هوش مصنوعی **تمرین شماره‌ی ۱**

مدرسین: **دکتر فدایی و دکتر یعقوب‌زاده**

طراحان: پاشا براهیمی، کسری حاجی‌حیدری، محمدصادق ابوفاضلی

مهلت تحویل: جمعه ۱۸ اسفند ۱۴۰۲، ساعت ۲۳:۵۹

بخش کتبی

Agents

جدول زیر را پر کنید.

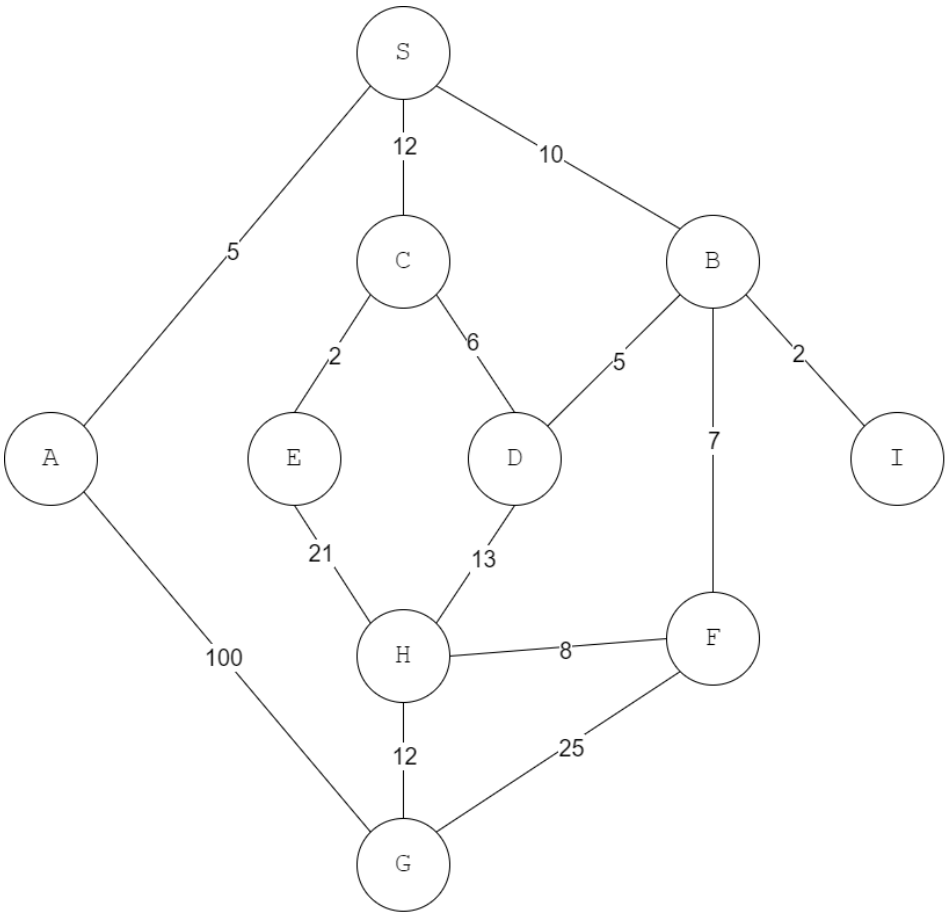
Environment	CS-GO	Sudoku
Fully/Partially Observable	Partially	fully
Deterministic/Stochastic	stochastic	Deterministic
Episodic/Sequential	sequential	sequential
Static/Dynamic/Semi-Dynamic	Dynamic	Static
Discrete/Continuous	Continous	Discrete
Single-agent/Multi-agent	Multi agent	Single agent

بازی CS-GO (Counter-Strike) یک بازی شوتر اول شخص است که در آن تعدادی عامل (بازیکن) در یک map قرار گرفته و به بازی می‌پردازند.

Search

سوال اول

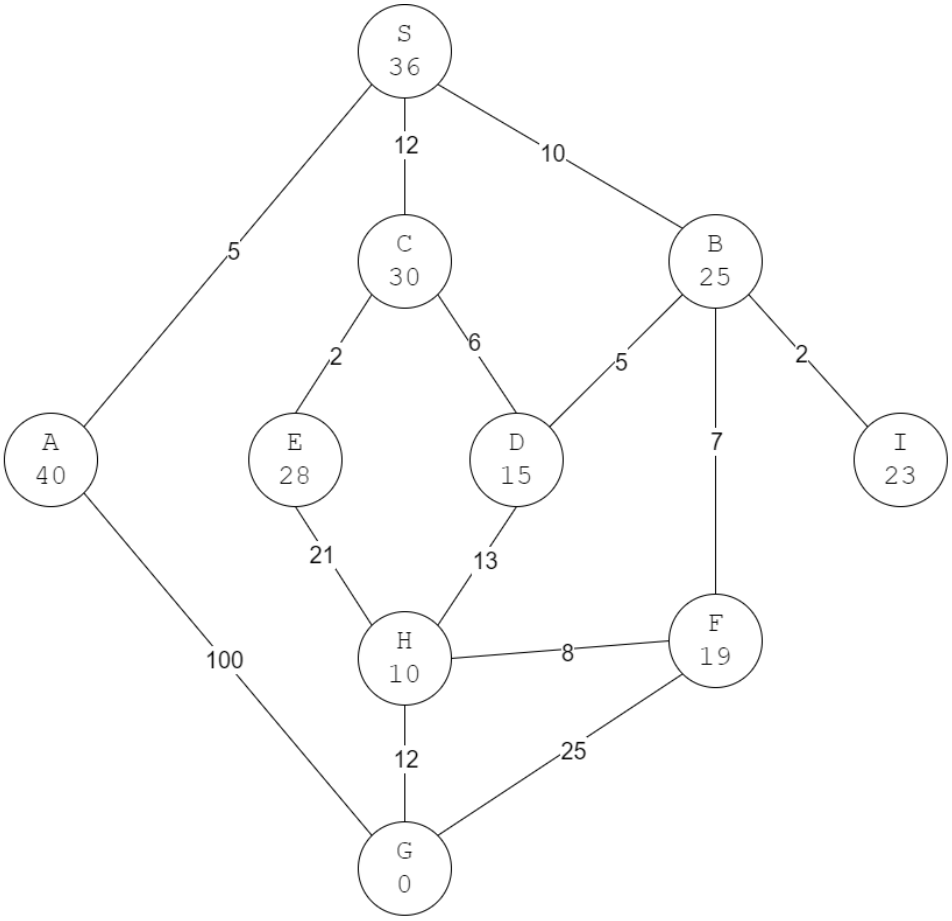
الف) گراف زیر را در نظر بگیرید. آیا استفاده از الگوریتم های bfs و dfs برای این گراف، جواب بهینه را به ما می‌دهد؟ توضیح دهید در چه صورتی استفاده از این دو الگوریتم پیشنهاد می‌شود.
ب) حداقل هزینه برای رسیدن از راس S به G را با استفاده از الگوریتم Uniform Cost Search محاسبه کنید. به ازای تمام state های دیده شده، راسی که روی آن قرار دارید، مسیر طی شده، هزینه صرف شده و مجموعه‌های Explored و Frontier را به صورت یک جدول بنویسید.
اگر در هر مرحله چند انتخاب داشتید، راسی که از لحاظ ترتیب الفبایی کوچکتر است را انتخاب کنید.



سوال دوم

الف) در این مرحله به هر راس یک مقدار هیوریستیک نسبت داده شده است. الگوریتم A* را روی گراف اجرا کنید و حداقل هزینه برای رسیدن از راس S به G را به دست آورید. به ازای تمام state های دیده شده، راسی که روی آن قرار دارید، مسیر طی شده، هزینه صرف شده، مجموع هیوریستیک و هزینه صرف شده و مجموعه‌های Explored و Frontier را به صورت یک جدول بنویسید.

هر state را تنها یک بار بررسی کنید. یعنی در صورت وجود راسی در مجموعه Explored، دوباره آن راس را بررسی نکنید (حتی اگر مسیر کوتاه‌تری برای آن یافت شده باشد). اگر در هر مرحله چند انتخاب داشتید، راسی که از لحاظ ترتیب الفبایی کوچکتر است را انتخاب کنید.
ب) هیوریستیک را از لحاظ Admissible و Consistent بودن بررسی کنید.



سوال سوم

الف) در چه زمانی رویکرد local search نسبت به سایر روش های جستجو برتری دارد؟
ب) یکی از الگوریتم های local search، الگوریتم hill climbing می‌باشد. یکی از مشکلات اصلی این الگوریتم احتمال پیدا کردن local maximum به عنوان جواب نهایی می‌باشد. دو راه حل برای این مشکل ارائه دهید.

بخش عملی

مقدمه

در این قسمت، با روش‌هایی که برگرفته از طبیعت و انتخاب طبیعی هستند، آشنا می‌شویم. در این روش‌ها که به طور کلی الگوریتم‌های ژنتیک نامیده می‌شوند، ایده‌هایی برای مدل‌سازی جفت‌گیری، جهش و انتخاب طبیعی به کار گرفته می‌شود. در این گونه الگوریتم‌ها، ممکن است با انتخاب معیارهای ساده‌ی انتخاب طبیعی، نتایج مطلوب به دست نیاید و باید معیاری در نظر بگیریم که علاوه بر عملکرد فردی، به گوناگونی جمعیت نیز اهمیت دهد.
الگوریتم‌های ژنتیک عموماً در مسئله‌هایی با فضای حالت بزرگ کاربرد دارند؛ این الگوریتم‌ها نمونه گرفتن از جمعیت و ترکیب و تغییر افراد و ارزیابی آن‌ها انجام می‌دهند و سعی می‌کنند که نسل به نسل جواب‌ها را بهبود دهند تا به جواب مورد نظر برسند.
در این پروژه قصد داریم با استفاده از الگوریتم‌های ژنتیک، یکی از مسائل مربوط به بهینه‌سازی را پیاده‌سازی کنیم. هدف یک مسئله بهینه‌سازی، یافتن بهترین راه‌حل از مجموعه بزرگی از راه‌حل‌های ممکن است، جایی که «بهترین» راه‌حل با مجموعه‌ای از معیارها یا اهداف تعریف می‌شود. حل مسائل بهینه‌سازی می‌تواند به مزایای قابل توجهی مانند افزایش کارایی، کاهش هزینه‌ها، بهبود عملکرد و موارد دیگر منجر شود.

توضیح مسئله

در این قسمت شما به مسئله Knapsack Problem می‌پردازید. در صورتی که درس طراحی الگوریتم را گذرانده باشید، احتمالا با این مسئله آشنا هستید. در این تمرین، لازم است این مسئله را با کمی تغییر پیاده‌سازی کنید.
فرض کنید یک کوله‌پشتی در اختیار دارید و با همراه دوستان خود، قصد رفتن به یک پیک‌نیک دارید. تعدادی خوراکی و تنقلات دارید که قصد دارید برخی از آن‌ها را به پیک‌نیک ببرید. برای هر خوراکی یک ارزش در نظر گرفته شده و همچنین از هر خوراکی وزن خاصی در اختیار داریم که می‌توانیم هر مقداری از آن را برای به همراه بردن انتخاب کنیم. بدیهی‌ست که اگر برای مثال نیمی از یک خوراکی را انتخاب کنیم، ارزشش هم نصف خواهد شد. لازم به ذکر است که در انتخاب خود باید چند محدودیت را لحاظ کنیم:

- ارزش مجموع خوراکی‌هایی که به همراه می‌بریم باید از یک حد خاصی که در ورودی به شما داده می‌شود، بیشتر باشد.
- وزن مجموع خوراکی‌هایی که به همراه می‌بریم باید از حد خاصی که در ورودی داده می‌شود کمتر باشد.
- تعداد انواع (تنوع) خوراکی‌هایی که انتخاب می‌کنیم باید در بازه خاصی قرار داشته باشد که در ورودی به شما داده می‌شود.
- وزن انتخاب شده از هر خوراکی نباید از وزن موجود از آن خوراکی بیشتر باشد.

خوراکی‌ها در ورودی به صورت فایل CSV به شما داده می‌شود که نمونه آن در ادامه قرار داده شده است:

نمونه ورودی خوراکی‌ها به صورت CSV

```
Snack,Available Weight,Value
MazMaz,10,10
Doogh-e-Abali,15,10
Nani,5,5
Jooj,7,15
Hot-Dog,20,15
```

حال مسئله را به ازای ورودی‌های زیر در نظر بگیرید:

- حداکثر وزنی که می‌توانیم انتخاب کنیم: ۱۰ واحد
- حداقل ارزشی که نیاز است انتخاب کنیم: ۱۲ واحد
- بازه تعداد انواع خوراکی‌هایی که انتخاب می‌کنیم: از ۲ تا ۴ عدد

برای این مسئله می‌توانیم پاسخ زیر را در نظر بگیریم:

Snack	Weight	Value
Doogh-e-Abali	3	2
Jooj	3.5	7.5
Nani	3	3
Total	9.5	12.5

در این حالت، خروجی برنامه به صورت زیر خواهد بود:

نمونه خروجی

```
Doogh-e-Abali: 3
Jooj: 3.5
Nani: 3
Total Weight: 9.5
Total Value: 12.5
```

پیاده‌سازی مسئله

بخش یک: مشخص‌کردن مفاهیم اولیه

در الگوریتم‌های ژنتیک ابتدا باید یک تعریف برای ژن ارائه دهید و سپس با استفاده از آن، یک کروموزوم بسازید. هر کروموزوم مجموعه‌ای از ژن‌ها است و این مجموعه یا همان کروموزوم، یک راه پیشنهادی برای حل مسئله مورد نظر می‌باشد.
توجه داشته باشید که در الگوریتم‌های ژنتیک باید اکثر کارها را با استفاده از تصادفی کردن وقایع انجام دهید، چرا که اگر فضای حالت بزرگ باشد پیدا کردن شرطی که همه‌ی محدودیت‌ها را برقرار سازد بسیار دشوار است. به همین دلیل، تعریف کروموزوم‌ها اهمیت ویژه‌ای دارد و باید به گونه‌ای باشد که امکان اعمال تابع تناسب و توابع دیگر بر روی آن فراهم باشد.

بخش دو: تولید جمعیت اولیه

پس از تعریف و پیاده‌سازی کروموزوم‌ها، باید جمعیت اولیه‌ای از کروموزوم‌ها به صورت تصادفی بسازید. تعداد این جمعیت می‌تواند به عنوان یک پارامتر حل مسئله باشد و به انتخاب‌های شما بستگی دارد.

بخش سه: پیاده‌سازی و مشخص‌کردن تابع معیار سازگاری

بعد از تولید جمعیت اولیه، نیاز داریم تا تابع معیاری تعریف کنیم که بتواند برای شناسایی کروموزوم‌های برتر که شرایط و محدودیت‌های مسئله را بهتر مدل می‌کنند استفاده شود. ابتدا یک تعریف مناسب برای این تابع معیار ارائه دهید، و سپس آن را برای این مسئله پیاده‌سازی کرده، و میزان سازگاری جمعیت خود را بدست آورید.

بخش چهار: پیاده‌سازی crossover و mutation و تولید نسل بعدی

حال برای اینکه به یک پاسخ از مسئله داده شده نزدیک شویم، نیاز است در هر نسل، جمعیت جدیدی با استفاده از جمعیت نسل قبل آن تولید گردد. برای این کار، باید از روش‌های crossover و mutation استفاده گردد.
تابع crossover بر روی دو کروموزوم اعمال می‌شود، و آن‌ها را ترکیب می‌کند تا به کروموزوم‌هایی از ترکیب آن دو که در حالت ایده‌آل بهترین ویژگی‌های دو ژن اولیه را دارند برسد. این ترکیب و نرخ ایجاد آن باید به عنوان پارامترهای مسئله در نظر گرفته شوند.
تابع mutation بر روی یک کروموزوم اعمال می‌شود، و آن را جهش و یا تغییر می‌دهد؛ به این امید که بتواند به کروموزوم بهتری جهش پیدا کند. می‌توانید درصد معقولی از ژن‌های برتر را نیز برای انتقال مستقیم به نسل‌های آینده در نظر بگیرید.

بخش پنج: ایجاد الگوریتم ژنتیک روی مسئله

پس از انجام بخش‌های بالا، باید این توابع پیاده‌سازی شده را در یک الگوریتم استفاده کنید. توجه کنید که نیاز است هایپرپارامترهایی برای میزان randomness و نحوه نزدیک شدن به پاسخ نهایی خود داشته باشید که با تغییر آن‌ها به جواب بهتری برسید.

بخش شش: ارزیابی نتایج

در نهایت می‌توانید الگوریتم را توسط تست‌کیس‌های مختلف بیازمایید و خروجی آن را با خروجی مد نظر خود مقایسه کنید تا از صحت عملکرد الگوریتم مطمئن شوید. در این مرحله ممکن است نیاز به تغییر هایپرپارامترها برای رسیدن به پاسخ بهتر نیز وجود داشته باشد.

سوالات

1. جمعیت اولیه‌ی بسیار کم یا بسیار زیاد چه مشکلاتی را به وجود می‌آورند؟
2. اگر تعداد جمعیت در هر دوره افزایش یابد، چه تاثیری روی دقت و سرعت الگوریتم می‌گذارد؟
3. تاثیر هر یک از عملیات crossover و mutation را بیان و مقایسه کنید. آیا می‌توان فقط یکی از آن‌ها را استفاده کرد؟ چرا؟
4. به نظر شما چه راهکارهایی برای سریعتر به جواب رسیدن در این مسئله‌ی خاص وجود دارد؟
5. با وجود استفاده از این روش‌ها، باز هم ممکن است که کروموزوم‌ها پس از چند مرحله دیگر تغییر نکنند. دلیل این اتفاق و مشکلاتی که به وجود می آورد را شرح دهید. برای حل آن چه پیشنهادی می‌دهید؟
6. چه راه‌حلی برای تمام شدن برنامه در صورتی که مسئله جواب نداشته باشد پیشنهاد می‌دهید؟

*** جواب این سوالات را به صورت کامل در گزارش خود بنویسید.**

نکات پایانی

- دقت کنید که کد شما باید به نحوی زده شده باشد که پارامترهای سوال مثل حداقل ارزش و حداکثر وزن در حین تحویل پروژه قابل تغییر و اجرای دوباره باشد.
- توضیحات مربوط به هر بخش از پروژه را بطور خلاصه و در عین حال مفید در گزارش خود ذکر کنید. از ابزارهای تحلیل داده مانند نمودارها استفاده کنید.
- پس از مطالعه کامل و دقیق صورت پروژه، در صورت وجود هرگونه ابهام یا سوال با طراحان پروژه در ارتباط باشید.
- نتایج، گزارش و کدهای خود را در قالب یک فایل فشرده با فرمت AI_A1_[stdNumber].zip در سامانه ایلرن بارگذاری کنید.
- محتویات پوشه باید شامل فایل پاسخ‌های شما به سوالات کتبی، فایل jupyter-notebook، خروجی html و فایل‌های مورد نیاز برای اجرای آن باشد. از نمایش درست خروجی‌های مورد نیاز در فایل html مطمئن شوید.
- هدف از تمرین، یادگیری شماست. لطفا تمرین را خودتان انجام دهید

موفق باشید