Subject:
Date

هوش مصنوعی تمرین اول

متین باقری ۴۰۲۱۰۵۷۲۷

۱) آ) نادرست ، در محیط partialy observable ، agent بدون داشتن آگاهی کامل از محیط شروع به کار می کند.

ب) نادرست ، به سنسور نیاز دارد تا بررسی کند آیا به هدف خود نزدیک شده است یا خیر.

ج) صحیح ، صرفاً بر اساس شرایط موجود، واکنش مربوط را انجام می دهد.

د) نادرست ، انتخاب حالت بعدی به صورت رندوم است و درصورتی که بهتر از حالت فعلی نبود، انجام دادن آن به صورت تصادفی است ، اگر بهتر بود، آن را انجام می دهیم.

۲) بخش اول

DFS: S, B, E, F, G

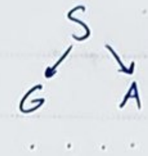
BFS: S, C, G

UCS: S, B, E, G cost: 11

A^*: S, B, E, G

بخش دوم:

DFS, BFS: S, G

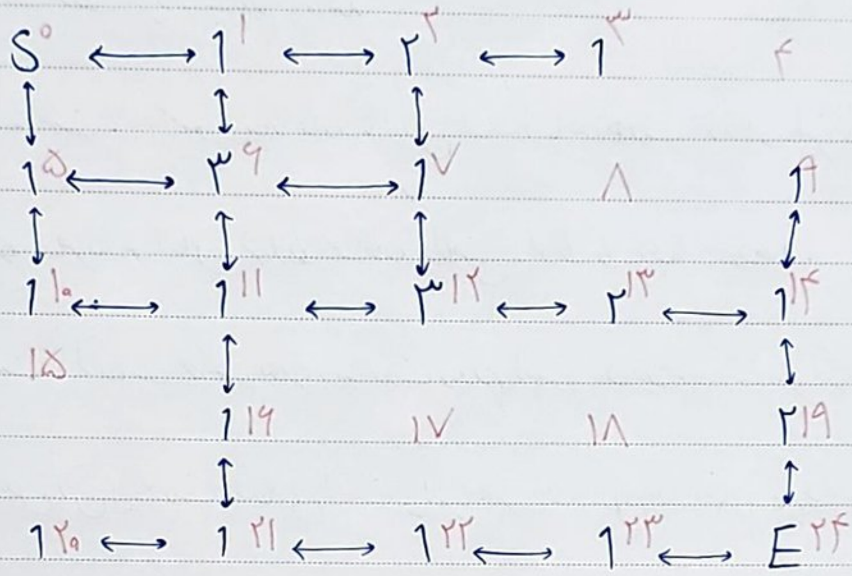


UCS: S, A, C, G   cost: 4

A*: S, A, C, G   cost: ۴

بهینه ترین مسیر را خروجی می دهد.

۳) الف)

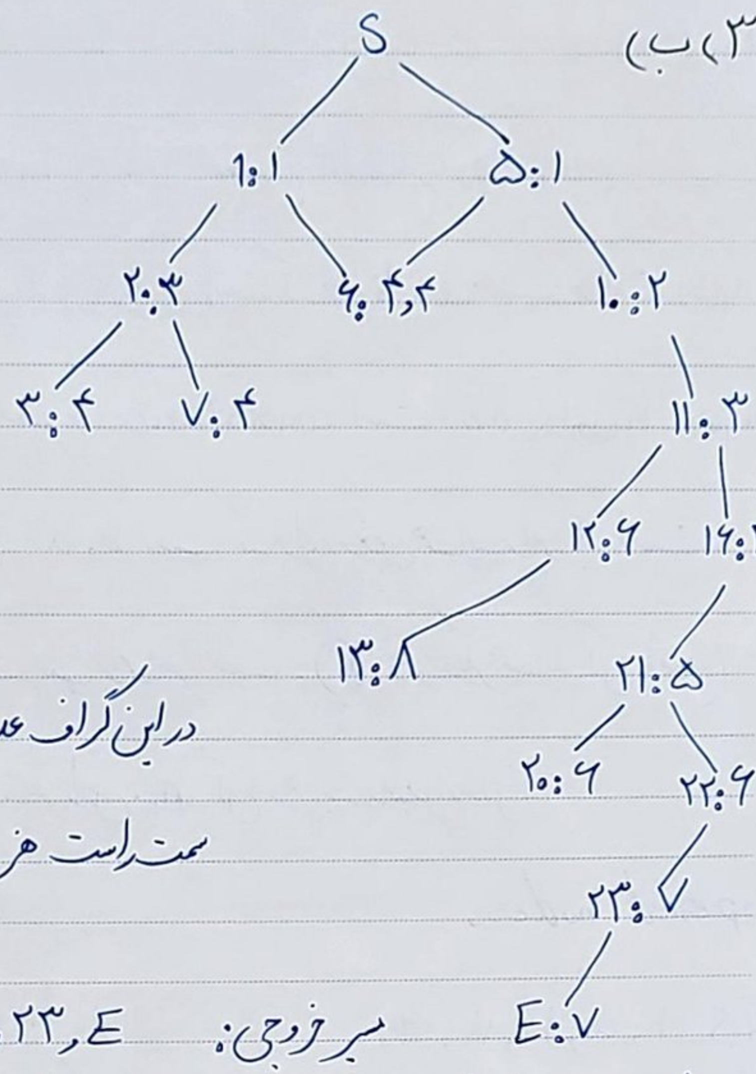


S ریشهٔ گراف است، هزینهٔ هر راس در آن نوشته شده. یال ها دوطرفه اند. (بدون جهت)

راس ها را با این رنگ شماره گذاری می کنیم.

Subject:
Date

۳) ب)



visited nodes = {

S, 1, ۵, ۱۰, ۲,

۱۱, ۶, ۳, ۷, ۱۶,

۲۱, ۱۲, ۲۰, ۲۲, ۲۳, E }

گره‌های باز شده به ترتیب:

در این گراف عدد سمت چپ شماره‌ی گره و عدد سمت راست هزینه‌ی رسیدن به آن است.

مسیر خروجی: S, ۵, ۱۰, ۱۱, ۱۶, ۲۱, ۲۲, ۲۳, E

هزینه‌ی هر گره درصورتی که کمتر از هزینه‌ی قبلی آن باشد، جایگزین آن می‌شود.

ج) در حالت کلی فاصله‌ی منهتن می‌تواند جواب باشد، چرا که هزینه‌ی هر راس حداقل ۱ است، اما در اینجا هزینه‌ی راس s و e صفر است که باعث می‌شود $h^*(s)=۷$ اما فاصله‌ی منهتن s تا E ۸ است. پس h را به این صورت تعریف می‌کنیم:

ادامهٔ ۳) ج) $h_1(x) =$ فاصلهٔ منهتن تا E − 1 $(x=S)$

$h_1(x)$ admissible است چراکه $h_1(x) \geq 0$ و همچنین همواره کمتر از هزینهٔ واقعی است

چون هزینهٔ جابه‌جایی بین هر دو راس ۱ یا ۲ یا ۳ است که حداقل 1 هست. حالت $h_1(S)$

نیز در توضیحات قبلی بررسی شد. همچنین consistent است $|h_1(a) - h_1(b)| \leq \overline{ab}$

سمت چپ معادله برابر فاصلهٔ منهتن a و b است. که برای رئوس همسایه برابر یک است.

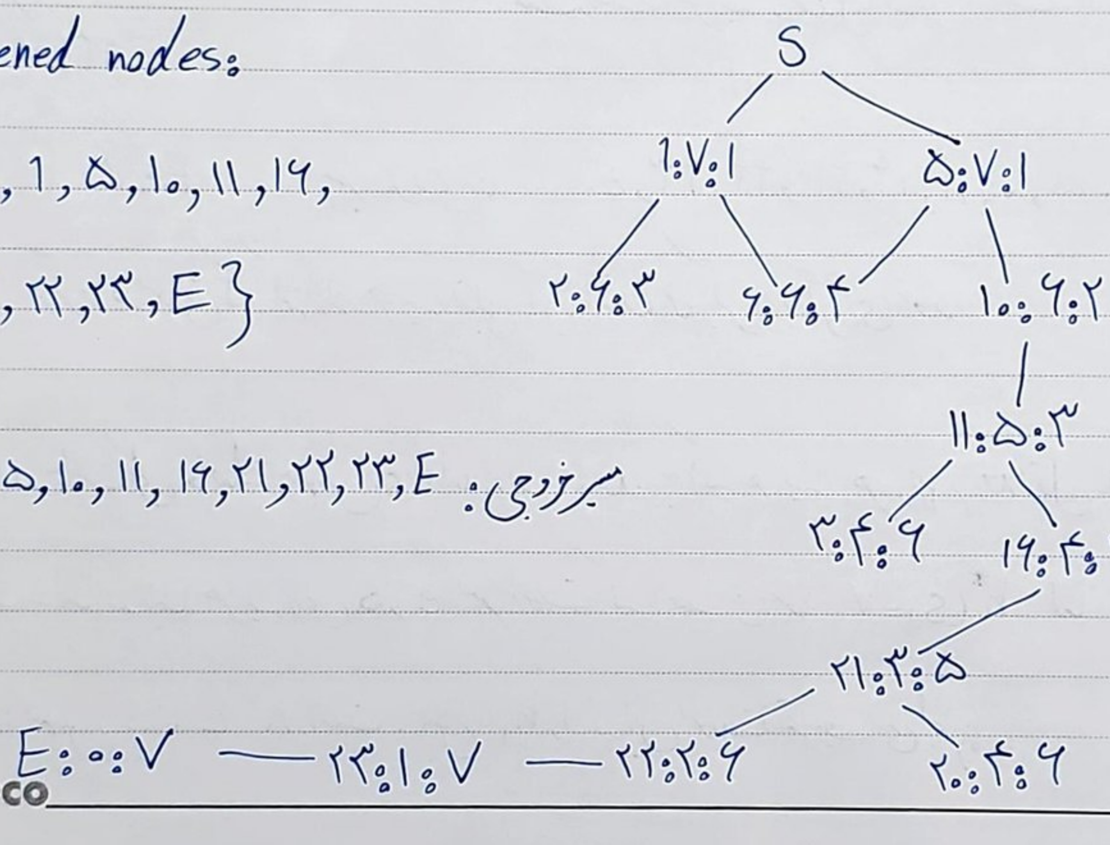
اگر یکی از a و b، S باشند، سمت چپ برابر صفر است. (یکی S و دیگر همسایهٔ آن باشند)

د) به ترتیب از چپ به راست: شمارهٔ هر راس، h_1 آن، هزینهٔ رسیدن به آن

opened nodes:

{S, ۱, ۵, ۱۰, ۱۱, ۱۶, ۲۱, ۲۲, ۲۳, E}

مسیر خروجی: S, ۵, ۱۰, ۱۱, ۱۶, ۲۱, ۲۲, ۲۳, E



Subject:
Date

۳) هـ) در UCS ۱۶ گره باز شد و در h_1 ۱۰ گره باز شد، که نشان می‌دهد الگوریتم A^* عملکرد بهتری داشته. علت آن این است که می‌توان UCS را یک A^* با $h_0(x) = 0$ درنظر گرفت. داریم $h_0 \leqslant h_1 \leqslant h^*$. پس h_1 به هزینه واقعی نزدیک‌تر است و واقع‌بینانه‌تر است. به همین علت عملکرد بهتری دارد.

و) دیگر admissible نخواهد بود چرا که فاصله و هزینه‌ی واقعی دو خانه با هزینه ۱ که در یک گوشه مشترک اند برابر یک خواهد بود اما h_1 آن ۲ است. تابع پیشنهادی جدید:

کمینه بین فاصله عرضی و فاصله طولی تا مقصد. همچنین: $h(S) = h(1) = h(5) = h(6) = 3$

$$h_2(x) = \begin{cases} 3 & ; x = S \\ \min(\text{فاصله عمودی}, \text{فاصله افقی}) & ; o.w \end{cases}$$

PAPCO

۴) آ) left most path: ۰,۰ → ۱,۰ → ۱,۱ → ۱,۲ → ۰,۲ → ۰,۳ → ۰,۴

required actions: turn right, fast, slow, turn left, fast, maintain, slow, turn left, fast, slow, turn right, fast, maintain, slow, turn left.

ب) ۴ جهت دارد و $V_{max}+1$ سرعت (سرعت صفر را هم در نظر می‌گیریم. پس

اندازهٔ فضای حالات: $M \times N \times 4 \times (V_{max}+1)$

پ) if $v=0$, then: turn right, turn left, fast → $b=3$

else: fast, maintain, slow $(v \neq V_{max})$ → $b=3$

اگر فرض کنیم در حالتی که سرعت صفر است، می‌توان maintain کرد، $b_{max}=4$

مثال: در خانهٔ (۰,۳) هستیم و سرعت صفر است، اعمال: fast, right, left

و به سمت شمال هستیم

Subject:
Date

با فرض اینکه $V_{max} > 1$

۴) ت) در حالت کلی admissible نیست. چراکه در این مسأله می توان سرعتی بیش از یک داشت و در یک واحد زمانی بیش از یک فاصلهٔ منهتن را طی کرد. مثال: فرض کنید عامل در خانهٔ (۰و۵) و به سمت شرق باشد. حالت خروج (۰و۶) و به سمت شرق باشد. فاصلهٔ منهتن در این حالت ۶ است اما می توان با ۵ دستور زیر به مقصد رسید: fast, fast, maintain, slow, slow

البته اگر فقط هزارتوی داده شده برای قسمت الف را در نظر بگیریم، این هیستوریک مجاز است.

ث) بدیهتاً این هیستوریک مجاز است، چراکه برای رسیدن به هدف قطعاً باید رو به آن قرار بگیریم.

ج) مجاز است، در قسمت ت چیزی که باعث می شد مجاز نباشد این بود که می شد در یک واحد زمانی بیش از یک خانه حرکت کرد. با تقسیم کردن فاصلهٔ منهتن بر V_{max}، این موضوع جبران شده و مقدار هیستوریک بیش از هزینهٔ واقعی نخواهد بود.

Subject:
Date

۵) الف) نادرست، در این الگوریتم‌ها پیکربندی کامل ذخیره نمی‌شود و حافظه کمتری مصرف می‌شود.

ب) درست، در این الگوریتم‌ها از وضعیتی به وضعیت دیگری رویم و هر یک می‌تواند جواب احتمالی مسأله مانند.

ج) درست، چون فضا متناهی است و فرض شده زمان کافی داریم.

د) درست، از آنجا که این الگوریتم احتمالی است (و زمان هم متناهی است) نمی‌توان فرار از بهینه محلی را تضمین کرد.

ه) درست، در صورت بهتر نبودن وضعیت بعدی، احتمال رفتن به آن $e^{\Delta E/T}$ است
و وقتی $\infty \leftarrow T$ آنگاه در هر صورت به وضعیت بعدی که به صورت رندوم انتخاب شده بود می‌رویم.

و) نادرست، در local beam search از هر K فرزند مجدداً K وضعیت بررسی می‌شود و این کار ادامه پیدا می‌کند و با K بار اجرای hill-climbing متفاوت است.

ز) درست، فرض کنیم در این مرحله K گره باز شده و هر کدام نهایتاً b فرزند دارند، بنابراین در مرحله بعد نهایتاً bK گره بررسی می‌شود که $O(bK)$ است.

۶) آ) $(X_0)_{10} = 205$, $f(X_0) = 205 \times (1+4+8+64+128) = 205^2 = 42025$

$f(X_1) = 105^2 = 11025$, $f(X_2) = 182^2 = 33124$, $f(X_3) = 87^2 = 7569$

ب) خروجی این تابع مربع ورودی آن است، اگر هر کروموزوم را به عنوان نمایش باینری یک عدد در نظر بگیریم. این تابع برای ورودی نامنفی، اکیداً صعودی است و به ازای ورودی $255 = (1111\ 1111)_2$ بیشینه می‌شود.

ج) متناسب با مقادیر محاسبه شده در قسمت الف، بازه‌ی بین ۰ و ۱ را به کروموزوم‌ها اختصاص می‌دهیم و پس از هر selection به goal (مقدار بهینه) نزدیک‌تر می‌شویم. مقادیر بازه‌ها نوشته شده تا دقت ۳ رقم اعشار است.

$[0, 0.448] \rightarrow X_0$, $(0.448, 0.566] \rightarrow X_1$

$(0.566, 0.919] \rightarrow X_2$, $(0.919, 1] \rightarrow X_3$

$0.748 \rightarrow \underline{X_2}$, $0.362 \rightarrow \underline{X_0}$, $0.915 \rightarrow \underline{X_2}$, $0.531 \rightarrow \underline{X_1}$

د) با ترکیب دو کروموزوم و crossover کردن، state های بهتری خواهیم داشت. می‌توانیم ویژگی‌های خوب را حفظ و باهم ترکیب کنیم.

Subject:
Date

ادامه ۶) د) برای حریصانه، رسیدن به بیشترین عدد ممکن، باید بین بیت ۶ و ۷ را برش بزنیم و دو بیت سمت چپ را عوض کنیم. در این حالت حریصانه و ترکیب اولی با دومی و سپس سومی با چهارمی خواهیم داشت:

$X_0' = 01001101$, $X_1' = 11101001$

$X_2' = 01110110$, $X_3' = 10\ 010111$

و اگر از وسط نصف کنیم:

$X_0' = 1100\ 1001$, $X_1' = 0110\ 1101$

$X_2' = 1011\ 0111$, $X_3' = 0101\ 0110$

هـ) تا اینجا، ممکن است به local goal رسیده و در آن گیر بیفتیم بنابراین با انجام mutation از آن جلوگیری می‌کنیم تا با احتمال بیشتری به هدف اصلی برسیم. کروموزوم ها جدید:

$X_0 = 1100\ 1000$, $X_1 = 0110\ 1111$, $X_2 = 1011\ 0011$, $X_3 = 0101\ 1110$

جمع فیتنس ها بعد از mutation برابر است با $200^2 + 111^2 + 179^2 + 94^2 = 93198$

و قبل از آن برابر بود با $201^2 + 109^2 + 183^2 + 86^2 = 93167$

بنابراین مقدار فیتنس کل کمی بیشتر شده و به مقدار بهینه نزدیک‌تر شدیم.

Subject:
Date

۶) و) خیر، چون ما در crossover از وسط نصف می‌کنیم و مقدار بهینه رشته ای از 1 ها است و ما در کروموزوم های اولیه ۱۱۱۱ نداریم، بدون matation نمی‌توان به مقدار بهینه رسید.