

فصل اول

تعاریف اولیه در هوش مصنوعی

۱-۱-۱ AI چیست؟

تعاریف هوش مصنوعی در شکل ۱-۱ نشان داده شده است. این تعاریف در دو بعد اصلی متمایز می‌شوند. آنهایی که در بالا هستند به پردازشهای فکری و استدلالی پرداخته و پایینی‌ها بر پردازشهای رفتاری متمرکز شده‌اند. همچنین تعاریف سمت چپ بیانگر ارائه انسانی است و آنهایی که در سمت راست هستند مفهوم ایده‌آل هوشمندی را بیان می‌کنند که ما آن را منطقی بودن (rationality) می‌نامیم. سیحتم منطقی است اگر عمل درست را انجام دهد. این تقسیم‌بندی چهار هدف را برای هوش مصنوعی مشخص می‌سازد که بدنبال آنها هستیم و این موضوع در عنوان شکل ۱-۱ نشان داده شده است: **پردازش فکری و استدلالی اراده انسانی**

«مطالعه تواناییهای ذهنی از طریق مدلهای کامپیوتری» (چارنیاک و مک در موت، ۱۹۸۵) (Charniak, MC Dermott)	«کوشش جدید هیجان‌انگیز برای ساختن کامپیوترهایی که فکر کنند، ماشینهایی با قدرت تفکر و با حس کامل» (هاوگلدن، ۱۹۸۵) «اتوماسیون عملیاتی که با اعمال تفکر انسان نظیر تصمیم‌گیری، حل مسئله، یادگیری مربوط می‌شوند» (بلمن، ۱۹۷۸) (Bellman)
«محیطه‌ای از مطالعه که رفتار هوشمند را تحت عنوان فرآیندهای کامپیوتری شرح داده و مورد رقابت قرار دهد» (شالکوف، ۱۹۹۰) (Schalkoff)	«هنر خلق ماشینهایی که توانایی انجام عملیاتی داشته باشند که آن عملیات توسط انسان نیاز به هوشمندی داشته باشند» (کروزویل، ۱۹۹۱) (kurzweil)
«شاخه‌ای از علم کامپیوتر که با اتوماسیون رفتار هوشمند مربوط می‌شود» (لجر، استبل‌فیلد) (luger, Stubblefield)	«مطالعه بروی چگونگی ساخت کامپیوترهایی که کارها را در هر لحظه بهتر از انسانها انجام دهند» (ریچ و نایت، ۱۹۹۱) (Rich, knight)

شکل ۱-۱: تعاریفی از AI که به چهار طبقه، تقسیم شده‌اند.

سیستم‌هایی که به طور منطقی فکر می‌کنند	سیستم‌هایی که مانند انسان فکر می‌کنند
سیستم‌هایی که به طور منطقی عمل می‌کنند	سیستم‌هایی که مانند انسان عمل می‌کنند

از نظر تاریخی، تمامی این چهار رهیافت دنبال شده‌اند. همان طور که ممکن است حدس زده شود، تنش بین رهیافت انسانی و رهیافت منطقی وجود دارد. نگرش مبتنی بر انسانیت علمی تجربی است که شامل فرضیات و نائیر آنها توسط تجربیات است. نگرش منطقی ترکیبی از ریاضیات و مهندسی است. دانشمندی که به هر گروه تعلق دارند گاهی به کار دیگری خرده می‌گیرند، اما واقعیت آن است که هر گروه اطلاعات با ارزشی را ایجاد کرده است. بیایید به هر کدام از آنها نگاه دقیق‌تری داشته باشیم.

۱-۲- انسان گونه عمل کردن: رهیافت آزمون تورینگ (Turing)

تست تورینگ که توسط آن تورینگ (۱۹۵۰) پیشنهاد شد، تعریف علمی رضایت بخشی از هوش را ایجاد کرده است. تورینگ رفتار هوشمند را به عنوان توانایی رسیدن به سطح ارائه انسانی در تمامی وظایف ادراکی تعریف کرد که حتی قادر به فریفتن یک محقق نیز باشد. تستی که او پیشنهاد کرد آن بود که کامپیوتر می‌بایست توسط فردی که از طریق تله تایپ (tele type) به آن دسترسی دارد مورد تحقیق قرار گیرد و زمانی در آزمون موفق می‌شود که محقق نتواند دریابد در آن طرف انسان قرار دارد یا کامپیوتر. اما اکنون، برنامه‌ریزی کامپیوتری که بتواند این تست را انجام دهد کار زیادی می‌برد. کامپیوتر مذکور باید قابلیت‌های زیر را داشته باشد:

- پردازش زبانی طبیعی (natural language processing) تا قادر به محاوره به زبان انگلیسی (یا زبان انسانی دیگر) گردد.

- باز نمایی دانش (Knowledge representation) تا اطلاعات تولید شده قبل یا در حین آزمون را ذخیره سازد.

- یادگیری ماشینی (machine learning) تا خود را با شرایط تازه وفق دهد و الگوها را کشف و برون ریزی کند.

تست تورینگ اندیشمندان از ارتباط فیزیکی مستقیم بین کامپیوتر و محقق اجتناب می‌کند، زیرا شبیه‌سازی فیزیکی فرد برای هوشمندی ضروری نیست. به هر حال، تست تورینگ کلی مشهور شامل یک سیگنال ویدیویی است که محقق بتواند از طریق آن قابلیت‌های ادراکی موضوع را مورد آزمون قرار دهد. برای قبول شدن در تست تورینگ کلی کامپیوتر به موارد زیر احتیاج دارد.

- بینایی ماشین درک اشیاء

- روباتیک به منظور حرکت آنها

در AI، سعی زیادی برای موفقیت در تست تورینگ انجام نگرفته است. عقیده عمل کردن نظیر انسان زمانی مطرح شد که برنامه‌های AI می‌بایست با انسان در کنش متقابل قرار می‌گرفتند، همانند زمانی که سیستم خبره (expert system) چگونگی حصول تشخیص نهایی را تعریف می‌کند و یا سیستم پردازش زبان طبیعی مکالمه‌ای با کاربر خود دارد. بازنمایی و استدلال در چنین سیستمی ممکن است بر مبنای مدل انسانی باشد و یا نباشد.

۱-۳- انسانی فکر کردن: رهیافت مدل‌سازی شناختی

اگر قصد آن را داریم بگوییم یک برنامه همانند انسان فکر می‌کند، باید راهی برای درک چگونگی فکر انسان داشته باشیم. پس باید راهی به درون چگونگی عملکرد افکار انسان پیدا کنیم. برای انجام این امر دو راه وجود دارد: از طریق درون‌گرایی (سعی در بدست آوردن افکار خودمان) یا از طریق تجارب روانشناسی. اگر قادر به ایجاد تئوری دقیقی درباره ذهن باشیم، آنگاه قادر خواهیم بودن این تئوری را به برنامه کامپیوتری تبدیل کنیم. اگر ورودی و خروجی و زمانبندی با رفتار انسان تطبیق داشته باشد، گواهی بر آن دارد که برخی از مکانیزم‌های برنامه ما در انسان هم عمل خواهد کرد. پری، مثال Newell و Simon که سیستم حل‌کننده عمومی مسائل (GPS) را طراحی نموده‌اند، از اینکه برنامه‌شان به درستی مسائل را حل کند، راضی نبودند. آنها بیشتر بر روی تعقیب مراحل حل مسائل توسط انسان تمرکز کرده بودند. این طرز فکر با محققان دیگر در همان زمان در تضاد بود. (مانند وانگ ۱۹۶۰) که بیشتر در پی یافتن پاسخ درست بدون توجه به رفتار انسانی بود. حیطه علوم شناختی (cognitive) مدل‌های کامپیوتری از AI و همچنین تکنیک‌های روانشناختی را گرد هم می‌آورد تا بتواند تئوریهای دقیقی از کارکرد ذهن انسان بدست آورد.

۱-۴- منطقی فکر کردن: قوانین رهیافت تفکر

ارسطو (فیلسوف شهر یونان باستان)، یکی از اولین کسانی بود که سعی داشت تا رمز «تفکر درست» را بگشاید که فرآیند استدلال غیر قابل نفی است. قیاس (Syllogism) معروف وی الگوهایی برای ساختار توافقی ایجاد کرد که همواره نتایج صحیحی به اندازه مقدمات صحیح بدست می‌آورد. برای مثال، «سقراط انسان است، تمام انسانها می‌میرند، پس سقراط خواهد مرد» این طرز فکر عملیات ذهن را سازمان می‌دهد و حیطه منطق (logic) را پایه‌گذاری می‌کند.

توسعه منطق رسمی (formal) در قرون نوزده و بیست دستور زبان دقیقی برای جملاتی در مورد تمامی انواع اشیاء در جهان و رابطه بین آنها ایجاد کرد. (برخلاف حساب معمولی که فقط برای جملات تساوی اعداد دستور زبان دارد) در سال ۱۹۶۵، برنامه‌هایی وجود داشت که قادر بود با وجود حافظه و زمان کافی، شرحی از مسئله به زبان منطق دریافت کند و اگر راه‌حلی داشته باشد آن را پیدا کند. (اگر راه‌حلی وجود نداشت، برنامه هرگز از جستجو دست بردار نبود). این رسم منطقی گرایی در هوش مصنوعی امید ایجاد ساخت برنامه‌هایی برای سیستم‌های هوشمند را ایجاد می‌کند.

دو مشکل عمده در این راه وجود داشت. اول اینکه دریافت دانش غیر رسمی (informal) و تبدیل آن به شکل رسمی توسط علائم منطقی ساده نیست، علی‌الخصوص زمانی که دانش ما از درجه اطمینان کمتر از ۱۰۰٪ برخوردار باشد. دوم اینکه تفاوت عمده‌ای بین قادر به حل مسئله بودن در «اصول» و انجام آن در عمل وجود دارد. حتی مسائلی که یک دوجین واقعیت را با خود یدک می‌کشند می‌توانند بپراحتی هر کامپیوتری را از نظر محاسباتی به بن بست برسانند مگر اینکه راهنمایی‌هایی را برای اعمال مراحل استدلال وجود داشته باشد اگر چه هر دوی این موانع با هر تلاشی برای ساخت سیستم‌های استدلال‌پذیر محاسباتی برخورد می‌کند، اما ابتدا در سمت منطق‌گرایی ظاهر خواهد شد زیرا توانایی نمایش و استدلال سیستم‌ها به خوبی تعریف و درک شده‌اند.

۵-۱- منطقی عمل کردن: رهیافت عامل منطقی

رفتار منطقی بدین معناست که با داشتن عقیده واحد به هدف واحدی برسیم. یک عامل، در اصل چیزی است که ابتدا درک می‌کند و سپس عمل می‌کند. (البته ممکن است استفاده از این کلمه غیر معمول باشد اما شما در اینجا به آن عادت می‌کنید) در این رهیافت، AI به عنوان مبنای عامل‌های منطقی بکار برده می‌شود. در نگرش «قوانین تفکر» هوش مصنوعی، تأکید عمده بر روی استنتاج‌های صحیح بوده است. تولید استنتاج‌های صحیح گاهی قسمتی از وجود یک عامل عقلانی است، زیرا یک راه برای رفتار عقلانی، استدلال منطقی است که منتهی به نتیجه منطقی می‌شود که در نهایت هدفی را برآورده می‌سازد. از سوی دیگر، استنتاج منطقی تمام عقلانی بودن نیست زیرا شرایط خاصی وجود دارد که هیچ چیز صحیحی برای انجام وجود ندارد ولی هنوز چیزی باید انجام شود. همچنین راه‌هایی از رفتار وجود دارند که نمی‌توانند به صورت استدلالی بیان شوند تا قادر به ذکر استنتاج گردند. برای مثال، عقب کشیدن دست از یک اجاق داغ، عکس‌العملی که موفق‌آمیزتر از یک عمل آهسته، بعد از تفکر سنجیده است.

تمامی «مهارت‌های شناخت» که برای آزمون تورینگ مورد نیاز است، برای انجام فعالیت‌های منطقی وجود دارند. از این رو، ما نیازمند بازنمایی دانش و استدلال با آن هستیم زیرا ما قادر می‌شویم از میان شرایط مختلف به تصمیم مطلوبی برسیم. ما نیازمند ادراک بصری نه فقط برای لذت از دیدن هستیم، بلکه می‌خواهیم ایده بهتری از نتیجه احتمالی یک عمل بدست آوریم، برای مثال دیدن یک غذای خوشمزه به ما کمک می‌کند تا به طرف آن حرکت کنیم.

مطالعه AI به عنوان طراحی عامل منطقی دارای دو مزیت است، اول اینکه، از رهیافت «قوانین تفکر» عمومی‌تر است، زیرا استنتاج صحیح تنها فرآیند صحیحی برای حصول منطقی بودن است و نه یک شرط لازم. دوم اینکه، پیشرفت علمی بسیار قانون‌پذیرتر از رهیافت‌هایی است که خاص بخوبی قابل تطبیق است و محصول از یک پردازش تکامل یافته ناشناخته و پیچیده است که هنوز ممکن است دور از دسترس باشد. این کتاب بروی قوانین عمومی منطقی و بر روی مولفه‌های ساخت آنها تمرکز می‌کند. خواهیم دید که بر خلاف ظاهر ساده‌ای که ممکن است مسئله با آن بیان شود، برای حل آن باید نکات متنوعی مطرح شوند. این نکته مهم را به خاطر بسپارید: رسیدن به منطق کامل و انجام عمل صحیح در محیط‌های بفرنج همواره امکان‌پذیر نخواهد بود. به هر حال، در بیشتر این کتاب، مفروضات کاری را که تصمیم‌گیری کامل را درک می‌کند و جای مناسبی برای آغاز است را قبول می‌کنیم.

۶-۱- سیستم‌های تولیدی (production system)

از آنجا که بخش مهمی از هوش مصنوعی سنتی شامل جستجو و روش‌های مربوط به آن بوده است می‌توان برنامه هوشمند را بر این مبنای طراحی کرد و تحت این شرایط سیستم تولیدی ایجاد شده است. این سیستم شامل عناصر زیر است.

۱- مجموعه‌ای از قوانین، که شامل بخش مقدم (شرط) و تالی (عمل) است. بخش مقدم شرایط برقراری قانون را معین کرده و تالی عملیات مربوط به زمان برقراری شرایط را نشان می‌دهد.

۲- یک یا چند بانک اطلاعاتی که در آن اطلاعات مناسب برای کار خاص تعبیه شده است. برخی از بخش‌های بانک دانشی بوده ولی برخی دیگر از بخش‌ها مربوط به حل یک مسئله خاص خواهد بود. اطلاعات موجود در این بانک به هر شکل ممکن سازماندهی خواهد شد.

۳- استراتژی کنترل که مشخص می‌کند با چه توالی قوانین با بانک اطلاعاتی مقایسه می‌شود و راهی برای حل تناقض در آن پیش‌بینی شده است منظور از تناقض زمانیست که چند قانون همزمان مقدمشان برقرار می‌شوند.

سیستم‌های تولیدی یک دیدگاه عام در طراحی است و می‌تواند سامی حیف گسترده‌ای از کاربردها و راه‌حل‌ها گردد.

مهمترین خصیصه یک استراتژی کنترل مناسب، ایجاد حرکت است. خصیصه دوم سیستماتیک بودن آن است، یعنی بتواند حالات موجود در مسئله را تماماً بررسی کرده تا به پاسخ نهائی برسد.

سیستم‌ها تولیدی را به کلاس‌هایی تقسیم می‌کنند. سیستم تولیدی یکنوا (monotonic) سیستمی است که در آن اعمال یک قانون هرگز نمی‌تواند اعمال بعدی قانون دیگری را که در حال حاضر قابل اعمال است، نفی کند. سیستم تولیدی نیمه تبدیلی (partially commutative) سیستمی است که در آن اگر دنباله خاص از قوانین بتواند X را به Y تبدیل کند، هر جایگشت (permuntation) از آن قوانین نیز کاربردی داشته باشند. سیستم تولیدی تبدیلی نامیده می‌شود اگر یکنوا و نیمه تبدیلی باشد.

۷-۱- گراف فضای حالت (state space graph)

قبل از آنکه به مفهوم گراف فضای حالت و کاربرد آن در هوش مصنوعی بپردازیم، ابتدا لازم است تعاریف اولیه در مورد گراف را که در این بحث مورد استفاده قرار می‌گیرند، مرور شوند.

گراف: مجموعه‌ای از گره‌ها که متناهی هستند، مجموعه‌ای از لبه‌ها که زوج گره‌ها را به هم مرتبط می‌کنند و رابطه ریاضی که بیان می‌کند هر لبه به یک زوج گره متصل است.

گراف جهت دار: در این گراف لبه‌ها دارای جهت هستند (هر لبه‌ای) برای مثال بین زوج گره n_i و n_j ممکن است لبه‌ای باشد ولی لزوماً بین n_j و n_i ممکن است لبه‌ای وجود نداشته باشد. حال اگر لبه مستقیمی n_j را به n_i متصل کند، می‌گوئیم گره n_j والد و گره n_i فرزند دیگری می‌باشد. حال اگر n_j والد n_k و n_k باشد، آنگاه می‌گوئیم n_k و n_j خواهر - برادر یکدیگر هستند.

گراف ریشه‌دار: دارای گره‌ای همانند n_s یا ریشه است که این گره واحد بوده و از آن تمامی گره‌های دیگر، مسیری وجود دارد. بدیهی است گره برای گراف‌های جهت‌دار فاقد والد است.

دنباله‌ای همانند $(n_1, n_2, n_3, \dots, n_n)$ از گره‌های گراف که هر زوج گره n_i, n_{i+1} در آن مجاور هستند مسیر (path) نامیده می‌شود.

در گراف ریشه‌دار و در یک مسیر معین، گره‌ای همانند n_k نیا (ancestor) برای تمامی گره‌های ما بعد خود نامیده می‌شود. متقابلاً هر گره‌ای در مسیر برای گره‌ای ما قبل آن نواده (descendent) نام دارد.

اگر در مسیر، یک گره بیش از یکبار تکرار شده باشد، مسیر دارای حلقه (cycle) خواهد شد.

درخت (tree) گرافی است که بین هر زوج گره آن دقیقاً یک مسیر وجود داشته باشد. به زبان دیگر درخت فاقد حلقه است.

در گره در گراف همبند (Connected) نامیده می‌شوند، اگر حداقل بین آنها یک مسیر وجود داشته باشد. حال اگر کلیه زوج گره‌های اختیاری در گراف همبند باشند، گراف نیز همبند خواهد بود.

گراف فضای حالت (state space graph) به وسیله یک تائی بصورت

$G=(V,E,S,D)$ نشان داده می‌شود که در آن:

V مجموعه گره‌های یا وضعیت گراف است.

E مجموعه لبه‌های متعلق به گراف است.

S زیر مجموعه غیر تهی از V بوده و شامل گره (گره‌های) آغازین می‌باشد.

D زیر مجموعه غیر تهی از V بوده و شامل گره (گره‌های) هدف می‌باشد.

اما این چهار عضو چه معنی می‌دهند؟ V یا گره‌های گراف وضعیت‌های حاکم به محیط مسئله را معین می‌کند، یعنی هر وضعیت پایدار نسبی در مسئله بوسیله یک گره نمایش داده می‌شود. لبه‌های گراف یا E معرف گذر از یک وضعیت (گره) به وضعیت دیگر هستند. وضعیت یا وضعیت‌های آغازین، منظور نقطه یا نقاطی است که از آن مسئله شروع می‌شود و در شرح صورت مسئله معین شده است، متقابلاً وضعیت پایانی شامل وضعیتی است که هدف نامیده می‌شود و در آن نقطه راه‌حل به اتمام خواهد رسید (پاسخ مسئله).

نکته قابل توجه در این تعریف آنست که کدام گره هدف شناخته می‌شود؟ معمولاً تعیین گره هدف از طریق دو حالت صورت می‌گیرد اول آنکه تابعی معین کند که وضعیت جاری (بدون توجه به مسیر طی شده) هدف است و دوم آنکه مسیر طی شده معین کند که به هدف رسیده‌ایم.

برای تعریف گراف فضای حالت با شرح فوق به عوامل زیر نیاز داریم:

- وضعیت آغازین (initial state) که عامل خودش از بودن در آن آگاه است.
- مجموعه‌ای از عملیات ممکن، که برای عامل قابل دسترسی باشد. واژه عملگر (operator) بر تعریف یک عمل دلالت می‌کند، بدین معنی که پس از انجام یک عمل در یک وضعیت ویژه، عامل به کدام وضعیت بعدی خواهد رسید. در روش بیان دیگر از یک تابع ما بعد (successor) استفاده می‌شود.
- آزمون هدف (goal test)، که عامل می‌تواند در یک تعریف وضعیت منفرد آن را تقاضا کند تا تعیین گردد که آن حالت، وضعیت هدف است یا خیر. بعضی اوقات مجموعه صریحی از وضعیت‌های هدف ممکن وجود دارد و آزمون به سادگی چک می‌کند تا رسیدن به حالت هدف را تشخیص دهد. مواقعی پیش می‌آید که هدف توسط یک خاصیت مجرد معین می‌شود و نه با مجموعه‌ای شمارش‌پذیر از حالات برای مثال، در بازی شطرنج، هدف رسیدن به وضعیتی است که «کیش و مات» نامیده می‌شود، بدون توجه به اینکه طرف مقابل چه کاری انجام دهد.

بالاخره، موردی پیش می‌آید که ممکن است یک راه‌حل بر راه‌حل دیگر ترجیح داشته باشد حتی اگر هر دوی آنها به هدف برسند. برای مثال، ممکن است مسیری را با هزینه عملیات کمتر طی کنیم.

- تابع هزینه مسیر، تابعی است که برای هر مسیر، هزینه‌ای را در نظر می‌گیرد. در تمام موارد، در نظر می‌گیریم، هزینه یک مسیر، مجموع هزینه‌ای عملیات اختصاصی در طول مسیر است. تابع هزینه مسیر اغلب با حرف g مشخص می‌شود.

۸-۱ مسائل نمونه

گستره محیط عملیاتی که می‌تواند توسط مسائل تعریف شده، مشخص شود بسیار وسیع است. ما می‌توانیم بین مسائل ساده که برای نمایش روش‌های حل مسائل مختلف به کار برده می‌شوند و مسائل دنیای واقعی که مشکل‌تر هستند و راه‌حل آنها مورد توجه مردم است، تبعیض قائل شویم. بالطبع، مسائل ساده از شرح دقیق و مختصری برخوردار هستند. بدین معنا که این مسائل برای مقایسه کارایی الگوریتم‌هایی که توسط محققین طراحی می‌شوند، قابل استفاده هستند. از طرف دیگر، مسائل دنیای واقعی، توصیف جزئی توافقی ندارند.

مسائل ساده

معمای ۸

معمای ۸ نمونه‌ای است که در شکل زیر نشان داده شده و شامل یک صفحه 3×3 با ۸ مربع شماره دار در یک صفحه خالی است. هر مربع که مجاور خانه خالی است، می‌تواند به درون آن خانه برود. هدف رسیدن به ساختاری است که در سمت راست شکل نشان داده شده است. نکته مهم این است که بجای اینکه بگوییم «مربع شماره ۴ را به داخل فضای خالی حرکت بده» بهتر است بگوییم «فضای خالی جایش را با مربع سمت چپش عوض کند».

۵	۴	
۶	۱	۸
۷	۳	۲

Start State

۱	۲	۳
۸		۴
۷	۶	۵

Goal State

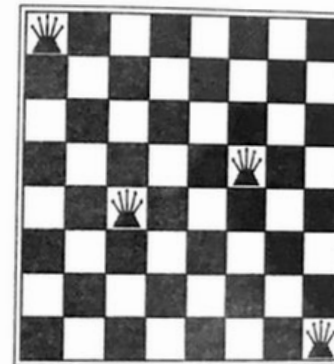
شکل ۱- نمونه مسئله معمای ۸

حال کار را بر مبنای تعاریف زیر دنبال می‌کنیم.
حالتها: توصیف وضعیت مکان هر ۸ مربع را در یک از ۶ خانه صفحه مشخص می‌کند. برای کارایی بیشتر، بهتر است که فضاهای خالی نیز ذکر شود.
عملگرها: فضای خالی به چپ، راست، بالا و پایین حرکت می‌کند.
آزمون هدف: وضعیت با ساختار هدف شکل مطابقت می‌کند.
هزینه مسیر: هر قدم ارزش ۱ دارد، بنابراین هزینه مسیر همان طول مسیر است.

معمای ۸ متعلق به خانواده Sliding - block Puzzle است. این کلاس عمومی به عنوان NP-complete شناخته می‌شود. بنابراین کسی انتظار ندارد که روشهایی پیدا شوند که بهتر از الگوریتم‌های جستجوی تعریف شده باشند. معمای ۸ و آشنای بزرگترش معمای ۱۵، مسائل استناداری برای الگوریتم‌های نوین جستجو در AI هستند.

مسئله ۸ وزیر

هدف از مسئله ۸ وزیر، قرار دادن ۸ وزیر بر روی صفحه شطرنج به صورتی است که هیچ وزیری نتواند به دیگری حمله کند. (یک وزیر در حالت افقی و عمودی و قطری حمله می‌کند) شکل زیر راه‌حلی برای این مسئله نشان می‌دهد که با شکست مواجه شده است. وزیر در سمت راست‌ترین ستون توسط وزیر در سمت چپ بالا مورد حمله قرار می‌گیرد.



شکل ۲- راه‌حل تقریبی برای مسئله هشت وزیر

اگر چه الگوریتم‌های هدف -ویژه کارآ برای این مسئله وجود دارند، اما این مسئله برای الگوریتم‌های جستجو جالب باقی می‌ماند. دو نوع بیان ریاضی اصلی وجود دارد. بیان افزایشی که با جایگزینی وزیرها، به صورت یکی یکی کار می‌کند و دیگری بیان وضعیت کامل که با تمام ۸ وزیر روی صفحه شروع می‌کند و آنها را حرکت می‌دهد. در هر دو مورد، هزینه مسیر به غیر از مرحله آخر، قابل توجه نیست. از این‌رو الگوریتم‌ها فقط روی هزینه جستجو مقایسه می‌شوند. بنابراین ما تست هدف و هزینه مسیر را به صورت زیر خواهیم داشت:

- آزمون هدف: ۸ وزیر روی صفحه، که با هم برخورد ندارند.
- هزینه مسیر: صفر.

همچنین حالات و عملگرهای متفاوتی نیز وجود دارند. بیان ریاضی ساده زیر را در نظر بگیرید:

- حالات: هر ترتیبی از صفر تا ۸ وزیر روی صفحه
- عملگرها: یک وزیر به هر خانه اضافه کنید.

در این فرمول ما ۶۴ امکان داریم. حقیقت اینکه جایگزینی یک وزیر در جایی که قبلاً مورد حمله قرار گرفته، نمی‌تواند کارآ باشد، چرا که جایگزینی‌های بعدی دیگر وزیرها، برخورد را از بین نمی‌برد. بنابراین ما باید مانند زیر عمل کنیم.

- حالات: ترتیب از صفر تا ۸ وزیر بدون هیچ برخورد.
 - عملگرها: یک وزیر را در خالی‌ترین ستون سمت چپ جایگزین کنید که هیچ برخوردی با بقیه نداشته باشد.
- به سادگی می‌بینیم که عملیات داده شده فقط حالاتی را می‌توانند تولید کنند که هیچ برخوردی در آنها صورت نگیرد، اما بعضی اوقات هیچ عملی ممکن نیست. پردازش جستجو باید انتخاب دیگری را آزمایش کند. یک محاسبه سرانگشتی نشان می‌دهد که فقط ۲۰۵۷ دنباله ممکن برای انجام وجود دارد. فرموله سازی درست تفاوت بزرگی را برای اندازه فضای جستجو ایجاد می‌کند. توجهات مشابه، برای یک فرمول وضعیت کامل به کار برده می‌شود برای مثال، می‌توانیم مسئله را مانند زیر بررسی کنیم:

- حالات: ترتیب ۸ وزیر، هر کدام در یک ستون
 - عملگرها: هر وزیری که با برخورد مواجه می‌شود به مربع دیگری در همان ستون انتقال داده شود.
- این فرمول به الگوریتم اجازه می‌دهد که راه‌حلی را بالاخره پیدا کند، اما بهتر است در صورت امکان به مربع دیگری که خطر حمله وجود ندارد، حرکت کند.

Crypt arithmetic

در مسائلی کریپتاریتمیک، حروف به جای ارقام می‌نشینند و هدف یافتن جایگزینی از اعداد برای حروف است که مجموع نتیجه از نظر ریاضی درست باشد. معمولاً هر حرف باید به جای یک رقم مختلف بنشینند. مثال زیر یک نمونه شناخته شده از این مورد است:

FORTY	Solution :	29786	F = 2, O = 9, R = 7, etc.
+ TEN		850	
+ TEN		850	
-----		-----	
SIXTY		31486	

فرمول زیر شاید ساده‌ترین فرم باشد:

- حالات: یک معادلی cryptarithmic با چند حروف جایگزین شده توسط ارقام.
 - عملگرها: وقوع یک حرف را با یک رقم جایگزین کنید که قبلاً در معنا ظاهر نشده باشد.
 - آزمون هدف: معنا فقط شامل ارقام است و یک مجموع صحیح را بر می‌گرداند.
 - هزینه مسیر: صفر - تمام راه‌حلهای صحیح است.
- یک لحظه تامل نشان می‌دهد که جایگزینی E توسط ۶ و سپس F مشابه حالتی است که F را توسط ۷ و سپس E توسط ۶ جایگزینی‌های مشابه اجتناب کنیم. یک راه، قبول یک ترتیب ثابت است مانند ترتیب الفبایی، انتخاب بهتر دیگر آنکه هر کدام که بیشترین محدودیت جایگزینی را دارد، انتخاب کنیم، یعنی حرفی است که کمترین امکان مجاز را دارند، محدودیتهای معنا را می‌دهد.

دنیای مکش

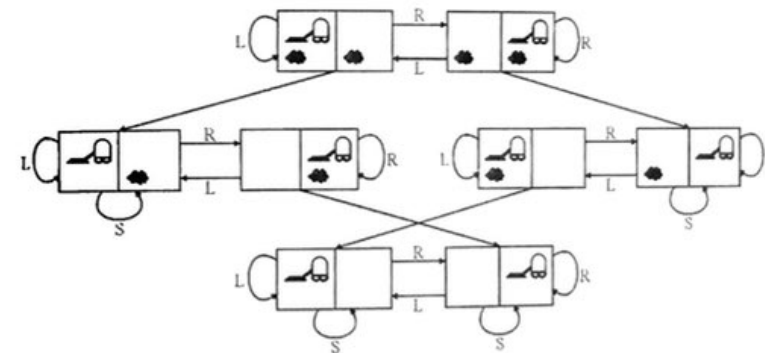
ابتدا اجازه دهید که مورد تک وضعیت را با اطلاعات کامل بازنگری کنیم. فرض می‌گیریم که عامل از جای خودش اطلاع دارد و تمام مکانهای آلوده را می‌شناسد و دستگاه مکنده ما درست کار می‌کند.

- حالات: یکی از ۸ حالت نشان داده شده در شکل ۳
- عملگرها: حرکت به چپ، حرکت به راست، عمل مکش.
- آزمون هدف: هیچ خاکی در چهار گوشه نباشد.
- هزینه مسیر: هر عمل ارزش ۱ دارد.

شکل زیر فضای کاملی از حالات که شامل تمام مسیرهای ممکن است را نشان می‌دهد. حل مسئله از هر حالت شروع، موجب می‌شود تا ما به سادگی فلتها را دنبال کنیم تا به حالت هدف برسیم.

این روشی برای تمامی مسائل است، اما در بیشتر آنها، فضای حالت بسیار گسترده‌تر و پیچیده‌تر است. حال اجازه دهید تا موردی را بررسی کنیم که در آن عامل دارای حس گر نباشد اما هنوز مجبور است که تمام خاکها را جاروب کند. به علت اینکه این یک مسئله چند وضعیت است، ما موارد زیر را خواهیم داشت:

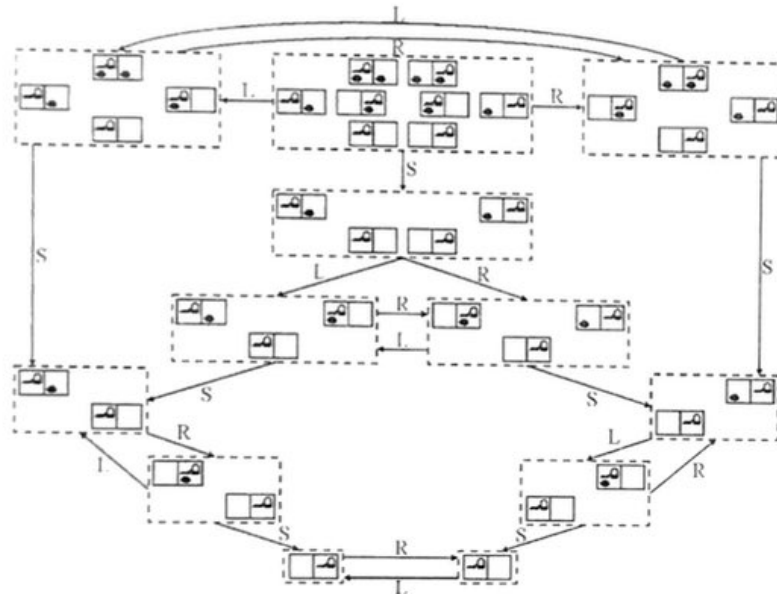
- مجموعه وضعیت‌ها: زیر مجموعه‌ای از حالات که در شکل ۳ نشان داده شده است.



شکل ۳- گراف فضای حالت برای دنیای مکش با حسگر

- عملگرها: حرکت به چپ، حرکت به راست، عمل مکش.
- آزمون هدف: تمام حالات در مجموعه حالتها فاقد خاک باشند.
- هزینه مسیر: هر عمل هزینه ۱ را دارد.

مجموعه حالت شروع، مجموعه‌ای از تمام حالات است زیرا عمل ما فاقد حسگر است. یک راه‌حل، انتخاب دنباله‌ای از مجموعه حالت شروع است که به مجموعه‌ای از حالات بدون خاک منتهی می‌شود. (شکل ۴) فضاهای مجموعه حالت مشابه، می‌توانند برای مورد عدم قطعیت درباره عملیات و عدم قطعیت درباره حالات و عملیات، ساخته شوند.



شکل ۴- گراف فضای حالت بدون حسگر دنیای مکش

مسئله کشیشها و آدمخوارها

مسئله کشیشها و آدمخوارها معمولاً به قرار زیر است. سه کشیش و سه آدمخوار در یک طرف رودخانه قرار دارند و همچنین قایقی که قادر است یک یا دو نفر را حمل کند. راهی را بیابید که هر نفر به سمت دیگر رودخانه برود، بدون آنکه تعداد کشیشها در یکجا کمتر از آدمخوارها باشند.

این مسئله در AI به علت این معروف است که موضوع اولین مقاله‌ای بود که فرموله‌سازی مسئله را از نظر آنالیتیکی بررسی کرد. در چنین مسائلی واقعی قبل از اینکه ما بتوانیم از یک استراتژی حل مسئله استفاده کنیم، باید به اندازه کافی خلاصه شود. صحنه را در زندگی واقعی تصور کنید: سه عضو قبیله Arawaskan، آلپس، باب و چارلز در یک طرف رودخانه آمازون با دوستان جدیدشان، Yolanda، Xavier، Zelda ایستادند. اطراف آنها پرندوها فریاد می‌زدند، باران می‌بارد و غیره. کشیشها که همان Yolanda، Xavier، Zelda، و آلپس، باب و چارلز دلوپس هستند که گرفتار موعظه کشیشها شوند.

برای فرموله سازی مسئله اولین قدم فراموش کردن صدای پرندگان و بارش باران و خلاصه تمام جزئیاتی است که تأثیری در حل مسئله ندارند. قدم بعدی تصمیم‌گیری در مورد انتخاب مجموعه عملگرهای صحیح است. ما می‌دانیم که فقط یک یا حداکثر دو نفر توسط قایق می‌توانند جابه‌جا شوند که همان

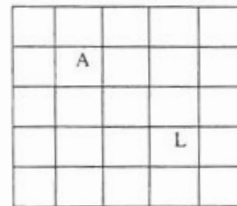
عملگره محسوب می‌شود، اما باید تصمیم بگیریم که ممکن است به حالتی نیاز داشته باشیم که وضعیتی را برای ما بازنمایی کند که افراد در قایق هستند یا به طرف دیگر رودخانه منتقل شده‌اند. زیرا قایق فقط ظرفیت دو نفر را دارد و هیچ افزایشی در تعداد افراد نباید پدید بیاید؛ از این رو، فقط نقاط پایان عبور برای ما مهم هستند. سپس ما به انتزاع نیاز داریم. مطمئناً هر کدام از این شش نفر یک انسان واحد هستند، اما در حل ما، زمانی که یک آدمخوار وارد قایق می‌شود مهم نیست که او آلیس یا باب یا چارلز باشد. این موارد منجر به حل مسئله به صورت زیر می‌شود:

- حالات: یک حالت شامل یک دنباله مرتب شده از عدد است که تعداد کشیش‌ها، تعداد آدمخوارها و محل قایق در ساحلی از رودخانه که از آنجا مسئله شروع شده را نمایش می‌دهد. از این رو حالت شروع به صورت (۱ و ۲ و ۳) است.
- عملگرها: از حالت، عملگرهای ممکن یک کشیش، یک آدمخوار، دو کشیش، دو آدمخوار یا یکی از هر کدام را در قایق جا می‌دهند. از این رو، حداکثر ۵ عملگر وجود دارد، اگر چه حالات کمتر هستند زیرا ضروری است که از حالات نادرست اجتناب کنیم. توجه کنید که اگر بخواهیم بین افراد مشخص فرق قائل شویم، ۲۷ عملگر باید به جای ۵ عملگر وجود داشته باشند.
- آزمون هدف: رسیدن به حالت (۵ و ۵ و ۵).
- هزینه مسیر: تعداد دفعات عبور از رودخانه.

این فضای حالت به قدر کافی کوچک است که یک مسئله بدیهی را برای حل کامپیوتری آماده کند. مردم در زمان سختی زندگی می‌کنند، به‌رحال تعدادی از حرکات ضروری به نظر می‌رسد که به قهقرا رفتن است. به احتمال زیاد، مردم عبارت «پیشرفت» را برای هدایت جستجوهایشان استفاده می‌کنند.

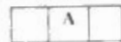
تست‌های فصل اول

- ۱- یک جنگل که بصورت یک شبکه 5×5 در شکل مقابل نشان داده شده است. را در نظر بگیرید عامل A می‌تواند 90° درجه به یکی از جهات چهارگانه بچرخد و یا یک خانه به جلو حرکت کند. شیر L نیز می‌تواند آزادانه در جنگل حرکت کند. عامل A می‌تواند مکان و جهت خود را بداند. اگر بخواهیم از منطق گزاردها (Propositional logic) برای نمایش دانش استفاده نماییم، برای بیان اینکه «اگر در مقابل عامل، شیر قرار ندارد، می‌تواند جلو برود» به چند گزاره احتیاج داریم؟ (سراسری - ۸۳)



۱ (۱) ۲۵ (۲) ۴ (۳) ۸۰ (۴)

- ۲- فرض کنید A یک جاروبرقی اتوماتیک است. محیط این جاروبرقی، مطابق شکل زیر، از سه خانه کنار هم تشکیل شده است. این جاروبرقی می‌تواند از هر یک از این خانه‌ها با انجام یک حرکت به خانه مجاور نقل مکان نماید و زباله‌ی موجود در آن خانه را (در صورت وجود) جمع‌آوری کند. با توجه به این که این جاروبرقی برای جمع‌آوری هر زباله باید در همان خانه‌ای که زباله در آن وجود دارد، قرار بگیرد، فضای حالت این مسئله دارای چند وضعیت منحصر بفرد است؟ (سراسری - ۸۳)



۲۴ (۱) ۹ (۲) ۶۴ (۳) ۸۱ (۴)

پاسخ تستهای فصل اول

۱- گزینه «۱» صحیح است.

برای حل این مسئله در منطق گزارها، بابت عدم وجود سور تعداد زیادی قانون باید تعریف شود. اگر عامل A در خانه (a) قرار داشته باشد و بفرضیم تصمیم‌گیری کنیم که به جلو حرکت کند یا نه، برای تک خانه‌های مجاور باید آزمون صورت گیرد. توجه کنید حرکت روبه جلو بستگی به جهت آخرین چرخش عامل وابسته است. برای مثال به قانون زیر توجه کنید.

$$A_{i+1} \wedge East A_i \wedge L_{i+1} \Rightarrow \sim Forward$$

این قانون بیان می‌کند که اگر عامل در خانه a و اصرار دارد و رو به سمت شرق دارد و شیر در خانه $a+1$ قرار دارد، به سمت جلو حرکت نکند.

اما تعداد خانه‌های مجاور هر خانه صفحه 5×5 چه تعدادی است؟ بدیهی است 4×4 خانه داخلی چهار مجاور، ۴ خانه گوشه دو مجاور و ۱۲ خانه دیگر موزی، هر یک سه مجاور دارند. پس در مجموع نیاز به تعریف ۸۰ قانون متغیر داریم.

۲- گزینه «۱» صحیح است.

این پوشش نوع توسعه یافته مثال ذکر شده در متن درس است. پس بسادگی می‌توان ثابت کرد که به 4×4 حالت در فضای حالت وجود دارد.

$$2^4 \times 4 \times 2^4$$

فصل دوم

روشهای جستجو

تاکنون به دنبال توصیف ریاضی از یک مسئله واقعی بودیم و دیدیم چگونه می‌توان آن را کراف فضایی حالت بتوان یک مدل ریاضی برای توصیف مسائل واقعی استفاده کرد. حال زمان آن فرا رسیده تا سعی کنیم از این عمل برای حل مسئله استفاده کنیم. به عبارت دیگر چگونه می‌توان عمل جستجو بر روی این کراف را انجام داد تا راه‌حل که شامل پاتین مسیری از گره آغازین تا هدف است، حاصل شود. برای این منظور همانطور که در بخش قبلی اشاره شد، استراتژی‌های مختلفی وجود دارند که می‌توانند آگاهانه و یا غیر آگاهانه و یا حتی تصادفی باشند.

بطور کلی هر الگوریتم جستجو را می‌توان یکم چهار زیر مورد آزمون قرار داد:

۱- کامل بودن، در صورت وجود راه‌حل، آیا قطعاً پیدا خواهد شد؟

۲- پیچیدگی، در صورت رسیدن به پاسخ، آیا کارآمدترین مسیر انتخاب می‌شود؟

۳- پیچیدگی زمانی، زمان اجرای الگوریتم چیست؟

۴- پیچیدگی فضای مصرفی، الگوریتم به چقدر فضا در حافظه اصلی نیاز دارد؟

۱-۲- روش‌های جستجوی غیر آگاهانه

جستجوی سطحی

یکی از استراتژی‌های ساده جستجوی سطحی Breadth-first search است. در این استراتژی ابتدا گره ریشه، گسترش می‌یابد، سپس تمام گره‌های که توسط ریشه تولید شده‌اند، خودشان گسترش می‌یابند و سپس مولدهای آنها، و به همین ترتیب در حالت کلی در درخت جستجو تمام گره‌های عمق l قبل از گره‌های عمق $l+1$ گسترش داده می‌شوند. جستجوی سطحی توسط یک صف پیاده‌سازی می‌شود.

جستجوی سطحی یک استراتژی بسیار سیستماتیک است زیرا ابتدا تمام مسیرهای با طول 1 را در نظر می‌گیرد و سپس مسیرهای با طول 2 والی آخر. شکل ۵ پیشرفت جستجو را روی یک درخت دو بومی ساده نشان می‌دهد. اگر راه‌حل وجود داشته باشد، جستجوی سطحی ابتدا کم عمق‌ترین وضعیت هدف را پیدا می‌کند. مطابق با چهار معیار گفته شده در قبل، جستجوی سطحی کامل و کم چتین، بهینه است چرا که مزیت مسیر، یک تابع کاهش نیابنده از عمق گره است. (این شرط معمولاً فقط زمانی برقرار می‌شود که تمام عملگرها هزینه مشابهی داشته باشند. برای حالت عمومی، به بخش بعد مراجعه کنید)