DOR: https://csj.isi.org.ir/article_245093.html

DOI: https://doi.org/10.22034/csj.2026.545441.1126

بهار ۱۴۰۵، دوره ۱۱، شماره ۱، پیاپی ۴۰

دریافت مقاله: ۱۴۰۴/۰۶/۲۵

پذیرش مقاله: ۱۴۰۴/۰۸/۱۲

نوع مقاله: مروری

بررسی کارایی سیستم‌های دسته‌بند یادگیر با رویکردهای مختلف یادگیری تقویتی برای حل مسئله ماز

کمال میرزایی*

گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی میبد، ایران

پست الکترونیک: k.mirzaie@maybodiau.ac.ir

عزیزه آقایی میبدی

گروه مهندسی کامپیوتر، واحد میبد، دانشگاه آزاد اسلامی میبد، ایران

پست الکترونیک: Azizeh.aghaee@maybodiau.ac.ir

چکیده

در این پژوهش، عملکرد سه رویکرد هوش مصنوعی شامل الگوریتم ژنتیکی، یادگیری تقویتی و سیستم‌های دسته‌بند یادگیر در حل مسئله مسیریابی در محیط‌های ماز بررسی و مقایسه شده است. هدف اصلی، ارزیابی توانایی این الگوریتم‌ها در یافتن مسیر بهینه از مبدأ به مقصد در مازهای پیچیده با موانع متعدد است. برای این منظور، هر الگوریتم در محیطی یکسان پیاده‌سازی شده و با استفاده از معیارهایی نظیر تعداد گام‌ها تا رسیدن به هدف، تعداد اپیزودهای آموزشی تا رسیدن به عملکرد قابل قبول، میزان مصرف حافظه و نرخ همگرایی مورد ارزیابی قرارگرفته است. این مقاله، با حل مسئله ماز با الگوریتم ژنتیک، انواع الگوریتم‌های یادگیری تقویتی و سیستم‌های دسته‌بند یادگیر، نقاط قوت و ضعف متمایز این رویکردها را نشان‌می‌دهد. هر الگوریتم روش‌های منحصر به فردی برای یافتن مسیر ارائه می‌دهد که تحت تأثیر اصول اساسی و کارایی عملیاتی آنها قرار دارد. نتایج نشان‌می‌دهد که الگوریتم ژنتیک با وجود همگرایی سریع‌تر، در برخی موارد دچار نوسان در کیفیت مسیر می‌شود. در مقابل، الگوریتم Q-Learning به‌صورت تدریجی ولی پایدار به راه‌حل بهینه نزدیک می‌شود. سیستم‌های دسته‌بند یادگیر نیز با ارائه قوانین قابل تفسیر، عملکرد قابل قبولی از خود نشان داده‌اند، هر چند همگرایی آنها نسبت به دو الگوریتم دیگر کندتر است و ترکیب ویژگی‌های الگوریتم‌های تقویتی و ژنتیکی در سیستم‌های دسته‌بند یادگیر نیز منجر به بهبود نسبی نتایج شده است. در مجموع، یافته‌های این پژوهش می‌تواند راهنمایی مؤثر برای انتخاب الگوریتم مناسب در کاربردهای مختلف مسیریابی و تصمیم‌گیری هوشمند فراهم آورد.

کلید واژه‌ها: مسئله ماز، سیستم دسته‌بند یادگیر، الگوریتم تقویتی، الگوریتم ژنتیک

۱- مقدمه

مسئلهٔ مسیریابی در محیط‌های پیچیده یکی از چالش‌های اساسی در حوزهٔ هوش مصنوعی و روباتیک

* نویسندهٔ مسئول

محسوب می‌شود. یافتن کوتاه‌ترین یا بهینه‌ترین مسیر از نقطهٔ شروع به مقصد نهایی در حضور موانع، کاربردهای گسترده‌ای در سامانه‌های ناوبری، بازی‌های رایانه‌ای و روبات‌های خودمختار دارد. در سال‌های اخیر، الگوریتم‌های مختلفی برای حل این مسئله پیشنهاد شده‌اند که هر یک مزایا و محدودیت‌های خاص خود را دارند. حل مسئله ماز یکی از مسائل کلاسیک در نظریه گراف است[۱]. سیستم دسته‌بند یادگیر[۱] یک تکنیک یادگیری ماشینی برای تولید سیستم‌های سازگار است. هسته یک سیستم دسته‌بند یادگیر مجموعه‌ای از قوانین است که به آن جمعیت، دسته‌بندها گفته می‌شود. سیستم‌های دسته‌بند یادگیر از دو استعاره زیست‌شناسی تکامل و یادگیری استفاده می‌کنند. جایی که یادگیری، مولفه تکاملی را هدایت می‌کند تا به سمت مجموعهٔ بهتری از قوانین حرکت‌کند. این مفاهیم به ترتیب توسط دو سازوکار الگوریتم ژنتیک و یادگیری مناسب تجسم می‌شوند. در سیستم دسته‌بند یادگیر، محیط، منبع داده‌های ورودی به این الگوریتم است. مولفه‌های اصلی سیستم دسته‌بند یادگیر به‌ترتیب زیر است:

- مولفه جمعیت[۲]: نشان‌دهنده دانش فعلی سیستم است.
- مولفه کارایی[۳]: تعامل بین محیط و جمعیت دسته‌بند را تنظیم می‌کند.
- مولفه تقویت‌کننده[۴]: پاداش دریافتی از محیط را به دسته‌بندها توزیع می‌کند.
- مولفه کشف[۵]: از عملگرهای مختلف برای کشف قوانین بهتر و بهبود قوانین موجود استفاده می‌کند.

چهار مولفه بالا چارچوب اصلی الگوریتم را در بر دارد که تنها، دو سازوکار اصلی هستند که وظیفه هدایت سیستم را برعهده می‌گیرند. این سازوکارها مولفه‌های «کشف» و «یادگیری تقویتی[۶]» هستند[۱]. لذا در این تحقیق انواع الگوریتم‌های تقویتی و کشف برای مسئله ماز را بررسی و با هم مقایسه می‌شود. لذا در ادامه معرفی انواع الگوریتم‌های مربوطه می‌پردازیم.

1- LCS = Learning Classifier System
2- Population
3- Performance
4- Reinforcement
5- Discovery
6- RL: Reinforcement Learning

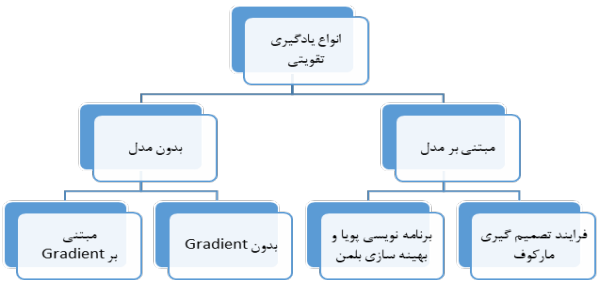
الف-الگوریتم‌های ژنتیکی

الگوریتم‌های ژنتیکی از اصول تکاملی برای بهینه‌سازی مسیریابی در مازها استفاده می‌کنند. مطالعات نشان می‌دهد که GAها به‌طور موثر می‌توانند مسیرهای کوتاه‌تری را در مقایسه با روش‌های سنتی مانند BFS و DFS به‌ویژه در مازهای پیچیده با موانع متعدد پیدا کنند[۲]. برنامه‌ریزی مسیر الگوریتم ژنتیک۷ عملکرد رقابتی را از نظر طول مسیر و زمان محاسباتی نشان می‌دهد[۳].

ب-یادگیری تقویتی

یادگیری تقویتی با اجازه دادن به عامل، به‌طور پویا از محیط خود یاد می‌گیرد. یادگیری تقویتی یکی از شاخه‌های جذاب یادگیری ماشین است که در آن یک عامل با تعامل با محیط و دریافت پاداش، یاد می‌گیرد چگونه تصمیم‌گیری کند.

الگوریتم‌های تقویتی به دو دسته اصلی تقسیم می‌شوند (شکل ۱):



شکل ۱: انواع یادگیری تقویتی

الگوریتم‌های معروف یادگیری تقویتی را می‌توان در دو دستهٔ کلی روش‌های مبتنی بر مقدار[۸] و روش‌های مبتنی

7- GAPP: Genetic Algorithm Path Planning
8- Value-Based

بر سیاســت۹ و همچنین ترکیبی از این دو طبقه‌بندی کرد [٤].

روش‌هــای مبتنی بر مقــدار: این الگوریتم‌هــا به‌دنبال تخمیــن تابع ارزش[۱۰] برای هــر حالت یا جفت حالت- عمل هستند.

- Q-Learning: ســاده‌ترین و پرکاربردتریــن الگوریتــم یادگیری تقویتی خارج از سیاست[۱۱] است. هدف آن یادگیری تابع Q(s,a) برای انتخاب بهترین عمل در هر حالت است.
- SARSA[۱۲]: شــبیه Q-Learning است، اما درون‌سیاست[۱۳] عمل‌می‌کند؛ یعنی بر اســاس همان سیاستی که یاد گرفته، عمل بعدی را انتخاب می‌کند.
- Deep Q-Network (DQN): نسخهٔ عمیق Q-Learning است که از شبکه‌های عصبی عمیق برای تقریب تابع Q استفاده می‌کند توسط DeepMind، معروف شده است.
- روش‌های مبتنی بر سیاست[۱٤]: در این روش‌ها مستقیماً سیاست[۱۵] بهینه π(als) یاد گرفته می‌شود.
- REINFORCE (Monte Carlo Policy Gradient)
- یکی از ساده‌ترین الگوریتم‌های گرادیان سیاست است که سیاست را با استفاده از گرادیان تابع هدف تنظیم می‌کند.
- Actor–Critic
- ترکیبی از روش‌های مقدار و سیاســت؛ شامل دو بخش است Actor: برای انتخاب عمل و Critic برای ارزیابی عمل
- روش‌های ترکیبــی[۱٦]: این الگوریتم‌ها از مفاهیم عمیق و شبکه‌های عصبی استفاده می‌کنند و در کاربردهای مدرن (روباتیک، بازی، کنترل) رایج‌اند.
- A3C (Asynchronous Advantage Actor–Critic)
- نسخه‌ای بهینه‌تر و پایدارتر از Actor–Critic با چند عامل یادگیرنده موازی.
- DDPG (Deep Deterministic Policy Gradient)

مناســب برای محیط‌های پیوسته[۱۷]، ترکیب DQN و Actor–Critic.

- TD3 (Twin Delayed DDPG)

بهبودیافتهٔ DDPG با کاهش بیش‌براوردی مقدار Q.

- PPO (Proximal Policy Optimization)

(الگوریتمی بسیار پایدار و محبوب) توسعه‌یافته توسط (OpenAI) که تعادل خوبی بین ســادگی، پایداری و کارایی دارد.

- SAC (Soft Actor–Critic)

الگوریتم پیشرفته‌ای بر پایه‌ی یادگیری حداکثر آنتروپی، با هدف کاوش بیشــتر و پایداری بالاتر در یادگیری (جدول ۱).

جدول ۱ : الگوریتم‌های معروف یادگیری تقویتی

الگوریتم	نوع	ویژگی‌ها
Q-Learning	Off-policy	یادگیری بدون مدل، بهینه‌سازی تابع Q برای بیشینه‌سازی پاداش
SARSA	On-policy	یادگیری با توجه به سیاست فعلی، مناسب برای محیط‌های پویا
DQN (Deep Q Network)	Off-policy	ترکیب Q-Learning با شبکه‌های عصبی برای محیط‌های پیچیده
Policy Gradient	سیاست‌محور	بهینه‌سازی مستقیم سیاست بدون استفاده از تابع ارزش
Actor-Critic	ترکیبی	ترکیب دو بخش Actor (سیاست) و Critic (ارزیابی) برای یادگیری مؤثرتر
PPO (Proximal Policy Optimization)	سیاست‌محور	الگوریتم پایدار و قدرتمند برای یادگیری در محیط‌های پیچیده

تفاوت‌های اصلی بین SARSA و Q-Learning

در SARSA از عمل واقعی بعدی (a′) که طبق سیاســت فعلی انتخاب شــده اســتفاده می‌کند و همان سیاستی که عامل با آن تعامل می‌کند مثلاً ε-greedy استفاده می‌کند ولی در Q-Learning از بهتریــن عمل ممکن در حالت بعدی (a′) استفاده می‌کند، حتی اگر طبق سیاست فعلی انتخاب نشده باشد. سیاستی که عامل یاد می‌گیرد، معمولاً greedy است و مستقل از سیاست فعلی است [٥] (جدول ۲).

الگوریتــم DQN (Deep Q Network): ترکیبی قدرتمند از یادگیری تقویتی و شــبکه‌های عصبی عمیق است که برای حل مســائل با فضای حالت پیچیده و بزرگ طراحی شده

9- Policy-Based
10-Value Function
11-Off-Policy
12-State–Action–Reward–State–Action
13-On-Policy
14-Policy-Based Methods
15-Policy
16-Advanced/Hybrid Methods
17- Continuous Action Space

جدول۲: تفاوت‌های اصلی بین SARSA و Q-Learning

ویژگی	SARSA	Q-Learning
نوع سیاست	On-policy	Off-policy
فرمول به‌روزرسانی	از اقدام واقعی بعدی استفاده می‌کند	از بیشینه مقدار Q در حالت بعدی استفاده می‌کند
اکتشاف و بهره‌برداری	محتاط‌تر و مطابق با سیاست فعلی	تهاجمی‌تر و فرض بر اقدام بهینه
پایداری یادگیری	پایدارتر و کمتر دچار بیش‌براوردی	ممکن است مقادیر Q را بیش‌براورد کند
سرعت همگرایی	کندتر ولی مطمئن‌تر	سریع‌تر ولی با ریسک بیشتر
مناسب برای	محیط‌های پرریسک یا واقعی	محیط‌های شبیه‌سازی‌شده یا کم‌ریسک

است. این الگوریتم توسط شرکت DeepMind معرفی شد و توانست در بازی‌های Atari عملکردی در سطح انسان داشته باشد[٦] (جدول۳).

جدول ۳: اجزای کلیدی الگوریتم DQN

مؤلفه	توضیح
Replay Buffer	حافظه‌ای برای ذخیره تجربه‌ها و نمونه‌گیری تصادفی
Target Network	شبکه‌ای جداگانه برای محاسبه مقدار هدف Q، به‌روزرسانی دوره‌ای
Policy Network	شبکه اصلی که Q را تقریب می‌زند و اقدام را انتخاب می‌کند
Loss Function	معمولاً MSE بین Q پیش‌بینی‌شده و Q هدف
Epsilon Decay	کاهش تدریجی ε برای حرکت از اکتشاف به بهره‌برداری

الگوریتم Policy Gradient: یکی از روش‌های قدرتمند در یادگیری تقویتی هستند که به‌جای تخمین تابع ارزش، مستقیماً سیاست را بهینه‌سازی می‌کنند. این روش‌ها به‌ویژه در محیط‌هایی با فضای اقدام پیوسته یا پیچیده بسیار مؤثرند. در روش‌های مبتنی بر ارزش مثل Q-Learning، عامل سعی می‌کند ارزش هر حالت یا اقدام رو تخمین بزند [٧]. اما در Policy Gradient، سیاست به‌صورت تابعی پارامتری شده (مثلاً با شبکه عصبی) تعریف می‌شود $\pi_\theta(a|s)$: یعنی احتمال انتخاب اقدام a در حالت s با پارامترهای θ و هدف بیشینه‌سازی تابع هدف $J(\theta) = \{E\pi_\theta\}[R]$ که R مجموع پاداش‌های دریافتی در طول اپیزود است. این الگوریتم‌ها برای فضای اقدام پیوسته مناسب هستند و یادگیری سیاست به‌صورت مستقیم انجام می‌شود و امکان یادگیری سیاست تصادفی نیز هست ولی دارای واریانس بالا در گرادیان‌ها هستند و نیاز به نمونه‌های زیاد دارند و احتمال گیرافتادن در مینیمم‌های محلی نیز جز معایب آنهاست (جدول ٤). انواع آن عبارتند از:

- ساده‌ترین نوع، تقویتی مبتنی بر مونت‌کارلو
- Actor-Critic: ترکیب سیاست‌ساز و ارزیاب[18] برای کاهش واریانس
- PPO[19]: به‌روزرسانی‌های پایدار و مؤثر

جدول ۴ : مزایا و چالش‌های الگوریتم Policy Gradient

چالش‌ها	مزایا
واریانس بالا در گرادیان‌ها	مناسب برای فضای اقدام پیوسته
نیاز به نمونه‌های زیاد	یادگیری سیاست به‌صورت مستقیم
گیر افتادن در مینیمم‌های محلی	امکان یادگیری سیاست تصادفی

پ- سیستم‌های دسته‌بند یادگیر

LCS، به ویژه مدل XCS، در وظایف یادگیری تقویتی چند هدفه برتر است و راه‌حل‌های مبتنی بر قانون ارائه می‌دهد که به‌خوبی در تنظیمات مختلف ماز تعمیم می‌گیرند. سازگاری LCS امکان یادگیری موثر راه‌حل‌های بهینه و متعادل کردن اهداف متعدد در مسائل ماز بدون نیاز به ذخیره‌سازی بیش از حد را فراهم می‌کند[٨]. در حالی که الگوریتم‌های ژنتیکی و یادگیری تقویتی بر بهینه‌سازی مسیریابی از طریق فرایندهای تکاملی و یادگیری تمرکز می‌کنند، LCS یک چارچوب قوی برای مدیریت سناریوهای پیچیده و چندهدفه ارائه می‌دهد این تنوع در رویکردها پتانسیل سیستم‌های ترکیبی را برجسته می‌کند که می‌توانند از نقاط قوت هر روش برای قابلیت‌های پیشرفته حل ماز استفاده کنند.

انواع معماری LCS: در سبک میشیگان[20] هر قانون به‌صورت مستقل ارزیابی و تکامل می‌یابد و در سبک

18-Critic
19-Proximal Policy Optimization
20-Michigan-style

پیتزبورگ[۲۱] مجموعه‌ای از قوانین به‌عنوان یک فرد در جمعیت در نظر گرفته می‌شود. تفاوت اصلی بین معماری‌های سبک میشیگان و سبک پیتزبورگ در سیستم‌های دسته‌بند یادگیر [۱۰] در جدول ۵ آمده است.

جدول۵: تفاوت‌های کلیدی بین معماری سبک میشیگان و معماری سبک پیتزبورگ

ویژگی	Michigan-style	Pittsburgh-style
واحد تکامل	قانون منفرد	مجموعه‌ای از قوانین
نمایش فرد در جمعیت	یک قانون	یک مجموعه قوانین (Rule-set)
عملکرد الگوریتم ژنتیک	روی قوانین جداگانه	روی کل مجموعه قوانین
نوع یادگیری	یادگیری افزایشی (online)	یادگیری دسته‌ای (batch)
انعطاف‌پذیری	بالا در محیط‌های پویا	مناسب برای مسائل ایستا
پیچیدگی محاسباتی	کمتر	بیشتر
مثال معروف	XCS, UCS, ExST-raCS	Gassist, Pittsburgh-style XCS

دو سازوکار اصلی در سیستم یادگیر کشف[۲۲] یادگیری تقویتی[۲۳] است. کشف به این معناست که عامل محیط را بررسی می‌کند تا مسیرها و وضعیت‌هایی را تجربه کند که قبلاً ندیده است. در ماز، عامل به جای انتخاب همیشهٔ بهترین مسیر فعلی، گاهی مسیرهای جدید را امتحان می‌کند؛ این رفتار باعث می‌شود از «گیر افتادن» در یک مسیر محلی جلوگیری شود. در الگوریتم‌های تقویتی مثل Q-Learning یا SARSA، این کار معمولاً با سیاست ε-greedy انجام می‌شود با احتمال ε عمل تصادفی انتخاب می‌شود (کشف)، و با احتمال 1−ε بهترین عمل فعلی (بهره‌برداری) انتخاب می‌شود. در بخش یادگیری تقویتی، عامل پاداش[۲۴] را از محیط دریافت‌می‌کند؛ بر اساس این پاداش‌ها، ارزش (Q-value) هر عمل را در هر وضعیت به‌روزرسانی می‌کند و در نتیجه به تدریج یاد می‌گیرد کدام توالی اعمال بیشترین پاداش (کمترین هزینه‌ی مسیر) را دارد. در مسیریابی ماز، هدف یادگیری تقویتی این است که مسیرهایی را بیاموزد که با کمترین تعداد گام و کمترین برخورد به دیوار، عامل را از نقطه شروع به هدف برساند.

تعامل «کشف» و «یادگیری تقویتی»: وقتی این دو با هم ترکیب شوند، کشف، محیط را «پوشش» می‌دهد و یادگیری تقویتی، مسیرهای مفید در این پوشش را «تقویت» می‌کند. در نتیجه، عامل به تدریج تعادلی بین «آزمایش مسیرهای جدید» و «استفاده از مسیرهای یاد گرفته شده» برقرار می‌کند و به مسیر بهینه نزدیک می‌شود[۹] (جدول ٦).

جدول۶: تعامل کشف و یادگیری تقویتی

سازوکار	اثر در یادگیری	تعامل با دیگری
کشف	یافتن مسیرها و حالت‌های جدید، غنی‌سازی تجربه	دادهٔ تازه برای یادگیری تقویتی تولید می‌کند
یادگیری تقویتی	ارزیابی مسیرها و به‌روزرسانی Q-value یا سیاست	با پاداش‌دهی مناسب، مسیرهای مفید را تقویت و مسیرهای بد را حذف می‌کند

مثال در ماز

در ماز، عامل ابتدا به کمک الگوریتم ژنتیک مسیرهای احتمالی زیادی را تولیدمی‌کند (کشف اولیه تصادفی اما هدایت‌شده). سپس الگوریتم SARSA یا Q-Learning از این مسیرها برای مقداردهی اولیهٔ Q-Table استفاده می‌کند (یادگیری تقویتی).

این تعامل دو اثر دارد:

۱. کاهش زمان همگرایی زیرا عامل با Q اولیه بهتر شروع می‌کند.

۲. کاهش احتمال گیر افتادن در مسیرهای محلی زیرا GA فضای حالت را متنوع پوشش داده است.

معیارهای پیشنهادی برای ارزیابی بهبود

برای سنجش میزان بهبود کارایی ناشی از تعامل کشف و یادگیری، می‌توان از معیارهای زیر استفاده کرد (جدول ۷):

21-Pittsburgh-style
22-Exploration
23-Reinforcement Learning
24-Rewards

جدول۷ : معیارهای پیشنهادی برای ارزیابی بهبود

نوع معیار	توضیح	هدف
میانگین طول مسیر (Average Path Length)	میانگین تعداد گام‌هایی که عامل برای رسیدن به هدف طی می‌کند	کوتاه‌تر = کارایی بالاتر
نرخ موفقیت (Success Rate)	درصد اپیزودهایی که عامل واقعاً به هدف رسیده است	نشان‌دهندهٔ پایداری یادگیری
تعداد اپیزود تا همگرایی (Episodes to Con-vergence)	تعداد دفعات آموزش لازم تا Q-Valueها یا سیاست پایدار شوند	کمتر = یادگیری سریع‌تر
مجموع پاداش تجمعی (Cumulative Re-ward)	جمع کل پاداش‌های کسب‌شده در طول اپیزودها	بالاتر = مسیر بهینه‌تر
زمان آموزش (Training Time)	مدت زمانی که عامل برای رسیدن به عملکرد مطلوب نیاز دارد	کمتر = تعامل مؤثرتر

تفاوت‌های کلیـــدی تأثیرگذار بر تعمیم‌پذیری و عملکرد در ماز چندهدفه[۲۵]

در مســائل ماز چندهدفه با هدف‌های متعددی که باید همزمان یا در تراز با هم بهینه شــوند، دو نگرش معماری سبک میشیگان و ســبک پیتزبورگ می‌تواند به شکل‌های مختلف روی تعمیم‌پذیری و عملکرد تأثیر بگذارد.

سبک میشیگان معمولاً به یادگیری گسسته‌تر و محلی‌تر می‌انجامــد. معمولاً هزینه نگهــداری و به‌روزرســانی‌ها را به‌صــورت محــلی دارد و می‌توانــد کارایی بهتری در محیط‌های با منابع محدود و نیاز به پاسخ‌دهی سریع ارائه دهد. در ســبک میشیگان دســته‌بندهایی که نقشی در چند هدف دارند ممکن اســت با هم تداخل ایجاد کنند، اما امکان جداسازی وظایف و به‌روزرسانی مستقل آنها وجوددارد. این مســئله می‌تواند به مدیریت تداخــل و بهبود تعمیم در تغییرات هدف کمک‌کند.

ســبک پیتزبورگ بــا انتــزاع بالاتر، امــکان یادگیری راهبردهای سیاست کلی را می‌دهد که می‌تواند در مواجهه با توازن بین اهداف مختلف مفید باشد، اما در برابر تغییرات هدفی که نیازمنــد بازنگری‌های جزءبه‌جزء اســت، ممکن است انعطاف‌پذیری کمتری نشــان‌دهد چون کل سیاست باید تغییرکند و جست‌وجوی سیاست مناسب جدید گاهی پرهزینه است. این سبک بیشتر به بازنمایی کلی از سیاست تصمیم‌گیری متمرکز اســت. اگر سیاست کلی به‌خوبی با فضای مسئله هم‌ساز شود، تعمیم می‌تواند خوب باشد اما اگر فضای مســئله دارای تغییرات پویا در اهداف یا قوانین ارزش‌گذاری باشــد، ممکن است نیازمند بازنگری گسترده در سیاست کلی باشد که هزینه‌های بیشتری دارد. به‌طوری که اگر ماز به ســمت چند هدف با وزن‌های نسبتاً پایدار و تغییرات کم در اولویت‌ها می‌رود، سبک پیتزبورگ می‌تواند با ارائه یک راهبرد تصمیم‌گیری واحد و همســو با اهداف، عملکردی پایــدار ارائه دهد. به‌طورکلی، ســبک پیتزبورگ به‌دلیل ارزیابی سیاســت‌های کلی و جست‌وجوی فضای بزرگتر برای سیاســت‌ها، هزینه محاسباتی بالاتری دارد، به‌خصوص در مسائل چندهدفه که نیازمند ارزیابی مکرر چندین سیاست اســت. در این سبک هم‌بستگی بین قوانین می‌تواند به بهبود هماهنگی بین اهداف کمک کند اما احتمالاً منجر بــه جســت‌وجوی پیچیده‌تر برای یافتن سیاســتی می‌شود که به‌طور هم‌زمان به همه اهداف پاسخ دهد.

اگــر aliasing states و فضای حالت بزرگ باشــد LCS می‌تواند با استفاده از قوانین کوچک و محلی، نسبت به یک مدل جهانی که ارزش‌های Q را پوشــش می‌دهد، بازنمایی مناســب‌تری ارائه دهــد. ترکیب LCS با GA یــا RL برای استفاده از قابلیت‌های ترکیبی جست‌وجو، بهبود کارایی و پوشش فضای جســت‌وجو را فراهم می‌کند مثلاً GA برای تولیــد و مرور راه‌حل‌های جدید، RL یــا Q-Learning برای ارزیابی پایداری و بهبود سیاست‌ها در سطح‌های مختلف. اگــر اولویت اصلی گرایش به تعــادل چندهدفه و بازنمایی سیاســت‌های کلان اســت GA/ LCS همراه با رویکردهای چندهدفــه می‌تواند سیاســت‌های Pareto-front را به‌خوبی بازنمــایی کند. RL یــا Q-Learning می‌توانــد به‌عنوان لایه تصمیم‌گیری ســطح بالا برای وزن‌دهی اهداف اســتفاده شود. اگر محیط بسیار پویا و تغییرساز است، رویکردهای مبتــنی بر LCS با بازآموزی پویا و به‌روزرســانی قوانین ممکن اســت ســریع‌تر از رویکردهــای purely Q-learning

25-Multi-objective

یا GA به تغییرات پاسخ پیاده‌سازی دهند. کارآمد شامل استفاده از موازی‌سازی برای ارزیابی نسل‌های GA و به‌روزرسانی‌های RL و کاربرد ترکیبی از حافظه تجربه (replay) و تجربه‌های گوناگون برای RL و طراحی پاداش چندهدفه با توجه به تغییرات اولویت‌ها است.

در محیط‌های ماز (Multi-objective) با فضای حالت بزرگ و پیچیده، ترکیب الگوریتم ژنتیک (GA) با یادگیری تقویتی (RL) می‌تواند هم سرعت همگرایی را بهبود دهد و هم کیفیت مسیرهای بهینه را افزایش دهد. با در نظر گرفتن موارد ذیل می‌تواند مؤثر باشد:

- جست‌وجوی مبهم و ترکیبی با ژنتیک
  - GA می‌تواند به‌طور موثری فضای راه‌حل‌های پیشنهادی را ترکیب کند (crossover) و جهش‌های تصادفی ایجاد کند تا راه‌حل‌های جدید و غیرمحسوسی تولید شود.
  - در مسائل ماز، هدف‌ها غالباً با هم در تعارض‌اند.GA با استفاده از چند هدف وزن‌دهی یا چند شاخص کارایی، مجموعه‌ای از راه‌حل‌های Pareto-frontی را می‌سازد و از تنوع بالا پشتیبانی می‌کند.
- وظیفه‌گریزی RL برای بهبود اضطرابی و بازانگاشت تجربه
  - RL می‌تواند به‌طور پویا راهبردهای تصمیم‌گیری برای اجرای راه‌حل‌های GA را بهبود دهد: مثلاً تعیین نرخ جهش، اندازه جابجائی، یا انتخاب راه‌حل‌های برتر برای افزایش احتمال همگرایی به قسمت‌های با کیفیت.
  - RL می‌تواند به مدل‌سازی ارزش‌های بلندمدت مرتبط با تعادل بین هزینه‌های محاسباتی و کیفیت راه‌حل‌ها کمک کند.
- همکاری دو سطحی
  - سطح بالایی RL: (Policy/Strategic Level) مسئول یادگیری سیاست‌هایی مانند زمان‌بندی اجرای نسل‌های GA، تنظیم معیارهای ارزیابی، یا انتخاب مجموعه‌ای از راه‌حل‌ها برای ارزیابی بیشتر.
  - سطح پایینی GA: (Genetic Level) مسئول تولید و بهبود راه‌حل‌ها با استفاده از سازوکارهای جست‌وجو و ترکیب ژنتیک.
- بهبود سرعت همگرایی و کیفیت مسیر
  - تنوع ژنتیک اولیه با GA می‌تواند فضای جست‌وجو را پوشش دهد و از گرفتار شدن در کمینه‌های محلی جلوگیری کند.
  - RL می‌تواند با یادگیری از تجربیات گذشته، برخی مسیرهای جست‌وجو را به‌تصادف یا با احتمال کمتر دنبال نکند و به سمت مسیرهای با پاداش بلندمدت بهتر هدایت کند.
  - با استفاده از ارزیابی چندهدفه، الگوریتم می‌تواند Pareto-frontی از مسیرهای با تعادل مناسب بین اهداف فراهم کند و راه‌حل‌های با کارایی بلندمدت را تشویق کند.

معماری‌های پیاده‌سازی رایج

- یکپارچه‌سازی سطحی
  - GA برای تولید و ارزیابی راه‌حل‌ها، RL برای تنظیم پارامترهای GA (مانند نرخ جهش، نرخ جابجائی، اندازه جمعیت، فهرست انتخابی) و یا انتخاب سیاست ارزیابی استفاده می‌شود.
- معماری مبتنی بر تجربه
  - یک میانگیر تجربی (Replay Buffer) برای تجربیات RL از عملکرد GA نگهداری می‌شود تا سیاست‌ها را بر پایه تجربیات گذشته بهبود دهد.
- معماری چندهدفه
  - در فضای ماز با چند هدف، می‌توان از مقیاس‌های وزن‌دهی پویا یا معیارهای Pareto استفاده کرد تا RL به تعادل بین اهداف کمک کند و GA با حفظ تنوع راه‌حل‌ها، پوشش مناسبی از سطح پاسخ‌ها ارائه دهد.

مزایا در مسایل ماز با فضای حالت بزرگ

- افزایش تنوع جست‌وجو بدون ازدست‌دادن کارایی
  - GA بهره‌گیری از ترکیبات نوآورانه را حفظ می‌کند، در

حالی که RL با هدایت تجربه‌ها، جست‌وجو را به سمت مناطق با پاداش بلندمدت هدایت می‌کند.

● بهبود تعادل بین هزینه‌های محاسباتی و کیفیت راه‌حل

○ RL می‌تواند سیاست‌هایی برای کاهش محاسبات غیرضروری یا ارزیابی‌های غیرضروری در مواقع پر هزینه تعریف کند.

● توانایی یادگیری از تغییرات محیط

○ در محیط‌های پویا که اولویت‌های اهداف یا محدودیت‌ها تغییر می‌کند، ترکیب GA-RL می‌تواند به سرعت به شرایط جدید پاسخ بدهد.

چالش‌های کلیدی پیاده‌سازی

● هم‌زمانی و هماهنگی بین دو حلقه

○ هماهنگی نرخ یادگیری RL با نرخ جهش و جابجائی GA می‌تواند حساس باشد. تنظیم نامناسب ممکن است منجر به ناپایداری یا همگرایی کند.

● مقیاس‌پذیری محاسباتی

○ ترکیب GA با RL معمولاً به هزینه‌های محاسباتی بیشتری منجر می‌شود. مدیریت منابع، موازی‌سازی و استفاده از نمونه‌های کوچکتر برای تجربه‌ها ضروری است.

● نمایش بازخورد و پایداری یادگیری

○ پاداش RL باید به‌طور دقیق با ارزش‌های ماز و اهداف چندگانه همسو باشد. طراحی تابع پاداش نادرست می‌تواند منجر به رفتار غیرمنطقی یا بیش‌اظهارگر شود.

● تعارض بین اهداف

○ در مسائل ماز، توازن بین چند هدف متنوع می‌تواند منجر به ترکیب ناهمسو شود. استفاده از روش‌های Pareto، یا ترمیم مستمر سیاست‌های RL به حفظ تعادل کمک می‌کند.

● پایداری داده‌ها و تجربه‌های طولانی

○ جمع‌آوری تجربه‌های طولانی برای RL می‌تواند به حافظه و زمان cuisson زیادی نیاز داشته باشد. روش‌های نمونه‌برداری کارآمد و مدل‌های approximate ارزش (مثل شبیه‌سازی محدود با تخمین ارزش) مفید خواهند بود.

● تردیدهای نظری و تفسیرپذیری

○ ترکیب دو حوزه با نظریه‌های مختلف می‌تواند تحلیل را پیچیده کند و تفسیرپذیری مدل برای تصمیم‌گیران حیاتی باشد، به‌ویژه در کاربردهای حساس.

نکات عملی برای پیاده‌سازی موفق

● طراحی تابع پاداش دقیق

○ تابع پاداش باید به وضوح ارتباط بین حرکت GA و دستاوردهای ماز را بازنمایی کند مثلاً کاهش فاصله تا اهداف، یا کاهش هزینه‌های اجرایی.

● مدیریت منابع با رویکرد سطحی و جامع

○ از اجرای موازی برای ارزیابی راه‌حل‌های GA در هر دوره استفاده کنید و RL را به‌گونه‌ای تنظیم کنید که از این ارزیابی‌ها به‌کارگیری بهینه انجام دهد.

● استفاده از یادگیری انتقالی

○ تجربه‌های قبلی را برای سرعت دادن به همگرایی در محیط‌های مشابه به کار بگیرید. این کار می‌تواند از بار محاسباتی جلوگیری کند.

● اعتبارسنجی راهبردها

○ به‌صورت تجربی و با مجموعه‌های مختلف از محیط‌های ماز، راهبرد ترکیبی را ارزیابی کنید تا از پایداری و عمومیت آن مطمئن شوید.

۲- پیشینه پژوهش

ماز یا هزارتو [۲۶] به معنای یک مسیر پیچ درپیچ است که معمولاً جنبهٔ معمایی دارد. در ماز، فرد هنگام حرکت با چندین مسیر روبه‌رو می‌شود و باید مسیر درست را انتخاب کند. قبلا از روش‌های سنتی مانند BFS و DFS برای حل مسئله ماز استفاده شده است.

در مقاله [۱۲] عملکرد یک سیستم دسته‌بند یادگیر مبتنی بر الگوریتمی که به نام معماری تقویت و طبقه‌بندی موقتی شناخته می‌شود را در وظایف ناوبری در ماز که شامل

26-Maze

حالت‌های پنهان است، توصیف و ارزیابی می‌کند. ارزیابی شامل مقایسه با سایر الگوریتم‌های یادگیری در وظایف ناوبری در ماز انتخابی و دشوار است. همه LCSها قادر به یادگیری تمام انواع مازهای حالت پنهان نیستند، بنابراین این معماری به‌طور خاص با سایر روش‌های مبتنی بر LCS مقایسه می‌شود که در این وظایف بیشترین توانایی را دارند. مقایسه‌ها بین الگوریتم‌ها شامل زمان آموزش، عملکرد آزمون و اندازه مجموعه قوانین یادگیری شده است. نتایج نشان می‌دهد که هر الگوریتم مزایا و معایب خاص خود را دارد. به‌عنوان مثال، در دشوارترین معمّا، میانگین مراحل این معماری برای رسیدن به هدف ۱۰/۱ است. معماری پیشنهادی، در یک وظیفه رانندگی کامیون آزمایش می‌شود که در آن باید یاد بگیرد چگونه در چهار خط وسیله نقلیه حرکت‌کند در حالی که از تصادف جلوگیری می‌کند. نتایج نشان می‌دهد که می‌تواند با تعداد کمی از تصادفات و نسبتاً کم آزمایش‌ها (به اندازه ۲۴ تصادف در ۵۰۰۰ مرحله زمانی پس از ۱۰/۰۰۰ مرحله آموزشی) به موفقیت دست یابد، اما ممکن است نیاز به چندین تلاش برای ساخت شبکه برای دستیابی به عملکرد بالا داشته باشد.

در مقاله [۱۳] کارگاه بین‌المللی سیستم‌های دسته‌بند یادگیر (IWLCS) یک کارگاه سالانه در کنفرانس GEC-CO است که در آن مفاهیم و نتایج جدید مربوط به سیستم‌های دسته‌بند یادگیر (LCSs) ارائه و بحث می‌شود. یکی از بخش‌های تکراری برنامه کارگاه، یک ارائه است که پیشرفت‌های صورت گرفته در این زمینه را در طول سال گذشته مرور و خلاصه می‌کند؛ این به منظور ارائه یک نقطه ورود آسان به جدیدترین پیشرفت‌ها و دستاوردها طراحی شده است. ارائه‌های سال‌های ۲۰۲۰ و ۲۰۲۱ با مقالات کارگاهی بررسی همراه بودند، نمای کلی از تمام آثار منتشر شده مرتبط با LCS از ۱۱ مارس ۲۰۲۱ تا ۱۰ مارس ۲۰۲۲ ارائه می‌دهد. ۴۲ مقاله‌ای که بررسی می‌کند در شش موضوع کلی گروه‌بندی شده‌اند: پیشرفت‌های نظری رسمی، معماری‌های جدید LCS، یادگیری تقویتی مبتنی بر LCS، بهبودهای الگوریتمی بر LCSهای موجود، ترکیب‌های LCS و مدل‌های یادگیری عمیق و در نهایت، انواع مختلفی از کاربردهای LCS.

در مقاله [۱۴] مشکلات ماز نمایانگر یک مدل مجازی ساده شده از محیط واقعی هستند و می‌توانند برای توسعه الگوریتم‌های اصلی بسیاری از کاربردهای دنیای واقعی مرتبط با مشکل ناوبری استفاده شوند. سیستم‌های سیستم‌های دسته‌بند یادگیر، رایج‌ترین دسته‌ای از الگوریتم‌ها برای یادگیری تقویتی در مازها هستند. با این حال، بهترین دستاوردهای LCS در مشکلات ماز هنوز عمدتاً محدود به محیط‌های بدون هم‌پوشانی است، در حالی که به نظر می‌رسد پیچیدگی LCS مانع تحلیل صحیح دلایل شکست می‌شود. علاوه بر این، کمبود دانش در مورد عواملی که یک مشکل ماز را سخت می‌کند تا توسط یک عامل یادگیری حل شود وجود دارد. برای غلبه بر این محدودیت، سعی دارد درک خود را از ماهیت و ساختار محیط‌های ماز بهبود بخشد. در این مقاله، یک عامل LCS جدید را توصیف‌می‌کند که سازوکار عملکرد ساده‌تر و شفاف‌تری دارد. از ساختار یک مدل پیش‌بینی‌کننده LCS استفاده می‌شود، سازوکار تکاملی را حذف می‌کند، روند یادگیری تقویتی را ساده‌سازی می‌کند و عامل را با توانایی ادراک تداعی که از روان‌شناسی اقتباس شده است، تجهیز می‌شود. سپس LCS جدید را با ادراک تداعی بر روی مجموعه گسترده‌ای از هزارتوها ارزیابی‌می‌شود یک ویژگی به‌خصوص دشوار برای یادگیری در ماز به نام کلون‌های مشابه شناسایی‌می‌کند که زمانی به‌وجود می‌آیند که گروه‌هی از سلول‌های مشابه در الگوهای مشابه در قسمت‌های مختلف هزارتو رخ می‌دهند. تأثیر کلون‌های مشابه و دیگر انواع مشابه‌سازی را بر الگوریتم‌های یادگیری مورد بحث قرار می‌دهد.

در [۱۵] الگوریتم ژنتیک که نمونه‌ای از زیرمجموعه الگوریتم‌های تکاملی است، رویکرد حل مسئله‌ای را دنبال

جدول ۸: مقایسه ساختاری و محتوایی مقالات

شماره مقاله	عنوان/ موضوع پژوهش	الگوریتم‌ها یا روش‌های بررسی‌شده	هدف اصلی پژوهش	محیط یا مسئله مورد استفاده	نتایج و دستاوردهای کلیدی	نقاط قوت	محدودیت‌ها / چالش‌ها
[۱۲]	سیستم دسته‌بند یادگیر TRACA برای ناوبری در ماز با حالت‌های پنهان	TRACA، XCSMH، AgentP(G)، AgentP(SA)	مقایسه عملکرد LCS های مختلف در مازهای با حالت‌های پنهان	مازهای پیچیده با مناطق تکرارشونده و وظایف رانندگی کامیون	TRACA با آموزش کمتر عملکرد مشابه سایر روش‌ها رسید؛ در وظایف رانندگی عملکرد پایدار داشت	کاهش نیاز به آموزش طولانی، یادگیری پایدار	نیاز به چندین تلاش برای ساخت شبکه بهینه
[۱۳]	مروری بر پیشرفت‌های سیستم‌های یادگیرنده طبقه‌بندی‌کننده (IWLCS 2021–2022)	مجموعه LCSها و ترکیب با یادگیری عمیق	مرور جامع پیشرفت‌ها در LCS طی یک سال اخیر	مرور نظری و تجربی	دسته‌بندی ۴۲ مقاله در ۶ محور از جمله معماری‌های جدید، یادگیری تقویتی و ترکیب LCS با یادگیری عمیق	جامعیت و به‌روز بودن، ارائه دید کلی از تحولات LCS	فاقد تحلیل تجربی مستقیم یا مقایسه عددی
[۱۴]	تحلیل LCS در مازهای دارای aliasing و کلون‌های مشابه	LCS مبتنی بر مدل پیش‌بینی‌کننده با حذف بخش تکاملی	بررسی علل شکست LCS در مازهای پیچیده و شناسایی الگوهای دشوار	مازهای دارای الگوهای تکرارشونده و aliasing	شناسایی پدیده کلون‌های مشابه به‌عنوان عامل دشواری یادگیری، مدل ساده‌تر و قابل‌درک‌تر	افزایش شفافیت و تحلیل‌پذیری سیستم یادگیر	حذف سازوکار تکاملی تعمیم‌پذیری را کاهش می‌دهد
[۱۵]	چارچوب الگوریتم ژنتیکی برای حل ماز دوبعدی (Gamaze)	GA کلاسیک با مکانیزم به‌روزرسانی جمعیت جدید	بهبود فرآیند بهینه‌سازی مسیر در مازهای دوبعدی	مازهای دوبعدی با موانع متعدد	Gamaze مسیرهای بهینه را سریع‌تر یافت و عملکرد بهتری نسبت به GA سنتی داشت	بهبود سرعت همگرایی، طراحی ساده	محدود به مازهای دوبعدی؛ نیاز به آزمون در محیط‌های پویا
[۱۶]	بهبود الگوریتم ژنتیک با استفاده از دانش خاص دامنه برای تصمیم‌گیری	GA سنتی و GA ترکیب‌شده با قوانین تصمیم‌گیری	افزایش دقت و سرعت همگرایی در مسایل تصمیم‌گیری و ماز	مجموعه معماها و هزارتوهای تصمیم‌گیری	GA ترکیبی زمان محاسباتی را ۹۵ درصد کاهش و کیفیت پاسخ‌ها را افزایش داده است	کارایی بالا، بهبود کیفیت و سرعت در حل مسائل	نیاز به تعریف دستی دانش دامنه، کاهش تعمیم‌پذیری

می‌کند که از فعالیت‌های زیست‌شناختی مانند جهش، متقاطع و انتخاب الهام گرفته شده است. به‌عنوان تلاشی برای نمایش عملکرد GA در مسائل بهینه‌سازی ساده دنیای واقعی، این مقاله پیشنهاد می‌کند که یک چارچوب الگوریتم ژنتیکی برای حل مسئله ماز دو بعدی معرفی شود. چارچوب پیشنهادی به نام Gamaze نام‌گذاری شده است. نوآوری Gamaze در نحوه به‌روزرسانی جمعیت از نسلی به نسل دیگر نهفته است. Gamaze از اطلاعات تناسب راه‌حل‌های نامزد برای بهینه‌سازی ظرفیت یادگیری افراد استفاده می‌کند و به آنها کمک می‌کند تا به هدف نهایی برسند و درک کنند که چگونه بر موانع غلبه کنند و راهی پیدا کنند. روشگان تفصیلی Gamaze پیشنهادی در این مقاله ارائه شده است. نتایج تجربی نشان داد که Gamaze می‌تواند به حل نمونه‌های مختلف مسئله ماز بپردازد. استنتاج‌های دقیق همراه با محدودیت‌ها در این مقاله ارائه شده است.

در [۱۶] یکی از جالب‌ترین کاربردهای الگوریتم‌های ژنتیکی در حوزهٔ پشتیبانی از تصمیم‌گیری قرار دارد. مشکلات پشتیبانی از تصمیم‌گیری شامل مجموعه‌ای از تصمیمات است که هر کدام تحت تاثیر تمامی تصمیماتی هستند که قبل از آن اتخاذ شده‌اند. این دسته از مشکلات به‌طور مکرر در مدیریت بنگاه‌ها رخ می‌دهد، به‌ویژه در زمینه زمان‌بندی یا اختصاص منابع. به‌منظور نشان دادن فرمول‌بندی این دسته از مشکلات، مجموعه‌ای از مسائل

معما ارائه می‌شـــود. پیچیدگی معماها با معرفی هر معمای جدید افزایش می‌یابد. دو ســـناریوی حل مســـائل معرفی می‌شـــوند و نتایج مقایسه‌ای ارائه می‌شوند. سناریوی اول شامل رویه سنتی الگوریتم ژنتیکی برای هدف دستیابی به یک راه‌حل بر اساس یک رویکرد کاملاً تکاملی بود. سناریوی دوم از الگوریتم ژنتیکی در کنار دانش خاص دامنه در قالب قوانین تصمیم‌گیری استفاده کرد. پیاده‌سازی دانش خاص دامنه با هدف بهبود راه‌حل‌ها انجام می‌شود. اجرای دانش خاص دامنه به‌منظور بهبود زمان همگرایی راه‌حل و بهبود کیفیت کلی نسل‌های تولید شده است که به‌طور قابل‌توجهی احتمال به‌دســـت آوردن یک راه‌حـــل دقیق و هماهنگ‌تر را افزایش‌می‌دهد. نتایج برای تمام هزارتوهای در نظر گرفته شده ارائه شده است. این نتایج شامل نتیجه نهایی الگوریتم ژنتیک ســـنتی و رویکرد بهینه‌ســـازی الگوریتم ژنتیک با نتایـــج قوانین داخلی اســـت. هـــر دو نتیجه برای مقایســـه گنجانده شده است. به‌طورکلی، گنجاندن دانش خاص دامنه در مقایســـه با الگوریتم ژنتیک سنتی در هر دو عملکرد و زمان محاســـباتی برتری داشـــت. به‌طور خاص، الگوریتم ژنتیک سنتی نتوانســـت به‌طور کافی یک راه‌حل قابل‌قبول برای هر مثال ارائه شـــده، پیداکند و به‌طور میانگین در ٥٤ درصد از نســـل‌های مشخص شده خود به‌طور زود هنگام همگرا شـــد. علاوه‌بر این، پیچیده‌تریـــن هزارتو یک دنباله جهت مسیر بهینه یعـــنی N، S، E، W از طریق یک الگوریتم ژنتیکی سنتی که تنها ٥٠ درصد از دنباله‌های مسیری مجاز برای کامل کردن هزارتو را در اختیار داشت. ترکیب قوانین نهفته به الگوریتم ژنتیکی اجازه داد تا مسیر بهینه را برای تمام مثال‌های مورد بررسی در ٥ درصد زمان محاسباتی الگوریتم ژنتیکی سنتی پیدا کند.

در جدول ۸، مقایسهٔ ساختاری و محتوایی بین مقاله‌های [۱۲ تا ۱٦] که در حوزهٔ سیســـتم‌های دســـته‌بند یادگیر و الگوریتم‌های ژنتیک در مســـائل ماز و تصمیم‌گیری انجام شده‌اند، ارائه شده است.

۳- پیاده‌سازی و ارزیابی

در این پژوهش از الگوریتم‌های ژنتیک و تقویتی و LCS برای پیداکـــردن بهترین مسیر ماز توســـط کدنویسی در پایتون استفاده شده است. و در ادامه مقایسه خواهد شد از انواع الگوریتم و ترکیب آنها کدام یک برای حل مسئله ماز عملکرد بهتری دارد. مطالعات متعددی در زمینهٔ اســـتفاده از الگوریتم‌های هوشمند برای حل مسئلهٔ مسیریابی انجام شده اســـت. الگوریتم ژنتیکی به‌دلیل توانایی در جستجوی فضای بـــزرگ راه‌حل‌ها، در بســـیـــاری از پژوهش‌ها برای یافتن مسیرهای بهینه به‌کار رفته است. همچنین، یادگیری تقویـــتی به‌ویژه الگوریتـــم Q-Learning، به‌عنـــوان یکی از روش‌هـــای مؤثـــر در یادگیری مبتنی بـــر تعامل با محیط شناخته‌می‌شـــود. از ســـوی دیگر، سیســـتم‌های دسته‌بند یادگیر با ترکیـــب یادگیری تقویتی و منطـــق قاعده‌محور، امکان یادگیری قوانین قابل تفسیر را فراهم می‌کنند. با این حال، مقایسهٔ مستقیم این سه رویکرد در یک محیط یکسان کمتر مورد توجه قرار گرفته است. این پژوهش تلاش دارد تا این خلأ را پوشـــش دهـــد. ابتدا با اســـتفاده از الگوریتم ژنتیک به‌تنهایی مســـئله ماز، بررسی می‌شـــود و سپس از الگوریتم تقویتی Q-Learning برای حل مسئله ماز به‌تنهایی استفاده شد. در حالت سوم با ترکیب الگوریتم‌های تقویتی مختلف با ژنتیک مسئله ماز را بررسی خواهیم‌کرد و نتایج هرکدام ارائه شده است.

ســـناریوی اول، حل مســـاله مـــاز با پیاده‌ســـازی الگوریتم ژنتیک: پیاده‌سازی الگوریتم ژنتیک در پایتون برای حل مســـئله ماز در ذیل ارائه شده است. این روش از تکنیک‌هـــای الهام گرفته از تکامل برای بهینه‌ســـازی مسیرهـــا در ماز اســـتفاده می‌کند. مراحـــل:- ایجاد یک جمعیـــت از مسیرهـــای تصـــادفی.- ارزیابی تناســـب براســـاس نزدیکی مسیـــر به هدف.- انتخـــاب، تلاقی و جهش، نســـل‌ها را بهبود می‌بخشد.- تکرار تا زمانی که یک راه‌حل بهینه پیدا شود.

ســـناریوی دوم، پیاده سازی الگوریتم تقویتی: در ابتدا

فقط الگوریتم Q-Learning به تنهایی برای بررسی مسئله ماز مورد استفاده قرار گرفت و سپس انواع الگوریتم یادگیری تقویتی Q-Learning، Monte Carlo، SARSA، DQN، Actor-Critic و PPO که در قسمت‌های قبل معرفی گردید را برای مسئله ماز همراه با الگوریتم ژنتیک، بررسی می‌شود

سناریوی سوم سیستم دسته‌بند یادگیر: سیستم دسته‌بند یادگیر، یکی از روش‌های یادگیری ماشین است که برای حل مسئله ماز می‌توان از آن استفاده کرد. این سیستم با استفاده از الگوریتم ژنتیک و یادگیری تقویتی مسیر بهینه را پیدا می‌کند. در سناریوی سوم از سیستم دسته‌بند یادگیر و الگوریتم ژنتیک برای حل مسئله ماز استفاده شده است.

معیارهای مقایسه: برای مقایسه این الگوریتم‌ها، این موارد اندازه‌گیری و تحلیل می‌شود:

زمان اجرا[۲۷]: مدت زمانی که هر الگوریتم برای یافتن مسیر صرف می‌کند.

دقت[۲۸] یا Success Rate: چند درصد از دفعات اجرا ، مسیر صحیح رو پیدا می‌کند.

تعداد گام‌ها تا رسیدن به هدف: کوتاه‌ترین مسیر پیدا شده (کمترین تعداد قدم). مخصوصاً برای محیط‌هایی که هدف پیدا کردن بهینه‌ترین مسیر هست.

تعداد اپیزودهای آموزشی تا رسیدن به عملکرد قابل قبول: مخصوص الگوریتم‌های تقویتی و LCS الگوریتم ژنتیک هم به تعداد نسل‌ها برای رسیدن به حل مناسب نیاز دارد.

مصرف حافظه[۲۹]: منابعی که الگوریتم برای اجرا نیازدارد. الگوریتم‌های تقویتی مثل DQN ممکن است حافظه بیشتری مصرف‌کنند.

میزان همگرایی[۳۰]: سرعت رسیدن به راه‌حل پایدار در طول آموزش.

27-Execution Time
28-Accuracy
29-Memory Usage
30-Convergence Rate

مراحل کار:

مرحله ۱: تعریف محیط ماز: ابتدا یک ماز ساده تعریف‌می‌شود که با نقاط شروع و پایان مشخص است. مثلاً یک ماتریس دو بعدی که ۰ نشان‌دهنده مسیر باز و ۱ نشان‌دهنده مانع باشد.

```
maze = [[۰, ۱, ۰, ۰, ۰],  [۰, ۱, ۰, ۱, ۰], [۰, ۰, ۰, ۱, ۰], [۰, ۱,
۱, ۱, ۰], [۰, ۰, ۰, ۰, ۰]]
start = (۰, ۰)
goal = (٤, ٤)
```

در این پیاده‌سازی ساده، هر مسیر به‌صورت یک رشته ژنی نمایش داده می‌شود و هدف این است که بهترین مسیر (با کمترین قدم و رسیدن به مقصد) پیدا شود. در جدول ٦، مقایسه نهایی سه الگوریتم آمده است.

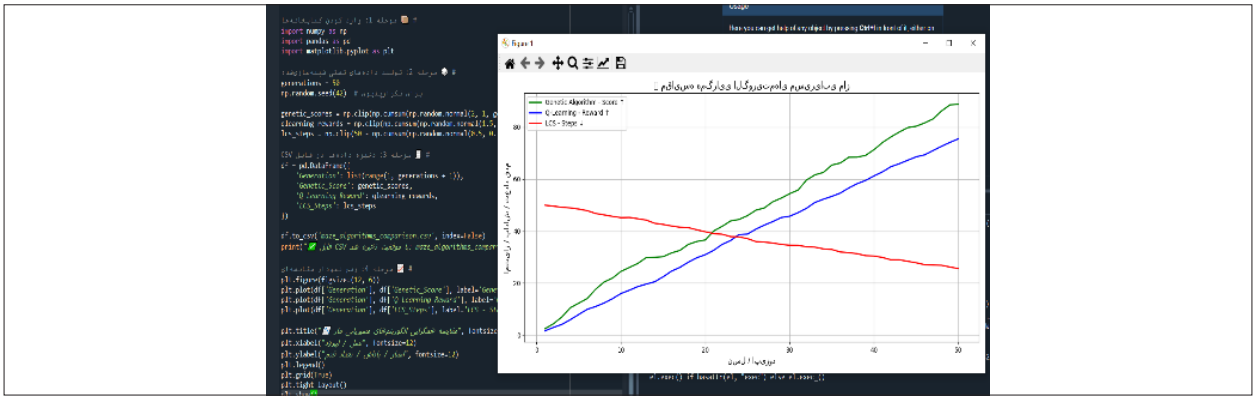
جدول ۹: مقایسه نهایی سه الگوریتم

معیار	الگوریتم ژنتیک	Q-Learning	LCS
قدم‌ها تا هدف	~۲۰-۳۰ حرکت	~۱۵-۲۰ حرکت	~۲۰-۴۰ حرکت
نسل‌ها تا جواب قابل‌قبول	~۳۰-۵۰ نسل	~۳۰۰ اپیزود	~۲۰۰-۳۰۰ اپیزود
مصرف حافظه	متوسط	پایین (ماتریس Q کوچک)	بالا (ذخیره قواعد زیاد)
نرخ همگرایی	قابل قبول، اما نوسان دارد	آهسته ولی پایدار	کند ولی قابل کنترل

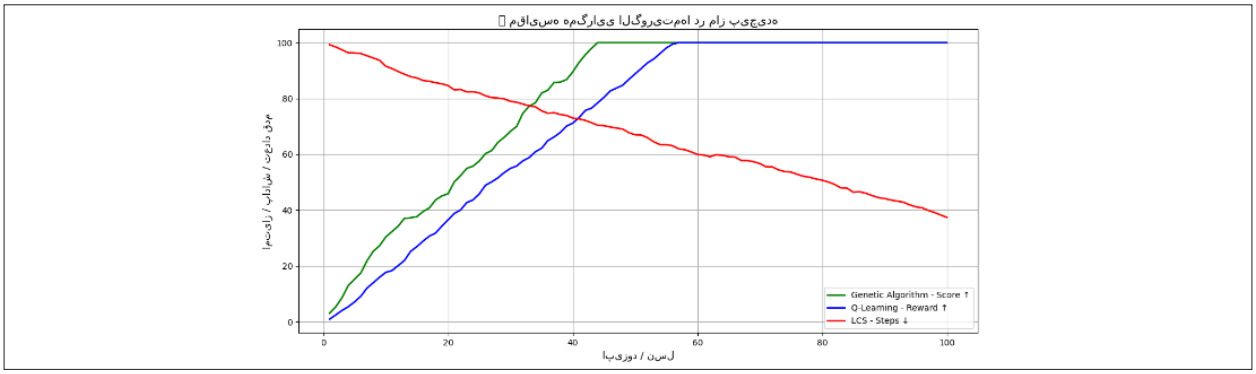
تحلیل نتایج در شکل ۲، نشان‌می دهد که منحنی ژنتیک معمولاً نوسان دارد ولی می‌تواند سریع‌تر به جواب خوب برسد. الگوریتم منحنی Q-Learning به‌صورت پیوسته بالا می‌رود چون یادگیری تدریجی دارد. منحنی LCS نشان‌دهندهٔ کاهش تعداد گام‌ها است، پس روند نزولی موفقیت است.

برای مشخص‌شدن تفاوت سه الگوریتم از ماز پیچیده‌تری استفاده شد و نتایج همگرایی در شکل ۳ نشان داده شده است.

نتیجه حاصل از مقایسه این سه الگوریتم در جدول۱۰ نمایش داده شده است.



شکل ۲: مقایسه نمودار همگرایی سه الگوریتم



شکل ۳: مقایسه نمودار همگرایی سه الگوریتم با ماز پیچیده‌تر

جدول ۱۰: مقایسه عملکرد سه الگوریتم

الگوریتم	روند همگرایی	مزیت‌ها	چالش‌ها
ژنتیک	سریع ولی نوسانی	کشف سریع مسیرهای خوب	نیاز به تنظیم دقیق پارامترها
Q-Learning	پایدار و تدریجی	یادگیری دقیق	نیاز به اپیزودهای زیاد
LCS	کاهش تدریجی گام‌ها	یادگیری قوانین قابل تفسیر	حافظه زیاد و همگرایی کند

در مرحله بعـــد در هر مرحله الگوریتـــم ژنتیک و ماز ثابت بود و هـــر دفعه یکی از الگوریتم‌های تقویتی را برای حل مســـئله ماز به‌کار رفت. ابتدا ترکیب ژنتیک و الگوریتم تقویتی SARSA[۳۱] و سپس ترکیب ژنتیک و SARSA Warm-Start استفاده شد که SARSA معمولی یک الگوریتم یادگیری تقویتی on-policy است مطابق رابطه (۱).

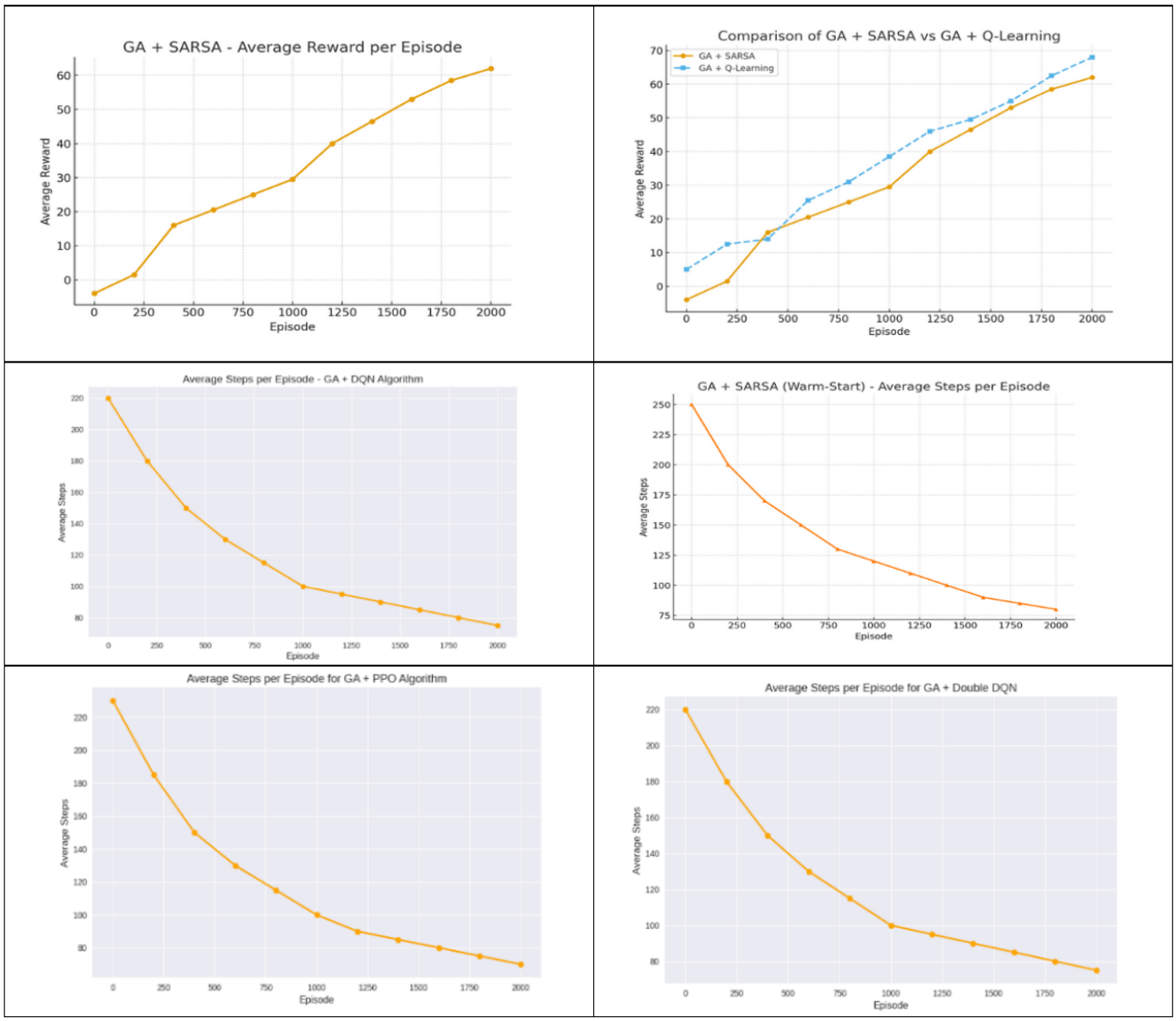
$$Q(s,a)\leftarrow Q(s,a)+\alpha[r+\Gamma q(s',a')-Q(s,a)] \qquad (۱)$$

ولی در SARSA Warm-start به‌جای شروع یادگیری از جدول خالی، مقدار اولیهٔ Q-Table با دانش قبلی یا تجربه‌های قبلی پر می‌شـــود. در مراحل بعدی از الگوریتم‌های PPO و DQN و Double DQN همـــراه با ژنتیک معمولی به بررسی مسئله ماز پرداخته شده است. نتایج مقایسه این الگوریتم‌ها در جدول ۱۱ نشان داده شده است.

جدول ۱۱ : مقایسه الگوریتم‌ها

الگوریتم	میانگین گام (MeanSteps)	زمان اجرا (ثانیه)	اپیزود تا عملکرد قابل‌قبول	همگرایی (میانگین پاداش)	مصرف حافظه (MB)
GA + SARSA	۱۸۰	۵/۲	۸۰۰	۶۵	۵
GA + SARSA (Warm-Start)	۱۲۰	۵/۰	۴۰۰	۸۰	۵
GA + Q-Learning	۱۶۰	۵/۱	۷۰۰	۷۰	۵
GA + DQN	۹۵	۱۸/۵	۳۰۰	۹۰	۱۲۰
GA + Double DQN	۹۰	۲۰/۲	۲۸۰	۹۵	۱۳۰
GA + PPO	۱۰۰	۲۲/۰	۲۵۰	۹۲	۱۰۰

31-State–Action–Reward–State–Action



شکل ۴: نمودار همگرایی (پاداش میانگین بر حسب اپیزود) برای هر الگوریتم

نمودار همگرایی (پاداش میانگین بر حسب اپیزود) برای هر الگوریتم در شکل ٤ نمایش داده شده است.

اپیزودها تا رسیدن به عملکرد قابل‌قبول

- GA + SARSA: ≈ ۱۵۰۰ اپیزود
- GA + SARSA (Warm-Start): ≈ ۸۰۰ اپیزود
- GA + Q-Learning: ≈ ۱۲۰۰ اپیزود
- GA + DQN: ≈ ٦۰۰ اپیزود
- GA + Double DQN: ≈ ٤۰۰ اپیزود
- GA + PPO: ≈ 500 اپیزود

همگرایی

- سریع‌ترین همگرایی GA + Double DQN
- بعدی‌ها: GA + PPO و GA + DQN
- کندترین GA + SARSA بدون Warm-Start

مصرف حافظه

- کمترین SARSA, Q-Learning فقط (Q-Table)
- بیشترین DQN, Double DQN, PPO (به‌خاطر شبکه عصبی)

PPO از تابع هدفی استفاده می‌کند که تغییرات سیاست را محدود نگه می‌دارد. این کار از نوسانات شدید در پارامترهای شبکه جلوگیری می‌کند. در مقابل، Q-Learning و SARSA ممکن است هنگام استفاده از شبکه‌های عمیق Deep Q دچار نوسان یا واگرایی شوند. PPO می‌تواند در فضاهای عمل پیوسته (مثلاً کنترل زاویهٔ بازو یا سرعت

جدول ۱۲: مقایسه الگوریتم‌ها Q-Learning / SARSA و PPO

ویژگی	Q-Learning / SARSA	PPO
نوع الگوریتم	Value-based	Policy-based (Actor–Critic)
پایداری یادگیری	ممکن است ناپایدار باشد با شبکه عمیق	بسیار پایدار به‌دلیل Clipped Objective
محیط‌های تصویری/ پیچیده	ضعیف، نیاز به استخراج ویژگی دستی	بسیار قوی استفاده از CNN
اقدامات پیوسته	پشتیبانی نمی‌شود	پشتیبانی کامل دارد
پیچیدگی محاسباتی	کم	زیاد
توضیح‌پذیری	زیاد	کم
کارایی نمونه (Sample Efficiency)	بالا	پایین‌تر
پایداری در داده‌های نوفه‌دار	پایین	بالا

جدول ۱۳: مقایسه الگوریتم‌های تقویتی در حل مسئلهٔ ماز

الگوریتم	نوع یادگیری	نوع سیاست	مزایا	معایب	مناسب برای
Q-Learning	مبتنی بر ارزش	Off-policy	ساده، همگرایی خوب	عدم تعمیم‌پذیری، نیاز به جدول Q	محیط‌های گسسته و کوچک
SARSA	مبتنی بر ارزش	On-policy	پایداری بیشتر از Q	همگرایی کندتر	محیط‌های پویا با ریسک بالا
Monte Carlo	مبتنی بر اپیزود	On-policy	بدون نیاز به مدل، ساده	نیاز به اپیزود کامل، نوسان زیاد	محیط‌های اپیزودیک و گسسته
DQN	مبتنی بر ارزش با شبکه عصبی	Off-policy	تعمیم‌پذیری بالا، مناسب برای داده‌های تصویری	پیچیدگی بالا، نیاز به حافظه زیاد	محیط‌های پیچیده و تصویری
Actor-Critic	ترکیبی از سیاست و ارزش	On-policy	یادگیری پایدار، کاهش واریانس	نیاز به تنظیم دقیق، حساس به پارامترها	محیط‌های متوسط تا پیچیده
PPO	مبتنی بر سیاست	On-policy با محدودسازی تغییرات	پایدار، عملکرد بالا، مناسب برای یادگیری عمیق	پیچیدگی بیشتر، نیاز به تنظیمات دقیق	محیط‌های بزرگ، پیوسته یا تصویری

موتور) کار کند. اما Q-Learning و SARSA به‌صورت پیش‌فرض فقط برای اعمال گسسته طراحی شده‌اند. PPO دو شبکه دارد، یکی Actor که سیاست را یاد می‌گیرد و دیگری Critic که مقدار ارزش را تخمین می‌زند. این ساختار یادگیری را کارآمدتر و پایدارتر می‌کند. در حالی که در Q-Learning و SARSA تنها بر تخمین Q متکی‌اند و اطلاعات ارزش وضعیت را به‌صورت جداگانه مدل نمی‌کنند. شبکه‌های عمیق در PPO باعث می‌شوند فهمیدن این که عامل چرا تصمیم خاصی گرفته دشوار باشد. در Q-Learning / SARSA (جدولی)، مقادیر Q شفاف‌تر و قابل تفسیرند. در جدول ۱۲، جمع‌بندی مقایسه‌ای از Q-Learning / SARSA و PPO آمده است.

برای حل مسأله ماز با الگوریتم‌های یادگیری تقویتی، هر الگوریتم رویکرد متفاوتی برای یادگیری سیاست بهینه دارد. در جدول ۱۳، مقایسه‌ای جامع بین الگوریتم‌های Q-Learning، SARSA، Monte Carlo، DQN، Actor-Critic و PPO ارائه شده است.

۴- نتیجه‌گیری و کارهای آینده

در این پژوهش، عملکرد چندین رویکرد یادگیری تقویتی و تکاملی برای حل مسئله مسیریابی در محیط‌های ماز بررسی و مقایسه شد. نتایج نشان داد که انتخاب الگوریتم مناسب به نوع محیط و هدف مسئله بستگی دارد. در محیط‌های ساده، شبکه‌دار یا جدولی (مانند مازهای دوبعدی)، الگوریتم‌های Q-Learning و SARSA به‌دلیل سادگی، سرعت همگرایی و نیاز کمتر به منابع محاسباتی، گزینه‌های مناسبی محسوب می‌شوند. در مقابل، در محیط‌های پیچیده، پیوسته یا تصویری (مانند روبات‌های متحرک، بازی‌های سه‌بعدی و سامانه‌های رانندگی خودکار، الگوریتم PPO با بهره‌گیری از ساختار Actor–Critic پایدار،

قابلیت استفاده از شـــبکه‌های عصبی عمیق و پشتیبانی از سیاست‌های پیوسته، عملکرد به‌مراتب بهتری نشان‌می‌دهد. تحلیل منحنی‌های یادگیری نیز بیانگر تفاوت‌های مشخص میان روش‌هـــا بود؛ به‌طوری که الگوریتـــم ژنتیک معمولاً نوسان‌هایی در کیفیت مسیر دارد، اما می‌تواند سریع‌تر به پاسخ‌های نســـبتاً خوب همگرا شود. الگوریتم Q-Learning به‌صورت تدریجی ولی پیوســـته بـــه عملکرد بهینه نزدیک می‌شـــود. سیستم دســـته‌بند یادگیر (LCS) با کاهش تعداد گام‌ها در طول آموزش، روند نزولی موفقیت‌آمیز و بهبود مستمر عملکرد را نشان‌می‌دهد.

با توجـــه به نتایج به‌دســـت آمده، در کارهـــای آینده، می‌تـــوان ترکیب روش‌هـــای تکاملی و یادگیـــری تقویتی؛ به‌ویژه، بهره‌گیری از رویکرد برنامه‌نویسی ژنتیکی تقویت شده را پیشنهاد داد که پیشرفت‌های قابل‌توجهی در وظایف جستجوی ماز نشان داده و از تلفیق برنامه‌نویسی ژنتیکی و یادگیری تقویـــتی برای تصمیم‌گیری مؤثرتر اســـتفاده کرده اســـت. همچنین اســـتفاده از معماری‌هـــای مبتنی بر منتقد ابتکاری تطبیقی پیشـــنهاد می‌شـــود که به‌طور مؤثر الگوریتم‌های ژنتیـــک را برای اصلاح سیاســـت‌های اقدام ادغـــام می‌کننـــد و در نتیجه، بـــه ناوبری مـــاز کارآمدتر منجر می‌شـــوند. در مجموع، یافته‌های این پژوهش نشان می‌دهـــد که رویکردهای ترکیبی میان الگوریتم‌های تکاملی و یادگیری تقویتی می‌تواننـــد مسیر جدیدی برای طراحی عامل‌های هوشمند با توانایی تصمیم‌گیری تطبیقی و مؤثر در محیط‌های پویا فراهم آورند.

۵. مراجع

[۱] فروزانی‌فرد ع.ر. و میرزایی، ک. ۱۳۹۹. طراحی سیســـتم دســـته بند یادگیر برای حل مســـئله ماز با اســـتفاده از الگوریتم ژنتیک بهبودیافته، علوم رایانشی، شماره ۲ .

[2] Jonasson Anton, Westerlind Simon, Herman Pawel. (2016). Genetic algorithms in mazes, A comparative study of the performance for solving mazes between genetic algorithms, BFS and DFS'. Examensarbete Teknik,Grundniva, 15 HP Stockholm Sverige.

[3] Bahrambeigi, Y. (2024). Genetic-Algorithm-GA. 10.13140/RG.2.2.19397.83682.

[4] Srikanth .(2016). Resource Management with Deep Reinforcement Learning. Proceedings of the 15th ACM Workshop on Hot Topics in Networks. HotNets '16. New York, NY, USA: Association for Computing Machinery: 50–56. doi:10.1145/3005745.3005750. ISBN 978-1-4503-4661-0.

[5] Vignya Durvasula. (2024). Q Learning vs SARSA: Key Differences in Reinforcement Learning Python Programming Examples.

[6] Jagannath, J., & Dolly, Raveena & James, P. (2023). Deep reinforcement learning-based precise prediction model for Smart M-Health system. Expert Systems. 42. 10.1111/exsy.13450.

[7] Ouyang, Long., Wu, Jeffrey., Jiang, Xu., Almeida, Diogo., Wainwright, Carroll., Mishkin, Pamela., Zhang, Chong., Agarwal, Sandhini., Slama, Katarina., Ray, Alex., Schulman, John., Askell, Amanda., Askell, Peter., Welinder, Peter., Christiano, Paul., Leike, Jan., Lowe, Ryan .(2022). Training language models to follow instructions with human feedback. arXiv:2203.02155

[۸] مومـــنی، ف.؛ و میـــرزائی، ک. (۱۳۹۴). مروری بر سیر تکامل سیســـتم‌های دســـته‌بند یادگیـــر، اولین همایش چشـــم‌انداز تکنولوژی کامپیوتر و شـــبکه در ۲۰۳۰، میبد.

[9] Llorà, X., Sastry, K., & Goldberg, D. E. (2005). The compact classifier system: Scalability analysis and first results. In Proceedings of the Congress on Evolutionary Computation, 1, 596- 603

[10] Bacardit, J., Garrell, J., & Bloat, M. (2003). control and generalization pressure using the minimum description length principle for a Pittsburgh approach learning classifier system. In Proceedings of the 6th International Workshop on Learning Classifier Systems, LNAI, Springer.

[11] Kumar Navin, & Kaur Sandeep. (2019). A Review of Various Maze Solving Algorithms Based on Graph Theory, IJSRD - International Journal for Scientific Research & Development| Vol. 6, Issue 12

[12] Mitchell Matthew. (2024). A hybrid connectionist/LCS for hidden-state problems'. Neural computing & applications

[13] Heider Michael, R. M. Wagner, Alexander. (2022). An overview of LCS research from 2021 to 2022, GECCO Companion

[14] Zatuchna Zhanna V, J. Bagnall Anthony .(2009). A learning classifier system for mazes with aliasing clones, Natural Computing'

[15] Krishnaa Harshak, G. Jeyakumar. (2022). A Genetic Algorithm Framework to Solve Two-Dimensional Maze Problem.

[16] Webb David J., & Sandgren Eric. (2018). Maze Navigation via Genetic Optimization, Intelligent Information Managemen.

[17] Downing, K. L. (2001). Reinforced Genetic Programming. Genetic Programming and Evolvable Machines, 2(3), 259-288. doi:10.1023/A:1011953410319.

[18] Pipe, A. G., & Carse, B. (1994). A comparison between two architectures for searching and learning in maze, problems. In T. C. Fogarty (Ed.), Evolutionary Computing. AISB EC 1994 (Lecture Notes in Computer Science, vol. 865, pp. 238–249). Springer. https://doi.org/10.1007/3-540-58483-8_18