سید علی علم بلادی

سوالات میان ترم

---

1)سیستمی که مانندانسان رفتارمی کندراباذکرمثال توضیح دهید؟

هنرساخت ماشینهایی که کارهایی انجام میدهندکه آن کارها فعلاتوسط انسان بافکرکردن انجام میشود.
مطالعه برای ساخت کامپیوترهایی که کارهایی راانجام دهندکه فعلاانسان آنهارابهترانجام میدهند.
دتست تورینگ مثالی مناسب برای این سیستم است.دراین تست کامپیوترتوسط فردی محققموردآزمایش قرارمیگیرد،به طوری که این فرددورازکامپیوترقراردارد،کامپیوتربه پرسش های مطرح شده پاسخ میدهد.کامپیوتروقتی ازاین تست عبورمیکندکه این شخص نتواندتشخیص دهد که پاسخ دهنده یک انسان است یاچیزدیگر.این تستباید قابلیت هایی نظیر(پردازش زبان طبیعی،بازنمایی دانش،استدلال خودکار،یادگیری ماشین،بینایی کامپیوتر،دانش روباتیک)داشته باشد.

---

2)هدف ازتفکرعاقلانه چیست وچه آورده ای درپی خواهدداشت؟

عاقلانه فکرکردن ،به معنایی ساخت الگوهایی برای ساختارهای استدلالی است.درواقع عاقلانه فکرکردن یعنی مطالعه ی توانایی های ذهنی ازطریق مدلهای محاسباتی(منطق گرایی)عاقلانه فکرکردن مطالعه ی محاسباتی است که منجذبه درک واستدلال شود.

عاقلانه تفکرکردن رسم منطق گرایی درهوش مصنوعیبرای ساخت سیستمهای هوشمنداست.درواقع برنامه هایی نوشته میشوندکه میتوانندمسائل قابل حلی که درنمادگذاری منطقیتوصیف میشوند راحل کنند.

موانع اصلی این تفکر:1)دریافت دانش غیررسمی وتبدیل آن به دانش رسمی 2)تفاوت میان قادربه حل مسئله بودن درتئوری ودرعمل(بن بست محاسباتی)

---

3)اجزای عامل ووظیفه عامل رابارسم شکل وتابع نویسی بررسی کنید؟

عامل هرچیزی است که قادراست محیط خودرا ازطریق حسگرهادرک کندوازطریق محرک هاعمل کند.

به عنوان مثال عامل روباتیک شامل دوربینهایی به عنوان سنسوریاحسگر.موتورهای متعددی به عنوان محرم است.یاعامل انسان دارای چشم وگوش واعضای دیگربرای حس کردن ودست وپاودهان واعضای دیگربه عنوان محرک است.

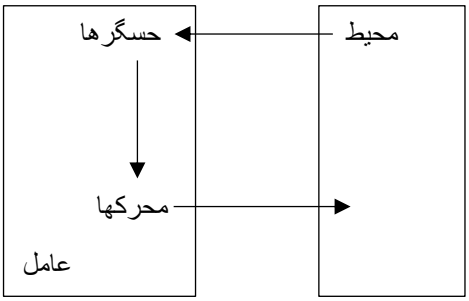
عاملهاازطریق حسگرهاومحرکهابامحیطدرتعامل هستند.

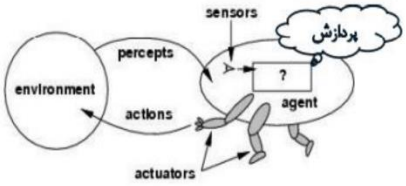
سنسوروظیفه دریافت مشخصه هایی ازمحیط رادارد‌ومحرک وظیفه انجام اعمال برروی محیط رادارد.

عامل وظیفه دارد رشته دریافتهای ورودی رابه دنباله ای ازاعمال نگاشت کند.بنابراین میتوان گفت عامل میتواندمانندتابع عمل کند.

$F : P^* \rightarrow A$

که A اعمال وPرشته دریافت هااست.عامل میتوانداعمال محیط خودرادرک کند،اماتأثیرآنهابرروی محیط همیشه قابل پیش بینی نیست.





---

4)PEASرابرای ربات فضانورد وفوتبالیست تشریح کنید؟

ربات فوتبالیست

معیار کارایی : برد بازی – رعایت قوانین – سرعت عمل مناسب

محیط : زمین چمن – زمین خاکی – سالن ورزشی – زمین آسفالت – تیم خود – توپ – تیم حری ف

عملگر : پاس دادن – گل زدن – حمله – دفاع

سنسو ر : سرعت سنج – فاصله یاب – بازوهای محرک – سنسور رو به عقب – سنسور رو به جلو – موقعیت یاب

ربات فضانور د

معیار کارایی : دسته بندی صحیح تصاویر - کمترین هزینه – سرعت عمل مناسب - ایمن ی

محیط : محل آزمایشگاه – فضا

عملگر : نمایش تصاویر طبقه بندی شده – تشخیص ها – نمایشگر

سنسور : آرایه ای از پیکسل های رنگی – دوربین – سونار)مکان یاب صوتی(- حسگر دما – حسگر فشار هوا – حسگر
های شیمیای

---

5) طبق شبه کد زیر چرا عامل مبتی بر جدول به شکست مواجه می شود ؟ راهکار های پیشنهادی خود را نام برده و مختصری در خصوص هر کدام توضیح دهید ؟

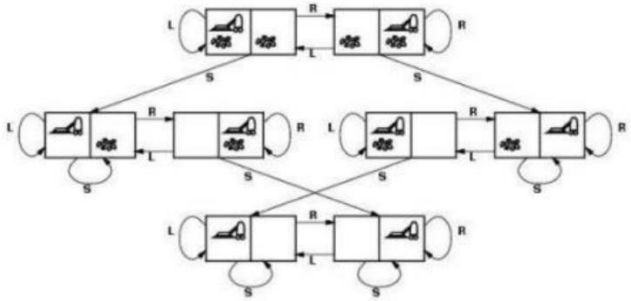
```
function TABLE-DRIVEN-AGENT(percept) returns an action
  persistent: percepts, a sequence, initially empty
              table, a table of actions, indexed by percept sequences, initially fully specified

  append percept to the end of percepts
  action ← LOOKUP(percepts, table)
  return action
```

این برنامه یک برنامه عامل ساده است که دنباله ی ادراک راردیابی کرده ووازآن به عنوان شاخصی درجدول فعالیتها استفاده می کندتاتصمیم بگیردچه کاری بایدانجام دهد.

برای ساخت عامل خردمندمابایدجدولی بسازیم که برای هردنباله ی ادراک ممکن،دارای فعالیتهای مناسبی باشد.
رهیافت جدولی برای ساخت عامل باشکست مواجه میشودچون ما به ازای مجموعه ای ادراکات ممکن وتعدادکل ادراکاتی که عامل دریافت میکند برای جدول جستجوبایدتعدادزیادی درایه داشته باشیم که امکان پذیرنیست.اندازه این جدول بیان میکندکه(1:هیچ عامل فیزیکی دراین دنیانمیتواندفضایی برای ذخیره این جدول داشته باشد.2:طراح برای ایجادجدول زمان مناسب ندارد.3:هیچ عاملی نمیتواندتمام درایه های جدول رادرتجربه خودبه کارگیرد.4:حتی اگرمحیط خیلی ساده باشد که اندازه جدول قابل تحمل باشد طراح نمیداند درایه های جدول راچگونه ذخیره کند.)
شبه کدبالایک تابع عامل مطلوب راپیاده سازی می کند،مابایدبدانیم چگونه یکبرنامه ای بنویسیم که بااستفاده ازیک کدکوچک به جای جدول بزرگ رفتارعقلایی راانجام دهد.

---

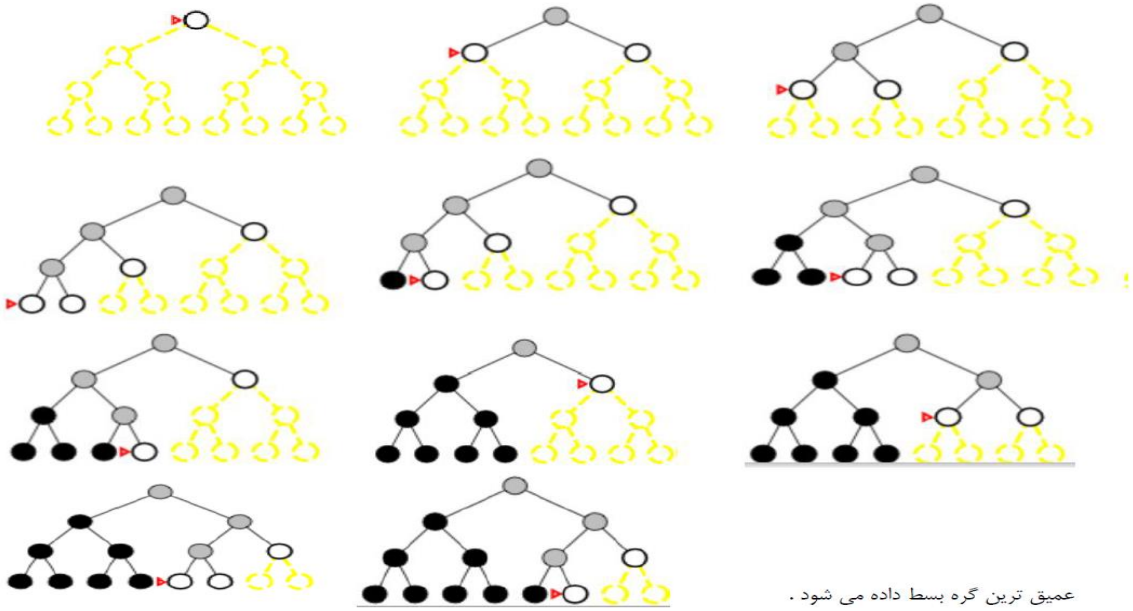
6)دنیای جاروبرقی راباتوجه به فرموله سازی مسئله تشریح کنید؟



حالتها(states):حالت به وسیله مکان عامل ومکانهای کثیف تعیین میشود.عامل دریکی ازدومکان است که هرکدام ممکن است کثیف باشندیانباشند.پس8=2^2*2 حالت وجوددارد.یعنی(n^2.n)حالت.
حالت شروع:هرحالتی میتواندبه عنوان حالت شروع باشد.
فعالیتها(action):دراین محیط جاروبرقی فقط سه فعالیت میتواندانجام دهد:1)حرکت به سمت چپ2)حرکت به سمت راست 3)عمل مکش
آزمون هدف:بررسی میکنذآیاتمام مکانهاتمیزاست یاخیر.
هزینه ی مسیر:تعدادمراحل موجوددرمسیر،هزینه ی مسیراست.

---

7)جست و جوی عمقی را با رسم مرحله به مرحله شرح دهید و در نهایت کارایی الگوریتم را با چهار معیار اندازه گیری بیان کنید ؟



عمیق ترین گره بسط داده می شود .

جستجوی عمقی ،عمیق ترین گره رابسط میدهد،جستجوازعمیق ترین سطح درخت جستجوادامه می یابد،وقتیوقتی گره ها بسط داده شدند ازمرزحذف میشوندوجستجوبه عمیق ترین گره بعدی برمی گردد.جستجوی عمقی ازصف LIFO استفاده میکند.دراین صف جدیدترین گره تولیدشده،برای بسط دادن انتخاب میشود،این گره بایدعمیق ترین گره بسط نداده شده باشد.

جستجوی عمقی:

کامل بودن : خیر ، مگر اینکه فضای حالت محدود باشد و حلقه تکرار وجود نداشته باشد.

بهینه بودن : خیر ، چون کامل نیست.

پیچیدگی زمانی(O(b^m ، اگر m خیلی بزرگتر از d باشد به مراتب بدتر است / در بسیاری از مسائل سریعتر از جست وجوی BF است.

پیچیدگی حافظه O(bm+1) : ، در زمان عقبگرد حافظه آزاد می شود .

---

8) ضمن بررسی الگوریتم جستجوی درختی شبه کد زیر را بررسی کنید که استراتژی در کدام از 4 توابع ، پیاده سازی شده است ، توابع را نام برده و عملکرد هر یک را بیان کنید ؟

```
function TREE-SEARCH(problem, fringe) return a solution or failure
    fringe ← INSERT(MAKE-NODE(INITIAL-STATE[problem]), fringe)
    loop do
        if EMPTY?(fringe) then return failure
        node ← REMOVE-FIRST(fringe)
        if      GOAL-TEST[node] then
                return SOLUTION(node)
        else
                fringe ← INSERT-ALL(EXPAND(node, problem), fringe)
```

درالگوریتم جستجوی درختی ،حالت شروع درریشه درخت قرارمی گیرد،انشعابها،فعالیتها و گره ها،حالتهای موجودهستند.ابتداریشه رابررسی میکنیم که ایاحالتهدف است یاخیر درصورتی که حالت هدف نبودان رابسط میدهیم تامجموعه ی جدیدی ازحالتهابه وجودآید،بعدازآن حالتهارایکی یکی بررسی کرده تازمانی که به آخرین گره برسیم که هیچ فرزندی ندارد.پس سراغ گره ها میرویم ویکی یکی بررسی

میکنیم پس ازآن گره هایی که مارا به هدف نمیرساندحذف میکنیم واین روش ادامه پیدامیکندتابه هدف برسیم.
استراتژی های متفاوتی برای رسیدن به حالت هدف وجوددارد.استراتژی مادراینجااین است که یک گره کاندید رابررسی کن اگر هدف نبود آن رابسط بده،آنقدراین کارراتکرار کن تا به هدف برسی.
تابع(remove first):اولین خانه را fringeمیکند
تابع(goal test):آیابه هدف رسیدیم؟خیر.یک گره باتوجه به استراتژی انتخاب کن
تابع(expand):وقتی به هدف نرسیدیم گره هارابسط بده.
تابع(insert):گره های فرزندرادرfringبسط بده ونتایج رابه جستجواضافه کن.
استراتژی در تابع insert پیاده سازی شده است.

---

9) شبه کد زیر مربوط به کدام جست و جوی ناآگاهانه می باشد ، از مزایای کدام جست و جو های دیگر بهره برده است ، با ترسیم شکل توضیح دهید ؟

```
function ITERATIVE-DEEPENING-SEARCH(problem) returns a solution, or failure
  for depth = 0 to ∞ do
    result ← DEPTH-LIMITED-SEARCH(problem, depth)
    if result ≠ cutoff then return result
```

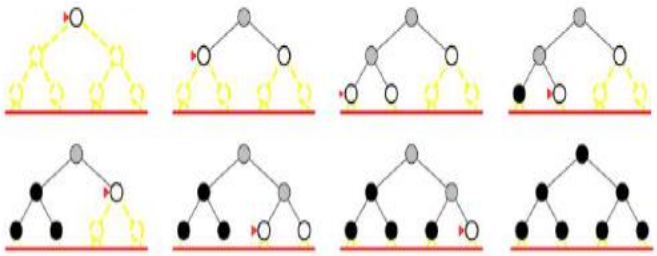
این شبه کدمربوط به جستجوی عمقی تکرارشونده است،که این الگوریتم ازلحاظ زمانی ازمرتبه جستجوی اول سطحی است و ازلحاظ پیچیدگی حافظه ازمرتبه جستجوی اول عمق بهره میبرد.
جست و جوی عمقی تکراری ، یک استراتژی کلی است . این الگوریتم با شروع از مقدار صفر به عنوان عمق محدود ، مقدارآن را به تدیج اضافه می کند مانند یک و .. تا ایکه هدفی پیدا شود . هدف وقتی پیدا می شود که عمق محدود به d برسد ،که d عمق مربوط به عمیق ترین گره هدف است . این الگوریتم از مزایای جست و جوی عمقی و جست وجوی عرضی استفاده می کند فواید مربوط به این دو الگوریتم را با هم ترکیب می کند . این الگوریتم برای تعیین عمق محدود است که جست و جو با عمق محدود را با حدود صعودی تکرار می کند و زمانی خاتمه می یابد که جوابی پیدا شود یا جست و جو با عمق محدود مقدار failure را برگرداند که این عمل نشان می دهد جوابی وجود ندارد.

شکل :

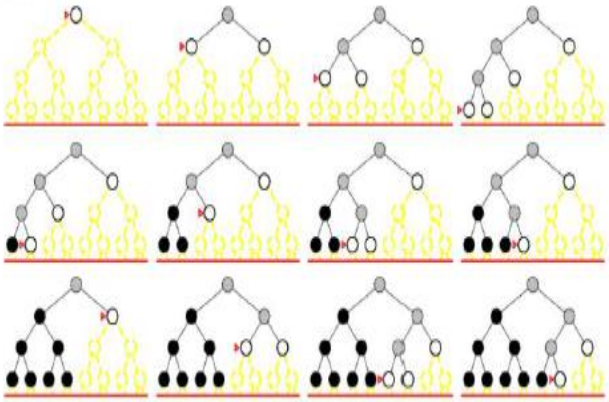
- Limit=0

- Limit=1

- Limit=2



- Limit=3



10)شش نوع جست و جو های ناآگاهانه جدول زیر را به تفکیک ، با چهار معیار مربوطه به اختصار شرح دهید ؟

Criterion	Breadth-First	Uniform-cost	Depth-First	Depth-limited	Iterative deepening	Bidirectional search
Complete?	YES*	YES*	NO	YES, if $l \geq d$	YES	YES*
Time	b^{d+1}	$b^{C^*/\varepsilon}$	b^m	b^l	b^d	$b^{d/2}$
Space	b^{d+1}	$b^{C^*/\varepsilon}$	bm	bl	bd	$b^{d/2}$
Optimal?	YES*	YES*	NO	NO	YES	YES

1)جست و جوی سطحی

کامل بودن : بله / شرط : جواب بهینه در عمق d قابل دسترس باشد . فاکتور انشعاب b محدود باشد.

بهینه بودن : بله / شرط : مسیر ها فاقد هزینه باشند.

پیچیدگی زمانی : گره ریشه حداکثر دارای b فرزند است / هر فرزند نیز حداکثر دارای b فرزند است بنابراین در سطح

دوم b 2گره وجود دارد / با فرض اینکه جواب در عمق d باشد در بدترین حالت جواب باید در سمت راست ترین گره باشد

/تعداد نود های تولید شده از رابطه زیر محاسبه می شود.

$$b^0 + b^1 + b^2 + b^3 + \ldots + b^d + (b^{d+1} - b) = O(b^{d+1})$$

$$b^{d+1} - b = O(b^{d+1})$$

پیچیدگی حافظه : هم مرتبه پیچیدگی زمانی است.

2)جست و جو با هزینه یکنواخت

کامل بودن : بله / شرط : جواب در عمق قابل دسترس باشد . هزینه ها مقدار مثبت داشته باشند.

بهینه بودن : بله / شرط : کامل باشد.

پیچیدگی زمانی : فرض شود c* هزینه مسیر بهینه است . فرض شود هزینه هر عمل حداقل e است . در بدترین حالت

o(b^c*e)پیچیدگی زمانی است.

پیچیدگی حافظه : هم مرتبه پیچیدگی زمانی است.

3)جست و جوی عمق ی

کامل بودن : خی ر / شرط : مگر اینکه فضای حالت محدود باشد و حلقه تکرار وجود نداشته باشد.

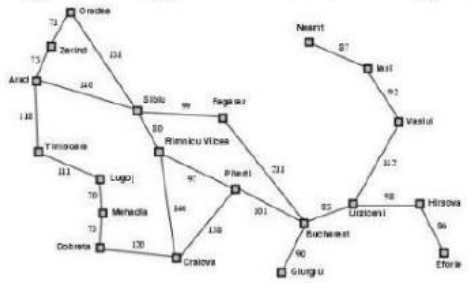
بهینه بودن : خیر / زیرا کامل نیست.

پیچیدگی زمانی:(ob^m)است، اگر m خیلی بزرگتر از d باشد به مراتب بدتر است . در بسیاری از مسائل سریعتر از جست
و جوی BF است.
پیچیدگی حافظه:(O(bm+1 در زمان عقبگرد حافظه آزاد می شود.
4)جست و جوی عمقی محدود
در حقیقت DF با عمق محدود L است.
تعیین در همه مسائل امکان پذیر نمی باشد.
اگر L<d آنگاه غیر کامل است.
اگر L>d آنگاه کامل اما غیر بهینه است.
اگر L=d آنگاه کامل و بهینه است.
پیچیدگی زمانی : O(b1)
پیچیدگی حافظه:O(bl)
5)جست و جوی عمق ی تکراری
کامل بودن : بله / شرط : حلقه تکرار وجود نداشته باشد.
بهینه بودن : بله / اگر هزینه مسیر ها با هم برابر باشد.
پیچیدگی زمانی :O(b^d)
پیچیدگی حافظه :O(bd)
6)جست و جوی دو طرفه
کامل بودن : بله / شرط : استفاده کردن از جست و جوی سطری
بهینه بودن : بل ه / شرط : استفاده کردن از جست و جوی سطری
پیچیدگی زمانی:O(b^d/2)
پیچیدگی حافظه:O(b^d/2)

---

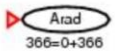
11)جست وجوی *A را با توجه به جدول h SLD با جست و جوی حریصانه Greedy search با رسم درختی به طور کامل توضیح داده و تفاوت ها را با دلیل ذکر کنید ؟

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Dobreta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Hirsova	151	Urziceni	80
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374

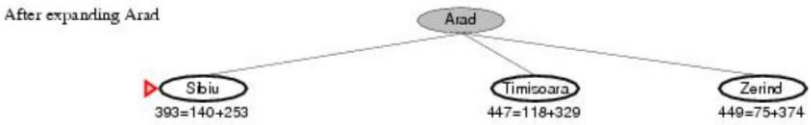


دراین روش گره هارا باترکیب(g(n یعنی هزینه رسیدن به گره و(h(n یعنی هزینه رسیدن ازاین گره به گره هدف ارزیابی می کند.
F(n)=g(n)+h(n) یعنی (f(n هزینه برآوردشده ی ارزانترین جوار ازطریق n است.پس باید به گره ای فکرکنیم که کمترین (g(nو(h(n راداشته باشد.
شناخته شده ترین جستجوی آگاهانه
- ایده: از بسط گرههایی که به صرفه به نظر نمیرسند، اجتناب میکند.
- تابع ارزیابی **f(n)= g(n) +h(n)** :
- **g(n)** هزینه واقعی از گره شروع تا گره**n**
- **h(n)** هزینه تخمینی از گره **n** تا هدف
- **f(n)** هزینه تخمینی از گره شروع تا هدف با عبور از گره**n**

جستجوی*A کامل وبهینه وبهینه موثراست.مرتبه زمانی ومکانی آن نمایی است.

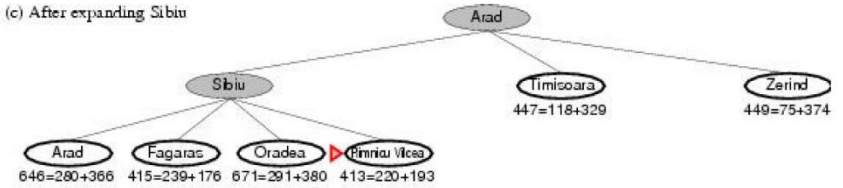


f(Arad) = c(??,Arad)+h(Arad)=0+366=366



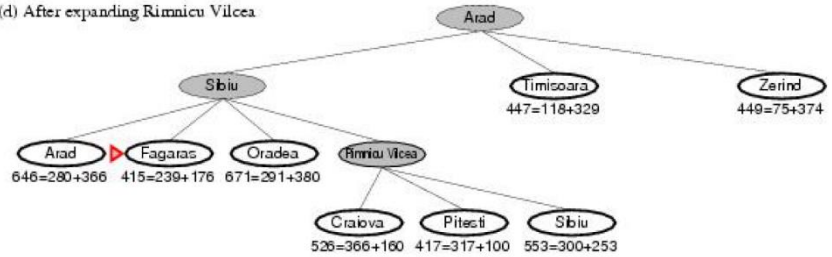
f(Sibiu)=c(Arad,Sibiu)+h(Sibiu)=140+253=393
f(Timisoara)=c(Arad,Timisoara)+h(Timisoara)=118+329=447
f(Zerind)=c(Arad,Zerind)+h(Zerind)=75+374=449

(c) After expanding Sibiu



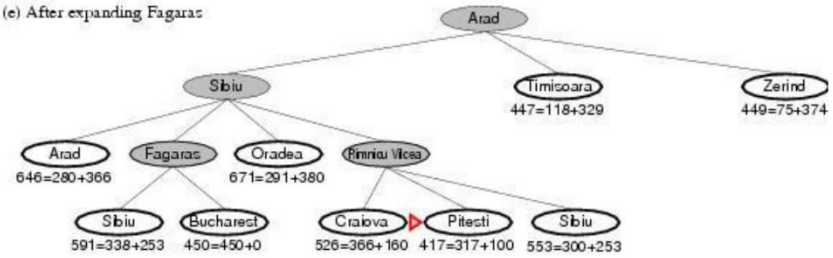
f(Arad)=c(Sibiu,Arad)+h(Arad)=280+366=646
f(Fagaras)=c(Sibiu,Fagaras)+h(Fagaras)=239+179=415
f(Oradea)=c(Sibiu,Oradea)+h(Oradea)=291+380=671
f(Rimnicu Vilcea)=c(Sibiu,Rimnicu Vilcea)+h(Rimnicu Vilcea)=220+192=413

(d) After expanding Rimnicu Vilcea



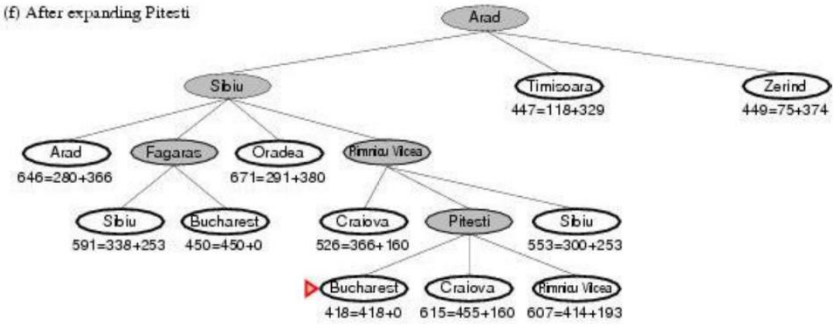
f(Craiova)=c(Rimnicu Vilcea, Craiova)+h(Craiova)=360+160=526
f(Pitesti)=c(Rimnicu Vilcea, Pitesti)+h(Pitesti)=317+100=417
f(Sibiu)=c(Rimnicu Vilcea,Sibiu)+h(Sibiu)=300+253=553

(e) After expanding Fagaras



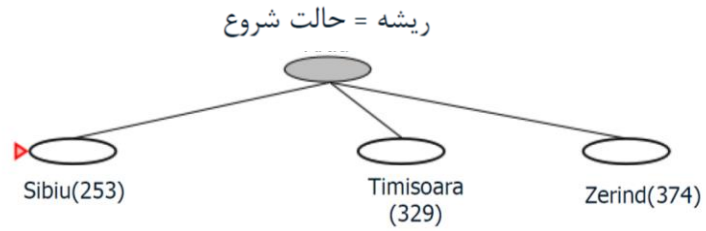
f(Sibiu)=c(Fagaras, Sibiu)+h(Sibiu)=338+253=591
f(Bucharest)=c(Fagaras,Bucharest)+h(Bucharest)=450+0=450

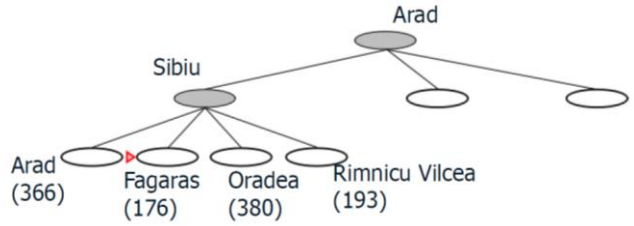
(f) After expanding Pitesti

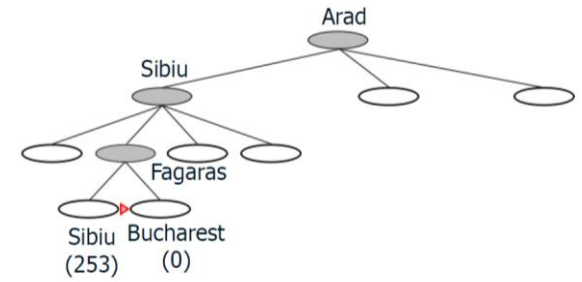


f(Bucharest)=c(Pitesti,Bucharest)+h(Bucharest)=418+0=418

جستجوی حریصانه:(n) =h(n) =f : گره ایی را بسط میدهدکه به هدف نزدیکتر باشد.
این جستجوکامل نیست چون حلقه تکراردارد وبهینه هم نیست ومرتبه زمانی ومکانی ان)(O(b^m است.







تفاوت الگوریتم حریصانه *Aدر(n)g یعنی هزینه واقعی است.*Aجستجورابهینه وکامل میکند.جستجوی حریصامه زودتصمیم می گیرد،امادر*A مینیمم ترین گره انتخاب شده وبه آن مینیمم هزینه واقعی اعتمادمیکند.

---

12) الگوریتم زیر را شرح دهید و با توجه به جدول و شکل سوال 11 با رسم درخت جست و جوتوضیح دهید ؟

Recursive best-first search

```
function RECURSIVE-BEST-FIRST-SEARCH(problem) return a solution or failure
    return RFBS(problem,MAKE-NODE(INITIAL-STATE[problem]),∞)

function RFBS( problem, node, f_limit) return a solution or failure and a new f-cost limit
    if GOAL-TEST[problem](STATE[node]) then return node
    successors ← EXPAND(node, problem)
    if successors is empty then return failure, ∞
    for each s in successors do
        f[s] ← max(g(s) + h(s), f[node])
    repeat
        best ← the lowest f-value node in successors
        if f[best] > f_limit then return failure, f[best]
        alternative ← the second lowest f-value among successors
        result, f[best] ← RBFS(problem, best, min(f_limit, alternative))
        if result ≠ failure then return result
```

این الگوریتم RBFSاست که درآن :

1)بهترین گره برگ و بهترین جانشین برای آن انتخاب شود.

2)اگر مقدار بهترین گره برگ از جانشین آن بیشتر شد، آنگاه به مسیر جانشین عقبگرد شود.

3)در حین عقبگرد، مقدار **f(n)** بروزرسانی شود.

4)گره جانشین بسط داده شود.

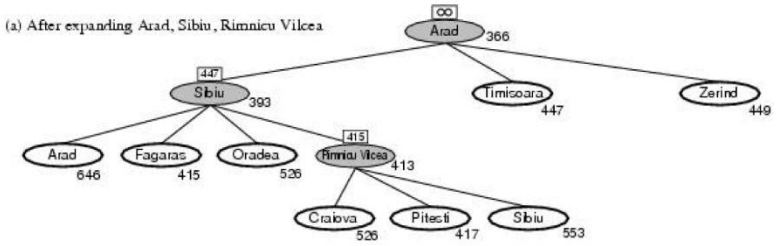
RBFSجستجوی به مراتب موثرتری از **ID A*** است.

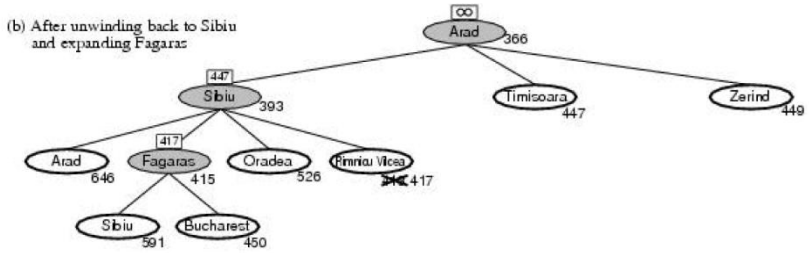
از تولید تعداد بسیار زیادی گره به دلیل **تغییر عقیده** رنج می برد.

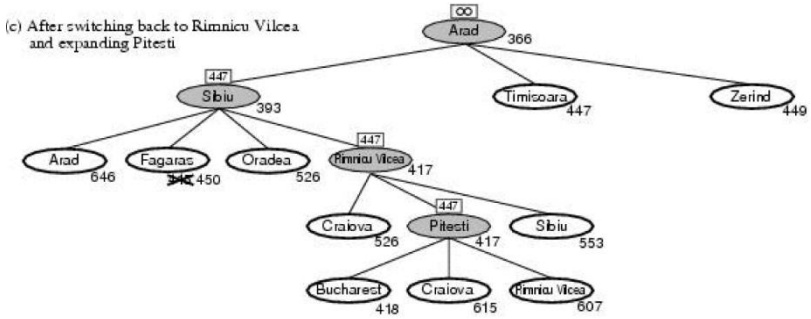
مانند **A*** اگر **h(n)** قابل پذیرش باشد، بهینه است.

پیچیدگی حافظه **o(bd)** است.

پیچیدگی زمانی به کیفیت تابع هیوریستیک و میزان تغییر عقیده بستگی دارد.







---

13)) چند نوع تابع هیوریستیک را می توان برای پازل اعداد معرفی کرد ، با رسم شکل بررسی کنید ؟

- h_1 تعداد کاشی هایی که سرجای خود نمی‌باشند.

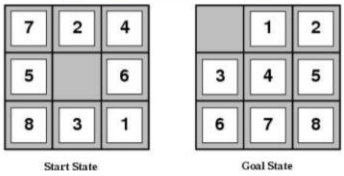
$h_1(s)=8$

- h_2 مجموع فاصله افقی – عمودی (منهتن) هر کاشی تا جای واقعی

$h_2(s)=3+1+2+2+2+3+3+2=18$

- h_3 مجموع فاصله افقی – عمودی – قطری هر کاشی تا جای واقعی

$h_3(s)=2+1+1+1+2+2+2+2=13$



تابع هیوریستیک قابل پذیرش1

- از طریق نسخه ساده شده از مساله(**relax version**

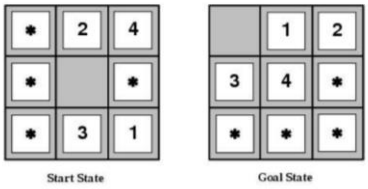
h۱.هر کاشی می تواند به هرجایی منتقل شود

h۲ ...هر کاشی می تواند به هر خانه همسایه منتقل شود.

ABSolver..هزینه راه حل برای مکعب روبیک را تخمین میزند.

ابداع تابع هیوریستیک قابل پذیرش (۲)

از طریق نسخه کوچتر از مساله (subproblem)



ابداع تابع هیوریستیک قابل پذیرش(3)

- از طریق یادگیری از تجربه(**learning from experience experience**)

تجربه : حل تعداد بسیار زیادی از مساله

---

14)سه راه حل جهت ابداع تابع هیوریستیک نام برده و شرح دهید ؟

1)از طریق نسخه ساده شده از مساله

H 1هر کاشی می تواند به هر جایی منتقل شود.

H 2هر کاشی می تواند به هر خانه همسایه منتقل شود.

ABSoloverهزینه راه حل برای مکعب روبیک را تخمین می زند.

2)از طریق نسخه کوچکتر از مساله

3)از طریق یادگیری از تجربه

تجربه : حل تعداد بسیار زیادی از مساله

---

15)انواع جست و جوی محلی را نام برده و ایده هر یک را بیان کنید ؟

جست و جوی تپه نوردی ، SA، پرتو محلی ، ژنتیک

الگوریتم جست و جوی محلی تپه نوردی : این الگوریتم حلقه ای است که در جهت افزایش مقدار حرکت می کند (به طرف بالای تپه) . وقتی به قله ای رسید که هیچ همسایه ای از آن بلند تر نیست خاتمه می یابد.

الگوریتم جست و جوی محلیSA :این الگوریتم نسخه ای از تپه نوردی اتفاقی است و پایین آمدن از تپه مجاز است. حرکت به طرف پایین و به آسانی در اوایل زمانبندی annealing پذیرفته شده و با گذشت طمان کمتر اتفاق می افتد.

الگوریتم جست و جوی پرتو محلی : نگهدار ی فقط یک گره در حافظه ، واکنش افراطی نسبت به مسئله محدودیت حافظه است . این الگوریتم به جای بک حالت ، kحالت را نگهداری می کند . این الگوریتم با kحالت که به طور تصادفی تولیدشدند ، شروع می کند . در هر مرحله تمام پسین های همه حالت ها تولید می شوند . اگر یکی از آن ها هدف بود ، الگوریتم متوقف می شود ؛ وگرنه بهترین پسین را انتخاب و عمل را تکرار می کند.

الگوریتم جست و جوی محلی ژنتیک : این الگوریتم شکلی از جست و جوی پرتو اتفاقی است که در آن ، حالت های پسین از طریق ترکیب دو حالت والد تولید می شوند . در مقایسه با انتخاب طبیعی ، مثل جست

و جوی پرتو اتفاقی است ، با این تفاوت که اینجا با تولید مثل جنسی سروکار داریم نه غیر جنسی . این الگوریتم همانند جست و جوی پرتو محلی ، با مجموعه ای از k حالت که به طور تصادفی تولید شدند شروع می کند که به آن جعیت گفته می شو د.

---

16)الگوریتم زیر را شرح داده و انواع آن را نام برده و بررسی کنید ؟

```
function HILL-CLIMBING( problem) return a state that is a local maximum
    input: problem, a problem
    local variables: current, a node.
                     neighbor, a node.

    current ← MAKE-NODE(INITIAL-STATE[problem])
    loop do
        neighbor ← a highest valued successor of current
        if VALUE [neighbor] ≤ VALUE[current] then return STATE[current]
        current ← neighbor
```

الگوریتم بالا مربوط به الگوریتم جست و جوی محلی تپه نوردی می باشد . این الگوریتم حلقه ای است که در جهت افزایش مقدار حرکت می کند (به طرف بالای تپه). وقتی به قله ای رسید که هیچ همسایه ای از آن بلند تر نیست خاتمه می یابد.
در این الگوریتم درخت جست و جو را نگهداری نمی کند . لذا ساختمان داده گره فعلی فقط باید حالت و مقدار تابع هدف رانگهداری کند . تپه نوردی به همسایه های حالت فعلی نگاه می کند . مثل تلاش برای یافتن قله کوه اورست در مه گرفتگی غلیظ ، در حالی که دچار فراموشی هستید . تپه نوردی گاهی جست و جوی محلی حریصانه نام دارد زیرا بدون اینکه قبلا فکر کند به کجا برود ، حالت همسایه خوبی را انتخاب می کند . تپه نوردی معمولا به سرعت به جواب پیش می رود ، زیرا به راحتی می تواند حالت بد را بهبود ببخشد.

انواع تپه نوردی ؛

تپه نوردی غیر قطعی : در بین حرکت های رو به بالا یکی به صورت تصادفی انتخاب می شود . البته احتمال انتخاب با شیب متناسب است.

تپه نوردی با انتخاب اولین گزینه : گره ها تا حصول یک گره بهتر بسط داده می شوند.

تپه نوردی تصادفی : از حالت شروع مجدد تصادفی تا حصول جواب مجددا شروع خواهد نمود.
