وب سایت گروه کامپیوتر دانشگاه آزاد اسلامی واحد تهران شرق

(قیامدشت)

www.Compg.ir

دانشگاه آزاد اسلامی
واحد تهران شرق

هوش مصنوعی

مهندس حمیدرضا طارمیان

سید رسول موسوی فیه
۸۹۱۰۹۱۲۴۱

کارشناسی ناپیوسته

مهندسی کامپیوتر-نرم افزار

زمستان، بهار ۹۱-۱۳۹۰ .روزهای چهارشنبه

۱) مقدمه

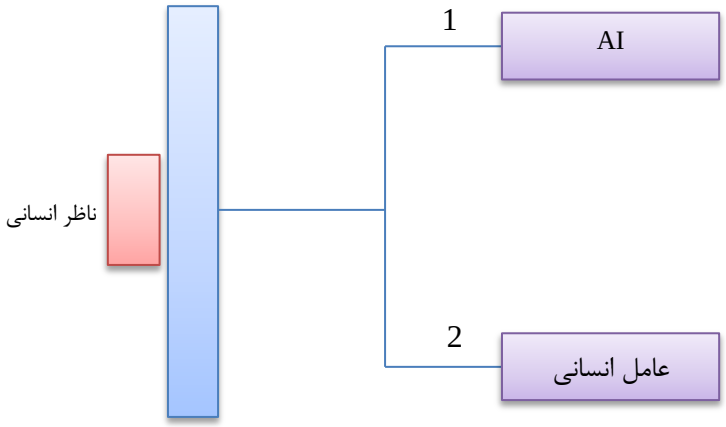
تعاریف هوش مصنوعی (Artificial Intelligence (AI)

هوش مصنوعی از چهار دیدگاه تعریف می‌شود:

۱ـ عملکرد انسانگونه

ساخت شئ که عملکرد آن مانند انسان باشد.

برای بررسی سیستم‌های هوشمند (هوش مصنوعی) شخصی بنام تورینگ آزمونی را طراحی کرده که به شکل زیر است؛



در این آزمون در پاسخ به ناظر انسانی باید شئ هوشمند توانایی فریب ناظر انسانی را داشته باشد و وی نتواند بفهمد که جواب صادر شده از طرف AI بوده یا عامل انسانی!

← قابلیت‌هایی که دستگاه (هوش مصنوعی) باید داشته باشد تا بتواند در آزمون تورینگ شرکت کند:

- ✓ قابلیت نمایش (بازنمایی) دانش
- ✓ استدلال خودکار
- ✓ پردازش زبانهای طبیعی (NLP)
- ✓ یادگیری ماشین

به دلیل عدم وجود حرکت در آزمون اولیه تورینگ که عملکرد انسانگونه دستگاه (هوش مصنوعی) را باید در نظر بگیرد، آزمون کلی تورینگ اضافه گردید که علاوه بر موارد فوق موارد دیگری که به شرح ذیل هستند اضافه گردید؛

- روباتیک
- بینایی ماشین
- پردازش صوت

۲ـ تفکر انسانگونه

در ادامه پس از عملکرد انسانگونه تعریف دیگری بوجود آمد که دستگاهی را هوشمند می‌نامیدند که دارای تفکر انسانگونه باشد.

* در تفکر انسانگونه ابتدا باید تفکر انسان را شناخت که طی موارد زیر انجام می‌گیرد:

الف) مطالعه و آزمایش رفتارهای انسان

ب) مستقیم (مطالعه فرآیند تفکر در انسان) ـ اتفاقاتی که در هنگام تفکر در بدن انسان رخ می‌دهند.

← به دلیل اینکه این تعریف (تفکر انسانگونه) پایه‌ی علمی محکمی جهت مطالعه ندارد دسته‌ی دیگری از تعریف‌ها بوجود آمد.

۳ـ تفکر منطقی (rational)

بر پایه‌ی علوم منطقی ریاضی ارسطو، افلاطون و ... صورت گرفته است.

مشکل این تفکر در این است که انسان همیشه تفکر منطقی ندارد و برخی مواقع به صورت عکس‌العملی رفتار می‌کند؛ مانند وقتی که انسان به یک شئ داغ دست می‌زند ناخودآگاه دست خود را عقب می‌کشد که ربطی به تفکر منطقی ندارد.

مشکل دیگر این تفکر این است که بسیاری از مسائل روزمره توسط منطق قابل پیاده‌سازی نیستند و یا به سختی قابل پیاده‌سازی هستند.

به دلیل مشکل مطرح شده فوق تفکر منطقی نیز تعریف مناسبی نیست و تعریف دیگری بوجود آمد که کاملتر است و مطلوب بحث ما می‌باشد.

۴ـ عملکرد منطقی یا رفتار منطقی

انجام کار درست را رفتار منطقی گویند. کار درست کاری است که قاعدتاً در راستای رسیدن به هدف بیشترین موفقیت را دارد. البته با توجه به اطلاعات موجود؛

از مزایای این تعریف گنجانده شدن مسائلی که رفتار منطقی ندارند، می‌باشد. مانند شئ داغ.

عامل (Agent)

چیزی است که بتواند قابلیت درک مطلب و اجرای کاری را داشته باشد.

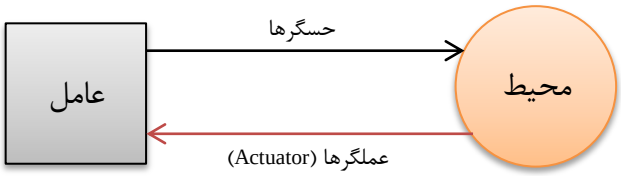
عامل‌های هوشمند

۱ـ عامل‌ها و محیط ۲ـ عامل‌های منطقی ۳ـ PEAS ۴ـ انواع محیط ۵ـ انواع عامل

۱ـ عامل‌ها و محیط

هر عاملی با محیط اطراف خود در ارتباط است و این عامل از طریق حسگرها اطلاعات محیط را درک می‌کند، سپس از طریق عملگرها روی محیط تأثیر می‌گذارد.

درک (Perceive) ـ ادراک (Percept)



* اولین مسئله‌ای که در طراحی عامل (Agent) باید در نظر گرفته شود سادگی است.
* عامل مانند یک تابع است که تعدادی ادراک گرفته و به یک خروجی تبدیل می‌کند.

f: P* → A f = تابع P = Percept A= Action

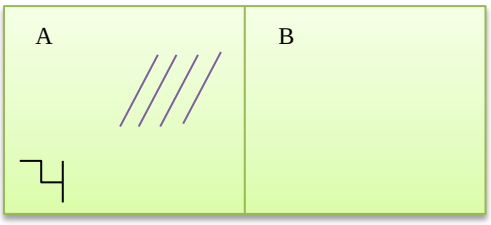
* تابع یک مفهوم است اما اگر آنرا به برنامه تبدیل کرده یا روی IC نوشته شود برنامه عامل (Agent Program) نامیده می‌شود.
* معماری: ساختار فیزیکی نیز از بخش‌های عامل است.

پس یک Agent از برنامه عامل و ساختار فیزیکی تشکیل شده است.

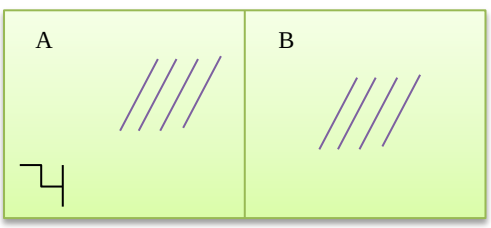
* دقت شود عامل همیشه یک ماشین نیست، مثلاً آنتی ویروس‌ها یک عامل هستند.

مثال) دنیای جاروبرقی: دو اتاق وجود دارد و یک عامل جاروبرقی، یکی از اتاق‌ها تمیز و دیگری کثیف، بر روی جاروبرقی دو سنسور وجود دارد؛ اولی مکان آن و دیگری تمیز بودن یا کثیف بودن اتاق فعلی را سنس می‌کند.

اکشن‌ها: تمیز کردن، حرکت کردن (Right, Left) ← اتاق‌ها
No OP (هیچ عملی انجام ندهد)



[A, Dirty]



[A, Dirty]

۲ـ عامل‌های منطقی

عاملی که بر اساس اطلاعاتی که از سنسور دریافت می‌کند و اعمالی که می‌تواند انجام دهد و دانش داخلی‌اش همـواره بتوانـد کاری را انجام دهد که قاعدتاً در راه رسیدن به هدف بیشترین موفقیت را کسب کند.

معیار کارایی

معیاری است که رفتار عامل را در راه رسیدن به هدف می‌سنجد؛ عامل باید اکشنی را انتخاب کند که معیار کـارایی آنـرا بـالا ببرد.

عامل‌های عقل کل (Omni - Science یا Know It All)

عاملی عقل کل است که نتیجه‌ی واقعی اعمالش را بداند. عامل عقل کل وجود خارجی ندارد و تنها عاملی است که همه چیـز را می‌داند.

* عامل منطقی معیار کارایی مورد انتظار را حداکثر می‌کند، اما عامل عقل کل معیار کارایی واقعی را حداکثر می‌کند.

عامل‌های اکتشافی

این عامل‌ها اکشن‌هایی که انجام می‌دهند ممکن است در راستای هدف نباشد و برای جمع‌آوری اطلاعات و ... باشد.

عامل‌های خود مختار

عاملی خود مختار است که اکشن‌های آن بر اساس ادراکاتش اجرا شود.

خود مختاری وقتی که اکشن‌ها توسط ما به حسگرها داده شوند از درجه‌ی پایین‌تری برخوردار است. اما شی‌ای که بر اسـاس یادگیری خود (اکتشاف) اکشن می‌کند درجه‌ی خود مختاری بالاتری دارد.

۳ـ PEAS

برای طراحی یک عامل به چهار چیز احتیاج است که به آنها PEAS گویند.

۱ـ معیار کارایی: Performance measure

۲ـ محیط: Environment

۳ـ عملگرها Actuator

۴ـ حسگرها Sensors

* دقت شود چهار مورد فوق حتماً باید به ترتیب بیان شوند.

الف) معیار کارایی اول است چرا که بر اساس هر معیاری یک سنسور خاص قرار می‌گیرد.

ب) شناسایی محیط، به ازای محیط‌های مختلف سنسورهایی با حساسیت مختلف نیاز است.

محیط (E)

برای معیارکارایی پس از شناسایی، سنسورها و عملگرها را تعریف کرده و سپس محیط را در نظر می‌گیریم. چرا که سنسورها بر اساس محیز عمل می‌کنند. لازم نیست که یک عامل به صورت فیزیکی در محیط باشد؛ ممکن است محیط به صورت نرم‌افزاری باشد. محیط عملیاتی تشکیل شده از: معیارکارایی، محیط و محرک‌هایی که برای عامل قابل دسترس باشند.

بر اساس اینکه محیط چه خاصیت‌هایی دارد، انواع محیط را تعریف می کنیم؛

دسته‌بندی اول

محیط‌های کاملاً مشاهده پذیر (Full Observable)

محیط‌های نسبتاً مشاهده پذیر (Partially Observable)

اگر کلیه‌ی اطلاعات از طریق حسگرها در دسترس باشد به این محیط، محیط کاملاً مشاهده‌پذیر گویند. و اگر برخی اطلاعات در دسترس نباشند محیط نسبتاً مشاهده پذیر است. برای مثال محیط بازی شطرنج کاملاً مشاهده‌پذیر است و محیط جاروبرقی نسبتاً مشاهده‌پذیر است چرا که در صورتی که در فضای مجاور کثیفی وجود داشته باشد نمی تواند آنرا ببیند.

* محیط‌های واقعی و طبیعی همیشه محیط نسبتاً مشاهده‌پذیر هستند.

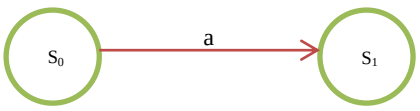
دسته‌بندی دوم

محیط‌های قطعی(معین)

محیط‌های غیر قطعی (نامعین ـ احتمالی) ـ Stochastic

اگر در یک محیط حالت فعلی مشخص باشد، با مشخص شدن عملی که می‌خواهیم انجام دهیم، حالت بعدی محیط به صورت قطع معین خواهد شد، به این محیط، محیط معین گویند. به عنوان مثال بازی شطرنج بازی‌ای قطعی است. دنیای جاروبرقی نیز محیط قطعی است، اما رانندگی یک اتومبیل غیرقطعی است چرا که مثلاً هنگام ترمز دقیقاً معلوم نیست که چند متر جلوتر ترمز (توقف کامل) صورت می‌گیرد.

شکل روبرو نشان دهنده‌ی یک محیط معین و قطعی است:



* محیط‌های طبیعی معمولاً غیرقطعی هستند.

← به برخی محیط‌ها نه قطعی و نه غیر قعی گفته می‌شود؛ که آنها را محیط «استراتژیک» گویند.

اگر محیط ذاتاً قطعی باشد، اما اعمال سایر عامل‌ها باعث شود که خروجی مورد نظر تولید نشود، آنگاه محیط استراتژیک نامیده می‌شود.

* در دنیای جاروبرقی اگر به عامل گفته شود تمیز کن، شروع به تمیز کردن می‌کند و در این لحظه یک عامل پس از تمیز کردن توسط جاروبرقی، آشغالی را بریزد، با اینکه زمان تمیز کردن جاروبرقی تمام شده اما به دلیل کار عامل دیگر اتاق کماکان کثیف است، در این حالت به این محیط استراتژیک گویند.

دسته‌بندی سوم

محیط اپیزودیک یا مرحله‌ای (Episodic)

محیط غیر اپیزودیک (ترتیبی)

اگر تجربه‌ی محیط را به صورت اپیزود تقسیم کنیم (اپیزود: در هر مرحله هر سنس، یک اکشن انجام دهد) اگر اکشن‌های انتخابی فقط به اپیزود جاری مربوط باشند و به اپیزود قبلی و بعدی ارتباطی نداشته باشند، محیط اپیزودیک است، در غیر اینصورت محیط غیر اپیزودیک است.

* تقریباً بین ۸۰ تا ۹۰ درصد محیط‌های تعریفی در این درس غیر اپیزودیک هستند.

دسته‌بندی چهارم

محیط‌های ایستا (Static)

محیط‌های پویا (Dynamic)

اگر در مدت سنجش محیط توسط عامل و اتخاذ تصمیم لازم، محیط هیچ تغییری نکند، آنگاه محیط ایستا است. در غیر اینصورت؛ یعنی اگر در مدت مذکور محیط تغییر کند، محیط پویا است.

به طور مثال در دنیای جاروبرقی محیط ایستا است؛ چرا که هنگام سنس کردن کثیفی، محیط تغییری نکرده است.

اما در راننده تاکسی، محیط پویا است؛ چرا که برای مثال سنس محیط به آن گفته که محیط خلوت است و اجازه‌ی شتاب گرفتن دارد اما به ناگاه یک انسان جلوی آن وارد مسیر حرکت می‌شود!

محیط‌های نیمه پویا

در این محیط شرط اول این است‌که محیط ذاتاً ایستا باشد، اما با گذشت زمان معیار کارایی عامل تغییر پیدا کند. در بازی شطرنج با ساعت با اینکه محیط ثابت است اما هر چه بازیکن بیشتر فکر کند و زمان می‌گذرد، معیارکارایی بازیکن کاهش می‌یابد.

در مثال دیگر می‌توان سیستم پردازش تصویر را مدنظر قرار داد، تصویر در سیستم ایستا است (محیط)؛ اما مدت زمان پردازش آن هر چه بیشتر شود معیار کارایی آن پایین‌تر می‌آید، که محیط نیمه پویا می‌شود.

* در محیط‌هایی که معیارکارایی براساس زمان، بررسی می‌شود، محیط نیمه‌پویا است.

دسته‌بندی پنجم

محیط‌های گسسته (Discrete)

محیط‌های پیوسته (Continuous)

اگر در یک محیط تعداد اعمال محدود باشد و بتوان آنها را به‌طـور مشـخص تعریـف کـرد، محـیط گسسـته خواهـد بـود؛ در غیراینصورت پیوسته است؛ به عبارت دیگر تعریف مشخصی برای اعمال نداریم.

برای مثال در دنیای جاروبرقی اعمال ذاتاً گسسته هستند، چرا که یک دستور و یک کار مشخص انجام می‌گیرد. امـا در مثـال راننده تاکسی در دستور حرکت به جلو معلوم نمی‌شود چند متر حرکت انجام شود.

* اگر در یک عاملی حتی یک محیط پیوسته داشته باشیم، همه‌ی آن پیوسته است.

دسته‌بندی ششم

محیط‌های تک عامله (Single)

محیط‌های چند عامله (Multi)

اگر در محیط فقط یک عامل باشد، تک عامله و در غیر اینصورت چند عامله است.

* درک عامل بودن؛ باید دید که آیا آن شی محیط را درک می‌کند و می‌تواندروی محیط تأثیر گذارد یا خیر.

مثلاً در بازی‌ها هنگام پیچیدن ماشین به پیاده‌رو، افراد فرار می‌کنند؛ که در حقیقت آنها محیط را سنس کرده و یـک اکشـن انجام می‌دهند، که بر همین اساس نتیجه گیری می‌شود که آنها عامل هستند.

محیط‌های چند عامله خود به چند دسته تقسیم می‌شوند. این تقسیم بندی بر اساس کار و ارتباطات صورت می‌گیرد؛

عامل‌های مستقل

کار و اهداف عامل‌ها به همدیگر ارتباطی ندارد.

عامل‌های همکار

معمولاً هدف مشترک دارند و برای رسیدن به این هدف مشترک، به یکدیگر کمک می‌کنند.

عامل‌های رقابتی

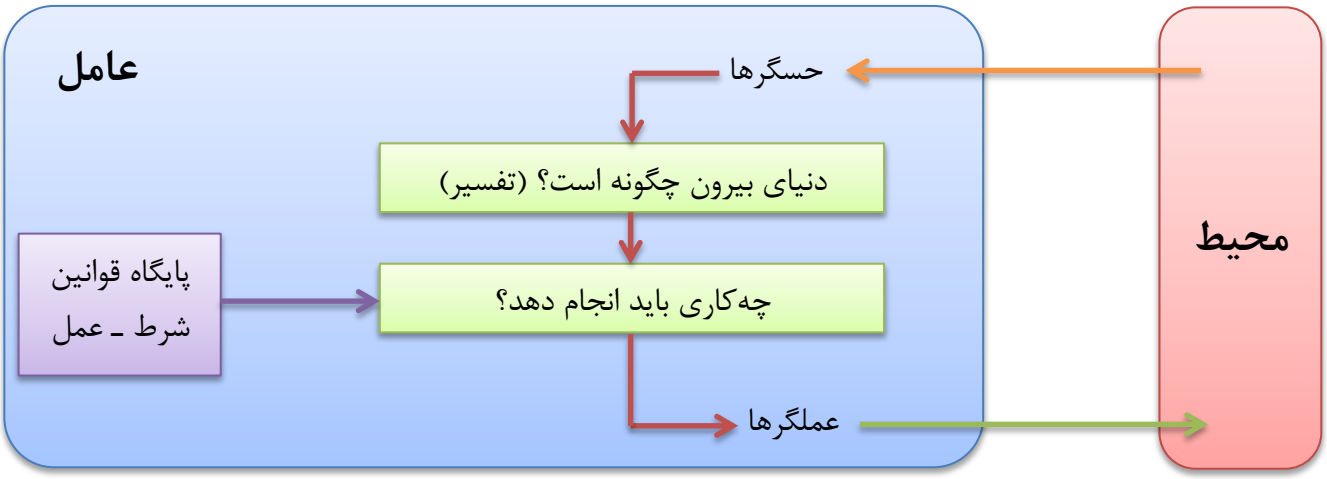
معمولاً هدف آنها باهم در تضاد است.

انواع عامل‌ها بر اساس برنامه‌ی عامل

۱ـ عامل‌های واکنشی ساده
۲ـ عامل‌های واکنشی مبتنی بر مدل
۳ـ عامل‌های هدفمند (هدف‌گرا)
۴ـ عامل مبتی بر سودمندی (مطلوبیت)

عامل‌های واکنشی ساده (Simple Reflex)

دارای جدول جستجوی ساده هستند. در آنها تعدادی از وضعیت‌ها می‌توانند توسط قانون‌های شرط ـ عملکرد خلاصـه شـوند. پیاده‌سازی ای نوع عامل‌ها آسان می‌باشد ولی دارای کاربرد کمی می‌باشند.

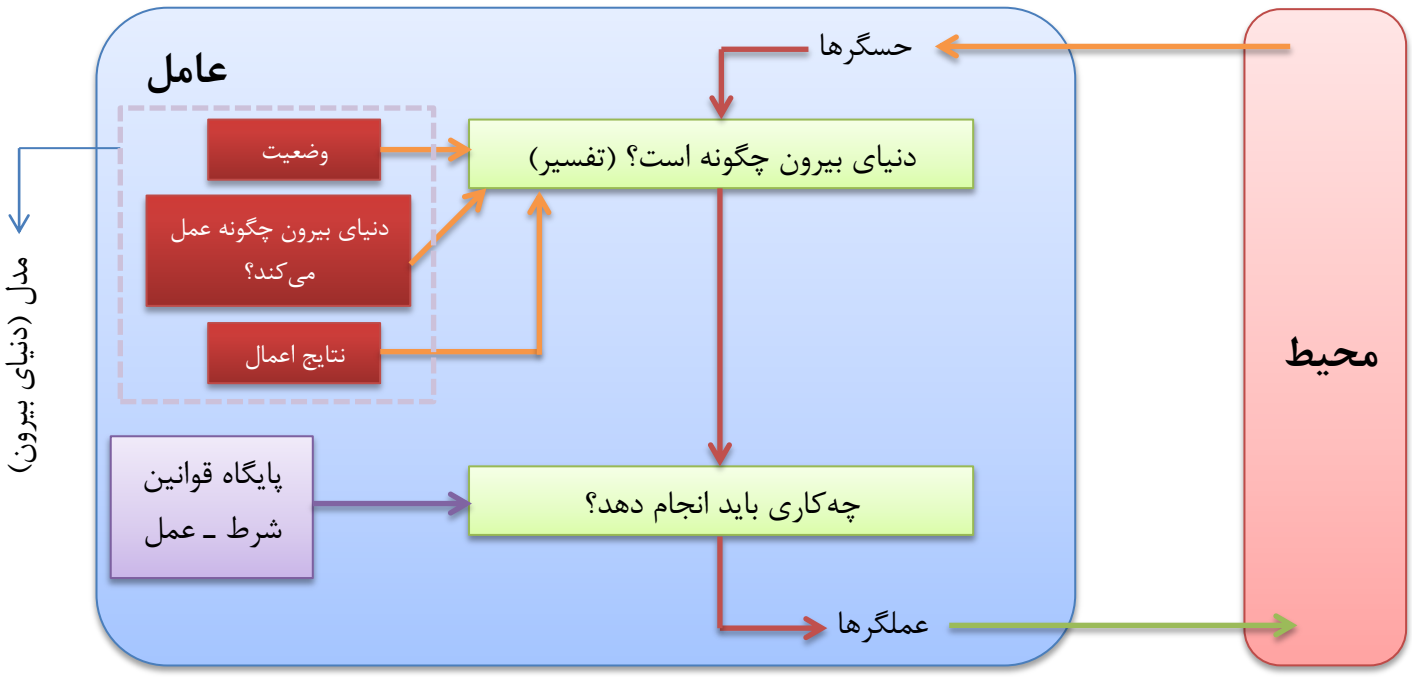


مثال) تابع Ref_vac_agent([location,status]) یک عملکرد از دنیای جاروبرقی را بر می‌گرداند، در صورتی که وضعیت اتـاق A برابر با کثیف (Dirty,A) بود، مکش (Suck,A) را برمی‌گرداند.

* اگر محیط کاملاً مشاهده‌پذیر نباشد، عامل‌های واکنشی ساده دچار مشکل می‌شوند.

عامل‌های واکنشی مبتنی بر مدل (Reflex Model Based)

اطلاعات عامل به تنهایی در مورد عامل‌های نسبتاً مشاهده‌پذیر کافی نیستند، لازم اسـت کـه جریـان تغییـرات جهـان را نیـز نگهداری نماییم (حافظه یا مدل) و تکامل‌ها، به طور مستقل از عامل، یا سبب شده توسط عملکرد عامل می‌باشند.



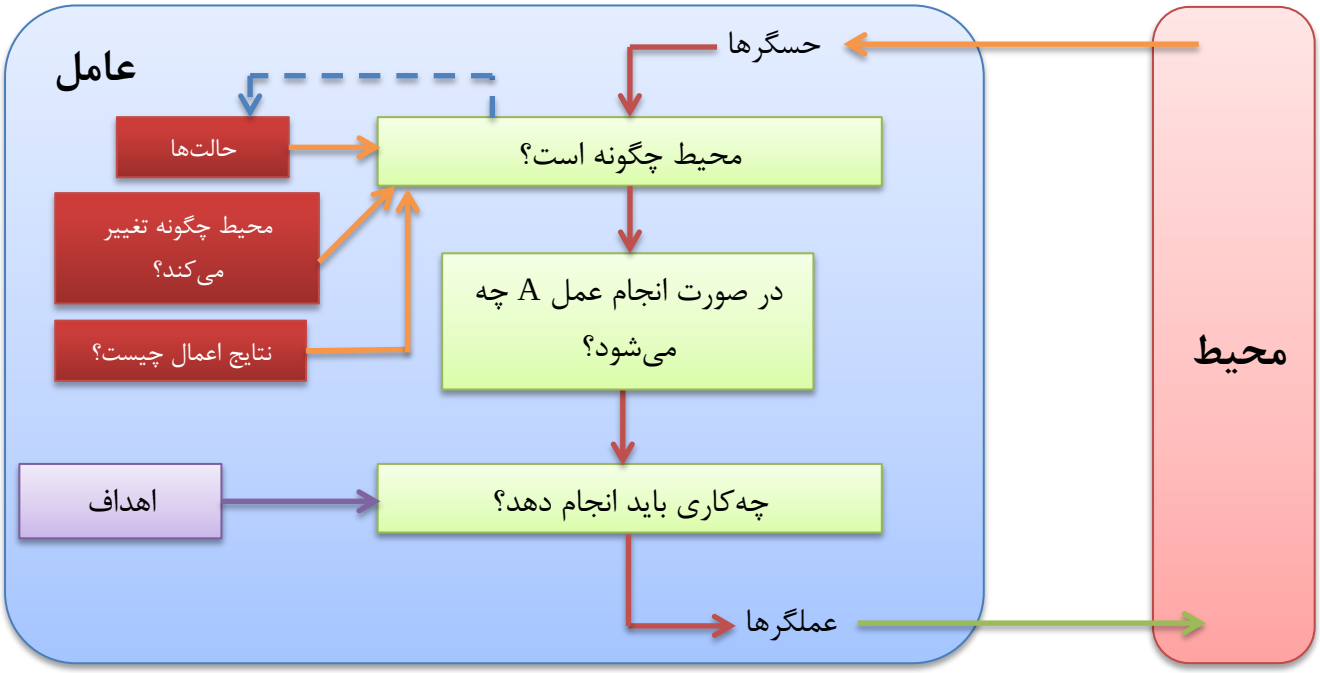
دنیای بیرون را مدل کرده و چگونگی عمل دنیای بیرون و نتایج اعمال را نیز بدست آورده و از طریق این سه مـورد مـی‌تـوان حدس زد دنیای بیرون چگونه است.

مثال) برای مثال اتاق A کثیف است، به قسمت حالت‌ها رفته (حالت‌های تعریـف شـده در مـدل) را بررسـی کـرده و از بـین حالت‌های موجود انتخاب عمل صورت می‌گیرد، به‌طور مثال در اینجا حالت‌های ثبت شده پس از انجام دستور فوق دو حالـت اتفاق می‌افتد: اتاق A تمیز شده، و اتاق B تمیز است. حالت دیگر اتاق A تمیز شـده و اتـاق B کثیـف اسـت. کـه در جـدول وضعیت‌های ثبت شده (مدل شده) موجود است.

* عامل‌های واکنشی در مواقعی که مقصد و هدف داریم کاربرد ندارند؛ (مانند راننده تاکسی).

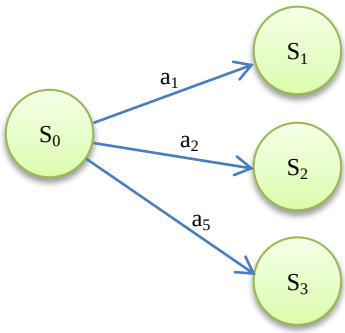
عامل‌های هدفمند (هدف‌گرا ـ مبتنی بر هدف) ـ (Goal Based Agents)

در این گونه عامل‌ها وضعیت و عملکردها نمی گویند که کجا برویم. از قوانین یکسان برای اهداف مختلف اسـتفاده مـی‌نمایـد.
استدلال هدف‌گرا برای محیط‌های ترتیبی خیلی مفید است: مانند بازی شطرنج، راننده تاکسی و خلبانی سفینه‌ی فضایی



* در صورت انجام عمل A چه می‌شود؟

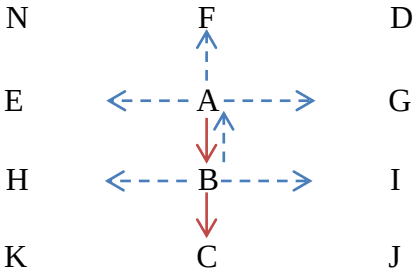
یعنی آنکه در وضعیت فعلی در صورت انجام کاری به چه وضعیتی می‌رسیم؛ (سیستم منطقاً این عملیات را بررسی می‌کند)



* چه کاری باید انجام شود؟

یعنی اینکه پس از بررسی، وضعیت‌های بعدی چک می‌شود که کدام وضعیت بـه هـدف نزدیـک‌تـر اسـت و اکشـن آن انجـام می‌شود و به عملگرها ارسال می‌گردد.

مثال) از نقطه A به نقطه C می‌خواهیم برویم، توسط عوامل مبتنی بر هدف تحلیل را انجام دهید؛



عامل‌های مبتنی بر هدف عامل‌های بسیار خوبی هستند اما همه جا کاربرد ندارند؛ برای مثال در راننده‌ی تاکسی دو هدف سرعت و ایمنی وجود دارد که سیستم دچار مشکل می‌گردد؛ به عبارت دیگر در جاهایی که اهداف متناقض وجود دارد، عوامل مبتنی بر هدف دچار مشکل می‌شوند. مشکل دیگر در اهدافی است که رسیدن به آنها قطعی نیست؛ مانند اینکه به راننده تاکسی گفته شود در مسیر اگر مغازه X باز بود خرید کن، در این موارد نیز عوامل مبتنی بر هدف دچار مشکل می‌شوند.
برای حل این مشکلات از عوامل دیگری که در ادامه بیان می‌شود استفاده خواهد شد.

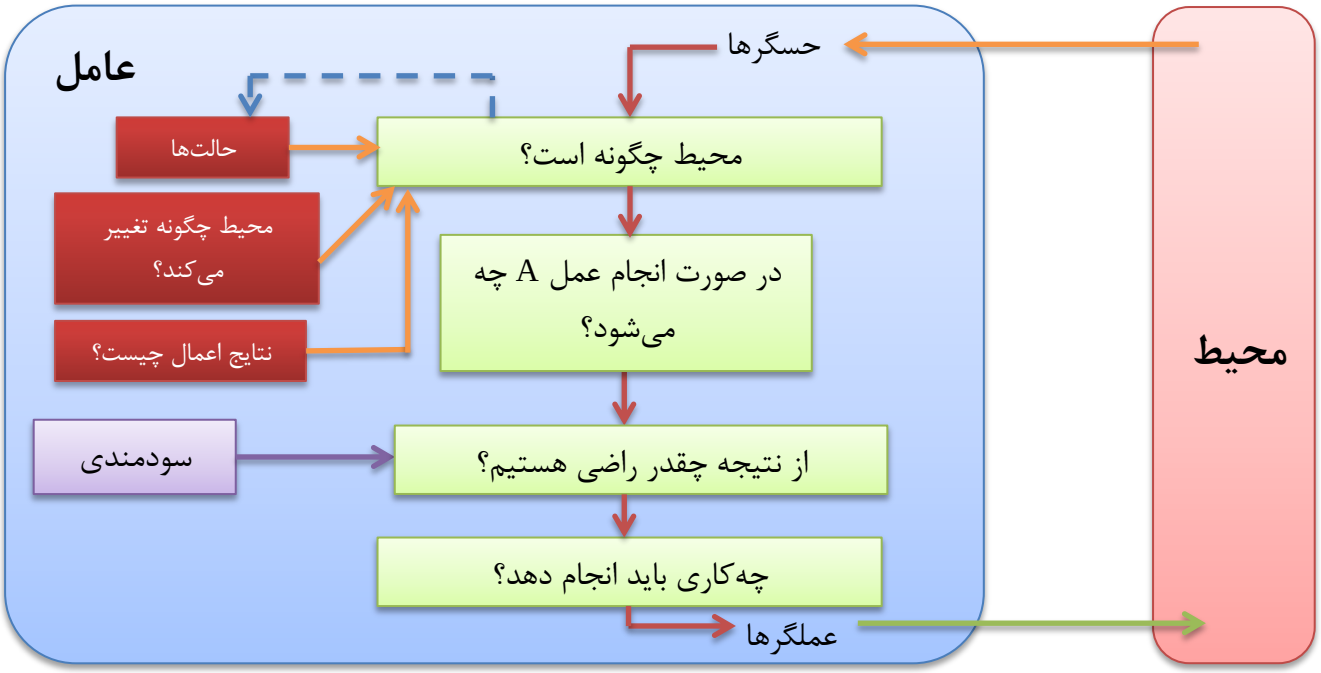
عوامل مبتنی بر سودمندی (Utility Based Agents)

قسمت اول این عوامل مانند عوامل مبتنی بر هدف است و تفاوت آن؛ در بررسی وضعیت به جای هدف از رضایت کار جویا می‌شود و سپس کار را بر اساس رضایت بیشتر (سودمندی بیشتر) انجام می‌دهد. حال ممکن است این سودمندی خود هدف نباشد.

سودمندی یک تابع است که ورودی آن یک حالت (وضعیت) است و خروجی آن یک عدد است که نمایاتگر میزان رضایت است. (U:حالت)

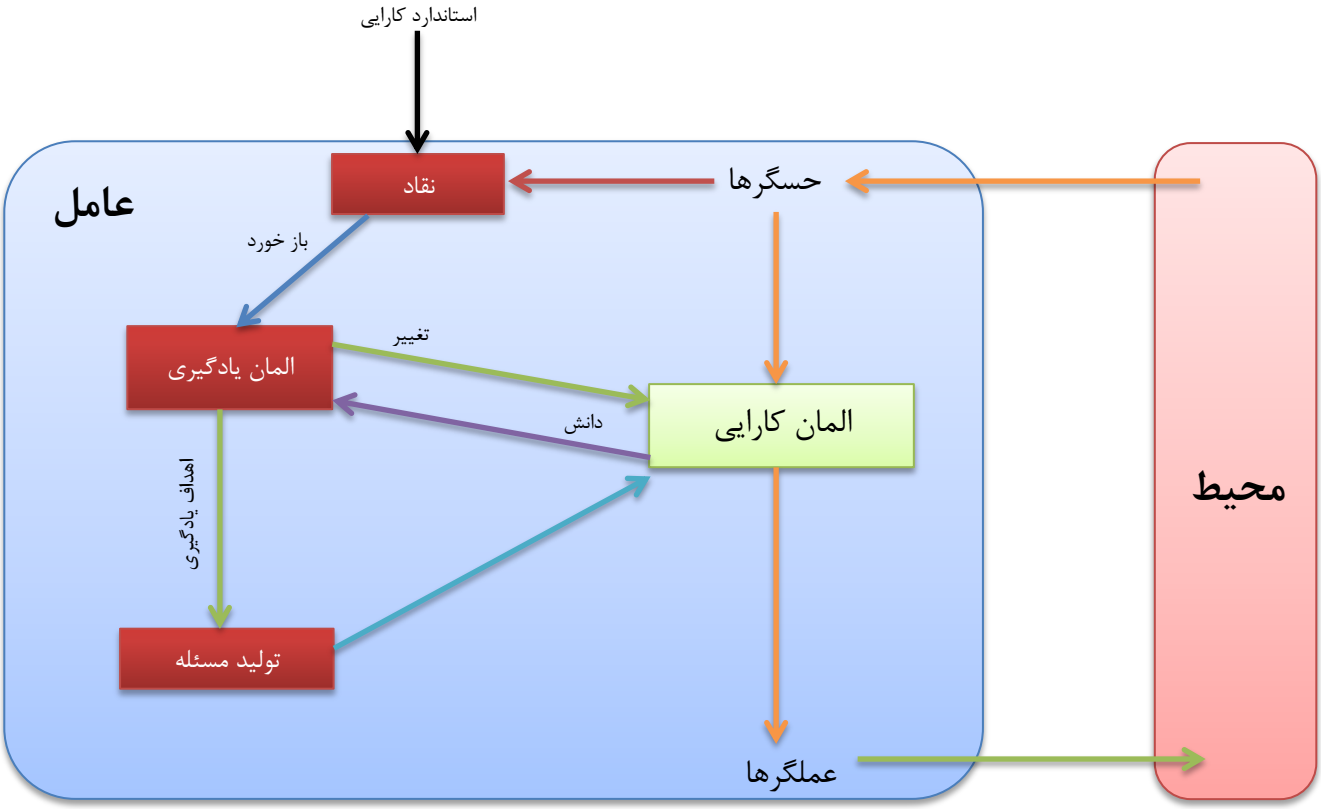
U=عدد

برای مثال در راننده تاکسی بیان می‌شود ایمنی در اولویت قرار دارد. (مهم‌تر است)، سرعت، ایمنی و هزینه بررسی شده و هر کدام امتیاز بیشتری آورد (ایمنی) اجرا می‌گردد.



نوع کلی‌تری وجود دارد که هر چهار مورد بالا نیز می‌توانند در آن جایی بگیرند!

عامل‌های یادگیری (Learning Agents)



* المان کارایی (Performance Element): کار المان کارایی این است که بر اساس ورودی‌های سنس شده عمل مناسب را انجام دهد. (عامل)

* وظیفه ی نقاد بررسی اطلاعات دریافتی و مقایسه‌ی آنها با اطلاعات استاندارد کارایی می‌باشد. نتیجه‌ی این بررسی بازخورد است.

* وظیفه‌ی المان یادگیری دریافت بازخورد است که بر اساس آن در المان کارایی تغییر ایجاد می‌کند که این کار دانش را نتیجه می‌دهد.

برای مثال یک بچه ی ۶ ماهه را در نظر می‌گیریم که به وی سوپ داغ داده می‌شود؛ این کودک تا کنون سوختن را تجربه نکرده است، پس از سنس اطلاعات به نقاد رفته در صورت درد بیش از اندازه بازخورد به المان یادگیری می‌دهدکه این المان باعث می شود که این کودک سوپ داغ را دیگر نخورد. اما در این یادگیری باید پیشرفت حاصل شود چرا که اگر دیگر هیچوقت سوپ داغ نخورد در زندگی دچار مشکل می‌شود! در اینجا تولید مسئله بر اساس آزمون و خطا پیشنهادهای دیگری به المان‌کارایی می‌دهد؛ برای مثال یک نفر نمی‌تواند سنگ یک تنی را جابه جا کند اما تولید مسئله به این نتیجه می‌رسد که با ۴ نفر می‌شود و این جریان را به یادگیری‌های این مسئله اضافه می‌کند.

حل مسئله و جستجو

۱ـ عامل‌های حل مسئله ۲ـ انواع مسائل ۳ـ فرموله سازی مسئله
۴ـ مسئله نمونه ۵ـ الگوریتم‌های جستجو

عوامل مبتنی بر هدف (فصل۲) خود دارای موارد مختلف است که یکی از آنها عوامل حل مسئله است!

عامل‌های حل مسئله

خود این عوامل می‌توانند بر اساس کارهای مختلفی انجام شود، اما هدف ما در اینجا بر اساس جستجو است که ابتـدایی‌تـرین روش برای حل یک مسئله است.

برای حل یک مسئله از طریق جستجو سه کار ۱ـ فرموله سازی هدف ۲ـ فرموله سازی مسئله ۳ـ جستجو، باید انجام شود.

** در این درس ممکن است برخی مثال‌ها از میان شهرهای کشورهای مختلف بیان گردد! **

برای مثال می‌خواهیم از شهر آراد به شهر بخارست برویم برای حل این مسئله؛ ابتدا مفصد مشخص می شود، یعنی بخارسـت (فرموله سازی هدف) و مورد بعدی که بررسی می‌شود اینکـه در چـه شـهری هسـتیم (فرمولـه سـازی مسـئله) و در نهایـت مسیرهای حرکتی را نیز مشخث می‌کنیم.

برای فرموله‌بندی مسئله مواردی همچون؛ وضعیت‌های ممکن وابسته به دنیا برای حل مسئله کدامند؟ چه اطلاعاتی بـرای عامل در دسترس می‌باشند؟ رفتن یک عامل از وضعیتی به وضعیت دیگر چگونه می‌تواند باشد؟ در نظر گرفته می‌شود.

برای فرموله بندی هدف نیز می‌توان موارد؛ وضعیت هدف چیست؟ ویژگی‌های مهم وضعیت هدف چه مـی‌باشـند؟ چگونـه عامل می‌فهمد به هدف رسیده است؟ در این مورد بررسی می‌کند که آیا چند وضعیت پایانی ممکن وجود دارد؟ و آیا آنها باهم برابراند یا برخی بهتراند؟ را بررسی کرد.

انواع مسائل

مسائل تک حالته (Single State)

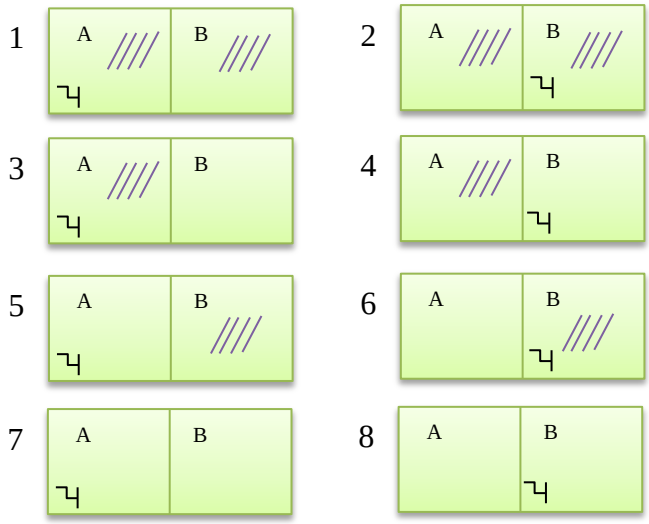
مسائلی هستند که در آنها محیط قطعی و کاملاً مشاهده‌پذیر است. در این مسائل راه حل دنباله‌ای از اکشن‌ها است.

مسائل بدون حسگر (Sensor Less)

محیطی غیرقابل مشاهده دارند. در این مسائل راه حل دنباله‌ای از اکشن‌ها است.

مثال جاروبرقی را بدون سنسور (حسگر) در نظر بگیرید. (قاعدتا هر ۸ حالت ممکن برای آن برنامه ریزی شده است)

حالت‌های باور ۱:



$\{1,2,3,4,5,6,7,8\} \rightarrow^{Right} \{2,4,6,8\} \rightarrow^{Clean} \{4,8\}$
$\rightarrow^{Left} \{3,7\} \rightarrow^{Clean} \{7\}$

در مسائل بدون سنسور حالت باور باید در هر مرحله کوچکتر شود.

از آنجا که به هدف رسیدیم، (در مثال فوق) مشاهده می‌شود که راه‌حل دنباله‌ای از اکشن‌ها است.

* حالت باور؛ مجموعه‌هایی را گویند که فکر می‌کنیم اکنون در آنها هستیم.

مسائل احتمالی (Contingency)

این مسائل، مسادلی هستند که محیطآنها غیر قطعی و یا مشاهده‌پذیر نسبی است که مهم غیرقطعی بودن محیط است و مشاهده‌پذیر نسبی بودن اهمیت چندانی ندارد. در حالت عادی سه مرحله حل مسدله (فرموله سازی ـ جستجو و اجرا) به ترتیب اجرا می‌شوند؛ اما در این حالت (احتمالی) پس از انجام کار محیط دوباره سنس شده تا تغییرات بررسی گردد. پس در این مسئله‌ها جستجو و اجرا یکی درمیان ممکن است تغییر کند.

مسائل اکتشافی (Exploration)

مسائلی هستند که فضای حالت آنها نامشخص است. یعنی آنکه امکان ایجاد وضعیت وجود ندارد. در اینگونه مسائل برای رسیدن به هدف کار انجام نمی‌شود، بلکه برای شناسایی محیط کار انجام می‌شود.

نکاتی که برای فرموله‌سازی باید رعایت شوند

تجدید یا ساده‌سازی حالات و اعمال.

چهار مورد زیر برای فرموله‌سازی مسئله استفاده می‌شوند.

۱ـ حالت‌ها ۲ـ اعمال ۳ـ آزمون هدف (Goal Test) ۴ـ هزینه‌ی مسیری

* به هر چیدمان یا صورت وضعیت محیط یک حالت می‌گوییم.

* در فرموله‌سازی حالت اولیه مهم است، حالت اولیه حالتی است که جستجو از آن آغاز می‌گردد.

* در تعریف اعمال باید مشخص شود چند اکشن وجود دارد و در هر حالت خاص چه اکشن‌هایی می‌توان انجام داد!؟

* تابع پسین (Successor Function): تابعی است که مشخص می‌کند در هر حالتی چه اکشن‌هایی می‌توان انجام داد و خروجی هر اکشن چه حالتی است!؟

* آزمون هدف یعنی انکه وقتی به حالتی رسیده‌ایم توانایی بررسی آنرا داشته باشیم و مشخص کنیم که به هدف رسیده‌ایم یا خیر!؟

* هزینه‌ی مسیری بیانگر مجموع هزینه‌های پرداخت شده از حالت شروع تا حالت فعلی است.

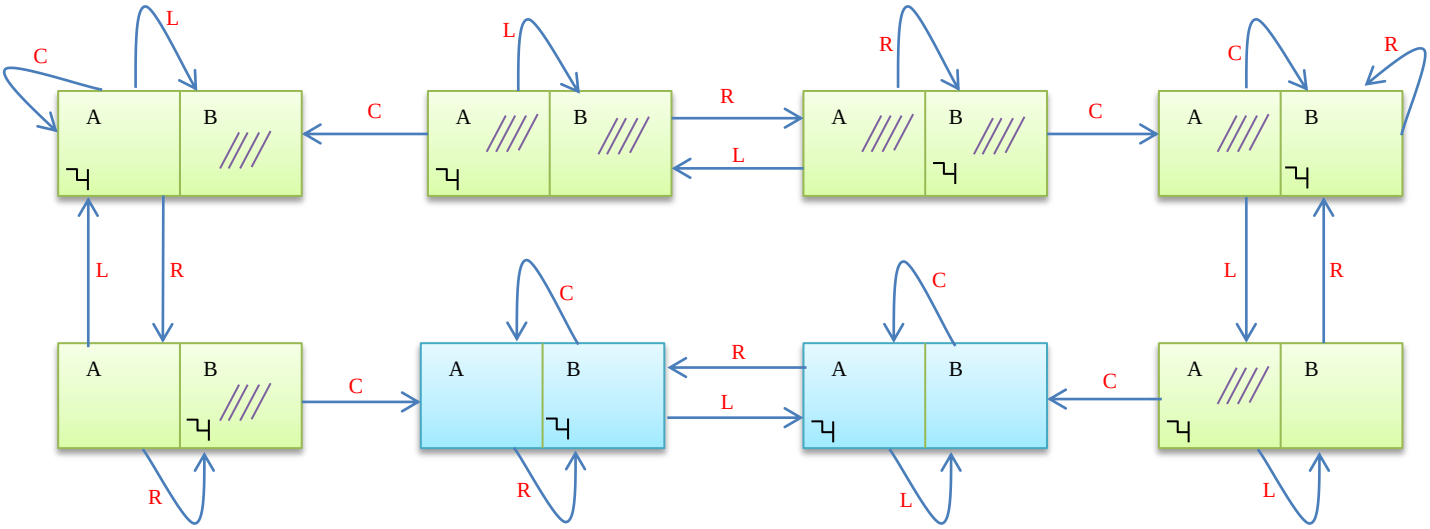
* در تعریف مسئله سه مورد اول (حالت‌ها، اعمال و آزمون هدف) اجباری و هزینه‌ی مسیری اختیاری است.

* راه‌حل دنباله‌ای از اعمال است که حالت ابتدایی را به حالت هدف می رساند و پس از انجام موارد فوق قابل انجـام وبررسـی است.

برای مثال ۴ مورد بیان شده را برای دنیای جاروبرقی بررسی می‌کنیم؛

شکل زیر تابع پسین دنیای جاروبرقی است:

Left :L Right :R Clean :C



ـ تعداد حالت‌ها: هشت

ـ اکشن‌ها: Left, Right, Clear

ـ Goal Test دو حالت آبی رنگ است (هر دو اتاق‌ها تمیز است)

ـ هزینه؛ هر اکشنی کخ به حالت آخر برود (تعداد حالت‌ها و اکشن‌ها)

مثال) معمای ۸تایی (8-Puzzle) زیر را فرموله‌سازی کنید؛

۷	۲	۴
۵		۶
۸	۳	۱

حالت اولیه

	۱	۲
۳	۴	۵
۶	۷	۸

حالت نهایی

ـ برای ساده‌سازی چهار اکشن بالا، پایین، چپ و راست را برای خانه‌ی خالی در نظر می‌گیریم؛ (به جای آنکه برای هر خانه در نظر بگیریم) که در این‌صورت از ۳۲ اکشن فقط چهار (۴) اکشن خواهیم داشت.

ـ آزمون هدف، قرار گرفتن خانه‌ها در جای خود و رسیدن به حالت نهایی است.

ـ هزینه‌ی مسیری؛ هر حرکت خانه‌ی خالی را یک هزینه می‌دهیم و در نهایت محاسبه می‌کنیم خانه‌ی خالی چقدر جابه‌جا شده است.

الگوریتم‌های جستجوی درختی (Tree Search Algorithm)

حالت اولیه ریشه‌ی درخت است. جستجوی درختی یک جستجوی Offline است؛ یعنی آنکه عامل ابتدا جستجو را کامل انجام می‌دهد و سپس اجرا می‌نماید و حساب‌ها و تفکرات داخلی در عموم اجرا نمی‌شوند.

تولید گره (Generate)

به معنی ایجاد یک گره و اختصاص حافظه به آن است.

گسترش ـ بسط (Expand)

ایجاد تمامی فرزندان یک گره را بسط گویند.

در جستجوی درختی ابتدا گره‌ی ریشه را وارد صف کاندیداها می‌کنیم. (در این جستجو صفی داریم که گره‌هایی که کاندید گسترش هستند وارد آن می‌شوند). سپس گره‌ی ریشه را گسترش می دهیم و تمامی فرزندان آن وارد صغ کاندیداها می‌شوند. پس از آن از میان گره‌های برگ موجود در صف کاندیدا با توجه به استراتژی جستجو یک گره را انتخاب و انرا گسترش می‌دهیم. این عمل آنقدر تکرار می‌شود تا دیگر گره‌ای برای گسترش باقی نماند، یا اینکه گره‌ی انتخاب شده برای گسترش، گره‌ی هدف باشد.

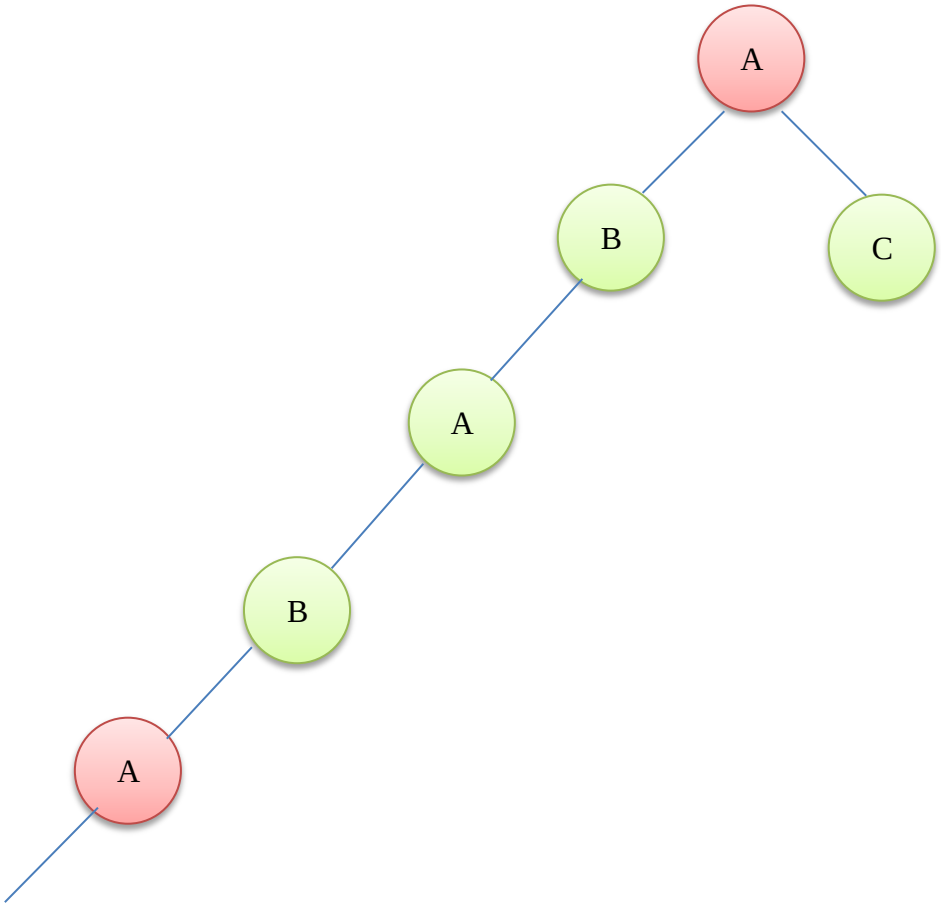
در درخت جستجو هر گره بیانگر یک حالت در مسئله است، اما Node با حالت متفاوت است. گره یک ساختمان داده است که اطلاعاتی در آن قرار می‌گیرد؛

۱ـ حالت ۲ـ عمق ۳ـ مشخص کردن والد ۴ـ اعمال قابل انجام گره ۵ـ هزینه‌ی مسیری

به کلیه‌ی حالت‌های یک عامل فضای حالت گفته می‌شود.

در یک درخت جستجو ممکن است چند حالت متنهای داشته باشیم اما خود درخت نامتنهای باشد.

در مثال زیر دو حالت A مشخص است که هر کدام با دیگری متفاوت است. (تفاوت از لحاظ عمق، هزینه‌ی مسیری و...)



استراتژی جستجو

برای بررسی الگوریتم‌های جستجو ۴ معیار تعریف می‌شوند که عبارتند از:

۱ـ کامل بودن (Completeness): اگر مسئله‌ای دارای جواب باشد و استراتژی جستجوی مورد نظر همیشه بتواند آنرا پیدا کند به آن استراتژی، استراتژی کامل گوییم.

استراتژی کامل همیشه در صورت وجود جواب آنرا پیدا می‌کند، در غیراینصورت کامل نیست.

۲ـ بهینه بودن (Optimality): اگر در مسئله‌ای بیش از یک مسیر به جواب وجـود داشـت و یـا دو جـواب متفـاوت وجـود داشت، الگوریتمی بهتر است (مسیری بهتر است) که هزینه‌ی مسیری کمتری داشته باشد. به عبارت دیگر همیشـه ارزانتـرین مسیر به جواب در این استراتژی مورد نظر است.

۳ـ پیچیدگی زمانی (Time Complexity): تعداد گره‌ی تولید شده تا رسیدن به جواب در هوش مصنوعی پیچیدگی زمانی در نظر گرفته می‌شود، چرا که جستجوی ما بر اساس درخت و گراف است.

برای محاسبه‌ی تعداد گره‌های تولید شده به متغیرهای زیر نیاز است.

الف) Branching Factor (ضریب انشعاب یا فاکتور شاخه‌بندی): حداکثر تعداد فرزندان یک گره در درخت کـه بـا b نمـایش داده می‌شود.

ب) Depth (عمق): عمق کم هزینه‌ترین جواب که با حرف d نمایش داده می‌شود.

ج) Max: عمق طولانی ترین مسیر درخت؛ به عبارتی دیگر اندازه‌ی طولانی‌تری مسیر یا حداکثر عمق درخـت و بـا حـرف m نمایش داده می‌شود.

۴ـ پیچیدگی مکانی (Space Complexity): میزان حافظه‌ی مورد نیاز برای رسیدن به جواب (در اینجا تعـداد گـره‌هـای کاندید بسط برای رسیدن به جواب) است.

به صورت کلی دو نوع استراتژی جستجو وجود دارد:

۱ـ جستجوی ناآگاهانه (Uninformed):

۲ـ جستجوی آگاهانه

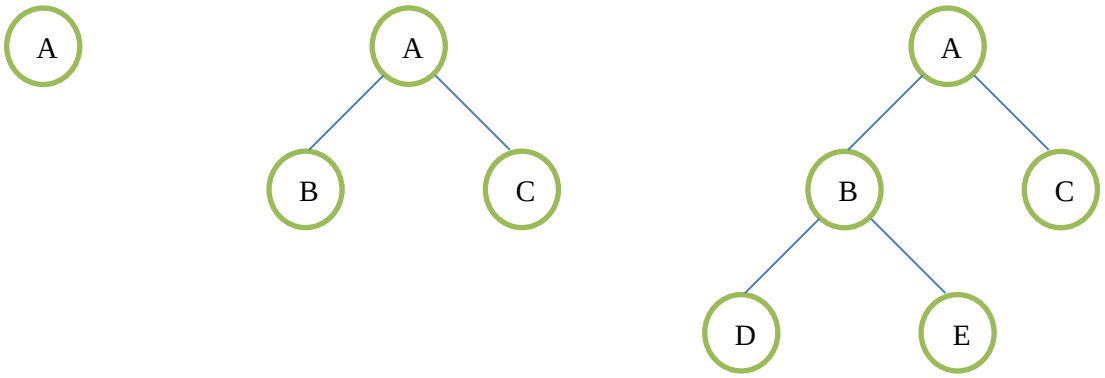
جستجوهای ناآگاهانه

ساده‌ترین الگوریتم‌ها جستجو هستند، چرا که هیچ اطلاعات اضافی که در شرح مسئله وجو دارد را استفاده نمی‌نمایند.

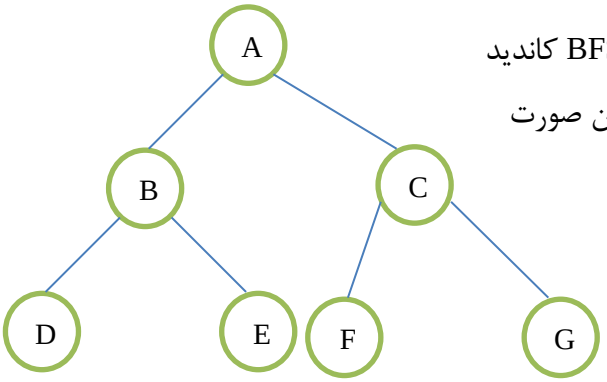
A) جستجوی اول سطح (Breadth Frist Search)

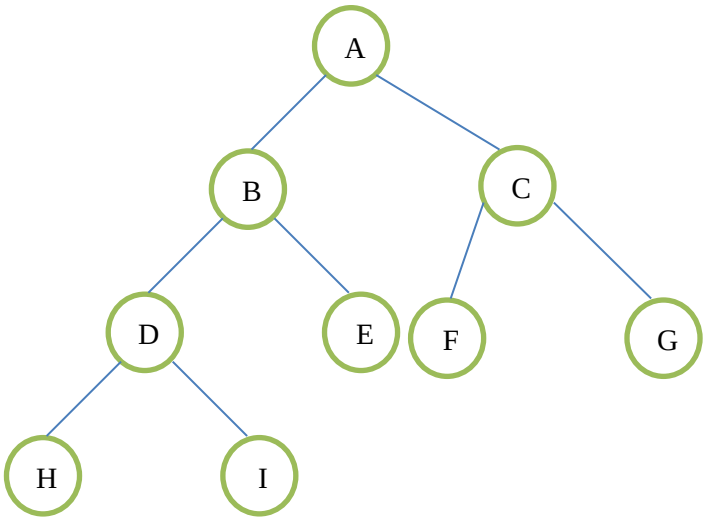
به این جستجو به صورت مخفف BFS گوییم. در این روش از جستجو در هنگام انتخاب گره‌ها برای گسترش گـره‌ای انتخـاب می‌شود که عمق کمتری داشته باشد؛ در صورتی که عمق گره‌ها برابر باشند، گره‌ای کـه زودتـر تولیـد شـده باشـد گسـترش می‌یابد. پیش‌فرض در درخت گره‌ی سمت چپ است که زودتر تولید می‌شود و در گراف پیش‌فرض بر اساس حروف الفبا است.

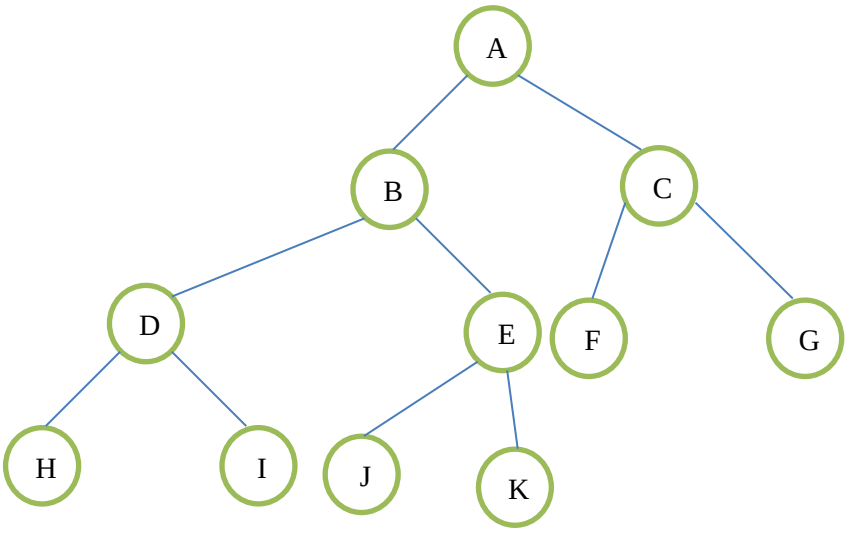
برای درک بهتر ایت الگوریتم به مثال زیر توجه کنید؛ در این مثال گره‌ی G به عنوان هدف فرض شده است.

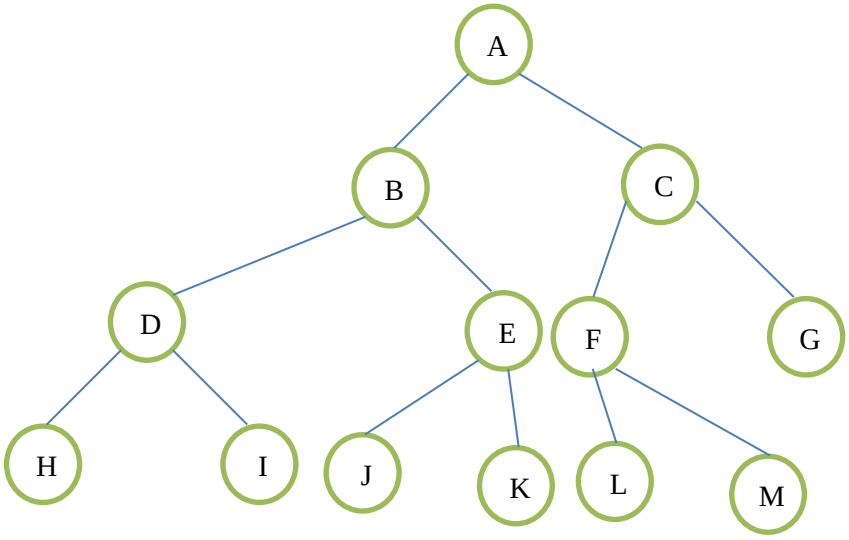


در اینجا با اینکه G تولید شده است اما به دلیل اینکه طبق تعریف BFS کاندید بسط گره‌ای است که زودتر آمده است (D) و آزمون هدف در مورد آن صورت می‌گیرد نه گره‌ی G









در نهایت گره‌ی G برای بسط انتخاب می‌شود که قبل از آن آزمون هدف روی آن انجام می شود و چونکه خود گـره‌ی هـدف است کار بسط متوقف شده و ادامه نمی‌یابد.

ترتیب تولید گره: A – B,C – D,E – F,G – H,I – J,K – L,M

ترتیب گسترش (ویزیت): یعنی اینکه تا رسید به جواب چه گره‌هایی گسترش پیدا کرده‌اند.

A – B – C – D – E – F – G

مسیر رسیدن به جواب: A – C – G

به دلیل اینکه پاسخ فوق یک مسیر است پس بهینه نیز است.

دقت شود در بهینه بودن ترتیب تولید و گسترش مهم نیست و فقط مسیر بهینه بررسی می‌شود.

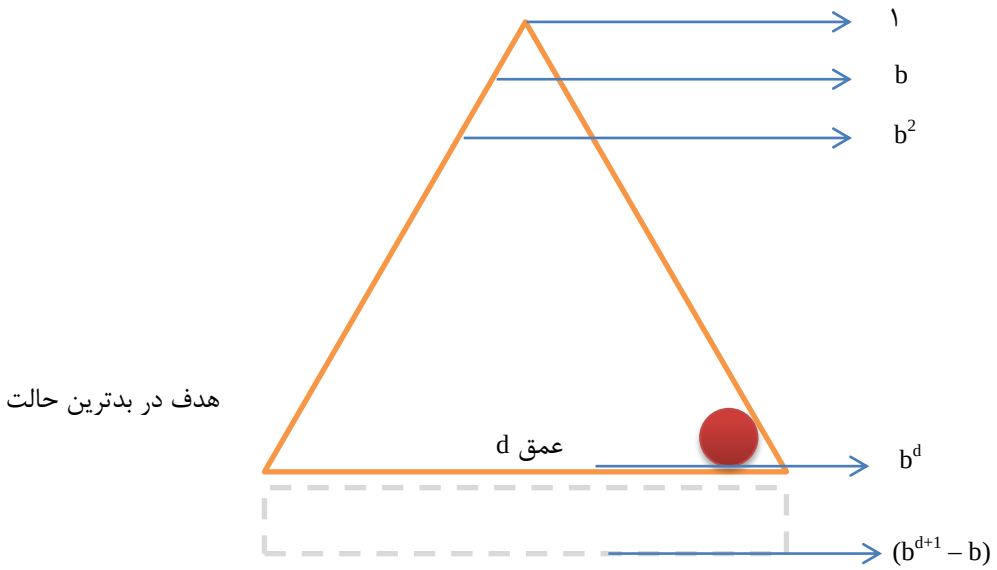
بررسی چهار معیار برای BFS:

۱ـ چون همیشه سطر به سطر جستجو انجام می‌شود و به جواب می رسد پس معیار کامل بودن را دارا است.

اگر در سطر دوم درخت تعداد گره‌ها بی‌نهایت باشد پس به سطر سوم نمی رود، اما چون این امر بعید است که از یک حالت به بی‌نهایت حالت برویم پس بدون در نظر گرفتن این موضوع می‌گوییم معیار کامل بودن را دارا است.

۲ـ روش BFS چون همیسه سطح به سطح بررسی را انجام می‌دهد و اگر جواب بیش از یکی بود همیشه کم عمق‌ترین جواب را پیدا می‌کند پس معیار بهینه را نیز دارا است.

۳ـ پیچیدگی زمانی: $1+b+b^2+\ldots+b^d+b^{d+1} = O(b^{d+1})$

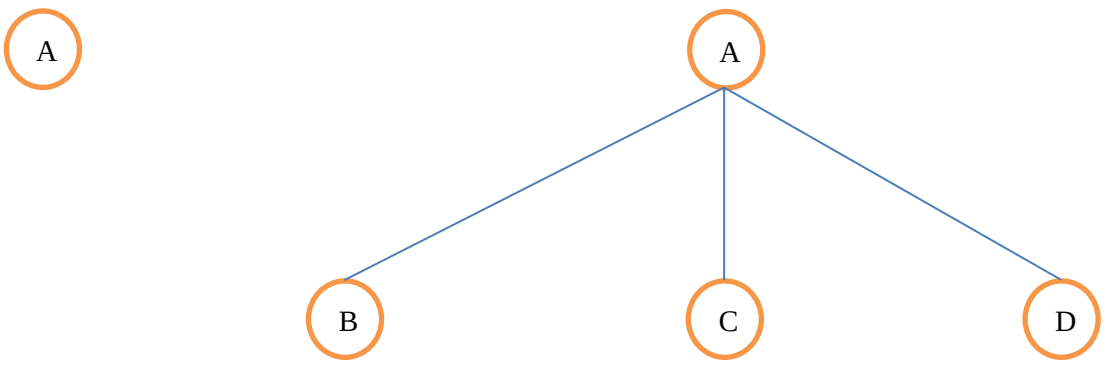


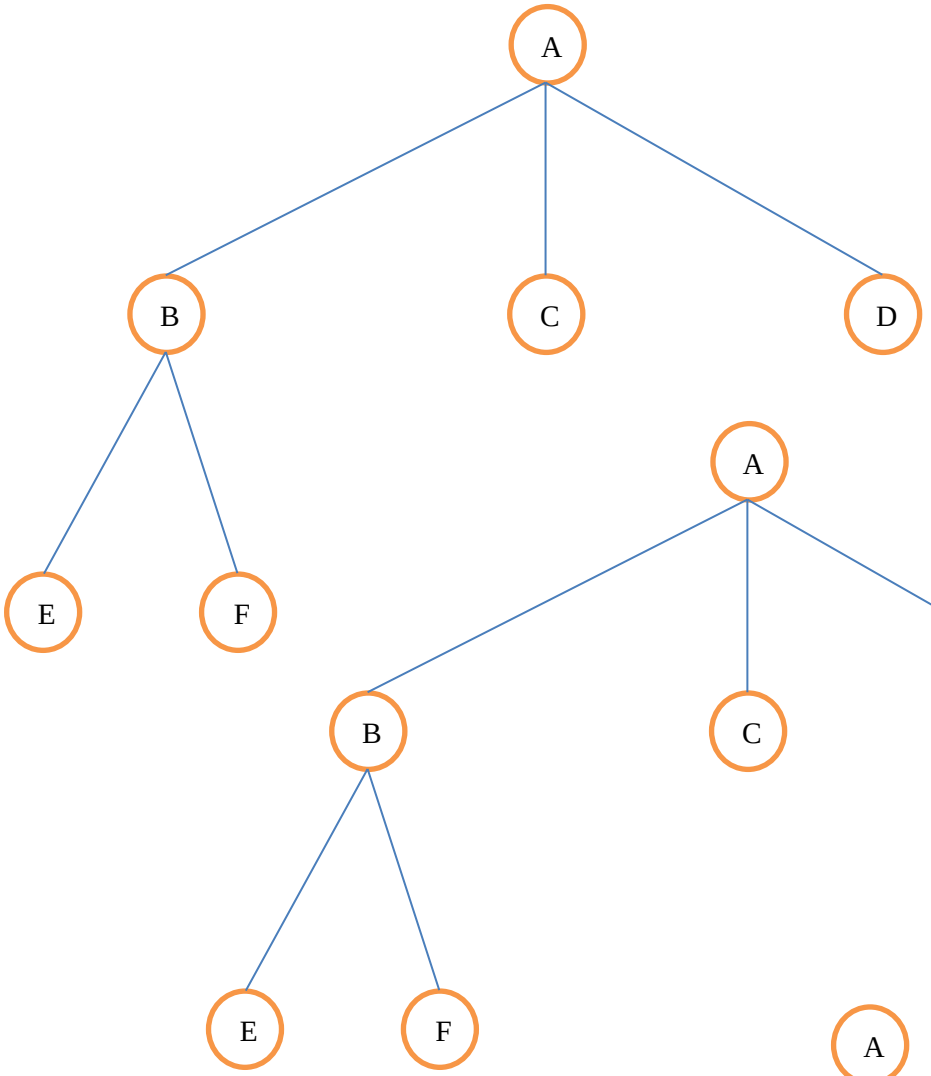
۴ـ پیچیدگی مکانی: در روش BFS چون سطحی عمل می‌کند و ممکن است جواب در فرزند گره باشد، پس هـیچ گـره‌ای از حافظه حذف نمی‌شود، در نتیجه پیچیدگی مکانی آن نیز b^{d+1}

B) جستجوی اول عمق (Depth First Search)

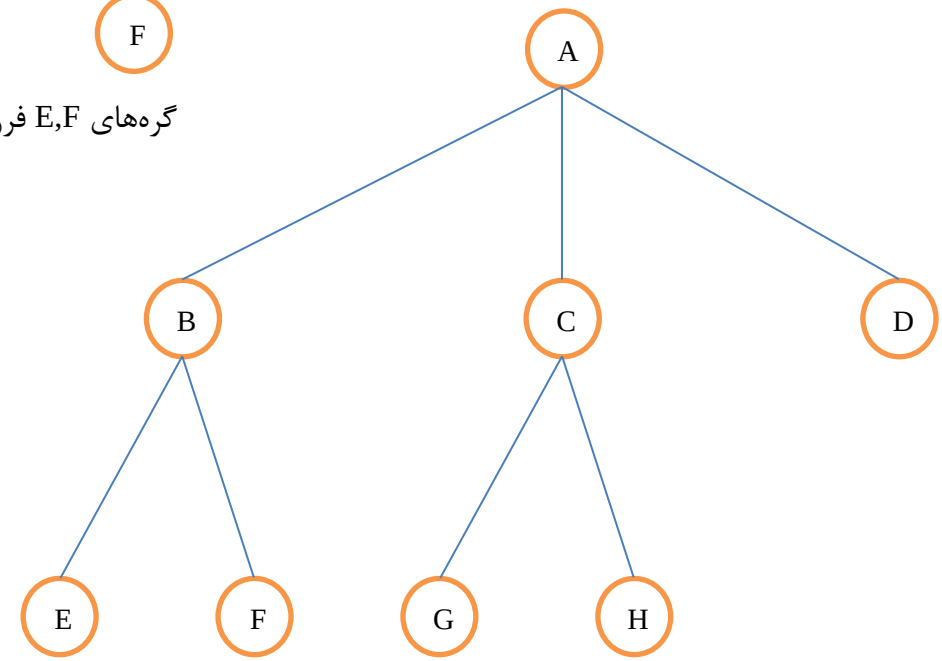
در این روش از بین کاندیداهای موجود، کاندیدایی که عمق بیشتری داشته باشد گسترش پیدا می‌کند. اگر عمق گره‌ها یکـی باشد گره‌ای که زودتر تولید شده گسترش پیدا می‌کند. این الگوریتم را به اختصار DFS گویند.

برای درک این الگوریتم مثال زیر را بررسی کنید: هدف در این مثال G فرض شده است.





گره‌های E,F فرزند ندارند



پس از تولید گره‌ی G برای بسط و بررسی آزمون هدف انتخاب می‌شود و خود هدف است

ترتیب تولید گره‌ها: A – B,C,D – E,F – G,H

ترتیب گسترش: A – B – E – F – C – G

مسیر رسید به جواب: A – C – G

بررسی معیارها:

۱ـ در این درخت اگر درخت نامتنهای باشد، یعنی اینکه یا برگشت پذیر باشد؛ E به B برگردد و یا اینکه گره‌ها ادامـه داشـته باشند، هیچوقت به جواب نمی‌رسیم. در DFS این اتفاق بسیار می‌افتد، پس معیار کامل بودن را ندارد.

۲ـ بدلیل اینکه ممکن است ابتدا گره‌ی هدف در عمق بیشتر (هزینه‌ی بیشتر) را پیدا کند (در هنگامی که بیشت از یک هدف وجو داشته باشد و یا چند مسیر به هدف باشد) به همین دلیل معیار بهینه بودن را دارا نمی‌باشد.

۳ـ با توجه به اینکه تا حداکثر عمق می‌رود و گاهی اوقات حداکثر عمق بی‌نهایت است ($m=\infty$) پس تعـداد گـره‌هـای تولیـد شده در نهایت b^m است، پس $O(b^m)$ خواهد بود.

۴ـ پیچیدگی مکانی: چون DFS وقتی به برگی می‌رسد که از آن به جواب نمی‌رسد می‌توانـد آنـرا حـذف کنـد پـس $O(bm)$ خواهد بود. به عبارتی دیگر فقط نیاز به نگه‌داشتن شاخه‌ای هستیم که به تازگی جستجو شده است. مزیت الگوریتم اول عمق این است.

با توجه به اینکه الگوریتم DFS از ۴ معیار فقط یک معیار را به صورت مطلـوب دارد پـس در ایـن حالـت الگـوریتم مناسـبی نیست، اما این الگوریتم دارای معیار پیچیدگی مکانی خطی است که بسیار مطلوب است (فضای آن به مراتب بهتر از BFS که نمایی می‌باشد، است.) و در اینجا می‌توانیم با تغییراتی برخی معیارها را به این الگوریتم اضافه نماییم.

جستجو با عمق محدود

اولین حالت تغییر، محدود کردن عمق در این جستجو است، یعنی جستجو تا یک عمقی در درخت پـایین بـرود و پـس از آن دیگر ادامه ندهد. در این الگوریتم، یک مشکل است که اگر جواب در عمق مورد نظر نبود و در عمق‌های زیـرین آن بـود چـه می‌شود!؟ پس در این الگوریتم پیدا کردن عمق مناسب بسیار مهم است.

جستجوی عمیق شونده‌ی تکراری

این حالت بر مبنای جستجوی با عمق محدود است؛ برای اینکه بتوانیم عمق مناسب را پیدا کنیم، جستجوی با عمق محدود را ابتدا با عمق مناسب یک انجام می‌دهیم، در صورت نبودن جواب، آنرا با عمق محدود شده‌ی ۲ انجام می‌دهـیم و ایـن کـار تـا عمق مناسب (محدود شده‌ی) d ادامه پیدا می‌کند. پس در این حالت الگوریتم جدید معیار کامل بودن را بدست می‌اورد چـرا که حتماً به جواب خواهد رسید.

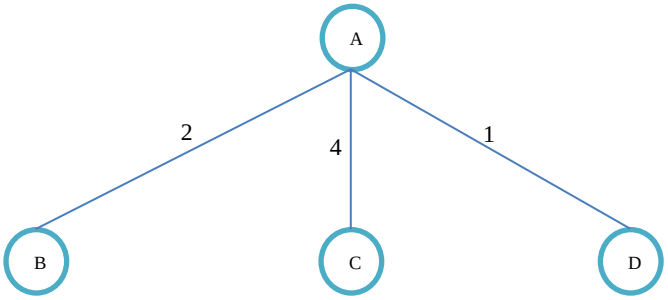
اشکال این روش این است که برخی گره‌ها تولید شده و چند بار، توسعه داده می‌شوند. در سطح یک؛ b گره، تعـداد d مرتبـه تولید می‌شوند، در سطح دو؛ b^2 گره، d-1 مرتبه تولید می‌شوند و... در نتیجه در سطح d، تعداد b^d بار تولید می‌شوند.

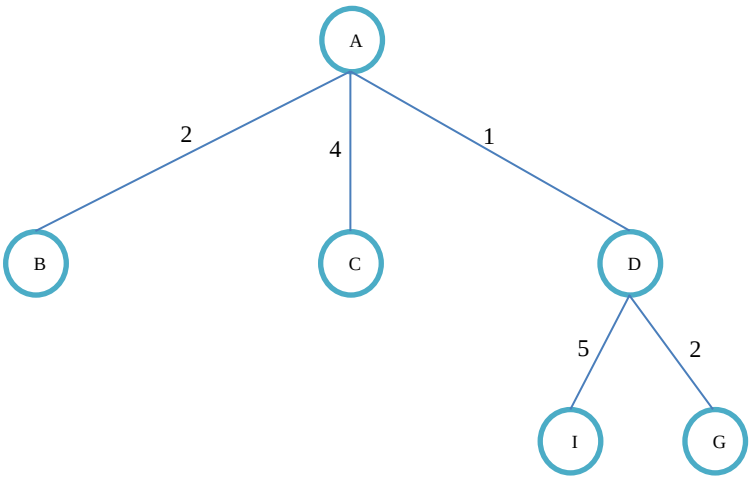
در این روش پیچیدگی زمانی $O(b^d)$ است که مقدار اندکی کمتر از جستجوی اول سطح است، اما مسئله مهم کامل بـودن آن و خطی بودن حافظه‌ی اشغالی آن است. و همچنین بهینه نیز خواهد بود.

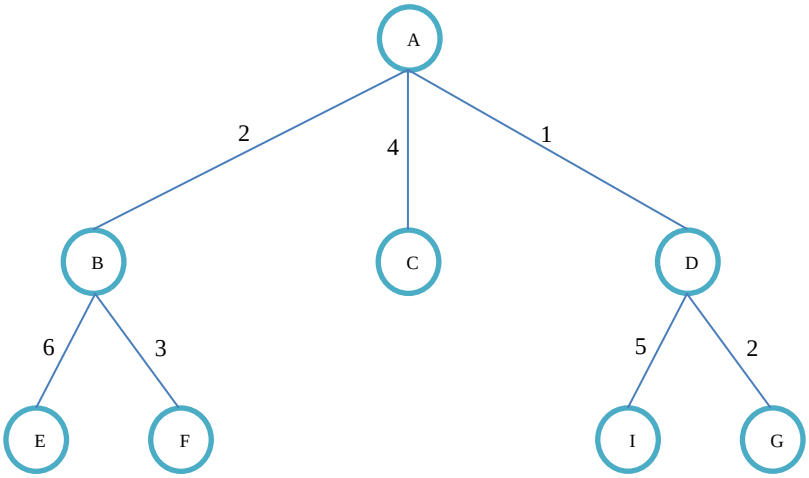
C) جستجوی هزینه‌ی یکسان (Uniform Cost Search)

در این جستجو، گره‌ای که هزینه‌ی مسیری آن کمتر باشد را گسترش می‌دهیم. یعنی اینکه در این الگوریتم حتماً هزینـه‌ی طی کردن مسیر از گره به گره‌ی بعدی نیز بر روی یال آن ثبت شده است. این الگوریتم برای حل کردن مشکل الگوریتم اولی سطح در زمان وجود هزینه‌ی مسیری بوجود آمده است، چرا که جستجوی اول سطح در صورتی که هزینه‌هـا یکسـان نباشـد بهینه نخواهد بود. دقت شود این الگوریتم در صورت وجود هزینه‌های یکسان همانند الگوریتم اول سطح عمل خواهد کرد.

برای درک این الگوریتم به مثال زیر توجه کنید؛







این عمل تا رسیدن به هدف ادامه می یابد.

دقت شود هزینه‌ی مسیر کمتر مورد نظر است نه هزینه‌ی هر یال. (هزینه‌ی مسیری یعنی جمع تمـام هزینـه‌هـا از مبـدا تـا گره‌ی مورد نظر است.)

بررسی معیارها:

۱ـ به شرطی که وزن‌های داده شده صفر و عدد منفی نباشد معیار کامل بودن را دارد؛ اگر وزن منفی باشت ممکـن اسـت در یک چرخه قرار گیرد.

۲ـ بهینه است چرا که همیشه دنبال مسیر کوتاه‌تر است.

۳ـ پیچیدگی زمانی: اگر هزینه‌ی مسیر بهینه (C^*) باشد، هزینه‌ها عدد کوچک اما بزرگتر از صفر است.(شرط کامل بودن (ε))

پس خواهیم داشت: $\frac{C^*}{\varepsilon}$ در نتیجه پیچدگی زمانی به صورت $O(b^{\lfloor \frac{C^*}{\varepsilon} \rfloor + 1})$ خواهد بود.

۴ـ پیچیدگی مکانی نیز همانند پیچیدگی زمانی خواهد بود.