویی در فضای طرح‌های ممکن انجام داد. هدف طراحی نیز پیدا کردن عضوی از فضای بزرگ طرح‌ها است که آن معیار عینی را بهینه کند. پس برای عملی کردن این کار، می‌بایست هر طرح را به صورت پارامترهایی در فضای پارامترها توصیف کرد. از دیدگاهی دو مساله مطرح می‌شود:

[12] Controlling
[13] Suppress

- هر طرح ممکن به چه صورت بازنمایی می‌شود؟
- جستجو بین طرح‌های مختلف چگونه انجام می‌شود؟

سوال اول به این موضوع می‌پردازد که پاسخ مساله به چه صورت باید توصیف یا بازنمایی شود. به عنوان مثال در بسیاری از موارد از شبکه‌ی عصبی‌ای به عنوان کنترلر عامل هوش‌مند استفاده می‌کنند و در نتیجه فضای طرح‌ها که توسط پارامترهای شبکه‌ی عصبی –مثلا وزن‌های آن– مشخص می‌شود معادل همه‌ی شبکه عصبی‌هایی است که با استفاده از معماری‌ی آن شبکه قابل ایجاد است. در این صورت هدف نیز پیدا کردن مجموعه‌ای از وزن‌ها (و یا پارامترهای دیگر) است تا معیار مورد نظر بهینه شود. در این حالت بازنمایی به صورت شبکه‌ی عصبی بوده است.

برای جستجو بین طرح‌های گوناگون روش‌های مختلفی وجود دارد. در این پژوهش از دو روش بهینه‌سازی‌ی یادگیری تقویتی و تکامل مصنوعی (یا تکامل به طور خلاصه) استفاده می‌کنم. البته فرض‌های پشت هر کدام از این روش‌ها آن‌قدر متفاوت است که نگاه کردن به آن دو تنها به عنوان روش‌ای برای جستجو، ساده‌سازی‌ی نادقیقی است.

ابتدا به یادگیری تقویتی و کاربردش در این پژوهش می‌پردازم: یادگیری تقویتی یکی از روش‌های بسیار متداول برای تطبیق موجود هوش‌مند با محیط است. یادگیری یکی از مشخصه‌های عامل هوش‌مند[14] است. عامل هوش‌مند می‌بایست برای فعالیت در محیط[15] ناشناس و تغییرپذیر با زمان خود را با رخدادهای ناشناخته و هم‌چنین تغییرات محیط تطبیق دهد. چنین عامل‌ای می‌باید با توجه به رفتارش در محیط و پاداش یا تنبیه‌ای که از طرف محیط (یا از امیال درونی‌اش) دریافت می‌کند در مورد رفتارهای آینده‌اش تصمیم‌گیری کند. چارچوب یادگیری تقویتی –یادگیری‌ای که عامل توسط سیگنال تقویت رفتار آتی‌اش را مشخص می‌کند– مدت‌هاست که در روان‌شناسی به عنوان توصیف‌گر یادگیری موجودات مطرح شده است [Hergenhahn01]. خوش‌بختانه این دیدگاه دارای فرمول‌بندی‌ی ریاضی‌ی دقیقی نیز هست و در سال‌های اخیر تلاش‌های بسیاری برای به کارگیری‌ی آن در عامل‌های هوش‌مند

[14] Intelligent Agent
[15] Environment

ماشینی شده است ([Sutton88]، [Watkins89]، [Watkins92]، [Jaakkola94]، [Singh00]). برای اطلاعات بیش‌تر و توضیح کلی در مورد این روش‌ها به [Kaelbling96] و [Sutton98] و هم‌چنین منابع معرفی شده در آن‌ها مراجعه کنید. برای توصیف روش‌های متداول یادگیری تقویتی سلسله مراتبی –که شباهت‌هایی با نتایج یادگیری این پژوهش دارند– به بخش یادگیری تقویتی سلسله مراتبی (فصل ۱) نگاه کنید.

اگر r_t را به عنوان سیگنال تقویت[۱۶] دریافتی (پاداش یا تنبیه) در زمان t بدانیم (که می‌تواند به حالت سیستم، عمل انتخابی، حالت جدید و ... بستگی داشته باشد)، آن‌گاه می‌توان پاداش بازگشته[۱۷،۱۸] را به یکی از صورت‌های زیر نوشت:

$$R = \frac{1}{N}\sum_{i=1}^{N} r_i \qquad (۲-۴)$$

$$R = \sum_{i=1}^{\infty} \gamma^i r_i \quad (0 \le \gamma < 1) \qquad (۲-۵)$$

$$R = \lim_{N\to\infty}\left(\frac{1}{N}\sum_{i=1}^{N} r_i\right) \qquad (۲-۶)$$

که در این‌جا تنها به مورد اول یعنی میزان پاداش در یک اپیزود می‌پردازم. در این صورت می‌توان ارزش کل معماری با ساختار T و مجموعه رفتارهای $\{B_i\}\ (i=1,...,n)$ را بدین صورت نوشت:

$$V_T = \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \middle| \text{the agent with structure } T \text{ and set of behaviors } \{B_i\}(i=1,...,n)\right]$$
$$= \underset{\pi}{E}\left[R \middle| \text{the agent with structure } T \text{ and set of behaviors } \{B_i\}(i=1,...,n)\right]$$

(۲-۷)

[16] Reinforcement Signal
[17] Return
[18] انتخاب پاداش بازگشته به جای *بازگشت* به دلیل جلوگیری از گنگ شدن کلمه‌ی متداول *بازگشت* بوده است.

که امید ریاضی بر روی همه‌ی حالت‌های سیستم در طول دوره‌ی زندگی‌ی عامل حساب می‌شود و توزیع احتمال آن نیز بر اساس سیاست عامل (π) و توزیع احتمال انتقال حالت[19] مشخص می‌شود. همان‌طور که بعدتر خواهید دید، سیاست عامل به سیاست هر کدام از رفتارها و ساختار آن رفتارها در معماری عامل بستگی دارد و در نتیجه این سیاستِ کل، تابع‌ای از سیاست‌های جزیی‌تر است. در جاهای دیگر این پایان‌نامه ممکن است این ارتباط امید به سیاست (π) به طور صریح ذکر نشود.

هدف یادگیری، بیشینه‌گی V_T است. چنین کاری را در دو سطح می‌توان انجام داد: *سطح ساختار* و *سطح رفتار*.

در یادگیری ساختار معماری فرض می‌کنیم که مجموعه‌ای مناسب از رفتارها ($\{B_i\}$ $(i=1,...,n)$) وجود دارد و هدف انتخاب m رفتار از n رفتار موجود و مرتب کردن آن‌ها در معماری‌ای (که به صورت T نمایش داده می‌شود) است به صورتی که (۷-۲) بیشینه شود:

$$T^* = \arg\max_T V_T . \qquad (۲-۸)$$

فرض وجود مجموعه‌ی رفتارهای مناسب چندان غیرواقعی نیست. در بسیار از موارد طراح از این‌که برای حاصل شدن رفتار مورد نظر، چه رفتارهای ساده‌تری لازم‌اند تا حدی آگاه است ولی نمی‌داند هر کدام از آن رفتارها چه اولویتی می‌بایست داشته باشند. پس او می‌تواند مجموعه‌ای از رفتارها را طراحی کند و آن را به صورت جعبه‌ابزاری در اختیار عامل یادگیرنده قرار دهد. این مساله در فصل ۳ بررسی می‌شود.

از سویی دیگر، مساله‌ی یادگیری رفتارها —به عنوان دیگر پارامتر توصیف‌کننده معماری— مطرح می‌شود. هدف یادگیری رفتار پیدا کردن نگاشت‌ای بین فضای حس‌شونده توسط هر کدام از رفتارها (S'_i) به فضای اعمال ممکن برای آن رفتار (A'_i) است. به عبارت دیگر عامل می‌بایست با فرض وجود ساختاری ثابت (ترتیب رفتارها در معماری) یاد بگیرد که به ازای هر s_i (یا s'_i از دید هر رفتار) چه a^*_i را انتخاب کند تا میزان پاداش دریافتی‌اش بیشینه شود:

[19] State Transition Probability

$$\{B_i^*\} = \arg\max_{B_i} V_T \,. \qquad (۲-۹)$$

مطابق آن‌چه در یادگیری تقویتی موسوم است می‌بایست بازنمایی‌ای برای نگاه‌داری اطلاعات حاصل از تعامل با محیط وجود داشته باشد. این بازنمایی می‌بایست نشان دهد که ارزش انجام هر عمل در هر حالت ممکن چیست. برای انجام این کار برای هر رفتار تابع ارزش حالت–عمل را تعریف می‌کنیم:

$$Q_i(s_i', a_i), s_i' \in S_i', a_i \in A_i' \quad \forall B_i \in \{B_i\} \qquad (۲-۱۰)$$

هم‌چنین نگاشت زیر بین آن‌چه هر رفتار حس می‌کند و عمل‌اش وجود دارد:

$$B_i(s_i') = \text{Action Selection Mechanism}(Q_i(s_i', \cdot)) \qquad (۲-۱۱)$$

که در هر حالت انتخاب عمل بهینه بدین‌صورت خواهد بود:

$$B_i(s_i') = \arg\max_{a_i} Q_i(s_i', a_i) \,. \qquad (۲-۱۲)$$

ذکر دو نکته مفید است: نخست آن‌که در بازنمایی بالا، تابع ارزش–حالت برای مجموعه اعمال توسعه‌یافته (A_i') تعریف می‌شود و نه فقط اعمال ممکن. دلیل آن نیز –همان‌طور که پیش‌تر به طور مختصر گفته شد و جلوتر دقیق‌تر بررسی خواهد شد– اجازه دادن به رفتار برای فعال نشدن با وجود تحریک شدن است. با این‌کار رفتارهای زیرین که ممکن است گزینه‌های به‌تری برای کنترل عامل باشند اجازه کنترل‌کننده بودن می‌یابند. نکته‌ی دوم این‌که انتخاب عمل معمولا دارای اندکی رفتار

کاوش‌گرانه[20] است و در نتیجه (۷) معمولا گزینه‌ی به‌کارگرفته شده نیست و به جای‌اش از مکانیزم‌های انتخاب عمل‌ای مانند ε-greedy یا انتخاب عمل بولتزمان[21] استفاده می‌شود.

می‌توان این دو نوع یادگیری را با هم به کار گرفت ؛ جستجوی طرح بهینه را در هر دو فضای پارامترهای رفتار و پارامترهای ساختار انجام داد. در این صورت ، هدف یادگیری به شکل زیر بیان‌پذیر است:

$$\left\{T^*, B_i^*\right\} = \arg\max_{T,B_i} V_T . \qquad (۲-۱۳)$$

یادگیری رفتار و یادگیری هم‌زمان رفتار و ساختار در فصل ۴ مورد مطالعه قرار می‌گیرد.

استفاده از تکامل مصنوعی نیز راه دیگری برای ایجاد کنترلرهای مناسب است. تکامل مصنوعی که الهام گرفته از فرآیند تکامل طبیعی و اصل انتخاب طبیعی است با ایجاد جمعیت‌ای از راه‌حل‌های ممکن برای مساله‌ی مورد نظر و مقایسه‌ی کیفیت هر کدام از راه‌حل‌ها با هم و استفاده از اپراتورهای تکاملی‌ای مانند تزویج[22] و جهش[23] به پاسخ مطلوب نزدیک می‌شود. در این‌جا چندان وارد بحث تکامل نمی‌شوم و خواننده را به نوشته‌های کلاسیکی چون [Goldberg89] ارجاع می‌دهم.

در این پژوهش، از تکامل برای ایجاد رفتارهای مطلوب استفاده شده است (فصل ۵). به عبارت دیگر برای یافتن رفتارهای بهینه اضافه بر استفاده از یادگیری، از تکامل نیز استفاده شده است. بازنمایی رفتارها با تغییرات اندکی مشابه با آن چیزی است که در چند پاراگراف پیش‌تر در بخش یادگیری رفتار توضیح دادم. اما در این‌جا دیگر تابع ارزش–عمل تعریف نمی‌شود و به طور مستقیم سیاست هر رفتار بازنمایی می‌شود. به عبارت دقیق‌تر هر رفتار B_i را به شکل نگاشت‌ای بین ورودی و خروجی‌ی آن رفتار مشخص می‌کنم:

[20] Exploration
[21] Boltzman Action Selection
[22] Crossover
[23] Mutation

$$B_i(s') : S'_i \rightarrow A'_i \text{.} \qquad (۲-۱۴)$$

خروجی این نگاشت از میان مجموعه‌ی توسعه‌یافته‌ی اعمال (A'_i) انتخاب می‌شود. در صورتی که عمل انتخابی NA بوده باشد، رفتار فعال نخواهد شد و به رفتارهای پایینی –در صورت وجود– اجازه‌ی فعال‌شدن می‌دهد. در پیاده‌سازی انجام شده این نگاشت به صورت جدول ارجاعی[۲۴] با ابعاد مناسب بازنمایی می‌شود در حالی که هیچ محدودیت ویژه‌ای برای نوع بازنمایی وجود ندارد.

در نهایت لازم به تاکید است که جستجوهای انجام شده سعی در پیدا کردن بهینه‌ی مقید می‌کنند. مثلا به هنگام حل مساله‌ی یادگیری ساختار، جستجو در فضای مقید شده به رفتارهای موجود انجام می‌شود – در نتیجه بدیهی است که مساله ممکن است پاسخ‌های به‌تری نیز داشته باشد.

[24] Lookup Table

۳) یادگیری ساختار

۳-۱) مقدمه

در دو فصل پیش درباره‌ی روی‌کرد کلی‌ی این پژوهش به فرآیند خودکارسازی طراحی‌ی معماری‌ی رفتارگرا توضیح دادم. در این فصل دو روش متفاوت یادگیری ساختار برای خودکارسازی‌ی طراحی را معرفی می‌کنم[1].

هدف یادگیری ساختار –همان‌طور که پیش‌تر گفته شد– پیدا کردن ساختار T^*ای است که (۷-۲) را بهینه کند. در یادگیری ساختار فرض بر آن است که مجموعه‌ای از رفتارها از-پیش در اختیار است و هدف پیدا کردن به‌ترین انتخاب و ترتیب از آن‌ها است. یک روش مناسب یادگیری ساختار می‌بایست به سوال‌های زیر پاسخ دهد:

- **بازنمایی:** عامل چگونه می‌بایست دانش کسب شده در دوره‌ی حیات‌اش را ذخیره کند؟
- **تخصیص اعتبار در ساختار سلسله مراتبی:** عامل دارای ساختار سلسله مراتبی چگونه می‌بایست رفتار مسوول در عمل‌کردش را تشخیص داده و به تناسب آن را تشویق یا تنبیه کند؟
- **به روز رسانی دانش:** عامل در مواجهه با تجربه‌ای جدید چگونه می‌بایست دانش قبلی‌اش را به روز رساند؟

قدم اول انتخاب بازنمایی‌ی مناسب‌ای از دانش‌ای است که عامل در طول زندگی‌اش کسب کرده. این بازنمایی می‌بایست آینه‌ی تمام‌نمایی از تجربیات عامل باشد. عامل به کمک این بازنمایی است که قادر به تصمیم‌گیری درباره‌ی به‌ترین ساختار ممکن می‌شود. به عبارت دیگر این بازنمایی باید به عامل اجازه دهد تا مقادیر مختلف V_T را محاسبه کرده و ساختاری را که بالاترین مقدار آن را پیش‌بینی می‌کند انتخاب کند. این بازنمایی لازم است چند ویژگی‌ی مختلف داشته باشد: در ابتدا این

[1] نتایج مقدماتی‌ی این بخش در [Farahmand04] منتشر شده است.

بازنمایی می‌بایست کامل باشد و بتواند همه‌ی حالت‌های ممکن ساختار را توصیف کند. مثلا فرض کنید تعداد n رفتار در دست‌رس است و ما می‌خواهیم معماری‌ای با m رفتار ایجاد کنیم. یک نوع بازنمایی می‌تواند این باشد که عامل امتیازی را که هر رفتار در طول دوره‌ی زندگی‌اش کسب کرده ثبت کند و در مرحله‌ی انتخاب معماری‌ی جدید با توجه به تجربیات پیشین‌اش m رفتار با امتیاز بالاتر را انتخاب کند. این روند باعث ایجاد معماری‌ای می‌شود که به احتمال بالا باعث دریافت متوسط پاداش بیش‌تری در طول عمر عامل نسبت به حالت انتخاب تصادفی‌ی معماری‌ها می‌شود. اما این معماری از بین تعدادی محدود از معماری‌های ممکن انتخاب شده است و فضای جستجوی آن بسیار کوچک‌تر از فضای n^m (انتخاب n رفتار برای هر کدام از m لایه) و یا $n!/(n-m)!$ (در نظرگیرنده‌ی ترتیب و بدون تکرار) معماری ممکن است. در نتیجه بسیاری از معماری‌ها در این فضا قابل توصیف نیستند و اولویت رفتارها نسبت به هم به هیچ صورت‌ای مشخص نمی‌شود.

به عنوان مثال‌ای از بازنمایی‌ی کامل می‌توان به بازنمایی‌ای اشاره کرد که در آن امید ریاضی پاداش (یا امتیاز) n^m معماری‌ی ممکن ذخیره شده است. این بازنمایی کامل است ولی دارای فضای بسیار بزرگی است. عامل برای آن‌که تصمیم بگیرد کدام رفتارها را و با چه ترتیب‌ای انتخاب کند، می‌بایست تجربیات بسیاری زیادی را انجام دهد در حالی که بسیاری از این تجربه‌ها لازم نیستند. این بازنمایی خاص –ولی کامل– استفاده‌ای از ارتباط احتمالی‌ی بین تجربه‌های دو معماری‌ی مختلف نمی‌کند و آن‌ها را از نظر منطقی کاملا مستقل از هم می‌داند در حالی که به عنوان مثال عامل با تجربه کردن این‌که یک جفت رفتار وقتی ترتیب خاص‌ای داشته باشند همیشه باعث پاداش گرفتن می‌شوند می‌تواند فضای جستجوی‌اش را بسیار کوچک کند. در واقع بازنمایی‌ای مطلوب است که عامل را قادر به تعمیم[۲] دانش‌اش بکند؛ گو این‌که این تعمیم ممکن است همیشه درست نباشد. پس به طور خلاصه ویژگی‌های یک بازنمایی خوب این‌های‌اند:

- کامل[۳] باشد.

[2] Generalization
[3] Complete

- فضای بازنمایی‌ی[۴] کوچک‌ای داشته باشد[۵].
- از تجربیات، هوش‌مندانه بهره ببرد.

مساله‌ی مهم دیگر برای یادگیری ساختار، تخصیص اعتبار در ساختار سلسله مراتبی‌ی معماری است. چگونگی‌ی اعمال نتیجه‌ی این تخصیص اعتبار بر دانش ذخیره شده در بازنمایی، مساله‌ی به روزرسانی‌ی بازنمایی را ایجاد خواهد کرد. انتخاب مناسب نحوه‌ی بازنمایی علاوه بر ویژگی‌های ذکر شده پیشین حل مساله‌ی تخصیص اعتبار سلسله مراتبی را ساده می‌کند.

برای حل این سه مساله از ساختار ویژه‌ی معماری Subsumption موازی نهایت استفاده را می‌برم: یعنی در انتخاب نوع بازنمایی به ساختار معماری توجه می‌کنم. برای این کار تلاش می‌کنم تا ساختار را به اجزای ساده‌تری تجزیه[۶] کنم. این تجزیه می‌بایست به گونه‌ای باشد که بازنمایی حاصل شده سه ویژگی‌ی ذکر شده در بالا را داشته باشد. سپس با روی‌کردی مرحله به مرحله ابتدا بازنمایی‌ی مناسب‌ای را برای ذخیره‌سازی‌ی دانش عامل پیش‌نهاد می‌کنم و سپس با توجه به آن بازنمایی نحوه‌ی تخصیص اعتبار و به‌روز رسانی‌ی دانش را مشخص می‌کنم.

در بخش‌های بعدی دو روش مختلف برای یادگیری ساختار توضیح داده خواهد شد.

۳-۲) یادگیری ساختار با بازنمایی مرتبه صفر

۳-۲-۱) بازنمایی

برای ذخیره‌سازی‌ی دانش عامل نسبت به محیط اطراف و تشخیص این‌که چه ترتیب‌ای از رفتارها مناسب‌تر است، از بازنمایی‌ای استفاده می‌کنیم که دانش ما را نسبت به اجزای ساده‌تری از آن معماری

---

[4] Representation Space

[5] در ادبیات یادگیری ماشینی استفاده از فضای فرضیه (hypothesis space) برای چنین تعریفی متداول است. با این‌حال برای تاکید بر کارکرد بازنمایانه‌ی این فضا، بیش‌تر از از فضای بازنمایی (representation space) استفاده می‌کنم. برای اطلاع بیش‌تر راجع به تعریف فضای فرضیه می‌توانید به [Mitchell97] و [Vidyasagar03] مراجعه کنید.

[6] Decompose

ثبت کند. بازنمایی‌ی مرتبه صفر –که در این بخش معرفی می‌شود– چنین وظیفه‌ای را بر عهده دارد. در این بازنمایی، ***امید ارزش دریافتی‌ی هر رفتار در هر لایه*** ذخیره می‌شود.

برای روشن شدن ایده‌ی این بازنمایی، فرض کنید که مجموعه‌ای دو تایی از رفتارها ($\{B_1, B_2\}$) داشته باشیم و قرار باشد معماری‌ای دو لایه از آن‌ها بسازیم، آن‌گاه –همان‌طور که نشان خواهم داد– می‌توان با ثبت ارزش‌های زیر، معماری‌ی بهینه را پیدا کرد. فرض کنید که ارزش قرارگیری‌ی هر رفتار در هر لایه –که به زودی تعریف دقیق‌ای از آن ارایه خواهم داد– بدین صورت باشد:

	B_1	B_2
L_1	1	-1
L_2	0.25	0.5

آن‌گاه ارزش معماری‌های ممکن غیرتکراری بدین‌گونه خواهد بود:

$$V_{[B_1 \quad B_2]} = 1 + 0.5 = 1.5 \qquad (۳-۱)$$

$$V_{[B_2 \quad B_1]} = 0.25 - 1 = -0.75 \qquad (۳-۲)$$

پس ترتیب مناسب $T = [B_1 \quad B_2]$ خواهد بود. برای حالت‌های کلی نیز با روندی مشابه می‌توان معماری‌ای یافت که باعث ایجاد با ارزش‌ترین معماری شود (مساله مطرح در رابطه‌ی (۸-۲)). به بیان ریاضی، برای معماری‌ی T می‌توان نوشت:

$$
\begin{aligned}
V_T &= \underset{\pi}{E}[R] = \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t\right] \\
&= \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N}\{r_t \wedge ("L_1 \text{ is controlling}" \vee "L_2 \text{ is controlling}" \vee \ldots \vee "L_m \text{ is controlling}")\}\right] \\
&= \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N}\{r_t \wedge "L_1 \text{ is controlling}"\}\right] + \ldots + \underset{\pi}{E}\left[\sum_{t=1}^{N}\{r_t \wedge "L_m \text{ is controlling}"\}\right] \\
&= \underset{\pi}{E}\left[\frac{1}{N}\sum r_t \mid L_1 \text{ is controlling}\right] \cdot P(L_1 \text{ is controlling}) \\
&+ \underset{\pi}{E}\left[\frac{1}{N}\sum r_t \mid L_2 \text{ is controlling}\right] \cdot P(L_2 \text{ is controlling}) \\
&+ \ldots + \underset{\pi}{E}\left[\frac{1}{N}\sum r_t \mid L_m \text{ is controlling}\right] \cdot P(L_m \text{ is controlling})
\end{aligned}
$$

(۳-۳)

که در آن $\underset{\pi}{E}\left[\frac{1}{N}\sum r_t \mid L_i \text{ is controlling}\right]$ امید ریاضی‌ی دریافت سیگنال تقویت در آن هنگامی است که لایه‌ی iام کنترل عامل را در اختیار گرفته است. $P(L_i \text{ is controlling})$ نیز احتمال در کنترل لایه‌ی iام بودن عامل است. توجه کنید که این تجزیه به دلیل دو به دو مجزا بودن[۷] بودن وقایع $\{t \mid "L_i \text{ is controlling}" \text{ in timestep } t\}$ ممکن شده: در هر زمان فقط یکی از لایه‌ها کنترل عامل را در اختیار دارد. در این فرمول‌بندی فرض شده است که حداقل یکی از لایه‌ها کنترل عامل را در اختیار گرفته‌اند.

با تعریف $V_{ZO}(i,j)$ –ارزش مرتبه صفر– به صورت زیر

$$
V_{ZO}(i,j) = V_{ij} = \underset{\pi}{E}\left[\frac{1}{N}\sum r_t \middle| B_j \text{ is the controlling behavior in the } i^{th}\text{layer}\right] \qquad (۳-۴)
$$

خواهیم داشت:

[7] Mutually Exclusive

$$\underset{\pi}{E}[\frac{1}{N}\sum r_t \mid L_i \text{ is controlling}] = \sum_{j=1}^{n} P\{B_j \mid L_i\} \underset{\pi}{E}\left[\frac{1}{N}\sum r_t \middle| B_j \text{ is the controlling behavior in } L_i\right]$$
$$= \sum_{j=1}^{n} P\{B_j \mid L_i\} V_{ij} \quad i = 1,...,m$$

(۳-۵)

که در آن $P\{B_j \mid L_i\}$ احتمال در اختیار B_j بودن کنترل عامل است مشروط به این‌که کنترل عامل در اختیار لایه‌ی L_i باشد. با در نظر گرفتن روابط به دست آمده، تابع ارزش کل معماری را می‌توان به صورت زیر نوشت:

$$V_T = \sum_{i=1}^{m}\sum_{j=1}^{n} P\{B_j \mid L_i\} V_{ij} P(L_i \text{ is controlling}). \qquad (۳-۶)$$

مشخص است که این بازنمایی *کامل* است چون قادر به بازنمایی همه‌ی ترکیب‌های ممکن از رفتارها است. از طرف دیگر ابعاد فضای بازنمایی در حالت بازنمایی مرتبه صفر بسیار کوچک‌تر از فضای بازنمایی‌ی کامل‌ای مانند آن‌که همه‌ی ترکیب‌های ممکن را ذخیره می‌کند است. ابعاد آن فضا برابر با مقدار زیر است:

$$\text{cardinality(ZO)} = n \cdot m . \qquad (۳-۷)$$

با داشتن ارزش هر رفتار در هر لایه می‌توان ارزش معماری‌های مختلف را محاسبه کرد و دیگر نیازی به تخمین جداگانه‌ی ارزش هر معماری نیست. این ویژگی باعث می‌شود بدون نیاز به تخمین همه‌ی معماری‌های مختلف، تخمین‌ای از ارزشِ هر معماری با داشتن تخمین‌ای از ارزش هر رفتار در هر لایه به دست آوریم. بدین‌گونه می‌توان با مقایسه‌ی چندین ساختار فرضی‌ی مختلف، به‌ترین آن‌ها را یافت. با توجه به روابط (۸-۲) و (۶-۳)، برای یافتن ساختار بهینه کافی است مساله‌ی زیر حل شود:

$$T^* = \arg\max_T V_T = \arg\max_T \sum_{i=1}^{m}\sum_{j=1}^{n} P\{B_j \mid L_i\} V_{ij} P(L_i \text{ is controlling}). \qquad (۳-۸)$$

برای حل این مساله می‌بایست تخمین‌ای از هر کدام از اجرای آن یعنی V_{ij}، $P(L_i \text{ is controlling})$ و $P\{B_j \mid L_i\}$ وجود داشته باشد. در زیربخش بعد نشان خواهم داد که همه‌ی این مقادیر در طول یادگیری قابل تخمین‌اند.

در یک معماری‌ی رفتارگرای متداول در هر لایه فقط یک رفتار وجود دارد. اما در معماری‌ی پیش‌نهادی می‌توان با احتمال‌های متفاوتی رفتارهای مختلف را در یک لایه‌ی ثابت انتخاب کرد. به عبارت دیگر یک معماری دیگر ساختاری ثابت نیست بلکه به وسیله‌ی $P\{B_j \mid L_i\}$هایی توصیف می‌شود که اجازه‌ی وجود چندین رفتار در یک لایه را به عامل می‌دهد.

حال اگر از چنین خاصیت‌ای استفاده نکنیم و همیشه تنها یکی از رفتارها را در هر لایه انتخاب کنیم —و به عبارت دیگر انتخاب معماری‌مان حریصانه[8] باشد– آن‌گاه داریم:

$$P\{B_j \mid L_i\} = \begin{cases} 1 & B_j \text{ is the controlling behavior in } L_i \\ 0 & \text{otherwise} \end{cases} \qquad (۳-۹)$$

و در نتیجه می‌توان نوشت:

$$T^* = \arg\max_T \sum_{i=1}^{m} V_{i,index(T(i))} P(L_i \text{ is controlling}). \qquad (۳-۱۰)$$

۳-۲-۲) تخصیص اعتبار و به روز رسانی بازنمایی

در زیربخش پیشین با بازنمایی مرتبه صفر دانش آشنا شدید. هنگامی که عامل پاداش یا تنبیه‌ای دریافت می‌کند اطلاعات ثبت شده در آن بازنمایی می‌بایست تغییر کند. مطابق با تعریف V_{ij} (۳-۴)، روش

[8] Greedy

تخصیص اعتبار مشخص است: *V_{ij} باید آن هنگام که B_j کنترل لایه‌ی iام را در اختیار داشته باشد به روز شود.* پس اگر لایه‌ی iام، *لایه‌ی کنترل کننده‌ی عامل باشد و* B_j در آن لایه فعال شده باشد و در آن هنگام –به دلیل عمل انتخاب شده توسط رفتار B_j در آن لایه– عامل سیگنال تقویت r_k را دریافت کند، ارزش V_{ij} مطابق با آن‌چه در تخمین پارامترهای تصادفی معمول است به روز می‌شود ([Robbins51] و [Watkins92]):

$$V_{ij_{k+1}} = (1-\alpha_{k,ij})V_{ij_k} + \alpha_{k,ij} r_k \quad \text{(whenever } B_j \text{ is the controlling behavior in the } i^{th} \text{ layer)} \qquad (۳-۱۱)$$

البته به دلیل آن‌که اثبات‌ای از جنس نشان دادن جمع‌شونده بودن[9] خطای بین V_{ij} و V_{ij}^* (V_{ij} بهینه – البته در صورت وجود) انجام نداده‌ام، نمی‌توان شروط هم‌گرایی‌ای مانند آن‌چه در [Jaakkola94] و موارد مشابه انجام شده است ارایه داد.

تخمین $P(L_i \text{ is controlling})$ و $P\{B_j \mid L_i\}$ پیچیده نیست. برای تخمین $P(L_i \text{ is controlling})$ می‌توان شمارنده‌ای برای هر لایه در نظر گرفت و هر باری که آن لایه کنترل‌کننده می‌شود آن را افزایش داد؛ یعنی:

$$P(L_i \text{ is controlling})\big|_{n+1} = \begin{cases} \dfrac{nP(L_i \text{ is controlling})\big|_n + 1}{n+1} & \text{if } L_i \text{ is controlling at time step } n \\ \dfrac{nP(L_i \text{ is controlling})\big|_n}{n+1} & \text{if } L_i \text{ is not controlling at time step } n \end{cases} \qquad (۳-۱۲)$$

یا این‌که از روش تخمین‌ای مبتنی بر روش روبینز–مونرو ([Robbins51]) –که در Q-Learning متداول است ([Watkins89] و [Watkins92])– استفاده کرد[10]:

[9] Contraction

$$P(L_i \text{ is controlling})\big|_{n+1} = (1-\alpha_{n,i})P(L_i \text{ is controlling})\big|_n + \alpha_{n,i}[\text{"}L_i \text{ is controlling in time step } n\text{"}]$$

(۳-۱۳) .

توجه کنید که این به روز رسانی در هر واحد زمانی‌ی عمومی‌ی عامل n انجام می‌شود و نه هر بار که رفتار فعال می‌شود. به روز رسانی‌ی $P\{B_j \mid L_i\}$ نیز مشابه است: یا از شمارنده‌ای استفاده می‌کنیم یا از روش تخمین روبینز-مونرو-مانند استفاده می‌کنیم:

$$P\{B_j \mid L_i\}\big|_{k+1} = (1-\alpha_{k,i})P\{B_j \mid L_i\}\big|_k + \alpha_{k,i}[\text{"}B_j \text{ is active in step } k\text{"}] \quad (\text{whenever } L_i \text{ is controlling})$$

(۳-۱۴)

و این به روز رسانی هر زمان که لایه‌ی L_i کنترل‌کننده است انجام می‌شود[11]. در صورت‌ای که $\{\alpha\}$ متناسب با $1/n$ (یا $1/k$ در (۳-۱۴)) کاهش یابد، این تخمین‌ها معادل تخمین متوسط —و با در نظرگیری‌ی کل پیشینه‌ی عامل با وزن یک‌سان- خواهند بود. با تغییر نحوه‌ی کاهش $\{\alpha\}$ می‌توان حافظه‌ی عامل نسبت به وقایع را نزدیک‌بین یا دوربین کرد. توجه کنید که مقادیر $\{\alpha_{k,ij}\}$، $\{\alpha_{n,i}\}$ و $\{\alpha_{k,i}\}$ الزاما یک‌سان نیستند.

روش دیگری برای تخمین هم‌زمان همه‌ی این مقادیر —یعنی V_{ij}، $P(L_i \text{ is controlling})$ و $P\{B_j \mid L_i\}$- وجود دارد. به جای تخمین هر کدام از اجزای (۳-۶) به طور مجزا، با تعریف $\tilde{V}_{ij}$ به صورت زیر می‌توان آن‌ها را هم‌زمان تخمین زد:

$$\tilde{V}_{ij} = P\{B_j \mid L_i\}V_{ij}P(L_i \text{ is controlling}). \qquad (۳-۱۵)$$

---

[10] $\text{"Expression"} = \begin{cases} 1 & \text{Expression is True} \\ 0 & \text{Expression is False} \end{cases}$

[11] به اختلاف بین این رابطه (۳-۱۴) و روابط به روز رسانی‌ی $P(L_i \text{ is controlling})$ ((۳-۱۲) و (۳-۱۳)) توجه کنید.

در آن صورت خواهیم داشت[۱۲]:

$$\widetilde{V}_{ij_{n+1}} = (1-\alpha_{n,ij})\widetilde{V}_{n,ij} + \alpha_{n,ij}\left[\text{" } B_j \text{ is active at time step } n\text{"}\times\text{" } L_i \text{ is controlling at time step } n\text{"}\times r_n\right] \quad (۳-۱۶)$$

در نتیجه با استفاده از یکی از این دو روش تخمین زدن مقادیر لازم برای (۶-۳) می‌توان آرایش بهینه‌ای از رفتارها را (که (۸-۲) را حل می‌کند) یافت. روش‌های مختلفی برای حل این مساله‌ی بهینه‌سازی ترکیبیاتی[۱۳] وجود دارد (استفاده از جستجوی کامل فضا[۱۴]، استفاده از انواع روش‌های سردستی و heuristicها، استفاده از meta-heuristicهایی مانند انجماد شبیه‌سازی‌شده[۱۵]، الگوریتم‌های تکاملی، یا شبکه‌های عصبی‌ی هاپفیلد[۱۶] و ...) اما در پیاده‌سازی‌های این پژوهش از جستجوی تصادفی[۱۷]ی تک نقطه‌ای فضا برای پیدا کردن بهینه آن مساله استفاده شده است.

۳-۳) یادگیری ساختار با بازنمایی مرتبه یک

۳-۳-۱) بازنمایی

در بازنمایی مرتبه صفر –که در بخش پیش معرفی شد– کمیت نگه‌دارنده‌ی دانش، ارزش قرارگیری‌ی هر رفتار در هر لایه بود. برخلاف آن در بازنمایی مرتبه یک، ارزش ترتیب رفتارها در معماری مورد توجه قرار می‌گیرد. در این بازنمایی، **امید ارزش دریافتی ترتیب‌های دو تایی رفتارها** در معماری ذخیره می‌شود. این بازنمایی کامل است و فضای بازنمایی کوچک‌تری نسبت به بازنمایی‌ای که همه‌ی حالت‌های ممکن یک معماری را در نظر می‌گیرد دارد. علاوه بر آن این بازنمایی از ساختار ویژه‌ی

[12] توجه به تفاوت بین این رابطه‌ی جدید (۱۶-۳) و رابطه‌ی (۱۱-۳) که در آن V_{ij} تنها هنگامی به روز می‌شد که B_j رفتار کنترل‌کننده در لایه‌ی iام بود ضروری است.

[13] Combinatorial Optimization Problem
[14] Exhaustive Search
[15] Simulated Annealing
[16] Hopfield
[17] Stochastic Search

معماری سلسله مراتبی‌مان که در آن رفتارها نسبت به هم ترتیب دارند –بالاتر یا پایین‌تر– بهره می‌گیرد و روند تقسیم امتیاز را راحت می‌کند. برای دقیق‌تر شدن بحث، ابتدا اپراتور ترتیب ‘$\overset{T}{\succ}$’ را بدین‌گونه تعریف می‌کنم:

$$B_i \overset{T}{\succ} B_j : B_i \text{ is upper than } B_j \text{ in the architecture } T\big|_{B_i,B_j \in T}. \qquad (۳-۱۷)$$

این اپراتور در معماری‌ای با ساختار T بیان‌گر بالا یا پایین بودن یک رفتار نسبت به رفتار دیگر است.

برای روشن شدن هدف پشت این بازنمایی، مثال ابتدای بخش بازنمایی مرتبه صفر را به گونه‌ای دیگر می‌نگریم. فرض کنید که مجموعه‌ای دو تایی از رفتارها ($\{B_1, B_2\}$) داشته باشیم و قرار باشد معماری‌ای دو لایه از آن‌ها بسازیم، آن‌گاه –همان‌طور که نشان خواهم داد– می‌توان با ثبت ارزش‌های زیر، معماری‌ی بهینه را پیدا کرد. فرض کنید که ارزش ترتیب‌های مختلف B_1 و B_2 این‌گونه باشند:

Order	Value
$B_1 > B_1$	1.25
$B_1 > B_2$	-0.75
$B_2 > B_1$	1.5
$B_2 > B_2$	-0.5

آن‌گاه به‌ترین انتخاب برای ترتیب دو رفتار، انتخاب $B_2 > B_1$ و یا معماری‌ی $T = [B_1 \quad B_2]$ خواهد بود.

همانند بخش پیش می‌توان ارزش کل عامل V_T را به صورت ارزش اجزای ساده‌تری تجزیه کرد:

$$V_T = \underset{\pi}{E}[\frac{1}{N}\sum_{t=1}^{N} r_t] = \sum_{i=1}^{m} \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \mid B_{index(i)[=k]} \text{ is controlling}\right] P(B_{index(i)} \text{ is controlling})$$

(۳-۱۸)

که هر کدام از ارزش‌های عبارت سمت راست بدین صورت قابل بسط است[۱۸]:

$$\begin{aligned}
&\underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \mid B_k \text{ is controlling}\right] \\
&= \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \middle| B_k \text{ is controlling and nobody else is active}\right] \\
&+ \sum_{\forall B_j; B_k \overset{T}{\succ} B_j} \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \middle| \begin{matrix} B_k \text{ is controlling and} \\ B_j \text{ is the next active behavior} \end{matrix}\right] = V_{k0} + \sum_{\forall B_j; B_k \overset{T}{\succ} B_j} V_{k \succ j}
\end{aligned}$$

(۳-۱۹)

که در آن

$$V_{FO}(B_i \succ B_j) = V_{i \succ j} = \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \middle| B_i \text{ is controlling and } B_j \text{ is the next active behavior}\right]$$

(۳-۲۰)

$$V_{i0} = \underset{\pi}{E}\left[\frac{1}{N}\sum_{t=1}^{N} r_t \middle| B_i \text{ is controlling and nobody else is active}\right]. \qquad (۳-۲۱)$$

$V_{i>j}(i,j=1,...,n)$ و $V_{i0}(i=1,...,n)$ اجزای پایه‌ای‌ی بازنمایی مرتبه یک هستند. لازمه‌ی معتبر بودن تجزیه‌ی انجام گرفته، دو به دو مجزا بودن جمله‌های (۱۹-۳) است که چنان چیزی از نحوه‌ی

[18] B_j رفتار فعال بعد از B_i (**the next active behavior**) به صورت اولین رفتار فعال پایین‌تر از B_i ($B_i \overset{T}{\succ} B_j$) در معماری T تعریف می‌شود.

تعریف V_{i0} و $V_{i\succ j}$ مشخص است[19]. با فرض آن‌که حداقل یکی از رفتارهای معماری فعال شده باشد، آن‌گاه ارزش عامل را می‌توان بدین‌گونه نوشت:

$$V_T = \sum_{i=1}^{m} \left\{ V_{index(i)0} + \sum_{j=1}^{i-1} V_{index(i)\succ index(j)} \right\} P(B_i \text{ is controlling}) . \qquad (۳-۲۲)$$

به‌ترین معماری، آن‌ای است که (۳-۲۲) را بیشینه می‌کند. با داشتن توابع ارزش و مقادیر $P(B_i \text{ is controlling})$، این مساله‌ی بهینه‌سازی‌ی ترکیبیاتی با استفاده از یکی از روش‌هایی که در انتهای زیربخش بازنمایی مرتبه صفر گفته شد قابل حل است.

ابعاد بازنمایی مرتبه این‌چنین است که نسبت به ابعاد فضای کامل بسیار کوچک‌تر است:

$$cardinality(FO) = n^2 + n . \qquad (۳-۲۳)$$

۳-۳-۲) تخصیص اعتبار و به روز رسانی بازنمایی

با مشاهده‌ی الگوی فعالیت رفتارها و با توجه به تعریف V_{i0} و $V_{i\succ j}$ می‌توان مساله‌ی تخصیص اعتبار را حل کرد. در این بازنمایی، دو دسته رخداد –فعال شدن یک رفتار و فعال شدن بیش از یک رفتار– پیش می‌آید. *در حالت‌ای که فقط یک رفتار فعال می‌شود، V_{i0} می‌بایست به روز شود. در صورت‌ای که تعداد دو یا بیش‌تر رفتار فعال شدند، $V_{i\succ j}$ به ازای iای که شماره‌ی رفتار کنترل‌کننده B_i و jای که شماره‌ی رفتار فعال بعدی B_j است می‌بایست به روز شود.*

قوانین به روز رسانی بدین‌گونه‌اند:

[19] می‌توان نشان داد که اگر در تعریف $V_{i\succ j}$ بقیه‌ی رفتارهای فعال نیز در نظر گرفته می‌شد (برخلاف آن‌چه در این‌جا به کار رفته است، یعنی فقط اولین رفتار فعال بعدی)، آن‌گاه اجزای تجزیه ارزش عامل دیگر دو به دو مجزا نبودند و V_T دیگر تجزیه‌پذیر نبود.

$$V_{i0_{k+1}} = (1-\alpha_{k,i0})V_{i0_k} + \alpha_{k,i0} r_k \qquad \begin{pmatrix}\text{whenever } B_i \text{ is controlling}\\ \text{and no other behavior is active}\end{pmatrix} \qquad (۳-۲۴)$$

$$V_{i\succ j_{k+1}} = (1-\alpha_{k,i\succ j})V_{i\succ j_k} + \alpha_{k,i\succ j} r_k \qquad \begin{pmatrix}\text{whenever } B_i \text{ is controlling} \wedge B_i \overset{T}{\succ} B_j\\ \wedge\ B_j \text{ is the next active behavior}\end{pmatrix} \qquad (۳-۲۵)$$

برای تخمین $P(B_i \text{ is controlling})$ می‌توان تعداد مرتبه‌ای را که B_i کنترل‌کننده بوده است شمرد. اما به مانند قبل می‌توان هر دو جمله‌ی ارزش و احتمال را با هم به روز رساند. با تعریف

$$\widetilde{V}_{i\succ j} = V_{i\succ j} P(B_i \text{ is controlling}) \qquad (۳-۲۶)$$

$$\widetilde{V}_{i0} = V_{i0} P(B_i \text{ is controlling}) \qquad (۳-۲۷)$$

و با استدلال‌ای مشابه بازنمایی پیشین، می‌توان قوانین به روز رسانی را بدین‌گونه نوشت:

$$\widetilde{V}_{i0_{n+1}} = (1-\alpha_{n,i0})\widetilde{V}_{i0_n} + \alpha_{n,i0} r'_{n,i} \qquad (۳-۲۸)$$

$$\widetilde{V}_{i\succ j_{n+1}} = (1-\alpha_{n,i\succ j})\widetilde{V}_{i\succ j_n} + \alpha_{n,i\succ j} r''_{n,i\succ j} \qquad (۳-۲۹)$$

که در آن‌ها

$$r'_{n,i} = \begin{cases} r_n & B_i \text{ is the controlling behavior and no other behavior is active} \\ 0 & \text{otherwise} \end{cases} \qquad (۳-۳۰)$$

$$r''_{n,i\succ j} = \begin{cases} r_n & B_i \text{ is the controlling behavior and } B_j \text{ is the next active behavior} \\ 0 & \text{otherwise} \end{cases} \qquad (۳-۳۱)$$

توجه داشته باشید که (۲۸-۳) و (۲۹-۳) در هر واحد زمانی به روز خواهند شد در حالی که (۲۴-۳) و (۲۵-۳) تنها هنگامی به روز می‌شوند که B_j رفتار کنترل‌کننده باشد.

۴) یادگیری رفتار

۴-۱) مقدمه

در فصل یادگیری ساختار (فصل ۳) فرض کرده بودم که جعبه‌ابزاری از رفتارهای مناسب در اختیار داریم و عامل می‌بایست با استفاده از رفتارهای از پیش تعیین‌شده، به‌ترین ساختار ممکن را بیابد. گاهی طراح دانش نسبی‌ای به مساله‌ی دنیای واقعی دارد و می‌تواند رفتارهایی را که باعث حل مساله می‌شوند تا حدی مشخص و طراحی کند. در این صورت، او می‌تواند از روش‌های معرفی‌شده‌ی یادگیری‌ی ساختار برای پیدا کردن به‌ترین ترکیب ممکن رفتارها بهره جوید. اما در بیش‌تر موارد دانش طراح به اندازه‌ی کافی و دقیق نیست تا بتواند جزییات هر کدام از رفتارهای مطلوب را مشخص کند و یا حتی اگر بتواند چنان کاری را انجام دهد، هیچ الزام‌ای بر بهینه‌گی‌ی طراحی وجود نخواهد داشت. او با توجه به شناخت‌ای که از مساله دارد سعی در پیدا کردن به‌ترین نگاشت سنسور–محرک دارد در حالی که این نگاشت بهینه کاملا به دینامیک کنش‌گرانه‌ی عامل–محیط بستگی دارد و تجربه‌ی بشری‌ی طراح معمولا به یافتن مناسب‌ترین نگاشت منجر نمی‌شود.

برای روشن شدن مسله فرض کنید که طراح قصد دارد تا رفتار حرکتی‌ی خاص‌ای را —مثلا جلوگیری از برخورد با اجسام– به صورت رفتاری برای روبات متحرک‌ای که از سونار استفاده می‌کند ایجاد کند. تجربه‌ی مستقیم او از عمل جلوگیری از برخورد از سنسور قدرت‌مندی چون چشم او ناشی می‌شود در حالی که روبات دارای چنان سنسوری نیست. از آن‌جا که طراح در طول زندگی‌اش با سنسور بینایی‌ی قوی‌ای چون چشم کار کرده است و هم‌چنین دینامیک بدن او کاملا متفاوت با دینامیک روبات است، به هنگام طراحی‌ی چنان رفتاری برای روبات می‌بایست حدس بزند که چگونه با استفاده از سونار می‌توان آن رفتار را ایجاد کرد. این مشابه‌سازی‌ی ذهنی به دور از هرگونه بهینه‌گی‌ای است و در به‌ترین حالت او تنها به طراحی‌ی قابل قبول‌ای دست می‌یابد.

مساله‌ی یادگیری رفتار برای حل این مشکل مطرح می‌شود. هدف یادگیری رفتار پیدا کردن نگاشت مناسب بین ورودی هر رفتار و خروجی‌ی آن است. مناسب بودن یک رفتار به تاثیر آن بر

عمل‌کرد عامل باز می‌گردد. به بیان‌ای دیگر، هدف پیدا کردن مجموعه پارامترهای توصیف‌گر رفتاری است که مساله‌ی بهینه‌سازی (۹-۲) را حل کند.

هدف این فصل معرفی‌ی روش‌ای برای انجام این کار است. در این فصل فرض می‌کنم که برای حلِ مساله‌ی یافتن نگاشت $a_i = B_i(s'_i)$ مطلوب، فضای حالت (S_i) و عمل (A_i) هر رفتار و هم‌چنین سیگنال تقویت دانسته شده است[1].

در ابتدا نشان خواهم داد که چگونه می‌توان ارزش کلِ عامل را به اجزای هر رفتار تقسیم کرد. این قدم شبیه به آن‌چیزی است که در یادگیری ساختار انجام دادم. بعد از این مرحله‌ی اولیه، چارچوب‌ای کلی برای حل مساله‌ی تقسیم امتیاز در خانواده‌ای از سیستم‌های چندعامله معرفی می‌کنم و سپس با استفاده از آن چگونگی‌ی تقسیم امتیاز بین رفتارها در معماری‌ی سلسله مراتبی‌ی PPSSA را مشخص خواهم کرد. در نهایت بررسی‌ای بر تاثیر وجود NA بر بازده عامل انجام خواهم داد و روابطی برای یادگیری و به روز رسانی‌ی دانش هر رفتار نسبت به محیط اطراف‌اش ارایه خواهم کرد.

۴-۲) تجزیه ارزش به اجزای رفتاری

در این بخش به ارتباط بین تجربه‌ی هر رفتار که به صورت تابع ارزش حالت-عمل ($Q_i(s'_i, a_i)$) ثبت می‌شود و ارزش کلِ عامل می‌پردازم. نشان می‌دهم که ارزش کل معماری به صورت مجموعه‌ای از ارزش‌های اجزای هر رفتار (یعنی حالت‌های هر رفتار) قابل بیان است.

در (۶-۳)، V_T به اجزای ساده‌تری که شامل V_{ij}، $P\{B_j|L_i\}$ و $P(L_i \text{ is controlling})$ بودند تجزیه شد. اگر ارزش هر حالتِ هر رفتار را به صورت امید ریاضی‌ی پاداش لحظه‌ای آن حالت در نظر بگیریم (در نتیجه $V_j(s'_j)$ یا $Q_j(s'_j, a_j)$ تخمین‌ای از متوسط پاداش در طول یک اپیزود یا پاداش discount شده نیست)، آن‌گاه V_{ij} را به صورت مجموعه‌ای از اجزای $V_j(s'_j)$اش می‌توان نوشت. این محدودیت به دلیل تعریف V_{ij} (۴-۳) است که در آن V_{ij} متوسطی از پاداش‌هایی است که B_iی که درjامین لایه بوده است هنگام کنترل‌کننده‌بودگی‌اش دریافت کرده است. حالت‌های کلی‌تر توابع ارزش حالت و حالت-عمل در بخش‌های بعدی استفاده می‌شوند اما این تجزیه تنها هنگامی که ارزش را به

[1] نتایج این فصل و نتایج تکمیل شده‌ی فصل یادگیری ساختار در [Farahmand05b] آمده است.

صورت پاداش لحظه‌ای تعریف می‌کنیم معتبر است. با این فرض، V_{ij} بازنمایی مرتبه صفر به صورت زیر قابل گسترش است[۲]:

$$V_{ij} = \sum_{s'_j \in S'_j} P\{s'_j | B_j \text{ is controlling in } L_i\} V_j(s'_j) \qquad (۴-۱)$$

که در آن S'_j فضای ورودی‌ی تحریک‌شده‌ای است که توسط M_j از فضای اصلی $S_j \subset S$ نگاشت شده است. هم‌چنین داریم:

$$V_j(s'_j) = \sum_{a_j \in A_j} \pi_j(s'_j, a_j) Q_j(s'_j, a_j) \overset{\text{greedy policy}}{\rightarrow} = \max_{a_j} Q_j(s'_j, a_j) \qquad (۴-۲)$$

که در آن $\pi_j(s'_j, a_j)$ احتمال انتخاب عمل a_j در حالت s'_j رفتار B_j است در آن هنگام که آن رفتار کنترل‌کننده باشد. با توجه به این تجزیه و در نظر گرفتن تجزیه‌ی رفتاری‌ای که در بازنمایی مرتبه صفر داشتیم، V_T را می‌توان بر حسب $P\{B_j | L_i\}$، $P(L_i \text{ is controlling})$، $P\{s | B_j \text{ is controlling in } L_i\}$ و $Q_j(s', a_j)$ نوشت[۳]:

$$V_T = \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{s' \in S'_j} \sum_{a_j \in A_j} P\{B_j \mid L_i\} P(L_i \text{ is controlling}) P\{s'_j | B_j \text{ is controlling in } L_i\} \pi_j(s'_j, a_j) Q_j(s'_j, a_j)$$

. (۴-۳)

به دست آوردن رابطه‌ای مشابه برای بازنمایی مرتبه یک نیز قابل انجام است.

---

[2] اگر S'_j فضایی پیوسته باشد، جمع‌کردن به انتگرال‌گیری تبدیل می‌شود. مثال‌ای از استفاده از انتگرال به جای جمع در بخش " بهینه‌گی، مجموعه عمل‌ها و به روز رسانی ارزش" این فصل آمده است.

[3] توجه کنید که همه‌ی این مقادیر را می‌توان در طول یادگیری تخمین زد.

۴-۳) تخصیص اعتبار به رفتارها

یکی از مسایل اساسی در یادگیری رفتار، چگونگی‌ی تخصیص اعتبار به رفتارها است. به طور معمول معماری‌ی رفتارگرا شامل چندین رفتار مختلف است که هر کدام نقش‌ای برای ایجاد رفتار مطلوب کلی ایفا می‌کنند. اگر هر کدام از رفتارها را یک عامل بدانیم، یک معماری‌ی رفتارگرا را می‌توان به صورت سیستم‌ای چندعامله در نظر گرفت و مساله‌ی تخصیص اعتبار در معماری‌ای رفتارگرا را به صورت تخصیص اعتبار در سیستم‌ای چندعامله در نظر گرفت.

با این وجود مساله‌ی تخصیص اعتبار در سیستم‌های چندعامله نیز چندان ساده نیست چون در حالت کلی اطلاعات کافی‌ای برای چگونگی‌ی تقسیم اعتبار تیمی به هر کدام از عامل‌ها در اختیار نیست. به عبارت دیگر در حالت کلی مشخص نیست که نقش هر کدام از عامل‌ها در تنبیه یا پاداش دریافتی چه بوده است ([Harati04]. هم‌چنین در [Shoham04] نگاه نقادانه‌ای به روش‌های مختلف یادگیری چندعامله‌ی تقویتی شده است). با این‌وجود گاهی اوقات قیدهایی که در ساختار تیم چندعامله وجود دارد ما را در تقسیم امتیاز راه‌نمایی می‌کند. در این بخش ابتدا چارچوب‌ای را برای تقسیم امتیاز در خانواده‌ای محدود از سیستم‌های چندعامله معرفی می‌کنم و سپس با استفاده از آن چارچوب، روش‌ای برای حلِ مساله‌ی تقسیم امتیاز در PPSSA پیش‌نهاد خواهم کرد.

۴-۳-۱) سیستم چندعامله به عنوان یک جبر بول[4]

فرض کنید که عامل‌های A_1، A_2، ... و A_n با هم‌دیگر تیم چندعامله‌ی T را می‌سازند. هم‌چنین فرض کنید که موفقیت یا شکست تیم T تابع‌ای منطقی از موفقیت یا شکست هر کدام از A_i باشد. به عنوان مثال $T = A_1 \wedge A_2$ تیم‌ای است که شرط موفقیت آن، صحیح عمل کردن A_1 **و** A_2 باشد (حالت **و**) و $T = A_1 \vee A_2$ تیم‌ای است که شرط موفقیت آن صحیح کار کردن حداقل یکی از A_1 **یا** A_2 باشد (حالت **یا**). تیم‌های "**و**"ای و "**یا**"ای دو حد نهایت مختلف سیستم‌های چندعامله‌ی هم‌کار[5] هستند.

[4] Boolean Algebra
[5] Cooperative Multi-Agent Systems

روش‌هایی برای حل این دو حالت حدی در [Harati04] پیش‌نهاد شده است. آن روش‌ها پیش‌نهاد می‌کنند که هنگامی که تیم "**و**"ای (**"یا"**ای) پاداش (تنبیه) دریافت می‌کند، آن پاداش (تنبیه) می‌بایست به همه‌ی عامل‌ها تخصیص داده شود. روش‌های تقریبی‌ای برای تخصیص اعتبار در زمان‌هایی که تیم **"**و**"**ای (**"یا"**ای) تنبیه (پاداش) دریافت می‌کند نیز پیش‌نهاد شده است که مبتنی بر استفاده از معیارهای خبرگی[6] ([Nili02]) هر عامل است. در این پایان‌نامه تنها از نتایج **"و"**ای-پاداش و **"یا"**ای-تنبیه استفاده کرده‌ام.

در مسایل‌ای که موفقیت یا شکست تیم T تابع‌ای منطقی از موفقیت یا شکست هر کدام از اجزای‌ای (A_i) باشد، تخصیص اعتبار از قوانین جبر بول تبعیت می‌کند و در نتیجه همه‌ی عملیات تعریف شده و استنتاج‌های ممکن در آن جبر قابل استفاده خواهد بود. به عنوان مثال دانستن این‌که تیم $T = A_1 \wedge A_2$ درست کار کرده است ما را قادر به این می‌کند که بدانیم هر کدام از A_1 و A_2 نیز به تنهایی درست کار کرده‌اند.

مجموعه‌ی B را که شامل همه‌ی ترکیب‌های عامل‌های $\{A_i\}$ است در نظر بگیرید. اصول جبر بول **B** برای تمام سیستم‌های چندعامله‌ای که صحت کارکرد تیم با عبارت‌ای منطقی به صحت کارکرد هر کدام از اجزای‌اش ارتباط دارد صادق است:

ب۰: *اگر* A_1 و A_2 در مجموعه‌ی B باشند، $A_1 \wedge A_2$ و $A_1 \vee A_2$ نیز در مجموعه‌ی B هستند.
این خاصیت از تعریف B مشخص است.

ب۱ – *تعویض‌پذیری*[7]: عمل‌کرد T مستقل از ترتیب در نظر گرفتن هر کدام از عامل‌های آن است، یعنی:

[6] Expertness
[7] Commutativity

$$A_1 \wedge A_2 \equiv A_2 \wedge A_1 \qquad (۴-۴)$$

$$A_1 \vee A_2 \equiv A_2 \vee A_1 \qquad (۴-۵)$$

ب۲ – *شرکت‌پذیری*[8]: عمل‌کرد T مستقل از مکان شروع محاسبه‌ی عمل‌کرد هر کدام از عامل‌های آن است. به عبارت دیگر اهمیت‌ای ندارد که اول درستی‌ی A_1 و A_2 بررسی شود و سپس نتیجه به درستی‌ی A_3 "**و**" شود و یا این‌که ابتدا عمل‌کرد A_2 و A_3 بررسی شود و سپس نتیجه به درستی‌ی A_1 "**و**" شود.

$$A_1 \wedge (A_2 \wedge A_3) \equiv (A_1 \wedge A_2) \wedge A_3 \qquad (۴-۶)$$

$$A_1 \vee (A_2 \vee A_3) \equiv (A_1 \vee A_2) \vee A_3 \qquad (۴-۷)$$

ب۳ – *توزیع‌پذیری*[9]: "**و**" ($\wedge$) و "**یا**" ($\vee$) به طور متقابل قابل توزیع هستند:

$$A_1 \wedge (A_2 \vee A_3) = (A_1 \wedge A_2) \vee (A_1 \wedge A_3) \qquad (۴-۸)$$

$$A_1 \vee (A_2 \wedge A_3) = (A_1 \vee A_2) \wedge (A_1 \vee A_3) \qquad (۴-۹)$$

سیستم چندعامله‌ای را مطابق با سمت چپ (۸-۴) در نظر بگیرید. اگر A_1 درست کار کند و دست‌کم یکی از A_2 یا A_3 درست کار کنند، آن‌گاه کل سیستم هم درست کار می‌کند. فرض کنید A_2 عامل‌ای است که درست کار می‌کند. در نتیجه درست کار کردن $A_1 \wedge A_2$ قابل استنتاج است. از این نتیجه می‌شود که $(A_1 \wedge A_2) \vee (A_1 \wedge A_3)$ نیز درست کار می‌کند و در نتیجه تساوی‌ی دو طرف رابطه‌ی (۸-۴) برقرار خواهد شد. حالت‌های دیگر (درست یا نادرست بودن هر کدام از A_1، A_2 و A_3) نیز به همین ترتیب قابل استنتاج است. اثبات صحت (۹-۴) نیز به روش‌ای مشابه انجام می‌شود.

[8] Associativity
[9] Distributivity

ب۴ – *عامل‌های خنثی*[10]: عامل‌های خنثی‌ی خیالی‌ی **1** و **0**ای برای اپراتور "**و**" ($\wedge$) و "**یا**" ($\vee$) وجود دارند. **1** عامل مجازی‌ای است که در همه‌ی حالات درست کار می‌کند و **0** نیز عامل مجازی‌ی دیگری است که در هیچ شرایطی درست کار نمی‌کند. روابط زیر برای این عامل‌ها برقرار است:

$$A \wedge \mathbf{1} = A \qquad (۴-۱۰)$$

$$A \vee \mathbf{0} = A \qquad (۴-۱۱)$$

ب۵ – عامل‌های متمم[11]: برای هر عامل اپراتور A، عامل مجازی‌ی $\overline{A}$ وجود دارد که هر هنگام این‌دو با هم "**یا**" ($\vee$) شوند، نتیجه همیشه درست خواهد بود. برای عامل Aای که درست عمل می‌کند، عامل متمم $\overline{A}$ آن نادرست عمل می‌کند و برای عامل‌ای که نادرست عمل می‌کند، عامل متمم درست عمل می‌کند. عامل متمم $\overline{A}$ برای حالت "**و**"ای عامل‌ای است که $A \wedge \overline{A}$ را نادرست می‌کند.

۴-۳-۲) تخصیص اعتبار به رفتارها در معماری Subsumption

حل مساله‌ی تخصیص اعتبار به رفتارهای معماری‌ی رفتارگرای PPSSA به کمک چارچوب معرفی شده ممکن است. همان‌طور که پیش‌تر گفتم معماری‌ی رفتارگرا را می‌توان به صورت سیستم‌ای چندعامله در نظر گرفت. نشان خواهم داد که درستی یا نادرستی‌ی عمل‌کرد معماری مورد نظر تابع‌ای منطقی از عمل‌کرد هر کدام از رفتارها است و در نتیجه استفاده از چارچوب معرفی شده ممکن خواهد بود.

فرض کنید B^* بیان‌گر رفتار کنترل‌کننده در لحظه‌ی مورد بررسی باشد، $\{B_{u_i}(NA)\}$ مجموعه‌ی رفتارهای تحریک‌شده‌ی بالاتر از B^* در معماری T که خروجی‌ی NA داده‌اند و $\{B_{\overline{u}_i}\}$ نیز رفتارهای بالاتر از B^*ای که تحریک‌نشده‌اند. هم‌چنین $\{B_{l_i}\}$ آن رفتارهای پایین‌تر از B^*ای هستند که فعال شده‌اند (یعنی خروجی‌ای غیر از NA *پیش‌نهاد* داده‌اند) اما توسط B^* سرکوب شده‌اند (در نتیجه

[10] Neutral
[11] Complement

یاری‌بهری[۱۲] در خروجی نداشته‌اند) و $\{B_{\bar{l}_i}\}$ نیز رفتارهای پایین‌تری هستند که اصلا تحریک نشده‌اند. مشخص است که این مجموعه همه‌ی رفتارها را افراز[۱۳] می‌کند – $\{\{B^*,\{B_{u_i}(NA)\}\},\{B_{\bar{u}_i}\},\{B_{l_i}\},\{B_{\bar{l}_i}\}\}$ همه‌ی رفتارها را یک و فقط یک بار شامل می‌شود. برای حل مساله‌ی تخصیص اعتبار در این معماری دو حالت مختلف جداگانه تحلیل می‌شوند: زمان‌هایی که عامل پاداش دریافت کرده است و زمان‌هایی که عامل تنبیه شده.

فرض کنید که معماری‌ی T در لحظه‌ی جاری پاداش‌ای دریافت کند. این پاداش حاصل عمل صحیح B^* و فعال نشدن $\{B_{u_i}(NA)\}$ بوده است. در نتیجه اعضای مجموعه‌ی $\{B^*,\{B_{u_i}(NA)\}\}$ به درستی عمل کرده‌اند. پس عبارت جبر بول زیر در حالت دریافت پاداش صحیح است:

$$T_{R^+}: B^* \wedge \left(B_{u_1}(NA) \wedge \ldots \wedge B_{u_k}(NA)\right). \qquad (۴-۱۲)$$

هم‌چنین اعضای مجموعه‌ی $\{B_{\bar{u}_i}\}$ یاری‌بهری در بازده کلی‌ی سیستم نداشته‌اند؛ آن‌ها هیچ تصمیم‌گیری‌ای انجام نداده‌اند که خوب یا بد باشد. پس می‌توان آن‌ها را به عنوان عامل‌های خنثی در نظر گرفت. عامل خنثی‌ی اپراتور "**و**"، با **1** نمایش داده می‌شود. پس می‌توان نوشت:

$$T_{R^+}: B^* \wedge \left(B_{u_1}(NA) \wedge \ldots \wedge B_{u_k}(NA) \wedge 1\right). \qquad (۴-۱۳)$$

در مورد رفتارهای پایین ($\{B_{l_i}\}$ و $\{B_{\bar{l}_i}\}$) می‌توان گفت که هیچ تاثیری در عمل‌کرد سیستم نداشته‌اند و در نتیجه حرف خاص‌ای در مورد درست یا نادرست عمل‌کردن آن‌ها نمی‌توان گفت:

$$T_{R^+}: \overbrace{\hat{1}}^{\text{lower behaviors}} \wedge \overbrace{B^*}^{\text{controlling behavior}} \wedge \overbrace{\left(B_{u_1}(NA) \wedge \ldots \wedge B_{u_k}(NA) \wedge \overbrace{\hat{1}}^{\text{not excited}}\right)}^{\text{upper behaviors}}. \qquad (۴-۱۴)$$

[12] Contribution
[13] Partition

اینک تخصیص اعتبار به رفتارها در حالت سیگنال تقویت مثبت مشخص شده است: در این حالت $T_{R^+} \equiv 1$ و در نتیجه استنتاج روبرو با توجه به خواص جبر بول صحیح است[۱۴]:

$$T_{R^+} : + \Rightarrow \begin{cases} B^* : + \\ \{B_{u_i}(NA)\} : + \\ \{B_{\bar{u}}\} : \text{unknown} . \\ \{B_l\} : \text{unknown} \\ \{B_{\bar{l}}\} : \text{unknown} \end{cases} \qquad (۴-۱۵)$$

راه حل مساله‌ی تخصیص اعتبار در حالت دریافت سیگنال تنبیه اندکی گنگ و نامشخص است. معماری‌ی $T = \lfloor B_{l_i} \quad B_{\bar{l}_i} \quad B^* \quad B_{u_i}(NA) \quad B_{\bar{u}_i} \rfloor$ را در نظر بگیرید که سیگنال تنبیه دریافت کرده است. در حالت سیگنال مثبت، درستی‌ی عمل‌کرد معماری را بدین‌گونه به درستی‌ی عمل‌کرد اجزای‌اش ارتباط داده بودم:

$$(T \equiv +) \Rightarrow \left[B^* \wedge \left(B_{u_1}(NA) \wedge \ldots \wedge B_{u_k}(NA)\right)\right] \equiv + . \qquad (۴-۱۶)$$

اما از نظر منطقی نمی‌توان عبارت زیر را به طور مستقیم با عوض کردن علامت مثبت به منفی نتیجه گرفت:

$$(T \equiv -) \Rightarrow \left[B^* \wedge \left(B_{u_1}(NA) \wedge \ldots \wedge B_{u_k}(NA)\right)\right] \equiv - . \qquad (۴-۱۷)$$

[14] قابل ذکر است که این روش تخصیص اعتبار ممکن است به پاسخ‌های زیربهینه‌ای منجر شود. دلیل چنین چیزی این است که ممکن است اگر رفتارهای بالاتر $B_{u_i}(NA)$ به جای NA، عمل‌ای پیش‌نهاد می‌دادند نتیجه‌ی به‌تری از عمل پیش‌نهادی‌ی B^* حاصل می‌شد. این مشکل در مسایل‌ای که در آن بیش از دو سطح درست و نادرست (یا مثبت و منفی) عمل‌کرد سیستم وجود دارد رخ می‌دهد. به نظر می‌رسد اگر سیستم جستجویی کاوش‌گرانه در انتخاب عمل‌های‌اش داشته باشد این مساله رفع شده و سیستم قادر به یافتن پاسخ‌های بهینه نیز شود.

برای تحلیل حالت تنبیهی، فرض کنید که عمل صحیح که منجر به پاداش می‌شود —که البته از پیش دانسته نیست– در فضای عملِ همه‌ی رفتارهای B^* و $\{B_{u_i}(NA)\}$ وجود داشته باشد. یعنی:

$$a^* \in \begin{cases} A_{B^*} \\ A_{B_{u_i}(NA)} \quad \forall B_{u_i}(NA) \end{cases}. \qquad (۴-۱۸)$$

در نتیجه هنگام دریافت تنبیه، ممکن است همه‌ی رفتارهای $\{B_{u_i}(NA)\}$ با پیش‌نهاد NAشان درست عمل کرده باشند اما ایراد از عملِ رفتار B^* بوده باشد. اگر از $\overline{B}$ به عنوان نماد رفتاری که پاسخ نادرستی داده است بهره جوییم، این وضعیت به صورت $\overline{B}^* B_{u_1} \ldots B_{u_k}$ قابل بیان است. حالت دیگر می‌تواند این باشد که $B_{u_1}(NA)$ به نادرستی خروجی داده (یعنی انتخاب NA انتخاب درستی نبوده است) و به جای‌اش باید $a_1 = a^* \in A_{B_{u_1}}$ را انتخاب می‌کرده است. اگر نماد $\widehat{B}$ را به عنوان نماد رفتار بدون یاری‌بهر در نتیجه‌ی حاصل در لحظه‌ی فعلی در نظر بگیریم، این شرایط به صورت $\widehat{B}^* \overline{B}_{u_1} \ldots B_{u_k}$ قابل بیان است. در حالت کلی یکی از شرایط زیر رخ می‌دهد:

$$\begin{cases} \overline{B}^* B_{u_1} \ldots B_{u_k} \\ \widehat{B}^* \widehat{B}_{u_1} \widehat{B}_{u_2} \ldots \overline{B}_{u_i} \ldots B_{u_k} \quad i = 1, \ldots, k \end{cases}. \qquad (۴-۱۹)$$

پس با فرض انجام شده همه‌ی رفتارهای $\{B^*, \{B_{u_i}(NA)\}\}$ می‌توانستند به گونه‌ای عمل کنند که سیستم پاداش دریافت کند. چون در هنگام دریافت تنبیه هیچ‌کدام از آن‌ها این‌گونه عمل نکرده‌اند، پس می‌توان نتیجه گرفت که همه‌شان می‌بایست تنبیه شوند. به مانند قبل، چیزی در مورد رفتارهای بی‌یاری‌بهر $\{B_{\overline{u}_i}\}$، $\{B_{l_i}\}$ و $\{B_{\overline{l}_i}\}$ نمی‌توان گفت: پس آن‌ها به مانند عامل‌های خنثی در سیستم چندعامله عمل می‌کنند. روی هم رفته می‌توان نوشت:

$$T_{R^-}: \quad \overbrace{\vec{0}}^{\text{lower behaviors}} \vee \overbrace{\vec{B}^*}^{\text{controlling}} \vee \overbrace{\left(B_{u_1}(NA) \vee B_{u_2}(NA) \vee \ldots \vee B_{u_k}(NA) \vee \overbrace{\vec{0}}^{\text{not excited}} \right)}^{\text{upper behaviors}}. \qquad (۴-۲۰)$$

در نتیجه تخصیص اعتبار در حالت دریافت سیگنال تنبیه در صورتی که a^* عضوی از فضای عمل همه‌ی رفتارهای B^* و $\{B_{u_i}(NA)\}$ باشد بدین‌گونه خواهد بود:

$$T_{R^-}: - \Rightarrow \begin{cases} B^*: - \\ \{B_{u_i}(NA)\}: - \\ \{B_{\bar{u}}\}: \text{unknown} \\ \{B_l\}: \text{unknown} \\ \{B_{\bar{l}}\}: \text{unknown} \end{cases}. \qquad (۴-۲۱)$$

نتیجه در ظاهر و با تغییر علامت مثبت به منفی شبیه به حالت سیگنال تقویت مثبت است اما فرآیند استنتاج پشت‌اش اندکی متفاوت است. در حالت منفی فرض کرده بودم که همه‌ی رفتارها، عملِ صحیح را در فضای عمل‌های‌شان دارند (۱۸-۴) ولی لزومی به چنین فرض‌ای در حالت مثبت نیست. گاهی اوقات این فرض انجام شده چندان صحیح نیست و ممکن است بعضی از رفتارهای تحریک شده‌ی بالایی $\{B_{u_i}(NA)\}$ قادر به خروجی‌ی صحیح دادن نباشند (یعنی $a^* \notin A_{B_{u_i}}$). در نتیجه تنبیه کردن آن‌ها عاقلانه نخواهد بود و به عنوان نویزی در سیگنال تقویت محسوب می‌شود[۱۵]. به هر حال از پیش دانسته نیست که آیا عملِ بهینه در مجموعه‌ی اعمال یک رفتار خاص وجود دارد یا نه و در نتیجه اشتباه در پاداش نادرست اجتناب‌ناپذیر است. البته در مسایل‌ای که خروجی‌ی همه‌ی رفتارها یک‌سان است —مثلا همه به کنترل سرعت چرخ‌های روبات مشغول‌اند- و یا این‌که فضای تحریک رفتارها جدا از هم است —و در نتیجه در هر زمان تنها یک رفتار تحریک می‌شود- چنین مشکل‌ای پیش نمی‌آید.

[15] برای مشاهده‌ی تحلیل‌هایی درباره‌ی نتایج تاثیر عدم قطعیتِ سیگنال تقویت و وجود نویز در آن بر تابع ارزش و سیاست عامل به [Singh94]، [Williams94] و [Farahmand05a] مراجعه کنید. نتایج مرجع آخر – به عنوان بخش‌ای از کار انجام شده در این پژوهش- در فصل ۸ آورده می‌شود.

به طور خلاصه از تحلیل‌های این بخش به این نتیجه می‌رسیم که برای حل مساله‌ی تخصیص اعتبار در PPSSA، *عمل پیش‌نهادی‌ی رفتار کنترل‌کننده‌ی* B^* *و همه‌ی* NA*های رفتارهای بالاتر* $\{B_{u_i}(NA)\}$ *می‌بایست متناسب با سیگنال تقویت دریافتی‌ی عامل تقویت شوند.*

۴-۴) بهینه‌گی، مجموعه عمل‌ها و به روز رسانی ارزش

در این بخش به بررسی‌ی تاثیر وجود NA بر بهینه‌گی عمل‌کرد عامل می‌پردازم و نشان خواهم داد که وجود آن عمل مجازی شانس بالا رفتن عمل‌کرد عامل را ایجاد می‌کند. پس از آن روش‌ای برای به روز رسانی‌ی ارزش‌های هر حالتِ رفتار برای دو وضعیت مختلف پاداش لحظه‌ای و پاداشِ بازگشته معرفی خواهم کرد.

فرض کنید می‌خواهیم ترتیب بهینه‌ی دو رفتار B_1 و B_2 را –که به ترتیب دارای زیرفضای تحریک S_1 و S_2 هستند– بیابیم (مساله‌ی یادگیری ساختار). مشخص است اگر عمل NA در مجموعه‌ی اعمال انتخابی‌ی آن دو رفتار نباشد، آن‌ای که بالاتر است (به آن B_{higher} می‌گویم که می‌تواند هر کدام از آن دو رفتار باشد) کنترل کامل عامل را در همه‌ی حالت‌هایی از فضای اصلی ($s \in S$) که جزو زیرفضای تحریک‌اش است می‌گیرد ($\{s|\forall s \in S; s \in S_{higher}\}$). رفتار دیگر ($B_{lower}$) در آن مجموعه سرکوب می‌شود و هیچ کنترلی در اختیار ندارد. برای این‌که بفهمیم این ترتیب خاص بهینه است یا نه، یاری‌بهر هر کدام از رفتارها بر پاداش دریافتی را با هم مقایسه می‌کنیم. منظور از یاری‌بهر یک رفتار در پاداش دریافتی‌ی عامل، پاداش‌ای است که عامل به دلیل سیاست خاص آن رفتار به دست آورده است. این پاداش ممکن است به دلیل عمل آن رفتار و دریافت پاداش لحظه‌ای باشد و یا این‌که به دلیل فعال نشدن و "اجازه‌ی کنترل‌کننده‌بودن به رفتارهای پایین دادن" باشد. یعنی یک رفتار اگر خود نیز کنترل‌کننده نباشد ولی با انتخاب NA در زمان مناسب به رفتار پایینی‌اش اجازه‌ی انجام عمل مناسب بدهد قابل تقدیر تصمیم گرفته است. هم‌چنین این پاداش ممکن است تاخیریافته یا لحظه‌ای باشد. برای روشن شدن معنای یاری‌بهر، فرض کنید B_{higher} کنترل عامل را در اختیار گرفته باشد و عامل پاداش دریافت کند. مشخص است که این پاداش به دلیل یاری‌بهر B_{higher} بوده است. حال فرض کنید که B_{higher} تحریک می‌شود ولی NA پیش‌نهاد می‌کند و در نتیجه پس از آن B_{lower}

کنترل‌کننده می‌شود. اگر عامل پاداش دریافت کند –همان‌طور که در زیربخش پیشین گفتم– نه فقط به دلیل عمل‌کرد مناسب عملِ انتخابی‌ی B_{lower} بوده که به دلیل پیش‌نهاد مناسب B_{higher} $\underset{\sim}{NA}$ نیز بوده است. "*پاداش‌ای که به دلیل یاری‌بهر* B_i *در معماری‌ی* T *در زیرفضای اکید*[۱۶] $S^*(\subseteq S)$ *حاصل شده است*" را بدین‌گونه تعریف می‌کنم:

$$\bar{r}_i\big|_{s\in S^*} = E\big[R_i(s)\big]_{s\in S^*} = E\Big[R(s)\Big|\text{Reward is achieved by the contribution of } B_i \wedge "s \in S^*"\Big]$$
$$= \int_{s\in S^*} R(s)p\{B_i \text{ is excited in } s\}p\{s|s\in S^*\}ds$$

(۴-۲۲)

که در آن $p\{s|s\in S^*\}$ احتمال قرارگیری حالت عامل در s است به شرط آن‌که حالت عامل در مجموعه‌ی S^* باشد. $R(s)$ نیز پاداشِ بازگشته عامل از حالت s است. پاداشِ بازگشته می‌تواند پاداش تک‌گام[۱۷] (که ارزش حالت‌های بعدی در ارزش حالت فعلی تاثیری ندارد) و یا پاداش بازگشته‌ی چندگام‌ای[۱۸] مانند متوسط پاداش‌های اپیزود باشد.

اینک با تعاریف انجام شده می‌توان دو رفتار را در کلِ فضای حالت عامل (S) بدین‌گونه مقایسه کرد:

$$\int_S R(s)p\{B_{higher} \text{ is excited in } s\}p\{s|s\in S\}ds > \int_S R(s)p\{B_{lower} \text{ is excited in } s\}p\{s|s\in S\}ds$$
$$\Leftrightarrow B_{higher} > B_{lower}$$

(۴-۲۳)

این مقایسه مشخص می‌کند که کدام رفتار یاری‌بهر بیش‌تری در پاداش دریافتی‌ی عامل در طول زندگی‌اش داشته است. اما با این وجود این ترکیب ممکن است به ازای زیرمجموعه‌ی

[16] زیرفضای اکید S^* (strict subspace) بدین‌گونه تعریف می‌شود: $S^* = \{s|\forall s\in S; M_i(s)\in S'^*\}$

[17] Single Step

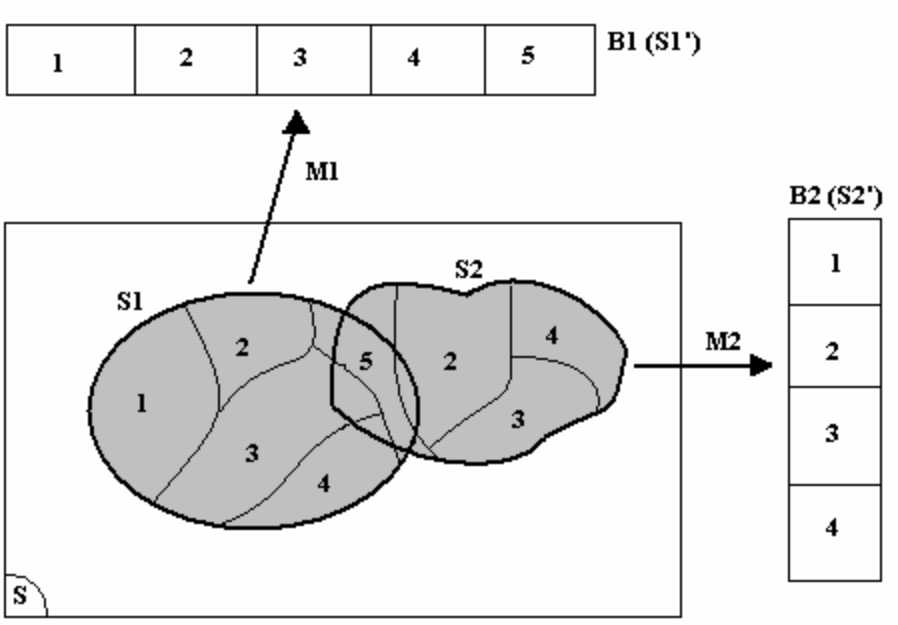
[18] Multi Step

$S^{\bullet} \subset S_{higher} \cap S_{lower}$ای که $\bar{r}_{lower}\big|_{s\in S^{\bullet}} > \bar{r}_{higher}\big|_{s\in S^{\bullet}}$ بهینه نباشد. اگر رفتار B_{higher} همیشه رفتار B_{lower} را بدون توجه به برتری‌ی دومی در بعضی از زیرمجموعه‌های فضای حالت سرکوب کند، بازده عامل کاهش می‌یابد. این مشکل با تعریف NA قابل حل است: عمل مجازی‌ای که در صورت انتخاب کاری انجام نمی‌دهد و اجازه می‌دهد که رفتار با وجود تحریک‌شدن، فعال نشود و شانس کنترل را به رفتارهای پایین واگذارد. این دلیل وجودی‌ی NA و تعمیم فضای عمل رفتارها است.

برای مشخص شدن این‌که در چه شرایطی B_{higher} می‌بایست کنترل‌کننده باشد و در چه شرایطی اجازه‌ی کنترل‌کننده‌گی به رفتارهای پایین بدهد، یاری‌بهر آن رفتار می‌بایست با یاری‌بهر رفتارهای پایین‌ترش مقایسه شود. برای این مقایسه، یاری‌بهر عمل‌ای از آن رفتار با یاری‌بهر انتخاب NA همان رفتار —که معادل یاری‌بهر رفتارهای پایین‌تر است– مقایسه می‌شود. در این مقایسه می‌بایست به این نکته توجه کرد که فضای تحریک رفتارهای متفاوت یک‌سان نیست $(S_i \neq S_j; i \neq j)$ و هم‌چنین از آن‌جا که در حالت کلی $M_i \neq M_j$، گاهی اوقات B_i و B_j ممکن است با هم تحریک شوند و گاهی تنها یکی از آن‌ها (شکل ۱-۴). به بیان‌ای دیگر، ممکن است به ازای زیرفضاهایی $(S_1, S_2 \subset S)$ از فضای حالت $s'_i = M_i(S_1)$ و $s'_j = M_j(S_1)$ (پس B_i و B_j هر دو تحریک شده‌اند) ولی $s'_i = M_i(S_2)$ اما $s'_j \neq M_j(S_2)$ (یعنی B_j دیگر در آن حالت درونی‌ی s'_j تحریک شده نیست). در نتیجه در معماری‌ی Tای با $B_i > \ldots > B_k$، مقایسه‌ی شایستگی‌ی B_i و رفتارهای پایین‌ترش می‌بایست در فضای تحریک مشترک آن رفتارها باشد. این نکته را در تحلیل‌های بعدی در نظر می‌گیرم.

رفتار B_i را در نظر بگیرید. $S^{\circ}(s'_i) \subset S$ را به صورت زیرمجموعه‌ای از S در نظر بگیرید که به یک نقطه s'_i از بازنمایی داخلی‌ی رفتار B_i نگاشت می‌یابد (هر کدام از بخش‌های شکل ۴-۱). تعریف $S^{\circ}(s'_i)$ بدین‌گونه است:

$$S^{\circ}(s'_i) = \left\{ s \middle| \forall s \in S; M_i(s) = s'_i \right\} \qquad (۴-۲۴)$$



شکل ۱-۴. زیرفضاهای تحریک‌پذیر رفتارهای B_1 و B_2 در فضای S و تناظر آن‌ها با S_1' و S_2'. توجه کنید که زیرفضاها تحریک‌پذیر ممکن است هم‌پوشانی داشته باشند.

حال می‌توان $\bar{r}_i\big|_{s\in S^\circ}$ را برای $S^\circ(s_i')$ بدین‌گونه نوشت:

$$\bar{r}_i\big|_{s\in S^\circ} = \int_{s\in S^\circ} R(s)p\{B_i \text{ is excited in } s\}p\{s|s\in S^\circ\}ds = R(s')p\{B_i \text{ is excited in } s'\} = V_i(s_i') \quad (۴-۲۵)$$

.

متوسط یاری‌بهر رفتار B_i در شرایطی که در حالت s_i' بوده است $V_i(s_i')$ نام‌گذاری می‌شود.

به نحوی مشابه می‌توان $Q_i(s',a_i)$ را به صورت امید یاری‌بهر رفتار B_i در حالت درونی‌ی s' پس از انتخاب عمل a_i بدین‌صورت نوشت:

$$Q_i(s',a_i) = \int_{s\in S^\circ} R(s)p\{B_i \text{ is excited in } s|B_i \text{ selects } a_i\}p\{s|s\in S^\circ\}ds = \quad (۴-۲۶)$$
$$R(s')p\{B_i \text{ is excited in } s'|B_i \text{ selects } a_i\}$$

و ارتباط $V_i(s')$ با $Q_i(s',a_i)$ نیز بدین‌گونه خواهد بود:

$$V_i(s') = \underset{a_i \sim \pi_i(s')}{E}[Q_i(s', a_i)]. \qquad (۴-۲۷)$$

برای این‌که مشخص شود آیا رفتارهای پایینی در حالت‌ای مانند s_i' می‌توانند به‌تر عمل کنند یا این‌که به‌تر است خود آن رفتارْ عمل‌ای پیش‌نهاد کند، می‌بایست بازده رفتارهای پایینی را در حالت‌هایی از فضای اصلی که B_i را در s_i' قرار داده (و در نتیجه B_i تحریک‌شده است) و دست‌کم یکی از آن‌ها (رفتارهای پایینی) تحریک شده باشد با بازده انتخاب یک عمل از B_i مقایسه کرد. این کار با مقایسه‌ی $Q_i(s', a_i)$ و $Q_i(s', NA)$ –که امید یاری‌بهر انتخاب NA و امید یاری‌بهر رفتارهای پایینی است– انجام می‌شود. $Q_i(s', NA)$ مشابه با $Q_i(s', a_i)$ بدین‌گونه مشخص می‌شود:

$$Q_i(s', NA) = \int_{s \in S^\circ} R(s)p\{B_i \text{ is excited in } s | B_i \text{ selects } NA\}p\{s | s \in S^\circ\}ds. \qquad (۴-۲۸)$$

توجه کنید که به هنگام انتخاب NA، B_i کنترل را به رفتارهای پایینی واگذار می‌کند. در نتیجه یاری‌بهر آن رفتار ناشی از این کار است و نه عمل‌ای که خود مستقیم به عامل ارسال کرده است.

برای بهینه عمل کردن می‌بایست ارزش انتخاب NA را با انتخاب عمل‌ای از مجموعه اعمال A_i مقایسه کرد: اگر ارزش عمل‌نکردن بالاتر بود، با شانس کنترل به رفتارهایی پایینی دادن به بازده بالاتری –از دیدی آماری– خواهیم رسید. انتخاب بین اعضای A_i' (که شامل NA نیز می‌شود) به مانند معمول است و می‌توان از روش‌های انتخاب عمل متداولی مانند روش بولتزمان یا $\varepsilon - greedy$ استفاده کرد. در حالت انتخاب عمل حریصانه، شرط زیر به هنگام انتخاب NA برقرار خواهد بود:

$$Q_i(s_i', NA) > Q_i(s', a_i) \quad \forall a \in A_i. \qquad (۴-۲۹)$$

برای به روز رسانی‌ی ارزش‌های حالت–عمل، می‌بایست تخمین‌ای از پاداشِ بازگشته انجام شود. اگر مساله تک–گام باشد، آن‌گاه پاداشِ بازگشته معادل با سیگنال تقویت لحظه‌ای خواهد بود. حالت دیگر وقتی است که پاداشِ بازگشته به صورت متوسط پاداش اپیزود یا پاداش discountشده با افق نامحدود و یا از این دست باشد (به روابط ۴–۲، ۵–۲ و ۶–۲ مراجعه کنید).

در ابتدا حالت پاداشِ بازگشته‌ی معادل با پاداش لحظه‌ای را در نظر می‌گیرم. با توجه به تعریف $Q_i(s',a_i)$ و تعریف تحریک شدن و توجه به این نکته که پاداشِ بازگشته این حالت معادل با سیگنال تقویت لحظه‌ای $r(s)$ است، $Q_i(s',a_i)$ را می‌توان با در نظر گرفتن سیگنال تقویت لحظه‌ای که عامل دریافت می‌کند تخمین زد. اگر B_i رفتار کنترل‌کننده باشد و عمل a_i را انتخاب کرده باشد، یاری‌بهر آن رفتار به طور مستقیم در سیگنال تقویت دریافتی آن لحظه‌ی عامل متجلی می‌شود. پس با داشتن سیگنال تقویت آن لحظه می‌توان تخمین $Q_i(s',a_i)$ را به روز کرد. حال اگر B_j عمل مجازی‌ی NA را انتخاب کند، رفتار پایین‌تری چون B_i کنترل‌کننده می‌شود و سیگنال تقویت $r(s)$ را دریافت می‌کند. با استدلالی به مانند قبل، سیگنال تقویت نتیجه‌ی یاری‌بهر عملِ انجام شده توسط B_i بوده است و در نتیجه یاری‌بهر رفتار کنترل‌کننده مشخص است. اما علاوه بر آن این سیگنال تقویت ناشی از سرکوب نکردن B_j و انتخاب NA نیز بوده است و میزان یاری‌بهر آن رفتار دقیقا معادل با سیگنال تقویت آن لحظه خواهد بود. قواعد به روز رسانی در حالت پاداش لحظه‌ای بدین‌گونه است:

$$\begin{aligned} &Q_{i_{k+1}}(s_i',a_i)=\big(1-\alpha_{k,i}(s_i',a_i)\big)Q_{i_k}(s_i',a_i)+\alpha_{k,i}(s_i',a_i)r(s) \\ &(B_i \text{ is controlling behavior in } s_i' \text{ and selects } a_i) \\ &Q_{j_{k+1}}(s_j',NA)=\big(1-\alpha_{k,j}(s_j',NA)\big)Q_{j_k}(s_j',NA)+\alpha_{k,i}(s_j',NA)r(s) \\ &(\forall B_j; B_j \overset{\mathrm{T}}{\succ} \mathrm{B_i} \text{ and } B_i \text{ is controlling behavior and } B_j\text{s are excited in } s_j' \text{ and select } NA) \end{aligned} \qquad (۴–۳۰)$$

(۴–۳۱)

توجه کنید که به روز رسانی‌ی NA می‌بایست برای همه‌ی رفتارهای تحریک‌شده‌ی بالاتر از رفتار کنترل‌کننده انجام شود. هم از تحلیل این بخش و هم از تحلیل بخش پیش بر می‌آید که رفتارهای پایین‌تر از رفتار کنترل‌کننده تغییری نمی‌کنند.

برای حالت پاداشِ بازگشته‌ی غیرلحظه‌ای (مثلا پاداش متوسط در طول اپیزود)، می‌توان از تخمین مونت‌کارلوی[19] پاداشِ بازگشته استفاده کرد. در این حالت از روش‌های bootstrapping مانند $TD(0)$ ([Sutton88]) به دلیل درست نبودن ارزش حالت بعدی به عنوان تخمین‌ای برای ارزش حالت فعلی نمی‌توان استفاده کرد.

دلیل این موضوع و عدم امکان استفاده از روش‌های bootstrapping تفاوت مناطق تحریک رفتارها و درک متفاوت آن‌ها نسبت به دنیا است. دو رفتار B_1 و B_2 را در نظر بگیرید. آن‌ها به کمک نگاشت‌های M_1 و M_2 دنیا را از طریق بازنمایی درونی‌ی حالت‌های‌شان مشاهده می‌کنند. تعریف $S^\circ(s'_i)$ را به یاد آورید (۲۴-۴). فرض کنید که دو حالت داخلی‌ی آن رفتارها به طور هم‌زمان در زیرمجموعه‌ی $S_1^\circ(s'_i)\cap S_2^\circ(s'_j)$ تحریک شده‌اند. برای راحت‌تر بودن تصور می‌توانید شکل (۱-۴) را ببینید و فرض کنید که درباره‌ی s'_5 از رفتار B_1 و s'_2 از رفتار B_2 صحبت می‌کنم. دیده می‌شود که آن‌ها با هم اندکی هم‌پوشانی دارند ولی قسمت‌های دیگری از آن‌ها جدا از هم است و در نتیجه وقتی یکی از آن‌ها تحریک می‌شود، دیگری ممکن است تحریک نشده باقی بماند. در حالت کلی $s\in S_1^\circ(s'_i)\cap S_2^\circ(s'_j)$ و $B_1\succ B_2$. اینک تصور کنید که $S_1^\circ(s'_i)\cap S_2^\circ(s'_j)\neq S_k^\circ(s'_i)\ \ (k=1,2)$ و همچنین B_1 –با توجه به $Q_1(s',\cdot)$اش تصمیم می‌گیرد که NA انتخاب کند. در نتیجه B_2 که تحریک شده است عمل‌ای پیش‌نهاد می‌کند و کنترل عامل را در اختیار می‌گیرد و در این میان عامل پاداش‌ای (یا تنبیه) دریافت می‌کند. اگر قصد استفاده از bootstrapping را داشته باشیم می‌بایست از $Q_2(s'_2,a_i)$ برای کمک در تخمین $Q_1(s'_1,NA)$ (به عنوان تخمین فعلی از پاداشِ بازگشته شروع شده از حالت فعلی) استفاده کنیم. اما با این‌کار به خطا می‌رویم چون $Q_2(s'_2,a_i)$ متوسط پاداشت بازگشته در زیرفضای مشترک $S_1^\circ(s'_i)\cap S_2^\circ(s'_j)$ نیست بلکه متوسط پاداشِ بازگشته در $S_2^\circ(s'_j)$ است. بدین ترتیب با انجام این کار تخمین نادرست‌ای به دست آمده است. به همین دلیل برای تخمین پاداشِ

[19] Monte Carlo

بازگشته هر حالت–عمل هر رفتار می‌بایست همه‌ی پاداش‌های دریافت شده از آن حالت–عمل را تا انتهای اپیزود نگه‌داری کرد و سپس با استفاده از روش مونت‌کارلو تخمین لازم را به دست آورد – مثلا با استفاده از روش مونت‌کارلو اولین–ملاقات[۲۰].

اگر شخص‌ای قصد پیاده‌سازی روش مونت‌کارلو را داشته باشد، احتمالا بازده سیستم در مقایسه با مسایلی که در آن‌ها bootstrapping ممکن است، کندتر به حد مطلوب‌اش هم‌گرا می‌شود. با وجود این‌که این مساله ناشی از دشواری‌ی ذاتی‌ی یادگیری رفتار در معماری‌ی رفتارگرای سلسله مراتبی‌ای است که هر کدام از رفتارهای‌اش نگاه‌ای مختص به خود به دنیای اطراف دارند، اما به نظر نمی‌رسد مشکل چندان محدودکننده‌ای باشد. بر این اعتقادم که در طراحی‌ی سیستم‌های رفتارگرا فرض داشتن سیگنال تقویت لحظه‌ای کاملا قابل قبول است. یک معماری‌ی رفتارگرا معمولا به عنوان کنترل‌کننده‌ی رآکتیو عامل در نظر گرفته می‌شود و در این شرایط انتظار می‌رود که خوبی یا بدی‌ی واکنش عامل بدون تاخیر مشخص شود و در نتیجه امید دریافت سیگنال تقویت لحظه‌ای برای حالت–عمل‌های مختلف درست همان چیزی است که می‌بایست بهینه شود. با این‌حال، چارچوب معرفی شده برای یادگیری رفتار سلسله مراتبی قابلیت کار با پاداش‌های بازگشته‌ی کلی را نیز دارد.

۴–۵) یادگیری هم‌زمان رفتار و ساختار

در دو فصل اخیر فرمول‌بندی‌ی مناسبی برای یادگیری ساختار و یادگیری رفتار معرفی کردم. ارزش کلِ عامل (V_T) به اجزای ساده‌تری تقسیم شد و علاوه بر آن روش تخصیص اعتبار مناسبی برای یادگیری ساختار و یادگیری رفتار پیش‌نهاد کردم. طراح با توجه به مساله‌ی مورد نظر و دانش اولیه‌ای که در اختیار دارد می‌تواند از یادگیری ساختار، یادگیری رفتار و یا هر دوی آن‌ها به صورت هم‌زمان استفاده کند. استفاده‌ی هم‌زمان بدان معنا است که عامل با توجه به سیگنال تقویت دریافتی، نگاشت مناسب بین ادراک و عمل هر رفتار را بیابد و در همان هنگام ترتیب مناسب آن رفتارها در معماری را نیز بیاموزد (حل مساله‌ی مطرح شده در (۱۳–۲)). البته باید توجه شود که ارزش هر عملِ هر رفتار ($Q_i(s_i, a_i)$) به مکان آن رفتار در ساختار نیز بستگی دارد. در صورت تغییر مکان رفتار در ساختار، دانش پیشین‌اش به

[20] First Visit Monte Carlo Method

- Initialize Learning parameters (different $\{\alpha\}$ depending on the method)
- For a lifetime do
  - Update learning parameters to ensure convergence
  - Select an architecture T^* that maximizes (3-6) (Zero Order representation) or (3-18) (First Order representation) using an optimization method (If the architecture is fixed, skip this step).
  - Using the provided architecture T^*, let the agent interact with the environment for a while
    - Receive reinforcement signal from the critic (that can be external or internal)
    - Update the estimation of necessary values ($\{\widetilde{V}_{ij}\}$ for Zero Order structure learning representation (3-16), $\{\widetilde{V}_{i\succ j}\}$ and $\{\widetilde{V}_{i0}\}$ for First Order structure learning representation ((3-28) and (3-29)), and $\{Q_i(s_i,a_i)\}$ for Behavior learning ((4-30) and (4-31) (immediate reward case) or Monte Carlo method for general return).

جدول ۱-۴. الگوریتم کلی برای نمایش عمل‌کرد عامل‌ای که ساختار و رفتارش را یاد می‌گیرد.

صورت دانش اولیه‌ای برای یادگیری شرایط جدید در می‌آید. در بسیار از تغییر مکان‌ها هم‌چنان اطلاعات مفیدی در ارزش حالت–عمل رفتارها باقی می‌ماند. با این وجود همیشه به هر رفتار اجازه‌ی تطبیق با شرایط جدید داده می‌شود و در نتیجه حتی اگر تخمین ارزش‌اش صحیح نباشد، رفتار می‌تواند با یادگیری‌ی مجدد آن را تصحیح کند. در این فرآیند تطبیق، لزومی به یک‌سان بودن سرعت یادگیری رفتارها و ساختار وجود ندارد: ساختار می‌تواند کندتر از رفتارها تغییر کند (با کم کردن نرخ یادگیری یا به روزرسانی‌ی ساختار در هر چند اپیزود یک‌بار). ایده‌ی پسِ این کار سریع‌تر کردن فرآیند تطبیق رفتارها در هنگام تغییر ساختار است. در نهایت، الگوریتم‌ای که نشان می‌دهد عامل یادگیر چگونه می‌بایست در طول دوره‌ی زندگی با کنش با محیطش ساختار و رفتارش را تغییر دهد در جدول ۱–۴ آمده است.

۵) تکامل رفتار

۵-۱) مقدمه

در این فصل با روی‌کردی تکاملی و جمعیتی به مساله‌ی طراحی‌ی سیستم‌های رفتارگرا می‌پردازم. برای شروع، اشاره مجدد به این نکته مفید است که طراحی‌ی یک سیستم را می‌توان به عنوان یک فرآیند جستجو در میان همه‌ی طراحی‌های ممکن در نظر گرفت. هدف طراحی نیز پیدا کردن عضوی از مجموعه‌ی فضای مساله[۱] است که معیاری از پیش تعیین شده را بهینه کند. در صورت پیدا کردن عضو مطلوب‌ای از آن فضا، می‌توان ادعا کرد که به پاسخ مناسب‌ای برای مساله‌ی طراحی رسیده‌ایم. روش‌های بسیاری به طور صریح یا ضمنی به بهینه‌سازی‌ی معیاری می‌پردازند. به عنوان مثال در شبکه‌های عصبی‌ی با سرپرست[۲] معیاری از خطای پیش‌بینی حداقل می‌شود. از همین دیدگاه بهینه‌سازی می‌توان به یادگیری‌ی تقویتی نگریست. یادگیری‌ی تقویتی معیاری از پاداش‌ها و تنبیه‌های دریافتی را حداکثر می‌کند. این معیار می‌تواند متوسط، متوسط وزن‌دار و یا مقدار لحظه‌ای پاداش‌ها و تنبیه‌های عامل باشد. این خاصیت یادگیری‌ی تقویتی را می‌توان به این صورت تفسیر کرد که یادگیری تقویتی تخمین‌ای از برنامه‌ریزی پویا[۳] است. با این دید، هدف فصل‌های پیش پیدا کردن پاسخ بهینه‌ای برای مساله‌ی طراحی عامل‌ای با معماری‌ی رفتارگرای سلسله مراتبی با استفاده از فرمول‌بندی‌ی خاص یادگیری‌ی تقویتی بود.

برای هدف ما –طراحی‌ی عامل‌ای رفتارگرا– یادگیری‌ی تقویتی تنها و یا به‌ترین گزینه نیست. هم‌گرایی روش‌های متداول در یادگیری‌ی تقویتی در همه‌ی مسایل تضمین شده نیست. برای مسایل‌ای که از دید عامل مارکوف نباشد و یا از تقریب‌زن‌های عمومی‌ای[۴] چون شبکه‌های عصبی برای نگه‌داری تابع ارزش (یا ارزش–عمل) استفاده شده باشد، رسیدن به نقطه‌ی بهینه و یا حتی هم‌گرایی به پاسخ‌ای تضمین شده نیست. به عنوان نمونه‌ای از قیدهای لازم برای هم‌گرایی، می‌توان به Q-Learning اشاره

[1] Problem Space
[2] Supervised
[3] Dynamic Programming
[4] General Function Approximator

کرد که تنها در صورتی به پاسخ بهینه هم‌گرا می‌شود که مساله از دید عامل مارکوف باشد، بازنمایی تابع ارزش–عمل به صورت جدول ارجاعی و البته کاوش عامل نیز به میزان کافی صورت گرفته باشد (مثلا شرط GLIE[5] برقرار باشد [Jaakkola94]). مثال‌های ساده‌ای از مسایل غیرمارکوف وجود دارد که باعث واگرایی پروسه‌ی یادگیری می‌شوند. هم‌چنین مشکل‌دار بودن استفاده از تقریب‌زن‌های عمومی‌ای مانند شبکه‌های عصبی نیز کاملا شناخته شده است ([Thrun93] و [Boyan95]). یا به عنوان نمونه‌ای دیگر می‌توان به روش‌های سیاست‌محور[6]ای اشاره کرد که به طور مستقیم سعی در پیدا کردن بهینه‌ی سیاست عامل با استفاده از روش‌های مبتنی بر گرادیان می‌کنند (به عنوان مثال به [Peshkin01] مراجعه کنید). این روش‌ها که اخیرا بسیار در حل مسایل POMDP به کار می‌رود و نتایج موفق‌ای نیز داشته‌اند (مثلا [Kohl04]) با وجود آن‌که در طیف وسیع‌تری از شرایط هم‌گرا می‌شوند ولی تنها هم‌گرایی‌ی آن‌ها به بهینه‌ی محلی تضمین شده است. به همین دلایل است که توجه به روش‌های دیگری که بتوانند این مسایل را حل کند توجیه‌پذیر خواهد بود.

روش‌های تکاملی به عنوان روش جستجوی‌ای که از طبیعت الهام گرفته شده است، گزینه‌ی مطلوبی برای حل بعضی از این مشکلات است. استفاده از روش‌های تکاملی نه تنها از لحاظ زیست‌شناختی توجیه‌پذیر است، بلکه به دلیل قابلیت‌های ویژه‌ی آن‌ها گزینه‌ی بسیار مطلوبی به نظر می‌آید. به عنوان مثال، ویژگی‌های زیر از خواص مطلوب روش‌های تکاملی‌اند:

- جستجوی آن‌ها کلی[7] است و نه محلی[8]: در نتیجه احتمال گیرافتادن در نقاط بهینه‌ی محلی کم‌تر است.
- نیازی به خوش‌تعریف بودن معیار مورد نظر ندارد: لزومی به مشتق‌پذیر بودن و یا حتی پیوسته‌بودن معیار مورد نظر نیست.

[5] Greedy In Limit with Infinite Exploration
[6] Policy-based
[7] Global
[8] Local

در نتیجه در صورت به‌کارگیری‌ی روش‌ای تکاملی برای جستجوی فضای مساله‌مان دیگر چندان نگران گیرافتادن در ماکزیمم‌های محلی‌ی تابع ارزش نخواهیم بود. هم‌چنین به دلیل آن‌که در اکثر مسایل متداول برای عامل‌های در محیط واقعی قرارگرفته، تابع ارزش نسبت به پارامترهای توصیف‌کننده‌ی آن مشتق‌پذیر نیست، استفاده از روش‌ای که نیاز به وجود مشتق نداشته باشد بسیار مطلوب می‌نماید. البته در به کارگیری این روش‌ها نیز می‌بایست توجه کافی به خرج داد. روش‌های تکاملی نیز دارای مشکلاتی چون هم‌گرایی زودهنگام[9] و یا گم‌راه کننده[10] بودن مساله دارند که باعث دشواری و یا نرسیدن به پاسخ مطلوب می‌شود. درست به همین دلیل است که پژوهش‌های بسیار زیادی در همین زمینه برای گسترش و تقویت روش‌های اولیه‌ی الهام‌گرفته شده از طبیعت در بین پژوهش‌گران آن رشته انجام می‌شود.

همان‌طور که در فصل اول توضیح دادم، استفاده از روش‌های تکاملی در روباتیک چندان تازه نیست. اکثر این روش‌ها سعی در پیدا کردن کنترلری یک‌پارچه برای تامین هدف مورد نظر دارند. این کنترل در اکثر موارد بوسیله‌ی شبکه‌ای عصبی توصیف می‌شود و نقش تکامل نیز پیدا کردن پارامترهای مطلوب است. تابع هدف می‌تواند از فرم‌های ساده‌ای مانند امتیاز منفی دادن به برخورد با اجسام تا فرم‌های پیچیده‌تری مانند امتیاز دادن به رفتار حرکت به سمت جسم‌ای مشخص باشد. با این‌همه یکی از بزرگ‌ترین ایرادهای روش‌های تکاملی‌ی فعلی، مقیاس‌پذیر[11] نبودن آن‌ها است. به‌کارگیری‌ی روش‌های تکاملی در مسایل اندکی پیچیده بسیار دشوار می‌شود. هم‌چنین به دلیل تمایل کم و بیش همیشگی به بهره‌گیری از شبکه‌های عصبی به عنوان کنترلر روبات، قابلیت استفاده‌ی مجدد[12] از کنترلرهای تکامل یافته به صورت ماجول‌های از پیش موجود بسیار کم و یا حتی غیرممکن است. البته در این میان تلاش‌هایی برای حل این قبیل مشکلات و حل مسایل پیچیده‌تر با استفاده از روش‌های تکاملی انجام گرفته است که می‌توان به روی‌کرد ماجولار و روی‌کرد افزایشی[13] –افزایش تدریجی‌ی پیچیدگی‌ی تابع سازگاری– اشاره کرد [Togelius04].

[9] Premature Convergence
[10] Deceptive
[11] Scalable
[12] Reusability
[13] Incremental

مطلوب است که متدولوژی‌ی ما برای طراحی‌ی یک معماری‌ی رفتارگرا ویژگی‌های زیر را تامین کند:

- به سرعت به پارامترهای کنترلر مطلوب هم‌گرا شود.
- سرعت واکنش‌اش به تغییرات سریع محیط بالا باشد.
- قابلیت کار در معماری‌ای سلسله مراتبی و ایجاد تصمیم‌گیری‌های چندسطحه را داشته باشد.
- طراحی به ماجول‌های مجزا و قابل استفاده‌ی مجددی بینجامد.

مکانیزم‌های متداول در روباتیک تکاملی عموما کندند. در اکثر آن‌ها کنترلر در طول دوره‌ی زندگی‌ی یک عامل ثابت است. پس در صورت تغییر محیط، کنترلر نمی‌تواند خود را با آن هم‌گام کند. ایده‌های یادگیری‌ای که در فصل‌های پیش مطرح شد چنین عیب‌ای ندارد ولی متاسفانه –به دلایل توضیح داده شده– همیشه قادر به پیدا کردن به‌ترین کنترلر نیست. در نتیجه، استفاده‌ی هم‌زمان از یادگیری و تکامل –به طور هم‌زمان– جالب توجه می‌نماید. اضافه بر این، اگر ساختار تکاملی به گونه‌ای باشد که بتواند کنترلرهای سلسله مراتبی تولید کند، آن‌گاه با چارچوب مطرح شده در این پایان‌نامه سازگار خواهد بود و ویژگی‌هایی چون اولویت‌بندی، تصمیم‌گیری در سطوح مختلف تجرید زمانی و مکانی و ... را دارا خواهد بود. هم‌چنین در صورت ماجولار بودن، خاصیت استفاده‌ی مجدد نیز پیدا خواهد کرد.

با در نظر گرفتن این نکات، سعی در استفاده از ایده‌های تکاملی کردم. در فصل‌های پیشین مساله‌ی یادگیری را به دو بخش مجزای یادگیری ساختار و یادگیری رفتار تقسیم کردم. در یادگیری ساختار فرض بر این بود که تعدادی رفتار *مناسب* از پیش موجود است و هدف پیدا کردن ترتیب بهینه برای آن‌ها است. در یادگیری‌ی رفتار، ساختار از پیش فرض می‌شد و هدف به پیدا کردن نگاشت بین ورودی و خروجی‌ی هر رفتار فروکاسته می‌شد.

در این پژوهش تصمیم بر آن شد که تغییر رفتارها به عهده‌ی فرآیندی تکاملی باشد. به عبارت دیگر، به جای استفاده از *یادگیری ساختار/یادگیری رفتار* در معماری‌ای که همه‌ی اجزای آن تغییر

می‌کند از *یادگیری ساختار/تکامل رفتار* استفاده شود. علت این انتخاب این فرض است که یک عامل با داشتن مجموعه‌ای کافی از رفتارها و استفاده از سیگنال تقویت می‌تواند به‌ترین ساختار ممکن را به سرعت و در طی زندگی‌اش به دست بیاورد در حالی که پیدا کردن نگاشت مناسب هر رفتار به دلیل آن‌که فضای رفتارها معمولا بزرگ‌تر از فضای ساختارها است، معمولا کندتر است. فرض کنید ساختار T دارای m لایه باشد که می‌بایست یکی از n رفتار موجود در آن انتخاب شود. در قسمت‌های پیش داشتیم که کاردینالیته‌ی فضای بازنمایی برای بازنمایی‌های یادگیری‌ی ساختارمان بدین‌گونه‌اند (تکرار ۳-۷ و ۳-۲۳):

$$\text{cardinality(ZO)} = n \cdot m \qquad (۵-۱)$$

$$\text{cardinality(FO)} = n^2 + n \qquad (۵-۲)$$

در مسایل متداول، n و m چندان بزرگ نیستند؛ مثلا در مساله‌ی بلندکردن جسم به صورت چندروباته که در فصل مربوط به آزمایش‌ها (فصل ۶) توضیح خواهم داد هر دوی این‌ها ۵اند و در نتیجه فضای جستجو خیلی بزرگ نیست. به همین دلیل می‌توان انتظار داشت که یادگیری ساختار بتواند در طول زندگی‌ی عامل به ساختار مناسبی دست‌پیدا کند. از طرف دیگر، در بسیاری از مسایل ممکن است ابعاد فضای جستجو برای رفتارها بسیار بزرگ باشد. به عنوان مثال رفتاری که در $|S|$ حالت مختلف واکنش نشان بدهد و دارای $|A|$ عمل مختلف است، دارای فضای جستجویی با بعد $|S|\cdot|A|$ خواهد بود. این‌که در مسایل عملی این مقادیر چقدرند موضوعی است که نیاز به بررسی‌ی چندین نوع مساله‌ی مختلف دارد اما من فرض را بر این گذاشتم که این بعد در اکثر مسایل بزرگ‌تر است و در نتیجه به احتمال زیاد مساله دشوارتر است. البته لازم به ذکر است که بزرگ‌بودن فضای بازنمایی مساله به تنهایی بیان‌گر میزان دشواری‌ی آن نیست. برای قضاوت در مورد پیچیدگی، ابتدا می‌بایست معیار پیچیدگی مشخص شود (مثلا پیچیدگی‌ی تعداد نمونه‌های لازم برای یادگیری با دقت مشخص (پیچیدگی

نمونه[۱۴]) اهمیت دارد یا مثلا کیفیت به‌ترین پاسخ‌ای که می‌توان به آن رسید) که بررسی‌ی دقیق‌ای در این‌باره صورت نگرفته است.

با این انتخاب و در صورت درست بودن فرض‌های انجام شده، به مکانیزمی می‌رسیم که ساختار معماری توسط یادگیری تغییر می‌یابد و مجموعه‌ی رفتارها نیز توسط تکامل. اما برای سازگار بودن با چارچوب مطرح در این پژوهش می‌بایست به این نکته نیز توجه کرد که لازم است حاصل تکامل توسط مکانیزم یادگیری‌ی ساختار قابل به کارگیری باشد. استفاده از ایده‌های متداول در اکثر پیاده‌سازی‌های روباتیک تکاملی —که به صورت کنترلر شبکه‌ی عصبی‌ای یک‌پارچه پیاده‌سازی می‌شوند – قابل استفاده نیست. عناصر اولیه یادگیری‌ی ساختار، مجموعه‌ای از رفتارها $\{B_i\}$ هستند و نه کنترلری که ترکیب همه‌ی این‌ها با هم باشد. به همین دلیل، تصمیم گرفتم تا روش مورد نظر به جای تکامل کل رفتار عامل با هم، به تکامل تک‌تک رفتارها به طور جداگانه بپردازد. در نتیجه حاصل تکامل نه یک کنترلر یک‌پارچه که مجموعه‌ای از رفتارهای مجزا خواهد بود. به همین دلیل ایده‌های سلسله مراتبی‌ی مطرح شده به همان شکل قبل قابل به کارگیری خواهد بود و هم‌چنین طراح در نهایت به مجموعه‌ای از رفتارها می رسد که که قابلیت استفاده‌ی مجدد دارند. بدین ترتیب، ایده‌ی کلی‌ی استفاده از تکامل برای ایجاد رفتارهای مناسب ایجاد شد. در بخش بعد، به توضیح جزییات این ایده می‌پردازم.

۵-۲) تکامل رفتارها

فرض کنید که می‌خواهیم معماری‌ی رفتارگرایی برای عامل بسازیم. برای این‌کار می‌بایست مجموعه‌ی مناسب‌ای از رفتارها $\{B_i\}$ و هم‌چنین ساختار مناسب T را بیابیم به طوری که اهداف‌مان برآورد شود. اگر هدف را به صورت تابع تقویت $R: s \times a \times s' \mapsto \Re$ (یا هر فرم مشابه دیگر) در نظر بگیریم، آن‌گاه هدف‌مان بیشینه کردن این تابع —یا تابع‌ای از این تابع[۱۵]– خواهد بود. تعریف V_T (۲-۷) را به یاد بیاورید. هدف طراحی‌ی یک معماری، پیدا کردن ساختار و رفتارهای مناسب است (۲-۱۳). مشاهده

[14] Sample Complexity

[15] در شرایطی که از حالت‌های کلی‌ی پاداش بازگشته استفاده می‌کنیم که خود تابع‌ای از سیگنال تقویت است.

می‌شود که دو مجموعه پارامتر مختلف برای بهینه‌سازی وجود دارد: پارامترهای بیان‌گر هر رفتار و پارامتر بیان‌گر ساختار.

اگر روش‌های حل مساله را به دو دسته‌ی روش‌های یادگیر و روش‌های تکاملی تقسیم کنیم، آن‌گاه یکی از روی‌کردهای زیر را می‌توان برگزید:

- یادگیری ساختار – یادگیری رفتار
- تکامل ساختار – یادگیری رفتار
- تکامل ساختار – تکامل رفتار
- یادگیری ساختار – تکامل رفتار
- یادگیری و تکامل ساختار – یادگیری و تکامل رفتار (و ترکیب‌های مشابه)

البته از ذکر حالت‌های بدیهی‌ای مانند ثابت فرض کردن ساختار و یادگیری (یا تکامل) رفتار و برعکس بدلیل آن‌که تنها سعی در حل بخشی از کل مساله می‌کنند صرف‌نظر شده است. همان‌طور که در مقدمه گفته شد، به نظر می‌رسد در اکثر موارد رفتارها آن بخش‌ای از مساله‌اند که تغییرات آن‌ها باعث تغییرات کم‌تری در رفتار کلی‌ی عامل می‌شود در حالی که تغییر کوچک‌ای در ساختار (جابه‌جا شدن دو رفتار یا اضافه شدن یک رفتار جدید) باعث تغییرات بسیار در رفتار کلی‌ی عامل می‌شود. هم‌چنین به دلیل آن‌که در اکثر مسایل فضای جستجوی رفتارها بسیار بزرگ‌تر از فضای جستجوی ساختار است، ایجاد تغییرات مورد نظر در سطح رفتارها نیاز به تغییرات بسیار بیش‌تری نسبت به تغییر در ساختار دارد و در نتیجه سرعت واکنش عامل به تغییرات محیط بسیار کم‌تر خواهد بود. به عبارت دیگر ساختار بخش‌ای از عامل است که تغییرات سریع از آن انتظار می‌رود و رفتارها بخش کندتر آن هستند. به همین دلیل، تصمیم بر آن شد که رفتارها تکامل یابند و ساختار یادگرفته شود. توضیح یادگیری ساختار در فصل‌های پیش آمده است و دوباره تکرار نمی‌شود. اینک به تکامل رفتار می‌پردازم.

۵-۲-۱) تکامل هم‌کارانه‌ی رفتارها[16]

برخلاف اکثر روش‌های متداول تکامل در روباتیک، در روش‌ای که به کار گرفته‌ام کلِ کنترلر یک‌جا تکامل نمی‌یابد بلکه کنترلر عامل به اجزای رفتاری تقسیم شده و هر رفتار به طور جداگانه تکامل می‌یابد. رفتارها با توجه به نقش از پیش مشخص‌شده‌شان جداگانه در مخزن ژنتیکی‌ی[17] خویش تکامل می‌یابند و تمام رفتارهای مرتبط —مانند انتخاب به‌ترین‌های هر جمعیت و تزویج و جهش ژنتیکی– در همان بستر و جدا از دیگر رفتارها انجام می‌شود. به عنوان مثال اگر در مساله‌ی بلند کردن جسم توسط چندروبات، دو رفتار بلند کردن و تنظیم زاویه‌ی جسم وجود داشته باشد دو جمعیت مختلف از رفتارها خواهیم داشت که رقابت بین اعضای آن و عملیات ژنتیکی‌شان جداگانه انجام می‌شود. طراح از پیش مشخص می‌کند که بازنمایی هر رفتار به چه صورت است، سپس جمعیت‌ای از اعضایی با همان بازنمایی برای هر رفتار ایجاد می‌کند و پس از آن هر جمعیت با توجه به سازگاری‌ی اعضای‌اش عملیات ژنتیکی را انجام می‌دهد.

البته هنوز نکات بسیاری می‌بایست مطرح شود تا این مکانیزم قابل پیاده‌سازی باشد:

- چگونه با استفاده از این جمعیت‌های رفتاری می‌توان یک عامل خاص را به وجود آورد؟
- ارتباط سازگاری‌ی عامل و سازگاری‌ی هر کدام از رفتارهای تشکیل‌دهنده‌اش چگونه است؟
- اپراتورهای ژنتیکی هر رفتار چگونه هستند؟

نحوه‌ی ارتباط این سیستم تکامل هم‌کارانه‌ی چندجمعیتی و عامل که الهام گرفته شده از Enforced Sub-Population (ESP) است [Gomez97] بدین‌صورت است: پس از ایجاد تعدادی جمعیت برای هر رفتار، یکی از رفتارهای هر کدام از جمعیت‌ها را به طور اتفاقی انتخاب می‌کنیم. در نتیجه اگر n جمعیت $\mathrm{B_i}$ داشته باشیم، مجموعه‌ای nتایی $\{B_i\}$ خواهیم داشت که در آن $B_i \in \mathrm{B_i}$. این مجموعه به عامل داده می‌شود و عامل با استفاده از این مجموعه می‌تواند با محیطش ارتباط برقرار

[16] Cooperative Evolution of Behaviors
[17] Genetic Pool

کند. اگر عامل از یادگیری ساختار بهره ببرد می‌تواند به‌ترین ترتیب این رفتارها را بیابد و اگر نه از ساختار از پیش تعیین‌شده‌ای استفاده می‌کند. بدیهی است که کیفیت کار عامل به کیفیت هر کدام از رفتارها بستگی دارد: اگر رفتارهای "مناسب"ای در اختیار عامل باشد، عامل می‌تواند بازده بالایی داشته باشد (یا به عبارت دیگر، رفتار کلی‌ی سازگارانه‌تری نسبت به محیط داشته باشد)؛ ولی اگر رفتارهای "مناسب"ای در اختیارش قرار داده نشود، عامل حتی با وجود یادگیری ساختار نیز نمی‌تواند به بازده بالایی برسد. می‌توان نتیجه گرفت که کیفیت رفتار کلی‌ی عامل به کیفیت هر کدام از رفتارهای‌اش ارتباط دارد. البته این‌که میزان یاری‌بهر[۱۸] هر رفتار در رفتار کلی‌ی عامل چگونه است مشخص نیست – بعضی از رفتارها ممکن است تاثیر بیش‌تری داشته باشند و بعضی تاثیر کم‌تری، یا حتی بعضی رفتارها ممکن است بتوانند اثر سوء رفتارهای دیگر را بپوشانند یا برعکس. در این پژوهش دو مکانیزم مختلف برای ارزیابی‌ی تاثیر هر رفتار پیش‌نهاد شده است که اینک آن‌ها را توضیح خواهم داد.

۵-۲-۲) اشتراک سازگاری یک‌نواخت

از آن‌جا که کیفیت و سازگاری‌ی عامل به سازگاری هر کدام از رفتارهای تشکیل‌دهنده‌اش بستگی دارد، می‌توان سازگاری‌ی هر کدام از آن رفتارها را متوسطی از سازگاری‌ی همه‌ی عامل‌هایی دانست که آن رفتار در آن‌ها نقش داشته است. به عبارت دیگر اگر نمونه‌ای[۱۹] خاص از یک جمعیت مانند B_i^j (عضو jام از جمعیت iام) در p عامل مختلف قرار بگیرد و آن عامل‌ها دارای سازگاری‌های $f_i \ (i:1\cdots p)$ باشند، آن‌گاه سازگاری‌ی این نمونه‌ی خاص رفتار به صورت زیر خواهد بود:

$$f^u\left(B_i^j\right)=\frac{1}{p}\sum_{i=1}^{p} f_p \,. \qquad (۵-۳)$$

برای f_i می‌بایست به کیفیت کار عامل در دوره‌ی زندگی‌اش توجه کرد. این کیفیت کار به میزان پاداش و تنبیه‌ای که دریافت می‌کند بستگی دارد. اگر طراح بخواهد همان معیاری را که در طول یادگیری سعی

[18] Contribution
[19] Instance

در بیشینه کردن‌اش داریم با روند تکامل بهینه کند، آن‌گاه می‌بایست سازگاری‌ی عامل را نیز از همان پاداش‌ها و تنبیه‌ها استخراج کند[۲۰]. برای این منظور می‌توان نوشت:

$$V_{\{B\}}\Big|_{\text{Last K episodes}} = f_{\{B\}}\Big|_{\text{Last K episodes}} = E\left[\frac{1}{K}\sum_{t\in\text{Last K episode}} r_t \;\Big|\text{the agent with } \{B\}, \text{t} \in \text{Last K episode}\right] \qquad (۵-۴)$$

که بیان‌گر میزان دریافتی‌ی سیگنال تقویت در K اپیزود آخر است. مقدار K هر چقدر کوچک‌تر باشد نشان‌دهنده‌ی تمایل بیش‌تر به کیفیت نهایی‌ی عامل است و هر چقدر بزرگ‌تر باشد –تا حدی که در نهایت کل اپیزودهای دوره‌ی زندگی‌ی عامل را در بر بگیرد– بیش‌تر به کیفیت عامل در کل دوره‌ی زندگی‌اش نزدیک می‌شود. در این صورت سازگاری‌ی هر نمونه‌ی رفتار در *اشتراک سازگاری‌ی یک‌نواخت* به صورت زیر تعریف می‌شود:

$$f^u\left(B_i^j\right) = \frac{1}{N}\sum_{\{B\}_i} V_{\{B\}_i}\Big|_{\text{Last } K \text{ episodes}} \qquad (۵-۵)$$

که $\{B\}_i$ها N مجموعه رفتار تصادفی انتخاب شده‌ای هستند که $B_i^j \in \{B\}_i$. از آن‌جا که $\{B\}_i$ها به طور تصادفی انتخاب شده‌اند ممکن است بعضی از رفتارهای جمعیت بیش از بقیه شانس شرکت در مجموعه‌ی رفتارهای عامل را داشته باشند. اما در صورتی که تعداد آزمون[۲۱]ها زیاد باشد می‌توان مطمئن بود که هر رفتار به میزان کافی در عامل‌هایی با ترکیب رفتارهای مختلف شرکت جسته و تخمین به دست آمده معیار قابل قبولی از سازگاری‌ی آن رفتار است.

[20] البته طراح می‌تواند معیارهای دیگری را نیز در نظر بگیرد. آن‌گاه مساله‌ی بهینه‌سازی‌ای که بخش‌ای از آن (ساختار) در فرآیند یادگیری حل می‌شود دیگر با مساله‌ی بهینه‌سازی‌ای که بخش دیگر آن (رفتارها) توسط تکامل حل می‌شود یک‌سان نخواهد بود. با این‌حال طراح می‌تواند معیارهای دیگری مانند یادگیری‌ی سریع‌تر و ... را نیز در چارچوب معرفی شده در این فصل در نظر بگیرد.

[21] Trial

۵-۲-۳) اشتراک سازگاری ارزش‌محور

اشتراک سازگاری یک‌نواخت بر پایه‌ی این فرض بود که سازگاری‌ی عامل و سازگاری‌ی هر کدام از رفتارهای درونی‌اش هم‌بسته‌اند و می‌توان سازگاری‌ی هر رفتار را میانگین‌ای از سازگاری‌ی همه‌ی عامل‌هایی دانست که آن رفتار در آن‌ها دخیل بوده است. می‌توان گفت رگه‌هایی از دیدگاه روان‌شناسی‌ی رفتارگرایانه در آن نوع اشتراک سازگاری وجود دارد که تلاش‌ای بر کنکاشِ درونی‌ی ذهنِ عامل ندارد. اما اگر به نحوی بتوان به دانش درونی‌ی عامل پی برد و میزان یاری‌بهر هر رفتار در سازگاری‌ی عامل را تشخیص داد، آن‌گاه ممکن است به نتایج به‌تر و دقیق‌تری در اشتراک سازگاری برسیم. خوش‌بختانه این اطلاعات در بازنمایی‌ای که استفاده می‌کنیم وجود دارد. بازنمایی مرتبه صفر را در نظر بگیرید. یاری‌بهر هر رفتار B_i برای عامل معادل کلِ ارزش‌ای است که در لایه‌های مختلفِ ساختار کسب کرده است. یعنی:

$$V_{T(B_i)} = \sum_{q=1}^{m} P\{B_i \mid L_q\} V_{qi} P\left(L_q \text{ is controlling}\right) = \sum_{q=1}^{m} \widetilde{V}_{qi} \qquad (۵-۶)$$

پس می‌توان یاری‌بهر هر رفتار در معماری‌ی T را از روی داده‌های بازنمایی مرتبه صفر تخمین زد. با توجه به این‌که هر نمونه‌ی رفتار B_i^j در چندین و چند معماری به کار می‌رود، عاقلانه است که سازگاری‌ی آن را به مانند روش پیشین با متوسط‌گیری بر روی چند رفتار محاسبه کنیم. به عبارت دیگر با همان تعاریف و علایم می‌توان نوشت:

$$f^v\left(B_i^j\right) = \frac{1}{N} \sum_{\{B\}_i} V_{T(B_i)} \qquad (۵-۷)$$

توجه کنید که تعاریف به کاررفته سخت و انعطاف‌ناپذیر نیستند. مثلا می‌توان K در (۵-۴) را به‌گونه‌ای عوض کرد که همه‌ی اپیزودها را شامل شود و معیار سازگاری، معیار سازگاری‌ی کلِ دوره‌ی زندگی‌ی عامل شود و یا این‌که آن را به چند اپیزود آخر محدود کرد تا معیاری از عمل‌کرد نهایی عامل باشد.

مشابه همین انعطاف برای اشتراک سازگاری ارزش‌محور نیز برقرار است. به عنوان نمونه اگر مقدار $\alpha_{n,ij}$ در به روزرسانی‌ی $\widetilde{V}_{ij}$ به صورت $\alpha_{n,ij}=1/n$ باشد، آن‌گاه $\widetilde{V}_{ij}$ متوسط ارزش رفتار در لایه L_j خاص از معماری خواهد بود و متوسط‌گیری مطابق (۶–۵) برابر متوسط یاری‌بهر رفتار در طول زندگی‌ی عامل خواهد بود. هم‌چنین می‌توان تخمین ارزش‌های مرتبه صفری که برای یادگیری ساختار به کار می‌رود متفاوت با تخمین‌ای که برای محاسبه سازگاری هر رفتار به کار می‌رود انتخاب کرد.

۵–۲–۴) اپراتورهای ژنتیکی

انتخاب اپراتورهای ژنتیکی وابسته به نوع بازنمایی‌ی دانش‌ای‌اند که برای حل مساله به کار می‌رود. برای شبیه‌سازی‌های این بخش از بازنمایی جدول ارجاعی استفاده کرده‌ام. بازنمایی داخلی رفتار B_i (و در نتیجه B_i^j که نمونه‌ای از آن خانواده رفتار است) بدین‌گونه است:

$$B_i(s'):S'_i\rightarrow A'_i. \qquad (۵–۸)$$

در نتیجه با دانستن این نگاشتِ حالت به عمل و دانستن حالت فعلی‌ی سیستم، عمل مناسب (یا NA) قابل انتخاب است. به مانند قبل هر رفتار متناسب با حالتِ سیستم می‌تواند تحریک شده یا نشده و متناسب با نوع نگاشت‌اش (انتخاب NA و یا غیر از آن) فعال شود یا نشود. این‌که کدام رفتار کنترل عامل را در اختیار بگیرد بستگی به فعال شدن یا نشدن بقیه‌ی رفتارهای معماری دارد. بُعد این نگاشت متناسب با بُعد حالت و بُعد خروجی‌ی رفتار است که توسط طراح مشخص می‌شود. در پیاده‌سازی‌هایی که انجام داده‌ام هر کدام از B_i^j به صورت ماتریس چندبعدی‌ی باینری‌ای ذخیره شده‌اند که فعال بودن بیت‌ای در آن معادل انتخاب عمل و فعال نبودن آن معادل انتخاب NA در آن حالت است. اپراتورهای ژنتیکی متناسب با این ساختار ویژه انتخاب شده‌اند. در این پژوهش دو نوع اپراتور ژنتیکی‌ی جهش و تزویج مخصوص فرم ماتریسی‌ی بازنمایی رفتار معرفی شده است.

اپراتور جهش به دو گونه‌اند: سخت و نرم. جهش سخت که با احتمال $p_{m_{hard}}$ انتخاب می‌شود، یک ماتریس B_i^j را با ماتریس تصادفی‌ی کاملا جدیدی جای‌گزین می‌کند. جهش نرم اختلال‌ای[۲۲] در هر ماتریس ایجاد می‌کند که این اختلال معادل جستجویی در اطراف پاسخ پیشین در فضای پاسخ‌ها است. در پیاده‌سازی‌ی باینری‌ی انجام گرفته، این اختلال با عوض کردن چند بیت در بازنمایی رفتار صورت می‌گیرد.

اپراتور تزویج بدین‌صورت تعریف شده است:

$$B_i^{j^{new}} = XB_i^{j^{old}} + \overline{X}B_i^{k^{old}} \qquad (۵-۹)$$

که در آن X ماتریس تصادفی‌ای با ابعاد مناسب است که یک بودن درایه‌های آن حاصل آزمون تصادفی‌ی مستقل از هم با توزیع یک‌سان[۲۳] برنولی[۲۴] با احتمال p_c است؛ $P(X_{ij\ldots l}=1)=p_c$.

۵-۳) الگوریتم ممتیک[۲۵]

در بخش پیش چارچوب‌ای کلی مبتنی بر یادگیری و تکامل برای ایجاد خودکار معماری‌ای رفتارگرا پیش‌نهاد کردم. تکامل به عنوان جزء کند ولی همه‌جانبه‌نگر به جستجوی فضای بزرگ رفتارها می‌پردازد و یادگیری به عنوان جزء سریع ولی محدودتر ترتیب مناسب رفتارها را در ساختار می‌یابد. می‌توان این مجموعه ساز و کار را تقلیدی از طبیعت دانست که در آن هم تکامل و هم یادگیری در تطبیق موجودات با محیطشان نقش دارند.

در مکانیزم مطرح شده پس از پایان عمر موجود، همه‌ی تجربیات‌اش از بین می‌رود و تنها تاثیر یادگیری بر سازگاری‌اش باقی می‌ماند. هر موجود جدیدی که در محیط شروع به فعالیت می‌کند بدون هیچ تجربه‌ی قبلی‌ای و تنها با اتکا با ذخیره‌ی ژنی‌اش شروع به یادگیری و کنش با محیط می‌کند.

[22] Perturbation
[23] Independent Identical Distribution (i.i.d.)
[24] Bernoulli
[25] Memetic Algorithm

استفاده‌ی بیش‌تر از تجربیات موجود در طول زندگی و انتقال آن به نسل‌های بعد وسوسه‌برانگیز می‌نماید. هدف این بخش پیش‌نهاد روش‌ای تازه برای استفاده از این دانش است. به دلیل شباهت کلی‌ی روش با ایده‌های مطرح در زمینه‌ی پژوهشی‌ی الگوریتم‌های ممتیک، نام این بخش را چنان انتخاب کردم، اگرچه وجود نام مشترک الزاما به معنای تفسیر دقیقا یک‌سان نیست.

الگوریتم‌های ممتیک [Moscato92] بر پایه‌ی ایده‌ی مم[۲۶] بنا شده‌اند. مم *واحد اطلاعات‌ای است که به هنگام ارتباط مردم با هم بازتولید می‌شود* [Dawkins76]. اما با وجود تعریف گسترده‌ی مم، در جامعه‌ی محققان محاسبات تکاملی، الگوریتم ممتیک اغلب به صورت ترکیب‌ای از روش‌ای تکاملی و جستجویی محلی دانسته می‌شود. بدین نحو که با استفاده از روش‌های مختلف جستجوی محلی که برای مساله‌ی مورد نظر طراحی شده است نتیجه‌ی حاصل از فرآیند تکامل را بهبود می‌بخشند. این نوع پیاده‌سازی‌ی مم حاصل برداشت‌ای از مم است که در آن بخش تکاملی‌مانند الگوریتم، دیگر فرآیندی ژنتیکی نیست بلکه شبیه‌ساز بستر جابه‌جایی مم‌ها است و جستجوی محلی نیز معادل با فرآیند یادگیری و تطبیق‌ای است که هر فرد جامعه بر واحدهای اطلاعاتی‌ای که با آن سر و کار دارد اعمال می‌کند. پس روش‌های متداول در تکامل در این‌جا بر فرهنگ‌ها و ایده‌ها اعمال می‌شوند و نه بر ژن‌ها. بهبود فردی‌ای که به عنوان بخش‌ای از الگوریتم ممتیک صورت می‌گیرد و معمولا به صورت جستجویی محل‌ای پیاده‌سازی می‌شود ممکن است در مراحل مختلفی از فرآیند بهینه‌سازی رخ دهد: پیش از محاسبه‌ی سازگاری و بدون تاثیر بر آن‌چه مشابه با ژنوتایپ[۲۷] در الگوریتم‌های تکاملی است، پس/پیش از اعمال اپراتورهای تکاملی و ... . با این دیدگاه، الگوریتم‌های ممتیک نام دیگر ترکیب‌های جستجوی محلی/تکاملی‌ای مانند تکامل بالدوینی[۲۸] و تکامل لامارکی[۲۹] است (و البته اکثر پیاده‌سازی‌های الگوریتم‌های ممتیک به تکامل لامارکی – تکامل‌ای که در آن ژنوتایپ نیز تغییر می‌کند– شبیه است). با این‌که از دید نویسنده، این نوع برداشت –یعنی ترکیب جستجویی محلی در بین عملیات ژنتیکی– از ایده‌ی مم و الگوریتم ممتیک محدودکننده است اما همین نوع استفاده نیز به نتایج قابل توجه‌ای دست

[26] Meme
[27] Genotype
[28] Baldwinian Evolution
[29] Lamarckian Evolution

یافته است و هم‌چنان به عنوان یکی از راه‌های پر فایده و آینده‌دار محاسبات تکاملی شناخته می‌شود. برای مشاهده‌ی نمونه‌هایی از پژوهش‌های انجام شده درباره‌ی الگوریتم‌های ممتیک به [Moscato92]، [Radcliffe94]، [Merz99]، [Merz00]، [Federici03]، [Buriol04]، [Krasnogor04]، [Ong04] و [Zou04] مراجعه کنید. برای آشنایی بیش‌تر با الگوریتم‌های ممتیک و پژوهش‌های انجام شده می‌توانید به [Moscato03] نگاه کنید.

در پژوهش حاضر، دو تعبیر مختلف از الگوریتم ممتیک شده است. تعبیر اول همان استفاده‌ی توام تکامل و یادگیری برای حل مساله‌ی طراحی‌ی رفتار و ساختار است که جزییات آن تاکنون معرفی شده. تفاوت اصلی‌ی این تعبیر با آن‌چه در بدنه‌ی اصلی‌ی الگوریتم‌های ممتیک استفاده می‌شود در اختلاف سوژه‌ی بهینه‌سازی در دو مرحله‌ی یادگیری و تکامل است. در اکثر الگوریتم‌های ممتیکی، بهینه‌سازی مرحله‌ی تکامل و مرحله‌ی جستجوی محلی (یادگیری) بر روی یک فضا صورت می‌گیرد در حالی که در این پژوهش جستجوی تکامل بر فضای رفتارها است و جستجوی یادگیری بر فضای ساختارها. این نوع به کارگیری جدا از تفاوت یاد شده مشابه با روش تکاملی‌ی بالدوینی است.

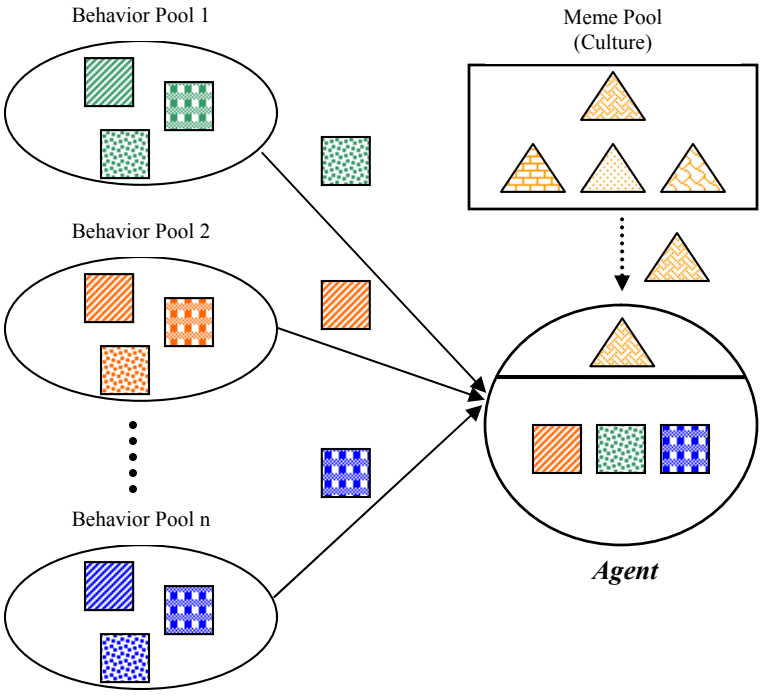
اما تعبیر استعاره‌ی مم به صورت‌های دیگری نیز ممکن است. به عنوان مثال مم می‌تواند به صورت سنت فرهنگی برای حل یک مساله‌ی مشترک برای اعضای جامعه نیز تعبیر شود. این سنت می‌تواند در شکل‌دهی‌ی باور افرادی که تازه به آن فرهنگ پا گذاشته‌اند —مثلا تازه به دنیا آمده‌اند– نقش داشته و آن‌ها را به سمت آن‌چه پیش از آن تجربه شده و آزمون پس داده ره‌نما باشد. در این تعبیر مم چیزی جدا از ژنِ موجودات است و در سطح‌ای دیگر عمل می‌کند. مم به صورت جهت‌ده[۳۰] اولیه و به عنوان نقطه‌ی آغازین‌ای برای جستجوی پاسخ مناسبِ مساله عمل می‌کند. برای روشن شدن قضیه این دو ایده‌ی به کارگیری مم را در قالب مساله‌ی اصلی‌ی این پایان‌نامه مطرح می‌کنم.

مکانیزم کلی‌ی طراحی‌ی سیستم رفتارگرای‌مان بدین‌گونه خواهد بود: چندین نوع رفتار مختلف هر کدام در مخزن ژنتیکی‌ی خویش به طور جداگانه تکامل می‌یابند. در این سیستم تکامل هم‌کارانه، هر مخزن ژنتیکی دارای ویژگی‌ها و ژنوتایپ خاص خود است. طبیعت به هنگام ایجاد موجودی (عامل) جدید یکی از اعضای هر کدام از این مخزن‌ها را به طور اتفاقی انتخاب می‌کند تا در نهایت مجموعه‌ای

[30] Bias

از رفتارهای $\{B_i\}$ ایجاد شود. عامل تازه به دنیا آمده با استفاده از این رفتارها (اطلاعات ژنتیکی) با کنش با محیط اطراف‌اش و دریافت سیگنال تقویت سعی در بهینه‌سازی‌ی معیار عمل‌کرد خویش –و در نتیجه یادگیری به‌ترین ساختار قابل ساخت با آن مجموعه رفتارها– می‌کند. پس از آن میزان سازگاری‌ی عامل به طبیعت برگردانده می‌شود و طبیعت با توجه به روش اشتراک سازگاری، سازگاری‌ی هر کدام از رفتارها را مشخص می‌کند. در این روند، ترکیب تکامل و یادگیری تا حدی مشابه با تفسیر غالب از الگوریتم‌های ممتیک است. اما برخلاف اکثر روش‌های الگوریتم ممتیک‌ای، یادگیری (و بهینه‌سازی) به طور مستقیم بر اطلاعات ژنتیکی (یعنی $\{B_i\}$) اعمال نمی‌شود بلکه بر روی ساختاری انجام می‌شود که اطلاعات ژنتیکی به عنوان اجزای آن (رفتارها) به کار می‌رود. به بیان‌ای دیگر فضای مساله‌ی یادگیری و تکامل –با وجود ارتباط تنگاتنگ با هم– یک‌سان نیستند. نحوه‌ی پیاده‌سازی‌ی برداشت دوم از مم – یعنی برداشت مم به عنوان سنت و باور اولیه– بدین‌صورت است: عامل در انتهای زندگی‌اش به‌ترین ساختار یافته‌شده را به عنوان مم به فرهنگ‌اش[۳۱] (مخزن مم[۳۲]) باز می‌گرداند. با توجه به میزان سازگاری‌ی عامل، شانس باقی ماندن و غلبه پیدا کردن مم در آن فرهنگ کم یا زیاد می‌شود. ساختارهایی که باعث ایجاد عامل‌های موفق شده‌اند، مم‌های غالب پیدا می‌کنند و ساختارهای غیرموثر و ناموفق قادر به ایجاد سنتِ باقی نخواهند شد. به هنگام تولید موجودی جدید، مخزن مم به عنوان باور اولیه‌ای (دانش اولیه‌ای) درباره‌ی ساختار مطلوب عمل می‌کند و عامل به جای شروع از ساختاری تصادفی، از آن ساختار از–پیش جواب‌داده استفاده می‌کند. اگر ساختار مناسب موجود فعلی دقیقا همان‌ای باشد که موجودات پیشین به آن رسیده بودند (مثلا به دلیل شباهت مجموعه رفتارهای در اختیارشان)، آن‌گاه عامل از همان اولین قدم‌های زندگی‌اش (اپیزودهای آغازین) درست و صحیح عمل می‌کند. البته از آن‌جا که رفتارها در طول نسل‌های متوالی تغییر می‌کنند، فرهنگ نسل‌های پیش ممکن است به‌ترین انتخاب برای نسل فعلی نباشد. متناسب با میزان تغییر اعضای پیشین جامعه نسبت به اعضای فعلی، مم‌های فرهنگ یا کاملا مناسب یا نسبتا مناسب خواهند بود. این نکته وقتی بیش‌تر آشکار می‌شود که به

[31] Culture
[32] Meme Pool



شکل ۱–۵. ساخت عامل جدید با استفاده از اطلاعات ژنتیکی و اطلاعات فرهنگی

خاطر بیاوریم تغییر رفتارها (تغییرات ژنتیکی) معمولا آهسته و پیوسته است. نمایش‌ای از این فرآیند در شکل ۱–۵ نشان داده شده است.

تاکید دوباره بر این شکل که در فرآیند مطرح شده از دو تفسیر مختلف الگوریتم ممتیکی استفاده شده است مفید خواهد بود: تفسیری که الگوریتم ممتیکْ معادل با جستجویی پس از عملیات تکامل است (استفاده از یادگیری) و تفسیری که در آن از مم‌ها به عنوان دانش اولیه‌ی عامل‌های جدید استفاده شده است. نحوه‌ی پیاده‌سازی‌ی برداشت اول در بخش پیشین این فصل توضیح داده شده است. هدفِ اصلی‌ی این بخش معرفی‌ی مکانیزم‌ای برای تفسیر دوم است. شبه‌کدی از این فرآیند را در جدول ۱–۵ می‌بینید.

بیان دقیق‌تر ایده و چگونگی‌ی پیاده‌سازی‌ی الگوریتم ممتیک در مساله‌ی خودکارسازی‌ی طراحی‌ی عامل رفتارگرا بدین‌گونه است: مجموعه‌ی عامل‌های $\{agent_i\}$ را در نظر بگیرید. هر عامل در این پژوهش شامل دو بخش‌اند: رفتارها و ساختار آن؛ $agent_i : (\{B_i\}, T_i)$. هر عامل در دوره‌ی زندگی‌اش تلاش می‌کند با تغییر ساختار، حداکثر پاداش ممکن را دریافت کند (به یاد داشته باشید که در

این بخش تنها ساختار یاد گرفته می‌شود و رفتارها ذخیره‌ی ژنتیکی‌ی به ارث رسیده هستند. گرچه اعمال ایده‌های این بخش به حالت‌ای که رفتار نیز یاد گرفته شود چندان متفاوت نیست). در انتهای زندگی‌ی عامل، نتیجه‌ی یادگیری $\left(\{B_i\},T_i^*\right)$ خواهد بود که در آن T_i^* ساختار بهینه شده است. این ساختار بهینه اینک به صورت مم‌ای وارد فرهنگ (یا همان مخزن مم) $\mathbf{M}$ می‌شود. در این مخزن، مم‌های عامل‌های سازگارتر شانس غلبه‌گی‌ی بیش‌تری دارند. این مم‌ها پایدارترند و شانس بیش‌تری برای تاثیرگذاری بر عامل‌های تازه به دنیا آمده دارند. به عبارت دیگر، فرهنگ $\mathbf{M}$ به صورت مجموعه‌ای از ترکیب‌های زیر است:

$$\mathbf{M}:\left\{\left(T_i^*,f_{T_i}\right)\right\} \qquad (۵-۱۰)$$

که در آن f_{T_i} سازگاری‌ی عامل‌ای با ساختار (مم) T_i است. بدیهی است که ساختار T_i می‌تواند نتیجه‌ی نهایی‌ی یادگیری عامل‌هایی با مجموعه رفتارهای متفاوت باشد – یعنی ما به ازای هر T_i چندین مجموعه $\{B_i\}$ داریم. مشخص است که به ازای رفتارهای مختلف، سازگاری‌ی عامل نیز تغییر می‌کند. در نتیجه برای این‌که معیاری حدودی از میزان ارزش هر مم داشته باشیم می‌بایست متوسط ارزش آن را به ازای همه‌ی عامل‌هایی که ساختارشان در نهایت به آن مم هم‌گرا خواهد شد محاسبه کنیم، یعنی:

$$f_{T_i}=\frac{1}{\mathrm{N}}\sum_{\mathrm{A_i}:(\{B_i\},T_i)}f(A_i) \qquad (۵-۱۱)$$

که در آن N تعداد عامل‌هایی است که در نهایت و پس از یادگیری به معماری T_i رسیده‌اند. اضافه کردن فاکتور فراموشی[۳۳] اجازه‌ی تحمل ناایستایی[۳۴] فرآیند تکاملی را می‌دهد:

[33] Forgetting Factor
[34] Non-Stationary

$$f_{T_{in+1}} = \left(1-\alpha_{T_i}\right) f_{T_{in}} + \alpha_{T_i} f\left(A\right) \qquad A:\left(\{B_i\},T_i\right) \tag{۵-۱۲}$$

اندازه‌ی مخزن مم محدود است: تنوع فرهنگی[۳۵] یک اجتماع بی‌نهایت نیست. در نتیجه می‌بایست راه‌کار مُعین‌ای برای حفظ و نگه‌داری‌ی مم‌ها داشته باشیم. فرض کنید مم‌ای جدید که پیش از آن عضوی از مخزن نبوده است پس از دوره‌ی زندگی‌ی موجودی ایجاد شود. اگر مخزن مم‌ها گنجایش داشته باشد، اضافه کردن مم تازه به آن مشکل‌ای ایجاد نمی‌کند. اگر مم‌ای ایجاد شده قبلا هم در مخزن وجود داشته بود، آن‌گاه به کمک معادله‌ی (۱۲-۵) ارزش آن مم در مخزن مم‌ها به روز می‌شود. اما اگر مم در مخزن وجود نداشت و از طرف دیگر مخزن نیز پر شده بود (ظرفیت مخزن را طراح از پیش مشخص می‌کند)، ممِ جدید جای‌گزین بدترین مم —مم با پایین‌ترین ارزش– می‌شود. چنین فرآیندی در جوامع بشری نیز رخ می‌دهد: الگوهای رفتاری جدید به تدریج جای‌گزین سنت‌های قدیمی و بی‌فایده می‌شوند. هم‌چنین در طول زمان ارزش یک سنت با توجه به سودی که می‌رساند کم یا زیاد می‌شود.

پس از تولد هر موجود، مم‌ای از مخزن مم‌ها به عنوان دانش اولیه به او منتقل می‌شود. این‌که کدام مم انتخاب شود بستگی به ارزش هر کدام از مم‌ها دارد: مم‌های با ارزش‌تر شانس بیش‌تری برای باور آغازین موجودات شدن دارند.

در این پژوهش از مکانیزم رقابتی‌ای استفاده کرده‌ام که شانس انتخاب هر مم متناسب با ارزش مقیاس‌شده‌ی[۳۶] هر کدام از مم‌ها است. چنین فرآیندی در روش‌های تکاملی‌ی متداول نیز به کار می‌رود. هم‌چنین گاهی مم‌ای به طور اتفاقی —بدون توجه به ارزش‌اش– انتخاب می‌شود. این‌کار تا حدی مشابه با اپراتور جهش در روش‌های تکاملی، باعث حفظ تنوع فرهنگی جامعه می‌شود. پس از انتخاب مم با هر روش ممکن، عامل از ساختار مشخص شده توسط مم با محیط کنش می‌کند و با استفاده از روش یادگیری ساختاری که در اختیار دارد سعی در به‌بود ساختار خویش و یافتن ساختار بهینه (البته مقید به

[35] Cultural Diversity
[36] Scaled

رفتارهای در اختیارش) می‌کند. اگر رفتارهای عامل تا حدی قابل قبولی شبیه به رفتارهایی باشند که عامل تولید کننده‌ی ممِ اولیه‌ی عامل اخیر از آن استفاده می‌کرده است، آن‌گاه عامل از همان ابتدا با ساختاری تقریبا درست و صحیح شروع به تصمیم‌گیری می‌کند. در این صورت دوره‌ی لازم برای یادگیری —و رسیدن به حد قابل قبول‌ای از بازده– بسیار کم خواهد شد.

البته اضافه کردن جهت‌دهی‌ی اولیه ممکن است باعث اضافه شدن احتمال گیر افتادن عامل در بیشینه‌گی‌ی محلی‌ای شود: تجربیات "خوب" —ولی نه "خیلی خوب"– موجودات قبلی باعث جلوگیری از کاوش و تجربه‌ی حالت‌های "خیلی خوب" می‌شود. البته کاوش‌گری در انتخاب ساختارها —و مثلا تجربه‌ی گاه به گاه ساختارهای تصادفی– و هم‌چنین ایجاد گه‌گاه مم‌های اتفاقی باعث کم شدن احتمال این اتفاق می‌شود.

با توجه به آن‌چه تاکنون در این فصل گفته شد، در چارچوب مطرح شده سه روش مختلف برای تطبیق عامل‌ها در نظر گرفته شده است: تکامل ژنتیکی که در آن مجموعه‌ای از رفتارها در مخزن‌های ژنتیکی‌ی جداگانه‌ای تکامل می‌یابند، یادگیری فردی‌ی هر عامل برای یافتن به‌ترین ساختار ممکن و جهت‌دهی‌ی فرهنگی‌ای که از تجربیات عامل‌های پیشین برای راه‌نمایی یادگیری عامل‌های جدید استفاده می‌شود.

- Initialize n different behavior pools $\{\mathbf{B}_i\}$ for each different behavior type B_i
- Initialize an empty culture (meme pool) $\mathbf{M}$
- While stopping condition are not met
    - Selects n different behavior B_i from each behavior pool to make a set of randomly chosen behaviors $\{B_i\}$
    - If there is any meme in the meme pool $\mathbf{M}$, select a meme $T^0 \in \mathbf{M}$ according to the merit of each meme
    - Pass $\{B_i\}$ and T^0 to the agent
        - Initialize Learning parameters
        - For a lifetime do
            - Update learning parameters to ensure convergence
            - Select an architecture T^* that maximizes (3-6) (Zero Order representation) using an optimization method (If the architecture is fixed, skip this step).
            - Using the provided architecture T^*, let the agent interact with the environment for a while
            - Receive reinforcement signal from the critic (that can be external or internal)
            - Update the estimation of necessary values ($\{\widetilde{V}_{ij}\}$ for Zero Order structure learning representation (3-16))
        - Return Fitness and the final structure T^* to the evolutionary process
    - Share fitness to behaviors according to sharing policy ((5-5) or (5-7))
    - Update meme pool using T^* and its fitness (5-12)
    - For each behavior pool
        - Apply conventional genetic operators to behaviors in order to generate a new population, i.e. Selection, Crossover, and Mutation.

جدول ۱-۵. الگوریتم ممنیک. رفتارها تکامل می‌یابند، ساختار یاد گرفته می‌شود و مم‌ها انتقال دانش ساختار مناسب را بر عهده دارند.

۶) آزمایش‌ها

در این فصل به بررسی‌ی تجربی‌ی روش‌های پیش‌نهاد شده برای یادگیری و تکامل معماری‌ی رفتارگرای سلسله مراتبی می‌پردازم. برای این‌کار آن روش‌ها را در دو مساله‌ی مختلف آزمایش می‌کنم. یکی از مسایل، مساله‌ای مجرد است که به طور خاص برای تست روش‌های پیش‌نهاد شده طراحی شده است. طراحی‌ی چنین مساله‌ای ما را قادر می‌سازد تا در شرایطی کنترل‌شده به بررسی‌ی روش‌ها بپردازیم. مساله‌ی دیگر، مساله‌ای از دنیای واقعی است: بلندکردن یک جسم به وسیله‌ی چند روبات. پیش از پژوهش اخیر، این مساله‌ی دشوار به وسیله‌ی معماری‌ای رفتارگرا با سعی و خطای بسیار طراح حل شده بود [Nili01]. از آن‌جا که پروسه‌ی طراحی‌ی کنترلر مناسب برای این مساله ساده نبوده است، طراحی‌ی خودکار معماری‌ی رفتارگرایی که بتواند آن مساله را حل کند اثبات تجربی‌ی قابل قبولی برای موثر بودن روش‌های پیش‌نهادی است.

بررسی‌های انجام گرفته را به دو بخش روش‌های مبتنی بر یادگیری و روش‌های مبتنی بر ترکیب یادگیری/تکامل تقسیم می‌کنم. در بخش یادگیری به یادگیری ساختار با رفتار از پیش مشخص، یادگیری رفتار با ساختار معلوم، و یادگیری هم‌زمان ساختار و رفتار می‌پردازم. در بخش ترکیب یادگیری/تکامل به تکامل رفتار با ساختار از پیش مشخص و تکامل رفتار و یادگیری ساختار به طور هم‌زمان می‌پردازم. هم‌چنین ایده‌ی الگوریتم ممتیک معرفی شده در فصل تکامل رفتار (فصل ۵) نیز هم‌زمان با ایده‌های تکامل رفتار مورد مطالعه قرار می‌گیرد.

پیش از ادامه لازم است بگویم که هیچ تلاش منظم و دقیق‌ای برای بهینه‌سازی‌ی پارامترها یادگیری یا تکامل صورت نداده‌ام و در نتیجه ممکن است با انتخاب مجموعه پارامترها یا سیگنال تقویت دیگری –همان‌طور که در جای لازم بیش‌تر در موردش توضیح خواهم داد– به نتایج به‌تری رسید.

۶-۱) مساله مجرد

هدف مساله‌ی مجردی نشان دادن قابلیت عامل برای تطابق با محیطی ناشناخته و اتفاقی طراحی شده از طریق کنش با آن و دریافت سیگنال پاداش یا تنبیه است. این تطبیق می‌تواند به صورت یادگیری یا

تکامل باشد. اگر عامل بتواند سیاست مناسب‌ای برای کنش با محیط پیدا کند پاداش می‌گیرد و در غیر این صورت تنبیه می‌شود. تنبیه و یا پاداش عامل به تطبیق دقیق و مو به موی سیاست عامل و سیاست صحیح و دل‌بخواه $a^* = \Pi(s)$ بستگی دارد. در این مساله $s \in \{1,2,...,r\} \times \{1,2,...,s\}$ و $a^* \in \{1,2,...,q\}$ و $\Pi(\cdot)$ نیز در ماتریس‌ای است که نگاشت بین حالت فعلی و عملِ مطلوب ولی ناشناخته برای عامل را مشخص می‌کند. عامل می‌بایست از طریق کنش با محیط سیاست $a = \pi^{agent}(s)$ای[1] را بیابد که کم‌ترین فاصله با a^* را داشته باشد. فاصله در فضای عمل‌ها بدین‌گونه تعریف می‌شود:

$$\|\Pi(s) - \pi(s)\| = \begin{cases} 0 & \text{if } \Pi(s) = \pi(s) \\ 1 & \text{otherwise} \end{cases} \qquad (۶-۱)$$

و کلِ فاصله برای همه‌ی حالت‌ها نیز بدین‌گونه مشخص می‌شود:

$$\|\Pi - \pi\| = \sum_{s \in S} \|\Pi(s) - \pi(s)\| . \qquad (۶-۲)$$

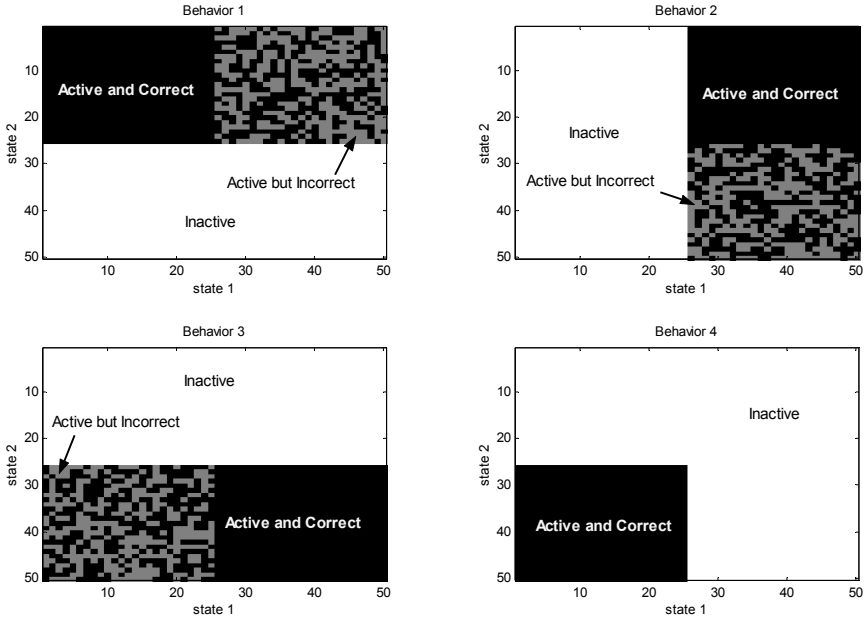
اندازه‌ی فاصله‌ی بین سیاست یادگرفته شده و سیاست بهینه (احتمال خطا) بدین‌گونه تعریف می‌شود:

$$P(error) = \frac{1}{|S|} \|\Pi - \pi\| \qquad (۶-۳)$$

که همان‌طور که می‌بینید اثر ابعاد فضای حالت در آن با نرمال‌کردن خنثی شده است. این اندازه، معیار خوبی برای بازده سیستم در حالت‌ای است که توزیع انتخاب حالت‌ها یک‌نواخت باشد.

سناریوی کنش عامل با محیط و یادگیری بدین‌گونه است: عامل با معماری‌ی تصادفی‌ی T – که در آن رفتارها از جعبه‌ابزاری از رفتارها انتخاب شده است– آغاز به کار می‌کند. عامل در حالت

[1] در این‌جا فرض کرده‌ام که عامل اعمال‌اش را حریصانه انتخاب می‌کند و π^{agent} نیز سیاست deterministic عامل است.



شکل ۱-۶. (مساله‌ی مجرد) چهار رفتار مختلف مساله‌ی مجرد. به مناطق مختلف تحریک و درستی یا نادرستی‌ی پاسخ در آن مناطق دقت کنید.

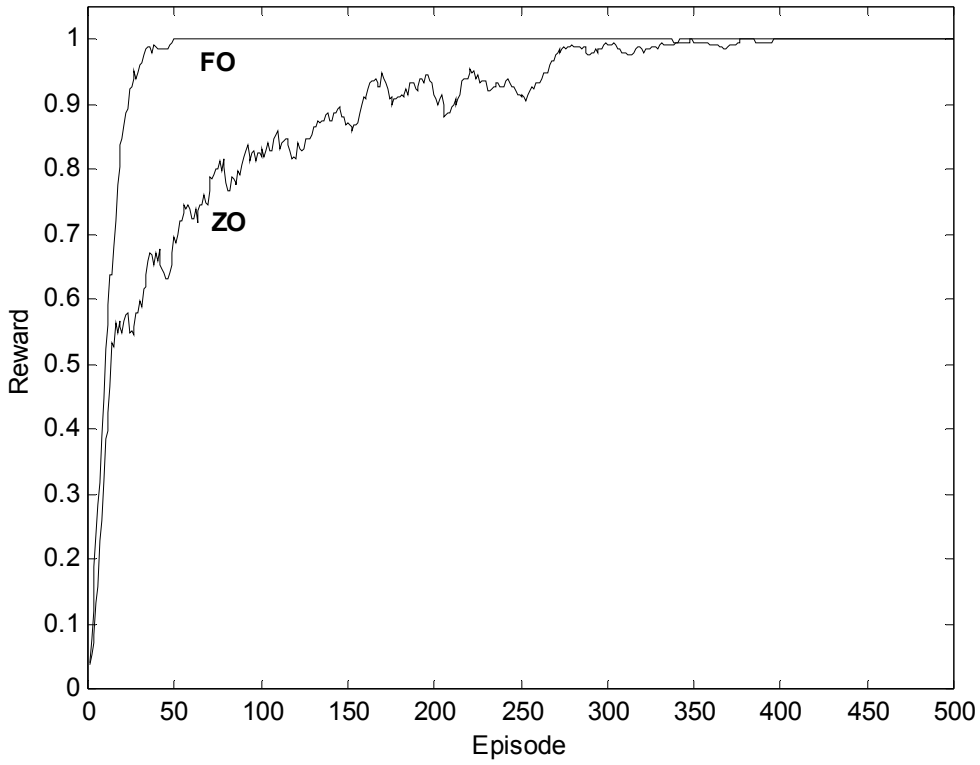
تصادفی‌ی s قرار می‌گیرد و سپس عمل a را با توجه به سلسله مراتب رفتارها و سیاست هر کدام از رفتارهای‌اش $B_i(s_i)$ انتخاب می‌کند. اگر عامل عمل صحیح –در مقایسه با پاسخ بهینه‌ی $-\Pi(s)$– را انتخاب کند، پاداش 1+ دریافت می‌کند و در غیر این‌صورت به اندازه‌ی 1- تنبیه می‌شود؛ یعنی:

$$r(s)=\begin{cases}+1 & \text{if } \Pi(\mathrm{s})=\pi^{agent}(\mathrm{s}) \\ -1 & \text{otherwise}\end{cases} \qquad (۶-۴)$$

که در آن $\pi^{agent}(s)$ عمل انتخاب شده‌ی عامل با توجه به سیاست هر رفتار و ساختار رفتارها در معماری است. این عمل با انتخاب s کاملا تصادفی‌ی جدیدی ادامه می‌یابد. در مساله‌ی مجرد هر کنش با محیط و دریافت سیگنال تقویت یک اپیزود را تشکیل می‌دهد.

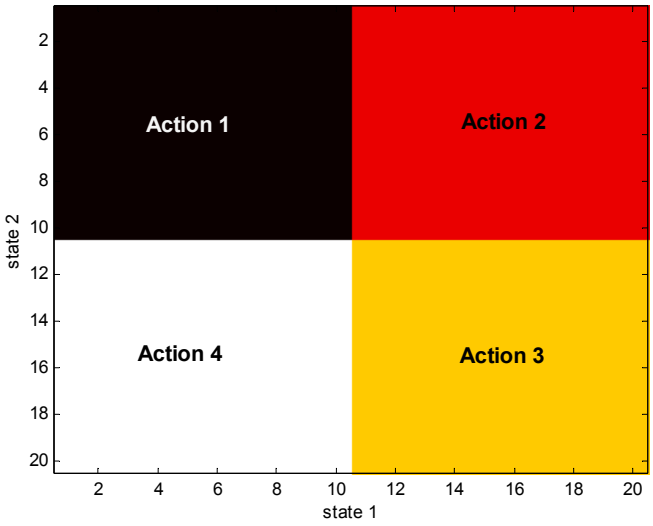
۶-۱-۱) یادگیری ساختار

اولین آزمون روش‌های یادگیری ساختار بر روی مساله‌ی مجرد انجام می‌شود. برای این‌کار مساله‌ای مجرد با ابعاد فضای حالت 50x50 و دارای دو عمل تعریف شده است (برای مشاهده‌ی جزییات آزمون



شکل ۲-۶. (مساله‌ی مجرد) مقایسه بین روش‌های یادگیری ساختار (مبتنی بر بازنمایی مرتبه صفر و بازنمایی مرتبه یک)

به جدول ۱-۶ مراجعه کنید). چهار رفتار مختلف (B_1، B_2، B_3 و B_4) تعریف شده‌اند که هر کدام در بخش‌ای از فضای حالت تحریک می‌شوند (شکل ۱-۶). در شکل، ناحیه‌ی تیره بیان‌گر ناحیه‌ای از فضای حالت است که رفتار تحریک شده و عمل‌ای را -درست یا نادرست- پیش‌نهاد می‌کند. نواحی‌ای که عمل انتخاب شده درست است با رنگ سیاه و نواحی‌ای که عمل انتخاب شده نادرست است با رنگ خاکستری مشخص شده است. رفتارها به گونه‌ای طراحی شده‌اند که مساله‌ی چینش رفتارها در ساختار، دارای یک پاسخ بهینه‌ی عمومی‌ی یکتا باشد ($T = [B_1 \quad B_2 \quad B_3 \quad B_4]$ در این مساله). در چنان حالت‌ای، معماری به ازای همه‌ی حالت‌ها به درستی پاسخ می‌دهد. نتیجه‌ی به کارگیری‌ی یادگیری‌ی ساختار مبتنی بر بازنمایی مرتبه صفر و مرتبه یک در شکل ۲-۶ نشان داده شده است. دیده می‌شود که هر دو روش به خوبی عمل کرده و عامل توانسته ساختار مناسب را به سرعت یاد بگیرد. با این‌وجود دیده می‌شود که روش یادگیری ساختار مبتنی بر بازنمایی مرتبه یک سریع‌تر هم‌گرا شده است. این پدیده به آن دلیل است که در روش مبتنی بر بازنمایی مرتبه یک، در هر کنش عامل با محیط، اطلاعات بیش‌تری

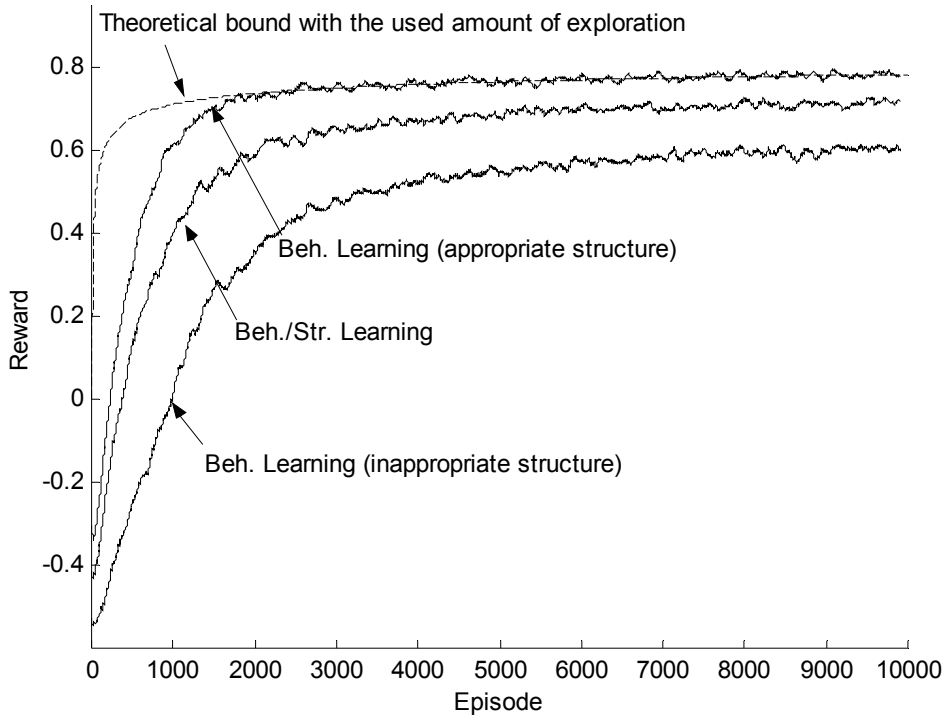


شکل ۳-۶. (مساله‌ی مجرد)فضای مساله برای آزمون‌های یادگیری رفتار/ساختار

درباره‌ی ساختار صحیح به شکل تقویت ترتیب رفتارها کسب می‌شود. هم‌چنین از شکل مشخص است که روش بازنمایی مرتبه یک دارای نویز کم‌تری در میزان سیگنال تقویت دریافتی نسبت به بازنمایی مرتبه صفر است. این نویز و تاخیر هم‌چنین به این خاطر است که در بعضی از تجربه‌های یادگیری با استفاده از بازنمایی مرتبه صفر، یادگیری دیرتر از معمول صورت می‌گیرد و آن تجربه‌های تاخیردار باعث کند شدن متوسط کلی‌ی بازنمایی مرتبه صفر می‌شود وگرنه در اکثر موارد یادگیری مرتبه صفر نیز به سرعت (سریع‌تر از آن‌چه در شکل دیده می‌شود) به پاسخ مناسب هم‌گرا می‌شود.

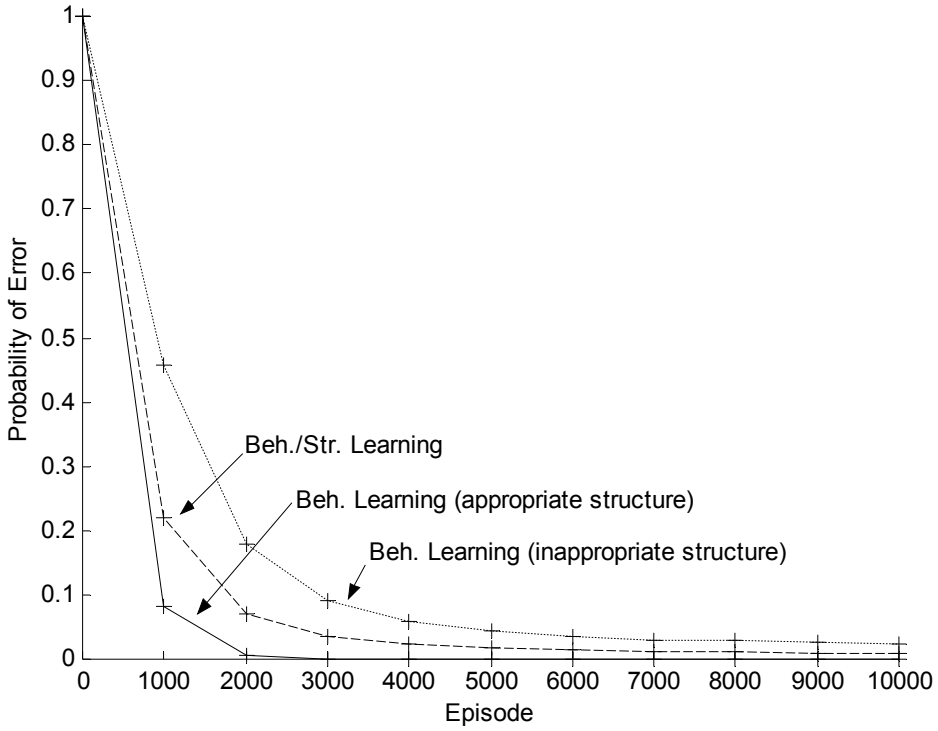
۶-۱-۲) یادگیری رفتار و یادگیری هم‌زمان رفتار/ساختار

در این زیربخش روش پیش‌نهادی برای یادگیری رفتار و هم‌چنین یادگیری ساختار و یادگیری رفتار به طور هم‌زمان برای مساله‌ی مجرد مورد آزمون واقع می‌شود. فضای مساله $\Pi(\cdot)$ از چهار ناحیه تشکیل شده است که در هر کدام از آن نواحی فقط یکی از چهار عمل ممکن قرار گرفته است (شکل ۳-۶). مطابق قبل چهار رفتار داریم که نواحی‌ی تحریک‌شان به مانند آزمایش یادگیری ساختار تعریف شده است ولی برخلاف آن، رفتارها از پیش دارای عمل‌ای مشخص نیستند و عملِ منسوب به هر حالتِ هر رفتار می‌بایست یاد گرفته شود. برای رفتار B_i فضای عمل توسعه یافته شامل عمل a_i و NA است. به دلیل ساختار ویژه‌ی مساله و نواحی‌ی تحریک رفتارها، همیشه نگاشت‌ای مناسب از حالت-عمل رفتارها



شکل ۴-۶. (مساله‌ی مجرد) مقایسه پاداش دریافتی برای یادگیری رفتار (ساختار ثابت) و یادگیری هم‌زمان ساختار و رفتار.

وجود دارد که منجر به تولید پاسخ صحیح معماری شود. با این وجود، با توجه به ساختار معماری، نگاشت‌های متفاوتی می‌بایست یاد گرفته شوند. اگر ساختار *مناسب* باشد (به این معنا که هیچ رفتاری برای پاسخ صحیح معماری لازم به یادگیری NA نباشد) آن‌گاه یادگیری‌ی تنها یک نگاشت از یک رفتار برای پاسخ مناسب دادن در هر حالت کافی است. اگر ساختار *نامناسب* باشد نه تنها نگاشت رفتار پیش‌نهاد دهنده‌ی عملِ صحیح می‌بایست به درستی یاد گرفته شود بلکه رفتارهای بالایی آن رفتار کنترل‌کننده نیز باید یاد بگیرند تا در آن حالت NA پیش‌نهاد کنند. برای این مساله، ساختار کاملا مناسب $T=\begin{bmatrix}B_1 & B_2 & B_3 & B_4\end{bmatrix}$ و ساختار کاملا نامناسب $T=\begin{bmatrix}B_4 & B_3 & B_2 & B_1\end{bmatrix}$ خواهد بود. انتظار دارم که ساختار مناسب با القای دانش اولیه‌ی صحیح، یادگیری را ساده‌تر و سریع‌تر و ساختار نامناسب آن را دشوارتر و کندتر کند.

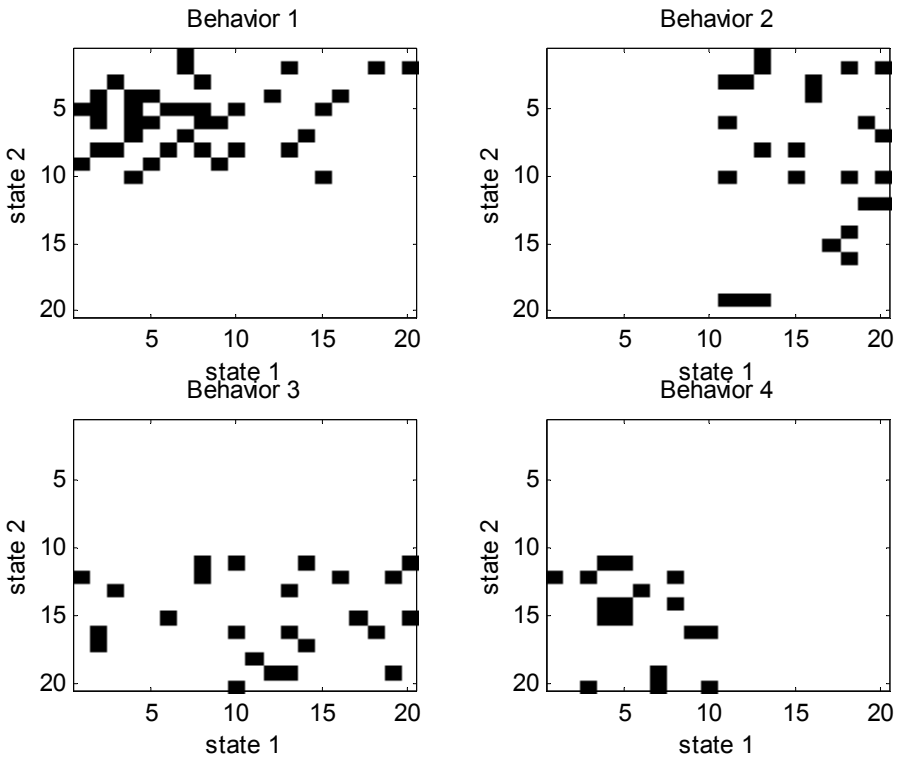


شکل ۵-۶. (مساله‌ی مجرد) مقایسه احتمال خطای یادگیری رفتار (ساختار ثابت) و یادگیری هم‌زمان ساختار و رفتار.

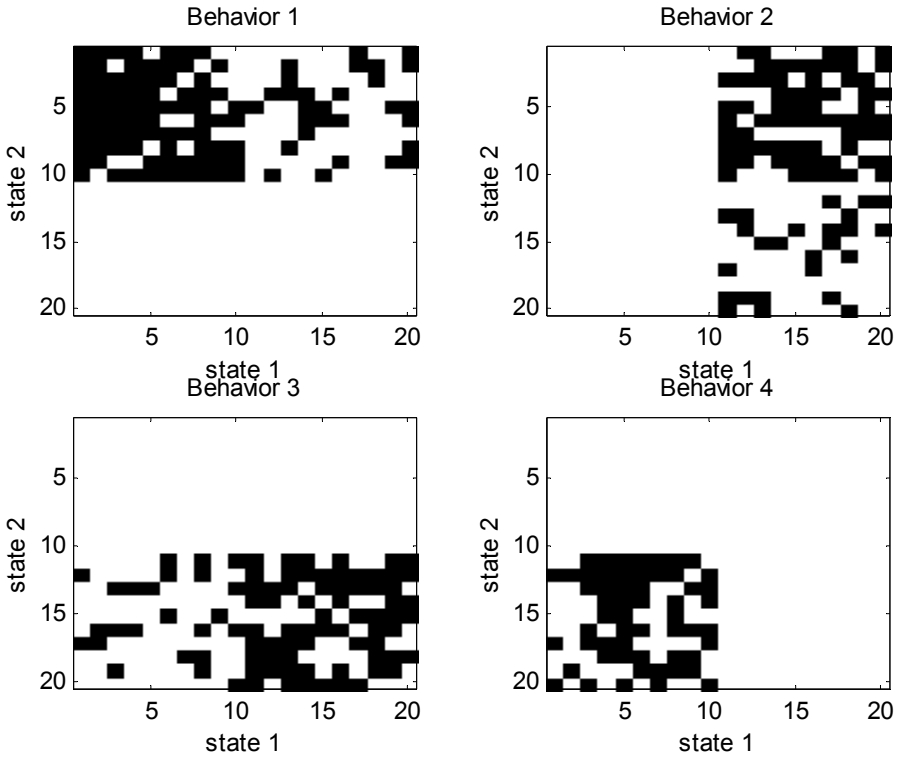
سه آزمون مختلف انجام می‌دهم: یادگیری رفتار با ساختار مناسب، یادگیری رفتار با ساختار نامناسب و یادگیری هم‌زمان ساختار و رفتار[۲]. برای بررسی‌ی تاثیر یادگیری در عمل‌کرد عامل دو معیار مختلف استفاده می‌شود. اولی میزان پاداش (و البته تنبیه) دریافتی‌ی عامل است که عمل‌کرد رفتاری عامل در طول یادگیری را نشان می‌دهد و دیگری احتمال خطا (۳-۶) است. معیار دوم نشان دهنده‌ی میزان دانش کسب شده‌ی عامل بدون توجه به استراتژی‌ی کاوشی‌ی به کار رفته است. پاداش کسب شده در شکل ۴-۶ و احتمال خطا در شکل ۵-۶ نمایش داده شده است. مطابق انتظار، عامل‌ای که از دانش اولیه‌ی ساختار مناسب برخوردار بوده است سریع‌تر از عامل با دانش اولیه‌ی نامناسب عمل‌کرد صحیح را یاد گرفته است. دلیل این موضوع همان‌طور که توضیح داده شد نیاز به یادگیری میزان کم‌تری از نگاشت‌های حالت-عمل است. به عبارت دیگر تنها کافی است که نگاشت حالت-عملِ عامل‌ای که می‌تواند عملِ مناسب را پیش‌نهاد کند یاد گرفته شود و دیگر نیازی به یادگیری NA در رفتارهای بالایی

[2] برای بخش یادگیری ساختار از روش یادگیری ساختار با بازنمایی مرتبه صفر استفاده شده است.

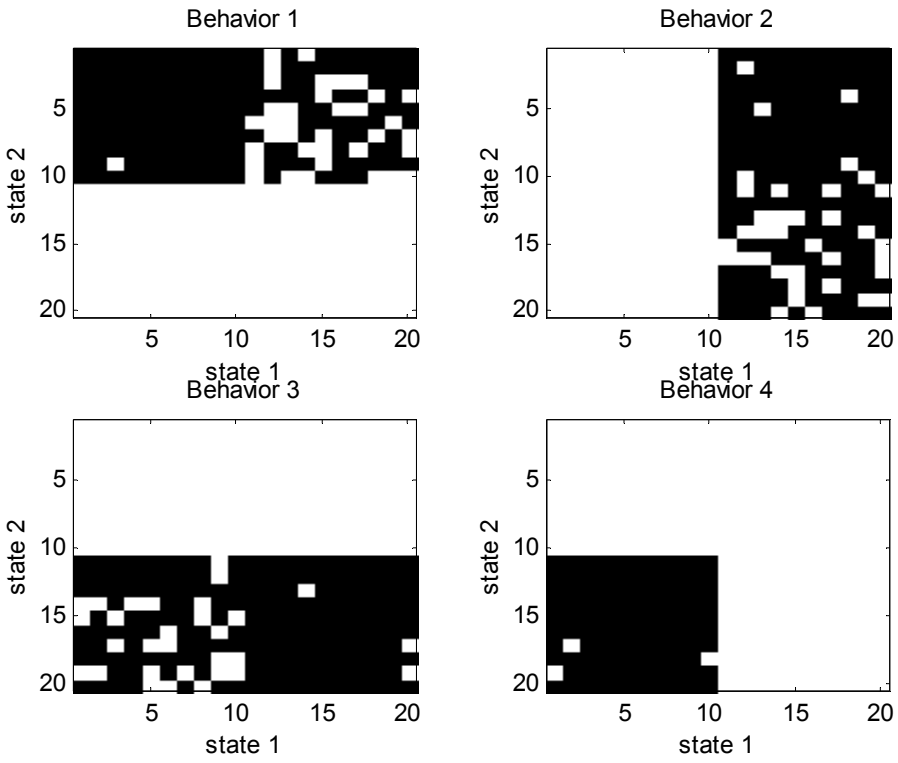
تحریک شده —در صورت وجود– نیست. عمل‌کرد یادگیری هم‌زمان ساختار و رفتار به دلیل آن‌که از اطلاعات اولیه‌ی کم‌تری نسبت به حالت یادگیری رفتار با ساختار مناسب استفاده می‌کند اندکی پایین‌تر است اما از حالت یادگیری رفتار با ساختار نامناسب به‌تر است. در این حالت یادگیری ساختار به کمک یادگیری رفتار آمده است و عامل به سمت ساختارهایی حرکت کرده که بیش‌تر پاداش‌آور بوده‌اند. کاوش موجود باعث پایین آمدن کیفیت عمل‌کرد سیستم شده است. میزان حداکثر پاداش قابل دریافت با آن سطح کاوش در شکل ۴–۶ نمایش داده شده است. دیده می‌شود که نتایج روش‌های پیش‌نهادی، به طور مجانبی به آن حد نزدیک می‌شوند. هم‌چنین، بررسی‌ی احتمال خطا (۳–۶) مشخص می‌کند که در انتخاب اپیزود ۱۰۰۰۰ام (آخرین اپیزود)، احتمال خطا برای حالت‌های یادگیری رفتار (ساختار نادرست)، یادگیری هم‌زمان ساختار و رفتار و یادگیری رفتار (ساختار درست) به ترتیب برابر با 0.024، 0.009 و 0 است. نتایج حاصل بیان‌گر قابلیت روش‌های یادگیری پیش‌نهادی هستند. برای درک بیش‌تر چگونگی‌ی روند یادگیری، نواحی‌ی یادگرفته شده را در طول فرآیند یادگیری مشخص کرده‌ام (شکل‌های ۶–۶ تا ۹–۶). این نتایج حاصل یادگیری هم‌زمان رفتار و ساختار است. مطابق انتظار نواحی‌ی یادگرفته شده به تدریج بزرگ می‌شوند. جزییات این آزمایش در جدول ۱–۶ آمده است.



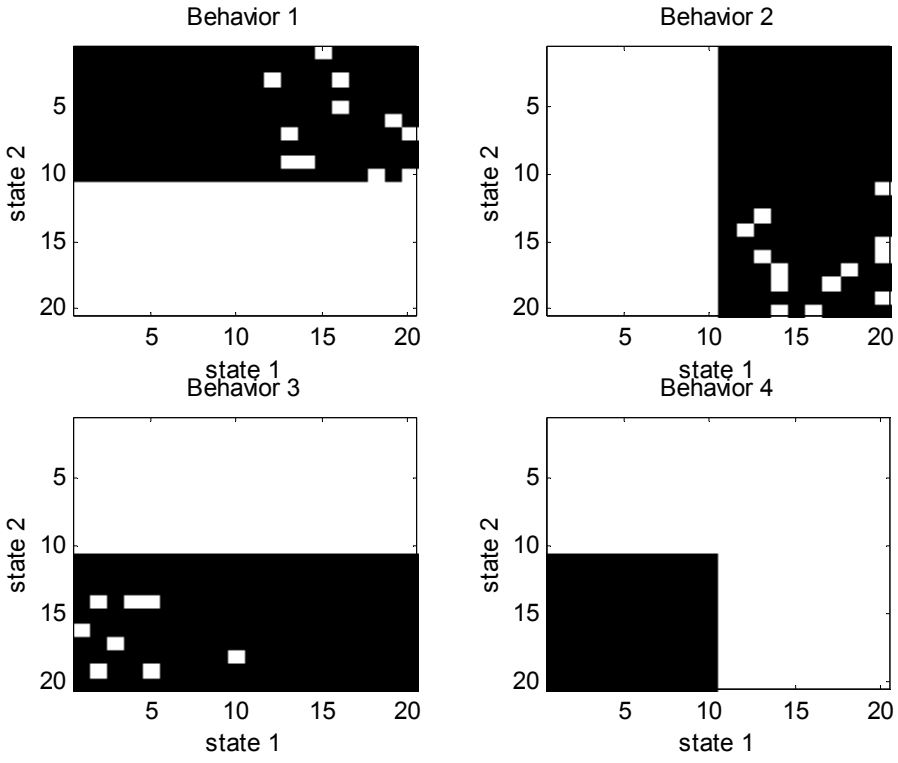
شکل ۶–۶. (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۱۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.



شکل ۷–۶. (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۵۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.



شکل ۸-۶. (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۱۵۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.



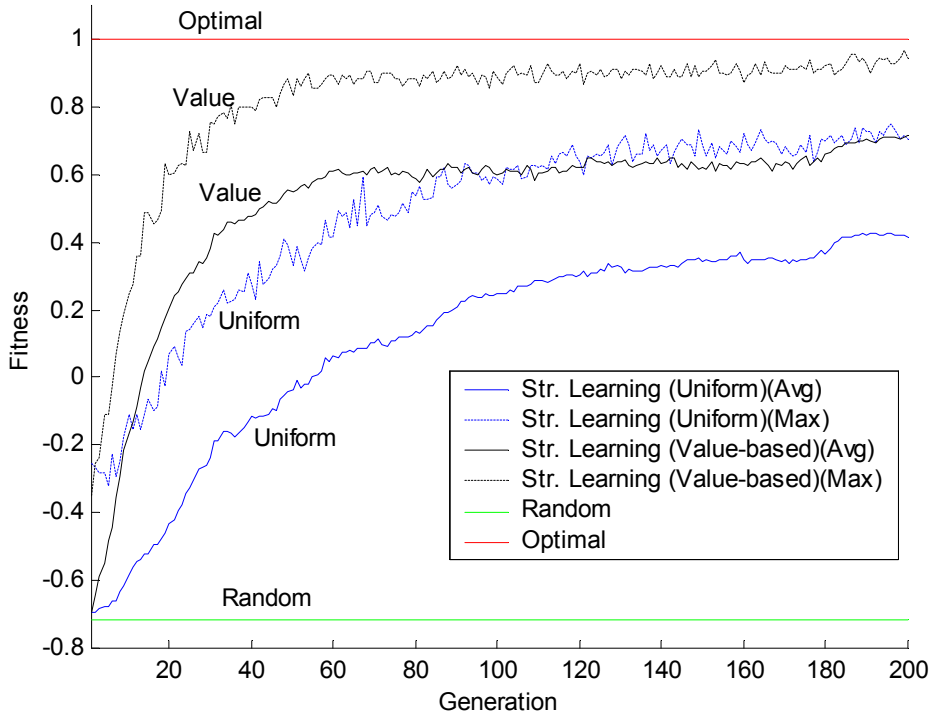
شکل ۹-۶. (مساله مجرد) نواحی‌ی یادگرفته شده (اپیزود ۳۰۰۰). نواحی‌ی تیره به درستی یاد گرفته شده‌اند.

۱۰۰

۶-۱-۳) تکامل رفتار و یادگیری ساختار

در این بخش، ایده‌های مطرح شده در فصل ۵ را بر روی مساله‌ی مجرد آزمایش می‌کنم. بررسی خواهم کرد که آیا تکامل رفتارها می‌تواند جای‌گزین مناسب‌ای برای یادگیری باشد یا خیر و هم‌چنین تاثیر ترکیب هم‌زمان یادگیری و تکامل را نیز مطالعه می‌کنم. تکامل رفتار را در دو حالت مختلف اشتراک سازگاری‌ی یک‌نواخت و اشتراک سازگاری‌ی ارزش‌محور بررسی می‌کنم. هم‌چنین، تاثیر استفاده از ایده‌های الگوریتم ممتیک –که در آن مجموعه‌ای از الگوهای فرهنگی شکل گرفته و اعضای جدید جامعه از ابتدای کنش‌شان با محیط به آن‌ها مجهز هستند– نیز مطالعه می‌شود. در این مساله، سازگاری‌ی عامل معادل با متوسط پاداش و تنبیه‌ای است که در طول دوره‌ی زندگی‌اش دریافت می‌کند. از آن‌جا که در مساله‌ی مجرد، پاسخ صحیح پاداش ۱ و پاسخ نادرستْ تنبیه ۱– دارد، عدد سازگاری بین ۱ و ۱– خواهد بود.

در این بخش، آزمایش‌های انجام شده بر روی مساله‌ای با ابعاد 5x5 و دارای هفت عمل صورت می‌گیرد. در این مساله هفت رفتار $B_i\ (i=1,...,7)$ داریم که هر کدام از آن‌ها می‌توانند در کلِ ۲۵ حالت مساله تحریک شوند. خروجی‌ی هر رفتار، متناسب با شماره‌ی آن رفتار است: یعنی رفتار B_i می‌تواند یا عمل a_i را انتخاب کند و یا NA را. واضح است که برای حلِ مساله‌ی $\Pi(\cdot)$ رفتارها می‌بایست یاد بگیرند که در چه شرایط لازم است فعال شوند و در چه شرایطی سکوت اختیار کنند. از آن‌جا که بازنمایی همه‌ی رفتارها یک‌سان است و فضای تحریک آن‌ها نیز تفاوت‌ای با هم ندارد، پس به ازای هر ترتیب رفتاری، نگاشت $B_i(s'):S_i'\rightarrow A_i'$ وجود دارد که به پاسخ مناسب می‌انجامد. البته توجه کنید که به ازای ساختارهای مختلف، این نگاشت یک‌سان نیست. توضیحات بیش‌تر در مورد اندازه‌ی جمعیت تکامل‌یابنده، پارامترهای یادگیری و پارامترهای الگوریتم ممتیک در جدول ۲-۶ آمده است. اینک نتایج آزمایش را توضیح می‌دهم.



شکل ۱۰-۶. (مساله مجرد) مقایسه‌ی بین متوسط و حداکثر سازگاری جمعیت به ازای روش‌های مختلف اشتراک سازگاری هم‌راه با یادگیری ساختار. اشتراک سازگاری یک‌نواخت (آبی)، اشتراک سازگاری ارزش‌محور (سیاه). خط سبز متوسط سازگاری معماری‌ای با ساختار و رفتار کاملا تصادفی است (با احتمال انتخاب عمل و NA برابر) و خط قرمز حداکثر سازگاری دست‌یافتنی است. خطوط پررنگ متوسط سازگاری جمعیت را نشان می‌دهند و خطوط نقطه‌چین، حداکثر سازگاری را.

در شکل ۱۰-۶، مقایسه‌ای بین سازگاری روش اشتراک سازگاری‌ی یک‌نواخت و ارزش‌محور با پاسخ بهینه و پاسخ کاملا تصادفی نشان شده است. دیده می‌شود که متوسط سازگاری‌ی جمعیت‌ای که از اشتراک سازگاری‌ی ارزش‌محور استفاده کرده است به طور قابل ملاحظه‌ای بالاتر از جمعیت‌ای است که از اشتراک سازگاری‌ی یک‌نواخت استفاده کرده. به‌ترین جمعیت روش اول نیز در حدود پاسخ بهینه – که در همه‌ی شرایط پاسخ صحیح تولید می‌کند و در نتیجه دارای سازگاری‌ی ۱ است– رفتار می‌کند. به‌ترین جمعیت با اشتراک سازگاری‌ی یک‌نواخت در حدود متوسط سازگاری‌ی جمعیت با سازگاری‌ی ارزش‌محور است. سازگاری‌ی پاسخ کاملا تصادفی (که در آن هر رفتار با احتمال $\frac{1}{2}$ عمل خود را انتخاب کرده و با همان احتمال NA را انتخاب می‌کند) بسیار پایین‌تر از بقیه‌ی حالت‌ها است. نکته‌ی جالب و البته طبیعی این‌که مقدار سازگاری‌ی حالت تصادفی که در حدود -0.7 است بسیار نزدیک به مقدار متوسط سازگاری در نسل اول روش‌های تکاملی است. برای محاسبه‌ی سازگاری‌ی حالت کاملا

تصادفی دو روش به کار بردنی است: روش عددی و روش تحلیلی. نتیجه‌ی روش عددی را –که با اجرای چندین باره‌ی معماری‌ای تصادفی به دست آمده– در شکل می‌بینید. برای محاسبه‌ی تحلیلی این مقدار در ساختار مورد نظر و با فرض این‌که عمل‌های $\Pi(\cdot)$ به صورت یک‌نواخت انتخاب شده باشند، یعنی:

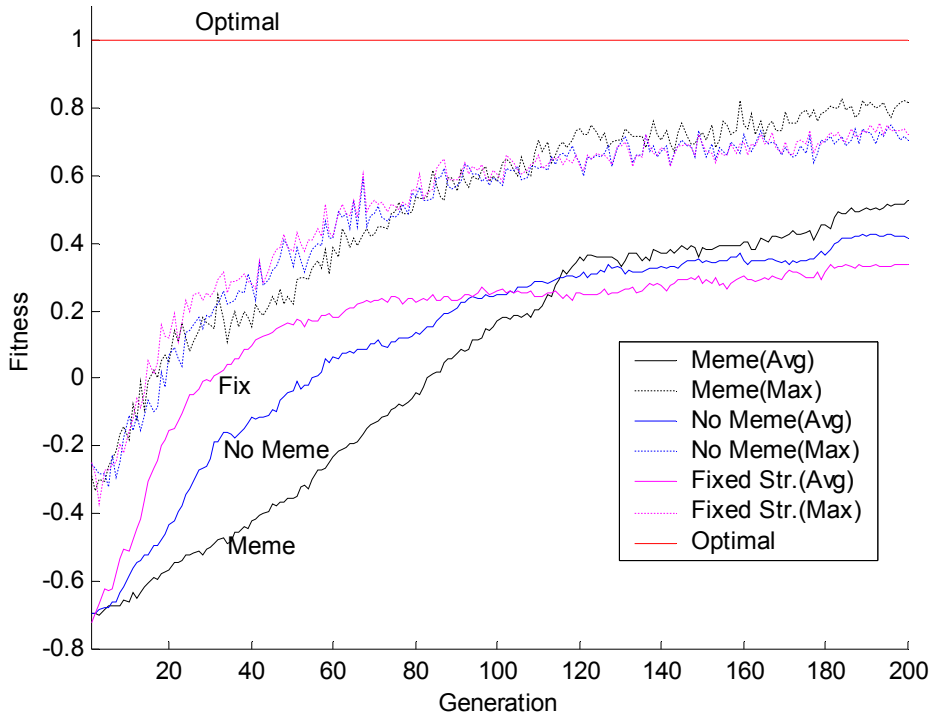
$$P(a_i \in \Pi) = P(a_j \in \Pi) \;\; \forall a_i, a_j \in \Pi \qquad (۶-۵)$$

امید ریاضی سیگنال تقویتی‌ی دریافتی را می‌توان بدین‌گونه محاسبه کرد:

$$\underset{\pi_{random}}{E}[r] = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2^i}\times(1) + \left(1-\frac{1}{2^i}\right)\times(-1)\right) = \frac{1}{m}\left[2\sum_{i=1}^{m}\frac{1}{2^i} - m\right] \qquad (۶-۶)$$

که در آن m مطابق معمول تعداد لایه‌ها (و رفتارها) است. جمله‌ی اول داخل اولین علامت جمع (با در نظر گرفتن $1/m$) بیان‌گر امید ریاضی‌ی پاداش‌ای است که از درست رفتار کردن لایه‌ی iام حاصل می‌شود. برای درست عمل کردن آن لایه، لایه‌های بالا می‌بایست NA پیش‌نهاد دهند (هر کدام با احتمال $1/2$) و خود آن لایه نیز می‌بایست عمل درست را انتخاب کند (احتمال $1/2$). جمله‌ی دوم همان عبارت، میزان امید ریاضی‌ی دریافت تنبیه برای همان لایه است. نتیجه‌ی این مقدار برای معماری‌ی هفت لایه‌ی این بخش برابر با $0.7165-$ خواهد بود.

دلیل کیفیت برتر روش سازگاری‌ی ارزش‌محور، استفاده‌ی مناسب از دانش حاصل از یادگیری است. دانش حاصل از یادگیری که به صورت متوسطی از $V_{T(B_i)}$ها منتقل می‌شود، اطلاعات دقیق‌ای در مورد عمل‌کرد هر رفتار دارند. در حالی که معیار سازگاری یک‌نواخت –که عمل‌کرد همه‌ی رفتارها با هم را در نظر می‌گیرد– به طور دقیق میزان سهم هم رفتار را مشخص نمی‌کند. علاوه بر این مورد، مساله‌ی مورد نظر دارای ویژگی‌ی جالبی است و آن هم این‌که بهینه‌سازی‌ی سازگاری‌ی هر رفتار به طور مستقیم به بهینه‌سازی‌ی سازگاری‌ی همه‌ی رفتارها می‌انجامد. دلیل این موضوع این است که سازگارتر

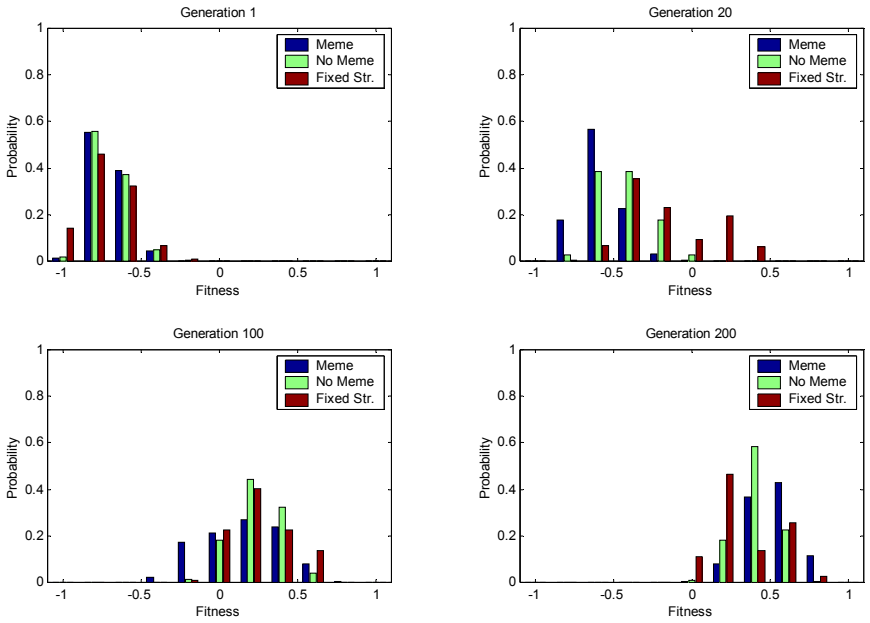


شکل ۱۱-۶. (مساله مجرد) مقایسه‌ی متوسط و حداکثر سازگاری به ازای روش‌های مختلف‌ای که از اشتراک سازگاری‌ی یک‌نواخت استفاده می‌کنند: تکامل رفتارها و یادگیری ساختار بدون استفاده از مم (آبی)، تکامل رفتارها و یادگیری ساختار با استفاده از مم (سیاه)، تکامل رفتارها و ساختار ثابت (بنفش). خطوط پررنگ متوسط سازگاری‌ی جمعیت و خطوط نقطه‌چین حداکثر سازگاری را نشان می‌دهند. حداکثر سازگاری‌ی ممکن با خط قرمز نشان داده شده است.

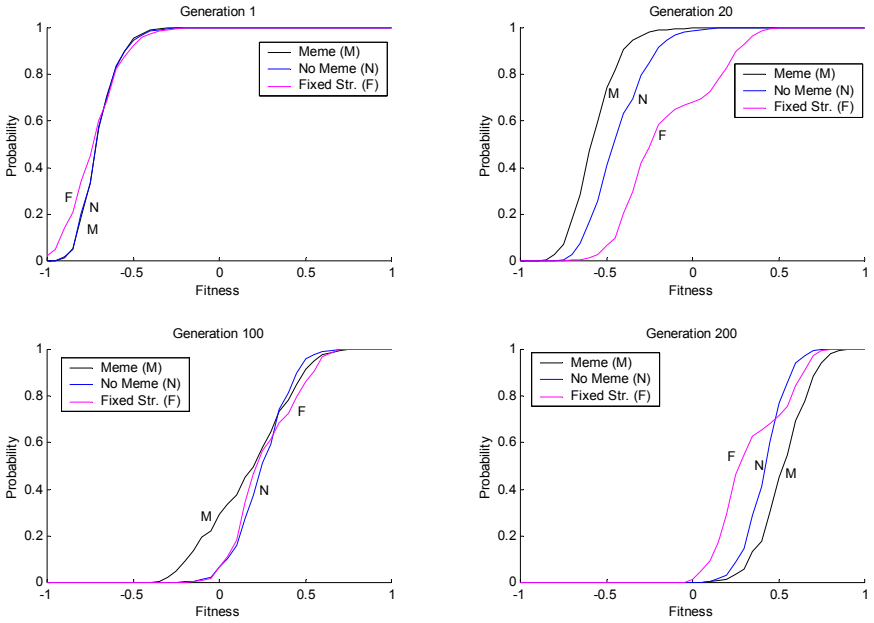
بودن هر رفتار —و بالا بودن امتیاز آن– معادل با سازگارتر بودن کل رفتارها است: ممکن نیست رفتاری سازگارتر باشد ولی کل رفتارها، معماری‌ای سازگارتر نسازند. این ویژگی در همه‌ی مسایل وجود ندارد. به همین دلایل است که در این مساله سرعت تکامل در روش اشتراک سازگاری‌ی ارزش‌محور به طور قابل ملاحظه‌ای بالاتر از سرعت تکامل در روش اشتراک سازگاری‌ی یک‌نواخت سریع‌تر است.

اینک تاثیر یادگیری بر تکامل و هم‌چنین تاثیر وجود مم بر یادگیری و در نتیجه بر تکامل بررسی می‌شود. در شکل ۱۱-۶، مقایسه‌ای بین چندین حالت مختلف‌ای که همه در بخش تکامل رفتارشان از اشتراک سازگاری‌ی یک‌نواخت سود می‌جویند انجام شده است. مطابق انتظار و تحلیل‌هایی که در بخش‌های مربوط به یادگیری نیز انجام شده است هرچه دانش اولیه‌ی بیش‌تری از مساله داشته باشیم، یادگیری به‌تر و سریع‌تر خواهد بود. برای همین انتظار می‌رود که اضافه کردن دانش ساختار مناسب باعث سرعت بخشیدن به فرآیند تکامل شود. چنین پدیده‌ای را در شکل ۱۱-۶ مشاهده می‌کنید.

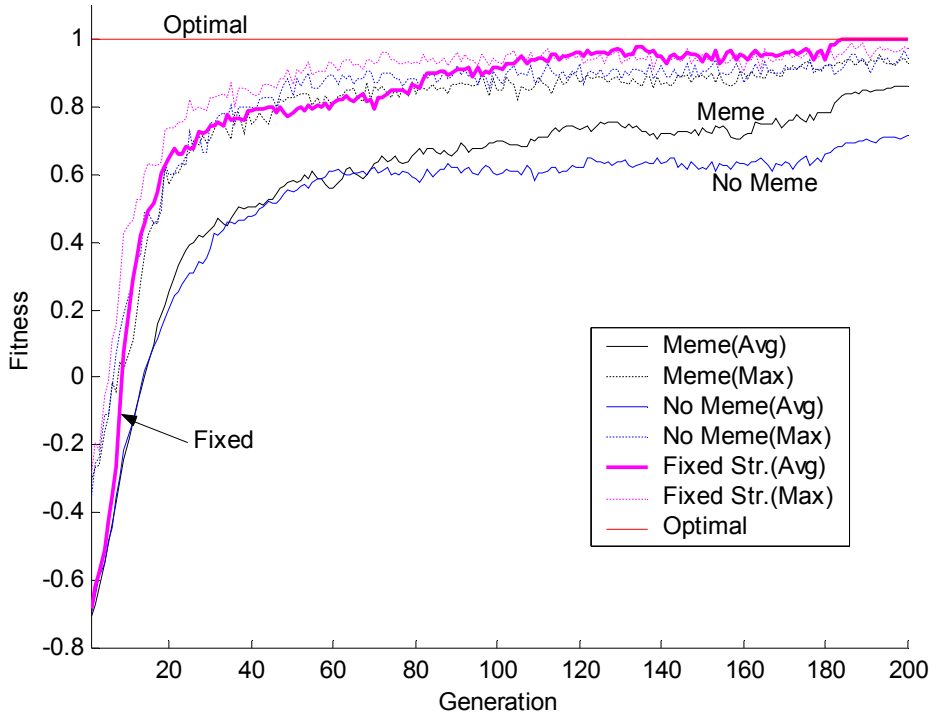
سرعت یادگیری حالت ساختار ثابت از دو حالت دیگری که از یادگیری استفاده می‌کنند بیش‌تر است. اما نکته‌ای که در این مساله مجرد وجود دارد و در ابتدای بخش توضیح دادم این است که این مساله — برخلاف مساله‌ی استفاده شده در آزمایش‌های مربوط به یادگیری– دارای پاسخ یگانه برای ساختار نیست: به ازای هر ترکیب ساختاری‌ی غیرتکراری، نگاشت متناسب حالت–عمل برای رفتارها وجود دارد. پس ثابت نگاه داشتن ساختار بیش از آن‌که عمل‌کردی ناشی از تزریق دانش صحیح داشته باشد، از پراکندگی‌ی ساختارهایی که عامل با آن‌ها کار می‌کند (و پراکندگی موجب پایین آمدن عمل‌کرد می‌شود) جلوگیری می‌کند. به عبارت دیگر اگر تعدادی رفتار برای ترتیب خاص T_1 تکامل یافته باشند، آن‌گاه در صورت‌ای که در آینده در ساختار T_2 به کار روند، بازده عامل پایین می‌آید. یادگیری برای پیدا کردن به‌ترین ترتیبْ کمی زمان لازم دارد و در نتیجه طبیعی است که کیفیت‌اش حداقل در چند اپیزود اولیه پایین باشد. حالت استفاده از یادگیری کندتر از ساختار ثابت است و حالت‌ای که یادگیری‌اش با استفاده از مم تقویت شده باشد نیز از هر دو کندتر است. اما در نسل‌های نهایی مشاهده می‌شود که حالت‌هایی که یادگیری دارند کمی به‌تر از حالت بدون یادگیری عمل می‌کند. دلیل این موضوع "شاید" این باشد که عامل با استفاده از یادگیری توانسته است ترتیب‌ای از رفتارهای هنوز––به–طور–کامل تکامل‌نیافته‌ای را بیابد که باعث عمل‌کرد به‌ترش می‌شوند. در شکل ۱۲–۶، چگالی‌ی احتمال و در شکل ۱۳–۶، احتمال تجمعی‌ی شایستگی‌ی عامل‌ها در چند نسل مختلف مشخص شده است. هیستوگرام شکل ۱۲–۶ هر چقدر در سمت راست تجمع بیش‌تری داشته باشد، به معنای وجود عامل‌هایی با کیفیت بالاتر است. تمایل به سمت راست در شکل ۱۳–۶ نیز به همان معنا است. دیده می‌شود که در نسل ۱ که هنوز تکامل‌ای صورت نگرفته عامل‌هایی که یادگیری داشته‌اند اندکی از عامل‌های با معماری ثابت به‌تر بوده‌اند. این به آن علت است که یادگیری توانسته ترتیب‌ای از رفتارها را بیابد که به بالا رفتن سازگاری‌ی عامل منجر شود. این پدیده در شکل ۱۳–۴ واضح‌تر است. از نمودارها بر می‌آید که پس از گذشت چند نسل، نمودار به سمت راست تمایل پیدا می‌کند. در ابتدای تکامل (مثلا نسل ۲۰)، تکامل با ساختار ثابت به وضوح جلوتر از تکامل با یادگیری است ولی پس از گذشت مدت زمان کافی برای تکامل کیفیت همه در حدود هم می‌شود و حتی در آخرین نسل (نسل ۲۰۰)، حالت‌های دارای یادگیری —و مخصوصا حالت تقویت شده با الگوریتم ممتیک– جلو می‌افتد.



شکل ۱۲-۶. (مساله‌ی مجرد) مقایسه‌ی چگالی احتمال سازگاری برای تکامل با اشتراک سازگاری یک‌نواخت در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (آبی تیره)، عامل‌هایی با ساختار اولیه‌ی تصادفی (سبز) و عامل‌هایی با ساختار ثابت (قرمز) انجام گرفته است.



شکل ۱۳-۶. (مساله‌ی مجرد) مقایسه‌ی توزیع احتمال تجمعی ($P\{Fitness \leq f\}$) سازگاری برای تکامل با اشتراک سازگاری یک‌نواخت در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (سیاه)، عامل‌هایی با ساختار اولیه تصادفی (آبی) و عامل‌هایی با ساختار اولیه ثابت (بنفش) انجام گرفته است. تمایلِ نمودار به سمت راست نشان‌گر بالاتر بودن احتمال تولید عامل‌های سازگارتر است.

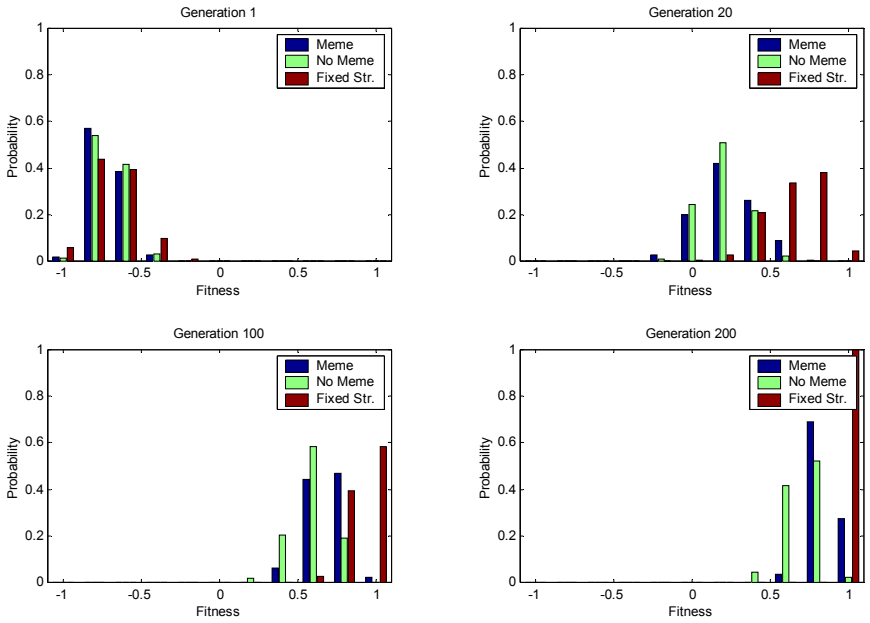


شکل ۱۴-۶. (مساله مجرد) مقایسه‌ی متوسط و حداکثر سازگاری به ازای روش‌های مختلف‌ای که از اشتراک سازگاری‌ی *ارزش‌محور* استفاده می‌کنند: تکامل رفتارها و یادگیری ساختار بدون استفاده از مم (آبی)، تکامل رفتارها و یادگیری ساختار با استفاده از مم (سیاه)، تکامل رفتارها و ساختار ثابت (بنفش). خطوط پررنگ متوسط سازگاری‌ی جمعیت و خطوط نقطه‌چین حداکثر سازگاری را نشان می‌دهند. حداکثر سازگاری‌ی ممکن با خط قرمز نشان داده شده است.

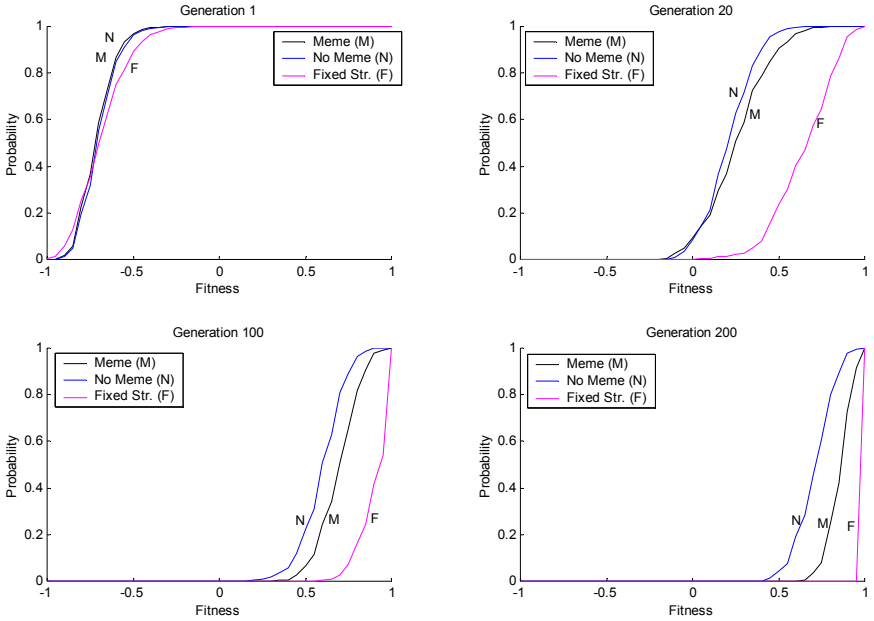
مشابه آزمایش‌های ذکر شده برای تکامل با اشتراک سازگاری ارزش‌محور نیز انجام شده است. در شکل ۱۴-۶ می‌بینید که تکامل رفتار با ساختار ثابت بسیار سریع به نتایج قابل رقابت با حالت بهینه می‌رسد. حالت‌های مربوط به یادگیری از آن کندترند و متوسط سازگاری‌ی جمعیت آن‌ها پایین‌تر از حالت ساختار ثابت است. مطابق آن‌چه پیش‌تر گفته شد، چنین نتیجه‌ای قابل انتظار است. حالت‌ای که از الگوریتم ممتیک و انتقال فرهنگ استفاده می‌کند در نهایت به سازگاری‌ی متوسط بالاتری می‌رسد. دلیل این موضوع این است که انتقال ساختارهای مناسب با رفتارهای تکامل یافته، کار یادگیری هر عامل را بسیار ساده‌تر می‌کند و عامل از همان قدم اول —و یا چند قدم ابتدایی– به ساختار مناسب برای مجموعه رفتارهای در اختیارش دست می‌یازد. به‌ترین عامل‌های ایجاد شده البته در حد و حدود هم‌اند و بسیار نزدیک به حالت بهینه. هیستوگرام (شکل ۱۵-۶) و توزیع تجمعی‌ی سازگاری (شکل ۱۶-۶) نیز اطلاعات مفیدی به ما می‌دهند. به عنوان نمونه در نسل ۲۰ دیده می‌شود که حالت ثابت به عامل‌هایی

می‌انجامد که کیفیت بسیار بالاتری نسبت به همتاهای یادگیر خود دارند. این اختلاف البته هم‌چنان در نسل‌های بعد نیز وجود دارد اما فاصله به تدریج کم می‌شود (شکل ۱۶-۶ را ببینید). هم‌چنین اختلاف حالت دارای مم و حالت بدون آن در نسل‌های بالا —که رفتارها تقریبا شکل گرفته‌اند– قابل ملاحظه است. همان‌طور که انتظار می‌رفت وجود مم باعث به‌تر شدن کیفیت کار و سریع‌تر شدن یادگیری می‌شود. وجود مم به نوعی بعضی از ویژگی‌های مثبت ساختار ثابت را به فرآیند تولید عامل اضافه می‌کند در حالی که نیازی به دانستن از-پیشِ ساختار ندارد.

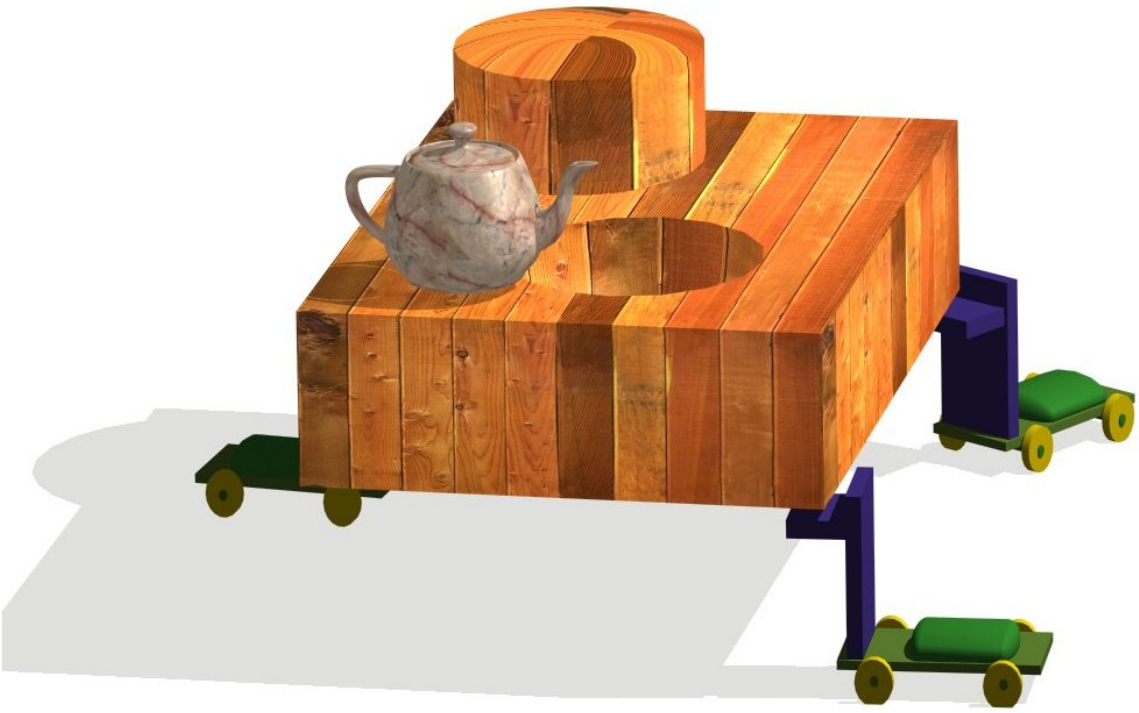
در این آزمایش‌ها دیده شد که تکامل رفتار می‌تواند جای‌گزینی برای یادگیری رفتار باشد. البته به دلیل نداشتن ارتباط مستقیم با محیط و تنها قابلیت مشاهده‌ی نتیجه‌ی کلی‌ی عمل‌کرد عامل، سرعت تکامل و میزان محاسبه‌ی لازم برای آن بسیار کم‌تر از یادگیری رفتار است. مخصوصا به این نکته توجه کنید که فضای مساله در این حالت بسیار کوچک‌تر از فضای مساله‌ی انجام شده برای آزمونِ یادگیری بوده است و عملا با مساله‌ای ساده‌تر طرف بودیم. با این حال در صورت‌ای که بتوان پاسخ بهینه‌ی مساله را به پاسخ بهینه‌ی تعدادی زیرمساله (ارزش هر رفتار) تجزیه کرد (مانند مساله‌ی مورد بررسی)، روش اشتراک سازگاری ارزش‌محور باعث سرعت بخشیدن به تکامل می‌شود.



شکل ۱۵-۶. (مساله‌ی مجرد) مقایسه‌ی چگالی احتمال سازگاری برای تکامل با اشتراک سازگاری *ارزش‌محور* در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (آبی تیره)، عامل‌هایی با ساختار اولیه‌ی تصادفی (سبز) و عامل‌هایی با ساختار ثابت (قرمز) انجام گرفته است.



شکل ۱۶-۶. (مساله‌ی مجرد) مقایسه‌ی توزیع احتمال تجمعی ($P\{Fitness \leq f\}$) سازگاری برای تکامل با اشتراک سازگاری ارزش‌محور در چندین نسل مختلف. مقایسه بین عامل‌هایی با سوگیری‌ی اولیه ساختار به کمک مم (سیاه)، عامل‌هایی با ساختار اولیه تصادفی (آبی) و عامل‌هایی با ساختار اولیه ثابت (بنفش) انجام گرفته است. تمایلِ نمودار به سمت راست نشان‌گر بالاتر بودن احتمال تولید عامل‌های سازگارتر است.



شکل ۱۷-۶. سه روبات در حال بلند کردن جسم‌ای بزرگ

۶-۲) بلند کردن جسم چند-روباتی

با وجود آن‌که مساله‌ی مجردْ سنگ محک مناسب و کافی‌ای برای ارزیابی‌ی روش‌های پیش‌نهادی است، اما حلِ مساله‌ای از دنیای واقعی نیز قابل توجه خواهد بود. مساله‌ی بلند کردن جسم توسط چند روبات را به عنوان مساله‌ی دیگر این پژوهش انتخاب کردم. این مساله پیش‌تر توسط معماری‌ی رفتارگرای سلسله مراتبی‌ای مشابه با ساختار PPSSA به کار رفته در این پژوهش توسط طراح و با روش سعی و خطا حل شده بود [Nili01]. در آن پژوهش ساختار و هر کدام از رفتارهای معماری با سعی و خطای بسیار پیدا شده بودند. از آن‌جا که هدف اصلی‌ی پژوهش فعلی یافتن روش‌هایی برای خودکار کردن فرآیند طراحی است، تلاش برای حلِ این مساله —که طراحی‌ای وقت‌گیر و دشوار داشته است- جالب توجه می‌نماید. در صورتی که روش‌های این پایان‌نامه بتواند به بازده‌ای قابل مقایسه با طراحی‌ی دستی —که حاصل هوش انسانی است- برسد، آن‌گاه می‌توان ادعا کرد که روش‌های پیش‌نهادی قابل

اعتنا هستند. پیش از توضیح بیش‌تر درباره‌ی روش کار، بیش‌تر راجع به مساله‌ی بلند کردن یک جسم توسط چند روبات توضیح می‌دهم.

فرض کنید که قرار باشد گروه‌ای از روبات‌ها جسم‌ای بزرگ و نامشخص را بلند کنند (شکل ۱۷–۶). جسم به اندازه‌ای است که هیچ کدام از روبات‌ها به تنهایی نمی‌توانند آن را در بر گرفته و بلند کنند و در نتیجه هم‌کاری‌ی چند روبات با هم ضروری می‌نماید. در این شرایط احتمالا مناسب‌ترین روش بلند کردن جسم استفاده از مکانیزم چنگالی[3] باشد. در هنگام بلند کردن جسم با توجه به موقعیت نسبی‌ی هر روبات به مرکز جرم جسم، نیروی اعمالی‌ی لازمِ هر روبات متفاوت با دیگری خواهد بود و این منجر به ناهمگنی بین رفتار روبات‌های تیم می‌شود. هر روبات باید قابلیت کار در گستره‌ای از بارگذاری‌های مختلف را داشته باشد. هم‌چنین در هنگام بلند کردن جسم هر کدام از روبات‌ها می‌باید به گونه‌ای حرکت کند تا لغزشی در نقطه‌ی اتصال‌شان با جسم به وجود نیاید. اما اگر بتوان زاویه‌ی کج‌شده‌گی[4] جسم را تا حد ممکن کوچک نگه داشت، با اضافه کردن گیره‌ای[5] در نقطه‌ی تماس موازی با سطح زیرین جسم، دیگر نیازی به حرکت کردن روبات‌ها وجود ندارد. هم‌چنین کم نگه داشته زاویه‌ی کج‌شده‌گی جسم از برخورد جسم با روبات‌ها جلوگیری می‌کند.

برای پایدار نگاه داشتن جسم در هنگام بلندکردن یا حرکت کافی است که جسم به گونه‌ای حرکت کند که Zero Moment Point (ZMP) آن درون سطح‌ای بیافتد که راس‌های آن سطح از نقطه‌ی تماس جسم و روبات تشکیل شده‌اند. مثلا اگر سه روبات داریم (مانند مساله‌ی فعلی) ZMP آن در درون مثلث نقطه‌ی تماس جسم و روبات بیافتد. حال با توجه شکل جسم و مکان مرکز جرم جسم می‌توان زاویه‌ی کج‌شده‌گی‌ای را به دست آورد که ZMP آن به مرز ناحیه‌ی ذکر شده برسد. پس اگر بتواند زاویه‌ی کج‌شده‌گی را همیشه کوچک‌تر از آن مقدار مشخص شده نگاه داشت، روبات‌ها می‌توانند بدون هیچ حرکت افقی‌ای جسم را بلند کنند، جسم به روبات‌ها برخورد نخواهد کرد و سیستم نیز پایدار خواهد بود. برای حلِ این مساله نیازی به کنترل‌کننده‌ی مرکزی نیست. نشان داده شده است که هر روبات با اطلاعات محلی‌ای که به دست می‌آورد می‌تواند این مساله را حل کند.

[3] Fork Lifting Mechanism
[4] Tilt Angle
[5] Compliance

به طور خلاصه، مساله مورد نظر، چگونگی‌ی بلند کردن هم‌کارانه‌ی جسم‌ای نامشخص به ارتفاع‌ای از پیش معلوم و در همان زمان کوچک نگه داشتن زاویه‌ی کج‌شده‌گی، توسط سه روبات یک‌سان است. در این مساله کنترل مرکزی یا ارتباطی بین روبات‌ها وجود ندارد و روبات‌ها تنها از اطلاعات محلی‌شان استفاده می‌کنند. این مساله را هم با استفاده از یادگیری و هم با استفاده از تکامل حل می‌کنم. روش‌های یادگیری را در سه مرحله‌ی مختلف بررسی می‌کنم: یافتن ساختار مناسب برای روبات‌ها با فرض داشتن رفتارهای مناسب، یادگیری نگاشت مناسب برای هر رفتار با فرض دانستن ساختار صحیح و یادگیری هم‌زمان ساختار و رفتار. ترکیب تکامل و یادگیری نیز در این مساله در دو حالت تکامل رفتار با فرض دانستن ساختار و تکامل رفتار و یادگیری ساختار توامان انجام می‌شود. آزمایش‌های مختلفی به ازای روش‌های گوناگون اشتراک سازگاری (یک‌نواخت و ارزش–محور) انجام شده و هم‌چنین تاثیر استفاده از مم بر روی بازده سیستم بررسی می‌شود.

۶-۲-۱) یادگیری ساختار

در گام اول فرض می‌کنم که مجموعه‌ای از رفتارهای مناسب در اختیار عامل قرار گرفته است و عامل می‌بایست رفتارها را به گونه‌ای بچیند تا وظیفه مورد نظر به درستی انجام شود. فرض داشتن مجموعه‌ای از رفتارهای مناسب در بعضی از موارد عملی قابل قبول است چون گاهی طراح درک نسبی‌ای از مساله‌ی مورد نظر و رفتارهای مناسب برای آن دارد ولی آگاهی‌ای نسبت به اولویت نسبی‌ی آن رفتارها و در نتیجه موقعیت‌شان در معماری ندارد. با این همه این شرط را نیز در گام‌های بعد بر می‌دارم و به عامل اجازه می‌دهم تا رفتارها را نیز یاد بگیرد.

این مساله را با دو روش یادگیری ساختار با بازنمایی مرتبه صفر و مرتبه یک حل می‌کنم. در این پژوهش از مجموعه رفتارهای مشابه با [Nili01] استفاده شده است. اگر $z(k)$ را ارتفاع نقطه‌ی تماس روبات–جسم، $v(k)$ سرعت بالا رفتن آن نقطه و $\tau(k)$ به عنوان زاویه‌ی کج‌شده‌گی‌ی روبات در گام kام بگیریم[6]، رفتارها بدین‌گونه توصیف می‌شوند:

[6] توجه کنید که همه‌ی این مقادیر به صورت محلی قابل اندازه‌گیری‌اند.

Push more: $v(k+1) = v(k) + \Delta v$ (۶-۷)

Do not go fast: if $v(k) > v_{\max}$ then $v(k+1) = v_{\max}$ else do nothing (۶-۸)

Stop at goal: if $z(k) \geq z_{goal}$ then stop ($v(k+1) = 0$) (۶-۹)

Hurry up: if $\tau(k) > \tau_0$ and the robot is the lowest one then $v(k+1) = \min(v(k) + \Delta v, v_{\max})$ (۶-۱۰)

Slow down: if $\tau(k) > \tau_0$ and the robot is the highest one then $v(k+1) = \max(v(k) - \Delta v, 0)$ and if it is the middle one $v(k) = \max(v(k) - \Delta v/2, 0)$ (۶-۱۱)

در شبیه‌سازی‌ی آزمایش‌های انجام گرفته برای مساله‌ی بالا بردن جسم توسط چند روبات از پارامترهای $\Delta v = 1$، $v_{\max} = 5$، $z_{goal} = 3$ و $\tau_0 = 5^\circ$ استفاده شده است. هم‌چنین در شبیه‌سازی‌ها هر واحد زمانی $\Delta T = 0.005$ بوده است.

یکی از مسایل مهم در یادگیری تقویتی طراحی‌ی سیگنال تقویت مناسب است ([Dorigo94]، [Dorigo97] و [Ng99]. فصل ۸ این پایان‌نامه را ببینید). عمل‌کرد قابل قبولِ عامل به شرطی حاصل می‌شود که همه‌ی آن‌چه مورد نظر طراح بوده است به نوعی در سیگنال تقویت گنجانده شده باشد. اما این کار مخصوصا به هنگامی که مساله‌ی مورد نظر کمی پیچیده باشد و مثلا بیش از یک هدف در آن مورد نظر باشد کاملا دشوار می‌شود. تاکنون راه کلی‌ای برای طراحی‌ی آسان این سیگنال به وجود نیامده است و در اکثر موارد با استفاده از روش سعی و خطا سعی در پیدا کردن سیگنال تقویت مناسب می‌کنند. برای مساله‌ی یادگیری ساختار سیگنال تقویت زیر مناسب تشخیص داده شد[۷]:

[7] توجه کنید که سیگنال تقویت به روبات فعلی اعمال می‌شود. در نتیجه مقادیری چون $z(k)$ و $v(k)$ می‌بایست به صورت مقادیر مکان و سرعت نقطه‌ی تماس روبات فعلی با جسم تفسیر شوند.

$$r_k = r_k^{\tau_1} + r_k^{\tau_2} + r_k^{z_1} + r_k^{z_2} + r_k^{v} \quad (۶-۱۲)$$

$$r_k^{\tau_1} = \begin{cases} 1 & \tau(k) - \tau(k-1) < -0.5 \\ -0.1 & \text{otherwise} \end{cases} \quad (۶-۱۳)$$

$$r_k^{\tau_2} = \begin{cases} \dfrac{1}{\sqrt{k}} & \tau(k) < \tau_0 \\ -0.1\sqrt{k} & \text{otherwise} \end{cases} \quad (۶-۱۴)$$

$$r_k^{z_1} = \begin{cases} 1 & |\, z(k) - z_{goal} \,| < 0.5 \\ -0.1 & \text{otherwise} \end{cases} \quad (۶-۱۵)$$

$$r_k^{z_2} = -1 \qquad \mathrm{z(k)} > \mathrm{z_{goal}} + 0.2 \quad (۶-۱۶)$$

$$r_k^{v} = -0.1 \qquad \mathrm{v(k)} > \mathrm{v_{max}} \quad (۶-۱۷)$$

رابطه‌ی (۶–۱۲)، سیگنال تقویت‌ای است که عامل دریافت می‌کند. این سیگنال از چندین بخش مختلف تشکیل شده است که هر کدام برقراری‌ی بخش‌ای از معیارهای طراحی را تقویت می‌کنند. همان‌طور که دیده می‌شود، (۶–۱۳) کم شدن زاویه‌ی کج‌شده‌گی را پاداش داده و زیاد شدن‌اش را تنبیه می‌کند. رابطه‌ی (۶–۱۴) کوچک بودن زاویه‌ی کج‌شده‌گی را پاداش داده و زیاد بودن‌اش را تنبیه می‌کند. این بخش از سیگنال تقویت کم بودن زاویه در شروع بلند کردن را بیش از کم بودن آن در انتهاب اپیزود تقویت می‌کند و به طور عکس، زیاد بودن زاویه را در انتهای اپیزود بیش از زیاد بودن آن در ابتدای مسیر تنبیه می‌کند. با این نوع تقویت، عامل یاد می‌گیرد تا هر چه سریع‌تر زاویه کج‌شده‌گی را کاهش دهد. رابطه‌ی (۶–۱۵) نزدیکِ هدف بودن را پاداش داده و دور بودن از آن را تنبیه می‌کند و (۶–۱۶) نیز عبور از هدف و نایستادن در نزدیک‌ی آن را تنبیه می‌کند. در نهایت (۶–۱۷) حرکت بسیار سریع روبات را تنبیه می‌کند.

برای محک زدن کیفیت عمل‌کرد سیستم، دو معیار مختلف طراحی شده است. اولین آن‌ها —به طور تقریبا بدیهی و مرسوم– متوسط سیگنال تقویتی (r_k) دریافتی توسط عامل است. این مقدار نشان می‌دهد که یادگیری تا چه حدی موفق بوده است. اما توجه به این نکته مهم است که حتی اگر سیگنال تقویت نادرست طراحی شده باشد این مقدار به خاطر عمل یادگیری بیشینه می‌شود در حالی که ممکن است بیشینگی‌ی آن معادل با عمل‌کرد مطلوب —از دید طراح– نباشد. به همین دلیل، معیار

رفتارگرایانه‌ای طراحی شده است که نمایان‌گر رفتار مطلوب عامل است. این معیار با توجه به آن‌چه از رفتار روبات بالابرنده‌ی "خوب" انتظار داریم طراحی شده است:

$$\text{Performance} = \text{Goal Closeness} + \tfrac{1}{3}\,\text{Tilt Angle Stabilization} \qquad (۶-۱۸)$$

$$\text{Goal Closeness} = \frac{1}{\text{N}}\sum_{k=1}^{N} w_k \left[\left|z_k - z_{goal}\right| < 0.5\right] \qquad (۶-۱۹)$$

$$\text{Tilt Angle Stabilization} = \frac{1}{\text{N}}\sum_{k=1}^{N} w_k \left[\tau(k) < 10^{\circ}\right] \qquad (۶-۲۰)$$

که در آن

$$w_k = \begin{cases} 1 & 0 \le k \le \dfrac{2}{3}N \\ 2 & \dfrac{2}{3}N < k \le N \end{cases}. \qquad (۶-۲۱)$$

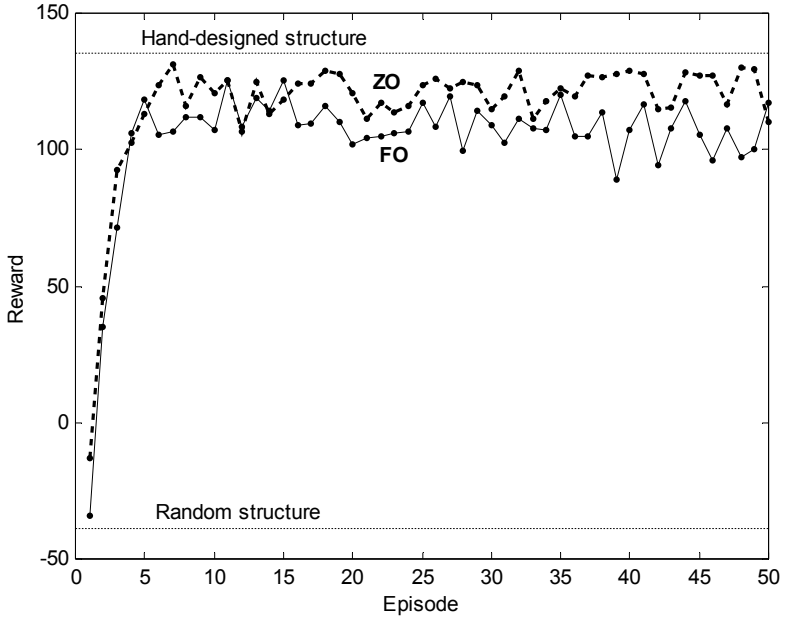
و N نیز طول اپیزود است. این معیار نزدیکی به هدف و کم بودن زاویه‌ی کج‌شده‌گی در انتهای مسیر را مطلوب می‌دارد.

در شکل ۱۸-۶، متوسط پاداش کسب شده‌ی عامل یادگیر با بازنمایی مرتبه صفر و مرتبه یک در طول ۵۰ اپیزود با ساختار طراحی شده توسط انسان[8] و هم‌چنین ساختار تصادفی مقایسه شده است. مقایسه‌ای مشابه با متوسط معیار رفتارگرایانه در شکل ۱۹-۶ نشان داده شده است. از آن‌جا که تعداد اپیزودها برخلاف مساله‌ی مجرد بسیار کم است و هم‌چنین هر اپیزود در این مساله، تجربه‌ای کامل است (سعی در بلند کردن جسم از ابتدا تا موفقیت یا شکست) از هیچ فیلتر متوسط‌گیر لغزان[9] استفاده نشده است. مشاهده می‌شود که هر دو روش یادگیری مبتنی بر دو بازنمایی مختلف عمل‌کرد مطلوبی داشته‌اند و روبات‌ها در اپیزودهای اندکی (به طور متوسط کم‌تر از ۱۰ اپیزود) توانسته‌اند به راه حل مساله دست

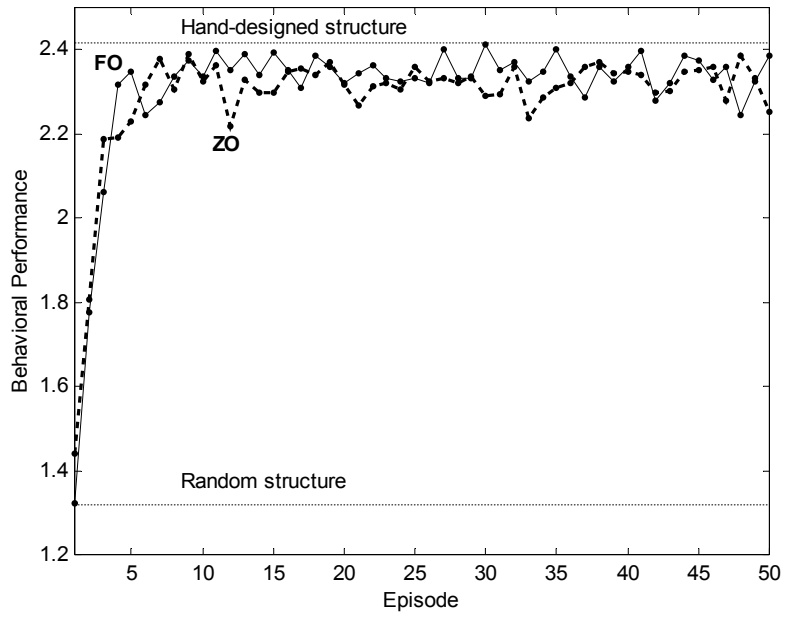
[8] ساختار طراحی شده توسط انسان این بوده است: $T = \left\lfloor B_{\text{Push more}} \quad B_{\text{Don't go fast}} \quad B_{\text{Hurry up}} \quad B_{\text{Slow down}} \quad B_{\text{Stop}} \right\rfloor$

[9] Moving Average

یابند. کیفیت کار هر دو بازنمایی قابل قبول است و هیچ کدام به طور مشخص‌ای برتر از دیگری نیست. هم‌چنین سطح متوسط مقادیری که ساختارهای تصادفی کسب می‌کنند به وضوح پایین‌تر از سطح عامل‌های یادگیرنده است و می‌توان مطمئن بود که نتایج حاصل از به خاطر قابلیت یادگیری بوده است و کیفیت اتفاقی نتایج. هم‌چنین مشاهده می‌شود که هر دوی معیارهای متوسط سیگنال تقویت دریافتی و معیار عمل‌کرد رفتارگرایانه در طول یادگیری افزایش می‌یابند. بررسی‌ی دقیق‌تر نشان می‌دهد که این دو دارای ضریب همبستگی‌ی 0.99 هستند. این نشان می‌دهد که بهینه‌سازی‌ی سیگنال تقویت طراحی شده باعث بهینه‌گی‌ی معیار رفتارگرایانه می‌شود و در نتیجه سیگنال تقویت طراحی شده تا حد قابل قبولی معادل همان معیار عمل‌کرد رفتارگرایانه‌ی مطلوب است. به مانند قبل جزییات شبیه‌سازی را در جدول ۱–۶ می‌توانید مشاهده کنید.



شکل ۱۸-۶. (مساله‌ی بلندکردن جسم) مقایسه متوسط سیگنال تقویتی دریافتی بین روش‌های یادگیری ساختار مبتنی بر بازنمایی مرتبه صفر (ZO) و بازنمایی مرتبه یک(FO)، ساختار از پیش طراحی شده و ساختار اتفاقی. رفتارها توسط انسان طراحی شده‌اند.



شکل ۱۹-۶. (مساله‌ی بلندکردن جسم) مقایسه معیار رفتارگرایانه بین روش‌های یادگیری ساختار مبتنی بر بازنمایی مرتبه صفر (ZO) و بازنمایی مرتبه یک(FO)، ساختار از پیش طراحی شده و ساختار اتفاقی. رفتارها توسط انسان طراحی شده‌اند.

۶-۲-۲) یادگیری رفتار و یادگیری هم‌زمان رفتار/ساختار

در گام بعد فرض وجود جعبه‌ابزاری از رفتارها را به کنار می‌نهیم و اجازه می‌دهیم عامل خود نگاشت مناسب حالت–عمل هر رفتار را یاد بگیرد. در این گام ساختار از پیش مشخص است. فرض این مرحله دانستن فضای ورودی و خروجی‌ی هر رفتار است. آن فضاها را بدین‌گونه تعریف می‌کنم:

Push more: $S'_{\text{Push more}} = \{\varnothing\},\ A'_{\text{Push more}} = \{v(k+1) = 1, NA\}$ (۶-۲۲)

Do not go fast[10]**:** $S'_{\text{Don't go fast}} = \left\{v(k) \middle| v(k) \in \{0^-, 1, 2, 3, 4, 5^+\}\right\},$

$$A'_{\text{Don't go fast}} = \{v(k+1) = \max(0, v(k) - 0.5), NA\} \quad (۶-۲۳)$$

Stop at goal: $S'_{\text{Stop}} = \{"z(k) - z_{goal} < -0.2", "z(k) - z_{goal} \geq -0.2"\},$

$$A'_{\text{Stop}} = \{v(k+1) = 0, NA\} \quad (۶-۲۴)$$

Hurry up:

$$S'_{\text{Hurry}} = \{\text{lowest robot, middle robot, highest robot}\} \times \{"\tau < \tau_0", "\tau > \tau_0"\},$$

$$A'_{\text{Hurry}} = \{v(k+1) = 1, NA\} \quad (۶-۲۵)$$

Slow down:

$$S'_{\text{Slow}} = \{\text{lowest robot, middle robot, highest robot}\} \times \{"\tau < \tau_0", "\tau > \tau_0"\},$$

$$A'_{\text{Slow}} = \{v(k+1) = -1, NA\} \quad (۶-۲۶)$$

و به عنوان سیگنال تقویت نیز از تابع زیر استفاده می‌کنم:

$$r_k = r_k^{\tau} + r_k^{z_1} + r_k^{z_2} + r_k^{v} \quad (۶-۲۷)$$

$$r_k^{\tau} = -0.5\big(\tau(k) - \tau(k-1)\big) \qquad \text{if } \tau > \tau_0 \quad (۶-۲۸)$$

[10] منظور از 5^+ همه‌ی عددهای بزرگ‌تر یا مساوی با ۵ است و منظور از 0^- همه‌ی عددهای کوچک‌تر یا مساوی با ۰ است.

$$r_k^{z_1} = 0.1 \qquad \text{if } \left[\left|\mathrm{z(k)} - \mathrm{z}_{\mathrm{goal}}\right| < 0.2\right] \wedge \left[\left(z(k) - z(k-1)\right) < 0.001\right] \tag{۶-۲۹}$$

$$r_k^{z_2} = 5\left(z(k) - z(k-1)\right) \qquad \text{if } z(k) < z_{goal} \tag{۶-۳۰}$$

$$r_k^{v} = -0.1 \qquad \text{if } v(k) > v_{\max} \tag{۶-۳۱}$$

برای یادگیری عامل از سیگنال تقویت (۶-۲۷) استفاده می‌کنم که خود از اجزای ساده‌تری تشکیل شده است. هر کدام از آن اجزا یکی از هدف‌های مساله را تامین می‌کند. به مانند پیش، سیگنال تقویت با استفاده از فرآیند سعی و خطا –و البته در نظر گرفتن دانش طراح نسبت به مساله– ایجاد شده است. سیگنال تقویت طراحی شده به سه جزء کارکردی قابل تقسیم است: جزءای که به کنترل زاویه‌ی کج‌شده‌گی می‌پردازد (۶-۲۸)، جزءای که مسوول کنترل ارتفاع جسم است ((۶-۲۹) و (۶-۳۰)) و جزءای که جلوی حرکت تند روبات‌ها را می‌گیرد (۶-۳۱). رابطه‌ی (۶-۲۸) با پاداش دادن (تنبیه کردن) کاهش (افزایش) زاویه‌ی کج‌شده‌گی به هنگامی که آن زاویه بزرگ‌تر از اندازه‌ی از پیش تعیین شده‌ی τ_0 باشد سعی در تقویت یادگیری کنترل زاویه می‌کند. رابطه‌ی (۶-۲۹) در ساکن ماندن نقطه‌ی تماس روبات–حسم را در نزدیکی‌ی هدف تشویق می‌کند. رابطه‌ی (۶-۳۰) بالا رفتن محرک روبات را به هنگامی که نقطه‌ی تماس روبات–جسم پایین‌تر از هدف است پاداش داده و پایین آمدن آن را تنبیه می‌کند. در نهایت، رابطه‌ی (۶-۳۱) حرکت تند روبات را تنبیه می‌کند. البته با توجه به ویژگی‌ی فضای عملِ رفتارها، روبات‌ها هیچ‌گاه نمی‌توانند تا آن حدی سریع حرکت کنند که این مکانیزم فعال شود.

آخرین گام یادگیری هم‌زمان رفتار و ساختار است. سیگنال تقویت به کار رفته برای یادگیری ساختار و رفتار یک‌سان و برابر با سیگنال به کار رفته در یادگیری رفتار –با ساختار ثابت– است (رابطه‌ی (۶-۲۷)). در نتایج این بخش از روش یادگیری ساختار با بازنمایی مرتبه صفر استفاده شده است.

برای محک قرار دادن روش‌های یادگیری رفتار و یادگیری هم‌زمان رفتار/ساختار از دو معیار متوسط پاداش دریافت شده و معیار رفتارگرایانه‌ی (۶-۱۸) بهره می‌جوییم. مقایسه در دو فاز یادگیری و آزمون انجام می‌شود. در فاز اول، این دو معیار در اپیزودهای آغازین یادگیری که میزان کاوش نسبتا زیاد است انجام می‌پذیرد و در فاز دوم این معیارها در شرایط که چندین اپیزود از یادگیری گذشته و کاوش به پایان

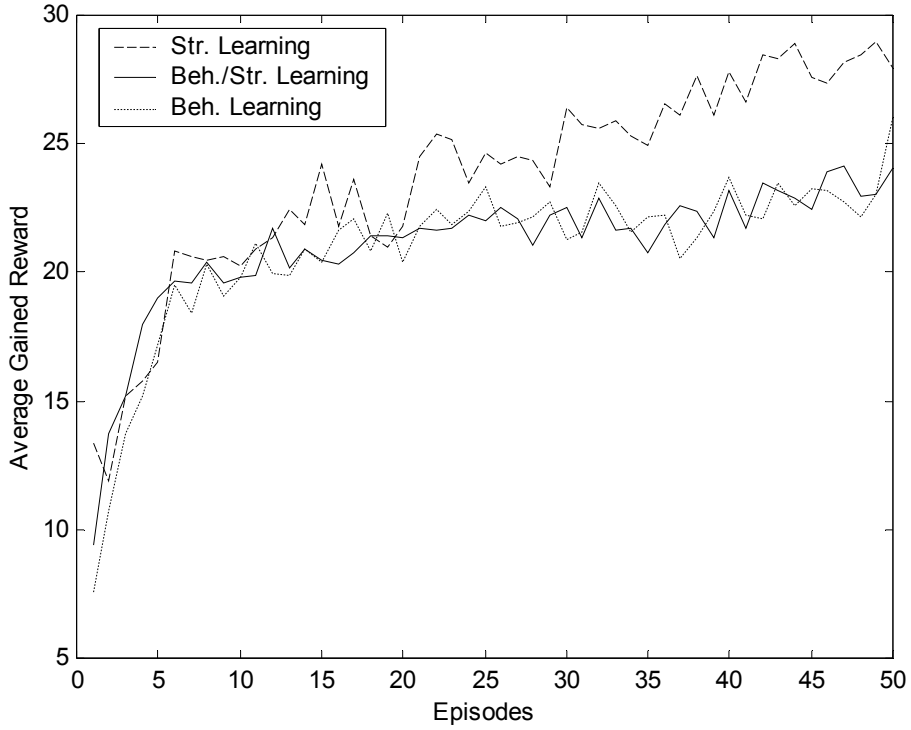
رسیده بررسی می‌شوند. توجه کنید که در فاز آزمون یادگیری متوقف نمی‌شود و ساختار و یا رفتارها را ثابت نمی‌کنم — گرچه به دلیل وجود فاز یادگیری، در این فاز رفتارها و ساختار تقریبا یاد گرفته شده‌اند و تغییر قابل ملاحظه‌ای نمی‌کنند. با وجود آن‌که جزییات آزمایش در جدول ۱-۶ آمده است، اما اشاره به چند نکته مفید می‌نماید. ابتدا این‌که توزیع احتمال مکان‌های شروع حرکت در دو فاز یک‌سان است. در یادگیری هم‌زمان رفتار و ساختار، تغییر ساختار در هر سه اپیزود یک بار صورت می‌گیرد. این تغییر آرام‌تر ساختار به آن دلیل است تا اجازه دهیم رفتارها در طول آن سه اپیزود به ثبات برسند. به یاد داشته باشید که با تغییر ساختار، مقادیر ارزش حالت–عمل رفتارها ممکن است تغییر کند. در یادگیری ساختار صرف (بدون یادگیری رفتار) این تغییر در هر اپیزود صورت می‌گیرد.

نکته‌ی مهم و قابل توجه دیگر این است که در طول آزمایش‌ها متوجه شدم که در بسیاری از موارد، عامل به خوبی وظیفه‌اش را یاد می‌گیرد اما در بعضی از موارد به پاسخ مناسب نمی‌رسد. دلایل ممکن برای این مشکل چندهدف بوده‌گی‌ی وظیفه، غیر–مارکوف بودن مساله از دید هر عامل به دلیل رویت‌پذیری جزئی و احتمالا اشکال در طراحی‌ی سیگنال تقویت —که باعث گیر افتادن در پاسخ بهینه‌ی محلی می‌شود– می‌تواند باشد. در آن شرایط، عامل معمولا پاسخ‌های ناقص‌ای چون یادگیری تنظیم زاویه بدون یادگیری توقف در کنار هدف، یادگیری توقف ولی حرکت نکردن به سمت بالا و ... را یاد می‌گرفت. با وجود آن‌که این پدیده مطلوب نیست، اما ممکن است بتوان مشکل وجود این عامل‌های درست یادنگرفته را با راه‌نمایی‌ی فعال عامل در طول یادگیری و بیش‌تر در معرض حالت‌های مشکل‌دار قرار دادن‌اش حل کرد[۱۱].

در شکل‌های در پیش رو مقایسه‌ای بین نتایج یادگیری رفتار، یادگیری هم‌زمان رفتار/ساختار و هم‌چنین یادگیری ساختار با رفتارهای از پیش طراحی‌شده (مشابه با آن‌چه در زیربخش پیش نمایش داده بودم)، معماری‌ی کاملا توسط انسان طراحی شده (هم ساختار و هم رفتار) و معماری‌ای تصادفی (که رفتار و ساختار در آن به تصادف انتخاب شده‌اند) آورده می‌شود[۱۲].

[11] این ایده از کارهای ژان پیاژه، روان‌شناس تکوینی، در یادگیری الهام گرفته شده است (به فصل مربوط به پیاژه در [Hergenhahn01] مراجعه کنید).

[12] ساختار از پیش مشخص طراحی شده مانند آن‌ای است که در آزمایش‌های طراحی‌ی ساختار آورده بودم و رفتارهای طراحی شده نیز مشابه آن‌ها هستند؛ یعنی:



شکل ۲۰-۶. (مساله‌ی بلندکردن جسم) مقایسه متوسط سیگنال تقویت دریافتی بین یادگیری ساختار (رفتار طراحی‌شده)، یادگیری رفتار (ساختار طراحی‌شده) و یادگیری هم‌زمان ساختار و رفتار.

در شکل ۲۰-۶ منحنی‌ی یادگیری متوسط‌گیری‌شده بر روی صد اجرای مختلف چندین روش مختلف یادگیری در ۵۰ اپیزود اول نمایش داده شده است. دیده می‌شود که رفتار متوسط یادگیری رفتار و یادگیری هم‌زمان رفتار/ساختار تا حدودی مشابه است اما منحنی‌ی متوسط یادگیری ساختار (با رفتارهای از پیش مشخص) بالاتر از آن دو قرار دارد. مقایسه‌ای دقیق بین این روش‌ها با استناد به اطلاعات آمارگان متوسط انجام‌پذیر نیست. به همین دلیل در شکل‌های بعدی از توزیع احتمال عمل‌کرد سیستم استفاده می‌شود. توزیع احتمال دانش بیش‌تری نسبت به فرآیند یادگیری در اختیارمان قرار می‌دهد.

Push more: $v(k+1)=1$; **Do not go fast:** if $v(k)>v_{\max}$ then $v(k+1)=\max(0,v(k)-0.5)$ else do nothing; **Stop at goal:** if $z(k)-z_{goal}>-0.2$ then stop ($v(k+1)=0$); **Hurry up:** if $\tau(k)>\tau_0$ and the robot is the lowest one then $v(k+1)=1$; **Slow down:** if $\tau(k)>\tau_0$ and the robot is the highest one then $v(k)=-1$.

توجه کنید که این رفتارها مشابه با رفتارهای (۷-۶)-(۱۱-۶) هستند اما به دلیل تفاوت فضای عمل‌شان تفاوت اندکی نیز بین این دو دسته رفتار وجود دارد.

شکل‌های ۲۱-۶ تا ۲۴-۶ توزیع احتمال تجمعی‌ی بیش‌تر بودن متوسط پاداش دریافتی و یا معیار عمل‌کرد رفتارگرایانه از میزان‌ای مشخص را نشان می‌دهند. برای نمونه، در شکل ۲۱-۶، احتمال این‌که عامل بیش از حد مشخص‌ای پاداش دریافت کند برای روش‌های مختلف یادگیری (و البته حالت تصادفی و از پیش طراحی‌شده) نمایش داده شده است. به عنوان نمونه‌ای برای چگونگی‌ی استفاده از این نمودار فرض کنید که می‌خواهیم درصد مواردی را که عامل‌مان با استفاده از یادگیری تقویتی به پاداش‌اش کم‌تر از ۲۰ می‌رسد پیدا کنیم. از نمودار مشخص است که حدود ٪۱۷ تجربه‌های یادگیری به پاداش‌ای کم‌تر از ۲۰ منجر می‌شود و ٪۸۳ نیز به پاداش‌اش بیش از ۲۰. احتمال دریافت متوسط پاداش کم‌تر از ۲۰ برای برای یادگیری هم‌زمان رفتار/ساختار ٪۳۰ است. مشخص است که هر چه دانش اولیه‌ی صحیح بیش‌تری داشته باشیم، احتمال رسیدن به پاسخ‌های به‌تر نیز بیش‌تر می‌شود. در نتیجه همان‌طور که در شکل ۲۱-۶ دیده می‌شود حالت‌ای که از پیش توسط انسان و با سعی و خطای بسیار طراحی شده است به دلیل آن‌که از ابتدا به بخش‌ای از دانش لازم برای حلِ مساله مجهز بوده است به پاسخ‌های بسیار به‌تری منجر شده است. هم‌چنین دیده می‌شود که معماری‌ای که در آن رفتارها و ساختار، هر دو، اتفاقی انتخاب شده‌اند بدتر از حالت‌های یادگرفته شده‌اند. عامل‌هایی که ساختارشان با روش یادگیری ساختار با بازنمایی مرتبه صفر طراحی شده به‌تر از عامل‌هایی هستند که رفتارهای‌شان یاد گرفته شده‌اند. نتایج یادگیری ساختار در مقادیر بزرگ پاداش به‌تر از یادگیری هم‌زمان است اما احتمال تولید عامل‌هایی با پاسخ‌های با کیفیت نه چندان مطلوب (کم‌تر از ۱۵) برای حالت هم‌زمان کم‌تر است. یادگیری رفتار صِرف در مقادیر کم و زیاد پاداش بدتر از یادگیری هم‌زمان عمل می‌کند اما در مقادیر متوسط از یادگیری هم‌زمان به‌تر است. هم‌چنین احتمال وجود داشتن عامل‌هایی با پاداشِ بسیار زیاد، اندکی برای حالت هم‌زمان بیش از حالت یادگیری رفتار صرف است. این بدین دلیل است که یادگیری هم‌زمان توانسته است تعدادی ساختار تخصصی شده برای بعضی از مسایل پیدا کند که به‌تر از ساختار ثابت یادگیری رفتار صرف است. در شکل ۲۲-۶، همین نمودار برای فاز دوم، فاز آزمون، رسم شده است. در این فاز، عاملْ تجربه‌ی اپیزودهای زیادی را در چنته دارد و تقریبا نگاشت صحیح رفتارها و هم‌چنین ساختار مناسب‌اش را پیدا کرده است. دیده می‌شود که این نمودار نسبت به نمودار پیشین به سمت راست

تمایل پیدا کرده و تفاوت بین حالت‌های یادگیر و حالت طراحی‌شده کم‌تر شده است. مشابه همین نمودارها برای معیار عمل‌کرد رفتارگرایانه در شکل‌های ۲۳-۶ و ۲۴-۶ آورده شده است.

نمودارهای ۲۵-۶ تا ۲۸-۶ میانگین پاداش‌های دریافت شده (یا میانگین معیار رفتارگرایانه) برای درصد مشخص‌ای از عامل‌های برتر را نمایش می‌دهند[۱۳]. مثلا با انتخاب ۰٫۳ بر روی محور افقی شکل ۲۵-۶ مشخص می‌شود که متوسط پاداش دریافتی‌ی ٪۷۰ برتر (حذف ٪۳۰ ضعیف‌تر) برابر با ۲۷ برای یادگیری ساختار و ۲۳ برای یادگیری هم‌زمان رفتار و ساختار است. اگر سمت چپ نمودارهای خیلی پایین نباشد مشخص می‌شود که تعداد حالت‌های خیلی بد محدود است. هم‌چنین نموداری تقریبا مسطح نمایان‌گر پاسخ‌های در حدود هم خواهد بود: توزیع احتمال عمل‌کرد عامل‌های ایجاد شده نازک است. بدیهی است که معماری‌ی از پیش طراحی شده و ثابت به دلیل تغییر نکردن اجزای‌اش باعث ایجاد نموداری تقریبا مسطح می‌شود. شیب اندک نمودار ایجاد شده به دلیل نقاط آغازین تصادفی‌ی تماس روبات‌ها-جسم است که باعث تغییر اندکی در بازده می‌شوند. در شکل ۲۵-۶ دیده می‌شود که حالت از پیش طراحی شده از بقیه‌ی حالت‌ها به‌تر است و یادگیری ساختار نیز برتر از یادگیری رفتار است. یک دلیل این پدیده کوچک‌تر بودن فضای فرضیه برای مساله‌ی یادگیری ساختار است – یادگیری ساختار مطلوب با فرض داشتن رفتارهای مناسب ساده‌تر از پیدا کردن رفتارهای مطلوب با فرض داشتن ساختار مناسب است. دلیل دیگر وجود مکانیزم انتخاب عمل کاوش‌گرایانه است که کیفیت عمل‌کرد عامل را در ابتدای یادگیری پایین می‌آورد. با صرف نظر کردن از انتخاب عمل کاوش‌گرایانه در فاز آزمون، تفاوت یادگیری ساختار و یادگیری رفتار کم‌تر می‌شود (شکل ۲۶-۶). در این فاز، با وجود آن‌که عمل‌کرد متوسط یادگیری رفتار پایین‌تر از عمل‌کرد متوسط یادگیری ساختار و معماری از پیش طراحی‌شده است، اما نتایج برترین عامل‌های ایجاد شده قابل مقایسه با نتایج آن دو است. به مانند قبل تعدادی (حدود ٪۴۰) از نتایج یادگیری هم‌زمان به‌تر از یادگیری رفتار صرف است و این به دلیل قابلیت انعطاف بیش‌تر ایجاد شده به دلیل یادگیری هم‌زمان رفتار و ساختار است. نکته‌ی قابل توجه دیگر تولید درصد قابل

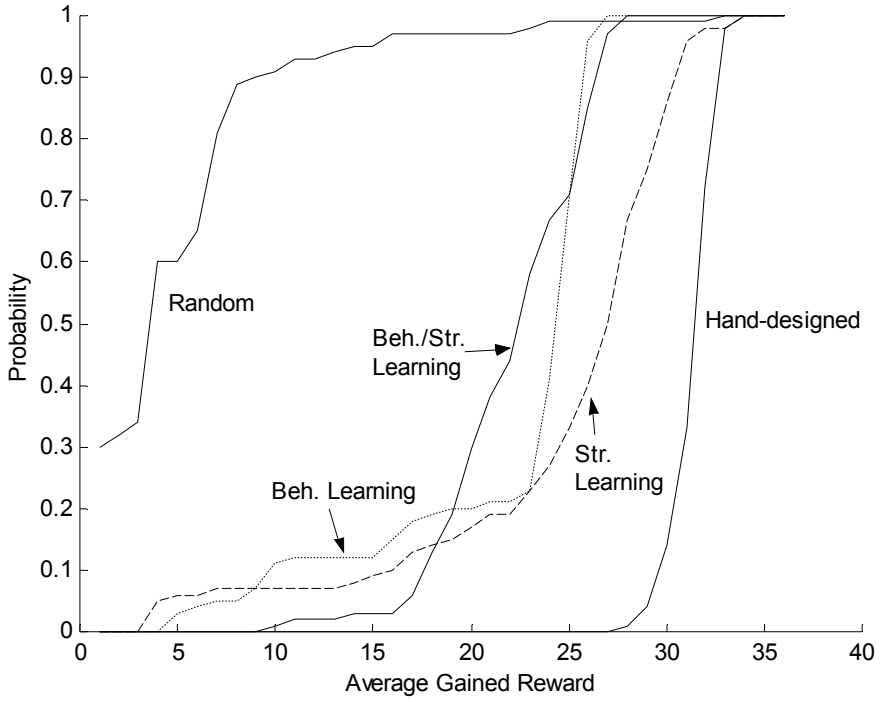
[13] قابل ذکر است که نمودارهای ۲۵-۶ تا ۲۸-۶ از نمودارهای ۲۱-۶ تا ۲۴-۶ قابل استنتاج‌اند –چون نمودارهای دسته‌ی دوم بیان‌گر توزیع احتمال پدیده هستند و در نتیجه هر آمارگان دیگری نیز از آن‌ها قابل استخراج است– اما با این حال آوردن نمودارهای اخیر روشن‌گر خواهند بود.

توجه‌ای (۹۰٪) عامل برتر از عامل‌های از پیش طراحی‌شده با استفاده از روش یادگیری ساختار است. نمودارهای ۲۷–۶ و ۲۸–۶ مشابه همین نوع نمودار برای معیار عمل‌کرد رفتارگرایانه است.

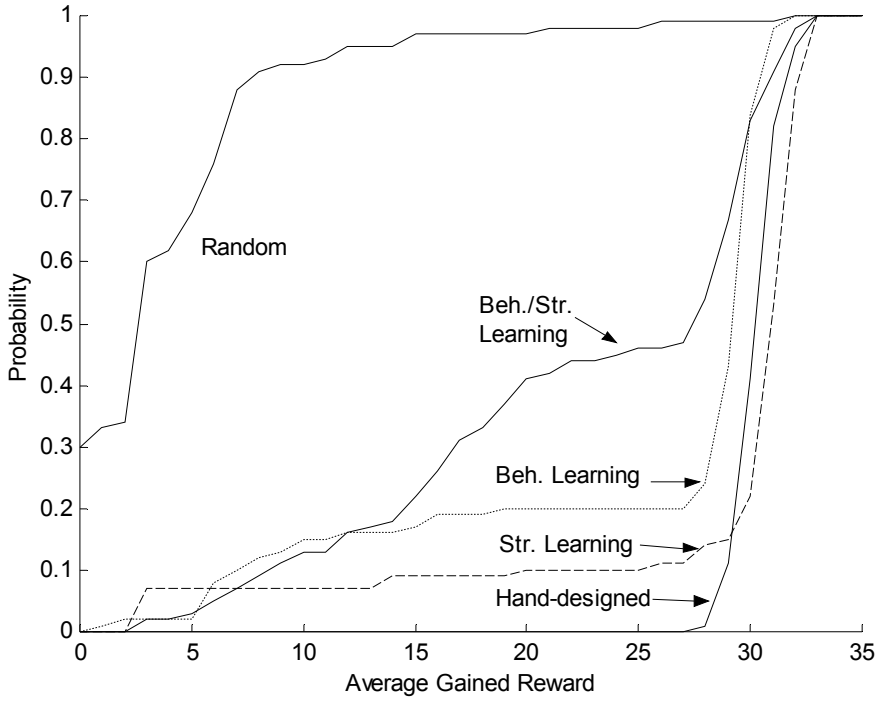
در نهایت، نمونه‌ای از مسیر نقاط اتصال جسم–روبات، زاویه‌ی کج‌شده‌گی جسم و رفتارهای کنترل‌کننده در طی عمل بلند کردن جسم را در شکل ۲۹–۶ آورده‌ام. این روبات‌ها عمل بلندکردن جسم را با استفاده از روش هم‌زمان یادگیری رفتار/ساختار آموخته‌اند. دیده می‌شود در ابتدای بلند کردن روبات‌های ۱ و ۲ ثابت‌اند و روبات ۳ رفتار ''Hurry up'' را انجام می‌دهد. سپس روبات ۱ شروع به انجام ''Slow down'' می‌کند و به سمت پایین حرکت می‌کند در حالی که روبات ۲ هم‌چنان ثابت است و روبات ۳ نیز عمل پیشین، ''Hurry up''، را انجام می‌دهد. پس از مدتی که زاویه‌ی کج‌شده‌گی به حد قابل قبول‌اش رسید، هر سه روبات شروع به هل دادن جسم به بالا (رفتار ''Push More'') می‌کنند تا به هدف برسند. پس از آن رفتار ''Don't go fast'' را انتخاب می‌کنند. این رفتار به گونه‌ای یاد گرفته شده است که در سرعت‌های پایین (سرعت‌ای که روبات عملا در آن محدوده عمل می‌کند)، هم‌چنان تصمیم به کم کردن سرعت نقاط اتصال روبات و جسم می‌کند. این کار با توجه به قید سرعت روبات‌ها (حداقل سرعت رفتار 'Don't go fast'' صفر است و جسم به کمک این رفتار نمی‌تواند به سمت پایین حرکت کند) باعث صفر شدن سرعت حرکت نقطه‌ی اتصال جسم–روبات می‌شود. در نتیجه به هنگامی که این رفتار کنترل‌کننده می‌شود، روبات توقف می‌کند.

اشاره به این نکته ضروری است که نام‌گذاری‌ی یک رفتار (مثلا رفتار "Don't go fast") الزاما به معنای انجام رفتار مورد انتظار (مثلا رفتار سریع حرکت نکردن) نیست. در فرآیند یادگیری، رفتارها به گونه‌ای یاد گرفته می‌شوند که منجر به افزایش بازده عامل —از دیدی عینی و مطابق با تابع تقویت– می‌شود که این در همه‌ی موارد معادل با انتظار ذهنی‌ی طراح (مثلا کم شدن سرعت‌های زیاد) نیست.

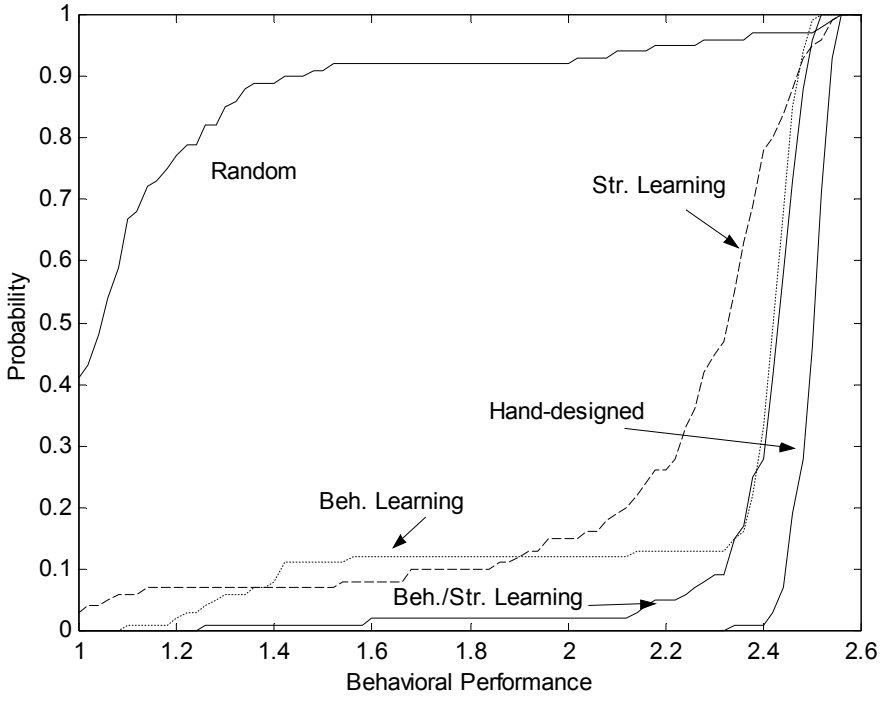
در نهایت می‌توان این‌گونه جمع‌بندی کرد که روش‌های یادگیری معرفی شده برای طراحی‌ی خودکار عامل‌های رفتارگرا می‌توانند به طراحی‌هایی قابل رقابت —و حتی به‌تر– از طراحی‌های انسانی بیانجامند.



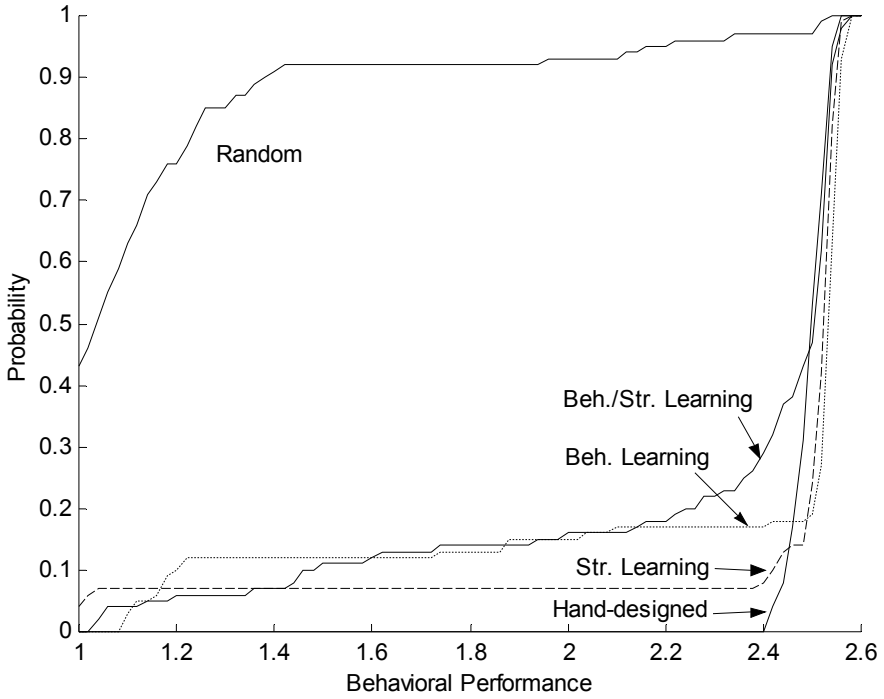
شکل ۲۱-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی سیگنال تقویت دریافتی ($P\{\text{average gained reward} \le x\}$) در فاز یادگیری (بین اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای ترکیب‌های مختلف روش‌های یادگیری.



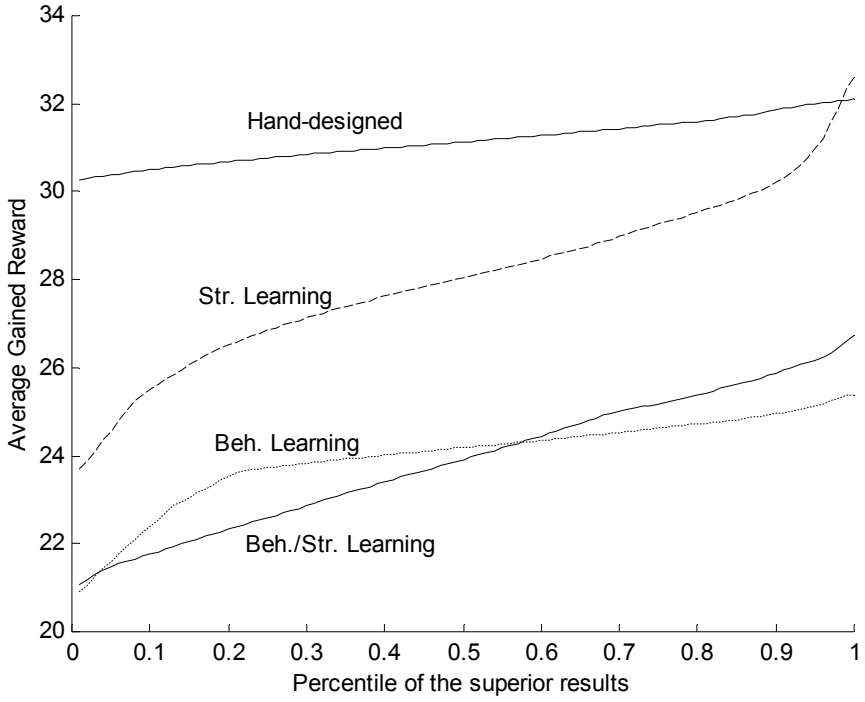
شکل ۲۲-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی سیگنال تقویت دریافتی ($P\{\text{average gained reward} \le x\}$) در فاز آزمون (بین اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای ترکیب‌های مختلف روش‌های یادگیری.



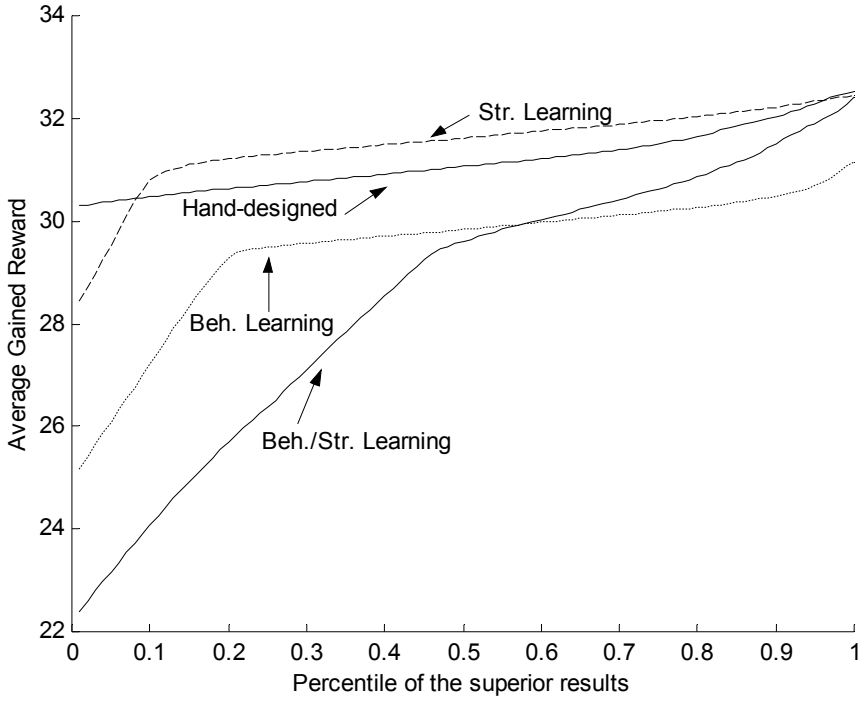
شکل ۲۳-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی معیار رفتارگرایانه ($P\{\text{behavioral performance} \leq x\}$) در فاز یادگیری (بین اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای ترکیب‌های مختلف روش‌های یادگیری.



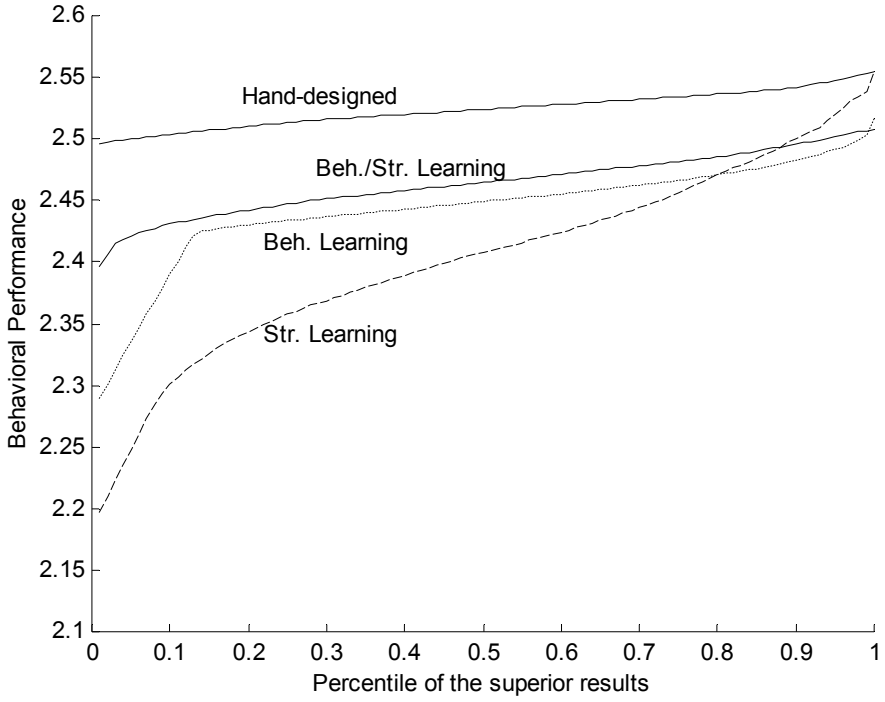
شکل ۲۴-۶. (مساله‌ی بلندکردن جسم) مقایسه توزیع تجمعی معیار رفتارگرایانه ($P\{\text{behavioral performance} \leq x\}$) در فاز آزمون (بین اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای ترکیب‌های مختلف روش‌های یادگیری.



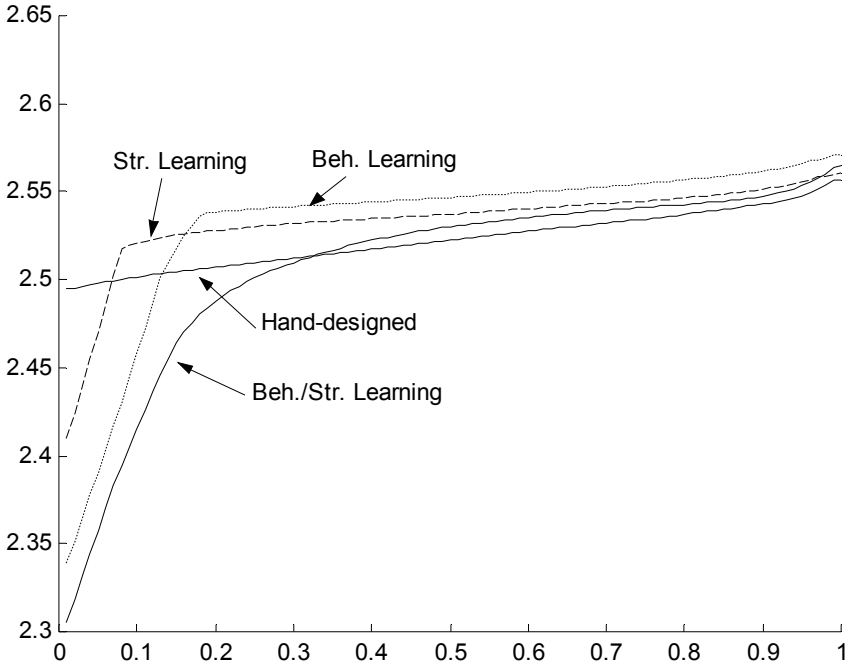
شکل ۲۵-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی پاداش دریافت شده در فاز یادگیری (اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای روش‌های مختلف یادگیری.



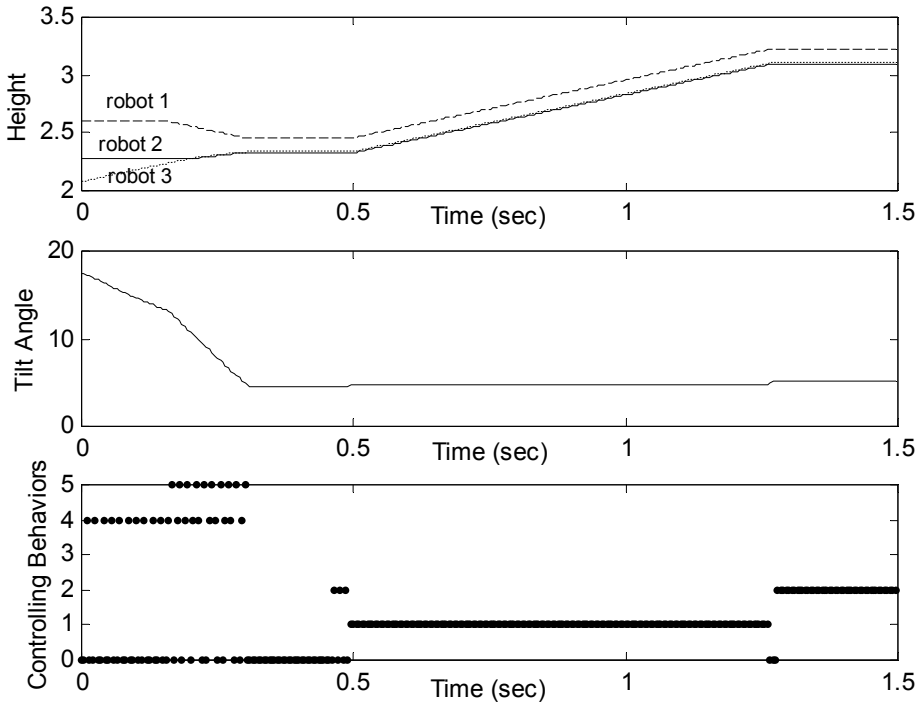
شکل ۲۶-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی پاداش دریافت شده در فاز آزمون (اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای روش‌های مختلف یادگیری.



شکل ۲۷-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی معیار رفتارگرایانه در فاز یادگیری (اپیزود ۱ تا ۵۰. کاوش برقرار است) به ازای روش‌های مختلف یادگیری.



شکل ۲۸-۶. (مساله‌ی بلندکردن جسم) متوسط تجمعی‌ی معیار رفتارگرایانه در فاز آزمون (اپیزود ۵۱ تا ۱۰۰. کاوش پایان یافته است) به ازای روش‌های مختلف یادگیری.

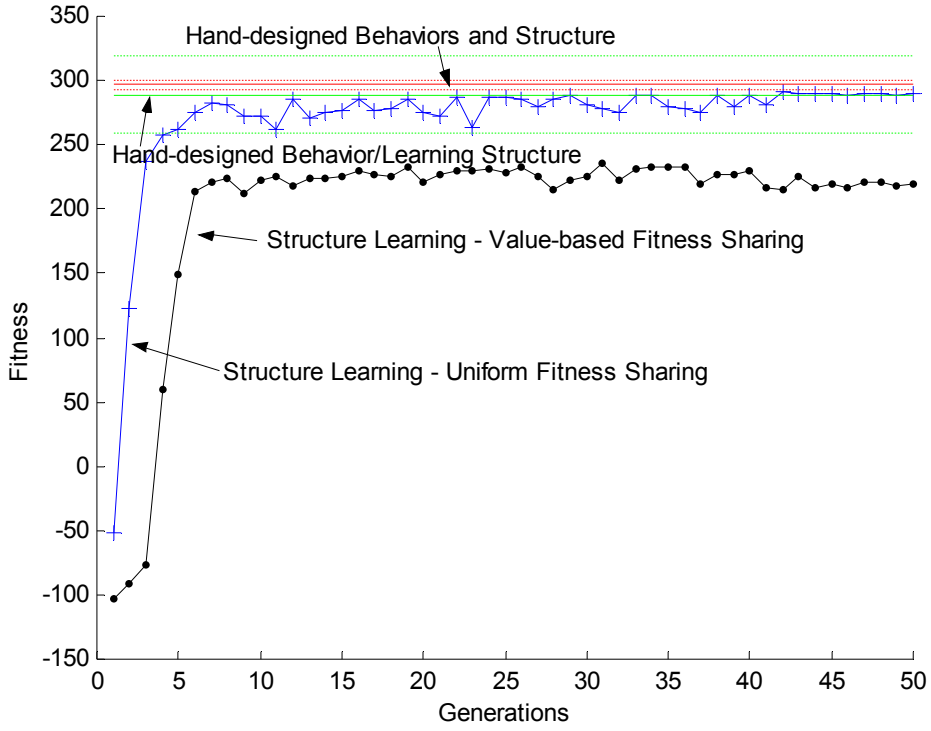


شکل ۲۹-۶. (مساله‌ی بلندکردن جسم) نمونه‌ای از مسیر حرکت نقاط تماس روبات-جسم، زاویه‌ی کج‌شده‌گی به هنگام بلند کردن جسم، و رفتارهای کنترل‌کننده‌ی روبات در هر واحد زمانی. این مسیر حاصل چندین اپیزود یادگیری هم‌زمان رفتار و ساختار است. تناظر رفتارها و شماره‌های روی نمودار پایین بدین‌گونه است:
0 (No Behavior), 1 (Push More), 2 (Don't Go Fast), 3 (Stop), 4 (Hurry up), 5 (Slow down).

۶-۲-۳) تکامل رفتار و یادگیری ساختار

در این بخش از ایده‌های تکاملی‌ی مطرح شده در فصل ۵ برای حلِ مساله‌ی بلند کردن جسم توسط چند روبات بهره می‌جویم. به مانند پیش، تکامل رفتار را در دو حالت اشتراک سازگاری‌ی یک‌نواخت و ارزش‌محور بررسی می‌کنم. علاوه بر این، تاثیر استفاده از ایده‌های الگوریتم ممتیک —که در اعضای جدید جامعه از ابتدای کنش‌شان با محیط به الگوهای فرهنگی‌ی پیشینیان‌شان مجهز هستند– نیز مطالعه می‌شود. در این مساله، دو معیار بررسی می‌شود: ۱) سازگاری‌ی چند اپیزود نهایی عامل که معادل با متوسط سیگنال تقویت دریافتی در چند اپیزود نهایی است و ۲) سازگاری کل دوره‌ی زندگی‌ی عامل که متوسط سیگنال تقویت دریافتی از ابتدا تا انتهای زندگی‌ی عامل است. معیار اول مشخص می‌کند که عامل در نهایت به چه وضعیت‌ای رسیده است؛ در حالی که معیار دوم بیان‌گر کلِ کیفیت کار عامل در طول زندگی‌اش است. مطلوب است که این دو معیار سازگاری تا حد ممکن بزرگ و به هم نزدیک باشند. در این صورت عامل در همه‌ی طول عمرش رفتار قابل قبول‌ای داشته است. اگر معیار دوم بسیار کوچک‌تر از معیار اول باشد، عامل در ابتدای زندگی تلاش‌های ناموفق‌ای برای یادگیری ساختار مناسب داشته است ولی به تدریج روش موفق حلِ مساله را فراگرفته است.

سازگاری‌ی عامل‌ها با توجه به نوع اشتراک سازگاری مشخص می‌شوند. در اشتراک یک‌نواخت، سازگاری عامل در چند اپیزود انتهایی به طور یک‌سان به همه‌ی رفتارها داده می‌شود. در اشتراک ارزش‌محور، $V_{T(B_i)}$ به هر عامل داده می‌شود که برای محاسبه‌اش از همان پارامتر $\alpha^{str}_{episode}$ای که در یادگیری ساختارِ عامل به کار رفته استفاده شده است. با توجه به نرخ کاهش $\alpha^{str}_{episode}$، این تابع ارزش بیش‌تری بر آخرین اپیزودها می‌گذارد. در هر حال برای نمایش عمل‌کرد سیستم، از دو معیار بیان شده استفاده می‌شود. پس باید به یاد داشته باشیم که معیار بهینه‌سازی و آن‌چه نمایش داده می‌شود دقیقا یک‌سان نیست. این تفاوت برای حالت اشتراک یک‌نواخت تنها در معیار متوسط سازگاری در کل زندگی‌ی عامل است در حالی که برای اشتراک ارزش‌محور، هر دو معیار نمایش داده شده با معیار بهینه‌گی تفاوت دارند.



شکل ۳۰-۶. (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی به ازای طراحی‌های مختلف: ۱) تکامل رفتارها (اشتراک سازگاری‌ی یک‌نواخت) و یادگیری ساختار (آبی)، ۲) تکامل رفتارها (اشتراک سازگاری ارزش‌محور) و یادگیری ساختار (سیاه)، ۳) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۴) رفتارها و ساختار طراحی شده (قرمز). خطوط نقطه‌چین اطراف حالت‌های طراحی‌شده (۳ و ۴) ناحیه‌ی یک انحراف معیار از متوسط را نشان می‌دهند.

در شکل ۳۰-۶ مقایسه‌ای بین سازگاری‌ی متوسط جمعیت در چند حالت مختلف نشان داده شده است. در آن شکل، نتیجه‌ی حاصل از عامل‌هایی که رفتارشان توسط یکی از دو روش تکاملی (اشتراک سازگاری‌ی یک‌نواخت و ارزش‌محور) به وجود آمده‌اند و ساختار را نیز یاد می‌گیرند، آن‌هایی که رفتارهای از پیش طراحی‌شده و با یادگیری ساختار دارند (با رفتارهای از پیش طراحی‌شده‌ای مطابق با آن‌چه در قسمت یادگیری ساختار معرفی کردم) و هم‌چنین عامل‌هایی که هم رفتار و هم ساختارشان (درست همان ساختار معرفی شده در بخش یادگیری ساختار) از پیش طراحی‌شده است با هم مقایسه شده است. مقایسه بر روی سازگاری‌ی حاصل از متوسط سیگنال تقویت پنج اپیزود نهایی یادگیری بوده است و در حالت‌ای که یادگیری‌ای وجود نداشته، هر آزمون از عامل تنها پنج اپیزود اجرا می‌شده است. برای جزییات بیش‌تر به جدول ۲-۶ مراجعه کنید.

در شکل ۳۰-۶ مشاهده می‌شود که متوسط سازگاری‌ی عامل یادگیر که رفتارهای‌اش با استفاده از اشتراک سازگاری‌ی یک‌نواخت تکامل می‌یابند بسیار سریع به حد طراحی رفتار و ساختار از پیش تعیین شده و طراحی‌ی رفتار از پیش تعیین شده هم‌راه با یادگیری ساختار می‌رسد. حالت دیگر، عامل یادگیر ساختار هم‌راه با تکامل رفتار از نوع اشتراک سازگاری‌ی ارزش‌محور، دارای متوسط پایین‌تری است. قابل ذکر است که هر دوی این روش‌ها عامل‌هایی تولید می‌کنند که سازگاری‌ای در حد به‌ترین نتایج روش از پیش طراحی‌شده دارند ولی برای جلوگیری از شلوغی‌ی شکل و به این دلیل که آن عامل‌ها از همان نسل‌های اولیه به وجود می‌آیند و در طول تکامل وجودشان (و نه تعدادشان) تغییری نمی‌کند نشان داده نشده‌اند. برخلاف مساله‌ی مجرد، این مساله دارای رفتارهایی نیست که بهینه‌گی‌ی هر کدام معادل با بهینه‌گی‌ی کل سیستم باشد. مثلا اگر نتیجه‌ی تکامل و یادگیری این باشد که معماری‌ای با ساختار $T = [... \quad PushMore \quad Stop]$ با رفتارهای "Push more" و "Stop" مطابق آن‌چه انتظار می‌رود (یعنی ۲۲-۶ و ۲۴-۶) ایجاد شود، این دو رفتار هر دو پاداش بسیار زیادی می‌گیرند؛ ولی این ترکیب به‌ترینِ ممکن برای عامل نیست چون از رفتارهای تنظیم زاویه هیچ استفاده‌ای نشده است. در نتیجه، عامل‌های خودخواه‌ای[۱۴] که حاصل این نوع اشتراک سازگاری هستند الزاما به به‌ترین پاسخ عامل نمی‌انجامند. اشتراک سازگاری‌ی یک‌نواخت با این‌که در اشتراک سازگاری چندان هوش‌مندانه عمل نمی‌کند اما به نظر می‌رسد که برای این نوع مسایل به دلیل طبیعت تیم‌نگر خود به‌تر عمل می‌کند.

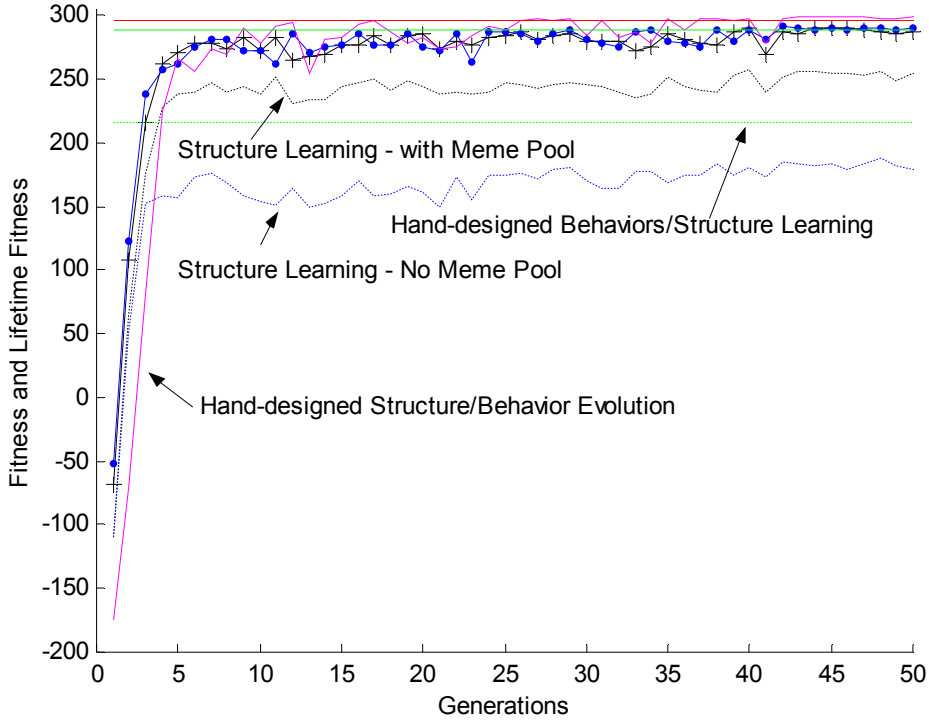
اینک بررسی‌ی دقیق‌تری روی هر کدام از این روش‌های تکامل و تاثیر یادگیری و الگوریتم ممتیک انجام می‌دهم. نتایج آزمون روش اشتراک سازگاری‌ی یک‌نواخت در حالت‌های مختلف در شکل ۳۱-۶ آمده است. مشاهده می‌شود که تفاوت محسوس‌ای بین متوسط سازگاری‌ی پنج اپیزود نهایی (خطوط پررنگ) حالت‌های ساختار از پیش مشخص، یادگیری ساختار بدون کمک مم و یادگیری ساختار با اطلاع اولیه‌ی منتقل شده توسط مم وجود ندارد. همه‌ی حالت‌ها منجر به نتایج قابل رقابت با ساختار/رفتار طراحی شده و یا رفتارهای طراحی شده/ساختار یادگرفته شده می‌شوند. اما مقایسه‌ی بین متوسط سازگاری در طول دوره‌ی زندگی‌ی عامل (خطوط نقطه‌چین) نتایج جالبی را نمایان می‌کند. وقتی

[14] Selfish

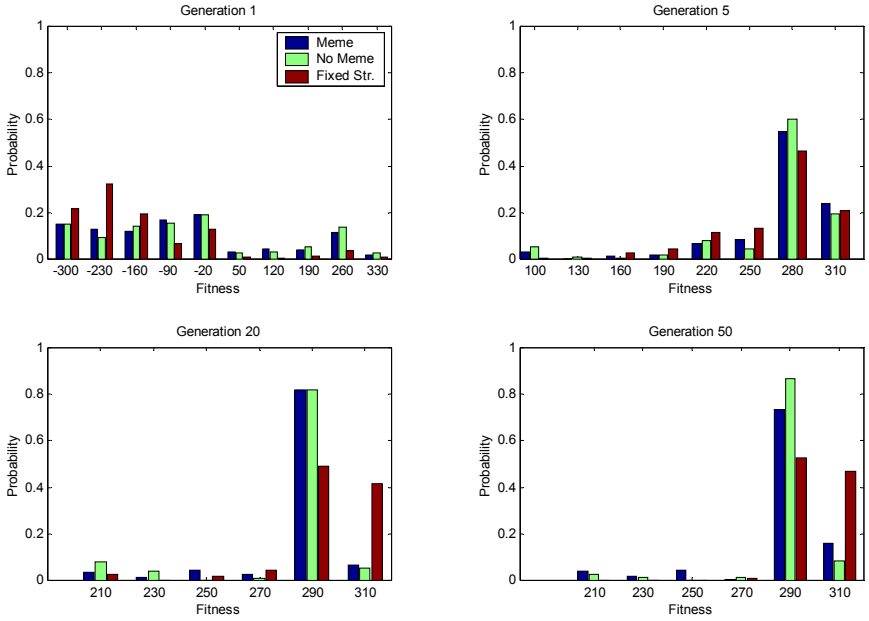
از الگوریتم ممتیک استفاده می‌کنیم، میزان سازگاری کلِ زندگی بسیار بالاتر از هنگامی است که از مم استفاده نکرده‌ایم. حتی این میزان از وقتی که طراحی‌ی رفتارها از قبل مشخص است و تنها ساختارشان یادگرفته می‌شود نیز بیش‌تر است و فاصله‌ی بین سازگاری‌ی نهایی‌ی عامل و سازگاری‌ی کلِ زندگی به میزان قابل توجه‌ای کم‌تر از حالت بدون مم است. در شکل ۳۲-۶، هیستوگرام سازگاری پنج اپیزود نهایی در چند نسل مختلف نمایش داده می‌شود. مشخص است که تفاوت آن‌چنانی‌ای بین حالت با مم و حالت بدون مم نیست (جز این‌که به نظر می‌رسد حالت با مم می‌تواند اندکی عامل بسیار خوب بیش از حالت بدون مم تولید کند)، اما در شکل۳۳-۶ که هیستوگرام سازگاری کلِ دوره زندگی‌ی عامل است تفاوت این دو حالت بسیار واضح است: الگوریتم ممتیک شانس تولید عامل‌های خوب را بسیار بالا می‌برد. نمودار توزیع تجمعی سازگاری‌ی عامل‌های ایجاد شده (برای هر دو تعریف سازگاری) در شکل ۳۴-۶ آمده است. دیده می‌شود که تفاوت بین سازگاری نهایی (خطوط پررنگ) و سازگاری کلِ زندگی (خطوط نقطه‌چین) وقتی از ایده‌ی انتقال مم بهره می‌گیریم، تا حد زیادی کم می‌شود.

نتایج آزمون روش اشتراک سازگاری‌ی ارزش‌محور در حالت‌های مختلف در شکل ۳۵-۶ آمده است. متوسط سازگاری نهایی جمعیت به ازای همه‌ی روش‌ها —یعنی تکامل با ساختار ثابت، یادگیری بدون مم و یادگیری با مم– تقریبا یک‌سان است و تنها سرعت هم‌گرایی به پاسخ مناسب برای حالت ساختار ثابت کمی بیش‌تر است. مطابق قبل تفاوت سازگاری‌ی کلِ زندگی برای وقتی که از الگوریتم ممتیک استفاده می‌کنیم کم‌تر از زمان‌هایی است که از آن بهره نمی‌جوییم. گرچه تفاوت بین دو معیار سازگاری به نسبت کم‌تر از وقتی است که از اشتراک سازگاری‌ی یک‌نواخت استفاده شده بود. در شکل ۳۶-۶ مشاهده می‌شود که سازگاری‌ی نهایی حالت‌ای که از الگوریتم ممتیک استفاده شده است در نهایت (نسل ۲۰۰) منجر به درصد بالاتری از عامل‌های برتر می‌شود و این تفاوت وقتی سازگاری‌ی کلِ زندگی‌ی عامل معیار قضاوت‌مان باشد، بسیار واضح‌تر است (شکل ۳۷-۶). توزیع تجمعی سازگاری را در شکل ۳۸-۶ آورده‌ام. برتری سازگاری کلِ زندگی برای حالت استفاده از مم نسبت به حالت بدون مم واضح است. نکته‌ی جالب —و متفاوت با اشتراک سازگاری‌ی یک‌نواخت– در این است که این تفاوت نه تنها در سازگاری‌ی کل زندگی که در سازگاری نهایی نیز دیده می‌شود. هم‌چنین تفاوت کم‌تری بین دو سازگاری نسبت به روش قبلی‌ی اشتراک سازگاری وجود دارد.

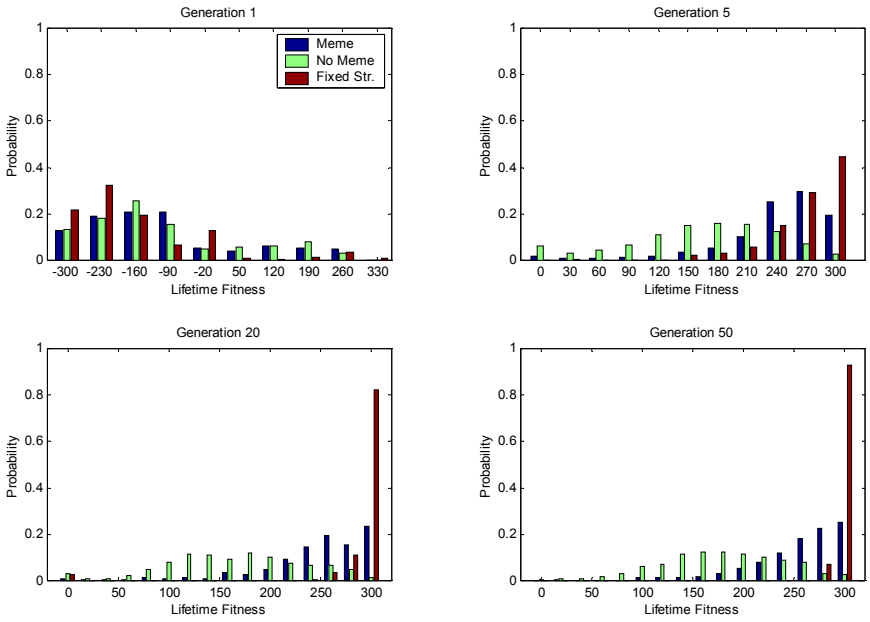
در نهایت می‌توان این‌گونه جمع‌بندی کرد که روش‌های تکامل معرفی شده برای طراحی‌ی خودکار عامل‌های رفتارگرا می‌توانند به طراحی‌هایی قابل رقابت —و حتی به‌تر– از طراحی‌های انسانی بیانجامند. هم‌چنین استفاده از ایده‌ی الگوریتم ممتیک می‌تواند باعث افزایش کیفیت عامل از ابتدای زندگی‌اش شده و یادگیری را بسیار ساده‌تر کند. در این مساله، اشتراک سازگاری‌ی یک‌نواخت به‌تر از اشتراک سازگاری‌ی ارزش‌محور عمل کرد.



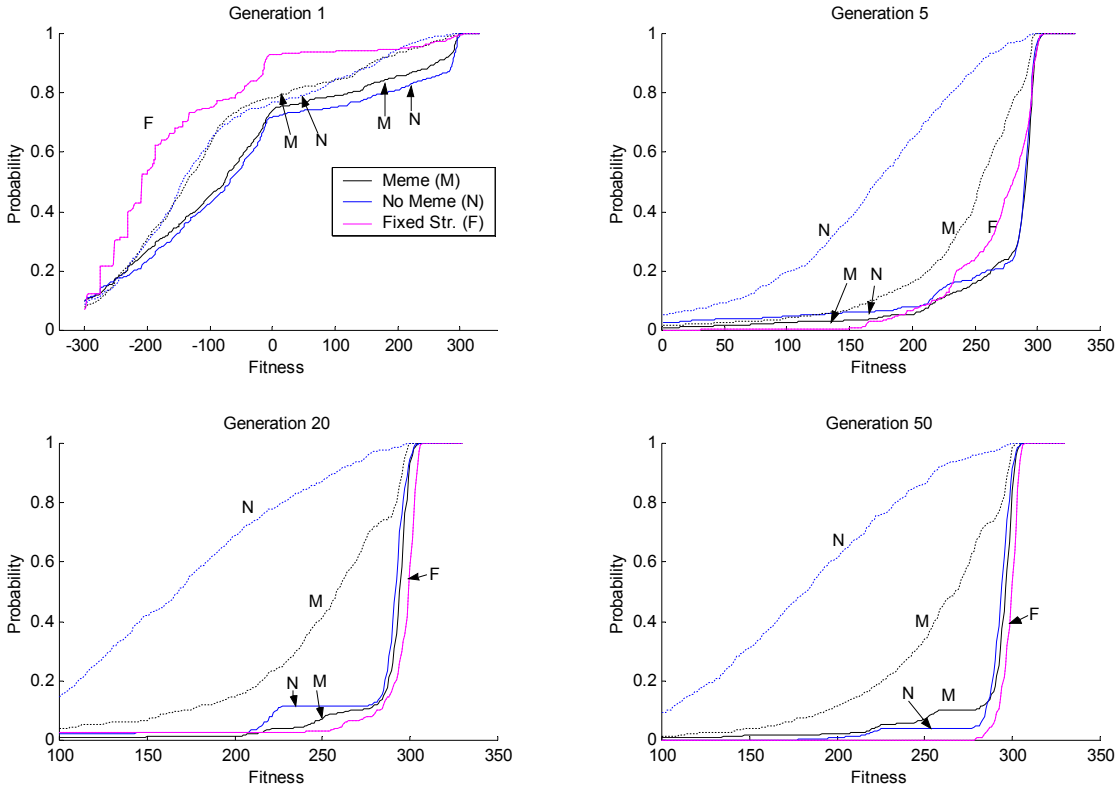
شکل ۳۱-۶. (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی و متوسط سازگاری کلِ دوره‌ی زندگی عامل در طراحی‌های مختلفی که از اشتراک سازگاری یک*نواخت* استفاده می‌کنند: ۱) تکامل رفتارها و یادگیری ساختار بدون کمک مم (آبی)، ۲) تکامل رفتارها و یادگیری ساختار با کمک مم (سیاه)، ۳) تکامل رفتارها و ساختار طراحی‌شده (بنفش)، ۴) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۵) رفتارها و ساختار طراحی‌شده (قرمز). خطوط پررنگ بیان‌گر متوسط سازگاری در پنج اپیزود انتهایی زندگی‌ی عامل هستند و خطوط نقطه‌چین سازگاری‌ی کلِ زندگی عامل را نشان می‌دهند. با وجود آن‌که عمل‌کرد آخرین اپیزودهای همه‌ی عامل‌ها تقریبا یک‌سان است اما عمل‌کرد کل دوره‌ی زندگی‌ی عامل‌هایی که از اطلاعات مم استفاده می‌کنند بسیار بالاتر است.



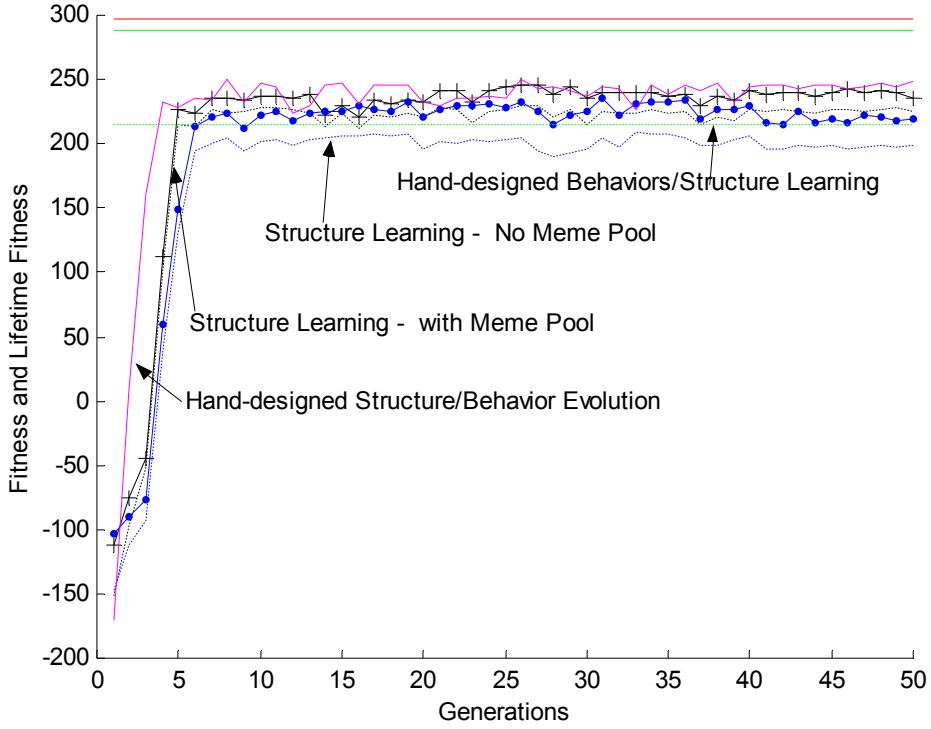
شکل ۳۲-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی یک‌نواخت. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.



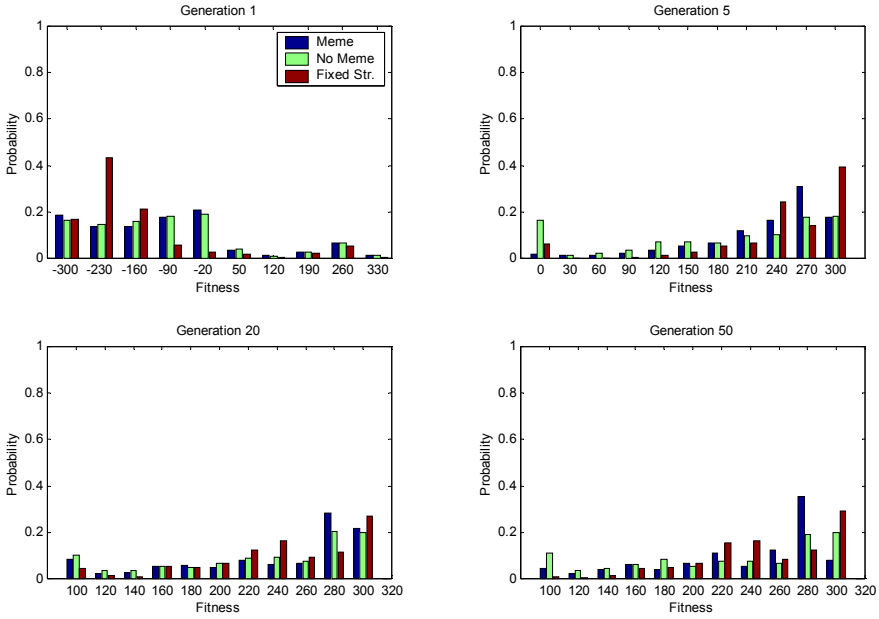
شکل ۳۳-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری کلِ دوره‌ی زندگی عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی یک‌نواخت. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.



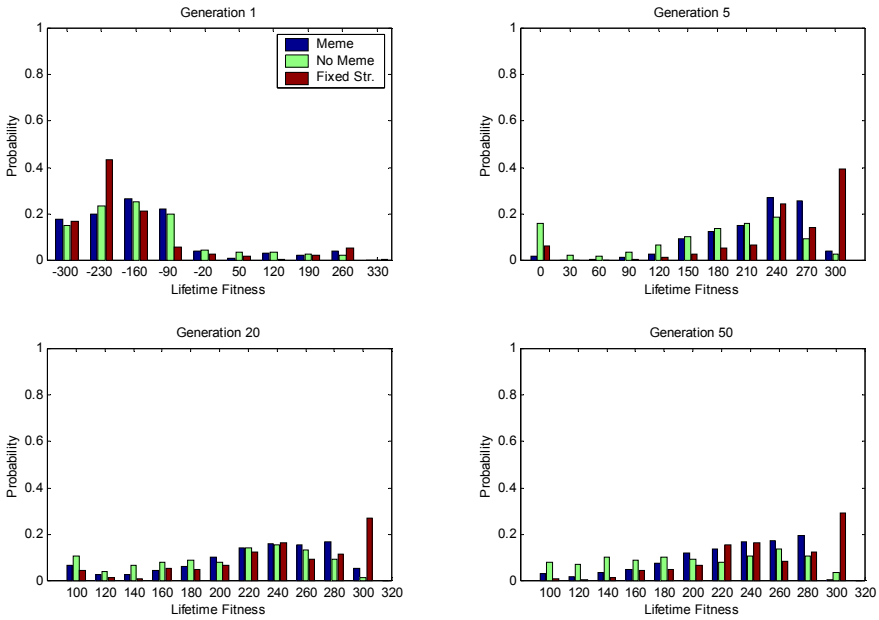
شکل ۳۴-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی احتمال تجمعی سازگاری ($P\{Fitness \leq f\}$) عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *یک‌نواخت*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (سیاه)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (آبی) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (بنفش) انجام شده است. خطوط نقطه‌چین سازگاری‌ی کلِ دوره‌ی زندگی‌ی عامل را نشان می‌دهند. هر چه توزیع بیش‌تر به سمت راست گرایش داشته باشد، شانس ایجاد عامل‌های خیلی خوب بیش‌تر است.



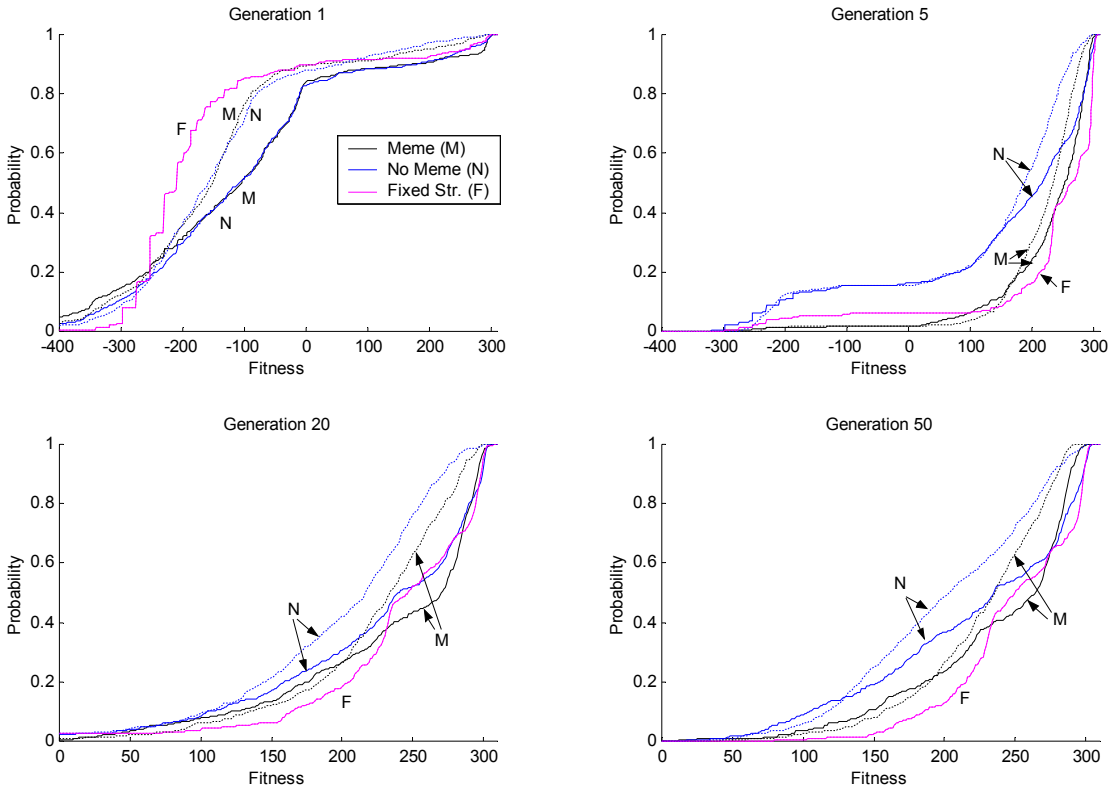
شکل ۳۵-۶. (مساله‌ی بلند کردن جسم) متوسط سازگاری پنج اپیزود نهایی و متوسط سازگاری کلِ دوره‌ی زندگی عامل در طراحی‌های مختلفی که از اشتراک سازگاری *ارزش‌محور* استفاده می‌کنند: ۱) تکامل رفتارها و یادگیری ساختار بدون کمک مم (آبی)، ۲) تکامل رفتارها و یادگیری ساختار با کمک مم (سیاه)، ۳) تکامل رفتارها و ساختار طراحی‌شده (بنفش)، ۴) رفتارهای طراحی‌شده و یادگیری ساختار (سبز) و ۵) رفتارها و ساختار طراحی‌شده (قرمز). خطوط پررنگ بیان‌گر متوسط سازگاری در پنج اپیزود انتهایی زندگی‌ی عامل هستند و خطوط نقطه‌چین سازگاری‌ی کلِ زندگی عامل را نشان می‌دهند. با وجود آن‌که عمل‌کرد آخرین اپیزودهای همه‌ی عامل‌ها تقریبا یک‌سان است اما عمل‌کرد کل دوره‌ی زندگی‌ی عامل‌هایی که از اطلاعات مم استفاده می‌کنند بالاتر است.



شکل ۳۶-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.



شکل ۳۷-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی چگالی احتمال سازگاری کلِ دوره‌ی زندگی عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (آبی تیره)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (سبز) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (قرمز) انجام شده است.



شکل ۳۸-۶. (مساله‌ی بلندکردن جسم) مقایسه‌ی احتمال تجمعی سازگاری ($P\{Fitness \leq f\}$) عامل‌های ایجاد شده با روش تکاملی با اشتراک سازگاری‌ی *ارزش‌محور*. مقایسه بین حالت‌هایی که عامل‌ها از اطلاعات مم استفاده کرده‌اند (سیاه)، عامل‌هایی که از ساختاری تصادفی شروع کرده‌اند (آبی) و عامل‌هایی که از ساختار طراحی‌شده استفاده کرده‌اند (بنفش) انجام شده است. خطوط نقطه‌چین سازگاری‌ی کلِ دوره‌ی زندگی‌ی عامل را نشان می‌دهند. هر چه توزیع بیش‌تر به سمت راست گرایش داشته باشد، شانس ایجاد عامل‌های خیلی خوب بیش‌تر است.

	Abstract Problem *(Structure Learning)*	**Abstract Problem** *(Behavior Learning)*	**Object Lifting** *(Structure Learning)*	**Object Lifting** *(Behavior(/Structure) Learning)*
Problem Space	50x50 problem space and two actions	50x50 problem space and four actions	Multi-robot object lifting	Multi-robot object lifting
Solution Space	Four behaviors $B_i(i=1,2,3,4)$ (see Fig. 3)	Four behaviors each of $B_i(i=1,2,3,4)$ and a_i them consists of NA	Five behaviors, i.e. *Push more*, *Don't go fast*, *Stop at goal*, *Hurry up*, and *Slow down*.	Five naïve behaviors' state and action space without any specified mapping.
Learning Rates	$\alpha_{episode} = \alpha_0(0.999)^{episode}$ $\alpha_0 = 0.05$ (ZO) $\alpha_0 = 0.1$ (FO)	$\alpha^{beh}_{episode} = \alpha^{str}_{episode} = \frac{0.2}{1+\mathrm{mod}(episode,1000)}$ ε-greedy action selection with $\varepsilon_{episode} = \frac{2}{\ln(episode+1)}$	$\alpha^{str}_{episode} = 0.1(0.99)^{episode}$	$\alpha^{beh}_{episode} = 0.1$ $\alpha^{str}_{episode} = 0.06(0.99)^{episode}$ [15] $\varepsilon_{epssode} = \frac{0.5}{\ln(episode+1)}$
Experimentation Condition	500 episodes trail 50 different trails	10000 episodes trial 100 different trials	150 time steps lift-up $\Delta T = 0.005$ 50 episodes trial 50 different trials $z^{1,2}(1) \in U(1,2)$ [16] $z^{3}(1) \in U(2,3)$	300 time steps lift-up $\Delta T = 0.005$ 50 episodes learning trial (exploration is on) 50 episodes testing trial (exploration is off) 100 different trials $z^{1,2,3}(1) \in U(2,3)$ Structure is updated every three (one) episodes for concurrent beh./str. (str.) learning
Performance Measures	Received reinforcement signal (MA[17] filtered with window size 10)	1-Received reinforcement signal (MA filtered with window size 100) 2-Probability of Error (calculated every 1000 episodes)	1-Received reinforcement signal 2-Behavioral performance	1-Received reinforcement signal 2-Behavioral performance

جدول ۱-۶. شرایط آزمایش برای آزمون‌های یادگیری

[15] Note that $\alpha^{str}_{episode}$ is not used in the only behavior learning case, and $\alpha^{beh}_{episode}$ and $\varepsilon_{episode}$ are not used in the comparison of structure learning alone case. $\varepsilon_{episode}$ is set zero in episodes 51-100 (testing episodes).

[16] $U(a,b)$ is a uniform random distribution between a and b.

[17] Moving Average

	Abstract Problem	**Object Lifting**
Problem Space	5x5 problem space and 7 actions	Multi-robot object lifting
Solution Space	Seven behaviors each can produce has B_i one type of action (behavior) $A_i = \{a_i, NA\}$	Five naïve behaviors' state and action space without any specific mapping (*Push more*, *Don't go fast*, *Stop at goal*, *Hurry up*, and *Slow down*.)
Evolution Parameters	Generations = 200, Population size = 30, No. of individual evaluations in each gen. = 300, Tournament selection (competition between 3 individuals), the best individual of each genetic pool goes directly to the $p_c = 0.5$ next generation, $p_m^{hard} = \begin{cases} \frac{0.01}{\log_2(gen+1)} & \text{gen} \le 179 \\ 0 & 180 \le \text{gen} \le 200 \end{cases}$ $p_m^{soft} = 2p_m^{hard}$	Generations = 50, Population size = 20, No. of individual evaluations in each gene.= 200, Tournament selection (competition between 3 individuals), the best individual of each genetic pool goes $p_c = 0.5$ directly to the next generation, $p_m^{hard} = \begin{cases} \frac{0.01}{\log_2(gen+1)} & \text{gen} \le 40 \\ 0 & 41 \le \text{gen} \le 50 \end{cases}$ $p_m^{soft} = 2p_m^{hard}$
Learning Parameters	ZO Structure Learning $\alpha_{episode} = 1/episodes$	ZO Structure Learning $\alpha_{episode}^{str} = 0.1(0.99)^{episode}$
Memetic Parameters	Culture size = 5 $(5\text{-}12)\,\alpha_{T_i} = 0.3$ Meme selection: scaled roulette wheel (p=0.9)/random (p=0.1)	Culture size = 5 $(5\text{-}12)\,\alpha_{T_i} = 0.3$ Meme selection: scaled roulette wheel (p=0.9)/random (p=0.1)
Experimentation Condition	100 episodes trial 3 runs Fitness is the mean value of reinforcement signal during the agent's lifetime	$\Delta T = 0.005$ 300 time step lift-up, 25 (5) episodes trial for with structure learning (fixed structure) case 3 runs for learning cases – 2 runs for fix structure case - 1 run for pre-defined behaviors (no evolution) $z^{1,2,3}(1) \in U(2,3)$ Fitness (Lift time fitness) is calculated by averaging the last 5 episodes (all episodes) - Fitness is used for evolutionary mechanism (and not Life time fitness)
Performance Measures	Received reinforcement signal (MA filtered with window size 10)	1-Received reinforcement signal 2-Behavioral performance

جدول ۲-۶. شرایط آزمایش برای آزمون‌های تکامل/الگوریتم ممتیک

۷) تحلیل احتمالاتی معماری PPSSA

۷-۱) مقدمه

در فصل‌های پیش به چگونگی‌ی یادگیری و تکامل در معماری‌ی رفتارگرای سلسله مراتبی پرداختم. چگونگی‌ی تخصیص امتیاز بین رفتارهای مختلف و روش به روز رسانی‌ی دانش عامل نسبت به محیط اطراف‌اش توضیح داده شد. این‌که عامل تا چه میزاد یاد بگیرد و به چه بازده‌ای برسد البته موضوع‌ای است مجزا که به مساله‌ی مورد نظر، چگونگی‌ی انتخاب زیرفضای حالت هر رفتار، سیگنال تقویت طراحی شده و ... بستگی دارد. با این همه می‌توان با فرض گرفتن چند ویژگی، رفتار معماری را تا حدی تحلیل کرد. در این فصل سعی در تحلیل احتمالاتی‌ی معماری‌ی رفتارگرای PPSSA خواهم کرد. در این راستا ابتدا احتمال کنترل‌کننده‌بودن هر کدام از رفتارهای معماری محاسبه می‌شود و سپس تخمین‌ای از زمان لازم برای یادگیری رفتارها به دست می‌آید. نتایج جالب توجه‌ای در این میان حاصل می‌شود که می‌تواند طراح را در انتخاب به‌تر رفتارها و معماری یاری کند.

مانند بخش‌های پیش، معماری‌ی PPSSA را برمی‌گزینم که هر رفتار B_i در زیرمجموعه‌ی $S_i \subset S$ تحریک می‌شود ومی‌اُواند عمل‌ای از مجموعه‌ی A_i انتخاب کند. تعاریف تحریک شدن، فعال شدن و کنترل‌کننده بودن مانند پیش است.

برای خوانایی فرض می‌کنم که هر B_i در لایه‌ی iام (L_i) قرار گرفته است – نام‌گذاری‌ی رفتارها با نام‌گذاری‌ی لایه‌ها یک‌سان است. مشخص است که این فرض محدود کننده نیست و تنها برای جلوگیری از پیچیده شدن روابط در نظر گرفته شده است.

۷-۲) احتمال کنترل‌کننده بودن رفتارها

در این بخش به محاسبه‌ی احتمال کنترل‌کننده بودن رفتارها می‌پردازم. ابتدا از حالت‌های ساده شروع می‌کنم و به تدریج بر پیچیدگی و دقت تحلیل می‌افزایم.

محاسبه‌ی احتمال این‌که عمل a_i رفتار B_i –که در لایه‌ی L_i قرار دارد– کنترل عامل را در اختیار بگیرد مورد نظر است. با انجام این‌کار می‌توان فهمید که هر کدام از رفتارها با چه احتمالی در تصمیم‌گیری‌ی سیستم عامل نقش دارند. با استفاده از این دانش می‌توان زمان لازم برای یادگیری‌ی هر کدام از رفتارها را تخمین زد. به این شکل که هر چه یک رفتار بیش‌تر کنترل عامل را در اختیار بگیرد و تجربه‌ی بیش‌تری از تعامل با محیط کسف کند، شانس بیش‌تری برای یادگیری پیدا می‌کند و احتمالا سرعت هم‌گرایی به پاسخ مطلوب‌اش بیش‌تر می‌شود.

ایده‌ی محاسبه‌ی احتمال در اختیار گرفتن کنترل عامل توسط یک رفتار در لایه‌ای خاص (مثلا لایه‌ی nli) که توسط $P\{a = B_i(s) \text{ and } B_i \text{ is controlling}\}$ نمایش‌اش می‌دهم بدین صورت است: این احتمال برابر است با احتمال این‌که هیچ‌کدام از لایه‌های بالایی لایه‌ی کنترلی نباشد و هم‌چنین این لایه می‌بایست لایه‌ی کنترلی باشد و عمل‌ای که انتخاب می‌کند نیز این عمل مورد نظر باشد.
شرط کنترل‌کننده نبودن یک لایه این است که فعال نشود. فعال نشدن معادل یکی از دو روی‌داد زیر است:

۱. رفتار آن لایه تحریک نشود.
۲. رفتار آن لایه تحریک بشود ولی عمل NA را انتخاب کند.

اگر فرض کنیم معماری دارای m لایه است که پایین‌ترین لایه با شماره‌ی ۱ و بالاترین آن با شماره‌ی m مشخص شده باشند، آن‌گاه می‌توان نوشت:

$$P\{a = B_i(s) \text{ and } B_i \text{ is controlling}\} = \\ \text{P}\{\text{upper layers are not active in } s\} \times P\{B_i(s) \text{is excited}\} \times P\{a = B_i(s) \mid B_i(s) \text{is excited}\} \quad (۷-۱)$$

که با بسط جمله‌ی اول داریم:

$$P\{a = B_i(s) \text{ and } B_i \text{ is controlling}\} = \left(\prod_{j=i+1}^{m} P\{B_i(s) \text{ is not active}\}\right) \times P_i^e(s) \times P\{a = B_i(s) \mid B_i(s) \text{ is excited}\} \quad (۷-۲)$$

که در آن $P_i^e(s)$ احتمال تحریک شدن رفتار B_i است.

برای محاسبه‌ی این عبارات می‌بایست $P_i^e(s)$ و احتمال فعال شدن هر لایه را بدانیم. دانستن اولی بستگی به مساله دارد. محاسبه‌ی دومی نیز نه تنها به احتمال تحریک شدن بستگی دارد، بلکه به نوع انتخاب عمل و تجربه‌ی عامل در محیط نیز بستگی دارد. با این همه با فرض‌هایی سعی در تحلیل احتمال‌های بالا می‌کنیم.

چگونگی‌ی ارتباط احتمال فعال شدن (یا نشدن) هر رفتار را با توجه به نوع روش انتخاب عمل و احتمال تحریک‌شدن‌اش بدین‌گونه می‌توان نوشت:

- Random Action Selection (or Uniform Q-Table)

  $P\{B_i(s) \text{ is not active}\} = P\{B_i(s) \text{ is not excited}\} + P\{B_i \text{ is excited and selects } NA\} =$

  - $\left(1 - P_i^e(s)\right) + P_i^e(s)\frac{1}{|A_i| + 1}$

(۷-۳)

که جمله‌ی اول بیان‌گر احتمال تحریک نشدن (و در نتیجه، فعال نشدن) است و جمله‌ی دوم بیان‌گر احتمال تحریک شدن و انتخاب NA. به همین ترتیب خواهیم داشت

- ε-greedy Action Selection

- $P\{B_i(s) \text{ is not active}\} = (1 - P_i^e(s)) + P_i^e(s)\begin{bmatrix} P\{Q_i(s_i,NA) > Q_i(s_i,a');\ \forall a' \neq NA\}(1-\varepsilon) \\ + \frac{1}{|A_i|+1}\varepsilon \end{bmatrix}$

(۷-۴)

- Boltzman Action Selection
  - $P\{B_i(s) \text{ is not active}\} = (1 - P_i^e(s)) + P_i^e(s)\frac{\exp\left(\frac{Q_i(s_i,NA)}{T}\right)}{\sum_{a' \in A_i'} \exp\left(\frac{Q_i(s_i,a')}{T}\right)}$ (۷-۵)

مشاهده می‌شود که برای محاسبه‌ی این احتمال هم نیاز به احتمال تحریک شدن وجود دارد و هم اطلاعات‌ای راجع به دانش عامل ذخیره شده در Q-Table آن.

ابتدا در نظر بگیرید که $S_i = S$ باشد و در نتیجه همه‌ی رفتارها در همه حالت تحریک‌شده‌اند. در ضمن فرض کنید که بعد فضای عمل‌های هر کدام نیز A_i باشد. آن‌گاه احتمال فعال نشدن برابر با احتمال انتخاب NA است. اگر نوع انتخاب عمل را تصادفی در نظر بگیریم، می‌توان نوشت:

$$P\{B_i \text{ is not active}\} = P\{B_i \text{ is not excited}\} + P\{B_i \text{ is excited and selects } NA\} = (1-1) + 1\frac{1}{|A_i|+1} = \frac{1}{|A_i|+1} \quad (۷-۶)$$

حال به عنوان نمونه اگر فرض کنیم که فضای عمل‌ها بین رفتارها افزار شده باشد و هر کدام از رفتارها مسوول انتخاب تنها یکی از رفتارها هستند (مثلا یکی رفتار بالا بردن جسم و یکی متوقف شدن و دیگری پایین آوردن) آن‌گاه خواهیم داشت:

$$P\{B_i \text{ is not active}\} = \frac{1}{1+1} = \frac{1}{2} \quad (۷-۷)$$

با استفاده از (۷-۲) می‌توان نوشت:

$$P\{a_{[\in A_i]} = B_i(s) \text{ and } B_i \text{ is controlling}\} = \left(\prod_{j=i+1}^{m=|A|}\frac{1}{2}\right)\times 1 \times \frac{1}{2} = \left(\frac{1}{2}\right)^{|A|-i+1} \quad (۷-۸)$$

البته توجه کنید که a مشخص شده در عبارت بالا، هر عمل‌ای نمی‌تواند باشد و حتما می‌بایست متعلق به مجموعه‌ی تک عضوی‌ی A_i باشد. در مقایسه با این عبارت، می‌توان احتمال انتخاب یک عمل خاص در flat Q-Table با انتخاب تصادفی را آورد

$$P\{a = B(s) \mid \text{flat}\} = \frac{1}{|A|} \quad (۷-۹)$$

این‌گونه تقسیم‌بندی‌ی فضا —که همه‌ی رفتارها با هم تحریک می‌شوند– عملا آن چیزی نیست که در یک PPSSA رخ می‌دهد. رفتارها تخصصی هستند و سنسورهای متفاوتی را درک می‌کنند. به همین دلیل، در هر لحظه تنها بخش کوچکی از کل رفتارها فعال می‌شوند. اگر بخواهیم این نوع نگاه به مساله را به بیان‌ای ریاضی بنویسیم، می‌توان احتمال تحریک هر رفتار را بدین‌گونه نوشت:

$$P_i^e(s) = \frac{l}{m} \quad (۷-۱۰)$$

که بدان معناست که در هر لحظه، به طور میانگین l رفتار از میان m رفتار تحریک شده‌اند. با چنین فرضی و با در نظر گرفتن انتخاب عمل تصادفی می‌توان نوشت:

$$P\{a = B_i(s) \text{and } B_i \text{ is controlling}\} = \left[\prod_{j=i+1}^{m}\left((1-\frac{l}{m}) + \frac{l}{m}\frac{1}{|A_i|+1}\right)\right]\times\frac{l}{m}\times\frac{1}{|A_i|+1} \quad (۷-۱۱)$$

که اگر مانند قبل عمل‌ها را بین رفتارها افراز تک‌تک کرده باشیم —یعنی $m = |A|$– خواهیم داشت

$$\left.\begin{array}{l}|A_i|=1\\ \bigcup_{\mathrm{i}=1}^{|\mathrm{A}|} A_i \cap A_j = \varnothing;\ \forall \mathrm{i},\mathrm{j} \wedge i \neq j\end{array}\right\} \Rightarrow P\{a = B_i(s) \text{ and } B_i \text{ is controlling}\} = \frac{l}{2m}\left(1-\frac{l}{2m}\right)^{m-i}$$

(۷-۱۲)

که اگر در آن تعداد کمی از رفتارها به طور هم‌زمان تحریک شوند، آن‌گاه

$$\frac{l}{m} << 1 \Rightarrow P\{a = B_i(s) \text{ and } B_i(s) \text{is controlling}\} \approx \frac{l}{2m}\left(1-\frac{l(m-i)}{2m}\right) \qquad (۷-۱۳)$$

حال فرض پیشین در مورد احتمال تحریک لایه‌ها را کمی تعمیم می‌بخشم: احتمال تحریک لایه‌ها را برابر نمی‌گیرم. می‌توان گفت که در SSA، فرض بر این است که لایه‌های پایین‌تر لایه‌هایی هستند که در اکثر مواقع تحریک شده‌اند و کنترل عامل را بر عهده دارند و در عوض لایه‌های بالایی آن‌هایی‌اند که به ندرت —و تنها در شرایط بحرانی– فعال می‌شوند. این فرض معقول را می‌توان بدین‌گونه بیان کرد:

$$P_i^e(s) > P_{i+1}^e(s) \quad i = 1,\ldots,m-1 \qquad (۷-۱۴)$$

دو مدل ساده برای تغییرات P_i^e در نظر می‌گیریم. یکی مدل‌ایست که احتمال‌های مورد بحث به صورت هندسی از پایین به بالا کاهش یابند و دیگر مدل‌ای است که آن احتمال‌ها به طور حسابی از پایین به بالا کاهش می‌یابند. به عبارت دقیق‌تر برای مدل کاهش حسابی داریم:

$$P_{i+1}^e = \eta P_i^e \qquad (۷-۱۵)$$

و برای مدل حسابی داریم:

$$P_{i+1}^{e} = \max(P_i^e - \delta, 0) \qquad (۷-۱۶)$$

محاسبات احتمال کنترل‌کننده بودن را برای مدل کاهش هندسی انجام می‌دهم: به مانند پیش انتخاب عمل‌مان را تصادفی فرض می‌کنم. هم‌چنین $A_i = A$ در نظر می‌گیرم. خواهیم داشت:

$$P_{i+1}^{e} = \eta P_i^e \Rightarrow P_i^e = \eta^{i-1} P_1^e \qquad (۷-۱۷)$$

$$P\{a = B_i(s) \mid B_i(s) \text{ is exited}\} = \frac{1}{|A|+1} \qquad (۷-۱۸)$$

$$P\{B_i(s) \text{ is not active}\} = \left(1 - \mathrm{P_i^e}(s)\right) + \mathrm{P_i^e}(s)\frac{1}{|A_i|+1} = \left(1-\eta^{i-1}P_1^e\right) + \frac{\eta^{i-1}P_1^e}{|A|+1} = \frac{1}{|A|+1}\left[1+|A|\left(1-\eta^{i-1}P_1^e\right)\right] \qquad (۷-۱۹)$$

پس در نتیجه داریم:

$$P\{a = B_i(s) \text{and } B_i(s) \text{is controlling}\} = \frac{1}{\left(|A|+1\right)^{m-i}} \prod_{\mathrm{j=i+1}}^{\mathrm{m}} \left(1+|A|\left(1-\eta^{i-1}P_1^e\right)\right) \cdot \frac{\eta^{i-1}P_1^e}{|A|+1} \qquad (۷-۲۰)$$

از تحلیل تحلیل ریاضی‌ی فرم نزول حسابی احتمال تحریک صرف نظر می‌شود و تنها نتایج شبیه‌سازی‌شده‌ی آن آورده می‌شود.

در شبیه‌سازی‌ها دیده می‌شود که پارامترهای مختلف مانند تعداد لایه‌ها (m)، بعد فضای خروجی (A) و مخصوصا نرخ کاهش احتمال تحریک (η و ε) می‌توانند تاثیری قابل توجه بر احتمال کنترل‌کننده بودن لایه‌ها بگذارند. نتایج این شبیه‌سازی‌ها در نمودارهای بعدی می‌آید.

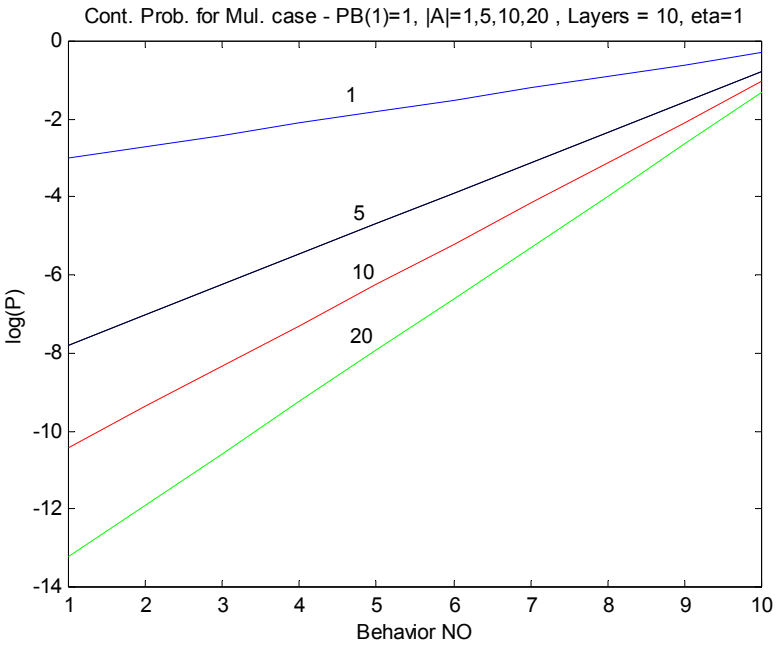
ابتدا فرض می‌کنیم هیچ کاهش احتمال تحریک‌ای وجود نداشته باشد ($\eta = 1$) و تنها بعد فضای خروجی تغییر کند. از رابطه‌ی به دست آمده مشاهده می‌شود که هر چه این بعد بزرگ‌تر باشد، احتمال کنترل‌کننده بودن برای لایه‌های پایین‌تر کم‌تر می‌شود. این اثر را در شکل ۱–۷ می‌بینید. هم‌چنین تاثیر تعداد لایه‌ها نیز در شکل ۲–۷ بررسی شده است. مشاهده می‌شود که هر چه تعداد لایه‌های بالای یک رفتار بیش‌تر باشد، احتمال کنترل‌کننده بودن آن کم‌تر می‌شود و نرخ کاهش آن نیز نمایی است. این دو نتیجه علاوه بر آن‌که به راحتی از روی رابطه‌ی قبل قابل حصول‌اند، از نظر شهودی نیز پذیرفتنی‌اند.

اینک تاثیر η را مطالعه می‌کنم. η میزان کاهش احتمال تحریک‌شدن در مدل هندسی را مشخص می‌کند و هر چقدر اندازه‌اش کم‌تر باشد، شانس تحریک لایه‌های بالایی کم‌تر می‌شود. تاثیر این پارامتر را در شکل‌های ۳–۷ و ۴–۷ می‌بینید. مشاهده می‌شود که هنگامی که η از مقدار ۱ کم‌تر می‌شود، شیب کاهش نمودار احتمال کنترل‌کننده بودن کم می‌شود و به حالت تخت نزدیک می‌شود. این به آن معنا است که احتمال کنترل‌کننده بودن همه‌ی لایه‌ها کم و بیش یک‌سان می‌شود. با کاهش بیش‌تر η، حتی جهت این شیب نیز تغییر می‌کند و احتمال کنترل‌کننده بودن لایه‌های بالا کم‌تر از لایه‌های پایین می‌شود.

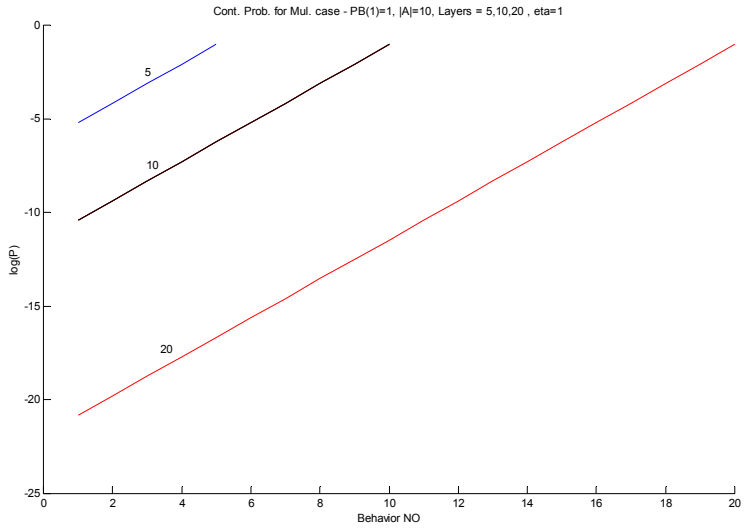
اینک مدل کاهش احتمال تحریک حسابی را مورد بررسی قرار می‌دهیم. در این مدل – همان‌طور که پیش‌تر گفته شد– احتمال تحریک پایین‌ترین لایه بیش‌تر از بقیه است و هر چه به لایه‌های بالا می‌رویم، این احتمال به اندازه‌ی ثابتی کاهش می‌یابد. مانند قبل حالت‌های مختلفی را بررسی می‌کنم. اگر $\varepsilon = 0.1$ و تعداد لایه‌ها ۱۰ باشد، آن‌گاه تاثیر تغییر بعد فضای عمل‌ها (A) در شکل ۵–۷ نمایش داده شده است. همان‌طور که انتظار می‌رفت، با افزایش بعد، احتمال کنترل‌کننده بودن لایه‌ها کاهش می‌یابد. لازم است توجه کنید که اگر $\varepsilon = 0$ می‌بود، آن‌گاه نمودارهایی دقیقا معادل شکل ۱–۷ به دست می‌آمد. در ضمن می‌بایست توجه شود که به علت غیر صفر بودن ε، منحنی‌ها در مقیاس لگاریتمی دارای انحنا هستند و کاملا با خط راست تطابق ندارند (خط راست معادل کاهش نمایی احتمال است. با توجه به شکل منحنی، وجود انحنا معادل نرخ کاهش کم‌تر یا بیش‌تر از نمایی ثابت است). با بررسی تاثیر تعداد لایه‌ها بر احتمال کنترل‌کننده بودن مشخص می‌شود که به مانند

قبل هر چه تعداد لایه‌ها بیش‌تر باشد، در مقادیر کوچک ε(مثلا $\varepsilon = 0.05$ در این مورد)، آن احتمال برای لایه‌های پایین‌تر کم‌تر می‌شود (شکل ۶–۷). البته این به شرطی است که مقدار ε به گونه‌ای تنظیم شده باشد که با پایین آمدن در لایه‌ها، احتمال نیز کم شود. دیده می‌شود که به ازای مقادیر بزرگ ε، این جهت معکوس می‌تواند باشد.

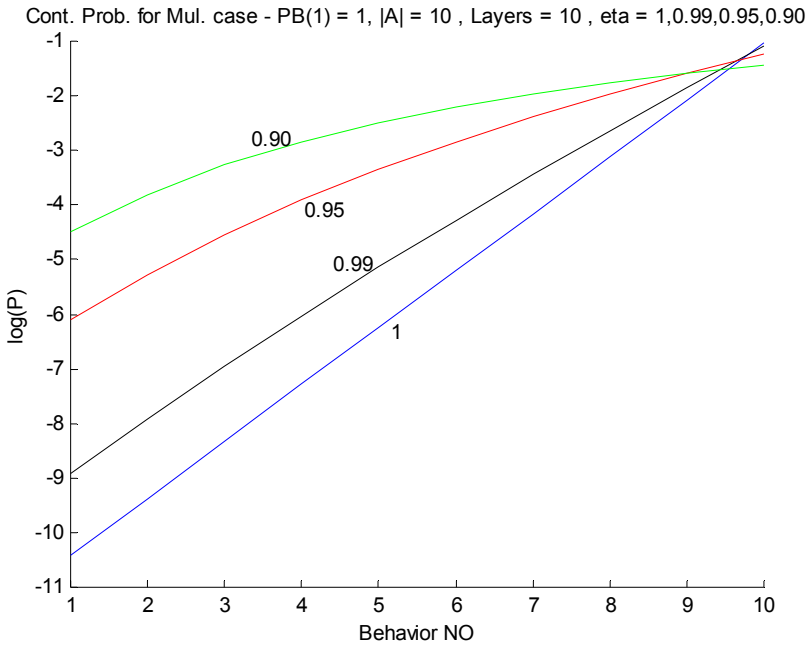
در نهایت تاثیر ε را در شکل ۷–۷ مشاهده می‌کنید. مطابق قبل می‌بینیم که با افزایش میزان کاهش، منحنی‌ها تخت‌تر می‌شوند و در نتیجه احتمال کنترل‌کننده بودن رفتار بیش‌تر می‌شود. توجه کنید که مقدار ε بیش از حدی نمی‌تواند باشد چون معادل احتمال تحریک شدن صفر برای لایه‌های بالا خواهد بود.



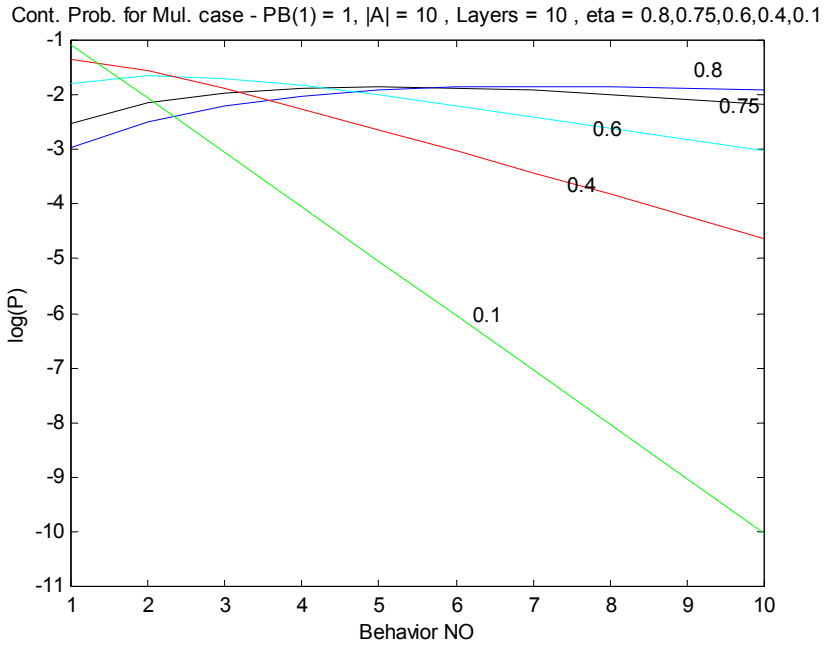
شکل ۱-۷. تاثیر بعد خروجی بر احتمال کنترل‌کننده بودن. هر چه بعد خروجی بزرگ‌تر باشد، این احتمال کم‌تر می‌شود.



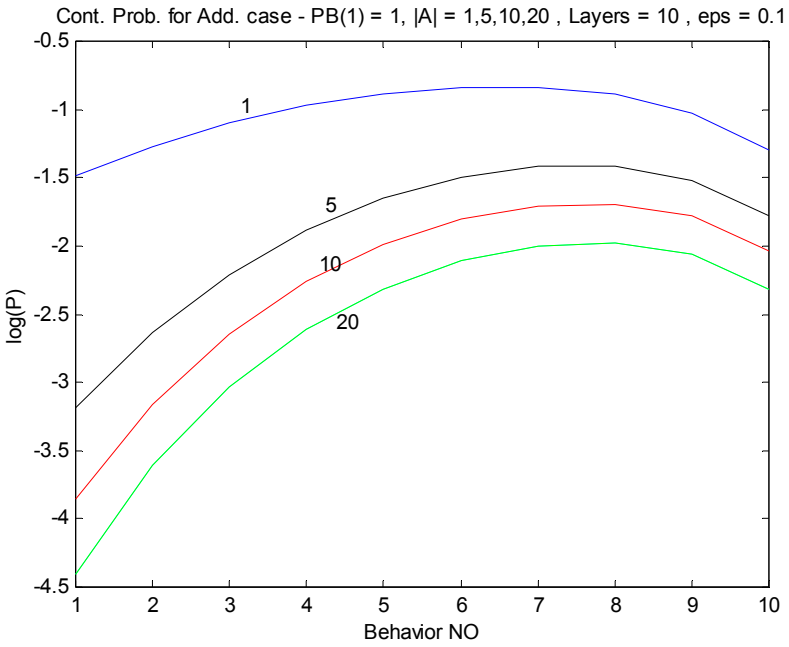
شکل ۲-۷. تاثیر تعداد رفتارها بر احتمال کنترل‌کننده بودن. هر چه تعداد رفتارها بیش‌تر باشد، رفتارهای پایین دارای احتمال کنترل‌کننده بودن کم‌تری هستند.



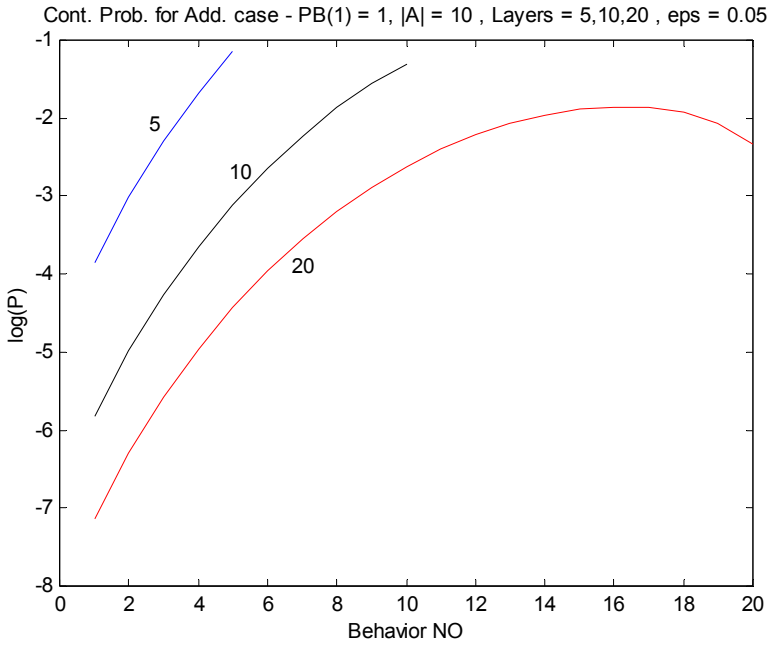
شکل ۳-۷. تاثیر η بر احتمال کنترل‌کننده بودن. در مقادیر نزدیک به ۱ η، با کاهش مقدار نمودارها تخت‌تر می‌شوند.



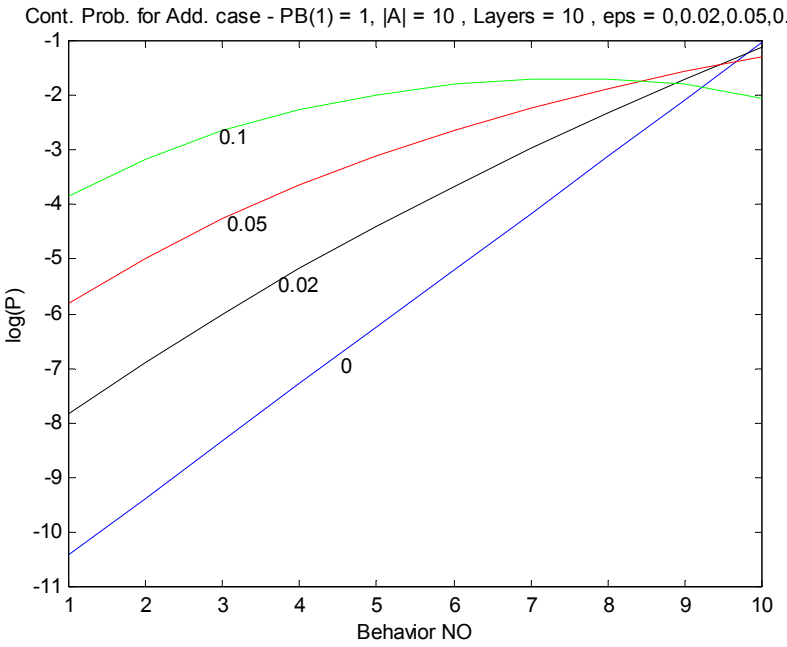
شکل ۴-۷. تاثیر η بر احتمال کنترل‌کننده بودن. مقادیر کوچک η باعث افزایش شدید احتمال کنترل‌کننده بودن لایه‌های پایین و کم شدن آن احتمال برای لایه‌های بالا می‌شود.با کاهش مقدار نمودارها تخت‌تر می‌شوند.



شکل ۵-۷. تاثیر بعد خروجی بر احتمال کنترل‌کننده بودن در مدل کاهش حسابی. هر چقدر بعد خروجی بزرگ‌تر باشد، این احتمال کم‌تر می‌شود.



شکل ۶-۷. تاثیر تعداد رفتارها بر احتمال کنترل‌کننده بودن. هر چه تعداد رفتارها بیش‌تر باشد، رفتارهای پایین دارای احتمال کنترل‌کننده بودن کم‌تری هستند.



شکل ۷-۷. تاثیر ε بر احتمال کنترل‌کننده بودن. با افزایش ε، منحنی‌ها تخت‌تر می‌شوند.

۷-۳) تحلیل‌ای بر سرعت یادگیری در معماری PPSSA

هدف از تحلیل‌های احتمالاتی‌ی PPSSA بخش پیش، درک بیش‌تر و عمیق‌تر از نحوه‌ی کارکرد عامل رفتارگرای دارای معماری PPSSA در محیط است. در قسمت قبل احتمال کنترل‌کننده بودن هر کدام از رفتارها در معماری را برای چند حالت مختلف به دست آوردم. در این بخش با استفاده از آن اطلاعات، تخمین‌ای از زمان لازم برای یادگیری‌ی هر کدام از آن رفتارها را محاسبه می‌کنم. برای مشخص شدن نحوه‌ی کار فرض کنید که معماری‌ی با ترتیب رفتارهای ثابت‌ای داریم که قرار است با استفاده از روش یادگیری تقویتی —همان‌طور که در فصل‌های پیش توضیح دادم– نحوه‌ی عمل‌کرد بهینه‌ی هر کدام از رفتارها را بیابد (مساله‌ی یادگیری رفتار). به طور تقریبی می‌توان گفت که هر چه یک رفتار (دقیق‌تر: حالت–عمل یک رفتار) بیش‌تر کنترل‌کننده شود و سیگنال تقویت دریافت کند آن‌گاه به احتمال زیاد تخمین دقیق‌تری نسبت به ارزشِ خود خواهد داشت و می‌توان ادعا کرد که آن رفتار بیش‌تر از رفتارهایی با تعداد تجربه‌ی کم‌تر یاد گرفته است. در این بخش به عنوان معیاری برای یادگیری تعداد تجربه‌های کنترل‌کنندگی (r) مورد نظر است. در زیربخش‌های بعدی توزیع احتمال تعداد تجربه‌های کنترل‌کنندگی را با تعمیم توزیع احتمال دو جمله‌ای منفی به دست می‌آورم.

۷-۳-۱) توزیع دو جمله‌ای منفی

توزیع دو جمله‌ای منفی، بیان‌گر تعداد آزمایش لازم برای r بار رخ دادن پیش‌آمدی X با احتمال p است. چگالی احتمال، متوسط و واریانس آن بدین‌گونه است:

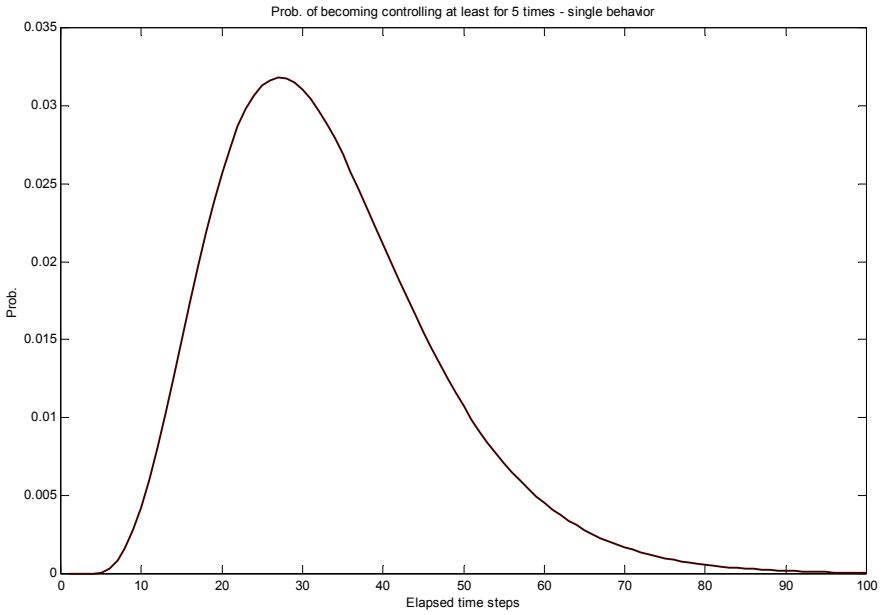
$$P_X(i)=\begin{cases}\binom{i-1}{r-1}p^r(1-p)^{i-r} & i=r,r+1,...\\ 0 & \text{otherwise}\end{cases} \qquad (۷-۲۱)$$

$$m_X=\frac{r}{p} \qquad (۷-۲۲)$$

$$\sigma_X^2=\frac{r(1-p)}{p^2} \qquad (۷-۲۳)$$

فرض کنید معماری‌ای تک-رفتاری داریم و احتمال کنترل‌کننده بودن آن تک-رفتار، P_{B_1} باشد. حال برای محاسبه‌ی چگالی‌ی احتمال تعداد آزمایش لازم برای r بار کنترل‌کننده بودن آن، می‌توان از رابطه‌ی چگالی احتمال توزیع دو جمله‌ای منفی استفاده کرد. در نتیجه می‌توان دانست به طور احتمالی چه میزان تجربه برای یادگیری‌ی این رفتار لازم است. به عنوان نمونه در شکل ۷-۸، این چگالی‌ی احتمال برای ۵ بار کنترل‌کننده بودن تک-رفتاری با احتمال کنترل‌کننده بودن ۰,۱۵ نمایش داده شده است. بدیهی‌ست که برای مقادیر کم‌تر از ۵، چگالی صفر باشد. به راحتی به دست می‌آید که متوسط زمان لازم، ۳۳,۳۳ واحد زمانی است.

حال این ایده را برای یک معماری‌ی کامل گسترش می‌دهم. در این‌جا ما به جای یک احتمال، مجموعه‌ای از احتمال‌های کنترل‌کننده بودن (P_{B_i}) را داریم. فرآیندهای این احتمال‌ها –احتمال کنترل‌کننده بودن یک رفتار- مستقل نیستند و رخداد یکی‌شان به دیگری وابسته است. البته مشخص است که در هر لحظه فقط یکی از این فرآیندها (کنترل‌کنندگی‌ی یک رفتار) رخ می‌دهد. برای به دست آوردن توزیع احتمال این‌که همه‌ی آن رفتارها به میزان r بار کنترل‌کننده باشند می‌توان ایده‌ی توزیع دوجمله‌ای منفی را گسترش داد. اما پیش از انجام آن کار، تقریبی حدودی از این زمان به دست می‌آوریم.



شکل ۸-۷. چگالی احتمال تعداد تلاش لازم برای ۵ بار کنترل‌کننده بودن رفتاری با احتمال کنترل‌کننده بودن ۰٫۱۵ در هر واحد زمانی.

فرض کنید که $X_1, X_2, ..., X_m$ فرآیند r بار کنترل‌کننده بودن هر کدام از رفتارهای $B_1, B_2, ..., B_m$ معماری‌اند در صورتی که فرض کنیم آن‌ها درون معماری نیستند و فقط احتمال کنترل‌کننده بودن‌شان جوری تنظیم شده است که P_{B_i} باشد[1]. در نتیجه رخداد هر کدام‌شان مستقل از بقیه‌اند – زیرا فرض شده بقیه‌ای وجود ندارد. پس می‌توان گفت هر کدام از این‌ها دارای توزیع دو جمله‌ای منفی با پارامترهای r و P_{B_i} هستند. توجه کنید که در این آزمایش شرط مستقل بودن به صورت اجباری لحاظ شده است و در نتیجه چنین فرآیندی معادل فرآیند اصلی –که رفتارها در معماری قرار دارند– نیست.

اگر توزیع زمان لازم برای r بار کنترل‌کننده بودن در معماری‌ی واقعی را با $Y_1, Y_2, ..., Y_m$ نشان دهیم، خواهیم داشت:

$$m_{Y_i} \geq m_{X_i} \quad \mathrm{i} = 1,..,\mathrm{m} \qquad (۷-۲۴)$$

[1] تعبیر دیگری از این تعریف فرآیند به زودی خواهد آمد.

دلیل این موضوع این است: هنگامی که رفتار B_i در معماری قرار ندارد، به احتمال P_{B_i} فعال می‌شود و به شمارنده‌ی تعداد فعالیت‌های‌اش یکی اضافه شود. این احتمال در هر کدام از واحدهای زمانی قابل اعمال است و انتخاب آزادانه است. اما وقتی این رفتار در معماری قرار گرفت، مسلم است که تعداد واحدهای زمانی‌ای که در اختیار دارد بیش‌تر نیست – چون بعضی از آن واحدهای زمانی به دیگر رفتارها اختصاص داده شده است. پس امید ریاضی زمان لازم برای r بار تکرار حداقل به اندازه‌ی امید ریاضی حالت اول است. به عبارت دیگر این قید برای رفتار درون معماری وجود دارد که در صورتی که یکی از رفتارهای دیگر کنترل‌کننده باشد، آن رفتار دیگر کنترل‌کننده نیست.

اگر متوسط زمان لازم را با $\overline{T}$ نشان دهیم، رابطه‌های زیر صادق‌اند:

$$\overline{T} \geq \max_i m_{X_i} \geq \min_i m_{X_i} \qquad (۷-۲۵)$$

فرآیند دیگری را در نظر می‌گیریم: فرض کنید که اجازه‌ی کنترل‌کننده بودن را به این نحو تقسیم کرده‌ایم که ابتدا می‌گذاریم B_1، r بار کنترل‌کننده باشد. یعنی در این حالت اگر رفتار دیگری کنترل‌کننده بود، آن را به عنوان تجربه‌ای برای‌اش در نظر نمی‌گیریم و تنها تجربه‌های B_1 را می‌شماریم. این نوع اجازه‌ی فعالیت دادن، توزیع احتمال کنترل‌کننده بودن B_1 را عوض نمی‌کند (B_1 هم‌چنان دارای احتمال کنترل‌کننده بودن P_{B_1} خواهد بود) ولی تجربه یادگیری آن را مستقل از بقیه می‌کند. این نوع تقسیم‌بندی، معادل فرآیند X_1 است[۲]. چنین فرآیندی به طور متوسط به m_{X_1} واحد زمانی نیاز دارد. پس از انجام چنین چیزی، همین عمل را برای B_2، B_3 و ... تا B_m انجام می‌دهیم. با این‌کار می‌توان گفت زمان به بخش‌هایی تقسیم شده است که هر بخش تنها به یکی از رفتارها تخصیص داده شده است. طبیعی است که چون تجربه‌های رفتارهای دیگر را هنگامی که در بخش

[2] تعبیر دیگر اشاره شده در قسمت پیش.

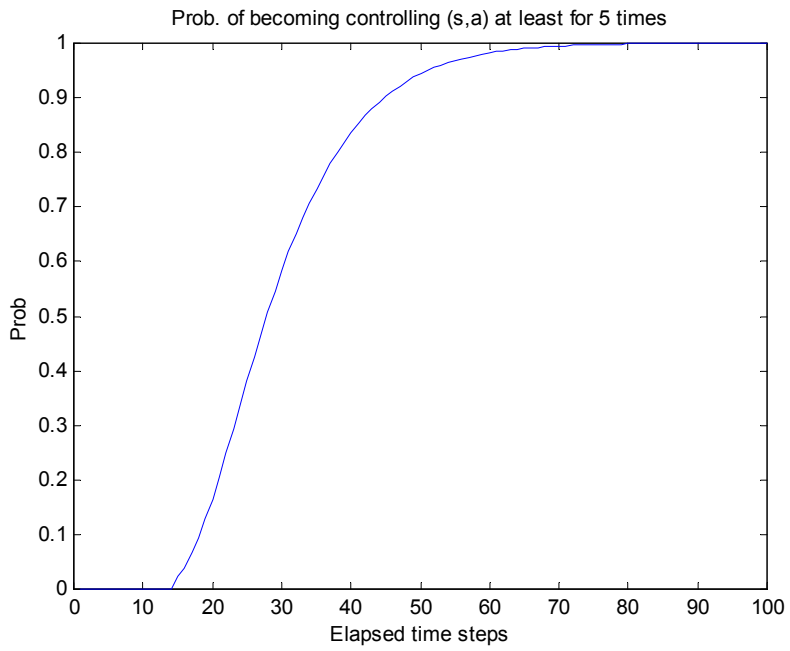
زمانی‌ی رفتار خاصی بوده‌ایم نشمارده‌ایم، پس زمان واقعی‌ی لازم (در صورتی که معماری مثل شرایط معمول اجرا می‌شد)، کم‌تر خواهد بود. پس خواهیم داشت:

$$\overline{T} \le m_{X_1} + m_{X_2} + ... + m_{X_m} \qquad (۷-۲۶)$$

بدین نحو می‌توان تخمین‌ای از میزان زمان لازم برای یادگیری به دست آورد. مثلا با توجه به روابط به دست آمده، می‌دانیم که کندترین رفتار (آن‌ای که دارای کوچک‌ترین P_{B_i} است) بخش عمده‌ی زمان لازم برای یادگیری را مشخص می‌کند و یا یادگیری کل معماری از مرتبه‌ی یادگیری‌ی هر کدام از رفتارهایی است که با احتمال مشخص شده تجربه می‌کنند. پس می‌توان نتیجه گرفت که اگر لازم است همه‌ی رفتارها یاد گرفته شوند، به‌تر است احتمال کنترل‌کننده بودن‌شان بیش از اندازه کوچک نباشد. البته SSA معمولا آن‌قدر مقاوم هست که حتی اگر یکی از رفتارها نیز کاملا یاد گرفته نشده باشد، باز هم رفتار مطلوب از کل معماری حاصل شود. اگرچه باید توجه شود که این گزاره‌ی آخر، الزاما درست نیست و گزاره‌ای تجربی است تا ریاضی. به طور خلاصه خواهیم داشت:

$$\max_i m_{X_i} \le \overline{T} \le m_{X_1} + m_{X_2} + ... + m_{X_m} \qquad (۷-۲۷)$$

پس از تحلیل کران بالا و پایین امید ریاضی زمان r تجربه داشتن برای هر رفتار، می‌توان تحلیل دقیق‌تری انجام داد و توزیع احتمال این فرآیند را به دست آورد. برای این‌کار ایده‌ی توزیع دوجمله‌ای منفی را تعمیم می‌دهم. بدین‌گونه که فرض می‌کنم همه‌ی رفتارها در معماری هستند و هر کدام احتمال کنترل‌کننده بودن مشخصی دارد و حال می‌خواهم احتمال حداقل rبار کنترل‌کننده بودن آن‌ها را محاسبه کنم. توجه کنید که به دلیل نرخ متفاوت کنترل‌کننده بودن، بعضی‌ها از رفتارها بیش‌تر کنترل‌کننده می‌شوند (و در نتیجه بارهای بیش‌تری سیگنال تقویت دریافت می‌کنند) و بعضی کم‌تر. شرط کافی برای یادگیری را این فرض می‌گیریم که همه‌ی آن‌ها حداقل rبار کنترل‌کننده بوده باشند.



شکل ۹-۷. احتمال بیش از ۵ بار کنترل‌کننده بودن برای معماری‌ی دو لایه

ابتدا از تعمیم توزیع دوجمله‌ای به بیش از یک متغیر تصادفی استفاده می‌کنم. اگر Y_1، Y_2، ... و Y_m بیان‌گر متغیرهای تصادفی با احتمال p_1، p_2، ... و p_m $(p_1 + p_2 + ... + p_m \le 1)$ باشند که در هر واحد زمانی تنها یکی از آن‌ها رخ می‌دهد، آن‌گاه احتمال رخداد r_1، r_2، ... و r_m هر کدام‌شان در r_0 بار تلاش $r_1 + r_2 + ... + r_m \le r_0$، بدین‌گونه خواهد بود[۳]:

$$P_{PD}\{Y_1 = r_1, Y_2 = r_2, ..., Y_m = r_m | r_0\} = \frac{r_0!}{\left(\prod_{i=1}^{m} r_i!\right)\left(r_0 - \sum_{i=1}^{m} r_i\right)!} \prod_{i=1}^{m} p_i^{r_i} \left(1 - \sum_{i=1}^{m} p_i\right)^{\left(r_0 - \sum_{i=1}^{m} r_i\right)}$$

(۷-۲۸)

که در صورتی که شرط‌های مساله ارضا نشود (یعنی $r_1 + r_2 + ... + r_m \ge r_0$)، این احتمال صفر خواهد بود.

[3] چنین توزیع‌ای را توزیع چندجمله‌ای می‌نامیم.

خواست مورد نظر –یعنی محاسبه بیش از rبار کنترل‌کننده بودن در r_0 تلاش– بدین‌گونه خواهد بود:

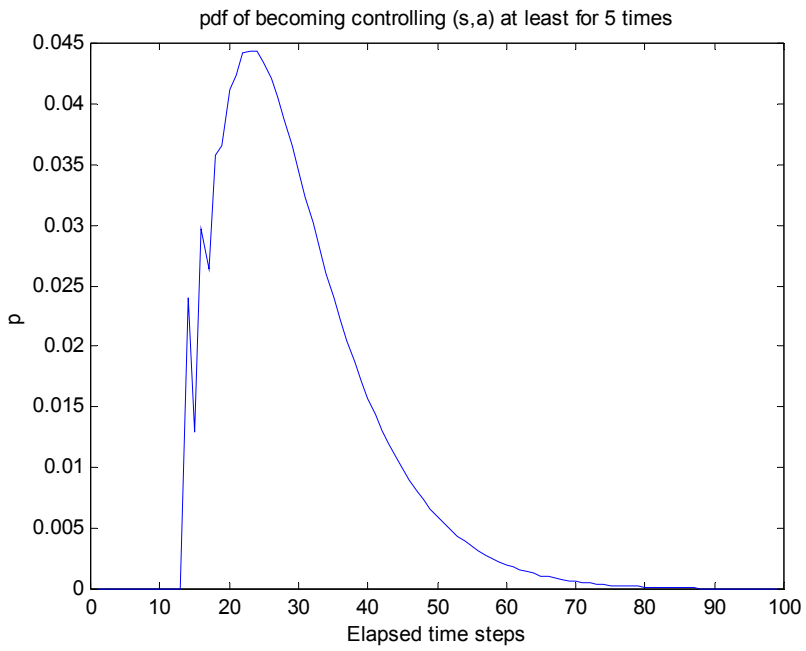
$$P_{GNB}\{(Y_1,Y_2,...,Y_m)\geq r|r_0\}=\sum_{r_1=r}^{r_0-(m-1)}\sum_{r_2=r}^{r_0-(m-1)}...\sum_{r_m=r}^{r_0-(m-1)}P_{NB}(r_1,r_2,...,r_m|r_0) \qquad (۷-۲۹)$$

به نظر نمی‌آید فرم بسته‌ای برای محاسبه‌ی این احتمال وجود داشته باشد. برای چنین محاسبه‌ای می‌بایست از روش‌های عددی استفاده کرد. با این وجود، استفاده از روش‌های عددی نیز بسیار ساده نیست چون با افزایش تعداد متغیرهای تصادفی، میزان محاسبات لازم (و در نتیجه زمان آن) به طور نمایی افزایش می‌یابد.

برای چنین محاسبه‌ای ابتدا از روش‌های تخمینی مانند تخمین مونت‌کارلو استفاده شد، اما پاسخ مناسب و دقیق‌ای به دست نیامد. برای همین همان عبارت بالا به طور مستقیم محاسبه شد با این تفاوت که به جای یک پله بالا رفتن هر کدام از حلقه‌های جمع ($\sum ...$)، از پله‌های چندتایی استفاده شده است. در نتیجه دقت محاسبات اندکی پایین آمده است ولی در عوض سرعت به مقدار قابل توجه‌ای بالا رفته است.

چگونگی‌ی استفاده از رابطه‌ی بالا برای حل مساله‌ی زمان لازم برای یادگیری رفتارها واضح است. فرض کنیم که می‌خواهیم احتمال این را که هر کدام از رفتارها r بار یا بیش‌تر در r_0 تلاش کنترل‌کننده شوند به دست آوریم. پس در رابطه‌های پیشین $P_i = P_{B_i} = P\{B_i \text{ is controlling}\}$ را – که در بخش‌های پیشین برای چند حالت مختلف به دست آمد– قرار می‌دهیم.

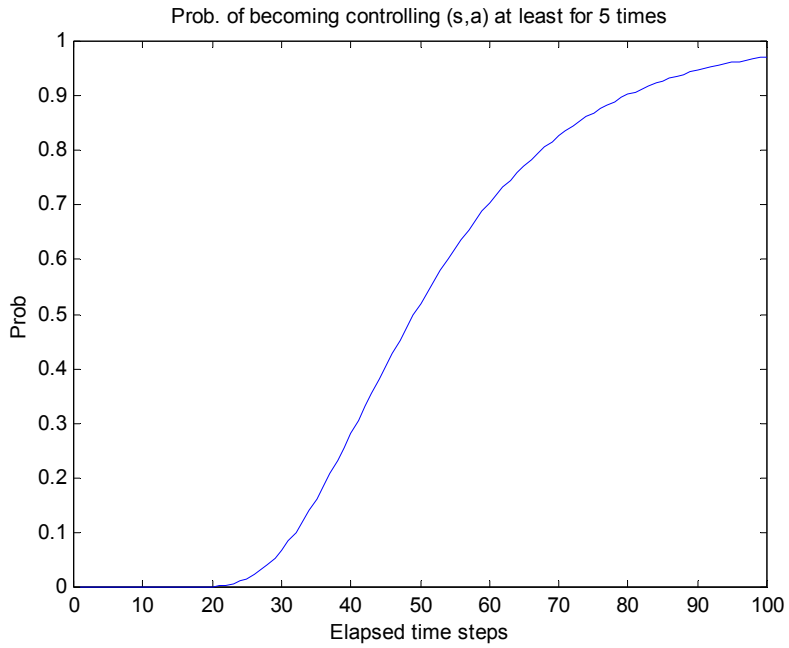
به عنوان مثال، فرض کنید که معماری‌مان دو لایه است و هر کدام از لایه‌ها دارای فضای عمل با بعد دو می‌باشد ($|A|=2$). احتمال تحریک شدن پایین‌ترین لایه را یک می‌گیریم ($P_1^e=1$) و ضریب کاهش احتمال را نیز $\eta=0.8$ (به عبارت دیگر $P_2^e=0.8$). در آن صورت، احتمال کنترل‌کننده بودن این‌گونه خواهد شد:



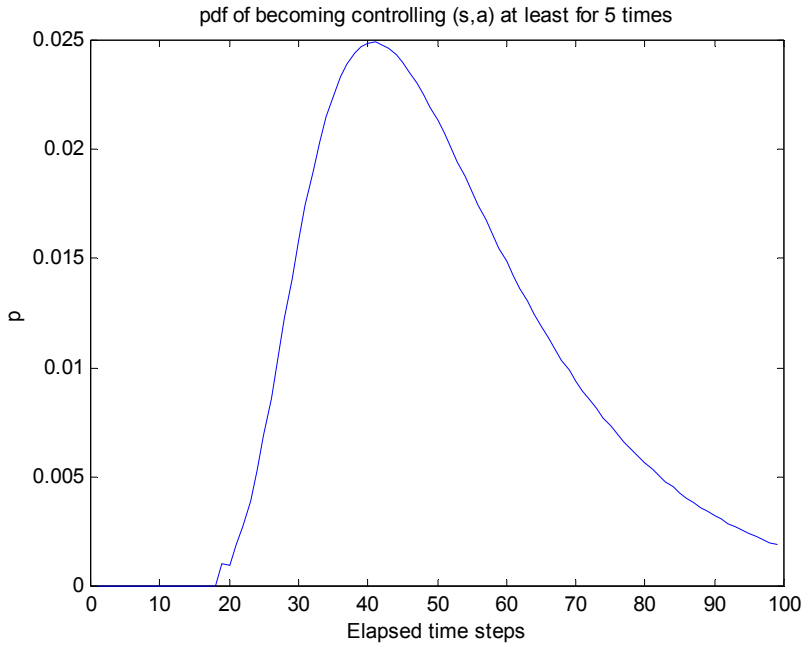
شکل ۱۰-۷. چگالی احتمال بیش از ۵ بار کنترل‌کننده بودن برای معماری‌ی دو لایه

$$P_{B_1} = 0.1556$$
$$P_{B_2} = 0.2667 \qquad (۷-۳۰)$$

$r = 5$ می‌گیریم و r_0 را تغییر می‌دهیم. نتیجه را در شکل ۹-۷ می‌بینید. اگر چگالی توزیع (بر حسب r_0) را رسم کنیم، شکل ۱۰-۷ را خواهیم داشت. قله نمودار در حوالی ۲۵-۳۰ است. با محاسبه‌ی متوسط زمان لازم برای حداقل ۵ بار کنترل‌کننده بودن، عدد 29.5 به دست می‌آید. این عدد با آن‌چه از توزیع دوجمله‌ای منفی هر کدام از رفتارها به تنهایی حاصل می‌شود (که در آن‌ها دیگر در معماری نخواهند بود)، نزدیک است، چون:



شکل ۱۱-۷. احتمال بیش از ۵ بار کنترل‌کننده بودن برای معماری‌ی سه لایه



شکل ۱۲-۷. چگالی احتمال بیش از ۵ بار کنترل‌کننده بودن برای معماری‌ی دو لایه

$$P_{B_1} = 0.1556 \Rightarrow m_{B_1} = \frac{5}{0.1556} \approx 32 \quad (۷-۳۱)$$

$$P_{B_2} = 0.2667 \Rightarrow m_{B_2} = \frac{5}{0.2667} \approx 29 \quad (۷-۳۲)$$

$$m_{T=[B_1 B_2]} = 29.5 \quad (۷-۳۳)$$

می‌بینیم که مقدار این دو سری عدد تقریبا نزدیک به هم است.

به عنوان نمونه‌ای دیگر، معماری‌ای سه لایه با همان بعد فضای عمل و $\eta = 0.8$ در نظر بگیرید. در این شرایط، احتمال کنترل‌کننده بودن بدین‌گونه است[۴]:

$$\begin{aligned} P_{B_1} &= 0.0892 \\ P_{B_2} &= 0.1529 \\ P_{B_3} &= 0.2133 \end{aligned} \quad (۷-۳۴)$$

در این حالت نیز توزیع احتمال و چگالی توزیع احتمال را در شکل‌های ۷-۱۱ و ۷-۱۲ می‌بینید. اگر متوسط زمان لازم برای حداقل ۵ بار کنترل‌کننده بودن را برای کل معماری و هم‌چنین هر کدام از رفتارها حساب کنیم، خواهیم داشت:

$$P_{B_1} = 0.0892 \Rightarrow m_{B_1} = \frac{5}{0.0892} \approx 56 \quad (۷-۳۵)$$

$$P_{B_2} = 0.1529 \Rightarrow m_{B_2} = \frac{5}{0.1529} \approx 32 \quad (۷-۳۶)$$

$$P_{B_3} = 0.2133 \Rightarrow m_{B_2} = \frac{5}{0.2133} \approx 23 \quad (۷-۳۷)$$

$$m_{T=[B_1 B_2 B_3]} \approx 49.3 \quad (۷-۳۸)$$

[4] دلیل تفاوت احتمال کنترل‌کننده بودن دو لایه‌ی پایین با حالت پیش –با وجود یک‌سان بودن پارامترها– عمل سرکوب لایه‌های بالایی‌ست.

مشاهده می‌شود که کندترین رفتار، زمان کلی را به سمت خود می‌کشاند و زمان کلی‌ی لازم –یعنی ۴۹,۳– به ۵۶ پایین‌ترین رفتار بسیار نزدیک است.

۸) تاثیر عدم قطعیت سیگنال تقویت بر تابع ارزش و سیاست عامل

۸-۱) مقدمه

بخش بزرگی از این پایان‌نامه درباره‌ی استفاده از روش‌های یادگیری تقویتی برای طراحی‌ی خودکار معماری‌ی سلسله مراتبی‌ی رفتارگرا بود. برای طراحی‌ی یک سیستم مبتنی بر یادگیری تقویتی نکاتی چند از جمله چگونگی‌ی انتخاب متغیرهای حالت، چگونگی‌ی بازنمایی دانش، ساختار سلسله مراتب و طراحی‌ی سیگنال تقویت مناسب می‌بایست مورد توجه قرار گیرد. حلِ هر کدام از این مسایل ساده نیست و پژوهش‌های بسیاری برای پاسخ دادن به آن‌ها انجام شده است و هم‌چنان ادامه دارد. در این فصل به بررسی‌ی بخشِ کوچک‌ای از یکی از این مسایل –یعنی طراحی سیگنال تقویت– می‌پردازم و تاثیر عدم قطعیت در سیگنال تقویت را بر تابع ارزش و هم‌چنین سیاست عامل به صورت کران‌هایی به دست خواهم آورد[1].

یکی از اصول فلسفی‌ی پس طراحی رفتار با استفاده از یادگیری تقویتی، مقاوم بودن پاداش و جزا نسبت به تغییرات محیطی است و در نتیجه توصیف سیگنال تقویت‌ای یک وظیفه مقاوم‌تر و قابل انتقال‌تر از توصیف سیاستِ انجام آن است. اگر دینامیک محیط یا عامل تغییر کند، سیاست از پیش مشخص شده دیگر مفید نخواهد بود اما عاملِ یادگیر در صورت دست‌رسی به سیگنال تقویتِ مناسب‌ای که بر اعمال‌اش ارزش‌گذاری کند می‌تواند هم‌چنان کارکرد خود را حفظ کند. با این وجود طراحی‌ی سیگنال تقویت‌ای که هدف طراح را برآورد کند چندان ساده نیست. جدا از مسایل‌ای که هدف بسیار مشخص است (مانند ماز که هدف آن رسیدن به یک نقطه است)، در بسیاری از مسایل دیگر طراح با روش سعی و خطا سیگنال تقویت‌ای عینی‌ای[2] را که خواست ذهنی‌ی[3] او را ارضا می‌کند می‌یابد. با این‌حال معمولا سیگنال تقویت دست‌یافته با آن *سیگنال تقویت بهینه‌ی ناشناخته* –که همان‌طور که از

[1] نتایج این فصل در [Farahmand05a] آمده است.

[2] Objective
[3] Subjective

نام‌اش برمی‌آید، طراح از شکل دقیق آن آگاه نیست- تفاوت‌هایی دارد. این تفاوت در سیگنال تقویت باعث تفاوت در تابع ارزش عامل می‌شود و در نتیجه سیاست عامل در مواجهه با محیط تغییر می‌کند. در این فصل فرض می‌کنم که میزان این خطا به صورت کران بالایی مشخص شده است و هدف پیدا کردن کران‌های برای تغییر در تابع ارزش و تغییر در سیاست است.

نتایج این فصل کاربردهای متفاوتی می‌تواند داشته باشد. اینک چند سناریوی ممکن را شرح می‌دهم:

- در [Harati02] روش‌هایی برای یادگیری تقویتی سیستم‌های چند-عامله هم‌کار پیش‌نهاد شد. نشان داده شد که با استفاده از معیاری به نام خبرگی می‌توان سرعت یادگیری را بسیاری بالا برد. اما نکته‌ی مهم این بود که در روش‌های تشخیص خبرگی امکان خطا وجود داشت و در نتیجه گاهی سیگنال تقویت به اشتباه به عامل‌های مختلف تخصیص داده می‌شد. میزان این خطا به میزان محافظه‌کار بودن طراحی‌ی ما بستگی دارد: می‌توان کاملا محافظه‌کار بود و سیگنال خطای دقیق‌تری داشت ولی در عوض سرعت یادگیری پایین‌تر باشد و یا این‌که کم‌تر محافظه‌کاری کرد و گاهی سیگنال خطای نادقیق‌ای پیش‌نهاد کرد و سرعت یادگیری را بالا برد ولی در عوض امکان رسیدن به تابع ارزش و در نتیجه سیاست اشتباه به وجود آید. با توجه به میزان اشتباه در سیاست قابل قبول و با استفاده از نتایج این فصل می‌توان میزان محافظه‌کاری‌ی روش تعیین خبرگی را مشخص کرد.
- در بسیاری از مسایل هدف نهایی خوش‌تعریف است – مثلا رسیدن به نقطه‌ای خاص در ماز، پایداری آونگ معکوس، گل زدن به حریف در فوتبال و ... . گاه به دلیل آن‌که احتمال رسیدن به حالت هدف بسیار کم است، طراح ممکن است تعدادی هدف میانی –به صورت توابع‌ای جمع شونده به سیگنال تقویت اصلی- برای مساله طراحی کند. این هدف‌های میانی به پاسخ رسیدن را راحت‌تر و سریع‌تر می‌کند. اما در عوض احتمال انتخاب هدف میانی‌ی نادرست نیز وجود دارد که ممکن است باعث پایین آمدن کیفیت کار شود. در صورتی که طراح بتواند به نوعی کرانِ خطای احتمالی در

اضافه کردن این هدف‌های میانی را به دست بیاورد می‌تواند از نتایج این فصل برای پیش‌بینی‌ی خطای تابع ارزش و سیاست عامل استفاده کند.

- نتایج این فصل علاوه بر طراحی سیگنال تقویت، در پیش‌بینی‌ی کیفیت به کارگیری‌ی یک تقریب‌زن تابع مانند شبکه‌ی عصبی در یادگیری تقویتی نیز قابل به کارگیری‌اند. اگر بدانیم که حداکثر خطای تقریب تابع ارزش چه میزان است می‌توان کران‌ای بر خطای سیاست عامل نوشت.

چند پژوهش مشابه با این نتایج این فصل وجود دارد. در [Williams94] نتایج‌ای در مورد تاثیر خطای تخمین تابع ارزش بر پاداش بازگشت‌یافته‌ی عامل آمده است. برخلاف پژوهش فعلی آن‌ها بررسی‌ی بر روی تاثیر خطای سیگنال تقویت انجام نداده‌اند و مهم‌تر از آن تاثیر این خطا بر سیاست عامل را –که معیاری رفتارگرایانه از کارکرد عامل است– نیز مطالعه نکرده‌اند ([Sing94] را نیز ببینید). در [Ng99] شرط ناوردایی[۴] سیگنال تقویت اثبات شده است: به ازای چه خانواده توابع سیگنال تقویت، سیاست عامل تغییری نمی‌کند. نتایج کارشان دستِ طراح را در ایجاد سیگنال تقویت برای شکل‌دهی[۵] و سرعت بخشیدن باز می‌کند. اما گاهی دقیق بودن و تغییر نکردن شرط سخت و محدودکننده‌ای است. ممکن است حد اندکی از خطا در سیاست قابل قبول باشد که در چارچوب معرفی شده در آن پژوهش جای نمی‌گیرد.

در این فصل ابتدا به بررسی‌ی تاثیر خطای سیگنال پاداش بر تابع ارزش می‌پردازم و به صورت ریاضی کران بالایی برای حداکثر تغییر آن تابع به دست می‌آورم و سپس تاثیر این خطا را بر رفتار عامل به صورت کران‌ای احتمالی به دست می‌آوریم و پس از آن نتایج حاصل را در مساله‌ای کلی نمایش می‌دهم.

۸-۲) تاثیر عدم قطعیت سیگنال تقویت بر توابع ارزش

در این بخش به بررسی‌ی تاثیر عدم قطعیت بر توابع $Q(s,a)$ و هم‌چنین سیاست $\pi(s)$ ناشی از عدم‌قطعیت القا شده می‌پردازیم. با دانستن تاثیر آن بر $\pi(s)$، می‌توان رفتار سیستم ایده‌آل و سیستم با عدم‌قطعیت را با هم مقایسه کرد.

[4] Invariance
[5] Shaping

فرض کنید که s بیان‌گر حالت عامل در محیطی مارکوف باشد. عامل عمل a را در محیط انجام می‌دهد و با توجه به تابع انتقال حالت محیط به حالت جدید s' می‌رود و در این گذر تقویت $R(s,a,s')$ را دریافت می‌کند. تابع ارزش بهینه‌ی حالت به صورت زیر تعریف می‌شود:

$$V^*(s) = \max_{\pi} \underset{\mathbf{s}\sim\pi}{E}\left[\sum_{k=0}^{\infty}\gamma^k r_k \middle| s_0 = s\right] \qquad (۸-۱)$$

که در آن r_k سیگنال تقویت دریافتی در kامین انتقال حالت سیستم است. امید ریاضی بر روی همه‌ی مسیرهای ممکن‌ای که از $s_0 = s$ آغاز می‌شوند و با سیاست بهینه ادامه می‌یابند گرفته می‌شود. برای سادگی‌ی فرمول‌بندی، گاهی از بیان تابع s بودن تابع ارزش صرف‌نظر می‌کنم و آن را بدین‌گونه می‌نویسم:

$$V^* = \max_{\pi} \underset{\mathbf{s}\sim\pi}{E}\left[\sum_{k=0}^{\infty}\gamma^k r_k\right] \qquad (۸-۲)$$

فرض کنید که r_k سیگنال تقویت اصلی ولی ناشناخته‌ای باشد که منجر به پاسخ بهینه‌ی مساله می‌شود (از دیدگاه طراح و نه از دیدگاه عینی که به هر حال فرمول‌بندی‌ی یادگیری تقویتی آن به پاسخ بهینه‌ی عینی می‌رسد) و r'_k نیز سیگنال تقویت خطاداری باشد که طراح آن را مشخص کرده است. فرض کنید که r'_k دارای خطایی با نرم محدود باشد، یعنی:

$$r'_k = r_k + \Delta r_k \qquad (۸-۳)$$

$$\left\|\Delta r_k\right\|_{\infty} < \varepsilon \qquad (۸-۴)$$

اندازه‌ی خطا بیان‌گر عدم‌قطعیت طراح نسبت به درستی‌ی سیگنال طراحی‌شده‌اش است.

اینک می‌توان تابع ارزش سیگنال تقویت جدید را بدین‌گونه نوشت:

$$V'^{*}(s) = \max_{\pi} \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\left(r_k + \Delta r_k\right)\right] < \max_{\pi} \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\left(r_k + \left\|\Delta r_k\right\|_{\infty}\right)\right] = \\ V^{*} + \varepsilon \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\right] = V^{*} + \frac{\varepsilon}{1-\gamma} \qquad (۸-۵)$$

که در آن $V'(s)$ تابع ارزش عامل با سیگنال تقویت خطادار است. به همین صورت می‌توان نوشت:

$$V'^{*}(s) = \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\left(r_k + \Delta r_k\right)\right] > \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\left(r_k - \left\|\Delta r_k\right\|_{\infty}\right)\right] = V^{*} - \varepsilon \underset{\mathbf{s}}{E}\left[\sum_{k=0}^{\infty}\gamma^{k}\right] = V^{*} - \frac{\varepsilon}{1-\gamma} \qquad (۸-۶)$$

و در نتیجه:

$$V^{*} - \frac{\varepsilon}{1-\gamma} < V' < V^{*} + \frac{\varepsilon}{1-\gamma} \qquad (۸-۷)$$

و یا

$$\left\|\Delta V\right\|_{\infty} < \frac{\varepsilon}{1-\gamma}$$

که در آن $\Delta V = V'^{*} - V^{*}$.

به همین صورت در مورد $Q(s,a)$ نیز می‌توان عبارت‌های مشابه‌ای به دست آورد. فرض کنید که عامل در حالت s باشد و با سیاست $\pi'(s)$ –که تابع‌ای از $Q'(s,a)$ است– عمل‌ای را انتخاب کند. محیط با احتمال $P_{sa} : (s,a) \mapsto s'$ به حالت جدید s' می‌رود. پس خواهیم داشت:

$$Q'^{*}(s,a) = \max_{\pi} \underset{s' \sim P_{sa}^{\pi}}{E}\left[r'_k + \gamma V'^{*}(s')\right] \qquad (۸-۹)$$

حال به مانند قبل می‌توان نوشت:

$$Q'^{*}(s,a)=\max_{\pi}\underset{s'\sim P_{sa}^{\pi'(s)}}{E}\left[r'_k+\gamma V'^{*}(s')\right]=\underset{s'}{E}\left[r(s)+\Delta r(s)+\gamma V'^{*}(s')\right]$$
$$<\underset{s'}{E}\left[r(s)+\Delta r(s)+\gamma\left(V^{*}(s')+\frac{\varepsilon}{1-\gamma}\right)\right]=\underset{s'}{E}\left[\left(r(s)+\gamma V^{*}(s')\right)+\left(\Delta r(s)+\frac{\varepsilon}{1-\gamma}\right)\right] \quad (۸-۱۰)$$

می‌بینید که کران بالا به دو قسمت تقسیم شده است: قسمت اول، همان $Q^{*}(s,a)$ است و قسمت دوم، ناشی از عدم قطعیت – یعنی:

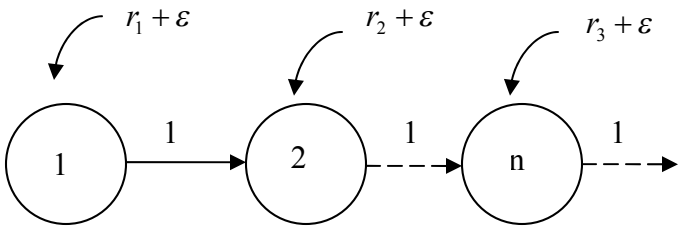
$$Q'^{*}(s,a)<Q^{*}(s,a)+\underset{s'}{E}\left[\Delta r(s)+\frac{\gamma\varepsilon}{1-\gamma}\right]=Q^{*}(s,a)+\frac{\varepsilon}{1-\gamma}+\underset{s'}{E}\left[\Delta r(s)\right]$$
$$<Q^{*}(s,a)+\frac{\gamma\varepsilon}{1-\gamma}+\varepsilon=Q^{*}(s,a)+\frac{\varepsilon}{1-\gamma} \quad (۸-۱۱)$$

و به همین ترتیب می‌توان نشان داد که:

$$Q'^{*}(s,a)=\underset{s'}{E}\left[r(s)+\Delta r(s)+\gamma V'^{*}(s')\right]>\underset{s'}{E}\left[r(s)+\Delta r(s)+\gamma\left(V^{*}(s')-\frac{\varepsilon}{1-\gamma}\right)\right]$$
$$=\underset{s'}{E}\left[\left(r(s)+\gamma V^{*}(s')\right)+\left(\Delta r(s)-\frac{\gamma\varepsilon}{1-\gamma}\right)\right]>Q^{*}(s,a)-\frac{\gamma\varepsilon}{1-\gamma}+\underset{s'}{E}\left[\Delta r(s)\right] \quad (۸-۱۲)$$
$$>Q^{*}(s,a)-\frac{\gamma\varepsilon}{1-\gamma}-\varepsilon=Q^{*}(s,a)-\frac{\varepsilon}{1-\gamma}$$

و در نتیجه داریم:

$$\left\|\Delta Q(s,a)\right\|_{\infty}<\frac{\varepsilon}{1-\gamma} \quad (۸-۱۳)$$



شکل ۱-۸. نمونه‌ای از مساله‌ی MDP که کران‌های ارایه‌شده دقیق‌اند.

که مشابه با قبل $\Delta Q = Q'^{*} - Q^{*}$.

ممکن است به نظر آید که کران‌های به دست آمده بسیار محافظه‌کارانه هستند. با این‌وجود مسایل‌ای وجود دارد که این کران‌ها دقیقا تخمین درست خطای تابع ارزش هستند. مثلا در شکل ۱-۸ که در آن انتقال حالت برای حرکت چپ به راست ۱ است و سیگنال تقویت همیشه با خطای ثابت‌ای معادل با ε آلوده شده است، کران به دست آمده معادل میزان دقیق خطا خواهد بود.

۸-۳) تاثیر عدم قطعیت سیگنال تقویت بر سیاست عامل

در این بخش تاثیر تغییر توابع ارزش بر سیاست عامل را بررسی می‌کنم. اگر عامل به *سیگنال تقویت بهینه‌ی ناشناخته* دست‌رسی داشته باشد، سیاست بهینه‌ی عامل $\pi(s,a)$ خواهد بود. حال سوال این است که به هنگام دریافت سیگنال تقویت خطادار ارتباط سیاست جدیدش $\pi'(s,a)$ با سیاست بهینه‌ی ناشناخته چگونه خواهد بود. با این‌کار می‌توانیم بفهمیم که از دیدگاه رفتارگرا تاثیر عدم قطعیت در سیگنال تقویت چگونه ظاهر می‌شود. برای مقایسه این دو چگالی احتمال از احتمال نسبی‌ی انتخاب عمل در حالت‌ای یک‌سان استفاده می‌کنیم: $\pi'(s,a)/\pi(s,a)$. اگر این مقدار به یک نزدیک بود، آن‌گاه می‌توان گفت که رفتار عامل خطادار چندان تفاوتی با عامل بهینه‌ی اصلی ندارد ولی اگر این مقدارتفاوت زیادی داشت (خیلی کوچک یا خیلی بزرگ بود) آن‌گاه رفتار عامل مطمئنا تغییر کرده است. در نتایج پیش رو فرض شده است که عامل از روش انتخاب عمل بولتزمن که از همه‌ی اطلاعات تابع ارزش حالت-عمل استفاده می‌کند برای انتخاب عمل بهره می‌گیرد. در روش انتخاب بولتزمن داریم:

$$P\left\{\text{selected action } = a^*(s) \middle| \begin{matrix} Q(s,a), \text{Boltzman} \\ \text{action selection, T} \end{matrix}\right\} = \frac{\exp\left(\frac{Q(s,a^*)}{T}\right)}{\sum_a \exp\left(\frac{Q(s,a)}{T}\right)} = \frac{N}{D} \qquad (۸-۱۴)$$

پس برای عامل خطادار خواهیم داشت[6]:

$$P\left\{a^*(s)\middle|Q'(s,a)\right\} = \frac{\exp\left(\frac{Q'(s,a^*)}{T}\right)}{\sum_a \exp\left(\frac{Q'(s,a)}{T}\right)} = \frac{\exp\left(\frac{Q(s,a^*)}{T}\right)\exp\left(\frac{\Delta Q(s,a^*)}{T}\right)}{\sum_a \left[\exp\left(\frac{Q(s,a)}{T}\right).\exp\left(\frac{\Delta Q(s,a)}{T}\right)\right]} = \frac{N'}{D'}$$

(۸-۱۵)

برای محاسبه‌ی کران بالا و پایینی برای این احتمال، می‌توان کران بالا و پایین صورت و مخرج را محاسبه کرد و سپس با استفاده از آن‌ها، کران کل را به دست آورد.

$$N' = \exp\left(\frac{Q(s,a^*)}{T}\right)\exp\left(\frac{\Delta Q(s,a^*)}{T}\right) = N.\exp\left(\frac{\Delta Q(s,a^*)}{T}\right) \qquad (۸-۱۶)$$

اگر تعریف کنیم:

$$\rho = \exp\left(\frac{\left\|\Delta Q(s,a)\right\|_\infty}{T}\right) \qquad (۸-۱۷)$$

آن‌گاه

$$\rho^{-1}N \le N' \le \rho N \qquad (۸-۱۸)$$

[6] توجه شود که شروط بدیهی‌ای مانند دما و انتخاب نوع عمل بولتزمن –به جز حالتی که یادآوری‌ی آن اهمیت داشته باشد– در فرمول‌های بعدی حذف شده است.

به همین صورت برای D' نیز می‌توان نوشت:

$$\sum_a \exp\left(\frac{Q(s,a)}{T}\right).\rho^{-1} \le D' = \sum_a \exp\left(\frac{Q(s,a)}{T}\right).\exp\left(\frac{\Delta Q(s,a)}{T}\right) \le \sum_a \exp\left(\frac{Q(s,a)}{T}\right).\rho$$

(۸-۱۹)

و در نتیجه:

$$\rho^{-1} D \le D' \le \rho D \qquad (۸-۲۰)$$

اکنون می‌توان نوشت:

$$\frac{\min(N')}{\max(D')} \le P\left\{a^*(s)\middle|Q'(s,a)\right\} \le \frac{\max(N')}{\min(D')} \qquad (۸-۲۱)$$

پس:

$$\rho^{-2} P\left\{a^*(s)\middle|Q(s,a)\right\} \le P\left\{a^*(s)\middle|Q'(s,a)\right\} \le \rho^2 P\left\{a^*(s)\middle|Q(s,a)\right\} \qquad (۸-۲۲)$$

و یا:

$$\rho^{-2} \le \frac{P\left\{a^*(s)\middle|Q'(s,a)\right\}}{P\left\{a^*(s)\middle|Q(s,a)\right\}} \le \rho^2 \qquad (۸-۲۳)$$

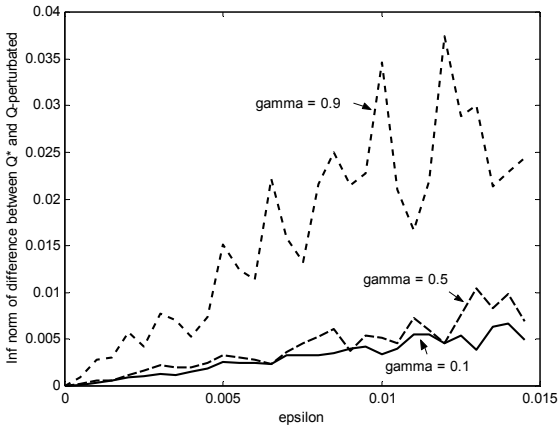
حال با در نظر گرفتن کران بخش پیش داریم:

$$\rho = \exp\left(\frac{\varepsilon}{(1-\gamma)T}\right) \qquad (۸-۲۴)$$

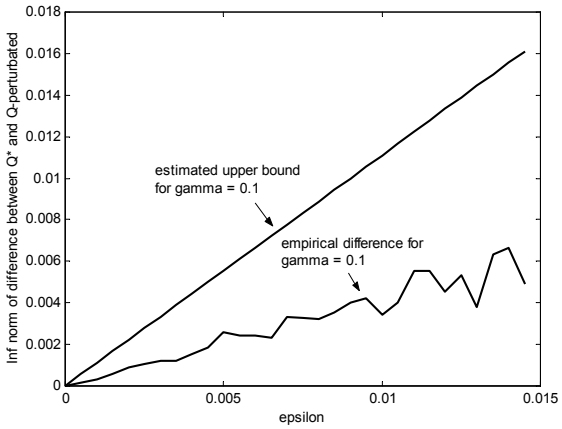
۸-۴) آزمایش‌ها

در این بخش درستی‌ی کران‌های محاسبه شده را با مقایسه با کران واقعی‌ای که از طریق شبیه‌سازی به دست آمده است نمایش می‌دهم. برای این‌کار به ازای مقادیر مختلف خطای (ε در (۸-۴)) میزان خطای ایجاد شده را $\|\Delta Q\|_\infty$ را با خطای پیش‌بینی در کران‌های به دست آمده شده مقایسه می‌کنم. مشابه با همین کار را برای تغییر در سیاست‌ها نیز انجام می‌دهم. با این‌کار گستره‌ی قابل استفاده کران‌های به دست آمده نیز تا حدودی مشخص می‌شود.

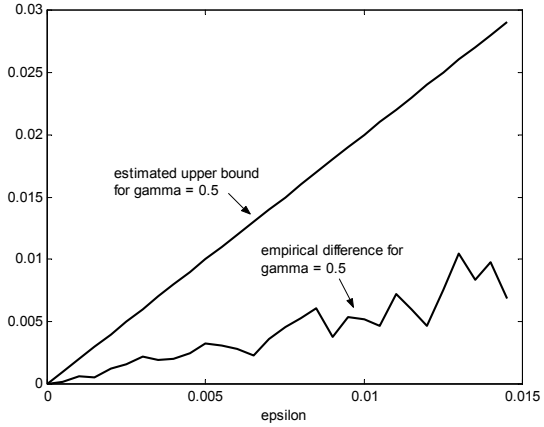
برای مقایسه می‌توان یکی از مسایل معمول و متداول‌ای را که با استفاده از روش یادگیری تقویتی به حل آن مبادرت می‌کنند انتخاب کرد. به عنوان مثال می‌توان به مساله‌ی مسیریابی در ماز، متعادل‌سازی‌ی آونگ معکوس، و یا حتی مسایل پیچیده‌تری مانند یادگیری‌ی رفتارها در شبیه‌سازی‌ی بازی‌ی فوتبال اشاره کرد. با این وجود، به دلیل آن‌که خصوصیات ذاتی‌ی مساله تاثیری بر نتایج نداشته باشد و هم‌چنین به دلیل آن‌که پیچیدگی‌هایی چون انتخاب حالت و نوع بازنمایی —که در مساله‌ای چون یادگیری رفتار در فوتبال مطرح است– باعث سخت شدن مساله حتی برای نسخه‌ی بدون خطای آن نشود از مسایل متداول استفاده نشده است، بلکه از مساله‌ای فرضی —ولی کلی– که به طور اتفاقی طراحی شده بهره گرفته شده است. بدین صورت که محیطای در نظر گرفته شده که دارای احتمال انتقال حالت ($P_{sa}:(s,a)\mapsto s'$) تصادفی است و پاداش انتقال به هر حالت جدید ($R(s)$) نیز به صورت تصادفی مشخص شده. در واقع این مساله معادل حرکت بهینه بر روی گراف‌ای تصادفی با یال‌های وزن‌دار است که وزن یال‌ها متناسب با زمان عبور از آن‌ها مشخص می‌شود و می‌توان آن را به عنوان کلی‌ترین حالت مساله‌ی یادگیری‌ی تقویتی (و در نتیجه برنامه‌ریزی‌ی پویا) در نظر گرفت. لازم است تاکید کنم که این مساله درست به همان اندازه‌ای که تئوری‌های این فصل احتیاج دارند کلی است و استفاده از مسایل دیگر یا زیر پا گذاشتن فرض‌هایی از تئوری است (مثلا با POMDP شدن مساله در حالی که فرمول‌بندی احتیاج به MDP بودن آن دارد) و یا این‌که قیدهایی بی‌فایده —جز فایده‌ی نمایش‌ای– به مساله اضافه می‌کند.



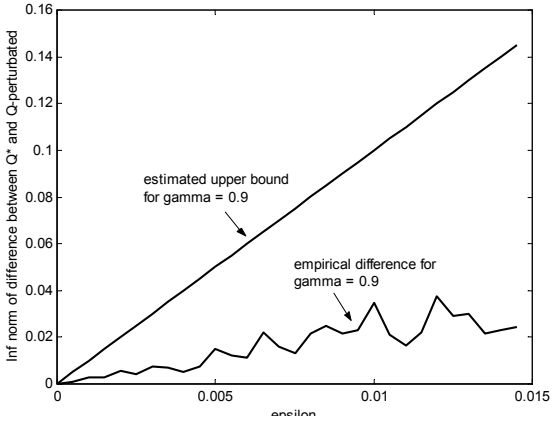
شکل ۲-۸. خطای $\|\Delta Q(s,a)\|_\infty$ به ازای مقادیر γ مختلف



شکل۳-۸. مقایسه بین خطای $\|\Delta Q(s,a)\|_\infty$ مشاهده شده و کران به دست آمده به ازای γ=0.1

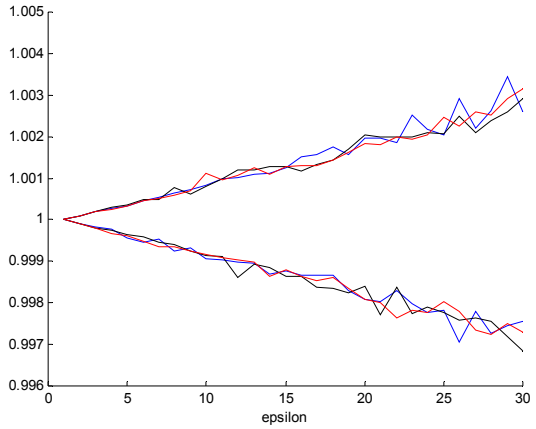


شکل ۴-۸. مقایسه بین خطای $\|\Delta Q(s,a)\|_\infty$ مشاهده شده و کران به‌دست آمده به ازای γ=0.5

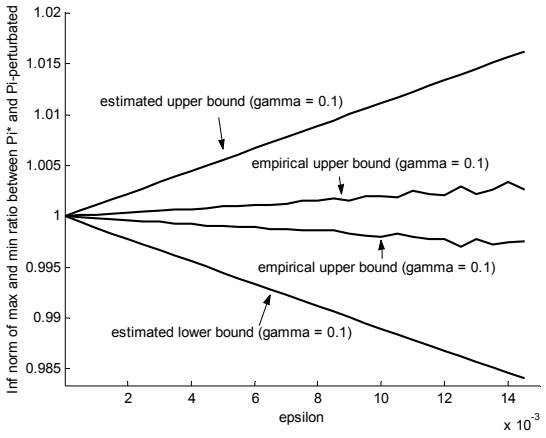


شکل ۵-۸. مقایسه بین خطای $\|\Delta Q(s,a)\|_\infty$ مشاهده شده و کران به دست آمده به ازای γ=0.9

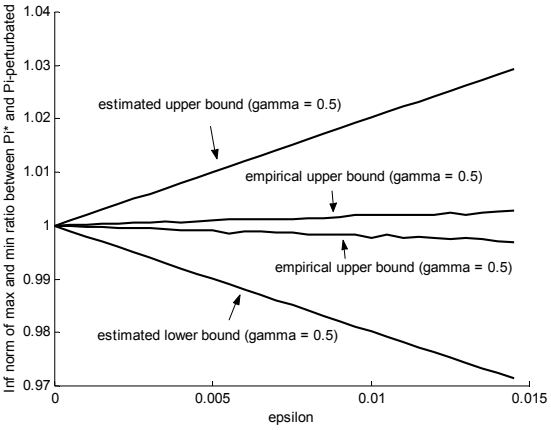
برای نتایجی که در این مقاله گزارش می‌شود، تعداد حالت‌ها (راس‌های گراف) را برابر ۳۰ در نظر می‌گیرم، تعداد عمل‌های ممکن در هر حالت را نیز ۵ قرار می‌دهم و احتمال انتقال را نیز از توزیعی یک‌نواخت انتخاب می‌کنیم با این شرط که در نهایت تابع انتقال احتمالی‌ی صحیحی‌ای تولید شود (یعنی مجموعه احتمال خروج از هر حالت–عمل برابر ۱ باشد). تابع تقویت را نیز از توزیع یک‌نواخت‌ای در بازه‌ی $(-1,1)$ انتخاب کرده‌ایم. هم‌چنین برای محاسبه‌ی سیاست در هر حالت نیز از انتخاب بولتزمان با دمای $T=1$ استفاده شده است. قابل ذکر است که برای بالا بردن دقت، به ازای هر مقدار ε، این آزمون ۱۰ مرتبه انجام شده است و بالاترین کران به دست آمده در نمودارها آمده است.



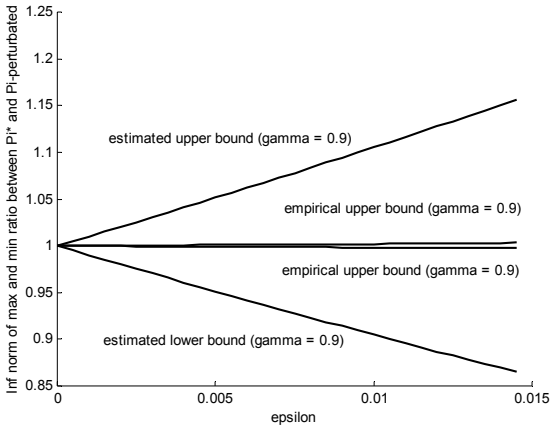
شکل۶-۸. کران بالا و پایین نسبت احتمالات عامل با سیگنال تقویت نادقیق به احتمالات عامل با سیگنال تقویت اصلی به ازای مقادیر مختلف γ (آبی: γ=0.1، مشکی: γ=0.5، قرمز: γ=0.9).



شکل ۷-۸. مقایسه بین نسبت احتمالات مشاهده شده و کران‌های به دست آمده به ازای γ=0.1



شکل ۸-۸. مقایسه بین نسبت احتمالات مشاهده شده و کران‌های به دست آمده به ازای γ=0.5



شکل ۹-۸. مقایسه بین نسبت احتمالات مشاهده شده و کران‌های به دست آمده به ازای γ=0.9

در شکل ۲-۸ میزان خطای عملی بین تابع ارزش دارای خطای سیگنال تقویت و تابع ارزش اصلی به ازای مقادیر مختلف ε و γ آمده است. همان‌طور که انتظار می‌رفت هر چقدر میزان خطای تابع تقویت بیش‌تر باشد، میزان خطای تابع ارزش نیز بیش‌تر خواهد بود. علاوه بر آن هنگامی که γ بزرگ‌تر باشد تاثیر خطا نیز بیش‌تر است. در شکل‌های ۳-۸ تا ۵-۸ کران خطایی که در عمل رخ داده با کران‌ای که در این مقاله به دست آوردیم (رابطه ۱۳-۸) مقایسه شده است. مشاهده می‌شود که در صورتی که γ و یا ε کوچک باشد، دقت کران بیش‌ترست. در شکل ۶-۸ کران‌های بالا و پایین نسبت احتمال انتخاب

عمل در حالت دارای سیگنال تقویت مخدوش شده با حالت اصلی در مثال مورد بررسی مقایسه شده است (به رابطه ۲۳–۸ نگاه کنید). نکته‌ی جالب این است که این مقادیر چندان به γ وابسته نیستند. به مانند قبل، این کران‌های عملی با آن‌چه محاسبات‌مان مشخص کرده است مقایسه می‌کنیم. این مقایسه به ازای مقادیر مختلف γ در شکل‌های ۷–۸ تا ۹–۸ آمده است. مشاهده می‌شود که به ازای مقادیر γ و یا ε بزرگ کران‌های‌مان بسیار محافظه‌کارانه هستند — گرچه به یاد بیاورید که کران‌های‌ی به دست آمده برای بعضی از مسایل کران دقیق هستند.

۸–۵) نتیجه‌گیری

در این فصل تاثیر خطای سیگنال تقویت بر ارزش هر حالت و هم‌چنین احتمال انتخاب عمل‌های هر حالت به صورت کران‌هایی به دست آمد. بدین صورت می‌توان تخمین‌ای کلی از این‌که در صورت خطا و عدم قطعیت در سیگنال تقویت به چه میزان عامل‌مان دچار تغییر خواهد شد به دست آورد. اما همان‌طور که در شبیه‌سازی‌ها دیده شد، این کران‌ها —به دلیل طبیعت کلی‌شان– محافظه‌کارانه هستند و به ازای مقادیر بزرگ خطا عملا بی‌فایده خواهند بود. به همین دلیل عقلایی می‌نماید که بتوان با توجه به ساختار خود مساله کران‌های کم محافظه‌کارانه‌تری به دست آورد. از طرف دیگر ممکن است بتوان با تحلیل‌های دقیق‌تر ریاضی و با استفاده از کران‌های متداول در احتمالات (مانند کران Hoeffding و ...) به کران‌های دقیق‌تری —که در اکثر موارد و نه همه‌ی موارد درست‌اند[۷]– رسید.

[7] مشابه با ایده‌ی Probably Approximately Correct (PAC) که در تئوری‌های یادگیری ماشینی‌ی آماری به کار می‌روند [Vidyasagar03].

۹) نتیجه‌گیری و پیش‌نهاد

در این پایان‌نامه چندین روش نوین برای خودکارسازی‌ی طراحی‌ی عامل‌های هوش‌مند رفتارگرا ارایه دادم. برخلاف بسیاری از پژوهش‌های پیشین، سعی در ارایه‌ی نگاه‌ای چندجانبه و دقیق به مساله‌ی طراحی‌ی عاملِ رفتارگرا کردم. تا حد ممکن مساله را به قالب ریاضی در آوردم و آن را به طور دقیق تحلیل کردم. این موضوع به خصوص در مورد یادگیری ساختار و یادگیری رفتار نمود بیش‌تری داشت. نتایج تحلیل‌های انجام شده منجر به چندین روش‌های مختلف یادگیری و تکامل شد. این روش‌ها به طور فهرست‌وار بدین شرح‌اند:

- یادگیری ساختار
  - بازنمایی مرتبه صفر
  - بازنمایی مرتبه یک
- یادگیری رفتار
- تکامل رفتار
  - اشتراک سازگاری یک‌نواخت
  - اشتراک سازگاری ارزش‌محور
- الگوریتم ممتیک

در فصل مربوط به یادگیری ساختار (فصل ۳) دو نوع بازنمایی دانش مختلف پیش‌نهاد شد که با استفاده از آن، عامل می‌تواند اطلاعات دریافتی از محیط را ثبت کرده و به‌ترین ساختار ممکن را بیابد. نشان دادم که چگونه می‌توان ارزش عامل را به اجزای ساده‌تری تجزیه کرد و متناسب با آن تجزیه، روش‌ای برای تخصیص اعتبار به رفتارها و چگونگی‌ی به روز آوری‌ی آن‌ها را معرفی کردم.

در فصل مربوط به یادگیری رفتار (فصل ۴) با معرفی‌ی عمل مجازی‌ای به نام NA به رفتارها اجازه دادم تا تخمین‌ای از عمل‌کرد رفتارهای دیگر به دست آورند و متناسب با آن تخمین، تصمیم بگیرند که آیا به‌تر است خودشان کنترل عامل را در اختیار بگیرند یا اجازه‌ی چنین کاری را به رفتارهای دیگر بدهند. با اضافه کردن این عملِ مجازی، امکان هم‌کاری‌ی بین عامل‌ها (رفتارها) بیش‌تر می‌شود.

روی‌کرد دیگری که در این پایان‌نامه مطرح شد، استفاده از تکامل هم‌کارانه‌ی رفتارها بود. روش‌ای برای تکامل رفتارها ارایه دادم (فصل ۵) که ماجولارتر و قابل استفاده‌ی مجددتر از روش‌های یک‌پارچه‌ی پیشین است. دو روش مختلف اشتراک سازگاری بین جمعیت‌های تکامل‌یابنده معرفی کرد که هر کدام برگرفته از فلسفه‌ای خاص بودند.

در نهایت تفسیری جدید و متفاوت از مم ارایه دادم (فصل ۵) که اجازه می‌دهد به جای هدر رفتن دانش کسب‌شده در طول زندگی‌ی عامل، آن دانش به صورت سنت‌های فرهنگی‌ی قابل استفاده‌ی مجدد برای عامل‌های جدید در می‌آید. نتایج حاصل در آزمایش‌های انجام گرفته موفقیت‌آمیز بود.

با در اختیار داشتن این مجموعه از روش‌ها، طراح با دست باز می‌تواند یک یا چند نوع از این روش‌ها را با توجه به دانش‌اش نسبت به مساله اختیار کند. به عنوان نمونه اگر او جعبه‌ابزاری از رفتارهای مناسب در اختیار داشته باشد می‌تواند از یکی از دو روش یادگیری ساختار بهره بگیرد و اگر نه، از یادگیری یا تکامل برای تولید رفتارهای جدید استفاده کند. به نظر می‌رسد روش‌های پیش‌نهادی تا حد قابل قبولی روند طراحی‌ی خودکار عامل هوش‌مند را ساده کنند. نتایج به‌کارگیری‌ی این روش‌ها در فصل آزمایش‌ها (فصل ۶) بر روی دو مساله‌ی مختلف به نمایش گذاشته شده است و منجر به نتایج قابل رقابت و حتی به‌تر از معماری‌های طراحی شده توسط انسان بود.

علاوه بر تحلیل‌هایی که به طور مستقیم به روش‌ای برای طراحی منجر شدند، تحلیل‌های دیگری نیز صورت گرفت که درکِ ما را از مساله‌ی یادگیری در ساختاری سلسله مراتبی بالا می‌برد. یکی از این تحلیل‌ها، بررسی‌ی احتمال کنترل‌کننده بودن و به دست آوردن توزیع احتمال تعداد دفعه‌ی کنترل‌کننده بودن برای معماری‌های رقابتی‌ی موازی‌ای مانند PPSSA است (فصل ۷). نتایج حاصل با این‌که شاید به طور مستقیم در روند طراحی به کار نروند، اما درک طراح از مساله را افزایش می‌دهد.

دسته‌ی دیگر تحلیل‌ها، بررسی‌ی اثر عدم قطعیت سیگنال تقویت بر تابع ارزش و تابع سیاست عامل بود (فصل ۸). با محاسبات ساده‌ای نشان دادم که عدم قطعیت چگونه می‌تواند رفتار عامل را عوض کند. این موضوع هنگامی اهمیت پیدا می‌کند که بدانیم در بسیاری از مسایل دنیای واقعی، دانشِ ما نسبت به سیگنال تقویت دقیق نیست و آن سیگنال ممکن است دارای خطا باشد

با این‌که این پژوهش تا حد ممکن وسیع و گسترده بوده است، اما هم‌چنان مسایل مهم‌ای برای بررسی باقی مانده. به عنوان مثال می‌توان به موارد زیر اشاره کرد:

- **تعمیم ایده‌های مطرح‌شده به ساختارهای پیچیده‌تر.**

در این پایان‌نامه فرض کرده‌ام که معماری‌ی رفتارگرای سلسله‌مراتبی‌مان دارای ساختاری اکیدا موازی است. با این‌که بسیاری از رفتارها را می‌توان با همین معماری ایجاد کرد، اما به نظر می‌رسد گسترش این ایده‌ها به معماری‌های کلی‌تر مفید باشد. گمان می‌کنم در صورت‌ای که بتوان ارزش معماری را به صورت ارزش اجزای ساده‌تری در آورد، آن‌گاه یادگیری ساختار آن معماری‌ها نیز ممکن باشد.

- **استخراج خودکار فضای حالت و عمل رفتارها**

در این پایان‌نامه فرض کرده بودم که بُعد فضای حالت و عمل رفتارها از پیش دانسته است. خودکاری‌سازی‌ی این فرآیند و استخراج خودکار ویژگی‌هایی که می‌توانند برای تصمیم‌گیری مطلوب باشند به نظر مساله‌ی دشواری می‌نماید. پژوهش‌های اولیه‌ای (و البته نه قطعی) که در این زمینه انجام داده‌ام نشان می‌دهد که استفاده از روش‌های خوشه‌بندی[۱۸۰] متداول نمی‌تواند خیلی موثر باشد و لازم است الگوریتم‌های خوشه‌بندی‌ی جدیدی پیش‌نهاد شوند که علاوه بر در نظر گرفتن ویژگی‌های توپولوژیک حالت‌های تجربه شده، از ارزش آن حالت‌ها نیز بهره بگیرد. دو پژوهش‌ای که تا حدی به این موضوع پرداخته‌اند HEXQ ([Hengst02]) و روش ادغام حالت [Engel01] است.

[180] Clustering

- **هم‌گرایی روش‌های یادگیری**

در این پایان‌نامه اثبات‌ای از هم‌گرایی توابع ارزش به مقادیر بهینه‌ی عمومی و یا حتی محلی‌ی خود انجام نشد. پژوهش راجع به این‌که آیا روش‌های یادگیری پیش‌نهادشده به بهینه‌ای (عمومی یا محلی) هم‌گرا می‌شوند یا خیر جالب توجه خواهد بود.

- **طراحی سیگنال تقویت**

مساله‌ی طراحی‌ی سیگنال تقویت، مساله‌ی اساسی در یادگیری تقویتی است و منحصر به این پایان‌نامه نمی‌شود. این‌که آیا روش‌ای کلی و سیستماتیک برای طراحی‌ی سیگنال تقویت وجود دارد یا خیر مشخص نیست. تنها چند پژوهش محدود درباره‌ی این موضوع انجام شده است که می‌توان به [Dorigo94]، [Dorigo97]، [Ng99] و [Farahmand05a] اشاره کرد. یکی از ایده‌های مطرح شده، ایجاد خودکار سیگنال تقویت با استفاده از شمایی کلی از آن‌چه عامل می‌بایست رفتار کند است. نتایج مقدماتی‌ای در این‌باره به دست آمده که البته در این پایان‌نامه گزارش نشده است.

- **تجرید دانش سطح‌پایین کسب شده و استفاده از آن در تصمیم‌گیری‌های سطح‌بالاتر**

یکی از زمینه‌های جالب توجه برای ادامه‌ی این پژوهش، تجرید دانش کسب شده توسط رفتارها و استفاده از آن در لایه‌های سطح‌بالاتر تصمیم‌گیری است. تلاش‌های اولیه‌ای در این زمینه انجام شده است که نتایج جالب توجه‌ای داشته است. این نتایج تاکنون گزارش نشده‌اند.

۱۰) مراجع

1. [Arkin98] R. C. Arkin, Behavior-Based Robotics, *MIT Press*, CA, 1998.
2. [Boyan95] J. Boyan and A. W. Moore, "Generalization in reinforcement learning: safely approximating the value function," in G. Tesauro, D. S. Touretzky, and T. K. Leen (Eds.), *Advances in Neural Information Processing System 7*, MIT Press, Cambridge MA, 1995.
3. [Brooks86] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation R.A-2*, 1986, pp. 14-23.
4. [Brooks89] R. A. Brook, "A robot that walks: emergent behavior from a carefully evolved network," *Neural Computation* 1(2), 1989, pp. 252-262.
5. [Brooks91a] R. A. Brooks, "New approaches to robotics," *Science* 253, 1991, pp. 1227-1232.
6. [Brooks91b] R A. Brooks, "Intelligence without reason," in Proc. *Int. Joint Conf. AI*, 1991, pp. 569-595.
7. [Brooks91c] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, 47, 1991, pp. 139-159.
8. [Buriol04] L. Buriol, P. M. Franca, P. Moscato, "A new memetic algorithm for the asymmetric traveling salesman problem," *Journal of Heuristics*, 10, pp. 483-506, 2004.
9. [Chernova04] S. Chernova and M. Veloso, "An Evolutionary Approach to Gait Learning for Four-Legged Robots," *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2004.
10. [Cliff93] D.T. Cliff, I. Harvey, and P. Husbands, "Explorations in Evolutionary Robotics," *Adaptive Behavior*, 2, pp. 73-110, 1993.
11. [Dawkins76] R. Dawkins, The Selfish Gene, *Oxford University Press*, 1976.
12. [Dayan93] P. Dayan and G. Hinton, "Feudal reinforcement learning," *Advances in Neural Information Processing Systems (NIPS)*, 5, Morgan Kaufmann, San Francisco, CA, 1993, pp. 271–278.
13. [Dorigo94] M. Dorigo and M. Colombetti, "Robot shaping: developing autonomous agents through learning," *Artificial Intelligence*, 71(4), 1994, pp. 321-370.
14. [Dorigo97] M. Dorigo and M. Colombetti, Robot Shaping: An Experiment in Behavior Engineering, *MIT Press*, CA, 1997.
15. [Engel01] Y. Engel and S. Mannor, "On finding good state aggregation functions," *Workshop on Hierarchy and Memory in Reinforcement Learning, International Conference on Machine Learning (ICML)*, 2001.
16. [Farahmand04] A. M. Farahmand, M. Nili Ahmadabadi, and B. N. Araabi, "Behavior hierarchy learning in a behavior-based system using reinforcement

learning," *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Sendai, Japan, 2004, pp. 2050-2055.
17. [Farahmand05a] A. M. Farahmand and M. Nili Ahmadabadi, "The effect of reinforcement signal in reinforcement learning," *The Computer Society of Iran Computer Conference (CSICC)*, 2005 (in Persian).
18. [Farahmand05b] A.M. Farahmand, M. Nili Ahmadabadi, B. N. Araabi, "Behavior and hierarchy development in behavior-based systems using reinforcement learning," *Submitted to Artificial Intelligence*, 2005.
19. [Federici03] D. Federici, "Combining genes and memes to speed up evolution," in Proceedings of *International Congress on Evolutionary Computation (CEC)*, 2003.
20. [Floreano94] D. Floreano, and F. Mondada, "Automatic creation of an agent: genetic evolution of a neural network driven robot," In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson (Eds.), *From Animals to Animats III*, Cambridge, MA: MIT Press, 1994.
21. [Floreano96] D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26, pp. 396-407, 1996.
22. [Floreano99] D. Floreano, S. Nolfi, and F. Mondada, "Co-evolution and ontogenetic change in competing robots," *Robotics and Autonomous Systems,* 1999.
23. [Floreano00] D. Floreano and J. Urzelai, "Evolutionary robotics: the next generation," in T. Gomi (ed.), Proceedings of *Evolutionary Robotics*, III, pp. 231-266, Ontario, 2000.
24. [Goldberg89] David E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, *Addison-Wesley Publishing*, 1989.
25. [Gomez97] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, 5, pp. 317-342, 1997.
26. [Harati04] A. Harati and M. Nili Ahmadabadi, "Experimental analysis for knowledge based multiagent credit assignment," *Neural Information Processing: Research and Development, Research and Development,* Rajapakse, Jagath C.; Wang, Lipo (Eds.), Springer-Verlag, May 2004.
27. [Harvey93] I. Harvey, P. Husbands, and D. Cliff, "Issues in evolutionary robotics," in *From Animals to Animats II: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, J. Meyer, H. L. Roitblat, and S. W. Wilson, Eds. MIT Press-Bradford Books, Cambridge, MA, 1993.
28. [Hengst02] B. Hengst, "Discovering hierarchies in reinforcement learning with HEXQ," in Claude Sammut and Achim Hoffmann, (Eds.), *Proceeding of Nineteenth International Conference on Machine Learning*, Morgan-Kaufman, 2002, pp. 243-250.
29. [Hergenhahn01] B. R. Hergenhahn and M. H. Olson, An Introduction to Theories of Learning, *Prentice-Hall*, 2001.
30. [Jaakkola94] T. Jaakkola, M. I. Jordan, and S. Singh, "On the convergence of stochastic iterative dynamic programming algorithms," *Neural Computation*, 6(6), 1994, pp. 1185-1201.

31. [Kaelbling96] L. P. Kaelbling and M. L. Littman, and A. W. Moore, "Reinforcement learning: a survey," *Journal of Artificial Intelligence Research*, 4, 1996, pp. 237-285.
32. [Kohl04] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," *Proceeding of the IEEE International Conference on Robotics and Automation (ICRA)*, 2004.
33. [Koza94] John Koza, "Evolution of a subsumption architecture that performs a wall following task for an autonomous mobile robot via genetic programming," In S.J. Hanson, T. Petsche, M. Kearns, and R.L. Rivest (editors), *Computational Learning Theory and Natural Learning Systems*, The MIT Press, vol. 2, pp. 321-346, 1994.
34. [Krasnogor04] N. Krasnogor and S. Gustafson, "A study on the use of 'Self-Generation' in memetic algorithms," *Natural Computing* 3 (1), pp. 53-76, 2004.
35. [Maes90] P. Maes and R. A. Brooks, "Learning to coordinate behaviors," in: *Proc. AAAI*, Boston, MA, 1990, pp. 796-802.
36. [Mahadevan92] S. Mahadevan and J. Connell, "Automatic programming of behavior-based robots using reinforcement learning," *Artificial Intelligence*, 55, 1992, pp. 311-365.
37. [Matarić92] M. J. Matarić, "Integration of representation into goal-driven behavior-based robots," *IEEE Transactions on Robotics and Automation*, 8(3), 1992, pp. 46-54.
38. [Matarić94] M. J. Matarić, "Reward function for accelerated learning," in: W. W. Cohen and H. Hirsh, (Eds.), Proc. 8th *International Conference on Machine Learning*, Morgan Kaufmann, 1994, pp. 181-189.
39. [Matarić97] M. J. Matarić, "Reinforcement learning in the multi-robot domains," *Autonomous Robots*, vol. 4, no. 1, 1997, pp. 73-88.
40. [Matarić98] M. J. Matarić, "Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior," *Trends in Cognitive Science*, vol. 2, no. 3, 1998, pp. 82-87.
41. [Matarić01] M. J. Matarić, "Learning in behavior-based multi-robot systems: policies, models, and other agents," *Cognitive System Research*, special issue on Multi-disciplinary studies of multi-agent learning, Ron Sun, (Ed.), 2(1), 2001, pp. 81-93.
42. [McCallum95a] R. A. McCallum, "Instance-based state identification for reinforcement learning," *Advances in Neural Information Processing Systems (NIPS)*, 1995.
43. [McCallum95b] R. A. McCallum, "Instance-based Utile Distinctions for reinforcement learning with hidden state," Proceedings of the 12th *International Machine Learning Conference (ML)*, 1995.
44. [McGovern01] A. McGovern and A. Barto, "Automatic discovery of subgoals in reinforcement learning using diverse density," in: C. Brodley and A. Danyluk, (Eds.), Proceeding of the 18th *International Conference on Machine Learning*, San Francisco, CA., Morgan Kaufmann, 2001, pp. 361-368.

45. [Merz99] P. Merz and B. Freisleben, "A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem," In Proceedings of the 1999 *International Congress of Evolutionary Computation*, pp. 2063-2070, 1999.
46. [Merz00] P. Merz and B. Freisleben, "Fitness landscape analysis and memetic algorithms for the quadratic assignment problem," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 337-352, 2000.
47. [Michaud98] F. Michaud and M. J. Matarić, "Learning from history for behavior-based mobile robots in non-stationary conditions," *Autonomous Robots and Machine Learning* AR:5, ML:31, 1998, AR:335-354, ML:141-167.
48. [Mitchell97] T. M. Mitchell, Machine Learning, *McGraw-Hill*, 1997.
49. [Moscato03] P. Moscato and C. Cotta, "A gentle introduction to memetic algorithms," In F. Glover and G. Kochenberger (Eds.), *Handbook of Metaheuristics*, pp. 105-144. Kluwer Academic Publishers, Boston MA, 2003.
50. [Moscato92] P. Moscato and M. Norman, "A 'Memetic' approach for the travelling salesman problem – implementation of a computational ecology for combinatorial optimisation on message-passing systems," In Proceeding of the *International Conference on Parallel Computing and Transputer Applications*, IOS Press, Amsterdam, 1992.
51. [Ng99] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: theory and application to reward shaping," Proc. 16th *International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, 1999.
52. [Nili01] M. Nili Ahmadabadi and E. Nakano, "A constrain and move approach to distributed object manipulation," *IEEE Transaction on Robotics and Automation*, vol. 17, no. 2, 2001, pp. 157-172.
53. [Nili02] M. Nili Ahmadabadi and M. Asadpour, "Expertness based cooperative Q-learning," *IEEE Transaction on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 32, no. 1, 2002, pp. 66-76.
54. [Nolfi00] S. Nolfi and D. Floreano, Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines, *MIT Press*, Cambridge, MA, 2000.
55. [Ong04] Y.S. Ong and A. J. Keane, "Meta-lamarckian learning in memetic algorithms," *IEEE Transaction on Evolutionary Computation*, vol. 8, no. 2, pp. 99-110, 2004.
56. [Parker98] L. Parker, "ALLIANCE: an architecture for fault-tolerant multi-robot cooperation," *IEEE Transactions on Robotics and Automation*, 14 (2), 1998, pp. 220-240.
57. [Peshkin01] L. Peshkin, Reinforcement Learning by Policy Search, Ph.D. Dissertation, *MIT*, 2001.
58. [Precup00] D. Precup, Temporal Abstraction in Reinforcement Learning, Ph.D. Dissertation, *University of Massachusetts*, Amherst, MA, 2000.
59. [Puterman94] M. L. Puterman, Markov Decision Process, *John Wiley and Sons*, New York, 1994.

60. [Radcliffe94] N. J. Radcliffe and P. D. Surry, "Formal memetic algorithm," In T. Fogarty (ed.), *Evolutionary Computing: AISB workshop*, vol. 865 of Lecture Notes in Computer Science, pp. 1-16, Springer-Verlag, Berlin, 1994.
61. [Robbins51] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics* 22, 1951, pp. 400-407.
62. [Shoham04] Y. Shoham and R. Powers, 'On the agenda(s) of research on multi-agent learning," *AAAI Fall Symposium on Artificial Multi-Agent Learning*, 2004.
63. [Sing94] S. Singh, R. Yee, "An upper bound on the loss from approximate optimal-value functions," *Machine Learning*, 16(3), pp. 227-233, 1994.
64. [Singh00] S. Singh, T. Jaakkola, M. Littman, and C. Szepesvari, "Convergence results for single-step on-policy reinforcement-learning algorithms," *Machine Learning*, vol. 38, no. 3, pp. 287-308, 2000.
65. [Singh04] S. Singh, A. G. Barto, S. Singh, N. Chentanez, "Intrinsically motivated reinforcement learning," *In Advances in Neural Information Processing (NIPS)*, 18, 2004.
66. [Sutton88] R. S. Sutton, "Learning to predict by the method of temporal differences," *Machine Learning*, 3(1), 1988, pp. 9-44.
67. [Sutton98] R. S. Sutton, and A. G. Barto, Reinforcement Learning: An Introduction, *MIT Press*, Cambridge, MA, 1998.
68. [Sutton99] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: a framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, 112, 1999, pp. 181-211.
69. [Thrun93] S. Thrun and A. Schwartz, "Issues in using function approximation for reinforcement learning," In Proceedings of the *4th Connectionist Models Summer School*, 1993.
70. [Togelius04] J. Togelius, "Evolution of a subsumption architecture neurocontroller," *Journal of Intelligent and Fuzzy Systems*, 15:1, pp. 15-20, 2004.
71. [Urzelai98] J. Urzelai, D. Floreano, M. Dorigo, and M. Colombetti, "Incremental robot shaping," *Connection Science*, 10, 341-360, 1998.
72. [Vidyasagar03] M. Vidyasagar, Learning and Generalization: with Applications to Neural Networks – 2nd edition, *Springer-Verlag*, 2003.
73. [Wang96] Z. D. Wang, E. Nakano, and T. Matsukawa, "Realizing cooperative object manipulation using multiple behavior-based robots," in: Proc. *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, vol. 1, Osaka, Japan, 1996, pp. 310–317.
74. [Watkins89] C. J. C. H. Watkins, Learning from Delayed Rewards, Ph.D. Dissertation, *Cambridge University*, England, 1989.
75. [Watkins92] C. J. C. H. Watkins and P. Dayan, Q-learning, *Machine Learning*, 8(3), 1992, pp. 279-292.
76. [Williams94] R. J. Williams and L. C. Baird, "Tight performance bounds on greedy policies based on imperfect value functions," in: Proc. 10th *Yale Workshop on Adaptive and Learning Systems*, 1994.

77. [Zou04] P. Zou, Z. Zhou, G. Chen, and X. Yao, “A novel memetic algorithm with random multi-local-search: a case study of tsp,” In Proceedings of the 2004 *Congress on Evolutionary Computation* (CEC04), pp. 2335-2340, 2004.

ضمیمه الف) واژه‌نامه فارسی–انگلیسی

از آن‌جا که بسیاری از واژه‌های تخصصی‌ی به‌کار رفته در این پایان‌نامه هنوز ترجمه‌ی مناسب و مقبول‌ای ندارند، لازم دیدم فهرستی از واژه‌هایی را که در طول متن به کار برده‌ام در ضمیمه‌ای جداگانه بیاورم. با وجود آن‌که هر کجا واژه‌ای جدید به کار رفته است معادل آن نیز در زیرنویس آمده، اما فهرست‌ای جداگانه بدون شک خواندن این پایان‌نامه را راحت‌تر می‌کند. پیشاپیش لازم است بگویم که نویسنده خود اعتقاد دارد برای بعضی از این واژه‌ها ترجمه‌ی مناسب‌تری نیز ممکن است.

ادراک	Perception
ارزش	Value
اضافه‌شونده	Incremental
اعمال گسترش–زمانی یافته	Temporally Extended Action
افراز	Partition
افزایشی	Incremental
الگوریتم ممتیک	Memetic Algorithm
انجماد شبیه‌سازی‌شده	Simulated Annealing
با سرپرست	Supervised
بازگشتی	Recursive
بازنمایی	Representation
برنامه‌ریزی	Planning
برنامه‌ریزی پویا	Dynamic Programming
برنامه‌ریزی ژنتیکی	Genetic Programming
برنولی	Bernoulli
بلندکردن جسم با استفاده از مکانیزم چنگالی	Fork Lifting Mechanism
به طور جزئی رویت‌پذیر	Partial Observable
پاداش بازگشته	Return
پایه‌ای	Primitive
پایین به بالا	Bottom up
پوشش‌دادن	Coverage
پیچیدگی نمونه	Sample Complexity
تجزیه	Decompose
تحریک	Excite
تخمین‌زن پیش‌رفت	Progress Estimator
تزویج	Crossover
تصویرکردن	Projection
تظاهر	Emergence
تعداد آزمون‌ها	Trial
تعقیب دیوار	Wall Following
تعمیم	Generalization
تعویض‌پذیری	Commutativity
تقریب‌زن عمومی	General Function Approximator
تنوع فرهنگی	Cultural Diversity

توزیع احتمال انتقال حالت	State Transition Probability
توزیع‌پذیری	Distributivity
توزیع‌شده	Distributed
تکامل بالدوینی	Baldwinian Evolution
تکامل لامارکی	Lamarckian Evolution
تکامل مصنوعی	Artificial Evolution
تکامل هم‌کارانه‌ی رفتارها	Cooperative Evolution of Behaviors
تک‌گام	Single Step
جبر بول	Boolean Algebra
جدول ارجاعی	Lookup Table
جستجوی تصادفی	Stochastic Search
جستجوی کامل فضا	Exhaustive Search
جسم‌دار	Embodied
جمع‌شونده بودن	Contraction
جهت‌ده اولیه	Bias
جهش	Mutation
چندگام	Multi Step
حافظه‌محور	Memory-based
حریصانه	Greedy
خبرگی	Expertness
خنثی	Neutral
خودخواه	Selfish
خوشه‌بندی	Clustering
در محیط قرار گرفته	Situated
دو به دو مجزا	Mutually Exclusive
ذهنی	Subjective
رآکتیو	Reactive
رفتارگرایی	Behaviorism
رویت‌پذیر	Observable
زاویه‌ی کج‌شدگی	Tilt Angle
زمین‌نشان	Landmark
زیربرنامه	Subroutine
زیرهدف	Subgoal
زیر–وظیفه	Subtask
ژنوتایپ	genotype
سازگاری	Fitness
سرکوب	Suppress
سنسورهای برخوردی	Bumper
سونار	Sonar
سیاست	Policy
سیاست‌محور	Policy-based
سیستم‌های چندعامله‌ی هم‌کار	Cooperative Multi-Agent Systems
سیگنال تقویت	Reinforcement Signal
سیگنال تقویت شکل‌دهی‌شده	Shaped Reinforcement Signal
شرکت‌پذیری	Associativity
عامل	Agents
عامل هوش‌مند	Intelligent Agent
عمل	Action
عمل پایه‌ای	Primitive Action
عمل پوچ	No Action
عینی	Objective
غیرقابل پی‌گیری	Intractable
فاکتور فراموشی	Forgetting Factor
فرهنگ	Culture

فضای بازنمایی	Representation Space
فضای فرضیه	Hypothesis Space
فضای مساله	Problem Space
فعال‌شده	Activated
قابلیت استفاده مجدد	Reusability
کارکرد	Function
کارکردگرایی	Functionalism
کامل	Complete
کاوش	Exploration
کلی	Global
کنترل‌کننده	Controlling
گره‌های انتخاب	Choice Node
گمراه‌کننده	Deceptive
گیره	Compliance
لایه‌ای	Layered
متمم	Complement
متوسط‌گیر لغزان	Moving Average
مجرد	Abstract
محرک	Actuator
محلی	Local
محیط	Environment
مخزن ژنتیکی	Genetic Pool
مخزن مم	Meme Pool
مساله‌ی بهینه‌سازی ترکیبیاتی	Combinatorial Optimization Problem
معماری Subsumption کاملا موازی	Purely Parallel SubSumption Architecture (PPSSA)
معماری‌های رفتارگرا	Behavior-based Architectures
مقیاس‌پذیر	Scalable
مقیاس‌شده	Scaled
مم	Meme
مونت‌کارلو	Monte Carlo
مونت‌کارلو اولین–ملاقات	First Visit Monte Carlo Method
مک‌کالوم	McCallum
ناایستا	Non-Stationary
ناعمل	No Action
ناوردا	Invariance
نمونه	Instance
هاپفیلد	Hopfield
هم‌گرایی زودهنگام	Premature Convergence
یادگیری تقویتی	Reinforcement Learning
یادگیری تقویتی سلسله‌مراتبی	Hierarchical Reinforcement Learning
یاری‌بهر	Contribution
یک‌پارچه	Monolithic

ضمیمه ب) واژه‌نامه انگلیسی–فارسی

Abstract	مجرد
Action	عمل
Activated	فعال‌شده
Actuator	محرک
Agents	عامل
Artificial Evolution	تکامل مصنوعی
Associativity	شرکت‌پذیری
Baldwinian Evolution	تکامل بالدوینی
Behavior-based Architectures	معماری‌های رفتارگرا
Behaviorism	رفتارگرایی
Bernoulli	برنولی
Bias	جهت‌ده اولیه
Boltzman Action Selection	انتخاب عمل بولتزمان
Boolean Algebra	جبر بول
Bottom up	پایین به بالا
Bumper	سنسورهای برخوردی
Choice Node	گره‌های انتخاب
Clustering	خوشه‌بندی
Combinatorial Optimization Problem	مساله‌ی بهینه‌سازی ترکیبیاتی
Commutativity	تعویض‌پذیری
Complement	متمم
Complete	کامل
Compliance	گیره
Contraction	جمع‌شونده بودن
Contribution	یاری‌بهر
Controlling	کنترل‌کننده
Cooperative Evolution of Behaviors	تکامل هم‌کارانه‌ی رفتارها
Cooperative Multi-Agent Systems	سیستم‌های چندعامله‌ی هم‌کار
Coverage	پوشش‌دادن
Crossover	تزویج
Cultural Diversity	تنوع فرهنگی
Culture	فرهنگ
Deceptive	گمراه‌کننده
Decompose	تجزیه
Distributed	توزیع‌شده
Distributivity	توزیع‌پذیری
Dynamic Programming	برنامه‌ریزی پویا
Embodied	جسم‌دار
Emergence	تظاهر
Environment	محیط
Excite	تحریک
Exhaustive Search	جستجوی کامل فضا
Expertness	خبرگی
Exploration	کاوش
First Visit Monte Carlo Method	مونت‌کارلو اولین–ملاقات
Fitness	سازگاری
Forgetting Factor	فاکتور فراموشی
Fork Lifting Mechanism	بلندکردن جسم با استفاده از مکانیزم چنگالی

Function	کارکرد
Functionalism	کارکردگرایی
General Function Approximator	تقریب‌زن عمومی
Generalization	تعمیم
Genetic Pool	مخزن ژنتیکی
Genetic Programming	برنامه‌ریزی ژنتیکی
Genotype	ژنوتایپ
Global	کلی
Greedy	حریصانه
Hierarchical Reinforcement Learning	یادگیری تقویتی سلسله‌مراتبی
Hopfield	هاپفیلد
Hypothesis Space	فضای فرضیه
Incremental	اضافه‌شونده
Incremental	افزایشی
Instance	نمونه
Intelligent Agent	عامل هوش‌مند
Intractable	غیرقابل پی‌گیری
Invariance	ناوردا
Lamarckian Evolution	تکامل لامارکی
Landmark	زمین‌نشان
Layered	لایه‌ای
Local	محلی
Lookup Table	جدول ارجاعی
McCallum	مک‌کالوم
Meme	مم
Meme Pool	مخزن مم
Memetic Algorithm	الگوریتم ممتیک
Memetic Algorithm	الگوریتم ممتیک
Memory-based	حافظه‌محور
Monolithic	یک‌پارچه
Monte Carlo	مونت‌کارلو
Moving Average	متوسط‌گیر لغزان
Multi Step	چندگام
Mutation	جهش
Mutually Exclusive	دو به دو مجزا
Neutral	خنثی
No Action	ناعمل، عمل پوچ
Non-Stationary	ناایستا
Objective	عینی
Observable	رویت‌پذیر
Partial Observable	به طور جزئی رویت‌پذیر
Partition	افراز
Perception	ادراک
Perturbation	اختلال
Planning	برنامه‌ریزی
Policy	سیاست
Policy-based	سیاست‌محور
Premature Convergence	هم‌گرایی زودهنگام
Primitive	پایه‌ای
Primitive Action	عمل پایه‌ای
Problem Space	فضای مساله
Progress Estimator	تخمین‌زن پیش‌رفت
Projection	تصویرکردن
Purely Parallel SubSumption Architecture (PPSSA)	معماری Subsumption کاملا موازی

Reactive	رآکتیو
Recursive	بازگشتی
Reinforcement Learning	یادگیری تقویتی
Reinforcement Signal	سیگنال تقویت
Representation	بازنمایی
Representation Space	فضای بازنمایی
Return	پاداش بازگشته
Reusability	قابلیت استفاده مجدد
Sample Complexity	پیچیدگی نمونه
Scalable	مقیاس‌پذیر
Scaled	مقیاس‌شده
Selfish	خودخواه
Shaped Reinforcement Signal	سیگنال تقویت شکل‌دهی‌شده
Simulated Annealing	انجماد شبیه‌سازی‌شده
Single Step	تک‌گام
Situated	در محیط قرار گرفته
Sonar	سونار
State Transition Probability	توزیع احتمال انتقال حالت
Stochastic Search	جستجوی تصادفی
Subgoal	زیرهدف
Subjective	ذهنی
Subroutine	زیربرنامه
Subtask	زیر–وظیفه
Supervised	با سرپرست
Suppress	سرکوب
Temporally Extended Action	اعمال گسترش–زمانی یافته
Tilt Angle	زاویه‌ی کج‌شدگی
Trial	تعداد آزمون‌ها
Value	ارزش
Wall Following	تعقیب دیوار

Abstract

The main goal of this research is the automatic design of behavior-based agents with hierarchical architecture. It is assumed that the designer can provide a reinforcement signal that reflects the behavior of the agent. Mathematically rigorous methods are provided to optimize this performance measure. I have divided the design problem to structure and behavior design sub-problems. Concurrent use of these methods can solve the whole design problem. In this research, I have provided learning-based and evolution-based methods for automating the design process.

Learning-based methods are offered for both behavior and structure design problems. To solve the structure learning problem, two different knowledge representations are introduced that by using them the agent's value is decomposed into simpler elements. Doing so, one can solve difficult issues such as credit assignment and value updating in this hierarchical architecture. A behavior learning method is provided that finds the suitable state-action mapping. By introducing a pseudo-action named NoAction, behaviors can cooperate with each other.

Co-evolution of behaviors is another method for behavior design. In contrast with the most other researches, behaviors are evolved separately and by using structure learning method, they are organized in the architecture. A new interpretation of cultural-inspired

method named Memetic Algorithm is offered that increases the performance of hybridization of evolution and learning.

The proposed methods are being tested in two different problems: multi-robot object lifting and an abstract problem. The results are competitive to human designed architectures.

In addition to the automatic design methods, a few other results are provided in this thesis. They are probabilistic analyzing of controlling probabilities of behaviors in our architecture and investigating the effect of reinforcement signal uncertainty on the agent's value function and its policy.

University of Tehran

Faculty of Engineering

Department of Electrical and Computer Engineering

Learning and Evolution in Hierarchical Behavior-based Systems

By: Amir massoud Farahmand

Advisor:
Majid Nili Ahmadabadi

Co-Advisors:
Caro Lucas - Babak N. Araabi

A thesis submitted to the Graduate Studies Office
in partial fulfillment of the requirements for
the degree of M.Sc. in
Electrical Engineering

August 2005