



هوش مصنوعی ۱

✓ AI ← بهترین هوشمندی ← فکر انسانی، ذخیره دانش نسبت به از تعامل محیط، توانایی یادگیری از محیط، تصمیم ستهای هوش
 عملکرد عاملان در محیط (این از هم جداست)

✓ هوشمندی ≡ یادگیری، قابلیت برقراری تعامل با محیط داشته باشد و بتواند یک دانش اولیه ای که درونش بوده از آنجا پس از تعامل با محیط بتواند
 ادراک را بهبود ببخشد

✓ عاطفی می تواند خوب عمل کند که عوامل را کنار بگذارد به هدف برسد.

✓ نیای هوشمندی ← هوشمندی انسان ← درخت بهترین روای آنقدر دور است که انسان تقریباً منقطع است. این به انسان باسد چون غوی عملکرد فکر
 باید آموختن و برای شناختن شمره سست انداختن از شیوه های یادگیری و در نهایت
 فضا عمل کردن ← بهترین روز game out یا عملکرد به نسبت دیگر

نحوه فکر agent vs نحوه عمل agent
 گاهی با نون و هم نیست (مثلاً agent) خطوری می بیند ولی عملکرد آن طوری هست که بهایار کارایی را
 بلا می برد
 به تفکر عاملان یعنی بر اساس توان منطقی (مثلاً) دارو عمل می کند ولی فنیان کار همیشه بهایار کارایی (برای ما ای دفعی نه!)
 بهیچ به طور کلی دارو درست عمل می کند یا نه به غوی فکر کردن کاری نه استیم
 ریز تر شدن در جزئیات

* عملکرد شبیه انسان ← تست تورینگ → این در ۳۰٪ مواقع نتواند کسی که دارد با ماشین chat می کند دفعه اول انسان هست یا ماشینی
 ادراک داشته تست در ۵۰٪ کرده (در ۵۰٪ دقیقه) → می شه هر سوالی پرسیم
 (دیده نمی شه) که ماشین هست یا نه

* ویژگی های لازم جهت پاسخ کردن این تست :
 ۱. استنتاج که نمره هائی که دارد (قدرت استنتاج)
 ۲. عین زبان طبیعی → عین انسان که هنوز دینی شمره سست
 ۳. توانایی یادگیری → از مجموعه توانی سبک از شی پرسیده شده چیزی دارو یاد می گیره
 ۴. غوی تشخیص داشتن ذخیره داشتن و تجزیه عین اطلاعات → بتوانیم از چیزهائی که داریم نتایج جدید در بیاریم
 این در تست قرار بود تعامل داشته باشد → به رویایک و عین تصویر هم داشته باشد
 چون عین دینی را نمی نارد چیزی برای ما از شمره سست! → بسیار با نون پس شمره سست باید فکر کنیم پاسی نیست!
 شناسایی عملکرد ذهن انسان و مدلی معادل با آن را درست کنیم عملکرد نود ها...

تفکر شبیه انسان → به چابقتی در ادراک این که فنی انسان عمل کنیم چنانچه این قرار دهیم که می خواهیم به گیری هدف برسیم.

* منطقی فکر کردن → از روی knowledge base که دارد → (استنتاج که برای ادراک این کاره * action) این که پیش پشته هار شده را این کاره
 پس این منطقی فکر کردن که باید فنی استنتاج که به در مادی فون

له (مثل بر خود چیزی به چشم)
 یعنی در حالت میانی performance را حد اکثر کند
 در محیط احتمالی

عملکرد منطقی → فنی عملی این کاره که به حد اکثر performance منجر شود
 منطقی رفتار کردن
 بهیچ فنی تر از فنی ای که گفته! چرا؟ مثلاً فنی منطقی که حالت خاصی تر همین هست → مثال شبیه عملکرد آنی که باید کاری در
 فوری این کاره بهیچ در جهای فنی منطقی فکر کنیم
 در دسر شمره سست انسان بر هم نزارد → مثل آموختن شمره سست و شمره سست

History: state های لحظه‌ای قبل و حال در حالت کنونی و بر اساس اون action
 $P: P^* \rightarrow A$ یک دنباله از عمل پیشنهاد دهنده
 agent: هدفی
 action: عملی که در لحظه فعلی در مورد state فعلی انجام می‌دهد
 perception: داده‌هایی که از محیط در لحظه فعلی در دسترس است

سطوح هوشمندی انسان:

- سطح ۱: باز کردن در
- سطح ۲: تشخیص چهره
- سطح ۳: تشخیص صدا
- سطح ۴: تشخیص عواطف
- سطح ۵: تشخیص نیت
- سطح ۶: تشخیص اهداف
- سطح ۷: تشخیص ارزش
- سطح ۸: تشخیص معنای عمیق
- سطح ۹: تشخیص خلاقیت
- سطح ۱۰: تشخیص خرد

 AI: هوش مصنوعی

۱) حل مسئله: problem solving = عمل سازی آن با search
 state: حالتی که در آن هستیم
 action: کاری که می‌توانیم انجام دهیم
 goal: هدفی که می‌خواهیم به آن برسیم
 مثال: ۸ پازل
 state: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 action: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 goal: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

۲) domain specific knowledge: دانش خاص در مورد مسئله
 مثال: ۸ پازل
 state: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 action: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 goal: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

۳) reasoning: استدلال
 agent: چیزی که می‌تواند با استفاده از دانش خود، تصمیم بگیرد
 state: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 action: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 goal: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

۴) learning: یادگیری
 perception: درک از محیط
 action: عملی که می‌توانیم انجام دهیم
 goal: هدفی که می‌خواهیم به آن برسیم

۵) AI: هوش مصنوعی
 state: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 action: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 goal: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

هوش مصنوعی ۱: agent
 state: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 action: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$
 goal: $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

هوش
۲

* نام دنیای واقعی هستیم که دنیای درونی را ساخت و act مناسب به agent پیشنهاد کند! $P \rightarrow A$

له ذخیره چنین تابعی نام ممکن است!
له فرض کنیم ورودی مادر هر لحظه $|P|$ (کارندایی: $percept$ ؛ هر وقت که در $|P|$ به حالتی باشد که ورودی در یک لحظه بوده داشته باشد به صورت $percept$ و $History$ تا به لحظه t پیش رفته P_1, \dots, P_t باشد و P_t ها همیشه باشد و مقدار آن ها در هر لحظه $|P|$ باشد
له به همین دلیل $table$ برای نگهداری حالات باید به نام $table$ باشد که $table$ را در یک یا چند تغییر نمی دهد
له خلاصه نام داشته اطلاعات دنیای بی جای نگهداری کل دنیای کل $table$ را در یک یا چند تغییر نمی دهد

A	B
---	---

* مثال جارچی: جارچی در هر خانه آن قرار می گیرد و آن خانه را تمیز می کند یا نه پس به وضعیت بعدی می رود و خانه را تمیز می کند

له $percept$: وضعیت تغییر بر روی درخت که درون هست و موقعیت اش که گویاست (۵، ۸) $\leftarrow [A, Dirty]$
له $action$ عملیات: حرکت کند (راست، چپ) - تمیز کند یا هیچ کاری نکند
له $agent\ table$: به جای جدول که برای دنیای act هست می باشد P و یک مقدار می بهتر کردش! $\rightarrow agent\ program$

agent منطقی: کار درست را از آن در هر $performance$ \uparrow ، $agent$ منطقی تر \rightarrow دنبال max کردن $performance$ شود
له این محیط تعدادی باشد گاهی اوقات یک act باعث $performance$ \uparrow شود ولی یکبار دیگر کم! $Expectation$ (اینه میزان کارایی)

میزان کارایی: نسبت به مسئله $agent$ این عامل تنظیم می شود
له مثال جارچی: ۱) تعداد خانه های تمیزی که در طول زمان داشته
۲) برای جارچی (کارایی) اشغال در هر خانه (کارایی) جارچی
۳) دفعاتی که تمیز کرده! \rightarrow ۱۱ معنی تمیز باشد! ۱۲ برای جارچی (کارایی) اشغال در هر خانه (کارایی) جارچی
له (خوب نیست این!)

* تو مثال جارچی: منطقی بودن به چیزهایی بستگی دارد؟ ۱) معیار کارایی \rightarrow می تواند هم در طول زمان تعداد خانه های تمیز باشد و هم انرژی که صرف می کند! $\rightarrow agent$ می تواند خانه تمیز می شود و خانه تمیز می شود و خانه تمیز می شود
له ۱۶ خصوصیات محیط: این به مثال محیط امکان تمیزی شدن خانه ها وجود داشته باشد! P معیار کارایی هم عدد تعداد خانه های تمیز در طول زمان هست \rightarrow باید با نرخ تمیزی شدن باید خانه های تمیز می شود
له ۱۷ سنسورها $actuators$ و (مثلاً) در زمین خانه در هر خانه خانه ها است $agent$ که وضعیت هم در زمین (ادون) دست کارایی $agent$ فعلی قرار می دهد

همه چیز دان بودن $omniscience$ vs منطقی بودن $rationality$
له اطلاعات باید \rightarrow امکان به دست یاریم \rightarrow اطلاعات تمیزی تر باشد! \rightarrow این به معنی همه چیز دان بودن نیست
له مثلاً $agent$ ما نمی دانیم که شهاب می افتد و می تواند از میان رد می شود! \rightarrow این برای منطقی بودن فایده دارد
له همه چیز دان بودن عملی نیست!

agent: $agent$ را طبق درونش تمیز می کند یا نه \rightarrow این $percept$ های act را از آن به! \rightarrow $if/else$
له عیب: محیط تغییر کند و نمی دانست که $learning$ و به روز رسانی ندارد
له $agent$ خود مختار خود مختار نیست و دستش باز است \rightarrow $agent$ برای یادگیری \rightarrow هر چه دستش باز تر و مستقل تر $agent$ خود مختار خود مختار! \rightarrow برای فهمش یک ماشین می یادگیری درونش قرار می شود!

* هر چه به انسان نزدیک تر باشد $agent$ خود مختار تر و هر چه ساده تر باشد خود مختار کمتر!
له هر چه که می یاد و می خواند (در دنیای) می تواند از آن یاد بگیرد و آن را به کار می برد (در دنیای) می تواند از آن یاد بگیرد و آن را به کار می برد

task Environment: \rightarrow برای شناختن دنیای مسئله، این $task$ در به \rightarrow بخش فعلی می کنیم!
P: ۱) معیار کارایی
E: ۲) وضعیت محیط
S: ۳) sensor
A: ۴) actuator
 $(PEAS) \rightarrow$

④

ناسی با الله خودنا

ربان روی خفا تو
برای اشعین و فغی
مغیوب در کاخانه

Pacman

fully observable

- partially observable

Deterministic
رسمی

این هکتار که از A به راست معلوم شد agent میماند به فرض فرضیه به خودی نقل
ما احتمال (از) خانه خودی فاند (9) میوه خونی B

stochastic non deterministic
fix ω - free ω

له بهر ایا
نم

single agent

multi agent

اگر چیزی که در دنیا حرکت
کند، خود را، لایم

Discrete

Continuous

هوش

⑤ Episodic vs sequential
 action ما می‌توانیم واقعاً بهر جا باشیم و در حالت فعلی فقط state الان مهم است. این حالت در آینده اثری ندارد.
 در sequential ما یک حرکتی را می‌کنیم و حرکت‌های بعدی را بر اساس حرکت‌های قبلی داریم. در اینجا ما یک action را می‌کنیم و در آینده آن اثر را داریم.

* همیشه در حالت فعلی فقط حرکت‌های قبلی را داریم و حرکت‌های بعدی را بر اساس حرکت‌های قبلی داریم.
 فقط در حالت فعلی در نظر می‌گیریم و حرکت‌های بعدی را بر اساس حرکت‌های قبلی داریم.
 * هیچ‌کدام از این‌ها در حالت فعلی اثری ندارند!

⑥ static vs dynamic
 static - وسیله‌دار agent با یک محیط ثابت است. تغییر نمی‌کند. تا زمانی که محیط ثابت است.
 dynamic - در زمان تغییر می‌کند agent با یک محیط تغییر می‌کند. تا زمانی که خود را.
 semidynamic - به جز زمان تغییر می‌کند agent با یک محیط تغییر می‌کند. تا زمانی که با سرعت!

⑦ known vs unknown
 known - محیطی که قواعد آن را می‌دانیم. بازی Pacman هر دو را می‌داند.
 unknown - قواعد آن را نمی‌دانیم. مثلاً نمی‌دانیم هر دو را می‌داند. بازی Pacman هر دو را می‌داند.
 state - این‌ها کار می‌کنند. به این معنی که state را می‌دانیم و به این معنی که state را می‌دانیم.

* فرق داره با این معنی! در واقع این معنی که هر state می‌داند و به این معنی که state را می‌داند.
 در این حالت‌ها اطلاعاتی نداریم و این‌ها حرکت‌های تازه می‌کنیم. به این معنی که state را می‌دانیم و به این معنی که state را می‌دانیم.
 (به هم ربطی ندارن) این معنی که state را می‌دانیم و به این معنی که state را می‌دانیم.

جلسه ۳



محیط: ۱. Fully observable? بله چون کل صفحه رو داریم می‌بینیم.
 ۲. Stochastic - چون این Pacman می‌خورد و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ۳. Dynamic - تا زمانی که داریم تغییر می‌کنیم و این‌ها حرکت می‌کنند.
 ۴. Sequential - چون به حرکت می‌کنیم و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.

بازی Pacman

* نکته مهم: هم روی فواصل تاثیر ندارد.
 ۱. partially observable - قوه بینایی ما در این بازی را نداریم و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ۲. Stochastic - مثلاً می‌توانیم چهار راه را بگیریم و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 * چون partially observable و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 * تا زمانی که این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ۳. Multi - مسافت (همه رفت و آمد) و در اینجا با سایر ماشین‌ها.
 ۴. Sequential - حرکت‌های ما را می‌بینیم و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ۵. Dynamic - تا زمانی که این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ۶. Continuous - حرکت‌های ما را می‌بینیم و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.

راننده تاکسی

agent - محیط مهم این است. action به ما می‌دهد و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 table Punc - این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ① وقتی که این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ② این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.
 ③ این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند و این‌ها حرکت می‌کنند.

9

agent Simple reflex

هوش ۲

دنباله سری را تک نگه داریم - هر ردی که او برایش تعین داریم که چه action ای انجام دهیم

حالا Pully observe باید باشد - چون ما این داریم state را بطور کامل در بریم که به دریم چه action ای باید

* جز عمل فعلی که فوری استغاده شده! طرحی بر می کند table را

flexible نیست! - برای بهانه که هر شب داره می نشیند و این هدف عوض شود برنامه ریزی به درونی خود

		X	
	1		
0		1	

تک action ها در هم باید ذخیره کنیم و هوش نیست داشتن استیلا که برنامه نویسی را تغییرش قرار داده وجود نمی آید

Model base reflex agent

قرن جدی!

به مثبته به ازای هر state ای که action ای آید به state دراز سنسورهای تیره : rule ها در هم داریم

یک چیزی به نا state تک می دارد که در محیط partially فقه هوش این هست که درم نیست کل چیزهایی که در زمان های مختلف دیدی لازم ناری بگیری تا ببینی چه action ای را ای

کند می کند به محیط partially

Goal base agent

به که rule و قواعد از پیش تعین شده به ما می رسد به ازای هر state چه کاری ای آید به ما می رسد که به اساس ادک خودی هرگز ما معارف می شود - با search کردن به رسیدن به هدف در بریم

بر اساس هدف به کاری باید ای آید به

utility-based agent

معینه اهداف مختلف داشته باشیم که بعضی ها در تعداد هستند - man - دانه ها در کمتر زن خود شود در نه های که گاه ها خود می شوند هم بخوریم (بوه)

Utility مفهومی که از لهو - در شرایطی که به کار می آید به پیچیده است در غلبه لهو نمی توانیم در نظر بگیریم درون اهداف متفاوت باشد یا اهداف در تعداد باشند کن رتبه utility را در داریم

learning agent

کی سافتاری که در بش یا به لهو دیده شده در performance element می نشیند و در نشین کارش کارش می کند

problem generator

learning element از دانش قبلی را اطلاعات که در جریان یادگیری درم آوریم تغییر در دانش agent دهیم

معنی می کند محیط ها مختلف سنسور را تجربه کند - مانع لهو نمی توانیم به رسم می خواهیم محیط ها خلقت را تجربه کنیم یک مقدار exploration در محیط هم داریم به مثبته آدرن احتمال کشف شدن

Critic یا قضا - می که مقدار عملکرد خوب بوده یا نه - برای یادگیری موفق باید agent به درونی داشته باشیم که حکم بدهد یا پاداش داشته باشد به اساس عملکرد

7

هوش ۲

Search برای حل مسائل از طریق

① نقش را به صورت یک نقش جستجو فرمول کنیم
پس از داریم به state و action ها جهت پیمایش در میان state

② goal که باید مشخص شده باشد تا بدان چه رسیدیم → طبق معیار کارایی goal به دست میاید

* Search: یعنی برای رسیدن به goal باید چه دنباله ای از action ها را اتخاذ کنیم. رسیدن از وضعیت اولیه به هدف با کمترین هزینه

* نمونه کسیر برای goal base: هم goal مشخص شده و هم مسئله فرمول شده

له مثل پیدا کردن شهر در نقش: معیار کارایی: در کمترین زمان برسیم به شهر هدف → کمترین مسیر برای رسیدن به شهر مورد نظر

state: شهرها
action: فاصله بین شهرها
state اولیه: Atad
state آخریه: Bucharest

این ساده بود کارون روش معیار ① know → مشخصات ای را داریم و می دانیم با هر action کجاییم ② Discrete → تعداد مشخص حالتها شمار داریم

fully obs → چون نقشه را داریم ③ طبق نقشه حرکت می کنیم deterministic

sensorless → دنی قطع و شافیه نداریم و مسئله ما single state هست چون هر لحظه درین دنیا در یک state قرار داریم

* آنی محیط ما non-observable → بدون سنسور باشد → یعنی هیچی از محیط نمی بینیم → می بینیم مسئله search امل کنیم؟!

مثلا در جابرجی نه می بینیم فونت تخته هست یا نه و نه می بینیم که کجا هستیم ولی می خواهیم به هدف تغییر بردن هم جابرسیم (action روی دنیما)

A	B
0	0

مثلا در جابرجی agent منطقی باشه؟ خب جابجی ایکنی agent که مجبوره به state ها ۸، ۶، ۴، ۲، ۰ می دریم و می بینیم این وضعیت ها قرار داریم به بعد stuck خب یا left stuck

Contingency → با این سنسور نداریم و محیط قطعیه بوده و معلوم بوده اثر هر act چه هست متوجه شدیم که به goal می رسیم

* محیط آنی non deterministic → partially observable هست → وضعیت تخته یا تخته فونت ای را می بینیم ولی بعضی را نه! + که آنی عمل suck ایکنی شوره ممکن است خانه ای که تمیز بوده باید (احتمالی روی آن خانه) کسبه اش رو خالی کنه (با یک احتمال منیم)

+ آنی عمل استرایک [Clean, ۲] باشد توی ص ۷ بودیم! → [Right if dirt then clean] → برای partially هم می ok هست

ولی برای غیر قطعی چون احتمال کشش کردن داره [Right while dirt do suck] → پیچیدگی بیشتر است

* جواب فر حالت های مختلف محیط در شان جابرجی: ① single state → sequence هست! چرا! چرا! چرا! ایضا! چون state اولیه مشخص است و به خاطر known بودیم

می دریم که هر action ما رو به یک محیط قطعیه است → متن به ترتیب ترتیب perception هم نداریم

② sensorless → sequence است! چرا باز جواب sequence می شد؟ چرا! if نداریم → چون احتمال سنسور نداریم برای دریافت ورودی!

③ Contingency: به حسب این که perception چه هست کار ایکنی → if ظاهر شده

مثلا تخته ای هست که آنی به طور کامل state اولیه را داشته باشیم که می بینیم مابقی اطلاعات خودمون بدست میایم!

action → توی این هست از فضای state $(S \rightarrow A)$. تابع action مشخص می کنه که در هر state که ام action درستی است
Action: $S \rightarrow A$ result:

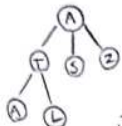
Transition Model → Result چیزی هست که می آید در دنیما → را ایکنی داریم به کجاییم درین → در حقیقت همین تلف transition منم است، این که خطری داریم بین state ها با action ها جابجا بشیم

Goal Test → آیا state ، goal state هست یا نه! ما می بینیم نقشه راه هست چه کردیم طاه مثل وضعیت کش و کش شغیر می بینیم

@sharifjovze96

هوش ۴
 * نکته مهمی که ما به تدریس algorithm می‌پردازیم که توان سازماندهی state ها نسبت به node ها، اما باید بدانیم که node ها (اشیاء) هدف دار و شروع از این و احوال را هر کدام یک state در نظر می‌گیریم یک مسیر در این پیدا کنیم!
 اما ۵ * سازماندهی بزرگ است، نمی‌توان آن را شکل داد!

پس کار در initial شروع می‌کنیم، و همسایه‌هاش را پیدا می‌کنیم، بعد همسایه‌های آن‌ها را جایی جلوی می‌آید به goal می‌رسیم
 * وقتی ما توانایی سازماندهی Tree Search



Tree Search نقشه حرکت agent به از نقشه شروع A، agent می‌توانست به یکی از همسایه‌ها برود، کاری که می‌کنیم این است که همسایه‌های آن را داخل ساختاری به نام Frontier می‌ذاریم. این بزرگ‌ترین حالت ممکن است

* Frontier به کسانی در این ساختار قرار می‌گیرند که گسترش پیدا نکردن و بزرگ‌ترین حالت ممکن است. مانند این گسترش پیدا نکردن
 Frontier [3, 5, 2]

له می‌شود به algorithm که در نظر می‌گیریم از node هایی که داخل این ساختار هستند بررسی داریم

* حافظه مورد نیاز به برابر با تعداد node هایی هست که در Frontier قرار دارند به گسترش آن با استفاده از الگوریتم‌ها (تکرار مایه)

* چک کردن رسیدن به هدف به وقتی node را از میان بررسی داریم می‌توانیم goal است یا نه را می‌بینیم بزرگ‌ترین می‌شود goal Test

Tree Search به شش search است که دنبال راه حل می‌گردیم، یک frontier داریم که ابتدای کار initial روش می‌ذاریم بعد expand می‌دهیم تا به فرزندان می‌رسیم، بعد فرزندان را در Frontier می‌ذاریم و به همین ترتیب...

همه * بدان که frontier حافظه‌ای نباشد (فانی باشد یعنی goal را پیدا نکردیم و search نتیجه نداشت) یک node از داخل آن بررسی می‌کنیم از بزرگ‌ترین است که هنوز گسترش پیدا نکرده است و چک می‌کنیم که goal هست یا نه اگر همون چیزی باشد که می‌خواستیم بهش می‌رسیم باشد به تمام! اگر نه فرزندان را به دست می‌آید (expand) داخل آن node

مشکل ۱: دیرین یک state، بارها بارها! به توقف state A، T، می‌بیند و T هم A را می‌بیند!

باست باف با بنای می‌شود! DFS = search باف با بنای می‌شود! DFS = search باف با بنای می‌شود! DFS = search

۵ اگر یک کنیم مسیرهای تکراری که به نقشه وجود دارد در نقشه از مسیرهای فنی زیادی دوباره به یک نقشه می‌رسیم که هزینه این کار!

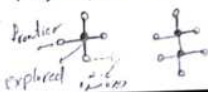


* اگر حالتی که بنای می‌شود می‌خواهیم مشکل را برای حالتی که متناهی هست حل کنیم:

۱ node هایی که گسترش داریم در mark کنیم: یعنی یک node را از frontier برداریم mark کنیم و دیگر در مجموعه explored بعد هر وقت تو frontier خواستیم node ای را بگیریم اگر تو مجموعه explored بود می‌توانیم تو frontier به جلوگیری از باف با بنای می‌شود و مسیر مقدر!

Graph Search می‌شود!

* یکم باقی نمی‌ماند به بعد از برداشتن node از frontier می‌توانیم چک کنیم که goal هست یا نه بهش می‌رسد، به گسترش به مجموعه explored اضافه می‌کنیم، به گسترش می‌دهد: فرزندان را به بهتری به frontier اضافه می‌کنند که قبلاً explored شده باشند! (مسیرهای مقدر)



Frontier به راه رسیدن به سایر node ها از این صفت می‌گذرد!

@sharifjozve96

هوش * node : هر node يك parent بايد داشته باشد كه مستقيماً كنند از node expand شدن ميبرود و اگر
وضيعة stale را بايد مستقيماً كنيم چون action سته روی اون انجام ميشه
اطلاعات اضافي : عتق اش ، با هم action اين node اوريدم
stale : تيممات ميكني

له چرا در داده ساختار نيست parent و act رو نمي داريم ؟
History هست ، چون سوال مسله ميرسيون از initial به goal هست

خوشگويي Search : * در صفت Frontier به بنيان بر داشتن ارزش مي باشد - نسبت به اين استراتژي برداشتن
از صفت براي الگوريتم هاي گيري خوشي ميورديار :

① Completeness : اگر جوابي وجود دارد حتماً الگوريتم حامي توانه آن را پيدا كنند .
له چه چيز هايي جلوي اين روش ميگيره : ① ماهيت مسله ② ماهيت Search : تعداد state ها
به نهايت باشد

① Time Complexity : روش سريج ما ميخواهيم بدانيم چه زمانه نياز دارد
له تعداد node ها تا قبل از رسيدن به جواب چندانست به تعداد node هاي نوعي زمان الگوريتم
را به مثال ميدهم

② Space Complexity : تعداد node هاي كه در طول الگوريتم نيمو داريمند داريم

③ optimality : بينه بران path اوليه به معني اوليه را از مسله پيدا كنيم كه هنريه ميبران حداقل باشد
مثال : start را كه گسترش ميدهم اگر اوليه به G برسيم به هنريه رسيده ولي هنريه
ميسريش 1 هست .
عوامل كردن Path Cost

مستقيماً كردن پيچيدگي زمان و حافظه - در درسا الگوريتم و ساختار داده با ساختار داده گيرن پيچيدگي را پيدا كنيم

* اين جا چون گران افاضل نمي سازيم در بحث Tree search ، تعداد node هاي كه ديده ميشه تا به اوليه ميرسيون نياز داريم .
b : فاكستور اشغال * تعداد node هاي توليد شدن تا عتق مورد نظر با استفاده از b و d
بدست مياد !
له تعداد act هاي قابل افعال

d : به كم عتق ترين اوليه به اوليه ها به اوليه ظاهر شده (چون تيممات ديده اوليه ديده شده)
m : حداقل عتق كه الگوريتم پيش رفته
مثال DFS

BFS : صف اوليه به تشكيل مي دهيم معيارش اين هست كه كسي كه اوليه دارد شيره ، زودتر خارج ميشه
تا به تقبيير ديگر هميشه كم عتق ترين node را گسترش مي دهيم

له بررسي خوشي ① Complete : افناي جستجو ميوردد باشد (طو d ميوردد باشد) به به ! (در صورت وجود داشتن جواب)
② Time : اين فاكستور اشغال با باشد
 $b + b^2 + b^3 + \dots + b^d = O(b^d)$

هوش { اداس Uniform-cost Search } * مسئله چیرا می‌تیم node ای که برای گسترش انتخاب (۱۱)

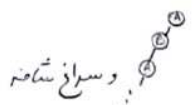
شده هم * مسیر بهینه (راه حل بهینه) تا اون node پیدا شده!

کجایه داریم؟ گفته ای که اولی را برای گسترش انتخاب می‌کنیم! به تابعی جواب ما optimal هست! یادون هست که گفته بودیم مسیر هر node ای از داخل Frontier را شب . مسیر هر node ای از گسترش node های که داخل Frontier بوده به وجود میاد . فزون کشید که n ، node ای است که قراره expand بشه یعنی $g(n)$ هزینه مسیری که به n وجود داشته از جمع گستر است هر ضایع بهینه که مسیر بهینه هم $g(n)$ است . پس فرض کنید مسیر یک node است . این مسیر باید از node ها داخل Frontier به وجود بیاد به فزون کشید n داریم و می‌تیم از n به n برسیم می‌دانیم که active ها متناهی هستند پس راهی که از n به n وجود دارد متناهی است . اگر همین چیزها وجود داشته باشد و $g(n)$ مقدار باشد باید عدد متناهی به $g(n)$ را به هم برابری معنی است که $g(n)$ کوچکتر از $g(n)$ است . پس این قراره بود چیزها استخراج می‌شده از صف اولویت n می‌بوده به بین انتخاب و انتخاب هم می‌شده!

* در هر لحظه عمیق ترین node رو گسترش میدی یا گسترش می‌دهی آخر از هم انتخاب شده

Last In First Out : استراتژی صف

که در این روش آخرین node انتخاب شده عمیق ترین node است *



Complete : با فرض متناهی بودن active ها ، Treesearch با شده به مارکس تونه نیاز به توقف به بنیاد ها که گسترش می‌دهیم \Leftarrow complete نیست!

نوع مشکل... که خوب حالا که graph بهینه مشکل حل می‌شده؟! این با هم explored ها رو تبه می‌داره ، مارا هتبه هم می‌تونیم جستجو رو به طرف نیم لحظه مانده داریم نوی مسیر تا این لحظه هم گمانه رو داریم ، به یک node جدید رسیدیم چه می‌تیم که می‌تیم به قبله بوده یا نه ، این داریم به جستجو می‌کنیم گسترش می‌کنیم \Leftarrow چون فقط node ها می‌تیم جستجو تعدادش متناهی کمتر از node ها می‌است که دیده شده تا الان پس نمایی هم نمایی شده (اشبات محبوبتر)

زمان (Time) : ضایع معمم نیست ولی (space) $O(b^m)$

که از اون جایی که DFS الگوریتمی است که در زمان جستجو می‌تونه عمیق از عمق m بهینه بشه $\Leftarrow m \leq d$

space : اگر تا عمق m بهینه رفته باشد ، در هر لحظه هم تا m node یونم داشته $\Leftarrow O(b^m)$

optimal : نیست! همچون مثال قبلی که گفته بودیم مسیر با طول بهینه داشته ولی ما از یک سمت بهینه با فزون بهینه بهینه رسیدیم (به عنوانی که اولی با شده قرارش می‌انیم)

Depth limited Search : تا عمق L جلو برو به وقتی به عمق L رسیدی گسترش می‌دهی همسایه ای تو تیرگی و به سراغ عمق دیگر می‌روی پس هیچ node ای در بیشتر از L مندا expand نمی‌شه!

complete : اگر عمق m بیشتر از L باشد به m فزون رسیده ولی این گسترش می‌شده به m پس اگر عمق m کمتر از L باشد هست!

space : $O(bL)$ چون عمق جستجو محدود شده به L

Time : $O(b^L)$ چون عمق محدود به L است

optimal : بهینه پرسش مشکل داره حتی همچون DFS

Iterative Deepening Search IDS

کاره که منتهی این است که ۵۵ رو با عقیق حسن را ببرد، آنس را در آب بپزد، عقیق را ،
عقیق دو ... به این رو با عقیق های مختلف این قدر را ببرد که تمام لعنه برسد (از اول لعنه
برآید)

optimal
شود

با این که داده نگار ممکنه یکسری کار رو ایا خدیه تو زمانش اشتباه نذاره و هم خصوصیات خوبی که می خواستیم برسیه
و حافظه را هم حفظ می دارد 0

Complete (دیکھو) DLS کے لیے، ازراہ بالا فریم، goal جو ہے یہی مستحقِ تکرار۔ Complete است!

time

دھر، پہلے node میں دیکھیں کہ g درجہ کی بات ہے یا نہیں، اگر ہاں ہے تو $d \times b + (d-1)b^r + \dots + 1 \times b^d = O(b^d)$

پہلے $(d-1)$ بار دیکھیں، پھر d بار

آخر میں d بار دیکھیں

بهینه‌ترین حالت ممکن را در زمان $O(bd)$ پیدا می‌کند. space
 در این حالت، d عمق درخت است و b شاخه‌های هر گره است.

۵. IDS و DLS - IDS = زبانی ہینے اے کہ طی و سیر پر اپنی کشتی
 * اگر عین امور پر اپنی DLS دیکھو!

* IDS مشکل حافظه BFS را حل کرد ؟
حالا مشکل UCS را چگونه حل کنیم که optimal 4 درج داشته استفاده کرد بران مسئله میانی که هزینه است. طریقی میسر نیست ؟ به چینی مثل IDS لازم داریم که روی هزینه ها اعتبار بیشتری مشکل داشته باشد که می توانیم سطح بندی انجام بدهیم و بعد در بین همواری و عیون داشته می دوند که بیش شیرین میگیریم بعد از اون ط برابر شد (در قبلی هزینه ها رو کش می آورد) چطوریه باسته Greedy اینها مثل Knapsack که بعد از اون که می بینیم اگر یک ضربه از کمره ها داریم و قبلی شد که code های با تا که دران بیشتر از اینها ها تو لیست می باشد به صورتی باشد به مشکل ؟! حالا سطح بندی کرد و هزینه هر یک بگیریم ولی !! سطح بندی نمی تواند جوریه باشد که روی زمان تاثیر نداشته باشد !
نذاره !

*** bidirectional Search**

از هر دو طرف شروع کنیم به جلو و عقب از در طرف شروع کنیم تا به ادون برسیم. یعنی start و backward

از خود ادون تشریح می کنیم، را تا به هم برسند

از هر دو سمت می رویم، جایی که ادون را می بینیم (یا داریم)

شکل مسئله ۱-puzzle

state - تک تک

شروع

از هر دو طرف به هم رسیدن

act ، backward است و هم درست باشد

[illegible]

* تا فردا لایه رویت
برای همین هست

* به عبارت دیگر باید هم دور که search رو میزنیم یک لایه عقبی باشیم، تا آخر صف check برافیم. مقصد تا آخر صف نیست
چون همه node های تارون لود رو بینیم بریم جلو مقصد تا یک تفاوت بتونیم یک صید کرد تا بریم به انت
اگر درت بر توانی هفتاد بریم جلو، مگه در یک باو ادا کنی دیگر با داریم به این شکل با خود داریم

bidirectional Search

۱۲ * دانسته باشیم که به بنایه یق $goal$ رو اگر نه داشته باشیم می توانیم backtrack شروع کنیم به حرکت کردن مثل مسئله act ما هم قابلیت برگشت پذیری داشته باشد (اینجا خاصیت مسئله هم بر می گرده)

۱ * چک کردن این که $node$ تولید شده در $frontier$ اون یکی هم هست یا نه باید $efficient$ باشه
یک $Hash$ خوب باید روی مسئله داشته باشیم که این چک رو تو زمان خوب و معقول انجام بدیم و نه چک کردن این که یک توصیف اون یکی هست یا نه می تونه زمان ببر باشه

Time $O(b^{d/2})$

۲ حافظه $O(b^{d/2})$ - مثل IDS داشته گذشت کرد؟! باید صحت از یک طرف BFS باشد و تریه الوریتم از سینی میوه تا نیاز به $frontier$ داره برای چک کردن که این دو تا به هم رسیدن یا نشن بین حافظه اش کمتر از $O(b^{d/2})$ نمی شه!

* در کجا ها از bidirectional استفاده کرد؟ جایی که $state\ space$ بزرگ هست IDS برای حفظ کردن حافظه لازم داریم و جایی که $goal$ در عمق کمتری پیدا می شه $state\ space$ صغیر بزرگ سینی و در حافظه میخوره $O(b^{d/2})$ در نتیجه bidirectional چون زمان کمتری داره روی پروسه یابی زمانه ما موثره است

Inform Search

جستجوی آگاهانه سعی میکنه جستجو را به سمتی که براش مناسب تر هست هدایت کنه!

Best first search

۱ $g(n)$ هزینه مسیر بهینه از حالت شروع به گره n

۲ می خواهیم به جای این که از نقطه شروع تا $goal$ اوپنیم دایره های به شعاع l ، $node$ هاش رو دیدیم
چون $g(n)$ که استفاده کنیم و مقیاس برداشتن از صفت است، چیزی استفاده کنیم که فاصله تا $goal$ را نیز لحاظ کنه
یعنی $f(n)$ می خواهیم داشته باشیم که ترکیبی از $g(n)$ و فاصله از هدف $h(n)$ باشد!
ولی فاصله ما نمی درشیم دقیقاً فاصله از هدف چقدر است!



* درست ما فاصله دقیق از هدف را نداریم: مثال ما جدولی داریم و می خواهیم به $goal$ برسیم
دکتری دیوار هم در سیر داریم. فاصله ما نمی دانیم که تعداد کشتن ها و لون بزرگ رسیدن به هدف چقدر است. فاصله $h(n)$ را نمی دانیم در این جا دقیقاً چیست و اما یک تخمینی از $h(n)$ را داریم.
✓ یک راه این هست که بدون در نظر گرفتن دیوارها فاصله مستقیم G را حساب کنیم. تخمینی به جای ده که چقدر تخمین پایینی است و می تونه که چقدر آکشن نیاز داریم

Best Fit

به $node$ ها نگاه می کنده هر کدوم که $f(n)$ کمتری دارد از صفت بر می داره $expand$ می کنه.

* در ساده ترین حالت، فاصله از نقطه شروع را در نظر نمی گیریم و می خواهیم یک تخمینی از رسیدن به هدف داشته باشیم که تابع h یعنی اگر در n $node$ h بیشترین هزینه ی کوتاه ترین مسیر برای رسیدن به هدف چقدر است؟ این رو وقتی میشناسیم

Heuristic Func

که $h(n) > h(n)$ برای $node$ و $h(n)$ برای $goal$ صفر باشد

فرض کنید در پیدا کردن مسیر در نقشه، نقشه را به صورت فیزیکی داریم و می خواهیم $search$ بر این بار طوری انجام بدیم که از تابع $h(n)$ استفاده کنیم و $h(n)$ فاصله شهر n تا شهر هدف است (مثلاً فاصله اقلیدسی دو شهر با خط کش) - تخمین پایینی از رسیدن به هدف به واسطه ده

هوش 5

Best First Search

①

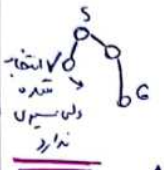
greedy BFS

تابع $h(n) \approx h_m$

15) یعنی node انتخاب شود که هزینه کمترین رسیدن از آن node به هدف از همه کمتر باشد

به نسبت همون UCS بود
 $P(n) = g(n)$

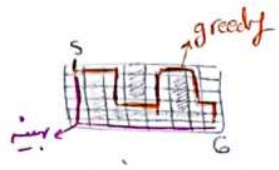
* له چون فقط به $h(n)$ نگاه میکنیم و هزینه رسیدن از مابقی که هستیم تا نره n لحاظ نمیکنیم پس بهترین نیست (optimal) نیست



* این در سیرش جلوی loop نمیگیره یعنی به جواب نرسه!
* برای جلوگیری از loop به تمام node ها طول سیرش رو نگاه داریم که اگر تکرار شد بفریم (برای tree search چون تکرار search هزینه)

complete

خبره به شبیه DFS هست. این graph بود چرا کامل بود (استه تو state های کم و محدود) یا جلوی یه کار رفتن loop ها و بینهایت با ترفندهای دیگه که این میشه!



Time

به اکثر step پیش بره $O(b^m)$

چون داره فقط به فاصله $h(n)$ نگاه میکنه داره مسیر پیچیده رو میاره!
* به ظاهر مسیر پیچیده حتی ممکنه $d \leq m$ باشه!

space

چون نیاز داریم که صف اولویت رو بشینیم، لازم است node ها را نگه داریم و مثلاً به DFS نمی توان آنرا حل نمود! $O(b^m)$ چون $h(n)$ هر کجای که بخواهیم از بینهایت کمتر باشه مجبوریم نگه داریم همه را در صف

A*

هزینه 4
اینه: می خوام علاوه بر تخمین فاصله که $h(n)$ نره هدف، فاصله جایی که هستیم تا نره n هم در نظر بگیریم $g(n)$
 $f(n) = g(n) + h(n)$

* می خوامیم ادعا کنیم که بعد از یافتن مسیر رسیدن به هدف، این مسیر بهترین است!
له پس لازم هست روی $h(n)$ خولصی قرار بدهیم تا اشیان کنیم!

① admissibility: ثابت قبول بودن $h(n)$ تخمین از فاصله ازن نقطه تا هدف که خوشبینانه باشه!
گفته از هزینه واقعی تا هدف باشه!

$h(n) \leq h^*(n)$

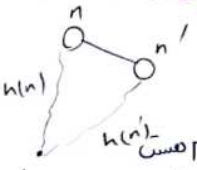
اگر $h^*(n)$ هزینه ی بهترین تا هدف است! \leq یعنی هزینه overestimates نمیکنه به شل فاصله احتمالی رو بیشتر شبیه به جاره ها

هر دو تای این شرط ها لازم نیست پس بولد هست شرایط ① و ②

Consistent بودن (monotonicity) consistency

$h(n) \leq c(n, a, n') + h(n')$

این نره n را داریم یک نره بعد از n که اسفند n است و از n میرویم به n' بهریم بهر شل ناسازی شل است پس این در همه جا باید act بشه به n' بهریم و ناسازی بهر تر باشه پس n consistent است!



* شرط ⑤ قوی تر از ① است یعنی اگر ⑤ برقرار باشه صافاً ① هم هست $h(n)$

اشیان Consistent بودن پس admissible هم هست! می باید بشنود به $h(n)$ هزینه اش کمتر از c ها روی مسیر بهترین است
اشیان شده $\rightarrow h(n_1) \leq c(n_1, a_1, n_2) + h(n_2) \leq c(n_1, a_1, n_2) + c(n_2, a_2, n_3) + h(n_3) \leq \dots \leq \sum c(n_i, a_i, n_{i+1}) + h(n_k)$

@sharifjozve96

ادامه بحث جستجوی آگاهانه - از هوش اول نگاه به ادوم داریم - بالاستاده از $h(n)$ که یک تخمین از فاصله است - یک تیره تا آخر بود!

* برای اثبات بهینه بودن نیاز داریم $h(n)$ باید تخمین خوشبینانه باشد و قطعی آن از هزینه ای (معمولاً کمتر باشد) (کمتر از فاصله باشد)
* در UCS $f(n) = g(n)$ هزینه مسیر رسیدن از ریشه شروع به n - یک نوع از Best First Search

Tree Search A*

* می خواهیم با $admissible$ بودن نشان بدهیم که (اثبات کنیم که) نسخه A^* بهینه است و ثابت که $h(n)$ تا بلی مقبول باشد!



فرض کنیم هدف است که بر اساس $P(n) = g(n) + h(n)$ انتخاب شده! مسیر از S به G مسیر بهینه باشد
مراحلی که به شاقق برسیم.
 $f(G_r) = g(G_r)$ $f(G) = g(G)$

فرض کنیم G ، suboptimal باشد $\Rightarrow g(G_r) > g(G)$ $f(G_r) > f(G)$

حقاً در Frontier، زانکه G انتخاب شده، گره G وجود داشته که از آن گره n داریم به G می رسیم (باید از راه به G برسیم که از طریق Frontier باشد) - با فرض قبلی مقبول بودن -

$$f(n) = g(n) + h(n) \leq g(n) + h^*(n)$$

اگر $f(n)$ روی مسیر بهینه G باشد، پس $f(G) = g(n) + h^*(n)$ چون تخمین روی مسیر بهینه است n ، و $g(n)$ که هزینه بهینه تا گره n است را از این به بعد هم با $h^*(n)$ هم که بهینه است پس می شود همان طول مسیر بهینه رسیدن به هدف داشتن می دهیم

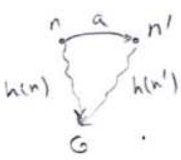
✓ شاقق رسیدیم! چرا که $f(G_r)$ در زمان انتخاب باید از گره n کمتر بوده باشد که انتخاب $f(G_r) > g(n) + h^*(n) \geq \frac{g(n) + h(n)}{f(n)}$ شاقق رلی این چیزها که به دست آوردیم یکسان می باشد

له این اثبات شد یعنی که $admissible$ بودن برای $expand$ شدن انتخاب می شود مسیر ما بهینه است! رلی برای $node$ ها میانی اثبات شده (یعنی برای هر گره اثبات شده) در A^* Tree Search - بهینه $admissible$ تا بهینه است

*** A* graph Search + heuristic Func Consistent**

به ازای هر گره n که $successor$ گره n است (منظور از $successor$ این است که از طریق یک $action$ از گره n به دست می آید) داریم که این همان $consistent$ بودن بود

$$h(n) \leq h(n') + c(n, a, n')$$



می خواهیم اثبات کنیم که هر $node$ ای که از من برداشته می شود، مسیر بهینه بهینه به دست آمده - پس $g(n)$ است که تا این در $f(n) = g(n) + h(n)$ بوده نیز بهینه است

* کم ای که می خواهیم نشان بدهیم این هست که $f(n)$ همیشه در طول مسیر دایره افزایش پیدا می کند - به خاطر $consistent$ بودن $f(n') \geq f(n)$ - چرا این طور می باشد؟



$$h(n) = 9 \leq 4 + 1$$

* تا بهینه که $admissible$ هست، $consistent$ نیست!

$$f(n') = g(n') + h(n'), \quad g(n') = g(n) + c(n, a, n')$$

طبق تعریف $h(n) \leq h(n') + c(n, a, n')$ پس داریم $f(n') \geq g(n) + h(n)$ $f(n') \geq f(n)$ ✓ - اثبات شد که مقدار f در طول مسیر کم نمی شود چون دایره به g مقدار هزینه اول یا از آن هزینه شد و مقدار h از h دار کم می شد کمتر از مقدار هزینه اول یا از آن هست.

* پس هر $node$ ای که از من استقراف شده، هزینه بهینه رسیدن به $node$ برایش به دست آمده! (دست می آید به صورت $graph$ استقراف) که به یک بار $expand$ شده و به $expand$ نمی شود! - (؟)

مثال جایی در بات :

خانه شروع : S ، خانه هدف : G ، action : جابجایی و ... (مانند ۸)
 state : خانه فعلی ، pathcost : $act + pathcost$ (تعداد گام ها)
 * این pathcost با فاصله آنتی متریک باشد یک فرق دارد ؟



۱* چه فاصله را Manhattan : $|x_1 - y_1| + |x_2 - y_2| + \dots$ در نظر بگیریم می توانه مقدار فاصله رسیدن به هدف را $overestimate$ یعنی بیشتر از فاصله بهینه بشود ؟ (در مثال قبل این نوع تابع heuristic قابل قبول نیست ؟) - فیه
 چون در ۸ هفت حرکت می کنه

۲* می خوام $h(n)$ را $admissible$ انتخاب کنم که $h < h^*$ که این سقف باعث می شه node ها کی را هر کی کنیم چون صافه هارون با عدد کوچکتری هج می شن و هر کی صافه $h(n)$ بزرگتر هم هر کی بیشتر = با این مورد آنتی متریک $admissible$ بودن در نظر بگیریم آشتی تر هر کی شدن ، چه در از دست می دهیم ؟ بهشتی در از دست می دهیم .
 (۳*) این صریحی از h^* با شمر به هدف هر کی

۳* در UCS ، h تا انتقال از یک شروع $admissible$ می شه ، در $greedy$ نزدیک ترین فاصله به هدف $admissible$ می شه : در A^* با این نسبت $greedy$ فاصله $admissible$ کرد و بهینه است .

مشکلات A^*

فضا (space) در A^* فزاینده می تونه = شن UCS ، BFS
 که نوع مشکلات این چنین نوع A^* دیک ساختن : SMA^* ، MA^* ، $RBFS$ ، IDA^*

۱ پیچیدگی زمانی = برای بهینه کردنش برای $h(n)$ صافه می ریزیم که $admissible$ نیست و بزرگتر از h^* هسته و به جای بهینه میار $suboptimal$ صافه میار می کنه که می شه توزین بهتر میار بشه ؟

ترکیبی Heuristic Func

۳۲ $d=22$ ، $b=3$: Tree Search : $\frac{4!}{1} = 24$: graph
 که در مسئله ۸ پازل : $d=22$ ، $b=3$: کمتر شد نسبت به قبل ولی باز هم زیاده هنوز

۱* پیشنهاد $h(n)$ برای سریع تر شدن : علاوه بر h چیزی دره (آنتی متریک) Cost یا act صافه هسته ما این آنتی متریک
 ① $h(n)$ ۲ مقدار اعدادی که سر جای خود نیستند بشماریم = حقیقی یا اینها است برای رساندن ما به هدف = چرا حقیقی یا اینها است ؟
 برای این که هدفمان blank جابجایی شود ، برای این که به هدف برسیم باید هر فونیک بار جابجایی (برای این صافه که سر جای خود نیستند هارون یک بار نیاز داریم جابجایی ؟)

② $h(n)$ ۲ فاصله هر کی که باید قرار بگیرد صافه کنیم و توهم جمع بزنیم . معیار فاصله Manhattan است .

۳* راهی برای مقایسه الگوریتم ها با همانه مثل A^* با غیره با همانه شن IDS
 تعداد node ها که در آنتی متریک A^* کمتر شنیدیم می کنه به از این چه b^* با عبارت بالا میاریم ؟
 $N+1 = 1 + b^* + (b^*)^2 + \dots + (b^*)^d$
 که حقیقی b^* هست هم داریم = اون b^* را بهشت (Effective branching Factor) →

هوش ۴ هرچی $h(n)$ را در A^* بهتر انتخاب کنیم * تا به $\frac{1}{2}$ نزدیکتر می‌شود! $\epsilon = (1+\epsilon)^d$ عملاً نمی‌توانست ولی یک نمای ضعیف بود. (۱۶)

Heuristic quality آبی به ازای هر n $h_1(n) \geq h_2(n)$ باشد (هر دو ϵ admissible) برای ما بهتر که از $h_1(n)$ استفاده کنیم.
 که h_1 و h_2 را h_1 h_2 $dominate$ می‌کند
 که گاهی اوقات باین که $h_1(n)$ را در درج اول استفاده نموده چون محاسبات سفیدی دارد!
 که اگر هزینه محاسبه h زیاد باشد حتی اگر خود h^* هم بیش به دردمون نمی‌خورد چون محاسبه سفیدی دارد!

Consistency (max خاصیت) * $h(n) \geq \max(h_1(n), h_2(n))$ اگر h h_1 داشته باشیم که هم دیگر را $dominate$ نکند،
 (یا حداقل نه)

جلسه ۷ * بحث هر کس کردن سری پیش و تخمین پائین بودن $h(n)$ به ما می‌تواند $expand$ شدن $node$ های که هر کس شدن ما اطلاعاتی برآز دست‌نمی‌دهیم.

$$g(n) + h^*(n) \geq g(n) + h(n) > C^*$$

روش‌های اوج‌نمایک به‌ت‌آوردن $h(n)$ * تا آخر ترم باهاش کار داریم!
 ۱- اولین راه حل: مسئله اصلی $relax$ یا ساده‌تر کنیم و جواب مسئله ساده شده را به عنوان تخمین برای جواب اصلی در نظر بگیریم.

* در مسئله $relax$ شده ما سعی می‌کنیم در مسئله اصلی که $initial$ و $goal$ داریم و یک گره n در این وسط ظاهر شده از گره n هزینه سیر بهینه رسیدن $goal$ چقدر است. چون این را نداریم ریشتم سرانجام تخمین پائین از این شروع.
 * $relax$ کردن مسئله یعنی چیدن $node$ و $edge$ جدید به مسئله اضافه کنیم به با اضافه کردن $node$ و $edge$
 سیر بهینه می‌تونه همون سیر قبلی بماند و یا هزینه‌اش کمتر بشه، پس این اضافه کردن $node$ ، $edge$ هزینه می‌را افزایش می‌دهد.
 حالا ما در گرهانی که کیری $node$ و $edge$ اضافه دارد داریم دنبال سیر بهینه می‌گردیم که می‌تونه سریع‌تر حتماً ما رو به جواب برسوند.

* مثال ۸- پازل: در این دنبال این بودیم که تخمین به ازای هر گره چقدر مان در صفحه تخمین بزنیم چقدر مان به حرکتی ز داریم، $goal$ سیر دقیق‌تر داریم. حداقل تخمین این تعداد حرکتی نیاز داریم.

* $h(n)$ = تخمین پائین تعداد حرکتی لازم برای رسیدن به میدان هدف

* act : جابجایی یک خانه غیر $blank$ با $blank$

* برای $relax$ کردن کیری شرط را بزرگ داریم ۱- در مسئله $relax$ شده تخمین برای رفتن از خانه A به خانه B لازم نیست حتماً $blank$ باشد به بتویم $move$ کنیم به هر خونهای (خانه‌های مجاور)
 فاصله $manhattan$ دو خانه $h(n)$
 جابجایی در اول با فاصله $manhattan$ تا رسیدن به هدف

در زمان پیدا کردن act $manhattan$ فاصله هر خانه که هر حرکتی می‌کنیم به خانه $blank$ برویم و لازم نیست خانه $blank$ جابجایی باشد
 ۵- $force$ تخمین که معتد حتماً مجاور خانه‌ای فعلی باشد!
 این مسئله حل می‌شود

* می‌شود یا یکی از ۵ و ۶ را هر دو را برداشت از برای مسئله
 ۷- سیر هزینه‌های مسیری باشد تعداد خانه‌هایی که سیر جانشون نیستند!
 ۸- در هر A می‌توانیم از برای $manhattan$ فاصله $h(n)$ که قبلاً محاسبه کردیم
 نسبت داریم (نسبت به ۲ ندارد) A دارد می‌تواند روی B حتماً توش هزینه باشد

چون برای هم قرار می‌تونه بگیرد، که تعداد زیادی $node$ دارد به مسئله اضافه می‌کنند این $node$ ، $edge$ اضافه شده به صورت فله‌ای و کمک کننده به مسئله است و حلاً روی اون ها $search$ صورت نمی‌گیرد بین هزینه سیر را اضافه نمی‌کنند

۵) راه درم برای پرسش آوردن $h(n)$: (برای بعضی از مسئله قابل اعمال است) - heuristic database pattern

مسئله act خاص (reversible هست) - جابجایی blank و عدد می تواند به یک بخش هم رخ دهد!
پس از goal به صورت backward شرح می کنیم به حرکت کردن از goal به گجها می شه رسید! BFS بهترین
و همی ادن هائی که باید act داشته باشه رسید را بدست آوریم و همین طور ادامه دهیم و جواب تک گد را به دست می آوریم
(تک گد pathها) و ذخیره می کنیم تا مسئله بعدی را با آن حل کنیم! (ذخیره در حالت)

* جنبه‌های حل مسئله می‌تواند از A^* استفاده کنیم و به $h(n)$ نیاز داریم. حال $h(n)$ را چه در نظر بگیریم؟! مثلاً $h(n)$ این باشد که $h(n)$ همان تعداد حرکت‌ها که نیاز داریم این n را به حالت مقصد برسانیم، $h(n)$ باشد بر این مسئله حل

* حالا فرض کنید در approx در نظر بگیریم (ϵ, δ) به ازای حالت اول یک h ، و حالت دوم h ، یک در آوریم پس برای حل مسئله Max این هارا در نظر بگیریم به دلی $!!$ جواب h به ما نمی دهد 0 چون عددش به اندازه δ کافی بزرگ نیست 0

میں نے اس سے مسئلہ کلی Sum - ولی چیرا Sum خوب؟! و تہ ما داشتیم حرکتی می دادیم قیمت مقدار عملیات بعضی خوبجا
 برای این که Δt از وضعیت شروع برسان به هدف یعنی لازم بود تاها را ستره ها را حرکت دهیم تا سایه خانه ها را به هدف
 برسانیم پس در جا به جا شدن Δt داریم جا به جا شدن Δt هم می شماریم . ممکنه بعضی از این جا به جا جایی ها همون طایفه باشد
 که Δt ۸ هم به مقصد می رساند ولی این مقدار باعث ممکنه بشود از h^* بیشتر بشه مقدار
Sum به خوبی کار کنیم؟!

① راه دورترین ۱-۲ به فاصله حرکت شده ها را نشان بدهم. ۰ ها را دایره ۱-۲ شماره نشود. شماره نشود دایره ۱-۲ شماره نشود. (راه دور)

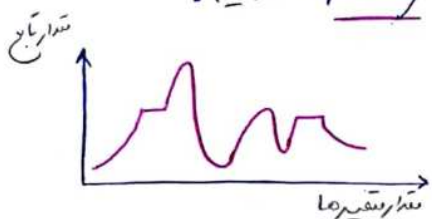
* در مسئله ۱۵- پازل (که ۱۲۴ غریبه هستند - با ۱۹۹) - به ازای \max میری h_1 ، h_2 مثبت به خود h_2' زبان ۶ $\frac{1}{1000}$ می شود. $\frac{1}{1000}$ نوی Sum پازل زبان بهتر می شه، می شه $\frac{1}{1000}$

[illegible]

هشتم * توی مسئله ۸ - پازل، ما شده زیر مساله هایی با این هم حل باشند (۱ تا ۵، ۸ تا ۱۵، در نظر گرفتن ستاره ها) ۱۱
ولی در مسئله RoboCube هر حرکت آن چیزی رو داره با هم جابجا کنه و این محدودیت رو برش ایجاد کرد!

مسائلی که وضعیت شروعی وجود ندارد و act هم به صورت تعریف هایی که تا الان دیدیم وجود ندارد که ما دنبال یک دنباله از act ها باشیم که ما رو از وضعیت شروع به هدف برسونه! به تنهایی این که به عنوان مطرح نیست، پیدا کردن وضعیت هدف است! (شماره ۸ زیر) $L + SP$ علاوه بر شماره ها به چینه که طول به دست آمده حل شدن باشد) یا از این جنبه دیگر کار داریم و بعضی از کارها پیش نیاز کارهای دیگه است! به عنوان مثال یک چطور مانع ما از چاره شدن (در این مسائل، توصیف مسئله رو داریم ولی وضعیت هدف نداریم! به نفعی داریم که چه میخوانی منطبق به این توصیف می شه!)

Local Search ما خواص استفاده از روش های greedy همراه با روش های دیگر به صورت مطلوب استفاده کنیم و در زمان خیلی خوب مسئله را حل کنیم. مسئله را حل کنیم به زمان خوب به دست میاریم ولی لزوماً به optimality نمی رسیم!



* این تابع به سبب تغییرش، تغییرات نرم داشته باشد، یک روش برای پیدا کردن بهینه های این تابع این هست که از یک نقطه در فضای محلی شروع کنیم و در هر نقطه از این همسایه ها انتخاب نقطه، نقطه ای رو انتخاب کنیم که مقدار P در آن بیشتر است! اگر همین کار را با نقطه بعدی انجام دهیم و به یک بهینه می رسیم ولی این بهینه، لزوماً بهینه سراسری نیست یک بهینه محلی است!

۱۰ * ما به دنبال این هستیم که از این ایده برای حل مسئله ها چون استفاده کنیم یعنی با پیدا کردن بهینه های این تابع به جواب مسئله برسیم ولی چه میزهایی را State ... در نظر بگیریم؟! پس محور افقی قاعده تا باید state ها ما باشند (شبه state ها را از چند به هم در نظر گرفت) در مسئله ۸ - وزیر
به دنبال max کردن مقدار زنجیر وزیر هایی هستیم که همسایه های آنها کمتر است (محور عمودی ما) یعنی با فواید state ما را پیدا کنیم که مقدار زنجیر وزیر هایی که همسایه های آنها کمتر است باشد به سبب بهینه های تابع را پیدا کنیم!

تعریف ارتباط بین state ها : در مجموعه ای x_1, x_2, \dots, x_n که همسایه های آنها کمتر است (محور عمودی ما) x_1, x_2, \dots, x_n که همسایه های آنها کمتر است! پس باید مفهوم تفاوتی مطرح کنیم تا بتوانیم بهینه های state ها را پیدا کنیم تا تابع ما داشته باشد!

۱۱ * هم شرطی باید به قرار باشد که کار هم باشد state ها : در مسئله ۸ وزیر، state = [] به این معنی که شماره سطر ها که وزیر ستون نام در آن تکرار دارد! (به این معنی که state ها را می توانیم با هم مقایسه کنیم) این حالتی که چیزی را در یک ستون قرار داده و اینکه شدن به این حالتی که در هر ستون فقط یک وزیر قرار می دهیم.

حالا قرار بزنیم این state ها کار هم باید معنی داشته باشد (در این از یک state به خانه همسایه باید با معنی باشد) * اما قرار بود فقط به وزیرین و کار هم قرار دهیم تا بهینه می شد که میزنیم و بهینه رو به سبب ما! (این P ها به هم مرتبط خوب این state = [] باشد به این معنی داشتن همسایه، به این معنی که مثلاً فقط یکی از این وزیرها با هم همسایه اند تا locality حفظ بماند و state مقدار P نشان به هم مرتبط باشد!

۱۲ * ما به state to به objective میزنیم که نقطه $Min \sim Max$ داره همون goal را نشان میده.

* Cost Function = \min کردن هزینه هست. پس تابع ما می شود مقدار زیر \min به عنوان هزینه در نظر می گیریم.

حالت state را می‌توان به دو روش تعریف کرد، هم به صورت یک مقدار (value) و هم به صورت یک تابع (function).
 اگر state را به صورت یک مقدار تعریف کنیم، آن را state value می‌نامیم. اگر state را به صورت یک تابع تعریف کنیم، آن را state function می‌نامیم.

hill climbing
Search

↓
۱۴٪ مواقع
مشکل حل نشد

* اگر بهترین همسایه value اش از state منفی بهتر نبود، یعنی بازگشت یا در shoulder یاد
→ به آن سمت که کوچکتر یا مساوی شود میاریم چون رایج دفعه دوم منتهی نمیشود عنوان جواب که بینهم نیست!

فلات *

راه حل :

در این توجههایی که مسأله هست اجازه دهیم که بره جلو ... به خطی مقدار shoulderها توسط خطی
زیاده است ، ما جای یک دور در ۸- دوبر را عوض کنیم شاید در h تغییراتی ایجاد کنیم ولی اگر یک دورنه داریم
چاش رو عوض کنیم (۱ step مثلا) بتونه مقدارش رو عوض کنه

* یک مشتق داریم وقتی هست که نقطه هرگاه از این طرف \max و از طرف دیگر \min باشد، نشانه از این گذشته به این به معنی اصلی هر

sideways moves $\odot \rightarrow$

این در مسئله ۸- زیر ϵ یک \max حرکتی (مثلاً ۰.۱ حرکتی) اگر باز هم مقدار h مادی شده که بدون اداس دهیم

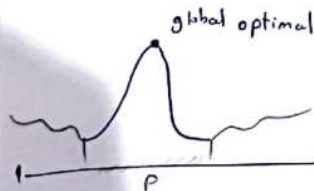
* این در وزن که یک تغییر کوچک بود و نیست موانع حل مسئله در زمینه افزایش به h ! (همه مسائل این سطح مسئله در h به مسائل از این دست جواب داده!) چون تعداد h ها زیاد است و در وقت پراکنده اند. از هر نقطه شروع کنید با h پنهان کردن می توانیم به اصل

۱. چکار کنیم که اون ۴۰٪ هم Pail بشه؟! (پایین اینارو مایه دراریم) - پاهون آتوریم و پاهون Successor

random restarting

به طبع شروع از یک نقطه، اگرچه goal فرسیده، دوباره یک نقطه شروع می‌گیریم، آن قدر حرکت می‌کنیم تا با نقطه $stale$ پیدا کنیم. هم بهینه‌تر و هم h آن برابر صفر است. به این معنی که بهینه‌ترین سراسری هم هست.

* عیسٰی مزید شور و دھواں دھماکا مچا کر :



درصد از نقاط در دامنه‌ی قبلی تکرار دارند یعنی اگر نقطه شروع را n بگیریم و دامنه را P بگیریم به نقطه می رسیدیم. یعنی در P درصد از state اگر از این نقطه شروع کنیم $\frac{1}{P}$ به global opt می رسیدیم. امید ریاضی که قراره random rest بزنیم که به احتمال $\frac{1}{P}$ به جواب می رسیدیم به این برابری $\frac{1}{P}$ است یعنی به طور میانگین به ازای $\frac{1}{P}$ مقدار دامنه قبلی قرار می گیریم و به global opt می رسیدیم.

هوش
* تعیین تعداد step مورد نیاز با آسوریت $\frac{1}{p}$: تعداد کل restart $\frac{1}{p}$ و $(1-\frac{1}{p})$ تعداد fail n_p : تعداد step تا به رسیدن به مقصد
 $(1-\frac{1}{p}) \times n_p + n_p$

* تعیین در حالت ساده در ۱/۲ مواقع به جواب می رسیم به این معنی همون ۲ به نسبت داشتن مقصد در کل که با global opt می رسونه در state
 ۲ در هر مواقع اشتباه می دهه $\frac{1}{2} \geq p$ یعنی با ۷ بار اجرای این آسوریت می توانیم با اطمینان ۱ به جوابیم که global opt پیدا کنیم

* اگر p ما عدد نسبتاً بزرگ و خوبی باشد، ما را به جواب می رساند! وگرنه باید تعداد restart ها را معنی ضعیف زد کنیم.

له مشق stochastic همیشه به بهترین همسایه اشاره! به همسایه ها رو نگاه می کنه بسته به این که مقدار از state فعلی بهتر هسته یا به احتمالاً به اون تفاوت صافه! همسایه های که ضعیف ترن با احتمال بیشتری میرو و همسایه های که یکم بهترن با احتمال یاسن تر.
 این کار (stochastic) ممکنه ما را به بهینه سراسری برساند. کل greedy greedy چرکن کردن ما را به local opt می رساند هر چقدری که یکم randomness وارد کنی به اپت رسیدن به global opt راحت تر می کنی. (راه دیگر randomness)

(بسته restart هم کن این رو هم)
 با step کم در فضا مواقع هدف می رسیم
 complete به هدف می رسیم. اگر از یک نقطه به صورت رندم
 همسایه ها شش به شش به هدف می رسیم ولی تعداد \uparrow step
 random + greedy
 Simulated annealing

random walk به منظور این هست که هر کدوم از همسایه ها رو با یک احتمال مساوی (در حالت ساده) انتخاب کنی از همسایه ها بچرخ.

* فرض کن می خای تا جایی که داریم می چرخیم Min پیدا کنیم، فکر کن که یک سطحی داریم به ازای نودی هست و یک توپ هم انداختیم تو این سطح.
 اگر ما توپ را از این جایی در سطح رو بچرخیم میرو در اولین نودی که بهش می رسیم می فانه. hill climb
 ولی در این آسوریت در شروع، شروع می کنی سطحی که توپ در آن قرار دارد را نگاه می دهی به هرچی در زمان بیشتر می رود، شان ما کمتر می دهی شروع با randomness زیاد و با پیشرفت آسوریت randomness را کم می کنی و greedy را زیاد.

* هر نقطه برای هر state، successor ها ش رو در نظر می گیریم، یکی از همسایه ها را رندم انتخاب می کنی اگر آن همسایه بسته به state فعلی بود، صفاً به آن می رود (با احتمال $\frac{1}{n}$). می توانی به بهترین state فعلی هم بروی اما با یک احتمالی به چپ می افتی
 انتخاب کنیم؟! به هرچی بهتر احتمال هم کمتر.

* T برای این بوده که در طول زمان randomness کم شه! یک کمینه می دارم مفهوم دما
 هرچی در طول زمان بیشتر پیش می ریم $\frac{1}{T}$ به می توان معنی بود هرچی T می شه دما احتمال می کنی به دما به سمت صفر می ره
 $T(t) \propto \frac{1}{t}$

جلسه ۸ مثال: TSP
 Hill-climbing به فرض کن که L_n شهر داریم می خایم دنباله ای از L_n شهر پیدا کنیم که هر L_n شهر را می پرشاند و هر شهر، یک بار ظاهر شه و طول دنباله ای که قرار است از شهر یکم تا شهر n ام همه رو به هم را از n ام به بریم به اول، مجموع این طول حداقل بشود. (حتی فرض کنی طول کامل است) به پت پتال کمینه کردن مجموع یال های بین شهرها هستیم!

۱. هوس : اداس سال TSP : Successor → state ها جایگشت های مختلف این شهرها هست (شهر آخر هم وصل هست به شهر اول) ← جایجایی در آن شهرها (نژاداً دو شهر که جایجایی نمی کنند)

همه نسبتی
 objective Func
 عدد چیزی که می خوام حداقل یا حداکثر کنم ← (جمع هزینه های جاری + جمع اولی را فکری) ← می خوام Min کنم

یعنی از این حالت TSP گفته می تونه فیند سریع ترین مسیر به بهترین (تو می بین)

$$P(S \rightarrow S', t) = \alpha \times \frac{(E(S) - E(S')) / T(t)}{e^{(E(S') - E(S)) / T(t)}}$$

if $E(S') > E(S)$
 o.w.

اداسی Simulated Annealing

Successor $S \rightarrow S'$

* برای حالت Max کرن E هست

* α → برای نرمالیز کردن احتمال ها بین صفر و یک و این که مجموع آن ها یک بشود

* تنظیم احتمال رفتن به state هایی که خوب نیستند $\frac{E(S) - E(S')}{T(t)}$ (هرچی ΔE مثبت تر بشه احتمالش کمتر هست)
 بین ما به دنبال این هستیم که به خانه هایی برسیم که خیلی بهر هست دنبال یک بهتر ها هستیم

* $T(t)$ دما هست که بر اساس زمان تغییر می شود

که درست داریم این دما در شروع زیاد و در طول زمان آن را کم کنیم ← باید تابع تدریجی به حسب t باید باشد ← اثر این تابع تدریجی این هست که رفته رفته تعداد تغییرات می کنیم یعنی در حرکت های اولیه به حرکت هایی که روی بهر شدن هستند احتمال شون در بالا می بریم

ایده

این که ما می خوام به بهترین سراسری را در الگوریتم پیدا کنیم چطور باید state شروع کنیم ؟! یک مجموعه state داشته باشیم و اون ها رو وضعیت شاد رو عوض کنیم و هر لحظه مناسب کنیم که بهترین می هست ؟
population

Local beam Search

* استفاده از ایده population ← به جای یک state L_k state داره ، برای هر کدام از L_k Successor ها رو پیدا می کنه و همه ی اون ها رو روی هم می ریزه ؟ و یک population تشکیل میده و تو این شاه می کنه که L_k بهترین کدوم ها هست ؟ و این ها هست L_k population گفته میشه ؟ و همین طور ادامه میده ؟ (واقع است که یک state اولی بهریم یا تویم ؟)
 (مثلاً در فریزر هادی که هم در آفریزر می کشه و هم در فریزر)

مشکل چه ؟! ممکن است این L_k به سمت خاصی از مقادیر L_k تنوع دین جاها که مختلفه فضا را می پیمایند. (مسایله ای که دشوار به یک سمت فضا کشیده میشه که ما این در فضا می خوام)

فوق کردنش : randomness لغت نامه کنیم و همیشه L_k بهترین رو انتخاب کنیم ؟

ورژن Stochastic

در جمعیت میانه ایما د شده به جای L_k بهترین رو انتخاب کنه ، هر کدام از این جمعیت را می توان به یک احتمال انتخاب کنه که احتمال انتخابش و اسبته است به h براساس ، هرچی h تابع $h \uparrow$ (مجموعه اون رو MAX کنیم) ، احتمال انتخابش بیشتره ؟ نژاداً L_k بزرگترین بهر شون بعد منتقل نمی شود .
 h ← چیزی بود که دنبال بهر است آوردن بهریم است بودن (در مسئله ۸ وزیر تعداد وزیر هایی بود که هر کدوم رو بهریم می کنه که می خواستیم MAX کنیم اون رو)


چون این L_k ها جدا بود دیکه با هم ارتباطی نداشتن این L_k و اون اثرش بود که با هم بود شون داره (معول میباشون هم هست دیکه) این هست که داره همسایه ها به دیکه سمت خوب کشیده می شوند و شون وصل به یک رو بهترین می کنه ؟

Genetic Algorithm

این جا هم state معادلی که موجود است (برای برقراری ارتباط با ژنتیک که در طبیعت وجود داره)
 ← objective Func : معادل با این است که این موجود هیتر اید به بقا تو لید و مثل داره هرچی این بیشتر باشه حرواقه مقدارش بیشتر است ← دنبال حداکثر کردنش هستیم .
 ← Successor ها : فرزندان این موجود ؟ (معادل همسایه ها را سوریم های دیکه هست)

Genetic Algorithm

State به صورت یک رشته مشخص می کنیم که به آن کروموزوم می گوییم
در هر عقرون ژن می نویسیم

له مثل قبل - در ۸- وزیر  - وزیر: این خون به واسطه وزیر ستون: اسم و ژن سیر - این ۸ تایی
پشت سر هم دارم کروموزوم ما و ژن سیر - به هر خون این کروموزوم هم ژن می گوییم.

Fitness Function - مقدارش نشان می دهد که این state چقدر state خوبی هست - تا جایی که دنبال بهترین باشیم

له ژنوتیپ های که تا الان معرفی کردیم، operator ها به صورت successor ها بودند
Combination - (معمولاً جبر) می خواند در آن موجود را با هم ترکیب کند و از آن offspring یا فرزند تولید کند - به این عمل
Cross-over می گویند

له عمل mutation و (یعنی به جای آن successor می که قبلاً داشتیم در آن operator، cross-over و mutation دارد)
چون تغییرات جزئی
هستند یعنی یکم دست می آید
local هستند
مثلاً یک مقدار state فعلی را عوض کن مثلاً یک خون در ژن
عوض کن مقدارش بر

natural selection - یعنی هر فردی که انسان بدی بیشتری دارد باقی می ماند و طبیعتاً - در این جا معادل این می باشد که هر کسی Fitness
دارد، در نسل بعد منتقل شود یعنی همون عملیات cross-over و mutation بر روی این معادل می شود که Fitness بالاتری دارند

* این الگوریتم و ژن های زیادی دارد و لی هر سوم آن one point cross-over انتخاب می دهد، یعنی از یک نقطه در آن کروموزوم را
قطع می کند، بخش اول کروموزوم اولی را با بخش دوم کروموزوم دومی را قاطع می کند و بالعکس! (این نقطه می تواند random انتخاب شود)

GA - اول میاد در آن را برای combination انتخاب می کند - چطور انتخاب می کند (مثلاً natural selection)؟! با random selection که هم randomness
نقشه هست هم مقدار Fitness دارد. بعد ترکیب آن دو انتخاب شده یک child می گوییم (گاهی در آن child در هم می
جعبه ای افشان می کنند) و با یک احتمال کوچک mutation را انجام می دهد یعنی نروسی ندارد که به cross-over تغییر می دهد، می تواند
امکان در نسل mutation انجام شود و بعد child به جمعیت اضافه می شود.

$$* \text{احتمال حضور هر کروموزوم توی نسل بعد} = \frac{\text{Fitness آن}}{\sum \text{همه Fitness}} \rightarrow \text{هر چه Fitness} \uparrow \rightarrow \text{احتمال حضور} \uparrow$$

له مقایسه با hill-climbing - هر در آن population دارند، معنی stochastic (GA) و Fitness هم مشابه بود
تنها تفاوت: مسافت successor ها $\frac{\text{local}}{\text{beam}}$ همسایه های خون های مفید هستند

له از طریق mutation (که شبیه همون Successor هست) - به جایی می میرد، معانی می توانیم

توانست فاضلی که خوب کروموزوم تقریباً باشد $\xrightarrow{+}$ Cross-over $\xrightarrow{+}$ چه نتایجی می دهیم؟ پرسش به جایی که در آن داره! چون هر بخش از آن
این یک جور سبکی می گوییم که در آن subproblem را حل
کنند و بهر آن را با هم ترکیب می کنند.


له آنی ما این ها می بینیم که اگر کروموزوم نداریم می بینیم چقدرت می شود با هم ترکیب به این معنی باشد که آن ژن از نظر محاسباتی به هم می
را پی می دارند (مثل مسئله ۸ وزیر که خون ها به هم می ترکیب دارند و نیز انتخاب نشده خون اول نوار خون هفتم) و سبکی که بخش از این
رو با یک بخش دیگر از یک کروموزوم دیگر ترکیب می کنیم معنی این است که آنی این داره یک جواب خوب به همون معنی و باعث می کنه
subproblem خوب باشد و اون کتبه هم باعث می شود که این subproblem خوب تر باشد با همی داریم که کنار هم گذاشتن این در آن مقبول
می بینیم بهتر می آید که نروسی این طوری نیست! - به جانب این کار زمتهای قبلی را از این نروسی به (این صبر بیا جابه)؟! نه! چون
این cross-over روی در صدهای از این جمعیت اتفاق می افتد و فرد های نسل قبل هم نسل می شن به این جمعیت و cross-over در طول زمان چون جمعیت
شبیه به هم می باشد احتمال رفتنش به جایی بهترت کم می شود!

۲۴) مقایسه با Simulated annealing : شباهت در این که در طول زمان در آشوبی هم می چرخد و به طایف شروع میزبان (randomness) کم می شود. چرا GA هم همین خاصیت را دارد؟! باین که operator ها با Crossover و mutation است و با هم می چرخد در طول زمان و به تغییرات ناگهانی می نهد.

* در GA چیزی که مهم است طریقی است که در دست طریقی است (درن چینه های که ربط دارند بهم، کنار هم قرار دارند) ممکنه Crossover هیچ کاری به ما نکند، رحمتی operator، Crossover و بهر دلیل جواب ها بهتر بشه 0.00 (یعنی فقط natural Selection و mutation انجام بشه) به شبیه معرکه Stochastic local beam Search است!
* اصطلاحاً باید یک سری بلاک باشد که این بلاک ها را یک سری زیر مسائل را حل می کند اگر این اتفاق بیفتد که بلاک های که زیر مسائل را حل می کند ترن های متوالی باشد، طریقی که در نرم خطی این آسان شود!
* توسعه TSP که نمی دینیم مقدار بهترین جواب می باشد، بهترین چیزی که در افزون مرحله به دست آورده نزارش باشد!

Local Search در جاهایی که جنبه محلی پیوسته است

* تغییراتی که می تواند پیوسته شود و مقدار state ما یک ماهیت پیوسته در فضای چند بعدی باشد و ما بخوایم یک سلسله بهینه سازی در حال کنیم.
* مثال تعدادی شهر و قرار دادن فروگاه که این درن ها - می فضای مجموع فاصله هر شهر از فروگاه حداقل بشود - کلاستر بندی؟! (یعنی بهینه تر)
? له حل کردن به صورت پیوسته - تعداد تغییرات به گونه ای که تغییر کمتر به عیب: مقدار لا خوب انتخاب شده جواب مسئله خوب نشه!

له حل کردن به صورت پیوسته - روش Gradient descent : برای Min کردن تابع است - شبیه hill-climbing است
* اگر هدف ما Max باشد -  - جایی که نزولی هست مقدار مشتق منفی هست
آنها با مقدار مشتق جمع بشه به این معنی هست که در یک مقدار x کوچک می شه پس به سمت Max می بریم
غلط نشه چه باشی دره؟! هر جایی که زیاد است ما آهای ما بزرگتر در دقت به Min و Max می ریم، طول
در $\nabla F(x_n)$ برای فید بک، مشتقات جزئی

له این روش که Gradient ascent هست برای دست هایی است که فضا به $\nabla F(x)$ قرار داد دتی شبه نردبان در برابر معبر قرار داد می توینم نقاط بهینه می مجلس را به سمت آ و ریم ولی معمولاً به این روش عمل نمی شود!
له این چه چیزی تاثیر داره در جواب؟! این که نقطه شروع رو چینی انتخاب کرده باشیم به صورت iterative داریم نقطه را عوض می کنیم (انتخاب شده در شروع) به دقیقاً مثل hill-climbing هست مشکلاتی که می تونه داشته باشه!
له تنظیم پارامتر γ خیلی مهم است، اگر خیلی بزرگ باشه، باعث بزرگ شدن γ و پیرش آن از بهینه می شود و به جایی می ریم!
والی که می باشد، الگوریتم می تونه خیلی کندتر بشه (رشتن به گونه ای که طول می کشد)

مقایسه بین Local Search و Systematic Search
له BFS, DFS, A*

* جواب: در سیستماتیک جواب ما یک راه حل است که مسیر از نقطه شروع هدف را نشان می دهد (جواب شکل مسیر است) - ۸ - پازل
* در Local Search جواب ما یک وضعیت هدف است - همیشه ۸ - وزیر
* Method: در سیستماتیک به مفهوم act را داریم به بحث Tree/graphs پیش میلا وصف Frontier داریم و مسیر ها متعددی را می بینیم برای
* (Current state داریم به بعضی حالتی تعیین می کنیم). (یک مقدار محدود به اندازه population نه می داریم و به ارزشانی می ریم)

هوش مصنوعی { ادراک مناسب } * حافظه : ① به جز DFS بقیه نیایی بودن ② حافظه یک مقدار ثابت است که دیتای نوی (47) population را شامل می شود.

* زمان : ⑤ ضعیف سریع تر از ① فقط ① ها در مسائل کوچک جواب بهینه را به ما می دهند ولی ② در مسائل کوچک هم می تواند جواب بهینه به ما بدهد! (حالتی جواب suboptimal باشد)

* Scope : ① search ② هم Search و هم optimization = یک تابع چند متغیره داریم و می خواهیم خوبی مقدار دهی طوری باشد که تابع Min - Max شود

* state space : ① توانی حالت برای مسئله ۸ وزیر ، state به صورت incremental ساخته می شه از ابتدا حالتی شروع می کنیم و یک وزیر ترستون اون بعد یک وزیر ترستون دوم و ... به این طری ساخته می شه
ولی در ② همیشه با یک Complete Configuration سروکار داریم - هر ۸ وزیر در همان پازلهای مکر در خانه ها می بینیم و state از این ها می شوند ۸ تایی می پره شده است و سعی می کنیم با یک تغییر این ها به state دیگر بزنیم!

باید act (در این بهای فعلی بزنیم)

* در محیط ما که پیچیده و غیر قابل پیش بینی است یعنی محیط non-deterministic باشد ، رندرنیم باید act به کی می ریم و non-observ \subseteq partially observ یعنی هر لحظه دیتای هم اطلاعات لازم برای تصمیم گیری را نداریم و بخش از اون از محیط به دست ما می رسد!

می خواهیم با Search به جواب برسیم ولی نوع ۴ الگوریتم های که تا حالا خونیم در فنی ترین مستقیم استفاده کنیم

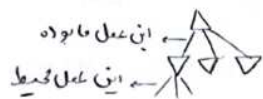
* محیط Non-deter \subseteq partially observ

* در این جا perception برای ما اهمیت پیدا می کند به مثابه مهم نبود چون initial روی درخت و در زمان اعمال act \subseteq K \subseteq اصلاً نیازی نبود که از محیط هیچ perception ای را بگیریم ، فقط مطمئن بودیم که بعد از K گشتن به کی می رسیم

در واقع این جابج result \subseteq transition نتیجه از هر state \Rightarrow این درخت حقیقی بود ، محیط ما در فنی که به کی می ریم به کی می ریم state با چه action ای می ریم و با استفاده از state شدن به رندرنیم در فنی که به کی می ریم
پس در محیط غیر قطعی نه به بودن در فنی اطلاعات بعد از K step می بینیم به کی می ریم

تبعاً تابع result : $S \times A \rightarrow S$ ولی الان $S \times A \rightarrow Y$ باید هر وزیر می بینیم ای از S باشد

* جنبه های حل Search اینها دهیم؟! پس این جا باید متوجه شد که level داشته باشیم و در فنی که act در یک mode انتخاب می کنیم جنبه های حل act که انتخاب داده ایم و گشتن دیتای تصمیم محیط است ، آثار محیط هم دارد می شود به حساب این که به انتخابی در محیط استفاده باشد به حسب انتخاب این action ممکن می ریم به حالت های مختلف



ما می خواهیم هر طوری که محیط حل کرد ، باز هم به جواب برسیم به فنی که به کی می ریم ، محیط هر طوری که انتخاب کرد ، در فنی که به کی می ریم پس mode های که تصمیم می گیریم را نشان می دهند باید به شکل گرهی (and Node) دیده بشن و همی زیر شاخه ها شوند به امی می ریم ولی اون جایی که انتخاب خودمون هست ، گانه یک شاخه انتخاب می کنیم که همی وضعیتی (یعنی رسیدن به امی) را داشته باشد.

* در جابجی که عمل suck ممکنه یک فون را کشن کند یا متا در فنی که را بکشن کند!

805

جلسه ۹

AND Node و تصمیمات محیط

OR Node تصمیماتی که عادل می‌گیرد

AND-OR Search Tree

* وقتی به دنبال جواب هستیم در AND-OR Search Tree ها هستیم - در واقع زیر درختی است که بعضی برگها آن باید min باشد و اون زیر درخت باید طوری باشد که هر گاه (AND Node) ظاهر شده، کل اون زیر شاخه‌ها رو باید شامل بشه - به جف هوش این هست که محیط هر طور عمل کرد اگر مایک تصمیمی گرفتیم مستقل از عملکرد محیط به اون برسیم.

له در مسیر انتخاب عمل ما داره if ظاهر شده - Contingency plan هست [else, ...] [if state sucks]

یعنی شد sequence نذاره توش if ظاهر می‌شه. بر اساس observation از محیط تصمیم می‌گیریم چه کاری بر این انجام دهیم.

AND-OR Tree در Search ← DFS

* خواص DFS استفاده کنیم باید تغییراتی در اون بدم، یکی از شاخه‌ها OR مثل قبل انتخاب میشه، ولی در and هم زیر شاخه‌ها - برای and ما یک and search مجری نوشته‌ست!

* به ازای هر state یکری act وجود داره که فرزندان اون در تشکیل می‌دهند - پس فرزندان هم AND Node هستند

* AND-Search: زیر شاخه‌های and هم OR هستند پس باید OR اون‌ها را مدیریت کنیم و این مثل DFS نیست که اگر در یکی به جواب رسید، جواب را برگرداند! بلکه هم زیر شاخه‌ها باید چک کنند (OR) و اگر محیط اون‌ها Fail برگردونه کل تابع Fail برگردونه!



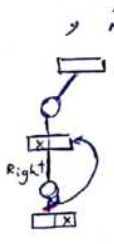
* حرایب جا باید plan می‌که درست می‌ار بر حسب این که محیط چه کار کرده کردیم plan باید اجرا بشه!

اگر عملکرد محیط عمل اول بوده برای P1 درست ادره، حالت دوم بوده P2 و ... - به با توجه به state مفید، عملکردی که از یک زیر درخت درست ادره در اینجا آید - همین قابل if

AO*

* AO* - به جای DFS که باید زیر شاخه‌ها بعضی می‌کنند که در این گرهان A* نیز به پس حساب کردن هزینه، جمع شاخه‌ها مختلف را حساب می‌کنند (چون توهمی شانه‌ها، فواید به جواب برس، جمع شاخه‌های مختلف را حساب می‌کنند و به عنوان هزینه در نظر می‌گیرند به عنوان هزینه گره) برای حالتی که می‌خواهیم از این هم فقط یک محقق بشه و نیازی نباشه هم رو بگذراند (مثل جمع)، از ۵۸۱۹۸ می‌شه استفاده کرد (کتاب هدف ما، شماره ۱)

DFS که زودیم optimal نیست! - چون اولین زیر درخت پیدا شده جواب رو برگردونه (زیر درخت ۱)



* فرض کنید تر محیط غیر قطعی جا برتری، حال اگر مثلاً بجوای به سرعت راست برو یا معبر به راست یا سر جاش باقی می‌ماند و عمل suck داره بر رسته عمل می‌کنه - قرار شد اگر AND داشتیم هم شاخه‌های بشوند ولی آیا این cycle ایجاد شده و روش به عنوان راه حل در نظر گرفت؟ (درخت نیست دیگه چون loop داره دیگه!)

loop می‌که اینجا ایجاد شده تو اکثریم ما به جای if ای که قبلاً ظاهر شده بود، نفس while رو داره!

[suck while state = Δ do Right, suck]

* لیموهای اوتان در plan ما cycle وجود دارد که معادل while می‌شه

* اگر در محیط غیر قطعی، میزان احتمال act ها برآوردن به state ها در برینم، به جای AND-OR Tree چه کاری می‌کنیم؟!

ما می‌خواهیم بین بعضی plan های مختلف می‌فکرم plan ای را انتخاب کنیم که به صورت expected، مقدار کمتر performance را به ما بدهد! (هر جا این معیار کارایی در مسیرهای رسیدن به اهدا بیشتر بود اون راه را انتخاب می‌کنیم)

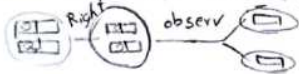
* فرقی با مکتبی این شد که در مکتب ما هر عملی در اینجا ای داریم تا به اهدا برسیم ولی در این مکتب عملی اینجا می‌شود که هر گاه این معیار کارایی را داشته‌باش و این که در مکتبی همیشه نمی‌توان plan ای رو طراحی کرد که در همه شافضا ما به اهدا برسونه (شدن نباشه یا plan قطعی توانایی شه)

پس از روش احتمالی Markov decision process استفاده می‌کنیم (هر گاه کردن اید ریاضی معیار کارایی) (AND-OR جواب نیوه) (مکتب ما)

هدش ۹ * چون تا حالا فقط در مورد observ می‌خوانیم ، بنابراین جواب ما فیزیکی ساده چیزیه که به دست آوریمش یک sequence بود
 * حالا اگر محیط ما partially observ باشه در باره در جواب ما if ظاهر می‌شه؟! (در sequence نشد)
 تو بدون سنسور چون چیزی از محیط دریافت نمی‌کنیم که بخوایم if قرار دهیم پیشش جواب ما باید یک sequence باشه

* تعریف تابع Result در partially observ $(Beliefs \times A \rightarrow Beliefs)$

فرض کنید محیط قطعی باشه ، تو مثلاً جابری هستی هم گفتیم partially observ هست چون می‌دونی تو کدوم فضا هستی ، این فضا فیزیکی هست یا نه ولی
 ایا به فضا جابری چیزی هستی دانی؟! پس در لحظه شروع اگر بگویم در فضا A هستی ، فضا کشف هست ← دوتا physical هست که باهاش
 می‌تونی به این شد Beliefs تا این جا به با act سمت راست رفتی ، (یعنی اینجا یک عمل) + مشاهده ای که از محیط دریافت
 می‌کنی state جدید به دست می‌آید ، به یقین از محیط به دست آوردی (تو مثلاً می‌دانی یک state تبدیل شد نتیجه عمل) + مشاهده
 (در تعریف نا قیستی بهر اش وجود ندارد)



پس اگر act داری Beliefs را بزرگ می‌کنی ولی به سبب این که observation ما می‌تونه در observation ما possible
Beliefs ها به روز رسانی می‌کنی Beliefs → ترکیب Beliefs میانی (که از act + Beliefs ساخته شده) + observation

در search در محیط partially observ به با توجه به Beliefs ، AND-OR Tree که می‌آید

له این جا چه محیط قطعی به غیر قطعی باشه چیزش به بودن پیش می‌آید (deterministic - nondeter) - باید اینجا act ممکن به state بریز
 * این جا هم چنانکه پیش می‌شه به تو متبداً در AND-OR ST ، state داشتیم ، این جا Beliefs داریم .

* در این جا برای AND-OR Search Tree ما state دینی مثل قبل نداریم که بخوایم یک عمل در روش انجام بدهیم ، چیزی و فقط Beliefs
 رو داریم پس لازم است plan ای که در می‌گیریم if توش ظاهر شده ، روی Beliefs در آمده باشه
 مثلاً $if\ Beliefs = [A, B]$ ؟ این شماره ها ، شماره ها Beliefs هستند state ما ؟!
 * چرا این جا هم در search یک sub Tree باید پیدا کنیم؟! چون این جا هم باید نتایج حالت های که برای AND پیش می‌آید رو در نظر بگیریم .

* گاهی اوقات که بخوایم هم می‌شه در دنبال کنیم AND-OR Search Tree کافیه سبازیم کار دشواری هست و مثل این می‌مونه که بخوایم برای هم از این

برنامه ریزی کنیم که حتی ممکن هست این کار ممکن هم نباشه !!! که هم حالات رو به جواب می‌رسونیم ؟ تو محیط دار می‌شی act
 اینجا می‌دیم و perception رو می‌گیریم و با توجه به این که اینجا می‌تیم کار رو ادامه می‌دهیم online search

اشاره دار به شفا از محیط در جویو ، یا محیط می‌تونه یک act انجام بده ، Beliefs ، state ،
 می‌بینی ، مثل یک sequence داره رفتار می‌کنه ← حالا مشکل اش چیه این؟! online search

دوره هر لحظه percept رو دریافت می‌کنی (خوبی Beliefs ها به ازای یک perception رفتی جلو ریت برادر نظر گرفته است)

* مشکل اینجا عملش به طور online ؟ complete نبودن هست چون که از قبل برنامه ریزی نکردی ممکنه در محیط کارهای اینجا
 دهه که غیر قابل بازگشت باشن ← اینجا کارهایی که غیر قابل بازگشت باشن ، ممکنه مانع رسیدن به اهدی نشوند ← مثلاً
 شفا ای پیشی بود که اصلاً به جواب نرسیده (یعنی می‌ری به اهدی وجود نداشته باشه توش) ← تو این حالت ها صحبت می‌کنی ، این
 خوب نیست و بهتر هست که هم چیز را از قبل برنامه ریزی کرده باشی بهر اش

* مثال دیگه ← می‌خواهد ببیند که در میان در که اسمی که از خانه ها دایره ای شکل هست! (تعبیر دیوار است) ربات شروع به حرکت کرده ، در یک
 نقطه روشن شده ، و حالا می‌خواهد ببیند که در کجا تار دارد ← محیط partially observ هست و ربات در هر لحظه می‌تونه در کجای
 خانه های بالا پایین ، چپ ، راست ، دیوار هستند یا نیستند ؟ (تو perception اش هست اینجا)

دھوش

۱۸ اسلاید Interview Search algo

* * map کنی، ادا درونی نمی رانند بکشی map است 0

فرض کنش در شروع، 1، 2، 3، 4، 5، 6، 7، 8، 9، 10، 11، 12، 13، 14، 15، 16، 17، 18، 19، 20، 21، 22، 23، 24، 25، 26، 27، 28، 29، 30، 31، 32، 33، 34، 35، 36، 37، 38، 39، 40، 41، 42، 43، 44، 45، 46، 47، 48، 49، 50، 51، 52، 53، 54، 55، 56، 57، 58، 59، 60، 61، 62، 63، 64، 65، 66، 67، 68، 69، 70، 71، 72، 73، 74، 75، 76، 77، 78، 79، 80، 81، 82، 83، 84، 85، 86، 87، 88، 89، 90، 91، 92، 93، 94، 95، 96، 97، 98، 99، 100، 101، 102، 103، 104، 105، 106، 107، 108، 109، 110، 111، 112، 113، 114، 115، 116، 117، 118، 119، 120، 121، 122، 123، 124، 125، 126، 127، 128، 129، 130، 131، 132، 133، 134، 135، 136، 137، 138، 139، 140، 141، 142، 143، 144، 145، 146، 147، 148، 149، 150، 151، 152، 153، 154، 155، 156، 157، 158، 159، 160، 161، 162، 163، 164، 165، 166، 167، 168، 169، 170، 171، 172، 173، 174، 175، 176، 177، 178، 179، 180، 181، 182، 183، 184، 185، 186، 187، 188، 189، 190، 191، 192، 193، 194، 195، 196، 197، 198، 199، 200، 201، 202، 203، 204، 205، 206، 207، 208، 209، 210، 211، 212، 213، 214، 215، 216، 217، 218، 219، 220، 221، 222، 223، 224، 225، 226، 227، 228، 229، 230، 231، 232، 233، 234، 235، 236، 237، 238، 239، 240، 241، 242، 243، 244، 245، 246، 247، 248، 249، 250، 251، 252، 253، 254، 255، 256، 257، 258، 259، 260، 261، 262، 263، 264، 265، 266، 267، 268، 269، 270، 271، 272، 273، 274، 275، 276، 277، 278، 279، 280، 281، 282، 283، 284، 285، 286، 287، 288، 289، 290، 291، 292، 293، 294، 295، 296، 297، 298، 299، 300، 301، 302، 303، 304، 305، 306، 307، 308، 309، 310، 311، 312، 313، 314، 315، 316، 317، 318، 319، 320، 321، 322، 323، 324، 325، 326، 327، 328، 329، 330، 331، 332، 333، 334، 335، 336، 337، 338، 339، 340، 341، 342، 343، 344، 345، 346، 347، 348، 349، 350، 351، 352، 353، 354، 355، 356، 357، 358، 359، 360، 361، 362، 363، 364، 365، 366، 367، 368، 369، 370، 371، 372، 373، 374، 375، 376، 377، 378، 379، 380، 381، 382، 383، 384، 385، 386، 387، 388، 389، 390، 391، 392، 393، 394، 395، 396، 397، 398، 399، 400، 401، 402، 403، 404، 405، 406، 407، 408، 409، 410، 411، 412، 413، 414، 415، 416، 417، 418، 419، 420، 421، 422، 423، 424، 425، 426، 427، 428، 429، 430، 431، 432، 433، 434، 435، 436، 437، 438، 439، 440، 441، 442، 443، 444، 445، 446، 447، 448، 449، 450، 451، 452، 453، 454، 455، 456، 457، 458، 459، 460، 461، 462، 463، 464، 465، 466، 467، 468، 469، 470، 471، 472، 473، 474، 475، 476، 477، 478، 479، 480، 481، 482، 483، 484، 485، 486، 487، 488، 489، 490، 491، 492، 493، 494، 495، 496، 497، 498، 499، 500، 501، 502، 503، 504، 505، 506، 507، 508، 509، 510، 511، 512، 513، 514، 515، 516، 517، 518، 519، 520، 521، 522، 523، 524، 525، 526، 527، 528، 529، 530، 531، 532، 533، 534، 535، 536، 537، 538، 539، 540، 541، 542، 543، 544، 545، 546، 547، 548، 549، 550، 551، 552، 553، 554، 555، 556، 557، 558، 559، 560، 561، 562، 563، 564، 565، 566، 567، 568، 569، 570، 571، 572، 573، 574، 575، 576، 577، 578، 579، 580، 581، 582، 583، 584، 585، 586، 587، 588، 589، 590، 591، 592، 593، 594، 595، 596، 597، 598، 599، 600، 601، 602، 603، 604، 605، 606، 607، 608، 609، 610، 611، 612، 613، 614، 615، 616، 617، 618، 619، 620، 621، 622، 623، 624، 625، 626، 627، 628، 629، 630، 631، 632، 633، 634، 635، 636، 637، 638، 639، 640، 641، 642، 643، 644، 645، 646، 647، 648، 649، 650، 651، 652، 653، 654، 655، 656، 657، 658، 659، 660، 661، 662، 663، 664، 665، 666، 667، 668، 669، 670، 671، 672، 673، 674، 675، 676، 677، 678، 679، 680، 681، 682، 683، 684، 685، 686، 687، 688، 689، 690، 691، 692، 693، 694، 695، 696، 697، 698، 699، 700، 701، 702، 703، 704، 705، 706، 707، 708، 709، 710، 711، 712، 713، 714، 715، 716، 717، 718، 719، 720، 721، 722، 723، 724، 725، 726، 727، 728، 729، 730، 731، 732، 733، 734، 735، 736، 737، 738، 739، 740، 741، 742، 743، 744، 745، 746، 747، 748، 749، 750، 751، 752، 753، 754، 755، 756، 757، 758، 759، 760، 761، 762، 763، 764، 765، 766، 767، 768، 769، 770، 771، 772، 773، 774، 775، 776، 777، 778، 779، 780، 781، 782، 783، 784، 785، 786، 787، 788، 789، 790، 791، 792، 793، 794، 795، 796، 797، 798، 799، 800، 801، 802، 803، 804، 805، 806، 807، 808، 809، 810، 811، 812، 813، 814، 815، 816، 817، 818، 819، 820، 821، 822، 823، 824، 825، 826، 827، 828، 829، 830، 831،

بلاور پائین robot (مادیار صحت) سے پس فقط کہ خونہ Belief این شرایط و دارہ ؟

online Search ← منقوعون این هستند که از قبل اینها

act فعل، انجام دهیم؛ یعنی بنی اثبات act و بنیاد زینت و صورت (inter lift) داریم که می بینیم؛
* percept های تارن گفته رد داریم

* چرا به دردس خورد؟! چرا از اسلحه

+ Fully obser

وینا عملی $full\ observation$ و تحقیق هست، act و اجرا کنیم و دیدیم که هیچی نیست. پس باید act را به $full\ observation$ تبدیل کنیم.

online search ها به درد خط های که partially observed و نه قیاسی خورد. این طور به جای برآورد از آن ها شایعتر، فقط برآورد
شان را که رخ داده برآورد می کنیم. پس یعنی در یک نوع قضایا از قضایای کلی می بینیم، به همین دلیل جواب هستیم

* نکته: در دین خود پیمبران فقط صایه نه dynamic یا semi-dynamic هست! جامایی نه وقت محدود

فیکٹا ← unknown (محیط آفرقیه) unknown باشد فی درینج: state A به act های قابل اعمال هسته؟

(طعمه ادرمان هم می بینیم Acet هاشمی هست، ولی نفس در نیمه راه ایستاده و می خنجر می زند Cost ای به ادرمان خوشه ها
می خنجره. تازه بعد از انجام act می فهمیم که Cost آن قیمت بوده.)

* **تشیخ همون** Search های که قبلاً داشتیم، مثل BFS، در این جا هم استفاده کنیم؟! چه تغییراتی نیاز داره؟
* (شاید این کار در online چند حرکت random نیاز داریم تا بتواند در راستای اطلاعات از محوطه فراموشی بدویم)

[illegible]

این سرورین در یک state هستیم. اگر بخوایم مثلاً BFS یا A^* عمل کنیم، تو یک state هستیم و باید در $expand$ می‌کنیم، بعد می‌خوریم state بعدی
رفتم به state ای که می‌خوریم به جای یک مجاری باقی می‌ماند. ! چون آفرین می‌خوریم به state بعدی.