تابستان ۱۴۰۰، دوره ۶، شماره ۲، پیاپی ۲۱
DOR :20.1001.1.2538161.1400.6.2.6.6

دریافت مقاله: ۱۳۹۹/۰۲/۰۱
پذیرش مقاله: ۱۳۹۹/۰۹/۰۹

نوع مقاله: پژوهشی

طراحی سیستم دسته‌بند یادگیر برای حل مسئله ماز با استفاده از الگوریتم ژنتیک بهبودیافته

علیرضا فروزانی فرد
دانشجوی دکتری گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی، میبد، ایران
پست الکترونیکی: froo_kia1390@yahoo.com

کمال میرزائی*
عضو هیئت علمی گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی، میبد، ایران
پست الکترونیکی: k.mirzaie@maybodiau.ac.ir

چکیده

حل مسئله ماز یکی از مسائل کلاسیک در نظریه گراف است. مسئله ماز انواع مختلفی دارد که می‌توان به مواردی چون ماز اعداد، ماز پیچ‌وخم، ماز دایره‌ای، ماز تله و حلقه، ماز همیلتون و ماز بلوک اشاره کرد. سیستم‌های دسته‌بند یادگیر به‌طور موفقیت‌آمیز در مسائل مربوط به دسته‌بندی و داده‌کاوی، مسائل یادگیری تقویت، مسائل رگرسیون، یادگیری نقشه شناختی و حتی مسائل کنترل ربات مورداستفاده قرار گرفته است. در این مقاله، برای حل مسئله ماز از سیستم دسته‌بند یادگیر تک عامل استفاده‌شده است که با شناخت محیط و یادگیری، مسیری را برای رسیدن به هدف پیدا می‌کند. به‌منظور یادگیری بهتر از اعمال تشویق و تنبیه مناسب استفاده می‌شود و همچنین جهت پوشش کامل فضای مسئله و فرار از بهینه محلی، از الگوریتم ژنتیک بهبودیافته، استفاده شده که شامل یک عامل جهش بهبودیافته است. نتایج پیاده‌سازی رویکرد پیشنهادی، بیانگر کاهش زمان حل مسئله و افزایش دقت الگوریتم است.

واژه‌های کلیدی: مسئله ماز، سیستم دسته‌بند یادگیر، الگوریتم ژنتیک بهبودیافته، جهش بهبودیافته

۱- مقدمه

یک سیستم دسته‌بند[1]، ماشینی است که قوانین رشته‌های ساده را با نام دسته‌بند می‌آموزد. این دسته‌بندها عملکرد سیستم را در یک محیط دلخواه هدایت می‌کنند. یک سیستم دسته‌بند، نام خود را از توانایی یادگیری دسته‌بندی پیام‌ها از محیط به مجموعه‌های کلی، مشتق می‌کند و از بسیاری جهات شبیه به یک سیستم کنترل است. همانطور که یک سیستم کنترل از بازخورد برای کنترل یا تطبیق خروجی خود برای یک محیط استفاده می‌کند، یک سیستم دسته‌بند از بازخورد برای آموزش یا تطبیق دسته‌بندهای خود برای یک محیط، بهره می‌گیرد. این سیستم‌ها از ترکیب سیستم‌های خبره و الگوریتم‌های ژنتیکی، حاصل شده‌اند. این ترکیب بر نقطه ضعف اصلی سیستم‌های خبره یعنی وظیفه طولانی کشف و واردکردن قوانین غلبه کرده است.

1 - Classifier System

* نویسندهٔ مسئول

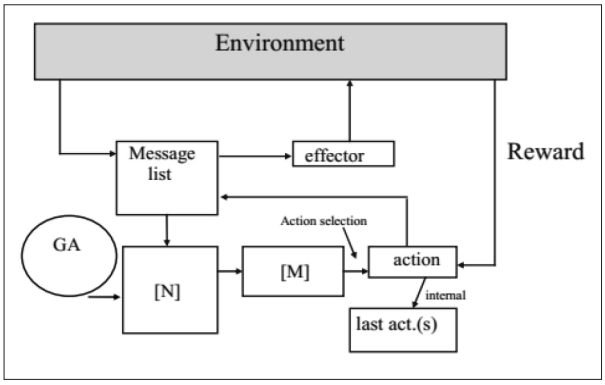
سیستم دسته‌بند با استفاده از یک الگوریتم ژنتیکی، قوانین مـورد نیاز برای انجـام یک وظیفـه را از محیط پیرامون خود می‌آموزد که بیانگر یادگیری سیستم دسته‌بند است. یادگیری ماشینی یکی از مباحث مهم در داده‌کاوی محسوب می‌شود [۳-۱].

سیستم دسـته‌بند یادگیر[۲] خانواده‌ای از الگوریتم‌های یادگیری ماشـینی هسـتند که معمولاً با رویکرد موردی طراحی می‌شـوند. به‌طورکلی، آن‌هـا را می‌توان با انجام وظایف تصمیم‌گیری متوالی با یک بازنمایی مبتنی بر قانون و با اسـتفاده از روش‌های محاسبه تکاملی مشخص‌کرد، اگرچـه برخـی از انـواع آن یادگیـری نظارت‌شـده را انجـام می‌دهنـد و برخی دیگر یادگیری بـدون نظارت را دنبال‌می‌کنند [٤].

ایده سیستم دسته‌بند یادگیر در سال ۱۹۷٦ توسط هلند مطابق شـکل ۱ معرفی شد و در سـال ۱۹۸٦ توسط وی به‌عنوان یک روش رمزگذاری شـده تکمیل گردید. سیستم هلنـد یک ورودی دودویی از محیط خـود دریافت می‌کند که درواقع ایـن ورودی دودویی، وضعیت فعلی محیط که می‌تواند هر سیستم کنترلی باشد را نشان می‌دهد. ورودی دودویـی در فضای حافظـه کاری داخلی در یک لیسـت پیام ذخیره می‌شـود. سپس سیستم پاسخی مناسب را بر اسـاس ورودی دریافتی تعیین می‌کند و بعد از جستجوی بین قوانین، عمل مشخص‌شده را انجام می‌دهد که معمولاً وضعیت محیط را تغییر می‌دهد. پس از آن، پاداش مناسب به رفتار رخ‌داده در محیط منتهی می‌شود که باعث تقویت یا تضعیف آن می‌گردد. مجموعه قوانین N شامل جمعیتی از قوانیـن مربوط بـه عمل یا به‌عبارت دیگر N دسـته‌بند است. شرط و عمل‌های قانون، رشته کاراکترهایی متشکل از الفبای سه‌گانه {0،1، #} است که در آن # حالت بی‌تفاوت را بیان‌می‌کند [۸-٥].

در حقیقت سیستم دسته‌بند یادگیر یک تکنیک یادگیری ماشـینی از ترکیب یادگیری تقویتی و محاسبات تکاملی و سایر اکتشافات برای تولید سیستم‌های سازگار است. این

2 - Learning Classifier System



شکل ۱: سیستم دسته‌بند هلند [۸].

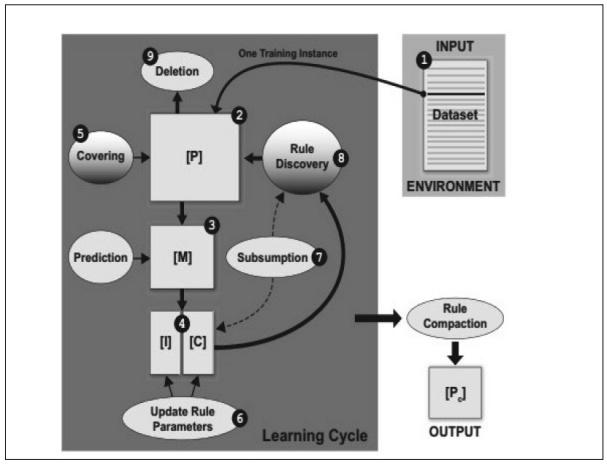
سیسـتم‌ها، مبتنی بر قانون هستند که معمولاً از قوانین در قالب سیسـتم تولید سـنتی اگر... آنگاه...[۳] شکل می‌گیرند. یادگیری در حقیقت از طریق دریافت پاداش اسـت. پاداش عـددی به قانون عمل‌کننده اختصاص می‌یابد. هسـته یک سیسـتم دسته‌بند یادگیر مجموعه‌ای از قوانین است که به آن جمعیت دسته‌بندها، گفته می‌شود. سیستم‌های دسته‌بند یادگیر از دو استعاره بیولوژیکی تکامل و یادگیری استفاده می‌کنند؛ تکامل و یادگیری. جایی که یادگیری مؤلفه تکاملی را هدایت می‌کند تا به سمت مجموعه‌های بهتری از قوانین حرکت کند. این مفاهیم به ترتیب توسط دو سازوکار تجسم می‌شوند. الگوریتم ژنتیک و یک سازوکار یادگیری مناسب. در چارچوب ادبیات سیسـتم دسـته‌بند یادگیر، محیط به سـادگی منبع داده‌های ورودی برای الگوریتم این سیستم اسـت. اطلاعات منتقل‌شـده از محیط فقط به دامنه مسئله مورد بررسـی محدود می‌شوند. مؤلفه‌های اصلی سیستم دسته‌بند یادگیر را می‌توان به ترتیب زیر تشریح کرد:

۱- جمعیت[٤] محدود دسته‌بند که نشان‌دهنده دانش فعلی سیستم است.

۲- مؤلفـه کارایی[٥] کـه تعامل بین محیـط و جمعیت دسته‌بند را تنظیم می‌کند.

۳- مؤلفـه تقویت‌کننده[٦] که مؤلفـه واگذاری اعتبار نیز نامیده می‌شود و پاداش دریافتی از محیط را به دسته‌بنده‌ها توزیع می‌کند.

3- If....Then....
4- Population
5- Performance
6- Reinforcement



شکل ۲: سیستم دسته‌بند یادگیر [۱۱]

٤- مؤلفه کشف[7] که از اپراتورهای مختلف برای کشف قوانین بهتر و بهبود قوانین موجود، استفاده می‌کند.

این مؤلفه‌ها یک چارچوب اساسی را نشان می‌دهند که بر اساس آن تعدادی از تغییرات جدید در الگوریتم سیستم دسته‌بند یادگیر ساخته شده است. شکل ۲ چگونگی تعامل سازوکارهای خاص این سیستم، برای این مؤلفه‌های اصلی نشان می‌دهد [۱۱-۹].

با وجود این‌که چهار مؤلفه اشاره شده در بالا چارچوب الگوریتم را در بردارد، ولیکن تنها دو سازوکار اصلی هستند که وظیفه هدایت سیستم را بر عهده می‌گیرند. این سازوکارها شامل مولفه‌های کشف و یادگیری می‌باشند که عموماً از طریق الگوریتم ژنتیک انجام می‌شود.

الف) کشف

کشف به معنای کشف قانون یا معرفی قوانینی است که در حال حاضر در جمعیت وجود ندارد. از ابتدای معرفی مفهوم سیستم دسته‌بند یادگیر، این کار تقریبا همیشه با استفاده از الگوریتم ژنتیک حاصل شده است. تغییر در ژن یا قانون به‌طور معمول توسط دو عملگر ژنتیکی ایجاد می‌شود. جهش[8] و ادغام[9]. عملگرهای ادغام با ترکیب زیرمجموعه‌های دو یا چند قوانین، قوانین جدیدی ایجاد می‌کنند. عملگرهای جهش به‌طور تصادفی یک عنصر در یک قانون را تغییر می‌دهد. عملگر انتخاب که باعث می‌شود اندامگان‌های بهتر برای تولید مثل، بیشتر انتخاب شوند و این انتخاب به میزان شایستگی بستگی دارد. عملکرد شایستگی، بهینه‌بودن یک قانون را تعیین می‌کند و باعث می‌شود آن قانون در برابر سایر قوانین دیگر در جمعیت رتبه‌بندی شود. در یک مسئله دسته‌بندی ساده، ممکن است از دقت دسته‌بندی به‌عنوان معیار شایستگی استفاده شود. اجرای یک الگوریتم ژنتیک برای تعدادی از تکرارها نیاز به حلقه‌های مختلف دارد. در ابتدا، کاربر باید بر اساس نیاز کاربر از تعدادی پارامتر مانند اندازه جمعیت و تعداد نسل‌ها، پیش تعریف کند. بعلاوه الگوریتم ژنتیک باید با جمعیتی از قوانینی که می‌تواند به‌طور تصادفی تولید شود، به‌طور گسترده‌ای پوشش دهد تا طیف وسیعی از راه‌حل‌های ممکن در فضای جستجو را تأمین کند [۱٤-۱۲].

ب) یادگیری

در زمینهٔ هوش مصنوعی، یادگیری را می‌توان بهبود عملکرد در برخی از محیط از طریق کسب دانش ناشی از تجربه در آن محیط تعریف‌کرد. مفهوم یادگیری از طریق تقویت یک سازوکار اساسی معماری سیستم دسته‌بند یادگیر است. غالباً اصطلاحات یادگیری، تقویت و واگذاری اعتبار به‌طور متناوب در ادبیات استفاده می‌شود. علاوه بر یک شرایط و عمل، هر دسته‌بند در جمعیت سیستم دسته‌بند یادگیر دارای یک یا چند مقدار پارامتر مرتبط با آن مثل شایستگی است. به‌روزرسانی تکراری این مقادیر پارامتر روند تقویت این سیستم را هدایت می‌کند. به‌طورکلی‌تر، به‌روزرسانی پارامترها پاداش دریافتی یا مجازات را به دسته‌بندهایی که پاسخگو هستند، توزیع می‌کند.

این سازوکار برای دو هدف مهم است:

۱- شناسایی دسته‌بندهایی که برای به دست‌آوردن پاداش‌های آینده مفید هستند

۲- تشویق به کشف قوانین بهتر

بسیاری از پیاده‌سازی‌های موجود سیستم دسته‌بند یادگیر از راهبردهای مختلف یادگیری استفاده می‌کنند. یکی از دلایل اصلی، این است که حوزه‌های مختلف مسئله

7- Discovery
8- Mutation
9- Crossover

خواستار سبک‌های مختلف یادگیری هستند. به‌عنوان مثال، یادگیری را می‌توان بر اســاس شیوه دریافت اطلاعات از این محیط دســته‌بندی کرد. یادگیری برون‌خط یا دسته‌ای دلالت بــر این دارد که تمــام نمونه‌های آمــوزش به‌طور هم‌زمان به زبان‌آموز ارائه می‌شود. نتیجه نهایی یک قانون واحد است که راه‌حلی را شکل می‌دهد که با توجه به زمان تغییر نمی‌کند. این نوع یادگیری اغلب مشــخصه مســائل داده‌کاوی است. از طرف دیگر، یادگیری برخط یا افزایشی دلالت بر این دارد که نمونه‌های آموزش به‌صورت یک‌بار به یادگیرنده ارائه می‌شــود، که نتیجه نهایی یک مجموعه قانون اســت که با اضافه‌کردن هر مشاهده اضافی به‌طور مداوم تغییر می‌کند. این نوع یادگیری ممکن است هیچ نقطه پایانی از پیش تعیین‌شده نداشته باشد، زیرا راه‌حل سیستم ممکن اســت به‌طور مداوم خود را بــا توجه به یک جریان مداوم از ورودی اصلاح کند [۱۵].

به‌عنــوان مثال، یــک روبات را در نظــر بگیرید که یک جریــان مــداوم از داده را در مورد محیطی که ســعی در پیمایــش آن دارد، دریافت می‌کند. با گذشــت زمان ممکن اســت لازم باشــد که حرکات خود را تطبیق دهد تا بتواند حول موانعی که هنوز با آن روبرو نشده است، مانور دهد. یادگیــری را نیز می‌توان با نــوع بازخوردی که در اختیار یادگیرنــده قرار می‌گیــرد، تفکیک کرد. در ایــن زمینه، دو ســبک یادگیری توسط سیستم دســته‌بند یادگیر استفاده شده اســت. یادگیری نظارت‌شــده و یادگیری تقویتی، که دومی اغلب به‌عنوان مترادف با سیستم دسته‌بند یادگیر در نظرگرفته‌می‌شود. یادگیری نظارت‌شده بر این نکته حاکی اســت که در هر نمونه آموزشی، یادگیری نه تنها اطلاعات مربوط به شرایط را در نظر دارد، بلکه با یک فرایند صحیح تهیه میگردد [۱۶].

درادامه، بخش ۲ به بیان مســئله پرداخته و مسئله ماز شرح داده می‌شود. در بخش ۳، به کارهای مرتبط و برخی از روش‌های حل مســئله ماز اشاره می‌شود. در بخش ٤، رویکرد پیشــنهادی ارائه و تشــریح می‌گردد. در بخش ٥، پیاده‌ســازی و ارزیابی روش پیشنهادی صورت می‌گیرد. در بخــش آخر، نتیجه‌گیــری و جهت‌گیری‌های آینده ارائه خواهد شد.

۲- بیان مسئله

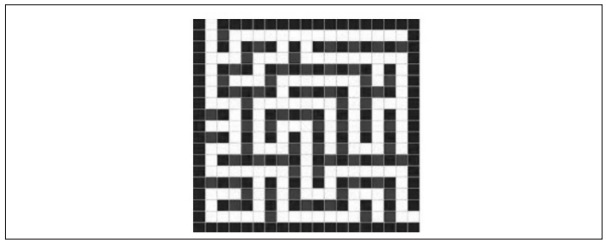
مســئله ماز عبارت اســت از یک محیط دوبعدی که از مجموعه‌ای سلول تشــکیل شده است و درواقع به‌صورت یک ماتریس دوبعدی تعریف میشود.

هر سلول از این مجموعه می‌تواند یکی از دو حالت کلی را بپذیرد. این دو حالت عبارت است از حالت آزاد و حالت مســدود. حالت مسدود موقعیتی اســت که ورود عامل به آن موقعیت مجاز نمی‌باشــد و در حقیقت از دیدگاه عامل به‌عنــوان یک مانع تلقی می‌گردد و برعکس آن، حالت آزاد مجموعه کلیه موقعیت‌هایی است که عامل مجاز به حرکت در آن‌ها می‌باشد. مفهوم دیگر قابل طرح در یک ماز، مفهوم نقطه شروع و نقطه پایان می‌باشد. هدف عامل آن است که با حرکت از نقطه شــروع و بدون ورود به نقاط مسدود به نقطه پایان برسد.

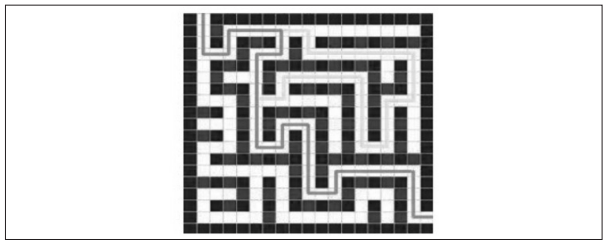
موش کوچک[10] یک روبات کوچک خودگردان است که برای تشــخیص یک ماز مونتاژ شده است. موضوع موش در مســئله ماز در ابتدا توسط کلود شانون در سال ۱۹۵۰ ارائه شــد و اولین ماشــین تشــخیص ماز را مونتاژ کرد. اولین رقابت موش کوچک در دهه ۱۹۷۰ آغاز شد. موش‌ها به‌گونه‌ای ساخته شــده‌اند که به تنهایی به‌وسیله ماز اجرا شوند و تصور می‌کنند که از گوشه‌ای از قبل تعیین‌شده از ماز متمرکز شده‌اند. موش‌ها از طریق الگوریتم و پیکربندی خود می‌توانند ظرفیت نگه‌داشتن حافظه از ماز، موقعیت آن در داخل ماز، ثبت اطلاعات ماز و پیشبرد یک دوره را برای اجرا در کوتاه‌ترین زمان قابل‌تصور داشته باشند [۱۷].

ماز یک جدول تور اســت که از طریق آن یک حل‌کننده بایــد راه‌حلی پیدا کند. این شــبکه‌ای از مســیرها و مانع‌ها اســت که به‌عنوان یک جدول طراحی شده است که از طریق آن می‌بایســت راهی را به ســمت هدف پیدا کــرد. یک ماز

10- Micromouse



شکل ۳: ماز ساده غیر ارتباطی [۱۷]



شکل ۴: ماز چند ارتباطی [۱۷]

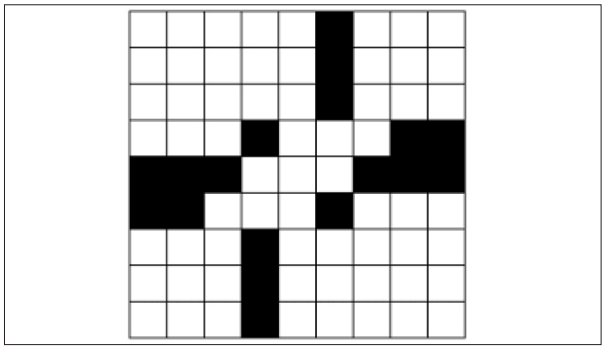
از اندازه‌هـــای مختلف ماننـــد ۸*۸، ۱۶*۱۶، ۳۲*۳۲ و غیره تشکیل می‌شـــود. حل مسئله ماز، پیداکردن مسیری از ابتدا تا انتها اســـت. برخی از روش‌های حـــل ماز طوری طراحی شـــده‌اند که عامل بدون اطلاعات قبلی در مورد ماز استفاده می‌شـــود، اگرچه برخی دیگر قرار اســـت توسط یک برنامه شخصی یا ماشینی مورداستفاده قرار بگیرند که بتواند کل ماز را فوراً مشاهده کند. مسئله ماز را می‌توان به‌طور کلی به دو نوع ساده غیر ارتباطی و چند ارتباطی تقسیم کرد [۱۷].

الف) ماز ساده غیر ارتباطی

این نوع ماز همانطور که در شـــکل ۳ نشان داده شده اســـت تنها دارای یک مســـیر به ســـمت هدف است. سایر مسیرها هرگز به هدف مرتبط نمی‌شوند، بنابراین هر راهی که انتخاب می‌شـــود یـــا به راه‌های اضافی یا به بن‌بســـت می‌رســـد. راه‌حل برای یک مـــاز تک‌ارتباطـــی را می‌توان همیشـــه با دنبال‌کردن قانون دســـت چپ یا قانون دســـت راست پیدا کرد.

ب) ماز چند ارتباطی

این نوع ماز دارای یک یا چند مســـیر حلقه‌ای مرتبط با ســـایر قسمت‌های مختلف مانند شکل ٤ است. یک ماز چند ارتباطی معمولی برنامه‌ریزی شده، دشوارتر از ماز اساساً پیوسته است، زیرا عامل‌ها زمان زیادی را صرف می‌کنند تا در حلقه‌ها بچرخند.



شکل ۵: نمونه جدول بلوک ساده

مسئله ماز انواع مختلفی دارد که می‌توان مواردی چون ماز اعـــداد، ماز پیچ‌وخم، ماز دایره‌ای، ماز تله و حلقه، ماز همیلتـــون و ماز بلوک نـــام برد. در این مقالـــه، از یک ماز بلوکی ساده مانند شـــکل ٥ که به صورت شطرنجی است و از مجموعه‌ای از بلوک‌ها و راه‌ها تشـــکیل شـــده است، استفاده می‌شود.

در این مقاله، فرض‌می‌شود که عامل در ابتدا با استفاده از بررسی خانه‌های اطراف خود در جدول، وضعیت محیط را تشخیص‌داده و ارزیابی می‌کند.

۱= بلوک یا مسدود بودن

۰ = راه باز یا موجود بودن مسیر

سپس با رصدنمودن محیط توسط عامل، یک وضعیت[11] اولیه مشـــخص می‌شـــود و عامل با توجه به این وضعیت و دســـتورهای موجود برای این حالت، اقـــدام به اجرای[12] تصمیم می‌کند. این همان اگر... آنگاه... تعیین شده است که عامـــل برابر آن اقدام به تصمیم‌گیری و حرکت می‌کند. این روند ادامه پیدا می‌کند تا عامل بتواند به هدف برسد. در این مســـئله، باید عامل یاد بگیرید تا بتواند با توجه به بررسی محیط و تشـــخیص شـــرایط و وضعیت موجـــود، بهترین تصمیم را در مســـیر حرکت اتخاذ نمایـــد. این یادگیری با انجام تشویق و تنبیه با توجه به حرکت عامل در مسیرهای موجـــود صـــورت می‌گیرد تا بـــا تغییر و اصـــلاح برخی تصمیمات بتواند وضعیت و مسیر حرکت را به سمت هدف بهبود دهد.

11- Condition
12- Action

۳- کارهای مرتبط

حل مسئله ماز یکی از مسائل کلاسیک در زمینه نظریه گراف است. حل ماز، یافتن کوتاه‌ترین مسیر بین منبع و نقطه هدف در یک هزارتوی خاص است که می‌تواند توسط الگوریتم‌های جستجوی پهنایی[13] [۱۸] یا جستجوی ژرفایی[14] [۱۸] حل شود. اما پیچیدگی محاسباتی و پیچیدگی زمانی این الگوریتم‌های جستجو، به‌صورت نمایی با افزایش اندازه ماز افزایش می‌یابد. بسیاری از الگوریتم‌های الهامگرفته از زیست‌شناختی برای یافتن مسیر بهینه در مازها، مانند الگوریتم مورچگان [۲۰،۱۹]، الگوریتم ژنتیکی [۲۲،۲۱]، الگوریتم موازی ممتیکی جستجوی ممنوعه [۲۳] و دیگران استفاده شده است. تکنیک‌های بسیاری برای حل کردن مسئله ماز وجود دارد که به برخی از آن‌ها به شرح زیر اشاره می‌گردد:

- wall follower: دنبال‌کننده دیواری که اولین، آسان‌ترین و بهترین تکنیک شناخته‌شده برای حل مسئله ماز است و با نام مستعار قانون دست چپ یا قانون دست راست اقدام می‌نماید. عامل از نقطه هدف دیواره‌ها را دنبال می‌کند و در صورت مواجه‌شدن با بن‌بست به سمت چپ یا راست می‌رود تا به هدف برسد. در این تکنیک اگر تمام تقسیم‌کننده‌ها در ماز به هم وصل باشند، دستیابی به جواب حتمی است ولی اگر دیوارها به هم وصل نباشند، این روش تضمین نمی‌کند که به هدف برسد [۲۴].
- lee's algorithm: این تکنیک یک برنامه از الگوریتم دایکسترا[15] است. در این تکنیک از روش جستجوی اول سطح برای حل مسئله ماز استفاده می‌شود که در دو فاز کار می‌کند. فاز پرکردن یا انتخاب سلول‌ها و مرحلهٔ بازپس‌گیری یا بازگشت به عقب. در مرحله اول شروع به پرکردن سلول‌های مجاور نقطه شروع می‌گردد و مقدار یک برای آن تعیین می‌شود. برای سلول‌های همسایه بعدی این مقادیر که فاصله هر سلول از ابتدا است با عدد دو علامت‌گذاری می‌شود. این کار تکرار می‌شود تا به نقطه

13- BFS
14- DFS
15- Dijkstra

جدول ۱: ارزیابی برخی تکنیک‌های حل مسئله ماز [۱۷]

پارامتر	wall follower	lee's algorithm	flood fill algorithm
قاعده و روش	قانون دست راست / چپ	جستجوی اول سطح	الگوریتم بلمن فورد
کارایی	پایین	متوسط	بالا
پیچیدگی فضایی	متوسط	بالا	بالا
پیچیدگی زمانی	متوسط	بالا	بالا

پایانی برسد. در مرحله بازگشت به عقب اگر انتهای خط یا مرحله I وارد شود، مسیر خود را با حرکت به سمت سلول به ارزش I1- دنبال می‌کند. این کار را مجدداً مورد استفاده قرار می‌دهد تا به مرحله آغازین S برسد [۱۷].

- flood fill algorithm: می‌توان از الگوریتم پرکردن سیل که یک برنامه از الگوریتم بلمن فورد است برای حل مسئله ماز استفاده کرد. این تکنیک بر روی مفهوم آب کار می‌کند که همیشه از ارتفاعی بالاتر به سمت پایین جریان می‌یابد. این مفهوم را با اختصاص یک مقدار به هر سلول در ماز که نشان می‌دهد سلول از مرکز تا چه فاصله قرار دارد، اعمال می‌کند. وقتی ماز به‌آرامی از آب پر شود، کوتاه‌ترین مسیر با اولین قطره آب برای رسیدن به مرکز طی می‌شود [۲۶،۲۵]. در جدول ۱، ارزیابی برخی تکنیک‌های حل مسئله ماز، آمده است.

مقاله‌ها و تحقیقات مناسب و شایسته‌ای به‌منظور بهبود روش‌های جدول۱ ارائه گردیده است که بعضاً به صورت ترکیبی از روش‌های کلاسیک و روش‌های هوشمند استفاده کرده‌اند. در ادامه به برخی از آن‌ها اشاره می‌گردد.

اسواتی و همکاران بررسی‌کردند که اگر هیچ محدودیت زمانی و سخت‌افزاری نباشد می‌توان از الگوریتم دایکسترا به‌طور مؤثری در حل مسئله ماز استفاده کرد، اما اگر این محدودیت‌ها هر دو باشند، الگوریتم پر کردن سیل از سایرین برتر خواهد بود [۲۷].

ژانگ و همکاران یک الگوریتم بهبود یافته مبتنی بر مدل ریاضی موجود با الهام از یک اندامگان آمبوئید برای حل مسئله ماز پیشنهاد دادهاند. سازوکار بازخورد مثبت در مدل ریاضی که منجر به هدایت بیشتر و منجر به شار بیشتر میشود و این به نوبه خود باعث افزایش هدایت میگردد. علاوه بر این، برخی از قوانین فازی ایجادشده از آزمایشها برای بهبود عملکرد با کاهش پیچیدگی محاسبه و سرعت بخشیدن بهسرعت همگرایی اعمال میشود [۲۸].

در سال ۲۰۱۰، دانگ و همکاران یک الگوریتم حل مسئله ماز با روبات استاندارد IEEE ارائه کردند. با توجه به وضعیت واقعی ماز جستجو ربات، این الگوریتم، الگوریتم پر کردن سیل را در حل مسئله ماز بهبود داده است [۲۹].

سلمان پور و همکاران از الگوریتم قطره آب هوشمند[16] برای حل برنامهریزی مسیر روبات برای نمودارهای کوچک استفاده کردند. برنامهریزی مسیر روبات مشابه حل یک ماز با چگالی کم در این گزارش است. آنها پیشنهاد کردند سطح دوم الگوریتم قطره آب هوشمند را در مسیرهای فرعی بهترین راهحل تکرار که باعث بهبود کیفیت راهحل میشود را اجرا کنند [۳۰].

جان و همکاران دو الگوریتم را برای بهبود حل مارپیچها با تراکمهای مختلف مقایسه کردهاند. نتایج در این گزارش برای اعلام بهتربودن قطعی یک الگوریتم از دیگری کافی نیست. برای مارپیچهای با تراکم و اندازه کم، الگوریتم قطره آب هوشمند راهحلهای بهتری ارائه کرده است و از میزان موفقیت بالاتری برخوردار می باشد. برای مارپیچهای بزرگتر با تراکم متوسط و بالا، الگوریتم ژنتیک در کیفیت راهحل از الگوریتم قطره آب هوشمند بهتر است [۳۱].

انتیناز و همکاران یک روش حل ماز مبتنی بر سختافزار را از طریق یک مدل الکترونیکی سازوکار حرکت داخلی نوسان پلاسمودیوم ارائه دادهاند. کارایی و کلی بودن مدار الکترونیکی پیشنهادی از طریق شبیهسازیهای سطح فضا در مقایسه با دادههای دو آزمایش بیولوژیکی مختلف، یعنی تقویت لولههای پروتوپلاسمی فیسارووم[17] در کوتاهترین مسیر و رشد شیمیایی- تاکتیکی قالب لجن تأیید شد. مدار پیشنهادی در منطقه حداکثر سازگاری (یادگیری) عمل میشود، بنابراین هر ماژول بسته به ویژگیهای شکل موج ورودی عملکرد خود را تنظیم میکند، درحالیکه گسترش شبکه با آموزش تدریجی ماژولهای بههمپیوسته مطابقت دارد. معرفی توانایی‌های صوتی ذاتی به مدار بیولوژیکی معادل منجر به چندین راهحل ماز (ازجمله بهینه) ماز شد که به رفتار اندامگانهای واقعی غیرقطعی نزدیکتر است [۳۲].

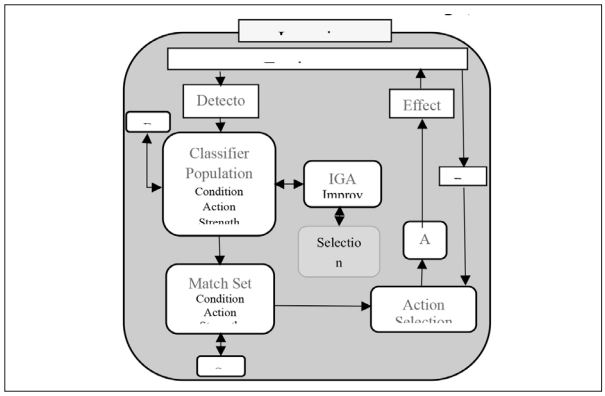
ناواف و همکاران از الگوریتم جستجوی A^* برای یافتن کوتاهترین مسیر بین مبدأ و مقصد روی تصویر که نشان‌دهنده نقشه یا ماز است استفاده کردهاند. با توجه به اینکه A^* یک الگوریتم جستجوی عمومی است، میتواند برای یافتن راهحل برای چندین مسئله مورد استفاده قرار گیرد و اساساً مسیریابی یکی از آنهاست. این الگوریتم ویژگی جستجوی یکنواخت و جستجوی اکتشافی را باهم جمع میکند. برای مسیریابی، الگوریتم A^* دوباره و دوباره بیشترین مکان کشف نشده را که دیده است، بررسی میکند [۳۳].

رویکرد پیشنهادی

در این مقاله، از روش سیستم دستهبند یادگیر و الگوریتم ژنتیک بهبودیافته استفاده میشود که توانسته نتیجه قابل قبولی را ارائه دهد. در این روش یک مدل پیشنهادی مطابق شکل ٦ ارائه گردیده که با استفاده از رویکرد مربوطه نسبت به حرکت عامل ناآگاه در محیط ماز پیشروی میکند. پس از حرکت عامل با توجه به بازخورد از محیط نسبت به یادگیری و بهبود گامهای بعدی اقدام مینماید.

در ابتدا عامل نسبت به بررسی وضعیت محیط میپردازد که همان سلولهای اطراف آن در جدول ماز است. این وضعیت با بیتهای صفر و یک که به ترتیب نشاندهنده باز یا بسته بودن مسیر است، مشخص میشود. اگر در جدول عامل به عدد صفر برخورد کند، یعنی میتواند به این سلول وارد شود؛ در غیر این صورت باید جهت دیگری را انتخاب نماید. عامل با به دستآوردن اطلاعات سلولهای همجوار خود تصمیم میگیرد که چه

16- IWD
17- Physarum



شکل ۶: مدل پیشنهادی سیستم دسته‌بند یادگیر با الگوریتم ژنتیک بهبودیافته

اقدامی را انجام دهد. این همان اگر... آنگاه...است. پس از دریافت اطلاعات از وضعیت سلول‌های هم‌جوار، یک عدد چهار بیتی به دست می‌آید. مثلاً عدد ۰۰۱۰ که نشان می‌دهد سه مسیر باز و یک مسیر مسدود است.

بخش Classifier شامل یک جمعیت اولیه تصادفی است که از سه قسمت Action، Condition و Strength تشکیل شده است. Condition شامل چهار بیت است که حالات فرضی محیط را نشان می‌دهند. Action شامل دو بیت تصادفی که نحوه حرکت عامل را با توجه با شرایط موجود ارائه می‌دهد. Strength نشان‌دهنده قدرت یا استحکام قانون است که در ابتدا برای تمامی آن‌ها مقدار پیش‌فرض ۱ منظور گردیده است و با اجرای آن‌ها توسط عامل و حرکت صحیح در مسیرهای موجود به سمت هدف، این مقادیر با تشویق و تنبیه‌های مربوطه تغییر می‌کنند. پس‌از این‌که وضعیت محیط توسط عامل بررسی گردید، این وضعیت با جمعیت اولیه تصادفی Classifier مطابقت داده می‌شود. پس از مطابقت موقعیت عامل با شرایط فرضی، یک مجموعه‌ای از جمعیت که دارای تطابق شرایط می‌باشند، احصاء شده و داخل Match Set ذخیره می‌گردند. ممکن است این مجموعه دارای قوانین متفاوتی برای این شرایط تطبیق داده شده باشد. برای اولین بار همه این شرایط دارای Strength یکسانی می‌باشند و انتخاب آن‌ها برای اجرا توسط عامل به‌صورت تصادفی است. ولی در گام‌ها و تکرارهای بعدی، از بین قوانین موجود در مجموعه، قوانینی که دارای Strength بهتری می‌باشند انتخاب خواهند شد. این احتمال وجود دارد که هیچ تطابقی در بین جمعیت اولیه تصادفی در Classifier و وضعیت محیط ارزیابی‌شده توسط عامل وجود نداشته باشد. بدین منظور به دلیل ممانعت از اختلال در روند اجرای الگوریتم، از Covering جهت پوشش این خلأ استفاده می‌شود و یک قانون پیش‌فرض مطابق با وضعیت محیط تعیین و داخل Match Set ذخیره می‌گردد. هر زمان که هیچ قانونی برای حرکت عامل وجود نداشته باشد این اقدام انجام خواهد شد. با توجه به این‌که جمعیت اولیه تصادفی در Classifier باید دارای تعداد معینی باشد، در هر تکرار از بین جمعیت موجود، قوانینی که دارای Strength ضعیف‌تر می‌باشند از مجموعه حذف تا حداکثر مجموعه برای ادامه کار حفظ گردد. یک مقدار بیشینه برای این روند تعیین تا در صورت افزایش جمعیت مورد نظر، نسبت به حذف آن‌ها اقدام گردد.

در مرحله ادغام، الگوریتم ژنتیک با استفاده از ترکیب کروموزوم‌ها با یکدیگر سعی دارد ژن‌های خوب هر کروموزوم را به فرزند منتقل‌کند تا به این وسیله بتواند در بدست‌آوردن نتیجه کلی مسئله به سمت جواب بهینه نزدیک گردد. در مسئله ماز، این کروموزوم‌های مناسب همان جمعیت موجود در Classifier است که دارای Strength قوی‌تری نسبت به بقیه هستند. دو عدد از قوانین در جمعیت که دارای وضعیت بهتری می‌باشد انتخاب و در بخشی از بیت‌ها که به‌صورت تصادفی انتخاب می‌شود شکسته و با یکدیگر ترکیب می‌شوند. دو قانون جدید یا همان فرزندان از ژن‌های قوی ایجاد می‌شوند. این قوانین در جمعیت جایگزین قوانین با Strength ضعیف‌تر، می‌شوند.

در گام جهش از الگوریتم ژنتیک، عملگر جهش سعی می‌کند تا با ایجاد تغییراتی در ژن‌های کروموزوم، از نقطه‌ای از فضای جستجو به نقطه‌ای دیگر دسترسی پیدا کند. در این خصوص این رویکرد می‌تواند نقش مؤثری را در فرار از بهینه محلی داشته باشد. در الگوریتم ژنتیک برای ایجاد مقادیر ژنی، از روش تولید تصادفی اعداد استفاده

می‌شـــود. اگرچه الگوریتم ژنتیک در طـــول فرایند تکاملی پیـــش می‌رود، اما تنوع جمعیـــت به‌تدریج کاهش می‌یابد و باعث کاهش راندمان جســـتجوی عملگر ترکیب و درنتیجه پدیده زودرس می‌شود. در این حالت، تنوع جمعیت باید از طریق عملگر جهش حفظ شود [۳۵,۳٤]. بنابراین یک عملگر جهش بهبودیافته که در کل منجر به بهبود الگوریتم ژنتیک می‌گردد و می‌تواند به‌طور مؤثـــر بر پدیده فرآیند تکاملی زودرس غلبه کند، مطابق رابطه (۱) پیشنهاد می‌گردد:

$$Div(k) = \sqrt{\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{x_j^i(k)-x_j^b(k)}{bj-aj}\right)^2} \qquad (۱)$$

$x_j^i(k)$ نشـــان‌دهنده عنصر jام از iامین عضو در نسل k می‌باشـــد. $x_j^b(k)$ نشان‌دهنده jام از بهترین عضو در نسل k می‌باشد. bj و aj نشان‌دهنده بالاترین و پایین‌ترین مقدار j امین متغیر اســـت. n نشان‌دهنده تعداد عنصر در یک عضو است. m نشـــان‌دهنده تعداد عضو (سایز جمعیت) است. در این مقاله، از الگوریتم ژنتیک بهبودیافته اســـتفاده شـــده که در قســـمت جهش آن تغییراتی ایجاد شده است. دیگر از یک جهش ساده که به‌صورت تصادفی در عناصر یک قانون رخ می‌داده استفاده نمی‌شود. بلکه این جهش هدفمند بوده و بین عناصر ضعیف‌ترین قانون و قوی‌ترین قانون اتفاق می‌افتد.

۴- پیاده‌سازی و ارزیابی

در مســـئله ماز، یک عامل می‌خواهد محیط را بشناسد و با یادگیری، مســـیری را برای رسیدن به هدف پیدا کند. به‌منظور یادگیری بهتر از سیستم تشویق و تنبیه استفاده می‌گردد. در این راستا در جهت پوشش کامل فضای مسئله و فرار از بهینه محلی، از الگوریتم ژنتیک بهبودیافته استفاده می‌گردد. مراحل پیاده‌سازی در ادامه آمده است.

- **تعریف محیط ماز**

در ابتـــدا محیـــط لازم بـــرای حضور یک عامـــل باید طراحی‌گـــردد که تهیـــه و ایجاد یک جدول اســـت. جدولی به‌انـــدازه دلخواه به‌صورت یـــک ماتریس تعریف می‌گردد که بیت‌های صفر نشـــان‌دهنده مســـیر باز و بیت‌های یک نشان‌دهنده مسیر بسته یا بن‌بست می‌باشند. در این مقاله، جدول موردنظر به ابعاد ۲۰*۲۰ در نظر گرفته شده است.

پس‌از این‌که جدول طراحی شـــد، باید ســـلولی را برای شـــروع یا موقعیت ابتدایی عامل مشخص کرد که با متغیر start_position در نظر گرفته شـــده است و همچنین باید سلول دیگری به‌عنوان هدف یا مقصد عامل تعیین گردد که آن‌هم با متغیر gold_position و به شرح زیر در نظر گرفته شده است. عامل از مبدأ به سمت مقصد حرکت خواهد کرد.

```
maze=[.....;];
start_position=[3,4];
gold_position=[19,19];
```

موقعیـــت فعلـــی عامـــل current_position به‌عنوان وضعیت شـــروع در نظر گرفته می‌شـــود. عامل باید برای حرکت آماده شود. برای این امر عامل باید محیط یا همان سلول‌های اطراف خود را بشناسد. لیکن محیط و سلول‌های هم‌جوار عامل که همان جهت‌های بالا– پایین– چپ و راست می‌باشـــد به‌وســـیله متغیرهای position_up - position_ down position_left - position_rigth به‌صورت کامل به عامل معرفی می‌شوند.

- **تعریف جمعیت تصادفی اولیه**

یک جمعیت تصادفـــی اولیه به‌منظـــور ایجاد مجموعه قوانین تصادفـــی و خام، به‌اندازه دلخـــواه در یک ماتریس تعریف می‌گردد. این ماتریس دارای ۷ بیت است. ٤ بیت اول نشان‌دهنده وضعیت یا همان شرایط می‌باشند، ۲ بیت بعدی نشان‌دهنده اقدام یا عملکرد برای هر شرط و بیت آخر همان Strength که نشـــان‌دهنده قدرت یا میزان توان و اهمیت هر قانون می‌باشد. در این مقاله، جمعیت تصادفی اولیه با متغیر pop.max به تعداد ۱۰۰۰ در نظر گرفته شده است.

- **تعریف الگوریتم ژنتیک بهبودیافته**

در سیستم دســـته‌بند یادگیر، به‌صورت پیش‌فرض از الگوریتـــم ژنتیک به‌منظور بهینه‌ســـازی جمعیت موجود و حذف قوانین ضعیف‌تر اســـتفاده شده اســـت. از انتخاب[۱۸] به‌منظور تعیین قوانین قوی‌تر و ضعیف‌تر استفاده می‌شود.

18- Selection

از ادغام در راستای جابجا نمودن برخی از اعضاء قوانین ضعیف‌تر به‌صورت تصادفی استفاده می‌شود و از جهش جهت جهش برخی از بیت‌های قوانین ضعیف‌تر و فرار از بهینه‌های محلی استفاده می‌شود. در این مقاله، از الگوریتم ژنتیک بهبودیافتهای استفاده شده است که در بخش جهش آن تغییراتی ایجاد شده است. دیگر از یک جهش ساده که به‌صورت تصادفی در عناصر یک قانون رخ می‌داده استفاده نمی‌شود. بلکه این جهش هدفمند بوده و بین عناصر ضعیف‌ترین قانون و قوی‌ترین قانون اتفاق می‌افتد.

- **تطابق دادن وضعیت محیط با شرایط**

در این مرحله، باید تطبیق وضعیت محیط که توسط عامل تشخیص داده شده است با شرایط موجود یا مجموعه قوانین تصادفی اولیه صورت پذیرد. پس‌از این‌که ارزیابی وضعیت عامل با مجموعه قوانین صورت گرفت، قوانین تطابق یافته در مجموعه match set قرار می‌گیرند. با توجه به تصادفی بودن مجموعه، احتمال تکراری بودن برخی قوانین وجود دارد که در ابتدا در مجموعه match set، قوانین تکراری با استفاده از دستور unique حذف شده و بقیه قوانین بر اساس Strength مرتب می‌شود تا بهترین قوانین تطبیق‌یافته در دسترس قرار گیرند.

- **پوشش شرایط در صورت عدم تطابق**

با توجه به تصادفی بودن جمعیت قوانین اولیه، در صورتی که هیچ‌کدام از شرایط با وضعیت محیط تطابق نداشته باشد، الگوریتم در این مرحله دچار سردرگمی شده و قفل می‌شود. در صورت نداشتن هیچ قانون تطبیق یافته با وضعیت محیط برای عامل، به‌منظور پیشگیری از این حالت، یک قانون تصادفی و تطابق داده‌شده با محیط به‌عنوان پیش‌فرض در Covering ارائه می‌شود.

- **حرکت عامل**

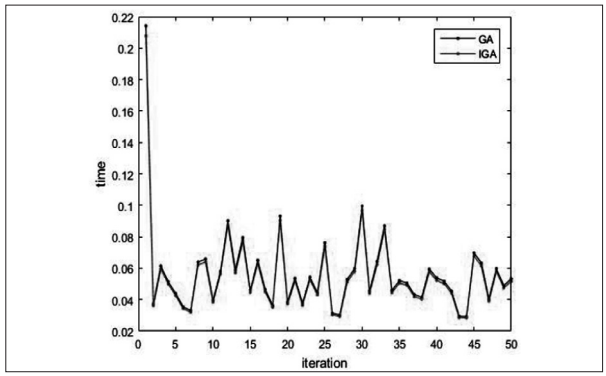
عامل پس‌ازاین‌که با مجموعه قوانین موجود از قسمت وضعیت مطابقت پیدا کرد، برابر عمل تعیین‌شده یکی از حالات را انتخاب و حرکت می‌کند. حرکت عامل دریکی از جهت‌های شمال، جنوب، شرق یا غرب و با توجه به پایش انجام‌شده خواهد بود.

- **اعمال تشویق و تنبیه**

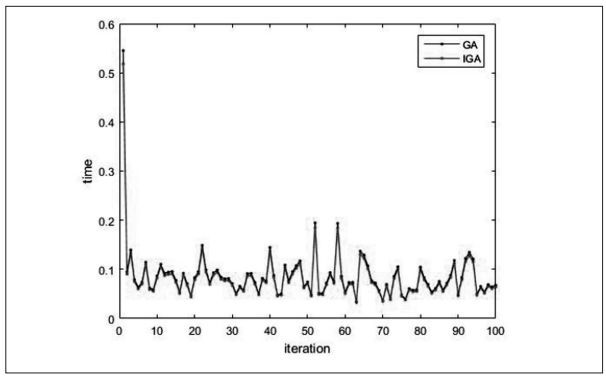
به‌منظور یادگیری الگوریتم و ایجاد قوانین بهتر باقدرت و تطابق بیشتر، باید اقداماتی اساسی در این راستا صورت گیرد. یکی از این روش‌ها، اعمال تشویق و تنبیه برای عامل با توجه به حرکت انجام‌شده می‌باشد. پس‌از این‌که از مجموعه قوانین موجود مطابقت لازم با وضعیت عامل در محیط صورت گرفت و بهترین قوانین متناظر با شرایط موجود انتخاب شدند، عامل شروع به حرکت در راستای این قوانین می‌کند. با توجه به این‌که عامل در ابتدا با ناآگاهی کامل در محیط ظاهر می‌شود، چنانچه این روند ادامه پیداکرده و عامل بدون هیچ تغییری فقط در محیط و با توجه به قوانین ساده حرکت کند، این امر به گیرافتادن در بن‌بست‌ها و تکرار مسیرهای احتمالی یا قفل شدن را در پی خواهد داشت یا شاید هم به هدف برسد. ولی این روش اصلاً ایده‌آل نبوده و نتیجه بخش نمی‌باشد. به‌منظور جلوگیری از بروز چنین مواردی باید عامل به‌صورت هدفمند حرکت نماید و در این مقاله، از اعمال تشویق و تنبیه برای عامل درحرکت خود به سمت هدف استفاده می‌شود و مجموعه قوانین که دارای Strength می‌باشند با توجه به این پیشامدها مورد تغییر و تأثیر قرار می‌گیرند. نمونه کد تشویق و تنبیه صورت گرفته به شرح زیر می‌باشد. در این مقاله، برای این‌که قرار گرفتن عامل در مسیر صحیح بیشتر مورد توجه قرار گیرد، حرکت درست عامل با تشویق ۲+ و حرکت نادرست عامل با ۱- در نظر گرفته شده است. به‌منظور ارزیابی و شبیه‌سازی این الگوریتم در حل مسئله ماز از نرم‌افزار متلب استفاده گردید و نتایج زیر پس از اجرا حاصل شد. پیکربندی سیستم مورد استفاده در شبیه‌سازی به شرح زیر می‌باشد:

- Processor: Intel(R) Core(TM) i5-3337U @ 1.80GHz 1.80GHz
- RAM: 4/00GB
- System type: windows 10 – 64-bit operating system

با استفاده از سیستم با مشخصات فوق، یک ارزیابی به‌منظور بررسی و تحلیل نتایج حاصل از استفاده سیستم



شکل ۷: زمان حل مسئله با ۵۰ بار تکرار



شکل ۸: زمان حل مسئله با ۱۰۰ بار تکرار

دسته‌بند یادگیر با الگوریتم ژنتیک بهبودیافته در حل مسئله ماز صورت گرفت. در این راستا از یک جدول ماز به ابعاد ۲۰*۲۰ استفاده شد. پس از اجرای الگوریتم، نتیجه اجرای مسئله و کاهش زمان با ۵۰ بار تکرار در شکل ۷ و ۱۰۰ بار تکرار در شکل ۸ نمایش داده شده است. محور افقی نشان‌دهنده تعداد تکرار و محور عمودی نشان‌دهنده زمان رسیدن عامل به هدف می‌باشد. در این نمودارها به‌صورت واضح مشخص است که در هر بار تکرار با توجه به یادگیری عامل مدت زمان رسیدن به هدف کمتر می‌گردد. ولی با در نظر داشتن چندین مسیر موجود، این زمان در بعضی از مواقع نیز زیادتر شده است. همچنین در مقایسه الگوریتم ژنتیک با الگوریتم ژنتیک بهبودیافته می‌توان کاهش زمان حل مسئله و بهبود وضعیت را دررسیدن به هدف مشاهده نمود. در برخی از تکرارها هم این عملکرد باهم برابر می‌باشد.

در ادامه به‌منظور ارزیابی و بررسی نتایج حاصل از اجرای الگوریتم فوق که با توجه به تعداد سلول‌های انتخاب شده توسط عامل در مسیر هدف و یادگیری با اعمال تشویق و تنبیه میباشد می‌توان دقت انجام فرایند را بررسی نمود.

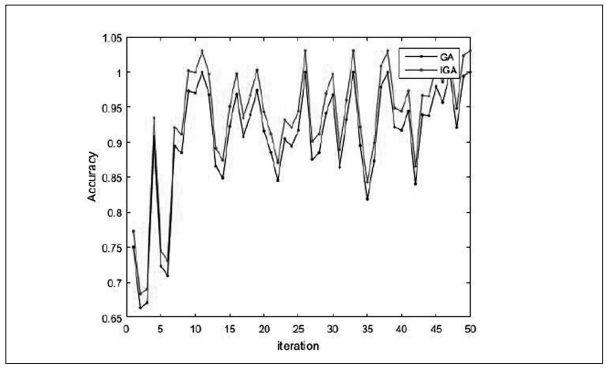
در این زمینه از متغیر correct_move به‌منظور شمارش تعداد سلول‌های انتخاب درست و در مسیر صحیح و از متغیر error_move به‌منظور شمارش تعداد سلول‌های انتخاب نادرست و در مسیر اشتباه استفاده گردیده است. در ادامه دقت انجام الگوریتم با به دست آوردن Accuracy مطابق رابطه(۲) محاسبه می‌شود.

در شکل‌های ۹ و ۱۰ نتایج به‌دست‌آمده در تعداد تکرار ۵۰ و ۱۰۰ مرحله نشان داده شده است و مشخص گردیده که با تکرارهای صورت گرفته و یادگیری الگوریتم، دقت محاسبات افزایش پیدا کرده است. هرچند که با توجه به وجود مسیرهای متفاوت و حتی وجود بن‌بست، احتمال بدتر شدن دقت نیز وجود دارد که این امر در برخی از قسمت‌های نمودار به‌صورت واضح در تکرارهای صورت گرفته مشخص می‌باشد. در این قسمت نیز بهتر شدن دقت در الگوریتم IGA نسبت به GA مشهود می‌باشد.

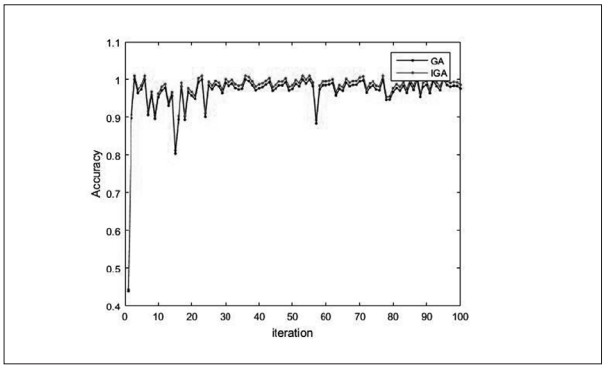
$$\text{Accuracy} = \frac{\text{correct\_move}}{(\text{correct move+error move})} \qquad (۲)$$

در انتها، بهبود نتایج حاصله از اجرای الگوریتم ژنتیک بهبودیافته (IGA) در روش سیستم دسته‌بند یادگیر را در مقایسه با الگوریتم ژنتیک ساده مورد ارزیابی قرار دادیم و خطاهای حاصل در حل مسئله مربوطه را با یکدیگر مقایسه کردیم که در شکل‌های ۱۱ و ۱۲ در تکرارهای ۵۰ و ۱۰۰ مورد نشان داده شده است. این خطا بنابر رابطه(۳) نشان می‌دهد از تعداد سلول انتخاب‌شده توسط عامل در مسیر حرکت، چه تعداد سلول به اشتباه انتخاب شده‌اند یا مورد تنبیه قرار گرفته‌اند. با توجه به یادگیری صورت گرفته توسط عامل در مسیر حرکت، این خطاها باید کاهش یابد تا منجر به انتخاب‌های دقیق‌تر شود.

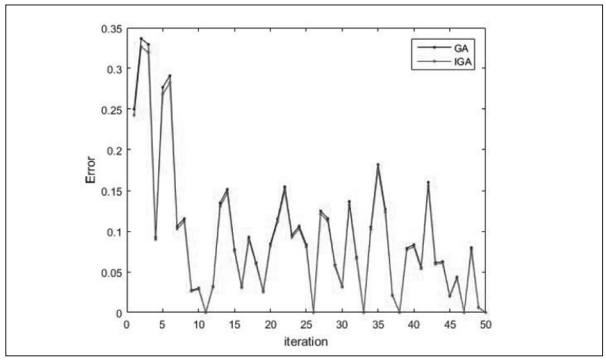
در این ارزیابی مشخص است که الگوریتم بهبودیافته ژنتیک توانسته خطای انتخاب مسیر را در حل مسئله کم کند. هرچند که در مقایسه با الگوریتم ساده ژنتیک تغییر چندانی نداشته ولی وضعیت بهتری را در این روش به



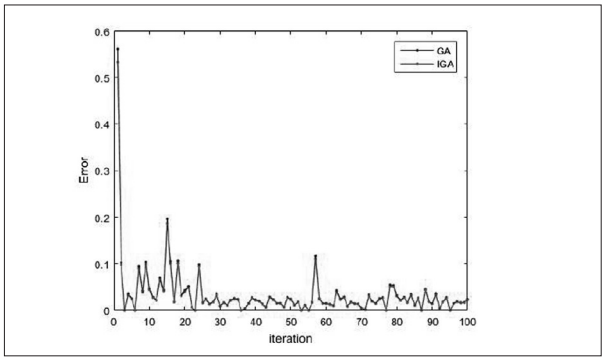
شکل ۹: دقت الگوریتم با ۵۰ بار تکرار



شکل ۱۰: دقت الگوریتم با ۱۰۰ بار تکرار



شکل ۱۱: خطای نتایج اجرای الگوریتم ژنتیک و ژنتیک بهبودیافته با ۵۰ بار تکرار



شکل ۱۲: خطای نتایج اجرای الگوریتم ژنتیک و ژنتیک بهبودیافته با ۱۰۰ بار تکرار

دست آورده است.

$$Error = \frac{\text{error\_move}}{(\text{correct move}+\text{error move})} \qquad (۳)$$

۵- نتیجه‌گیری و کارهای آینده

بررسی نتایج رویکرد پیشنهادی نشان می‌دهد که سیستم دسته‌بند یادگیر با موفقیت توانست عامل ناآگاه از محیط و وضعیت موجود در جدول را از موقعیت شروع به موقعیت هدف هدایت کند. به این معنی که با توجه به وجود موانع زیاد در جدول ماز، عامل توانست مسیر مناسبی را به سمت هدف پیدا کند. این روند با یادگیری عامل از بازخوردهای ناشی از حرکت در محیط و اعمال تشویقات و تنبیهات مرتبط صورت گرفت. همچنین در این مقاله، با به‌کارگیری الگوریتم ژنتیک بهبودیافته از گیرافتادن از بهینه‌های محلی ممانعت کرده و توانست تمام فضای مسئله را پوشش دهد. با تکرار حل این مسئله و آزمایش‌ها صورت گرفته مشخص گردید در تکرارهای بعدی با توجه به یادگیری عامل، سرعت رسیدن به هدف بیشتر شده و مدت‌زمان کمتری برای این امر طول می‌کشد.

جهت کارهای آینده پیشنهاد می‌گردد از این روش به‌صورت ترکیبی با الگوریتم‌های دیگر جهت دستیابی به جواب‌های بهتر استفاده شود و در صورت وجود چند مسیر، کوتاه‌ترین مسیر انتخاب گردد.

مراجع

1. L. B. Booker, D. E. Goldberg, and J. H. Holland, “Classifier systems and genetic algorithms,” Artif. Intell., vol. 40, no. 1–3, pp. 235–282, 1989.
2. S. W. Wilson, “Classifier Systems and the Animat Problem,” Mach. Learn., vol. 2, no. 3, pp. 199–228, 1987.
3. C. S. Lee and S. B. Cho, “Learning classifier systems for adaptive learning of intrusion detection system,” Adv. Intell. Syst. Comput., vol. 649, pp. 557–566, 2018.
4. J. Drugowitsch, “Design and Analysis of Learning Classifier Systems: A Probabilistic Approach,” Stud. Comput. Intell., vol. 139, no. March, 2009.
5. J. H. Holland et al., “What Is a Learning Classifier System?,” pp. 3–32, 2000.
6. J. H. Holland, “Genetic algorithms,” Sci. Am., vol. 267, no. 1, pp. 66–73, 1992.

7. L. Bull, “A brief history of learning classifier systems: from CS-1 to XCS and its variants,” Evol. Intell., vol. 8, no. 2–3, pp. 55–70, 2015.
8. L. Bull, “Learning classifier systems: A brief introduction,” in Applications of Learning Classifier Systems, Springer, pp. 1–12, 2004.
9. J. H. Holmes, P. L. Lanzi, W. Stolzmann, and S. W. Wilson, “Learning classifier systems: New models, successful applications,” Inf. Process. Lett., vol. 82, no. 1, pp. 23–30, 2002.
10. M. Minsky, “Steps Toward Artificial Intelligence,” Proc. IRE, vol. 49, no. 1, pp. 8–30, 1961.
11. R. J. Urbanowicz and W. N. Browne, Introduction to learning classifier systems. Springer, 2017.
12. I. M. Alvarez, W. N. Browne, and M. Zhang, “Human-inspired scaling in learning classifier systems: Case study on the N-bit multiplexer problem set,” GECCO 2016 - Proc. 2016 Genet. Evol. Comput. Conf., pp. 429–436, 2016.
13. L. Bull and T. Kovacs, “Foundations of Learning Classifier Systems: An Introduction,” Found. Learn. Classif. Syst., pp. 1–17, 2005.
14. J. Shapiro, “Genetic algorithms in machine learning,” in Advanced Course on Artificial Intelligence, pp. 146–168,1999.
15. E. Debie and K. Shafi, “Implications of the curse of dimensionality for supervised learning classifier systems: theoretical and empirical analyses,” Pattern Anal. Appl., vol. 22, no. 2, pp. 519–536, 2019.
16. S. Ben-David, E. Kushilevitz, and Y. Mansour, “Online Learning versus Offline Learning,” Mach. Learn., vol. 29, no. 1, pp. 45–63, 1997.
17. B. Gupta and S. Sehgal, “Survey on techniques used in Autonomous Maze Solving Robot,” Proc. 5th Int. Conf. Conflu. 2014 Next Gener. Inf. Technol. Summit, no. March, pp. 323–328, 2014.
18. Kaur, Navin Kumar1 Sandeep. “A Review of Various Maze Solving Algorithms Based on Graph Theory.” - International Journal for Scientific Research & Development| Vol. 6, Issue 12| ISSN (online): 2321-0613, 2019
19. M. Ahuja, B. Homchaudhuri, K. Cohen, and M. Kumar, “Fuzzy counter ant algorithm for maze problem,” 48th AIAA Aerosp. Sci. Meet. Incl. New Horizons Forum Aerosp. Expo., 2010.
20. Z. Husain, D. Ruta, F. Sare, Y. Al-Hammadi, and A. F. Isakovic, “Inverted ant colony optimization for search and rescue in an unknown maze-like indoor environment,” GECCO 2018 Companion - Proc. 2018 Genet. Evol. Comput. Conf. Companion, no. July, pp. 89–90, 2018.
21. J. W. Kim and S. K. Kim, “Genetic Algorithms for Solving Shortest Path Problem in Maze-Type Network with Precedence Constraints,” Wirel. Pers. Commun., vol. 105, no. 2, pp. 427–442, 2019.
22. A. Jonasson and S. Westerlind, “Genetic algorithms in mazes Geneti ska algoritm er i labyrinter,” 2016.
۲۳. محمدی هـادی، میرزایی کمـال، ، الگوریتم موازی ممتیکی جسـتجوی ممنوعه برای حل مسئله تخصیص درجه دوم»، مجله علوم رایانشی، شماره ۱۵، صفحه ۶۳-۵۰، زمستان ۹۸
24. A. BakarSayutiSaman and I. Abdramane, “Solving a Reconfigurable Maze using Hybrid Wall Follower Algorithm,” Int. J. Comput. Appl., vol. 82, no. 3, pp. 22–26, 2013.
25. I. Elshamarka and A. Bakar Sayuti Saman, “Design and Implementation of a Robot for Maze-Solving using Flood-Fill Algorithm,” Int. J. Comput. Appl., vol. 56, no. 5, pp. 8–13, 2012.
26. Z. Cai, L. Ye, and A. Yang, “FloodFill maze solving with expected toll of penetrating unknown walls for micromouse,” Proc. 14th IEEE Int. Conf. High Perform. Comput. Commun. HPCC-2012 - 9th IEEE Int. Conf. Embed. Softw. Syst. ICESS-2012, pp. 1428–1433, 2012.
27. S. Mishra and P. Bande, “Maze solving Algorithms for micro mouse,” SITIS 2008 - Proc. 4th Int. Conf. Signal Image Technol. Internet Based Syst., pp. 86–93, 2008.
28. Y. J. Zhang, Z. L. Zhang, and Y. Deng, “An improved maze solving algorithm based on an amoeboid organism,” Proc. 2011 Chinese Control Decis. Conf. CCDC 2011, pp. 1440–1443, 2011.
29. H. Dang, J. Song, and Q. Guo, “An efficient algorithm for robot maze-solving,” Proc. - 2010 2nd Int. Conf. Intell. Human-Machine Syst. Cybern. IHMSC 2010, vol. 2, pp. 79–82, 2010.
30. S. Salmanpour, H. Omranpour, and H. Motameni, “An intelligent water drops algorithm for solving robot path planning problem,” CINTI 2013 - 14th IEEE Int. Symp. Comput. Intell. Informatics, Proc., pp. 333–338, 2013.
31. J. Ledéus, “A comparison of Intelligent Water Drops and Genetic Algorithm for maze solving A comparison of Intelligent Water Drops and Genetic Algorithm for maze solving,” 2018.
32. V. Ntinas, I. Vourkas, G. C. Sirakoulis, and A. I. Adamatzky, “Oscillation-Based Slime Mould Electronic Circuit Model for Maze-Solving Computations,” IEEE Trans. Circuits Syst. I Regul. Pap., vol. 64, no. 6, pp. 1552–1563, 2017.
33. N. H. Barnouti, S. S. M. Al-Dabbagh, and M. A. Sahib Naser, “Pathfinding in Strategy Games and Maze Solving Using A* Search Algorithm,” J. Comput. Commun., vol. 04, no. 11, pp. 15–25, 2016.
34. X. Yuan, Y. Zhang, and Y. Yuan, “Improved self-adaptive chaotic genetic algorithm for hydrogeneration scheduling,” J. Water Resour. Plan. Manag., vol. 134, no. 4, pp. 319–325, 2008.
35. M. Yousefi, M. Yousefi, R. P. M. Ferreira, J. H. Kim, and F. S. Fogliatto, “Chaotic genetic algorithm and Adaboost ensemble metamodeling approach for optimum resource planning in emergency departments,” Artif. Intell. Med., vol. 84, no. October, pp. 23–33, 2018.