

دوره ۵ لغت و لست هولی مخصوصی - هفته اول

دوره‌نامه‌پیم فصل‌های ۳ و ۴

نوعی از عامل‌های هست برخلاف: عامل‌های حل مسئله

اطلاعاتی / درایم: تعریف مسئله → جستجوی نا آگاهان

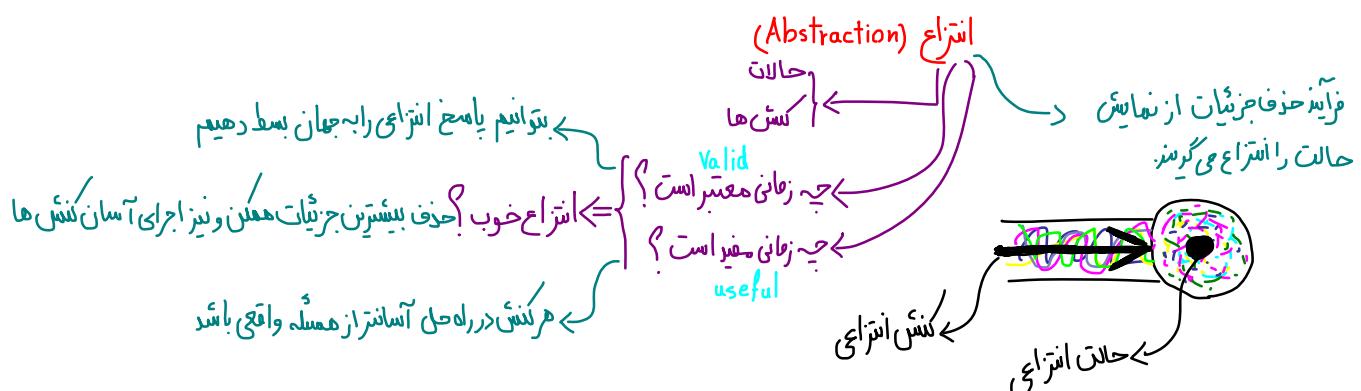
وظیله عامل‌های هستند. بینشید کردن معنای کارایی

- 1- فرموله سازی هدف (اولین قدم حل مسئله)
- 2- فرموله سازی مسئله
- 3- جستجو
- 4- اجرا

طراحی عامل: فرموله سازی (هدف و مسئله)، جستجو، اجرا - فرموله سازی هدفی دریم

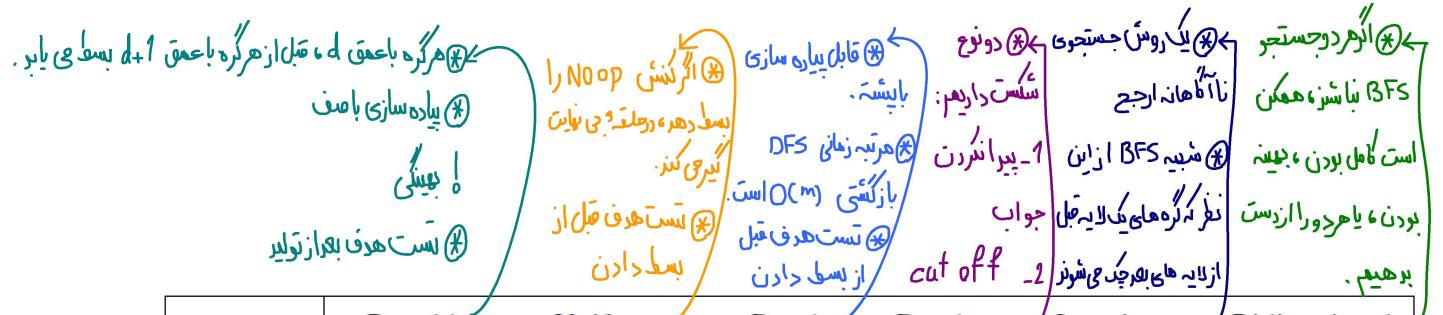
(C, A, B) ← یعنی هزینه رضت از A بـ B؛ با این a چقدری شود.

لینکت راه حل پیراشه، با هزینه همسیر ازازهایی می‌شود → پاسخ دهیم: همین هزینه



$$n \times 2^n \quad (*) \text{ خنای حالت جهان جارویی = }$$

* برخی اصطلاحات: تولید - بسط (لسترس) - هلاقات (visit) - بـ فالتو را شعاب - بـ عمق کم عمق ترین هوف - m هالزیم عمق درخت جستجو استراتژی جستجو - (n) و هزینه همسیر - حاسنه - Trade off
 که مثال: هزینه جستجو - هزینه اجرا - هزینه کلی
 ۴h + 22h + ۱۸h



Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes ^a	Yes ^{a,b}	No	No	Yes ^a	Yes ^{a,d}
Time	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes ^c	Yes	No	No	Yes ^c	Yes ^{c,d}

Figure 3.21 Evaluation of tree-search strategies. b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: ^a complete if b is finite; ^b complete if step costs $\geq \epsilon$ for positive ϵ ; ^c optimal if step costs are all identical; ^d if both directions use breadth-first search.

^a: کامل است اگر b محدود باشد.

^b: کامل است اگر هر یک حمله کام بزرگتر از مقادیر مثبت ϵ باشد.

^c: بینه است اگر هر یک تابع کامهای برابر باشد.

^d: اگر هر دو جستجوی BFS باشند.

* b^d کم عمق ترین هدف را بهتر دارد.

* در UCS: C^* یعنی مسیر بهینه از رسیده تا هدف و کوچکترین هزینه کننده

* هرتیب زمانی و حافظه UCS، یعنی $O(b^{1+\lfloor C^*/\epsilon \rfloor})$ می تواند بسیار بزرگتر از b^d باشد. زیرا اول مسافت هایی از درخت را که راهی کامهای زیاد با هر یک کم هستند بررسی نمی شوند و سپس به بررسی قصص هایی از درخت راه را که راهی کامهای بزرگ است (واحتمال یافتن جواب در آنها بیشتر است) می پردازند.

* وقت هزینه همه کامهای برابر باشد $O(b^{1+\lfloor C^*/\epsilon \rfloor})$ خواهد بود. (در UCS)

* در جستجوی عمق محدود (DL5)، کامی حدودیت عمق می تواند برایه داشت که مسله باشد \leq مثلاً نقصه روانی 20 شردار $\leq l=19$ خوب است. اما هر شرمندی برای داشتن l کام ممکن است (بنابراین قدرتمندی حالت محدودیت بیشتری است)

$$f(n) = g(n) + h(n)$$

Recursive Best-First Search

استراتژی های جستجوی آنالوگی (Greedy best-first search, A*, IDA*, RBFS, SMA*)

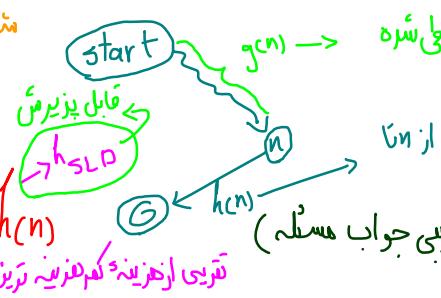
$f(n) \leftarrow$ تابع ارزیابی از هزینه کم هزینه ترین مسیر از نور تا هدف <=> دیگاهی: اگر بخواهیم از نور به هرن درسم جدار باشد هزینه بدم؟ - نگاه رو به جلو
 $f(n) \leftarrow$ تابع ارزیابی

+ جستجوی بهترین - اولین حریصانه \leftarrow چگونی کار کرد الگوریتم \leftarrow تحلیل
 (بالا) \leftarrow جستجوی بهترین اولین حریصانه از این جهت که همواره مسیر را
 (بالا) \leftarrow بجهود زمانی و حافظه در برترین حالت
 شاید جستجوی عمق اول است.

$f(n) = g(n)$ شبیه UCS با مین تراویت در UCS
 $f(n) = g(n) + h(n)$ هرینه تجھشانه کی سده از نور start تا نور n
 $f(n) \leftarrow$ ترتیب از هزینه کم هزینه ترین راه حل برای رسیدن به هدف از طبق نور n

start \rightarrow مسافت واقعی کی شده
 قابل پذیرش \leftarrow تجھن هزینه از آنها
 حداقل \leftarrow هدف
 (همینکه کردن هزینه کلی تقریبی جواب مسئله)
 \leftarrow جستجوی (رفت)

* باز اسن شطی، * کاملاً است و هم بشه
 (در جستجوی رفت)



شرط شرط
 ↓
 قابل پذیرش

یعنی (۱) هیچ گاه هزینه رسیدن به هر فارا بیشتر از مقدار
 واقعی تمیب نزدیکی او باشند

* یک نکته بسیار مهم: اگر $h(n)$ تابعی ساز کار باشد و آنکه مقادیر تابع $h(n)$ در همه مسیرها غیرنزوی خواهد بود:

(مسوا سازی $h(n)$)

در الگوریتم A نوزه هارا بر اساس تابع ارزیابی بسطی دهیم

* بایهاین <بالا> گرهای بسط داده شده تو سط الگوریتم A در جستجوی گراف به ترتیب غیرنزوی مقادیر تابع $f(n)$ است.

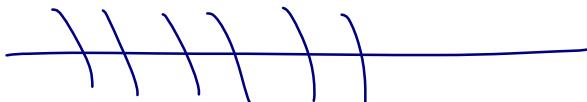
بنابراین اولین گره هر فری که بسط داده شود، پاسخ بهینه مسئله است و بعد از آن هر گرهای بسط داده شود، حداچی لشود، حداچی متذاری برابر نه
 هف بهینه خواهد داشت. \Leftarrow خطوط کانتر

کل مطالب این صفحه بافرض استفاده از A^* در جستجوی گرافی است

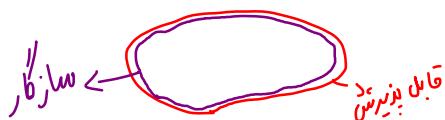
شرط قابل پذیرش بوان h ، برای A^* در جستجوی گرافی کافی نیست.

۱- این دو همسایه که به یک رُوّه ختم می شوند ممکن باشند.
برای A^* در جستجوی گرافی کافی نیست.

۲- تضاد نیم نه همراه هزینه همسایه ممکن است.
در صورت ظیور حالت های سازگاری، همان اولین حالت اتفاق افتاده باشد. \Rightarrow سروط سازگاری \Leftarrow سروط سازگاری بتابع h مربوط است.



در h^* اولین باری که نود برای بسط دادن انتخابی شود
همان دفعه بهینه است.



شرط سازگاری (consistency) یا یکنواختی

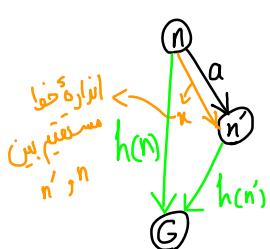
با ازای هر دو n, n' که n کره پسین n' باشد که بالش n تولید شده است، هزینه تخمینی رفن از n به n' بزرگ است و هزینه رفن از n' به n بزرگ است:
 $h(n) \leq c(n, a, n') + h(n')$

* هر تابع هموریستیک سازگار، قابل پذیرش هم هست (تمرین 29)

* از چه شرط سازگاری بسیار محدود شده تراز قابل پذیرش لدن برای تابع ابطری است
اگر نظر توابع را می توان یافت که قابل پذیرش ناشن افاسازگار نباشند.

(نوعی هر دو بحث ما سازگار نیز هستند)

* h_{SLD} سازگار است. (نامساوی هائی برای خطوط هسته هم برقرار است)



ادامه جملات مهم از کتاب:

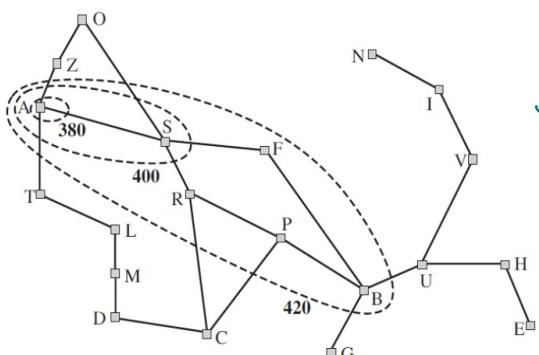
+) A^* معکن است بعفی گره های روی «حلقه هدف» (جایی که $C^{f(n)} = f(n)$) را قبل از انتخاب گره هدف بسط می کند.

+) به طور شمودی واضح است که اولین حواب ، همان جواب بعینه هسته است، زیرا تمامی گره های هرف دیگر در حلقة های بعیتی قرار ندارند ، ازای مقادیر $f(n)$ بیشتری هستند و برابر با مقادیر $f(n)$ بالاتری (از $f(n)$ زیرا در تمامی گره های هدف $= f(n)$ است) همچین با طور شمودی واضح است /
له الگوریتم A^* کامل است زیرا همیختن گره های ما حلقة هارا بر اساس افزایش مقادیر $f(n)$ اضافه می کنیم ، سراغم به حلقت ای هی رسید که در آن مقادیر $f(n)$ برای هر یکی همسیر رسیدن به حالت هدف است.

+) همچنان که ای با ویژگی $C^{f(n)}$ را بسط نخواهد داد ، برای مثال به سکل 3.24 نکاهتیز که تیمیسوارا با ایکه فرندر یعنی است ، همچنان که بسط داده نشود.

+) در هر دو الگوریتم های بعینه از این نوع (الگوریتم هایی که در آنها جستجو از ریشه آغاز و بسط می یابند) الگوریتم A^* به ازای هر تابع ابطری دلخواه به طور بعینه ای کار است ، بین معنی الگوریتم بعینه دیگری وجود ندارد که در آن تعداد گره های بسط داره سه که تراز الگوریتم A^* باشد (بجز اختصار آنها که انتخاب بین گره هایی با $C^{f(n)}$ اتفاق می افتد) (لیکن این است که در الگوریتم هایی که تمامی گره های با ویژگی $C^{f(n)}$ بسط داده هستند ، این خطا وجود دارد که حواب بعینه هسته از دست برود.

تعداد گره های بسط ادامه سده توسعه A^*



رشته ای تعداد گره های دارد که حلقة هدف اتفاق می افتد مگر اینکه رشته ای تابع همیزی از الگوریتم هایی که همیزی واقعی رفت از گره n به هدف از الگوریتم هایی که همیزی واقعی سریعتر باشند.

$$h^* = \text{هر یکی واقعی رفت از گره } n \text{ به هدف}$$

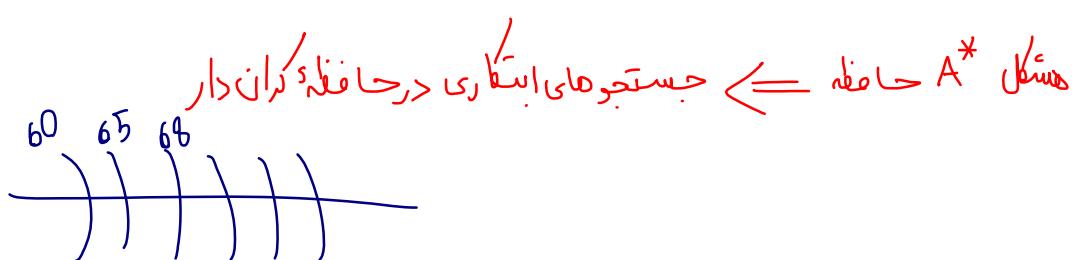
$$h^* = 1024$$

$$|h(n) - h^*| \leq O(\log h^*)$$

فقط همه واقعی رفت از یوز و تا هر چند از یوز و تا هر چند

$$\frac{n}{10} \leq h(n) \leq 1034$$

در عمل برای الگوریتم ابطری ، حد اعمال خطای نسبتی با همیزی همسیر است . \Rightarrow رشته ای اغلب اتفاق می افتد .



IDA*

$f(n) = g(n) + h(n)$ \leftarrow تو پسیح جلوتی کاربرد الگوریتم IDA*

\leftarrow افزایش تدریجی بر اساس

لکه در برتردار، هقار برعن برابر کوچکترین مقادیر $f(n)$ رای است نه از مقدار برعن قبلی تجاوز نموده است.

* IDA* در شرایطی که هزینه $f(n)$ ها واقعاً درست محاسبی است و از سر برآمد سایز برابر باشد.
لکه این صفت هریک شرایطی از ترتیب ها خود را ری تکن.

$f(n)$ \leftarrow

\leftarrow تو پسیح جلوتی کاربرد الگوریتم RBFS \leftarrow برهانی f بسط / فناوری حافظه خنثی

\leftarrow شبیه جستجوی عمق اول بازگشتی

\leftarrow تا حدودی از IDA* کاربرد است ولی همچنان
مسئل بسط دادن های تکراری گذرا ادارد.

در صورتی که n قابل پذیرش داشته باشد \leftarrow کامل است

مرتبه زمانی و حافظه
بیشترین \leftarrow مخفی
 \leftarrow بدقت n و اینسته است

* بزرگترین مسئل IDA* و RBFS : از حافظه بسیار کم استفاده می کند.

\leftarrow راه حل \leftarrow الگوریتم \leftarrow MA*
 \leftarrow SMA* \leftarrow SMA*

\leftarrow جلوتی کاربرد الگوریتم

لکه بیرون گره را دوری از ازار (گره ای که بین تین هنوار آزاد است) افاهتار گره و اهرس شده را به والد آن برمی گرداند

\leftarrow رمانی که الگوریتم غافیکه تمام مسیرهای (گله)، برتر آزمیز فراموش شده هستند، دوباره آن را ایجاد می کنند.

SMA* بیرون گره را بسطی دهد و بیرون گره را دوری از ازار (بر اساس f) \leftarrow سوال: آنکه گره های اداری مقادیر

مساوی بودند چه؟ بیرون گردید تین \checkmark قدیمی تین و بیرون X \leftarrow سوال: آنکه شیوه هردوی اینها یعنی بود چه؟

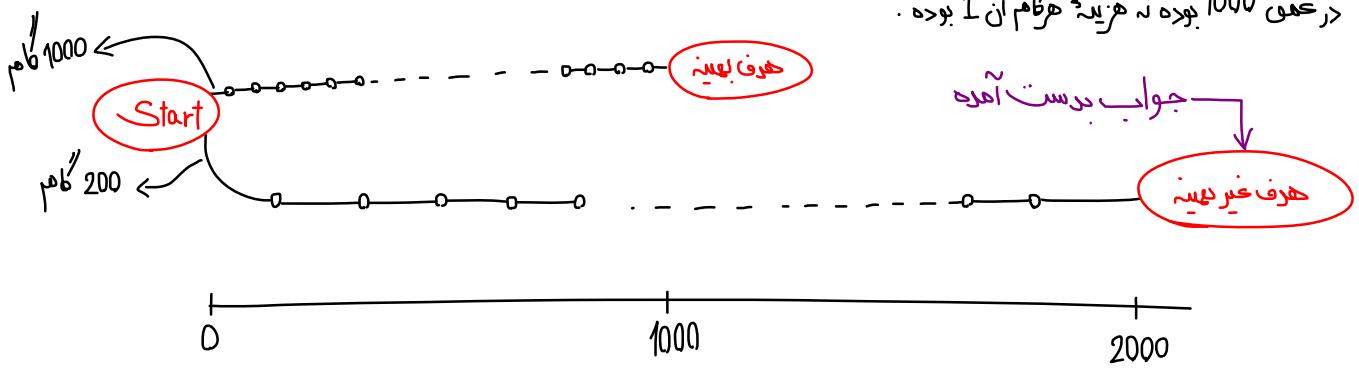
در این حالت درخت جستجوی جاری یک همسیر واحد از رشته تابگ است (همم) که تمام حافظه را

استخال کرده است بنابراین اگر بگ همود نظر هوف نباشد، جواب مرد نظر با فناوری حافظه مرجح است قابل سترس نخواهد بود.

کامل است اگر جواب قابل سترسی وجود داشته باشد؟ یعنی h (عنوان سطحی تین هرف) که تراز سایز حافظه در واحد گره باشد.

لکه این است اگر جواب بعینه مایل سترسی دارد و وجود داشته باشد \leftarrow غیر این صورت بیرون گره جواب مایل سترس را برمی گرداند.

جملات اضافی در هر دو الگوریتم SMA^* و $ابد$ توجه داشتند که این الگوریتم زمانی بجوابی رسید که d کمتر از سایز حافظه (در واحد) باشد، اما حیثیت این جواب بعینه هر کام (طریق همسیر اسمری به شفر دیگر) هنگفت است، ممکن است SMA^* قادر به پیدا کردن جواب بعینه نباشد چون تعداد کام‌های یافتن این جواب بعینه بین این زمانهای مختلف باشد مثلاً ۱۰۰۰ کام، اما این الگوریتم هی تواند بعینه در حقیقت ۲۰۰ است را باید، چون هنلاً این زمانهای مختلف این ۳۰۰ است اما این عدد ممکن است بعینه نباشد. برای مثال قابل توان فرض کرد هر یک کام از این ۲۰۰ کام ۱۰ است که بعینه ۲۰۰۰ و این همسیر بعینه در حقیقت ۱۰۰۰ بوده و هر کام آن ۱ بوده.



لوبایع ابتدایی (هیوریستیک)

نکته اولیه این است که هر راه را که از این راه رفته باشیم را نمی‌توانیم دوباره رفته باشیم.
 $h_1 = 8$ = مسافت ۸ - پازل \leftarrow تو صفحه در هر دو مساله
 $h_2 = 18$ = مسافت منتهی \leftarrow ترایج ابتدا کاری
 $Pattern\ Database = h_3$
 $Disjoint\ Pattern = h_4$
 Data base

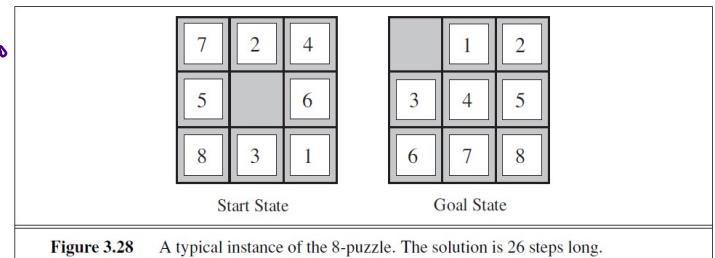


Figure 3.28 A typical instance of the 8-puzzle. The solution is 26 steps long.

$$h_2 = \underline{3} + \underline{1} + \underline{2} + \underline{2} + \underline{2} + \underline{3} + \underline{3} + \underline{2} = 9 + 8 + 1 = 18$$

$$\nexists h_1 \leq h_2 \leq h_3 \leq h_4 \nexists$$

$$N+1 = 1 + b^* + (b^*)^2 + (b^*)^3 + \dots + (b^*)^d$$

نحوی کام
توحدات های بسیار اندیشه
لوسٹر الگوریتم *

+) از دست نایج ابتدا کاری بکاری \leftarrow چیست؟
 فائزه اسناید هموز
 این راه را باید نایج ابتدا کاری

+) آنکه $h_1 \leq h_2 \leq h_3 \leq h_4$ و قابل پذیرش باشد و h_1, h_2, h_3, h_4 مرتبی باشند.

دسته ای که جزء مجموع همورسمیس را نشان می‌دهد، آنها قابل پذیرش هستند با بهتر است از تابع اینکاری بزرگ استفاده نمایم.

$$h(n) = \max\{h_1(n), h_2(n), h_3(n), \dots, h_m(n)\} \rightarrow \text{قابل پذیرش}$$

برنامه تابع اینکاری تعمیل رله زد این بر لرکار ارد

سازکار

اُسوي چستجوی کلاسیک

الگوریتم‌های جستجوی محلی و همسانی بین ساری \rightarrow معمات

- جستجوی تپه نوردي (عاري) - تهادفي - اولين انتخاب - شروع بعد تصادفي
- جستجوی شبیه سازی ذوب فلزات - بازیخت شبیه سازی شده - تبرید شبیه سازی شده
- جستجوی بروی محلی \leftarrow انواع
- بزرگ‌باشناهی \leftarrow ریزسته (صراب قابل تقویت بررسی آوردن)

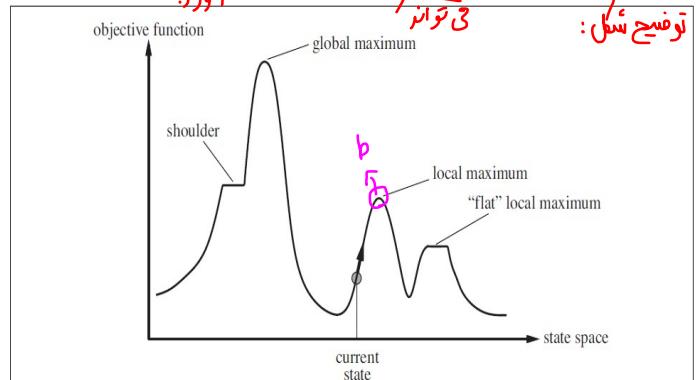


Figure 4.1 A one-dimensional state-space landscape in which elevation corresponds to the objective function. The aim is to find the global maximum. Hill-climbing search modifies the current state to try to improve it, as shown by the arrow. The various topographic features are defined in the text.

hill-climbing
(steepest-ascent version)

+ جستجوی تپه نوردي \leftarrow ايند و معمات (جستجوی محلی و همانه)

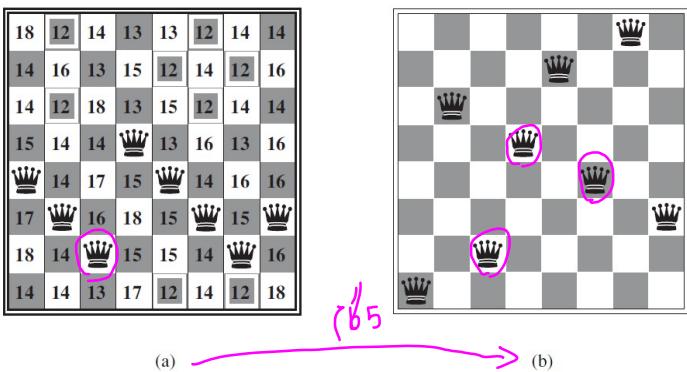


Figure 4.3 (a) An 8-queens state with heuristic cost estimate $h=17$, showing the value of h for each possible successor obtained by moving a queen within its column. The best moves are marked. (b) A local minimum in the 8-queens state space; the state has $h=1$ but every successor has a higher cost.

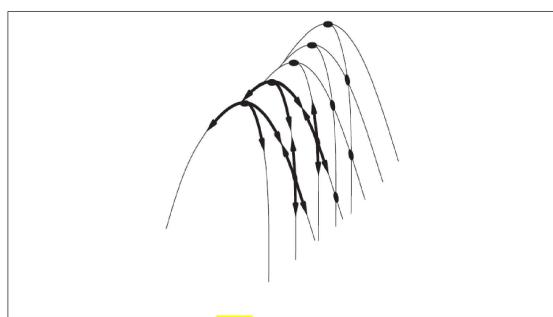
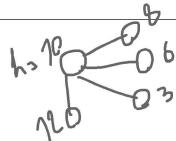


Figure 4.4 Illustration of why ridges cause difficulties for hill climbing. The grid of states (dark circles) is superimposed on a ridge rising from left to right, creating a sequence of local maxima that are not directly connected to each other. From each local maximum, all the available actions point downhill.



First-choice hill

Random-restart hill

نیمه نوردي تصادفي

$$\left(\frac{1-19}{20}\right) \times 64 + (1 \times 21) \sim 25$$

انواع الگوریتم تپه نوردي

نقرار شروع های قدر \rightarrow ابتدا P \rightarrow پنهانی شروع هجد تصادفي
مورد انتقال = $\frac{1-P}{P}$

نثاریه افزا، و نری فلز را بعدم:

*) هدفیت الگوریتم تا نوری به شون وابسته به سطح فناوری حالت است
باز پخت شبیه سازی شده - شبیه سازی دو ب فازان

+) جستجوی شبیه سازی لاختنی - تبرید شبیه سازی شده

چگونه برای ما مفید است؟

آرمانه در این راه الگوریتم

زیارت و به مرور زمان

کمی سرد.

$\sqrt{5}$ آنرا زیاد باش

حاصل سر بردار ب ۱ >

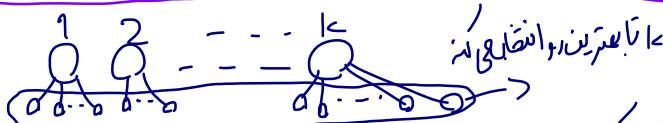
نرود است

$\sqrt{5}$ آنرا کم با سو

حاصل سر بردار بروید و تبرید بضم قیس

(*) درونی ترین حلقة الگوریتم تبرید شبیه سازی شده بسیار شبیه الگوریتم تا نوری است - همچنین

(*) می توان اینها را که آنها را در اینجا نمایند کافی نمایند اما این الگوریتم با این اتفاق نزدیک به ۱ می نمایند که اینها را بین اینها بروید و تبرید بضم قیس



Local beam search

+) جستجوی بروی محلی - چلوئی کار - بجای ۶ عالی ، ۲ عالی نموداری نیم

* ثالثه مهم - کار با الگوریتم تا نوری سروع بورصه صادری (کبار) همفاوئی است.

جستجوی بروی نهادی

stochastic beam search

(*) همین است از گهدان نوع ریاضی که این روز در زانه ای درین از مقاماتی خالص مدرنیتی دارد.
(*) از جستجوی بروی نهادی است. علیرغم انتساب طبعی بسیار شبیه جستجوی بروی نهادی است.

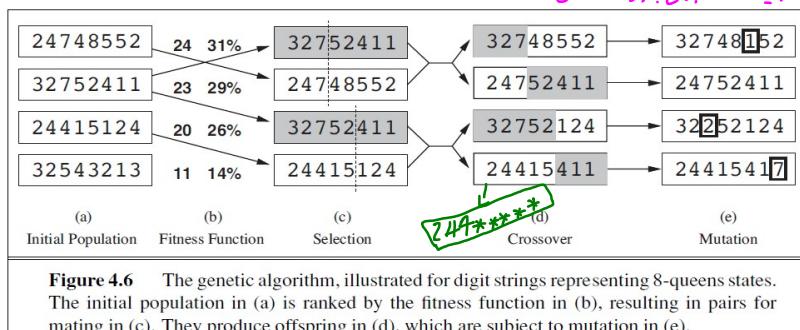


Figure 4.6 The genetic algorithm, illustrated for digit strings representing 8-queens states. The initial population in (a) is ranked by the fitness function in (b), resulting in pairs for mating in (c). They produce offspring in (d), which are subject to mutation in (e).

+) الگوریتم زیستی (GA)

Cross over Point
به صورت نهادی انتقالی شود

(*) و می رویانه وال طبله متناوب هستند آنها هم از چنانچه تو از طلبی را نمی نهند که وال دین آن بسیار متناوب باشند.

(*) الگوریتم زیستی در صورتی که الگوهای

آن با هم نهادی با معنی خاصه ای باشند داشته باشند، میتوان عمل در را در اینجا انجام داد.

GA
نهادی نهادی

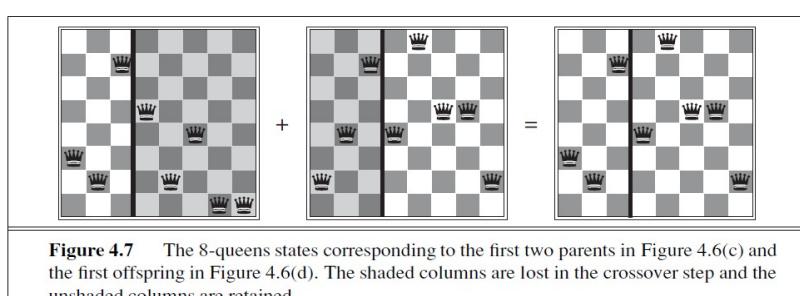


Figure 4.7 The 8-queens states corresponding to the first two parents in Figure 4.6(c) and the first offspring in Figure 4.6(d). The shaded columns are lost in the crossover step and the unshaded columns are retained.

AND-OR درخت‌های جستجوی

- 1- نورهارف کوی هر برس و بوداره
 2- تر نورهاری OR دیقاً یک اسن و هسته‌های مده باش
 3- تر نورهاری AND حکماً نسخابان هفتارو
 رفت است
 که
 ساصل بشه.

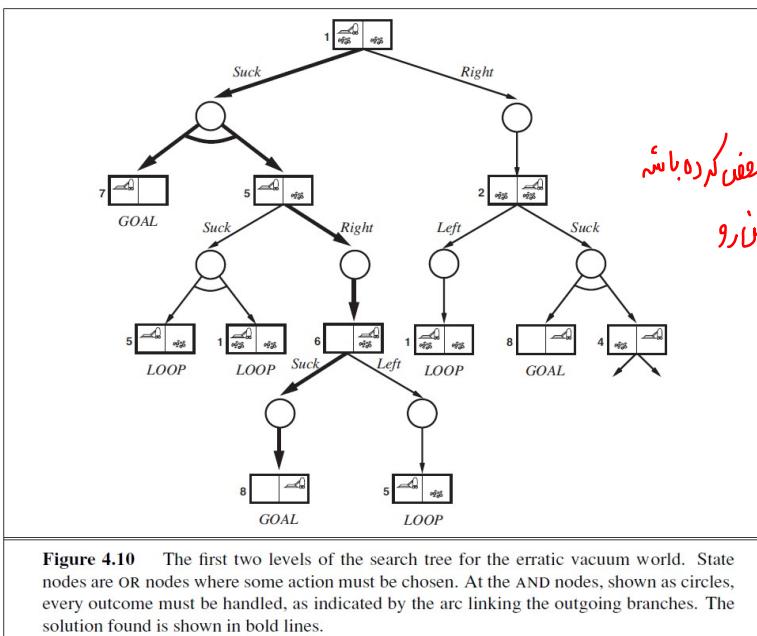
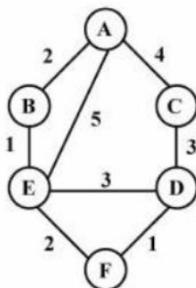


Figure 4.10 The first two levels of the search tree for the erratic vacuum world. State nodes are OR nodes where some action must be chosen. At the AND nodes, shown as circles, every outcome must be handled, as indicated by the arc linking the outgoing branches. The solution found is shown in bold lines.

سوال کنکور سال 95 فناوری اطلاعات

گراف زیر را در نظر بگیرید. گره A وضعیت شروع و گره F وضعیت هدف را نشان می‌دهند. با استفاده از روش جستجوی uniform cost search ترتیب ملاقات گره‌ها به صورت کدام یک از موارد زیر است؟ عدد کنار هر بال هزینه عمور از آن بال است.



- (۱) A,B,E,C,F
- (۲) A,B,E,D,F
- (۳) A,B,E,D,C,F
- (۴) A,B,E,C,D,F

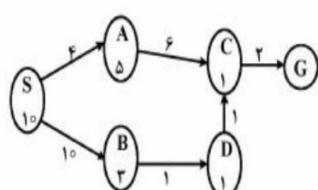
سوال کنکور سال 96 فناوری اطلاعات

در مورد تابع اکتشافی (heuristic)، کدام عورد نادرست است؟

- (۱) اگر h قابل قبول باشد سازگار هم هست.
- (۲) اگر تابع h قابل قبول باشد تابع $\frac{h}{2}$ هم قابل قبول است.
- (۳) اگر h_1 و h_2 هر دو سازگار باشند $\max(h_1, h_2)$ هم سازگار (consistent) است.
- (۴) اگر h_1 و h_2 هر دو قابل قبول (admissible) باشند $\max(h_1, h_2)$ هم قابل قبول است.

سوال کنکور سال 96 فناوری اطلاعات

مسئله جستجوی زیر را با نقطه شروع S در نظر بگیرید. مقدار تابع اکتشافی (heuristic) در گره S نوشته شده است. دو تا اولین گره‌هایی که بعد از S توسط الگوریتم^{*} A و حریصانه اول بهترین (greedy best first) گسترش می‌یابند را مشخص کنید؟



- (۱) A*: اول A و بعد B: Greedy + A*
- (۲) A*: اول A و بعد B: Greedy + A*
- (۳) A*: اول A و بعد C: Greedy + A*
- (۴) C: اول A و بعد C: Greedy + A*

سوال کنکور سال 97 فناوری اطلاعات

فرض کنید h یک تابع مکاشفه‌ای (heuristic function) است و تابع h^* مقدار بهینه هزینه رسیدن به هدف را از هر وضعیت در فضای وضعیت‌ها می‌دهد. همچنین فرض کنید برای یکی از وضعیت‌ها به نام S که روی یک مسیر بهینه از وضعیت اولیه به یک وضعیت هدف قرار دارد $h(s) = 2 \times h^*(s)$ و برای سایر وضعیت‌ها مقدار تابع h حداقل برابر مقدار تابع h^* است. کدام مورد در خصوص الگوریتم A^* که از تابع مکاشفه‌ای h و روش جستجوی درختی (tree Search) استفاده می‌کند، درست است؟

- (۱) چنانچه در فضای جستجو تنها یک مسیر بهینه به وضعیت هدف وجود داشته باشد، این الگوریتم آن مسیر را می‌باید.
- (۲) از آنجایی که h یک تابع قابل قبول (admissible) نیست، این الگوریتم هیچ گاه مسیر بهینه را نمی‌باید.
- (۳) هزینه مسیر یافته شده توسط این الگوریتم حداقل دو برابر هزینه مسیر بهینه است.
- (۴) وضعیت S با اجرای مسیر یافته شده توسط الگوریتم ملاقات نخواهد داشت.

98 - IT

5 -

کدام گزینه در خصوص روش‌های جستجو درست است؟

- (۱) اگر برای دو تابع مکاشفه‌ای h_1 و h_2 و برای هر وضعیت S داشته باشیم: $h_1(s) \geq h_2(s)$. آنگاه همیشه بهتر است در جستجوی A^* از تابع h_1 استفاده کنیم.
- (۲) اگر تابع مکاشفه‌ای به کار رفته در جستجوی A^* قابل قبول (Admissible) نباشد، این روش هرگز راه حل بهینه را نخواهد یافته.
- (۳) جستجوی عقب‌رو (backward search) همواره سریع‌تر از جستجوی جلو‌رو (forward search) عمل می‌کند.
- (۴) جستجوی دو سویه (Bidirectional) برای حل بعضی مسائل، کندر از جستجوی جلو‌رو عمل می‌کند.

98 - IT

6 -

کدام مورد در خصوص روش جستجوی اول-بهترین حریصانه (Greedy Best First search)، درست است؟

- (۱) از بین گره‌های موجود در صفحه، گره مرتبط با مسیری را که کمترین هزینه از وضعیت شروع را داشته است، گسترش می‌دهد.
- (۲) درصورتی که از تابع ابتکاری سازگار (consistent heuristic) استفاده کند، روشی بهینه است.
- (۳) حافظه مورد نیاز برای این روش به صورت خطی بر حسب حداقل عمق جستجو است.
- (۴) این روش یک روش جستجوی کامل است.

nite loop. (The graph search version is complete in finite spaces, but not in infinite ones.) The worst-case time and space complexity for the tree version is $O(b^m)$, where m is the maximum depth of the search space. With a good heuristic function, however, the complexity can be reduced substantially. The amount of the reduction depends on the particular problem and on the quality of the heuristic.

مهمه ۹۳ ارسن ۳

۷ - فرض کنید در یک درخت جستجو، مسیر بهینه برای دستیابی به اهداف به گره هدف G ختم می‌شود، و دو گره n و n' بر روی این مسیر قرار دارند به طوری که n' فرزند n است. فرض کنید h_1 یک تابع مکاشفه‌ای قابل قبول (admissible) است. h_2 یک تابع مکاشفه‌ای سازگار (consistent)، و g تابعی باشد که برای هر گره، هزینه رسیدن از گره ریشه تا آن گره را نشان می‌دهد. اگر توابع f_1 و f_2 به ترتیب از جمع هر یک از توابع h_1 و h_2 با تابع g حاصل شوند، کدام مورد الزاماً صحیح است؟

$$f_1(n) \leq f_1(n') \quad (1)$$

(۴) گزینه‌های ۱ و ۳ صحیح هستند.

$$f_1(n) \leq f_1(G) \quad (2)$$

(۳) $f_2(n) \leq f_2(n')$

سوال کنکور سال ۹۵ مهندسی کامپیوتر

-8

فضای زیر را در نظر بگیرید که عامل در هر خانه می‌تواند یک از چهار حرکت رفتن به بالا، پایین، چپ یا راست را انجام دهد. خانه شماره ۱ وضعیت شروع، و خانه شماره ۱۱ وضعیت هدف است. همین‌طور، خانه ۸ مسدود است. اگر عامل حرکتی انجام دهد که به خانه ۸ یا دیوارها برخورد کند، سرجایش باقی ماند. فرض کنید هر یک از حرکتها یک واحد هزینه دارد. اگر در هر گره، از فاصله موسوم به فاصله منهتن (Manhattan) آن گره تا هدف به عنوان مقدار تابع اکتشافی (heuristic) استفاده شود، سه گره اولی که در الگوریتم A^* گسترش می‌یابند کدام است؟ اگر شرایطی پیش آمد که دو خانه برای گسترش دقیقاً وضعیت یکسانی (از نظر A^*) داشته باشند، خانه با شماره کوچکتر انتخاب می‌شود.

۲	۶	۹	۱۲
۲	۵	۸	۱۱
۱	۴	۷	۱۰

۵,۴,۱ (۱)

۷,۴,۱ (۲)

۵,۲,۱ (۳)

۳,۲,۱ (۴)

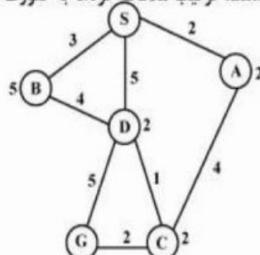
سوال کنکور سال ۹۵ مهندسی کامپیوتر

-9

گراف زیر را در نظر بگیرید:

گره S وضعیت شروع، گره G وضعیت هدف، اعداد کنار یالها هزینه عبور از آن یال و اعداد کنار گره‌ها تابع h را نشان می‌دهند. در صورت استفاده از روش جستجوی uniform cost search، ترتیب ملاقات گره‌ها به صورت

کدام یک از موارد زیر خواهد بود؟



S,A,B,D,G (۱)

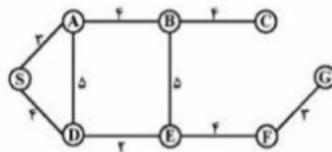
S,A,B,C,D,G (۲)

S,A,B,D,C,G (۳)

S,A,D,C,G (۴)

سوال کنکور سال 96 مهندسی کامپیوتر

در مسئله زیر برای رسیدن از S به G با استفاده از روش جستجوی هزینه یکنواخت (uniform cost search) در حالت جستجوی گرافی، کدام گره‌ها به ترتیب پیمایش می‌شوند؟



- SDEFG (۱)
- SADEFG (۲)
- SADEBFG (۳)
- SADEBCFG (۴)

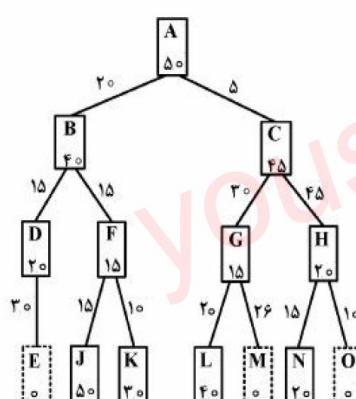
سوال کنکور سال 96 مهندسی کامپیوتر

در جستجوی درختی (tree-search) با استفاده از روش‌های جستجوی ناآگاهانه (uninformed) کدام مورد درست نیست؟

- (۱) حافظه لازم برای IDS بر حسب عمق، کم عمق‌ترین گره هدف خطی است.
- (۲) حافظه لازم برای BFS بر حسب عمق، کم عمق‌ترین گره هدف نمایی است.
- (۳) حافظه لازم برای DFS بر حسب عمق، کم عمق‌ترین گره هدف همیشه خطی است.
- (۴) حافظه لازم برای جستجوی دوچهتی (bidirectional) بر حسب عمق، کم عمق‌ترین گره هدف نمایی است.

درخت جستجوی زیر داده شده است. گره A، وضعیت اولیه می‌باشد. وضعیت‌های جواب نیز با مربع‌های نقطه‌چین نشان شده‌اند. اعداد روی یال‌ها هزینه استفاده از آن مسیر (یال) را نشان می‌دهد. اعداد داخل هر گره نیز تخمین فاصله تا هدف را مشخص می‌کند. اگر برای جستجو از روش *IDA* استفاده شود، میزان حد آستانه که جهت انتخاب گره‌ها، هنگام ورود به صفحه در نظر گرفته می‌شود پس از گرفتن مقدار اولیه، چند بار تغییر می‌کند؟

- (۱) ۰
- (۲) ۱
- (۳) ۲
- (۴) ۳



سوال کنکور سال 96 مهندسی کامپیوتر

با در نظر گرفتن الگوریتم جستجوی بهترین حریصانه (GBFS)، A^* و الگوریتم‌های

مشتق شده از آن: IDA^* و SMA^* چند مورد از گزینه‌های زیر درست است؟

الف) الگوریتم A^* همان GBFS است که ملاک انتخاب گره بعدی آن، به جای «نزدیک‌تر بودن فاصله تا هدف»، «کمینه بودن مجموع هزینه رسیدن به گره حاضر و هزینه رفتن از گره حاضر به هدف» می‌باشد.

ب) A^* گره‌هایی با تخمین هزینه‌ای کمتر از هزینه واقعی را نادیده گرفته و هرس می‌کند.

ج) IDA^* با الهام گرفتن از ایده عمیق‌کننده تکراری روی الگوریتم A^* کار می‌کند و هدف آن کاهش مشکل اصلی A^* یعنی پیچیدگی زمان توانی است.

د) SMA^* با الهام گرفتن از ایده «الگویتم بهترین جستجوی بازگشتی» روی A^* کار می‌کند و در صورتی که عمق سطحی ترین گره هدف کمتر از حافظه تخصیصی باشد کامل خواهد بود و اگر هدفی در دسترس باشد بهینه است.

۴)

۳)

۲)

۱)

102

سوال کنکور سال 96 مهندسی کامپیوتر

مسئله 8-puzzle را در نظر بگیرید. یک database داریم که به ازای هر ترکیب چهارتایی از اعداد ۱-۸، میانگین حل مسئله صرفاً آن چهار عدد تا جواب، در آن ذخیره شده است. به عنوانمثال $c(1,2,3,4)$ تعداد جایگزین مورد نیاز جهت جایه‌جا کردن صرفاً اعداد ۱ و ۲ و ۳ و ۴ تا محل‌های جواب است. آنگاه بهترین تابع admissible از بین گزینه‌ها کدام است؟

$$H(n) = \frac{(c(1,2,3,4) + c(5,6,7,8))}{2} \quad (1)$$

$$H(n) = \min(c(1,2,3,4), c(1,3,5,7)) \quad (2)$$

$$H(n) = \sqrt{c(1,2,3,4) + c(5,6,7,8)} \quad (3)$$

۴) نمی‌توان به قطعیت گفت کدام تابع بهتر است.

سوال کنکور سال 97 مهندسی کامپیوتر

در خصوص الگوریتم A^* در حالت استفاده از یک تابع ابتکاری سازگار (consistent) h (در صورتی که $h(n)$ هزینه مسیر طی شده تا گره n باشد)، کدام مورد نادرست است؟

۱) همواره مسیر بهینه به هدف را (در حالت جستجوی گرافی) پیدا می‌کند.

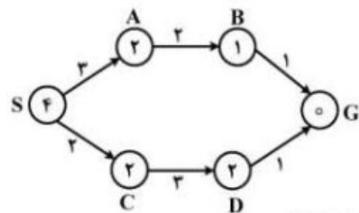
۲) ممکن است گره‌هایی را که مقدار $g(n) + h(n)$ آن‌ها بیشتر از طول مسیر بهینه است، گسترش دهد.

۳) ممکن است گره‌هایی را که مقدار $g(n)$ آن‌ها بیشتر از طول مسیر بهینه است، تولید کند (یعنی در صف بگذارد).

۴) ممکن است گره‌هایی را که مقدار $g(n) + h(n)$ آن‌ها بیشتر از طول مسیر بهینه است، تولید کند (یعنی در صف بگذارد).

سوال کنکور سال 97 مهندسی کامپیوتر

در شکل زیر هزینه گنش‌ها روی یال‌ها و مقدار تابع ابتكاری روی گره‌ها نوشته شده است. اگر S گره شروع و G گره هدف باشد، کدام مورد درست است؟ (در شرایط یکسان برای دو گره از ترتیب الفایی استفاده شود)



- (۱) تابع ابتكاری استفاده شده قابل قبول (admissible) است.
- (۲) ترتیب گسترش گره‌ها در الگوریتم A^* ، از چپ به راست S,C,A,B,G است.
- (۳) ترتیب تولید گره‌ها در الگوریتم A^* ، از چپ به راست S,A,C,B,D,G است.
- (۴) ترتیب تولید گره‌ها در الگوریتم UCS، از چپ به راست S,A,C,B,D,G است.

سوال کنکور سال 97 مهندسی کامپیوتر

فرض کنید در یک مسئله جستجو، فضای جستجو یک درخت محدود باشد که در آن هزینه هر یال یک عدد گویا است (هزینه‌ها می‌توانند منفی باشند). کدام عبارت در مورد یافتن مسیر بهینه توسط سه روش Breadth First Search و Uniform Cost Search و Depth First Search درست است؟

- (۱) هر سه روش، یافتن مسیر بهینه را برای مسئله گفته شده تضمین می‌کنند.
- (۲) فقط دو روش، یافتن مسیر بهینه را برای مسئله گفته شده تضمین می‌کنند.
- (۳) فقط یکی از این سه روش، یافتن مسیر بهینه را برای مسئله گفته شده تضمین می‌کند.
- (۴) هیجکدام از این سه روش، یافتن مسیر بهینه را برای مسئله گفته شده تضمین نمی‌کند.

۱۸_ اگر h_1 و h_2 دو تابع ابتكاری قابل قبول (admissible heuristic) باشند، کدام مورد درست است؟

- (۱) تابع $h = h_1 + h_2$ هم یک تابع قابل قبول است.
- (۲) اگر $\forall n (h_1(n) < h_2(n))$ باشد، ممکن است وضعیت شروعی وجود داشته باشد.
- (۳) تابع $h = \alpha h_1 + (1-\alpha)h_2$ به ازای $(0 < \alpha < 1)$ یک تابع قابل قبول است.
- (۴) تابع $h = \max(h_1, h_2)$ یک تابع سازگار (consistent) است.

در محیطی به شکل زیر که مستطیل $N \times M$ است، بعضی از خانه‌ها (خانه‌های طوسی رنگ) مسدود است و در برخی از خانه‌ها غذا وجود دارد. فرض کنید عامل از یک نقطه S شروع می‌کند و هدفش این است که خانه‌های حاوی غذا را ملاقات کند. در هر گنش، عامل می‌تواند در هر کدام از راستاهای بالا، پایین، چپ و راست تا جایی که به مانعی نرسیده به تعداد دلخواه حرکت کند. چنانچه مقصد عامل خانه‌ای باشد که در آن غذا وجود دارد. آن غذا ملاقات شده به حساب می‌آید. لازم به ذکر است که هر گنش عامل، مستقل از تعداد خانه‌ای که از آن‌ها عبور کرده، یک واحد هزینه دارد. در این خصوص کدام گزینه نادرست است؟

	غذا				
		غذا			
غذا				S	

- ۱) فاکتور انشعاب (branching factor) حداقل $M + N$ است.
- ۲) جستجوی سطح اول (BFS) می‌تواند جواب بهینه این مسئله را پیدا کند.
- ۳) تعداد غذاهای ملاقات نشده، یک تابع ابتکاری قابل قبول (admissible heuristic) است.
- ۴) تابع ابتکاری که مجموع فواصل منهتن (Manhattan distance) عامل تا خانه‌های حاوی غذاهای ملاقات نشده را نشان می‌دهد، یک تابع قابل قبول است.

سوال کنکور سال ۹۶ فناوری اطلاعات

در روش‌های جستجوی محلی برای پیدا کردن بیشینه یک تابع، کدام مورد درست است؟

- ۱) شبیه‌سازی ذوب هم مانند الگوریتم ژنتیک در هر دور یک جمعیت از وضعیت‌ها را نگه‌داری و بررسی می‌کند.

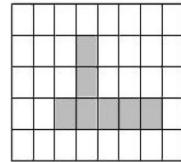
- ۲) تپه‌نوردی با k بار باز شروع تصادفی (random restart) به صورت موازی معادل با local beam search با جمعیت k تایی است.

- ۳) local beam search اگر با جمعیتی با k عضو که یکی از آنها s است شروع کند نسبت به تپه‌نوردی (hill climbing) که از s شروع شده هیچ وقت جواب نهاییش بدتر نیست.

- ۴) روش شبیه‌سازی ذوب (simulated annealing) وقتی دما به سمت صفر می‌رود به تپه‌نوردی به صورت اولین انتخاب (first choice) (که اولین همسایه‌ای از گره فعلی پیدا کرد به آن می‌رود) میل می‌کند.

۱۱۹- فرض کنید دو ربات در دو نقطه A و B از نقشه‌ای مستطیلی (مانند نمونه زیر) قرار دارند و در هر دور هر کدام از این ربات‌ها می‌تواند به یکی از خانه‌های بالا، پایین، چپ و راست در صورتی که مسدود نباشد، بروند و همچنین ربات‌ها همزمان با هم دیگر می‌توانند حرکت کنند. می‌خواهیم بهترین راه حل را پیدا کنیم که در آن دو ربات در کمترین زمان به یک خانه یکسان برسند. (در صورت مسئله محدود نشده که کدام خانه باشد) کدام یک از موارد زیر یک تابع ابتکاری admissible برای حل این مسئله است؟

فرض کنید $d_M(U, V)$ فاصله منتهن دو نقطه U و V را در محیط نشان می‌دهد. همچنین G کل مجموعه نقاط غیرمسدود نقشه را مشخص می‌کند.



$$\min_{C \in G} \max(d_M(A, C), d_M(B, C)) \quad (1)$$

$$\min_{C \in G} (d_M(A, C) + d_M(B, C)) \quad (2)$$

$$\max_{C \in G} \min(d_M(A, C), d_M(B, C)) \quad (3)$$

۱۲۲- اگر h_1 و h_2 توابعی سازگار (consistent) باشند، کدام عبارت نادرست است؟

$$0 \leq a \leq 1 \quad (1)$$

$$ah_1 + (1-a)h_2 \quad (2)$$

$$\frac{\min(h_1, h_2) + \max(h_1, h_2)}{2} \quad (3)$$

سازگار است.

$$ah_1 \quad (4)$$

سازگار است.

۱۲۴- در خصوص روش‌های جستجوی محلی (local search) کدام جمله درست است؟

۱) در روش local beam search ممکن نیست همه اعضای جمعیت در یک زمان t^* (از نظر تابع هدف) بدتر از همه اعضای جمعیت در یکی از زمان‌های قبلی $t' < t^*$ باشند.

۲) در روش local beam search چنانچه کنش‌ها برگشت‌پذیر باشند، اعضای جمعیت در طول زمان بهبود پیدا می‌کنند (یا حداقل تنزل پیدا نمی‌کنند).

۳) با نقطه شروع برابر ممکن نیست hill-climbing به نتیجه بهتری نسبت به (simulated Annealing) برسد.

۴) در الگوریتم hill-climbing ممکن است در انتهای نقطه‌ای بدتر از نقطه شروع بررسیم.

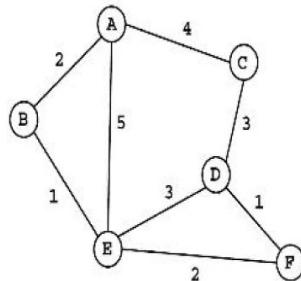
حلقه‌ی ۹۹

24

۱۲۷- در گراف زیر، گره A وضعیت شروع و گره F وضعیت هدف است. اگر تعداد یال‌های باقی‌مانده تا هدف را به عنوان مقدار تابع اکتشافی (h') هر گره در نظر بگیریم، در کدام‌یک از دو روش جستجوی Uniform Cost Search و

Mlavat (Visit) خواهد شد؟

هزینه عمور از هر یال کنار آن نوشته شده است. فرض کنید که هر گره حداقل یک مرتبه ملاقات می‌شود.



۱) فقط در روش USC

۲) فقط در روش A*

۳) در هر دو روش

۴) در هیچ کدام از دو روش

۹۹-۱۷

25

در مورد مسائل جستجویی که در آن‌ها هزینه کنش‌ها برابر یک و هزینه مسیر بهینه برابر ۰ و فاکتور انشعاب

(branching factor) برابر b است، کدام گزینه درست است؟

۱) مرتبه حافظه DFS با محدودیت عمق ۱ برابر (bd) است.

۲) روش DFS با محدودیت عمق ۱ در حالتی که $d = 1$ است، یک روش بهینه است.

۳) روش DFS با محدودیت عمق ۱ در حالتی که $d < 1$ است، یک روش کامل (complete) است.

۴) زمان روش DFS با محدودیت عمق $d < 1$ در تحلیل بدترین حالت (worst case) برابر با زمان BFS است.

۹۹-۱۷

26

در مورد روش A* که در آن $f(n) = g(n) + h(n)$ در نظر گرفته می‌شود و هزینه همه کنش‌ها بزرگتر از صفر است، کدام مورد نادرست است؟

۱) اگر h تابعی consistent باشد، مقدار f گره جدیدی که به صفت اضافه می‌شود نمی‌تواند از f همه گره‌های موجود در صفت کمتر باشد.

۲) اگر h تابعی admissible باشد، برای گرهی که برای گسترش انتخاب می‌شود، مسیر بهینه تا آن گره بدست آمده است.

۳) اگر h تابعی consistent باشد و مقدار f گرهی از مقدار f همه گره‌های موجود در صفت بزرگتر باشد ممکن است آن گره به صفت اضافه شود.

۴) اگر h تابعی admissible باشد، ممکن است مقدار f گره‌هایی که در ادامه در صفت قرار می‌گیرند کمتر از مقدار f گره‌های موجود در صفت باشد.