بسمه تعالی

هوش مصنوعی

حل مسئله ۲-

نیمسال اوّل ۰۲-۱۴۰۱

دکتر مازیار پالهنگ

آزمایشگاه هوش مصنوعی

دانشکدهٔ مهندسی برق و کامپیوتر

دانشگاه صنعتی اصفهان

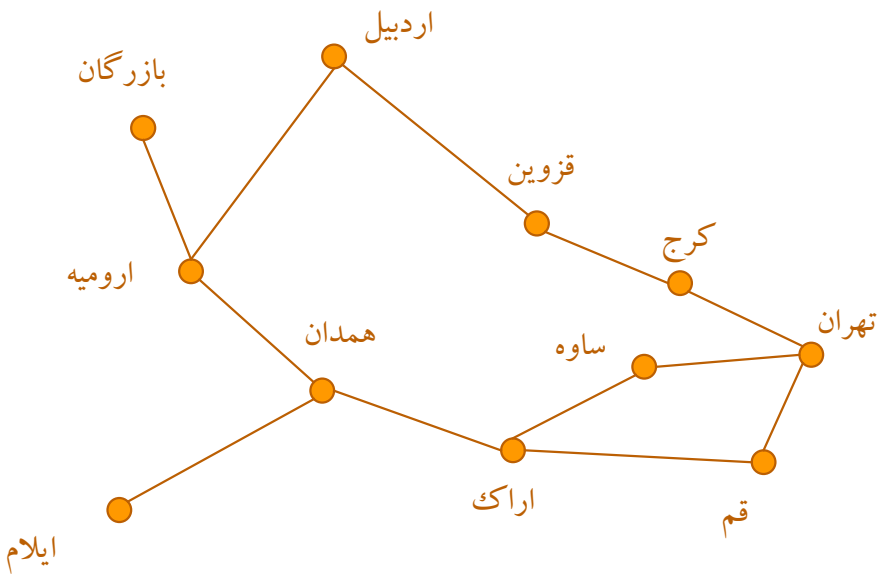
یادآوری

- مثال جهانگرد
- تدوین هدف
- تدوین مسئله
- شرایط محیط برای یک عامل مسئله حل کن:
  - مشاهده پذیر، قطعی، شناخته شده
- تدوین مسئله
  - حالت اولیه، مجموعهٔ اعمال ممکن، مدل انتقال، هدف، هزینهٔ مسیر
- چند مثال:
  - دنیای جارو، جورچین ۸، Knuth 4، مسیریابی، گردشگری، فروشندهٔ دوره گرد
- جستجو برای حل

جستجو برای حل

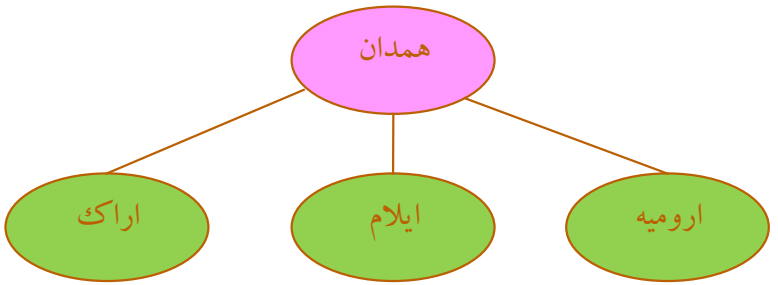
- پس از تدوین مسئله باید آن را حل نمود.
- یک حل دنباله ای است از اعمال
- الگوریتمهای جستجو، با در نظر گرفتن دنباله های اعمال متفاوت کار می کنند.
- دنباله های عمل ممکن با شروع از حالت اولیه یک درخت جستجو می سازند.
- حالت اولیه در ریشه
- شاخه ها متناظر با اعمال ممکن

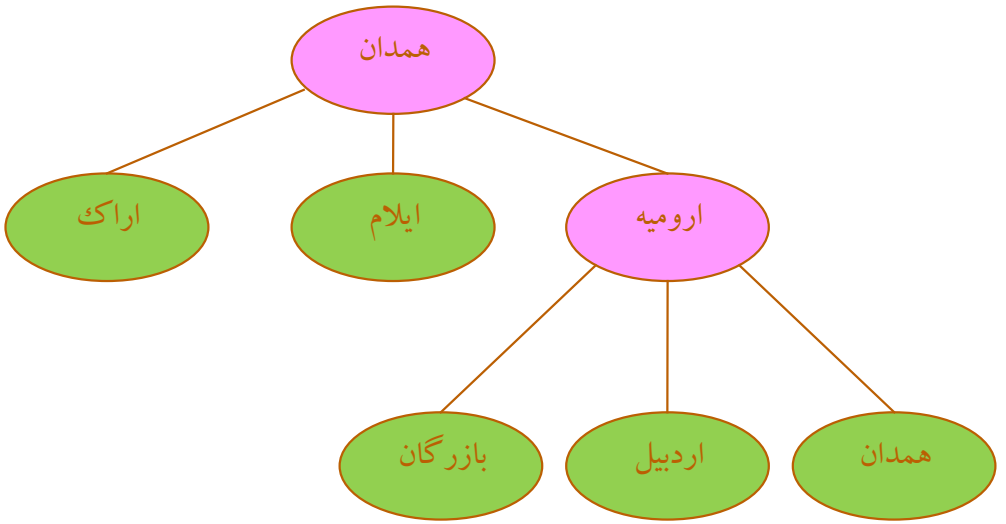
جستجو برای حل



- جستجو در فضای حالت

5

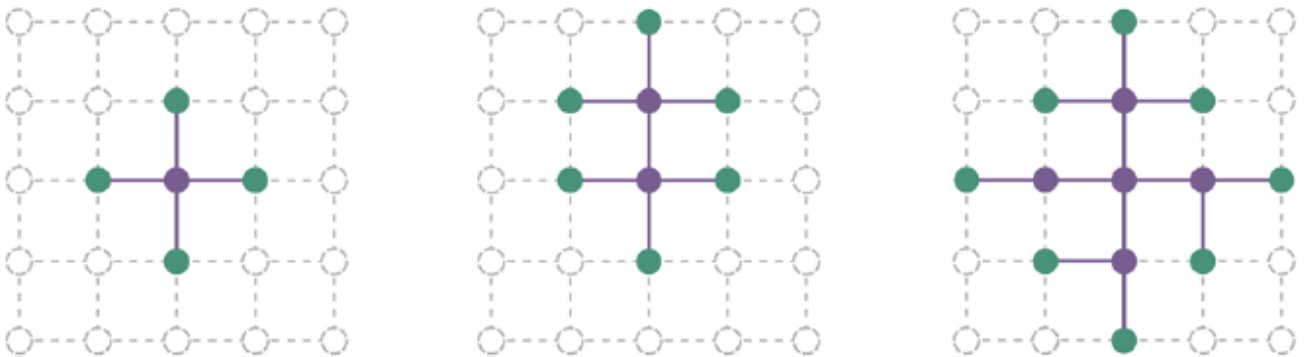

همدان
اراک
ایلام
ارومیه


همدان
اراک
ایلام
ارومیه
بازرگان
اردبیل
همدان

- مجموعهٔ همهٔ رئوس در دسترس برای بسط داده شدن در هر زمان مجموعهٔ پیشگام (frontier) نامیده می شود.
- گاهی به آن لیست باز (open list) نیز گفته می شود.
- حالتی که برای آن رأسی ایجاد شده، گفته می شود که به آن رسیده ایم (reached) (ممکن است هنوز بسط داده نشده باشد).

- مجموعهٔ پیشگام، فضای حالت را به دو ناحیه تقسیم می کند: داخلی و خارجی

Figure 3.6



The separation property of graph search, illustrated on a rectangular-grid problem. The frontier (green) separates the interior (lavender) from the exterior (faint dashed). The frontier is the set of nodes (and corresponding states) that have been reached but not yet expanded; the interior is the set of nodes (and corresponding states) that have been expanded; and the exterior is the set of states that have not been reached. In (a), just the root has been expanded. In (b), the top frontier node is expanded. In (c), the remaining successors of the root are expanded in clockwise order.

جستجوی بهترین نخست

- یک روش عمومی برای انتخاب رأسی که باید بسط داده شود، استفاده از جستجوی بهترین نخست است.
- انتخاب رأس n که کمترین مقدار یک تابع ارزیابی مثل $f(n)$ را داراست.

جستجوی بهترین نخست

Figure 3.7

```
function BEST-FIRST-SEARCH(problem, f) returns a solution node or failure
  node ← NODE(STATE=problem.INITIAL)
  frontier ← a priority queue ordered by f, with node as an element
  reached ← a lookup table, with one entry with key problem.INITIAL and value node
  while not IS-EMPTY(frontier) do
    node ← POP(frontier)
    if problem.IS-GOAL(node.STATE) then return node
    for each child in EXPAND(problem, node) do
      s ← child.STATE
      if s is not in reached or child.PATH-COST < reached[s].PATH-COST then
        reached[s] ← child
        add child to frontier
  return failure
```

```
function EXPAND(problem, node) yields nodes
  s ← node.STATE
  for each action in problem.ACTIONS(s) do
    s' ← problem.RESULT(s, action)
    cost ← node.PATH-COST + problem.ACTION-COST(s, action, s')
    yield NODE(STATE=s', PARENT=node, ACTION=action, PATH-COST=cost)
```

الگوریتمهای جستجوی درختی

```
function TREE-SEARCH(problem) returns a solution, or failure
    initialize the frontier using the initial state of problem
    loop do
        if the frontier is empty then return failure
        choose a leaf node and remove it from the frontier
        if the node contains a goal state then return the corresponding solution
        expand the chosen node, adding the resulting nodes to the frontier
```

جستجو برای حل

- اگر به درخت جستجوی مثال دقت شود، رأس همدان دوبار دیده می شود.
- به چنین رئوسی، رئوس تکراری گفته می شود.
- رئوس تکراری باعث ایجاد مسیر حلقوی می شوند.
- وجود چنین رئوسی باعث می شود که درخت بی نهایت بزرگ شود.
- ولی فضای حالت محدود است.
- مسیرهای حلقوی حالت خاص مسیرهای زائد هستند.
- مسیر زائد هنگامی وجود دارد که بیش از یک مسیر بین دو حالت وجود دارد.

جستجو برای حل

- گاهی می توان مسئله را به گونه ای تدوین کرد که دارای تکرار نباشد.
- بطور مثال در مسئلهٔ ۸ وزیر، اگر هر وزیر را بتوان در هر ستونی از صفحهٔ شطرنج گذاشت، در این حالت هر وضعیت قرار گیری n وزیر در صفحه دارای $n!$ مسیر مختلف خواهد بود.
- ولی اگر وزیر را فقط بتوان در چپترین ستون خالی قرار داد فقط یک مسیر وجود دارد.
- در برخی از مسائل که اعمال برگشت پذیر هستند، همانند مثال همدان، حالتهای تکراری اجتناب ناپذیر هستند.

- مثالی است که "**جامعه ای که تاریخ خود را فراموش کند، محکوم به تکرار آن است.**"
- برای اجتناب از مسیرهای زائد، لازم است مکانهائی که بوده ایم را به خاطر بسپاریم.
- استفاده از ساختمان داده ای به نام **مجموعهٔ اکتشاف شده** (explored set)
- یا لیست بسته
- الگوریتمی که از این مجموعه استفاده می کند جستجوی گرافی نامیده می شود.
- در الگوریتم بهترین نخست از ساختمان دادهٔ reached استفاده شده

function GRAPH-SEARCH(*problem*) **returns** a solution, or failure

initialize the frontier using the initial state of *problem*

initialize the explored set to be empty

loop do

if the frontier is empty **then return** failure

choose a leaf node and remove it from the frontier

if the node contains a goal state **then return** the corresponding solution

add the node to the explored set

expand the chosen node, adding the resulting nodes to the frontier

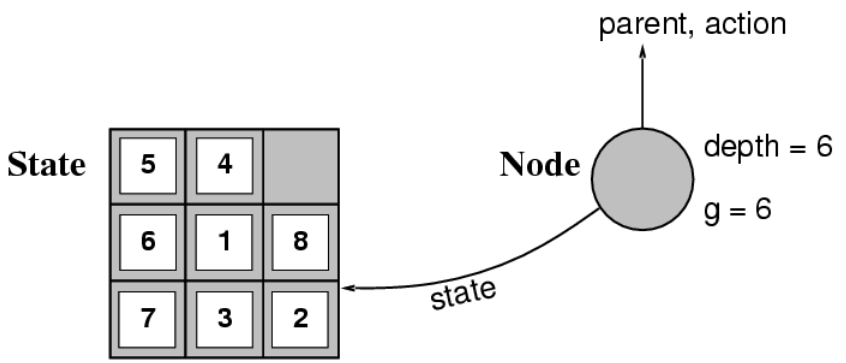
only if not in the frontier or explored set

ساختمانهای داده برای جستجو

- برای هر رأس درخت به ساختمان داده ای نیاز داریم که شامل ۴ جزء باشد:
- حالتی که رأس در فضای حالت به آن منتاظر است،
- رأس پدری که این رأس از آن ایجاد شده،
- عملی به رأس پدر اعمال شده تا این رأس بوجود آید، و
- هزینه مسیر از ریشه به این رأس
- می توان یک کلاس برای هر رأس تعریف کرد (مثلاً بنام node) و اجزاء فوق را در آن قرار داد.

رأس در مقابل حالت

- حالت یک (نمایش) پیکربندی فیزیکی (واقعی) می باشد.
- یک رأس ساختمان داده ای است که بخشی از درخت جستجو است و دارای میدانهای مختلفی است.



- رئوس ایجاد شده را می توان در یک لیست (صف) قرار داد.
- صف FIFO
- صف FILO (پشته)
- صف اولویت دار

کارآئی استراتژیهای جستجو

- استراتژی جستجو ترتیب بسط دادن رئوس را مشخص می نماید.
- استراتژیها بر اساس معیارهای زیر ارزیابی می شوند:
  - کامل بودن
    - آیا الگوریتم ضمنانت می دهد که اگر راه حلی وجود دارد، حلی بیابد و اگر نه بدرستی اعلام شکست کند؟
  - بهینه بودن هزینه
    - آیا الگوریتم حل با کمترین هزینه را می یابد؟
  - پیچیدگی فضا
    - میزان حافظه مصرف شده
  - پیچیدگی زمان
    - زمان صرف شده برای یافتن حل، قابل محاسبه بر حسب واحد زمان، تعداد حالات و اعمال

استراتژیهای جستجو

- انواع جستجو:
  - ناآگاهانه
  - آگاهانه

- برای محاسبهٔ پیچیدگی زمان و فضا از پارامترهای زیر استفاده می شود:
- b ضریب انشعاب (حداکثر مقدار آن)
- d عمق کم هزینه ترین حل
- m حداکثر عمق فضای حالت

خلاصه

- جستجو برای حل
  - ایجاد درخت، مجموعهٔ پیشگام
  - جستجوی بهترین نخست
  - جستجوی درختی
  - جستجوی گرافی
- ساختمان داده برای جستجو
- معیارهای ارزیابی استراتژیهای جستجو
  - کامل بودن، بهینه بودن، پیچیدگی فضا، پیچیدگی زمان
- جستجوی ناآگاهانه و آگاهانه


پایان